BEng Honours Thesis

Flinders University

Software Engineering, 18-unit

Three-Dimensional Dual-Energy Computed Tomography Baggage Scanner Image Segmentation Toolkit

Author:
John Dunstan

Academic Supervisor:
Mariusz Bajger, Flinders University
Industry Supervisors:
Chris Delnooz, Micro-X

14th October, 2022

Declaration

I certify that this thesis:

1. does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university
2. and the research within will not be submitted for any other future degree or diploma without the permission of Flinders University; and
3. to the best of my knowledge and belief, does not contain any material previously published or written by another person except where due reference is made in the text.

Signature of student .....  .....

Print name of student John Dunstan .....

Date 14/10/2022 .....

I certify that I have read this thesis. In my opinion it is/is not (please circle) fully adequate, in scope and in quality, as a thesis proposal for the degree Bachelors of Engineering (Software) (Honours). Furthermore, I confirm that I have provided feedback on this thesis and the student has implemented it minimally/partially/fully (please circle).

Signature of Principal Supervisor .....

Acknowledgements

I would like to acknowledge my mentor and supervisor Dr Mariusz Bajger who enthusiastically provided his great depth of knowledge to encourage and guide me throughout this honours thesis project. Furthermore, to Micro-X for providing me with this fantastic opportunity to undertake my honours with an industry partner. More specifically, Chris Delnooz for providing me substantial guidance to help bridge the gap between an academic and industry focused project.

Executive Summary

Micro-X has provided Flinders University the opportunity and resources to investigate the application of three-dimensional Dual-Energy Computed Tomography (DECT) image segmentation algorithms in the security domain. The aim of this project is create a C++ image segmentation toolkit comprised of the most appropriate algorithms for the Micro-X DECT baggage scanner. Much of the CT baggage segmentation literature discusses the success of common graph-based algorithms, more specifically minimal spanning tree (MST). However, using the material identification value at each individual voxel in the DECT images, efficient connected-component labelling (CCL) algorithms can be used for segmentation. GPU acceleration using NVIDIA's CUDA framework allow algorithms to be optimised using multi-threading, a possible necessity in the use case of an airport baggage scanner.

Two groups of implementations were included, CCL for the material values and MST for the greyscale values. Each group of algorithms used both sequential and GPU accelerated approaches, with the MST having an additional optimised sequential MST. The GPU CCL was able to be a 89% faster than the sequential counterpart with the same quality results. Contradicting the literature, the GPU MST is on average 33% and 92.8% slower than the sequential and optimised sequential MST, respectively. This, as discussed, is a result of using different implementations of the base MST algorithm.

The project is a step towards the future objective of creating a self-screening Micro-X Miniature Baggage Scanner (MBS), which would mitigate the risk from traditional error prone human operated airport security checkpoints. The open-source toolkit's algorithms can be easily integrated into the Micro-X baggage scanner software. The project also sets a solid foundation for a future Micro-X and Flinders University collaborated DECT threat detection PhD project.

Contents

List of Figures

List of Tables

1 | Introduction

This document is a formal honours thesis report for Bachelors of Engineering (Software) (Honours) at Flinders University. The project is titled *Three-Dimensional Dual-Energy Computed Tomography Baggage Scanner Image Segmentation*. Using literature resources supplied by Flinders University and Micro-X, an in-depth review on different combinations of software toolkits, DECT, CT, security and image segmentation literature will be discussed. This is followed by a detailed implementation that was used to obtain the results which are shown afterwards. Finally, the report will discuss and compare the results with findings from related literature.

1.1 Background

For many years, airport security checkpoints have established processes to conduct some sort of baggage inspection to prevent the transportation of prohibited and dangerous goods. The foremost implemented process requires a specially trained human operator, known as screeners, to visually observe scans produced by conventional two-dimensional X-ray machines. This standard process is tiresome, error prone and passenger privacy concerns are a catalyst for controversy [1].

In recent years, the aviation security domain has adapted the use of Dual-Energy X-ray Computed Tomography (CT), known as DECT. Conventional two-dimensional (2D) X-ray imaging approaches have limitations such as occlusion, clutter and density confusion [2]. The three-dimensional (3D) nondestructive DECT evaluation provides means to mitigate these limitations [3, 4]. Traditional DECT applications in the security domain are predominately comprised of material-based detection for explosives [5]. However, to cater for the increasing needs of aviation security regulations, dual-energy CT is becoming a popular research area for automated object segmentation and identification in the security industry [1, 2].

Figure 1: Micro-X Miniature Baggage Scanner (MBS) concept.

Micro-X is currently researching and developing their Miniature Baggage Scanner (MBS) [6], which is a DECT scanner, shown in Figure 1. The MBS is Micro-X's contribution to push for an automated self-screening airport checkpoint. Therefore, this goal, which requires accurate threat detection, provided the opportunity for image segmentation on 3D DECT baggage scans to be investigated. The outcome of this research provides a foundation for a future project that will involve object detection on the segmented scans.

1.2 Literature Overview

The literature suggests that threat detection is a well researched area. Threat detection uses techniques such as material identification, machine learning and segmentation [2, 7, 5]. Segmentation on baggage scans is a critical component of an automatic security system. Despite general image segmentation having lots of research completed in the computer science field [8], image segmentation for DECT baggage scans constitutes a small component of threat detection literature.

Classic graph-based segmentation techniques were developed as far back as the 1930s [9]. As far as image segmentation on baggage scans, studies have shown that it is possible to get high-quality results, such as Haralick et al. [8] from 1985 and Wiley et al. [10] from 2012, but the time computational cost is too dire for real-world applications. Automated baggage scanning would be most useful at large airports such as London Heathrow, which has a throughput of six seconds per item [5]. However, Wiley et al. was averaging 40 minutes for each CT baggage segmentation. Advances in GPU technology and taking additional measures to optimise the image segmentation, such as using superpixels [11], have proved to benefit image segmentation. The literature suggest that these approaches are possible but have not been extensively attempted with DECT baggage scanner data.

1.3 Project Scope

Current software used at Micro-X is developed using the C++ programming language. Therefore to maintain consistency and ease the process of adopting the toolkit, it will be developed using C++.

The toolkit must contain automatic image segmentation algorithms. Due to the intended use case of the Micro-X baggage scanner, semi-automatic image segmentation algorithms would not suffice because they require *a priori* information [2].

The algorithms must differentiate objects presented in the DECT reconstructions. The toolkit is not expected to classify each of the differentiated objects. The limitation is due to insufficient resources. In order to train the toolkit to not only segment the reconstructions but also classify, requires an amount of annotated reconstruction images that is estimated to be not available from Micro-X until 1st September, 2022. However, as the MBS will use dual-energy, the image segmentation techniques may utilise material identification per pixel.

The project will not focus on training-based image segmentation because of the limited data. However, if pre-trained implementations are discovered and deemed appropriate, they may be considered for the toolkit.

Removing image artifacts and streaking as a preprocessing step is not within the project scope. This is because the reconstruction software at Micro-X is currently still in development so the expected image defects is not known.

1.4 Knowledge Gap

CT image segmentation is a well researched area of interest, however, the majority being in the medical industry. Despite the increasing need for advances in threat detection, limited work has covered automatic image segmentation for 3D CT scans in the security domain [2]. Furthermore, conventional CT image segmentation doesn't utilise dual-energy information to differentiate the objects in the images [3]. With support from Micro-X, this project provides an opportunity to further investigate the appropriateness of applying image segmentation with assistance from dual-energy technology in the baggage scanning security industry.

1.5 Importance

An objective for the MBS is to integrate the system into a self-screening airport checkpoint, hence, image segmentation is a topic of interest for Micro-X. This segmentation project is the first step to implementing a dual-energy object identification solution into the MBS. The segmented 3D objects can later be classified by using learning-based techniques such as neural networks.

Also, the US Transportation Security Administration's (TSA) evident push for CT screening in airports [12] shows that there is a clear need to veer baggage segmentation from traditional 2D X-rays to use more advanced technologies, such as DECT. As the security screening technology used in the real-world advances, the technology used in studies for threat detection should as well.

The success of automated object segmentation and object detection will provide confidence on the ability to mitigate the risk from traditional error prone human operated airport security checkpoints. The success of this project is important for, not only MBS, but for future baggage self-screening systems.

1.6 Project Overview

This section will briefly cover the project and will include the problem, aim, hypothesis and experimental methodology.

1.6.1 Problem

The overall problem domain this study is intended for is to mitigate non-automated airport security systems. The project content specifically aims to solve the difficult problem of segmenting complex DECT baggage scans. It must be done effectively and efficiently due to the fast throughput and strict regulations in the application's environments, such as airports.

1.6.2 Aim

The aim of the project is to develop an image segmentation toolkit that Micro-X can embed into their MBS [6]. The toolkit must contain appropriate algorithmic implementations for the reconstructed 3D images produced by the DECT X-ray baggage scanner.

1.6.3 Hypotheses

The hypotheses for this study are that it is possible to use image automatic segmentation techniques to segment and differentiate complex and clustered objects in DECT baggage scans. Furthermore, it can be done effectively and efficiently due to added dual-energy information from the DECT images. The image segmentations will also be appropriate to use with the Micro-X MBS.

1.6.4 Experimental Methodology

There are three major components to the project; literature review, toolkit implementation then toolkit testing. The next chapter, Chapter 2, includes an in-depth literature review to investigate different combinations of software toolkits, DECT, CT, security and image segmentation literature. Required details from research include appropriate segmentation algorithms for MBS, advantages and disadvantages of the techniques and a means of statistical analysis to investigate the appropriateness for an MBS implementation. Using developed knowledge from the review, the selected segmentation approaches are implemented with Micro-X DECT data. The most outstanding segmentation approaches will be implemented into the toolkit, constituting the major deliverable for this project. The toolkit will be comprised of fully automated image segmentation algorithms that are appropriate for 3D DECT X-ray reconstructions produced by the Micro-X MBS. Using both simulated and test reconstructions of the baggage scanner from Micro-X, the implemented toolkit segmentation algorithms can be compared. The segmentation quality and time will be assessed.

2 | Literature Review

As part this thesis, an in-depth literature review of related published material is conducted. It provides readers a comprehensive and detailed overview of current research in closely related and neighbouring domains. In this literature review, the research has been divided into several main sections, baggage threat detection, DECT baggage screening, image segmentation, CT & DECT baggage segmentation and data & performance metrics. Each section builds upon each other as the literature reads. Following the review is a conclusion. This will outline the summary and critique the literature from the review and will address how the outcome of the review addresses the thesis problem and hypotheses.

2.1 Baggage Threat Detection

Security inspection plays a vital role in disabling prohibited items, such as guns and certain liquids passing through checkpoints. Growing populations demand more efficient and scalable threat detection solutions [7]. Image segmentation is a small piece of the puzzle when trying to develop an adequate threat detection system. This section is included to provide additional background information and to develop an understanding on what past and current research in the field has achieved. Techniques and tools for baggage threat detection can be divided into many different classes, which includes but is not limited to, segmentation, classification and explosive detection systems (EDSs) [2, 7, 5]. When threat detection is applied to passenger checkpoints, they should aim to meet a six second per item rate to meet with the throughout of airports, especially larger airports such as London Heathrow [5].

Object classification is common technique used for threat detection [13, 14]. Convolutional Neural Networks (CNNs) are a popular deep learning technique used to research the appropriateness of object classification as a means of threat detection. Even though deep learning has proven to be successful when dealing with natural images, X-ray images provide a number of new challenges to overcome. Common difficulties of deep learning models with X-ray data include but are not limited to object clutter and object overlapping [7].

EDSs aim to mitigate the continuing risk of explosives in the security industry. These systems can be commonly found in checkpoint systems. In 2015, the DHS released requirements that future EDSs require an increased probability of a true positive and a decreased probability of a false positive. The TSA also requires that the shape, position and orientation of the explosives should not be depended on the EDSs [3]. Though extensively researched, most EDSs can not be applied to commercial items due to not meeting real-world requirements [5].

This project focuses on using segmentation as a threat detection tool, which will be used as a stepping stone to assist classification for an airport baggage scanner. As Mouton et al. [2] discussed in his review, the success of medically applied image segmentation is difficult to extend

to baggage scans due to the need of *a priori* information, clustered objects and artefacts. Though that's not to say studies have not successfully segmented baggage scans. However, most are inapplicable to real-world scenarios due to either performance or quality issues [15, 10].

The representation of data can have a substantial impact on the quality and performance of the threat detection. Conventional 2D X-ray baggage images have many limitations such as single-view, lack of texture, low-intensity contrasts and cluttered images [13, 2]. Other representations such as multi-view imaging, CT and DECT, provide more information that would be otherwise hidden [1, 16, 17]. Informative baggage threat detection data is crucial to optimise the quality and performance of detection algorithms.

2.2 CT & DECT Baggage Screening

This section will shift the review focus of threat detection to specifically look at DECT baggage screening. But, the section will begin with investigating single-energy CT.

2.2.1 Computed Tomography

The most common type of scanning technique used for baggage screening is with the use of single-energy 2D X-rays. In recent years, computed tomography (CT) in checkpoint baggage screening is becoming a more common practice to assist with threat detection [18]. This is due to the increasingly strict security regulations causing more advanced imaging technologies to be utilised. The image obtained from a single-energy CT scan is dependent on the Linear Attenuation Coefficients (LACs) of the materials in the target. This is often depicted as a greyscale image. [4]. The LAC characterises how the X-ray's will interact with the target it passes through. This information is captured on the X-ray detector on the opposite side of the object [2]. The X-ray source and detector are traditionally rotated around the target in a spiral motion while taking many images. The images are then reconstructed using software to get a 3D volumetric representation of the target. Though, cases like Micro-X's MBS use several smaller fixed X-ray sources and detectors placed around the target [6].

The use of CT for baggage screening is most predominately used for material detection for explosives [2]. Megherbi et al. [19] discussed the advantages that CT imaging has over conventional 2D X-ray scans. It shows otherwise hidden information that 2D scans cannot show. CT imaging has much better viewing and interpretation of baggage contents. This can provide a new basis of automated threat detection techniques such as 3D object segmentation. However, despite the advantages of the additional information gained from using CT, traditional image analysis techniques used for medical-based CT scans have difficulty being applied to security CT scans [2, 19].

2.2.2 Dual-Energy CT

Conventional CT systems using for security purposes emit a single-energy from the X-ray emitter. However, in the more recent years, the use of two distinct energy levels emitted from the X-ray has been used to provide checkpoint baggage scanners more information about the target [3, 5, 17].

The process of using two distinct energies for CT is called dual-energy CT, known as DECT. Initial DECT studies were conducted in the 1970s, such as the study by Kelcz et al. [20] in 1979. However, CT technology limitations during that period, such as spatial resolution, unstable CT numbers, systematic errors and scan duration's hindered the success of DECT [16, 20].

In 2007, Johnson et al. [16] conducted a feasibility study for material differentiation using dual-energy CT and found that material differentiation in human tissue was possible. Singh et al. [5] conducted a review for explosive detection systems (EDS) technologies. The study detailed that DECT can successfully determine the atomic Z number of material, even when hidden behind metal objects. When using single-energy CT or X-ray, visualisation of less dense material would be lost when a single high energy source would be used (usually above 100 kV) [5]. Liu et al. [17] came to the conclusion that it was possible to create low cost DECT inspection systems. Liu et al. detailed that typical DECT implementations require two sets of X-ray detectors. All three studies outline the potential advantages to using DECT for baggage screening supporting the usefulness of the technology in the field.

2.3 Image Segmentation

The goal of image segmentation is to separate or differentiate regions in an image that have homogeneous texture features to provide more information about the content of the image [21]. Common approaches include region-based, boundary-based or edge-based techniques but there is no general solution [22]. This section will focus on the field of image segmentation and what past and current research has achieved. Outstanding approaches from this review may be deemed appropriate for this project. Before investigating specific image segmentation algorithms and their performance, the advantages of using a Graphics Processing Unit (GPU) is reviewed.

2.3.1 Segmentation on GPUs

The Micro-X MBS use NVIDIA GPUs to assist with the reconstruction of the DECT scans. Therefore, they can be utilised for the segmentation, and hence, this project. Compute Unified Device Architecture, known as CUDA, is a platform created by NVIDIA that allows developers to apply parallel programming on NVIDIA GPUs. This means tasks can be executed on multiple graphics processing threads simultaneously [23, 24]. In recent years, the increase in graphics processing power and chip technology advances have caused studies to utilise CUDA to optimize common sequential approaches [25, 26, 27, 28, 29, 30, 31, 32, 33, 34].

Over a decade ago, Zhug et al. [25] applied CUDA to a popular sequential segmentation approach Fuzzy Connectedness (FC), which is usually computationally expensive. A 24.4× and 10.3× speed up was achieved for small and large data, respectively, when the approach was optimised to use CUDA. Similarly, Sui et al. [27] optimised a Markov Random Field (MRF) segmentation approach and achieved a 10.52× speed increase without degrading the quality of the outputs. The study concluded that the GPU and central processing unit (CPU), which is what non-GPU implementations use, could work hand-in-hand to further optimise the approach. This further supports the suitability of GPU optimisation for image segmentation.

However, the MRF approach is more commonly used on Synthetic Aperture Rader (SAR) images [27], which is not closely related to X-ray baggage scans. A region-growing approach, an implementation with great potential for X-ray baggage scan segmentation [19], was optimised by Dalvand et al. [28] to have a 12.5× speed up over the corresponding sequential CPU counterpart. Happ et al. [29] also managed to achieve a 19× increase when optimising a region-growing segmentation approach. Futhermore, Studies by Vineet et al. [30], Park et el. [31], Allegretti et al. [33] and Playne et al. [34] applied CUDA GPU acceleration to CPU image segmentation methods. All the studies conclude speed increases with little, and often no, degradation to the the quality of the output. The conclusions of the studies described in this section, provide solid grounds to incorporate CUDA GPU optimisation into the segmentation of this project. Especially, if the implementation has to be apart of a 6 second per item checkpoint baggage scan to keep up with commercial scanner throughput [5].

2.3.2 Connected-Component Labelling

Connected-Component Labelling (CCL) is fundamental building block of many complex image segmentation approaches [35, 36, 37, 34, 32, 33]. CCL aims to uniquely label connected components in binary images. The DECT images supplied by Micro-X with categorical material information provide characteristics that would be ideal CCL approaches.

In 1992, Dillencourt et al. [35] successfully generalised an approach for CCL that could be adapted to arbitrary image representations, such as rasters and binary trees. In 2009, He et al. [37] developed a simple parallel implementation of CCL, which at the time was superior to conventional labelling algorithms. However, their study differs from [35] as their approach only considered a raster-scan approach with the image pixels stored in arrays. Lianqiang et al. [36], also took a similar pixel scan based approach but then converts the found labelling information into a tree structure in order to perform the popular union-find technique [36, 35]. Later in 2017, He et al. [38] continued researching CCL and conducted an in-depth review. An outstanding remark that has close relation to this project is that 2D algorithms can be simply adjusted for n-D images.

CCL is a well-researched area, however, many implementations are difficult to optimise using CUDA due to their sequential nature [32]. In 2018, Playne et al. [34] conducted a study to optimise a parallel CCL approach called Komura-Equivalence. They utilised CUDA and managed to not only improve on the algorithm, but the algorithm was faster than two other state-of-art methods. Allegretti et al. [32, 33] continued research in a similar area in 2018 and 2020. The 2020 study also optimised the Komura-Equivalence algorithm but used a block-based approach. Both Playne et al. and Allegretti et al. successfully optimised parallel CCL algorithms using CUDA, providing evidence for it's advantages in CLL implementations.

2.3.3 2D Image Segmentation

In 1985 Haralick et al. [8] investigated image segmentation techniques used in that current time. In the literature review, which is greatly cited by other publications, it discussed the use of general purpose segmentation techniques. This is interesting as it contradicts Ho et al.'s image

segmentation book from 2011 [22] that states there is no general solution. This may possibly demonstrate the increase in need for segmentation and the increasingly diverse applications since the 1980s [39].

A common technique for image segmentation is to utilise minimal spanning trees (MSTs) [40, 41, 42, 21]. Classic MSTs, Prim's algorithm and Kruskal's algorithm were discovered (or *rediscovered*) in the 1950s by Prim et al. [43] and Kruskal et al. [44]. The Prim's algorithm was originally developed in the 1930s by Vojtěch Jarník [9]. Today, MSTs are often used to assist with the process of segmenting images.

Felzenszwalb et al. [40] conducted a 1999 study which investigated and developed an efficient graph-based region-growing image segmentation using MSTs. Their approach, nearly ran at linear time depending on the number of edges in the graph. In 2009, referencing Felzenszwalb et al., Wassenberg et al. [41] also investigated a graph-based MST for image segmentation, which utilised a novel graph-cutting heuristic. Both studies tested their approach's on images of similar nature. [41] demonstrated a more efficient algorithm as it used parallel programming. However, [40] was able to correctly handle low contrast components in the image whilst high contrast image components are present. A more recent study from 2017, Saglam et al. [21] investigated sequential image segmentation based on MSTs. The method, which is heavily influenced by Wang et al. [45], provided quick segmentations but would benefit from a better threshold method. Although efficient, [21]'s method may be more efficient if parallel programming was utilised.

Ganin [46] implemented an efficient MST segmentation using CUDA. The implementation is based on studies from Felzenszwalb et al. [40] and Haxhimusa et al. [47]. The implementation utilised the Thrust API, which is a high-level CUDA interface. The 2D image segmentation algorithm was later included in NVIDIA's CUDA software development kit as an example algorithm.

Despite success using region-based image segmentation methods [40, 41, 21], Guo et al. [48] successfully developed an image segmentation approach that used active contour models for boundary detection. Guo et al. successfully created an accurate approach that was faster than state-of-the-art counterparts such [49]. However, both approaches require training, which fits outside the scope of this project.

2.3.4 3D Image Segmentation

Recently in 2021, Howes et al. [11] investigated a region-merging segmentation method for CT data. The merging criterion used a statistical process to quantitatively measure the texture of a region. Though, the method was successful, the most interesting component of the process was converting the image into *superpixels* as an initial segmentation step. The approach used in [11] was developed by Li et al. [50]. Superpixels group local data into uniform patches, which decreases the number of pixels ultimately simplifying the final segmentation process [11]. Ye et al. [51] successfully segmented lesions in CT data, which utilised a novel superpixel approach. Farag et al. [52] also successfully utilised superpixels in their CT segmentation bottom-up approach. The success of superpixels in [11, 50, 51, 52] provide evidence that they may be useful for creating

an efficient segmentation approach for this project.

Region growing algorithms commonly require the initial seeds to identify the segments. Often the location at which these seeds are placed highly dictate the final quality of the segmentation. In 2003, Wan et al. [53] proposed symmetric region growing (SymRG). Rather than proposing a new algorithm for segmentation, like many of the studies mentioned in this review, [53] defined theoretical criteria that researchers can follow when defining a region growing algorithm to ensure the independence of the initial seed placement. However, this does not ensure an automatic segmentation approach as some sort of initial seeds are required, just the location of those seeds are not as important. Lin et al. [54], however, proposes an unseeded region growing 3D segmentation approach, one year prior to the [53]. They were able to develop a relatively fast and reliable method, which also did not require tuning of parameters.

Graph cuts is a common graph theory strategy to separate a group of vertices into two disjoint sets. Both Privalov et al. [42] (2018) and Chen et al. [55] (2011) utilise graph-cutting to segment CT image data. [55] being the earlier study used a sequential approach to the segmentation. The approach was actually a hybrid of k-means clustering and graph cutting and was successful on cardiac dual-source CT images (not exactly the same as dual-energy CT). [42], with inspiration from [41], demonstrates how graph cutting segmentation approaches can be converted from sequential to parallel using the GPU. Privaloc et al. was able to achieve successful results on CT data and a considerable speed increase in comparison to it's sequential counterparts.

As discussed in Section 2.3.3, Guo et al. [48] and Ni et al. [49] both created 2D boundary-based segmentation methods. Their studies also successfully applied their 2D methods to 3D data. However, the approaches they used both required some sort of training, which is still outside the scope of this project. In 2008, Kruegar et al. [56], like [48], investigated active contours for 3D segmentation. However, [56]'s successful approach did not require the use of training and claim to achieve a linear computational complexity.

2.4 CT & DECT Baggage Image Segmentation

Image segmentation for X-ray baggage scans have been a topic of discussion for decades. In 1998, Paranjape et al. [15] investigated the segmentation of handguns. This study also focused on dual-energy X-rays, but didn't not use the dual-energy information to determine the atomic characteristics of the objects. Despite investigating three different methods, Paranjape et al. concluded that further work is needed. Despite segmentation for threat detection being a popular research topic, Mouton et al. [2] described the work for automated segmentation for unknown 3D objects as limited. However, Mouton et al.'s work has focused on low resolution CT image with cluttered baggage, which is related to that of real-world data.

Traditional CT imaging segmentation approaches yielding success in the medical industry cannot be directly translated to the security industry due to differences in the data for each use case, more specifically, the difference in image quality needs, *a priori* information provided to medical-based algorithms and a higher level of image complexity in security images [2]. The increased

image complexity is a result of a broader variety and density of object throughput. The presence of metals can result in image artefacts such as streaking and shading [3]. Megherbi et al. [19] found that medical region-based CT segmentation outperforms boundary-based techniques in the context of CT baggage screening. However, artifact noise, object complexity and the absence of *a priori* information limits the usefulness of applying medical-based segmentation techniques in the security domain [19]. The inability to confidently apply medical-based techniques allows DECT CT image segmentation to be an open problem in the security industry [4].

Whilst conventional CT implementations use a single energy, the use of two distinct energy scans allow the ability to determine the chemical composition (e.g. Atomic Z number) of the target object. With the use of a predefined dual-energy index (DEI) lookup table to determine the object chemical composition, the material ID can be found at each CT scan voxel. An example of a lookup table for lung tissue by the National Institute of Standards and Technology (NIST) is found in Appendix B. Appendix B also goes into more detail on how the material value can be determined from dual-energy. The material information can act as *a priori* information for the image segmentation algorithms that would otherwise yield results with inadequate quality [2, 4]. Despite previous CT segmentation focusing on single object segmentation in preparation for object classification [57], Mouton et al. [4] investigated the use of dual-energy in CT segmentation and successfully developed a novel technique to segment a scan of *clustered* objects from DECT data obtained from the baggage scanners in the security industry.

In 2006, Ding et al. [58] proposed an Attribute Relational Graph (ARG) matching based segmentation method for X-ray threat detection. The extra attention to create a model suitable for overlapping images was insufficient for a successful conclusion and stated that their "model base is incomplete." Ten years later, Zalia et al. [39] also proposed a graph based segmentation strategy. But despite, conducting three different experiments, was not able to conclude successful results and concluded that their approach required high parameter tuning and would succumb to overlapping objects, which is very common in X-ray baggage scans. Not only did both studies conclude that more research was needed for their strategies, the data they used for their segmentation was 2D and single-energy. It is an obvious choice to omit 2D single-energy segmentation studies but many 2D implementations can be applied to 3D data [34].

Figure 2: Example CT segmentation from Wiley et al. of a tripod (left) and candle (right). [10]

Megherbi et al.'s [19] findings that region-based implementations where better at segmenting objects in 3D baggage scans is supported by Wiley et al. [10]. Wiley et al. proposed the Stratovan Tumbler. Whilst their solution successfully addressed low-resolution, noise, poor contrast and streaking artifacts (which are common X-ray baggage scan characteristics), there implementation could take 40 minutes to execute (with their computing equipment). An example segmentation from the study is shown in Figure 2. In contrast, Grady et al. [57] also used a region splitting approach. The successful study used CCL (detailed in Section 2.3.2) to separate the background

and foreground and then recursively split the two components into the desired segments. The use of the connected components by Grady et al. [57] is the first use of superpixels as a technique in baggage segmentation. Howes et al. [11] took advantage of the time computationally advantage of superpixels but applied the technique (successfully) to a medical problem.

Mouton et al. [2] conducted an in-depth literature review on automated image segmentation for 3D CT baggage screening. In this study, multiple image segmentation studies were investigated. Mentions of necessary *a priori* knowledge deems medical-segmentation, for the most part, inappropriate. Despite the study focusing on CT and DECT implementations, Feng et al. [59] was mentioned for their study on 3D volumetric data. Similarly to the unseeded segmentation by Liu et al. [54] and unlike the seeded region growing by Wan et al. [53], Feng et al. [59] used homogeneous regions to automatically determine the initial seed map prior to the adaptive region growing.

A number of studies approach the baggage segmentation problem using some sort of training method [60, 1, 18, 61]. Babaheihari et al. [60] created a successful joint segmentation and material recognition model, which outperformed alternative pixel-based appearance models. However, the approach was highly dependent on availability of training data on the specific materials of interest. Wang et al. [18, 61] investigated the effectiveness of 2D semantic segmentation approaches for CT security data. Wang et al. found that the approach's results was comparable to the 3D counterparts. However, the approach required the target material to be known, and only three target materials were tested.

2.5 Data and Performance Metrics

2.5.1 Data

Throughout the literature review, similarities in the data used by the studies have been established. Two popular datasets have be identified and will be discussed in this section. This section is important as it may identify potential datasets that could be used in this thesis.

In 2010, the US DHS Northeastern University Center of Excellence (COE) initiated the Awareness and Localization of Explosives-Related Threats (ALERT) [59]. The goal of the program was to improve the performance and quality of present state-of-the-art baggage segmentation implementations. As a result, the ALERT dataset was created. The "Segmentation of Objects from Volumetric CT Data" project was the first phase of a multi-year project for the research and development of automated objects of interest detection algorithms for CT-based checkpoint baggage scanners [59].

Wiley et al. [10] used the ALERT dataset for the implementation of their 3D region growing based method. The dataset was also utilised by Karimi et al. [62] when developing flexible methods for the CT baggage segmentation evaluation. Mouton et al. [4, 63, 2] used the dataset as well in their threat detection related studies. This included but not limited to 3D unknown object classification and 3D unknown object segmentation.

Mouton et al. [4] also pointed out that this dataset was used to develop several other image segmentation techniques for baggage CT scans. These include but not limited to Song et al. [59] for their seedless region-growing segmentation-and-carve approach, Grady et al. [57] for their recursive partitioning technique and Harvey et al. [59] for the utilising the sieves class of algorithms for their CT baggage segmentation approach.

Unlike the ALERT dataset, the SIXray dataset contains 2D images. This dataset contains over 1,000,000 images of X-ray baggage scan [7]. The dataset has similar characteristics to the GDXray dataset [14], which was developed in 2015. However, the GDXray has less than 20,000 images and the images predominately do not contain many overlapping objects in the scans. This dataset was recently collected in 2019, almost a decade after the ALERT program was first initiated. Although the SIXray dataset is used by Hassan et al. [13, 64], and Miao et al. [7] for threat detection related studies, the ALERT dataset is better suited for CT baggage segmentation due the nature of the volumetric data.

2.5.2 Performance Metrics

Through reading literature on segmentation for not only baggage but also general use cases, there were multiple performance metrics used. These include but are not limited to, Automated QUality Assessment (AQUA), Receiver Operating Characteristic (ROC) curves, Random Forest scores, precision, recall, F-scores, intersection over unions, dice coefficients and mean average accuracy [57, 4, 3, 13, 21, 11].

Karim et al. [62] conducted a study to develop evaluation methods to measure over- and under-segmentation. The study concluded that the two segmentation evaluation methods they developed would successfully select the correct segmentation method. Grady et al. [57] also developed a segmentation evaluation method called AQUA. Their quality assessment tool is also utilised in their successful segmentation implementation; AQUA drives the recursive region splitting. Unlike Karim et al. [62], upon reading literature, AQUA was also used in other studies, such as Mouton et al. [4] in 2015.

Recall, precision and F-score are metrics used in machine learning threat detection studies [7, 64, 14], but were also used by Saglam et al. [21] in their sequential minimal spanning tree segmentation. Whilst these scores measure the confusion matrix, which includes true negative, false negative, false positive, true positive, studies such as Howes et al. [11] use used pixel-based metrics such as Area-Under-The-Curve (AUC) and Dice Similarity Coefficient (DSC).

Through reading through multiple segmentation studies, it is clear that there are a vast range of different methods on evaluating the performance of the implementations. The best choice for performance metrics is dependant on the nature of the problem, implementation and data.

2.6 Review Conclusion

This extensive literature review has mention over 60 references. Threat detection has an increasing demand for growing populations have higher passenger throughput. Although initial studies

began in the 1970s by Kelcz et al. [20], only recently has DECT been considered a feasible technology for threat detection [17]. DECT has great potential for creating safer travelling when used appropriately.

2D and 3D image segmentation is a very large research area. The common region-based, boundary-based and edge-based techniques have made tremendous advances especially with the use of GPU acceleration with NVIDIA's CUDA. Many of these segmentation implementations have had fantastic success in the medical industry. However, as Mouton et al. [2] has pointed out, image quality, *a priori* information and image complexity were the biggest complications to applying medical segmentation approaches to security data.

Though high quality segmentation algorithms for baggage data is possible, as shown in Wiley et al. [10], it's the performance that prevents many implementations from being suitable for real-world applications. The number of pixels in 3D CT and DECT data can dramatically increase the computational time as well. Despite this, the advancement of GPU technology has evidently shown that using the right tools, such as CUDA, it is possible to accelerate most segmentation algorithms, even those traditionally sequential in nature. For example Ganin's [46] CUDA MST segmentation, which is originally based of Felzenszwalb et al.'s sequential MST [40], achieved dramatic decreases in segmentation time. From reading segmentation literature, it is clear that no one or few performance metrics are the most appropriate. A range of metrics are used to evaluate problems of different nature.

DECT images have the necessary information to present categorical data by using the DEI to determine the Atomic Z number at each pixel. This itself is a powerful tool, but also has great qualities for segmentation, especially CCL. The use of superpixels for segmentation have shown great speed advantages in [11] and [50], but only Grady et al. [57] has utilised them for CT baggage data. Also, in that case, only two superpixels were used to separate the background and foreground. The use of CCL and superpixels have the potential to have great time computational advantages if applied before a high quality graph or region based segmentation algorithm. This gap of knowledge may provide the necessary performance qualities to implement a real-world applicable baggage segmentation algorithm. Following this, the application of the additional dual-energy material information found in baggage scans, and superpixels from CCL as an initial segmentation for baggage DECT data constitutes the knowledge gap that is addressed by this project.

Even with great advances in baggage segmentation, there is still work to be done for it to be applicable to the real-world. With the use of DECT baggage scans and optimisations using recent technology advances, feasible automatic segmentation is possible.

3 | Methodology

This section outlines the methodology carried out for this thesis. The toolkit algorithms are discussed, which is followed by the implementation of the toolkit. This section concludes by discussing the data and the testing procedure.

3.1 Algorithms

The aim for this thesis was to create an image segmentation toolkit for the Micro-X baggage scanner. Following in the theme of a toolkit, the implementation compromises of four different algorithms. One of the algorithms also has an optimised version. These are grouped into two categories; connected-component labelling (CCL) and minimal spanning trees (MST).

3.1.1 Connected-Component Labelling

Due to the nature of the material identification data from the DECT images, the images are represented as categorical values. This allows for simpler implementations in comparison to traditional greyscale X-ray image segmentation. Therefore, the first two algorithms used in the toolkit are CCL algorithms. To demonstrate the significant advantages of using the NVIDIA's CUDA framework, one algorithm is a sequential algorithm which executes on the CPU, whilst the second algorithm utilises multi-threading on the GPU using NVIDIA CUDA.

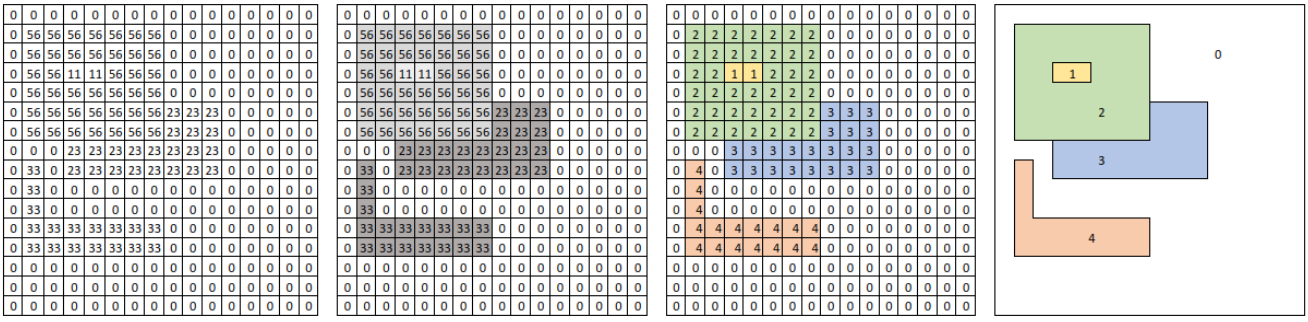


Figure 3: Connected-component labelling representation of components in a 2D slice.

3.1.1.1 Sequential Connected-Component Labelling

The first algorithm implementation for the DECT image segmentation toolkit is an arbitrary CCL algorithm. This particular CCL algorithm is component of the Watershed implementation by Vincent et al. [65] and uses graph traversal as the main means of finding components. Unlike the other three algorithms, the C++ code for this algorithm was developed without utilising publicly available open-source implementations.

There were two main changes made to the algorithm to work with Micro-X's DECT images. The first change was to let the algorithm distinguish between materials (categorical values). The second change was to convert the 2D implementation to 3D.

To convert the algorithm to work with categorical image pixels instead of binary, no threshold function was included at the start of the algorithm. As the image is traversed instead of checking whether the voxel value is 0 or 1, the voxel value was checked for whether it had the same material ID integer value or not. For example, in Figure 3, the values of 11, 13, 23, 56 and 0 are differentiated from each other.

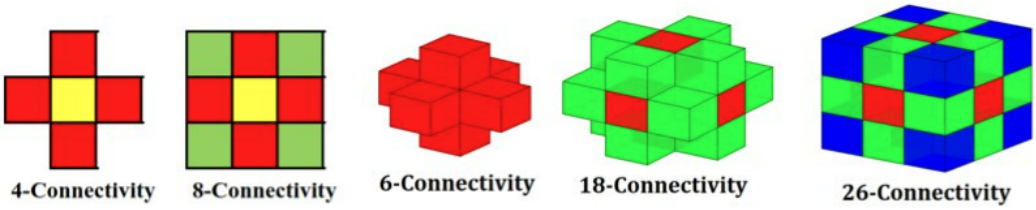


Figure 4: 2D (4, 8) and 3D (6, 18, 26) connectivities.
[66]

Converting the algorithm to work on the 3D Micro-X DECT images required two components of the algorithm to be altered. The first being that all voxels in 3D image needed to be traversed instead of just traversing the 2D pixels.

The next component that needed to be changed was that each voxel has more neighbouring voxels than the pixels in the 2D image. There are different sets of neighbouring pixels for 2D and voxels for 3D images. For 2D images the two sets of possible neighbours for a pixel are 4- and 8-Connectivity. This had to be changed to 6-, 18- and 26-Connectivity to work with the Micro-X DECT images. The connectivities are demonstrated into Figure 4. All 3D connectivities were implemented in the toolkit for this sequential CCL implementation. However, as discussed in Section 3.1.1.2, the parallel CUDA CCL implementation only uses the 6-connectivity. Therefore, when comparing the two implementations, only the 6-connectivity implementations were used.

3.1.1.2 Parallel CUDA Connected-Component Labelling

The second implementation in the toolkit is a CCL algorithm called the Playne-Equivalence CCL, developed by Playne et al. [34]. Unlike the method described in Section 3.1.1.1, this particular algorithm utilises the process of parallel processing on the GPU using the NVIDIA CUDA framework but still aims to find unique labels for each component, as shown in Figure 3. Playne et al. demonstrated the improvement that the Playne-Equivalence algorithm had on the Label- and Komura-Equivalence algorithms, which also use the NVIDIA CUDA framework [34].

To implement the Playne-Equivalence CCL for the segmentation toolkit, Playne et al.'s C++ code was used [34]. The code already had a 3D implementation, resulting in a simple transition to integrate the Micro-X material ID data. When assessing a voxels neighbours, it uses 6-connectivities, as shown in Figure 4. Also, the conversion from only being able to use binary

data to being able to use categorical data was done by removing any initial thresholding. When the algorithm checked when the a neighbouring pixel was apart of the same component or not, instead of checking whether it was 1 or 0, it checked if it had the same material ID value.

There are three GPU (or *device*, following CUDA naming conventions) kernels used in the Playne-Equivalence algorithm. These are initialisation, analysis and reduction. Device kernels are functions that are executed on the GPU threads in parallel. Each thread on the GPU will have it's own thread, block and grid identification value [23]. The algorithm uses the thread, block and grid identification values to give each voxel in the image a linearised address, as shown in Figure 5.a. The unique addresses are stored in a single 1D array on the device.

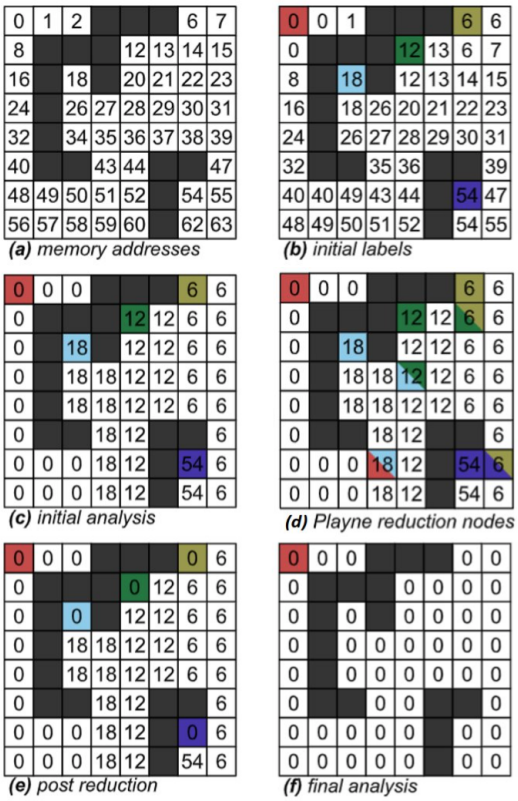


Figure 5: 2D example of the Playne-Equivalence algorithm. [34]

The method of the algorithm is to find the *root* voxel address for each component. Using the initialisation kernel, each voxel gets the address of a neighbouring voxel with the smallest linearised address. Only 3 neighbours are considered in the 6-connectivity range. As this is hard to visualise in a 2D format, an example of possible neighbours for a 2D use case where only 2 possibilities are available is shown in Figure 6. This process is shown in Figure 5.b. The analysis kernel then determines the lowest possible linearised address by following the chain of addresses, as shown in Figure 5.c and d. The reduction kernel merges touching components by joining their root nodes. This is shown in Figure 5.d, e and f.

3.1.2 Minimal Spanning Tree

Following success in image segmentation studies [40, 30], the implementations for the greyscale image segmentation of the provided DECT data utilise MST algorithms. Using the same pattern as the CCL algorithms, one algorithm will execute sequentially on the CPU, whilst the other will use the CUDA framework to perform parallel executions on the GPU. The sequential implementation also has an optimised version.

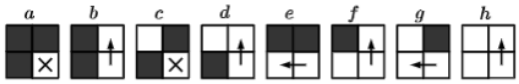


Figure 6: Possible CUDA connected-component labelling neighbourhoods for 2D images. Bottom-right pixel is pixel being analysed. [34]

In graph theory, a MST is a subgraph. The MST graph includes all nodes and any node can be traversed from any other node in the graph. The MST is minimal as it has a total weight that is equal to or less than any other spanning tree weight, as shown in Figure 7 [9, 44]. In terms of image segmentation, each pixel is a node of an MST and has an edge with each of it's neighbours in the connectivity. Felzenszwalb et al.'s [40] method of calculating the weight between two pixels

was to use the absolute value of the two pixels values. Several MST's are constructed for an image and each correspond to a segment. As pixels are fully connected in an image, often a stopping function is used to prevent the MST reaching it's final form to achieve the optimal segmentation.

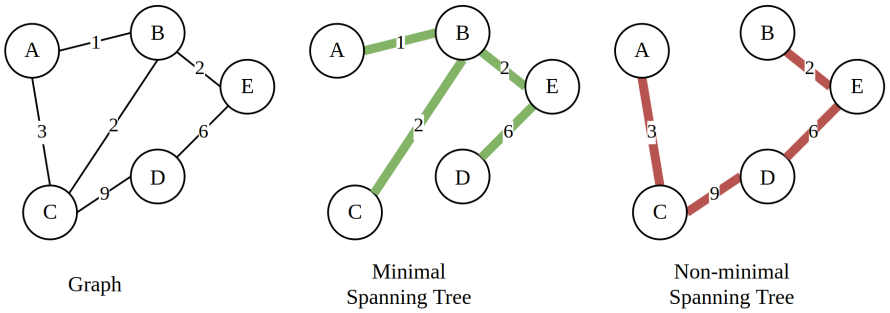


Figure 7: Minimal and non-minimal spanning tree examples.

3.1.2.1 Sequential Minimal Spanning Tree

The sequential MST algorithm used in the toolkit is a modification of Motwani's implementation [67] of Felzenszwalb et al.'s study [40]. This implementation is built upon Kruskal's MST [44].

As Motwani's code is open source [67], the segmentation implementation only had to be converted from 2D to work with the 3D CT data provided by Micro-X. This process involved reordering the path in which the image is scanned to obtain all the information. As well as traversing the X (horizontal) and Y (vertical) directions of the 2D image, there was a third direction, Z. The observable neighbourhood for each voxel (previously called pixel for the 2D image), had to be expanded from 4- to 6-connectivity, shown in Figure 4. Motwani's code had an implementation for both greyscale (single channel) and colour (three channel RBG). CT data is represented as a greyscale image due to the single channel nature of the attenuation information, so only the greyscale implementation was used.

The underlying minimal spanning tree algorithm used for this implementation is Kruskal's MST [44]. In terms of graph theory, to find the MST of a graph, first all edges betweeen nodes are sorted in ascending order. Until all nodes are apart of the MST, add the smallest weight (and it's two nodes) to the MST unless it creates a cycle in the graph (a path that starts and finishes at the same node).

In Felzenszwalb et al.'s [40] implementation of Kruskal's MST for 2D image segmentation, each pixel begins as a stand alone component. Edge weights are determined using the absolute value of the difference of two node pixel values. Iterating through the sorted list of edges, components are merged if they satisfy the condition obtained from Felzenszwalb et al.' study . In summary, the condition is satisfied if the external difference between to two merging components is less than the internal difference of any of the two components relative to their size. The relative internal difference being the maximum edge weight in the component with an additional parameterised thresholding component and the external difference being the minimum weight edge connecting the two components.

Motwani's implementation of Felzenszwalb et al.'s study [67, 40] used structs and struct pointers to represent the CT and segmented image. Standard C++ arrays and linked-lists were used to create a relationship between the entities. The basic structure of how this was implemented is shown in Figure 8. The linked-lists allowed components to be merged without having to reallocate memory for the new larger component.

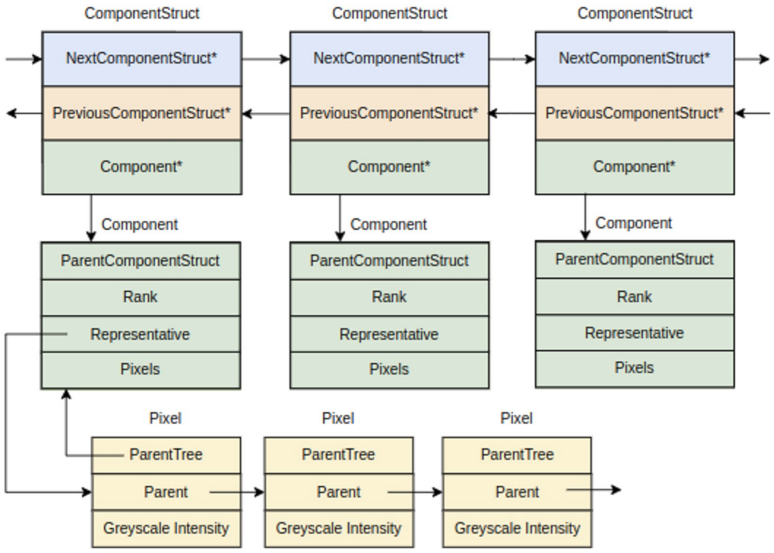


Figure 8: Motwani's struct and linked-list data structure for MST image segmentation. [67]

In addition to the regular sequential MST implementation by Motwani and Felzenszwalb et al. [67, 40], a slightly modified version is also included. As volume space in a DECT baggage scan is not required to be segmented, it can be ignored. In a complete DECT image, the material ID can be used to ignore voxels that are 'volume'. As Micro-X have not converted the test reconstructions to material ID images yet (discussed in Section 3.3), a threshold function has been included in a modified version of the sequential MST implementation. This optimisation reduces the number of voxels and edges that need to be analysed for the segmentation. In regards to the segmentation toolkit, this optimised variant is added as an additional option when executing the sequential MST algorithm on a greyscale DECT image.

3.1.2.2 Parallel CUDA Minimal Spanning Tree

The final implementation in the DECT baggage scanner image segmentation toolkit is a MST algorithm that uses the GPU to execute computations in a parallel manner. On top of being developed in C++, the implementation also uses the NVIDIA CUDA framework and Thrust API. The implementation is apart of the CUDA software development kit provided by NVIDIA. The implementation is by Ganin [46]. Ganin has detailed that the implementation is heavily based on Vineet et al. [30].

Unlike Motwani's implementation, which is based on Kruskal's MST, Ganin's Implementation is based on Boruvka's MST algorithm [67, 46, 68]. In summary, Boruvka's algorithm involves all nodes starting off as their own component. For each component, merge itself and the opposing

node on it's edge with the lowest weight. Iterate that process of finding the edge with the smallest weight for each component. An example of this process is shown in Figure 9. In terms of image segmentation, there can be a number of different segmentation results as the segmentation can be stopped after any of the iterations before the final MST is created. In Ganin's implementation of Boruvka's MST, there are several resulting segmented outputs. An example of eight different level outputs is shown in Figure 10. The image shows that after each iteration, the number of components decreases as their size increases.

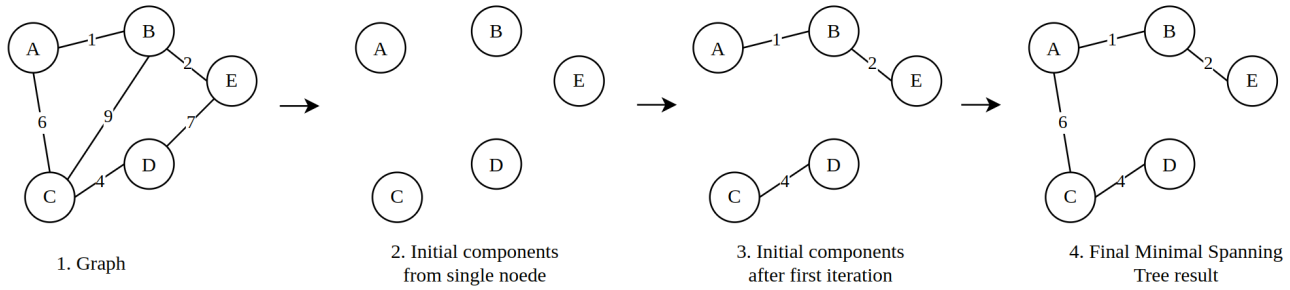


Figure 9: Boruvka's minimal spanning tree example.

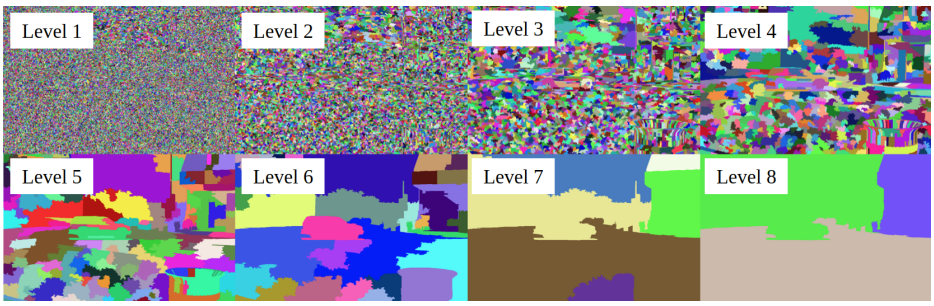


Figure 10: Ganin's minimal spanning tree example outputs for a 2D image.

The Ganin's MST implementation main features included the use of the Thrust API and like the Playne-Equivalence CCL in Section 3.1.1.2, linearising the representation of the image. The images were linearised by using an indexing function that would use the X, Y and Z coordinates as parameters to return an index location for a 1D array. This is discussed in more detail in Section 3.3. Three different 1D arrays were used to store voxels, edges between voxels and edge weights.

Like Motwani's implementation in Section 3.1.2.1, Ganin's MST had to be converted from 2D to work with the Micro-X 3D CT data. This process was similar to the conversion process for Motwani's implementation. As well as traversing the X and Y directions of the 2D image, the Z direction was also required for the 3D image. The indexing function also had to be altered from 2D to 3D.

Ganin's MST implementation used multiple Thrust API's to perform operations such as sort, sequence and inclusive scan. These high-level functions would use the CUDA framework underneath to perform the functions on the GPU in an optimised manner. Vineet et al.'s study, which is what Ganin's GPU MST is based on, proposed to use of the Thrust operations such as sort [30, 46].

3.2 Software Implementation

This section will detail the implementation of the segmentation toolkit. This includes the programming, reproducibility and architecture of the toolkit.

3.2.1 Programming

The toolkit had to not only be compatible with the Micro-X data, but also be compatible with their software development environments. The Micro-X MBS image software is developed using C++. Therefore, as mention previously, the toolkit is developed using C++. Developed in 1985 by Bjarne Stroustrup, C++ is a cross-platform low-level programming language and is an extension of the C programming language [69]. The toolkit only uses the C++ Standard Library to aid the integration into the MBS image system. The C++ Standard Library is apart of the C++ ISO standard [70]. CUDA and it's derivative Thrust are an exception and it is known that the Micro-X baggage scanner image system already utilise these frameworks for GPU acceleration.

3.2.2 Reproducibility

An important aspect of this toolkit is for Micro-X (and the reader) to be able to reproduce the results provided in this thesis. This is to allow the segmentation toolkit to be utilised by the imaging team for the Miniature Baggage Scanner (MBS) [6]. This also prevents the work from contributing to the 'reproducibility crisis', as mentioned in a CT segmentation study by Howes et al. [11].

The measure to ensure reproducibility was to make the source for the toolkit open-source. The GitHub repository for the toolkit has information on what computer system requirements are necessary and how to run the toolkit. The specifications of the computer that produced the results for this thesis is detailed in Section 3.3.2. The open-source Micro-X MBS DECT Image Segmentation Toolkit can be found at https://github.com/duns0089/Dual-Energy_CT_Image_Segmentation_Toolkit.

3.2.3 Architecture

The architecture for the toolkit is shown in Figure 11. There are several main features to the toolkit, the source code, which includes the Python build script, build outputs directory and the test images. The Python build script is included to aid the process of compiling and executing tests automatically. The test images are stored in their corresponding test images folder. The compiled toolkit is placed in a build outputs directory after it has been built. The main file that contains the *Main* C++ function is located in 'main.cu'. The '.cu' extension is used instead of the common '.cpp' C++ extension due to the use of CUDA. The 'main' calls each of the algorithm implementation classes to execute the segmentation on the inputted image. There are two options for the main file. If a simulated material ID image is inputted, the CCL algorithms are executed. If a greyscale test reconstruction is inputted, the MST algorithms are executed. The data is further discussed in Section 3.3. The utility folders support their corresponding implementation. Further information to run the software is provided in the code repository.

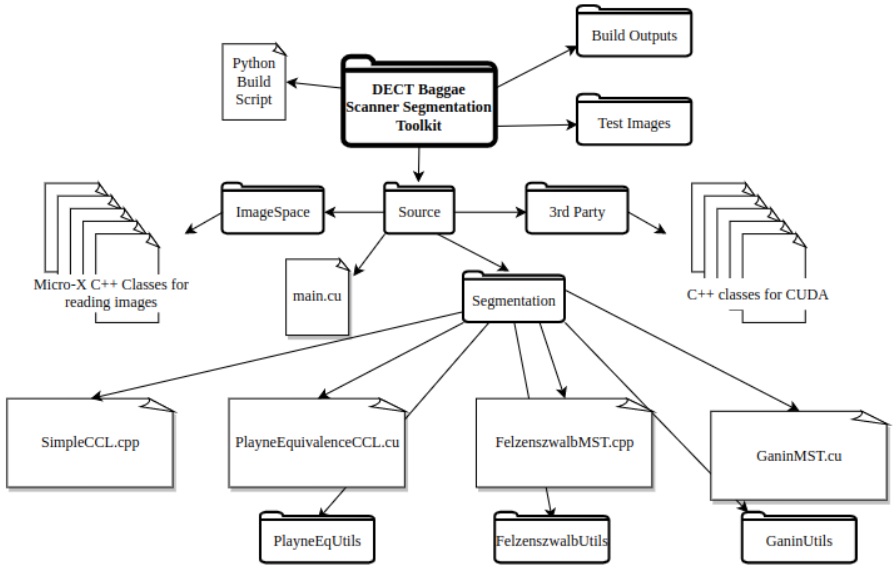


Figure 11: Toolkit architecture.

3.3 Data

As this thesis is an industry project, the data was supplied by Micro-X. The implementation of the toolkit was developed to work with the Micro-X data. There were two types of data, simulated DECT images and real reconstructions of test CT scans. For the duration of the project, there were limited test scans available. Therefore, the majority of the data used to develop and test the segmentation toolkit was the simulated DECT images.

Reading in the image data involved utilising C++ code provided by Micro-X. The code would allocate memory, which would depend on the size of the image (total voxel count) and the image type (8 bit for simulated material ID integers and 32 bit for greyscale reconstruction real numbers) The code would store the image information as a 1D linearised array to the allocated memory. Using a linearised index function, shown in Equation (3.1), a voxel value could be read or edited when given the X, Y and Z coordinates and image dimensions.

$$Index = Z + X * Z_{MAX} + Y * Z_{MAX} * X_{MAX} \tag{3.1}$$

3.3.1 Simulated Data

The main type of data used was the simulated data. An example image is shown in Figure 12. These images where generated using a Micro-X developed software. The image could be configured to have be any 3D size and contain any number of cuboids, spheres and/or cylinders. The material of these shapes can be specified for each voxel as well. This would result in a 3D DECT image, with the material ID at each pixel. However, this would not provide any information on the greyscale intensity value provided by the attenuation of the target matter. The two CCL implementations will use only the simulated data for testing as Micro-X have yet to use the dual-energy information to convert a greyscale DECT image to a categorical material ID image. The two MST implementations will use the simulated data as well as the test reconstructions.

In the simulated images, each voxel is represented by an integer (ID). These values correspond to a certain material in a lookup table. The lookup table is provided by Micro-X in the form of a JSON file. However, the lookup table is closely related to the National Institute of Standards and Technology (NIST) lookup table [71].

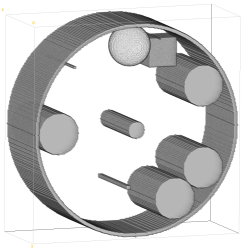


Figure 12: Simulated Dual-Energy CT example image.

3.3.2 Test Reconstructions

Test reconstructions are CT images that Micro-X provided for the development of the toolkit. They are the outputs of the reconstructed Micro-X baggage scanner CT scans. An example of test reconstruction of a Lego figure is shown in Figure 13. Only three reconstructions were provided. The test reconstructions will only be used by the two MST implementations for testing.

Throughout the project, attempts were made to obtain the ALERT dataset. The ALERT CT segmentation dataset is detailed in Section 2.5.1. The process to obtain the dataset involved a formal Non-Disclosure Agreement between Flinders University and Northeastern University. However, this process was unfortunately unsuccessful.

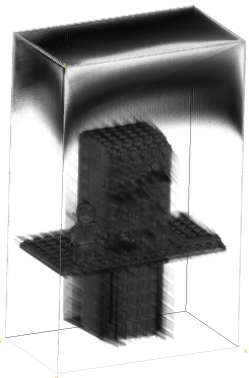


Figure 13: Micro-X test reconstruction example image.

There were obvious limitations to the data provided by Micro-X. At the time of writing, the Micro-X software responsible for reconstruction of the CT scans to obtain the 3D image is still in development. As a consequence, there are obvious artifacts and streaking in image. It is outside of the scope of the project to remove artifacts and streaking from the images. However, it was expected that the artifacts and streaking would negatively affect the MST segmentation results. Only being supplied three images to test the MST implementations is insufficient. Furthermore, no ground truth was supplied for the images so this limited any in-depth quality analysis.

3.4 Testing

This section will detail the process of conducting the tests to obtain the results shown in Chapter 4 and includes the computer specifications, software used and the procedure for running the tests.

3.4.1 Computer

Hardware	**Software**
CPU: Intel i7-11800H, 8 Core, 4.6 GHz GPU: NVIDIA GeForce RTX 3050 Ti Laptop RAM: 32 GB DDR4, 3200 MHz	OS: Ubuntu 22.04 LTS NVIDIA Driver: 515.65.01 CUDA: 11.7 C++: CPP 11.2

Table 1: Test and development hardware and software specifications.

It is important to document the computer and it's hardware and software specifications to aid the process of reproducibility, as discussed in Section 3.2.2. This is following the same convention as other studies such as ones by Ganin and Playne et al. [46, 34]. Table 1 lists the specifications

for the hardware and software for the toolkit test and development platform. The software also includes the version of drivers and compilers used.

3.4.2 Software

Due to the toolkit being able to run itself, limited additional software was needed for the project. Two other software programs were used when conducting the tests, *Fiji (ImageJ)* and *Microsoft Excel*. Fiji, shown in Figure 14, is a image viewing software that comes packaged with several plugins that are omitted from ImageJ. These plugins allow the input images and segmentation images to be viewed in a 3D plane instead of having to view a single slice at a time. This plugin was used to capture a 3D view for the results in Chapter 4.

Microsoft Excel was used to store and plot the data resulting from the tests. This includes but is not limited to, the time it takes to segment the image, the size of the image (total voxels), number of components and image dimensions. A screenshot of the data in Excel is shown in Appendix A.

3.4.3 Test Procedure

The two results that were of interest for this thesis were the time it takes to segment an image and the segmentation quality. In order to measure the time of the segmentation, the toolkit includes a basic timer class. This timer class allows a start and end point to be placed in any two points in the code. This timer class was used for all four implementations. For consistency between all implementations, the segmentation time includes allocating memory on the GPU (on the CUDA implementations) as this overhead should be considered in a real-world application. However, the time it takes to save the segmentation to a file is not included.

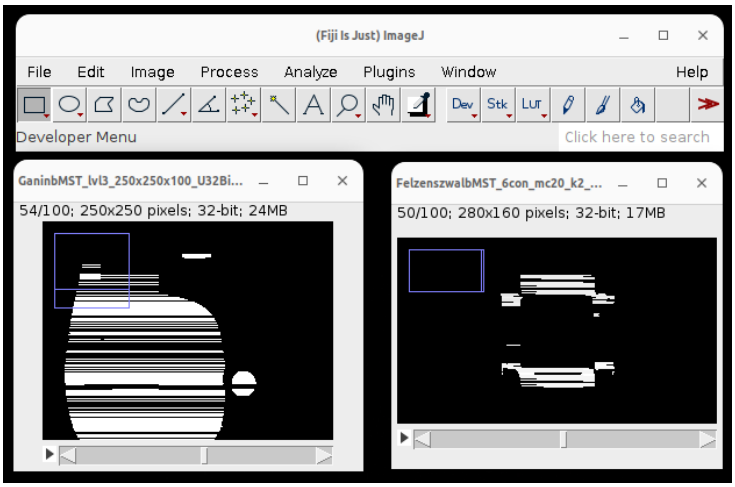


Figure 14: Fiji (ImageJ) graphical user interface.

In order to eliminate any potential uncontrolled variables affecting the test results, all tests were executed in the same environment. All tests were ran independently to minimise the risk of allocated memory not being deallocated before the next test. All other system processes (that weren't critical to the operating system) were stopped while running tests. This prevents other processes from using the computer resources at the same time.

The two CCL implementations had sufficient data as the material ID images were simulated. As only three reconstruction images were provided to test the toolkit, in order to get a better understanding on how the MST implementations perform with different sized images, different sizes of the same reconstructions were used.

4 | Results & Interpretation

This section will present the results of the toolkit. There are five implementations, two for connected-component labelling and three for minimal spanning tree. Each section will present the image results and the time difference for the sequential and parallel CUDA implementations in the form of a scatter plot. The results are interpreted in this section and further discussed in Chapter 5. All data for the graphs are shown in Appendix A.

4.1 Connected-Component Labelling

The results for the CCL implementations are split into two categories, results from simulated data and segmentation time.

4.1.1 Simulated Data

Due to the categorical nature of the CCL algorithms, displaying images of the outputs do not show the reader much information. Furthermore, outputs for sequential and parallel CUDA CCL algorithms yield the same output. The focus for the CCL results is the graph in Figure 17. Therefore, several example inputs and only one example output is presented.

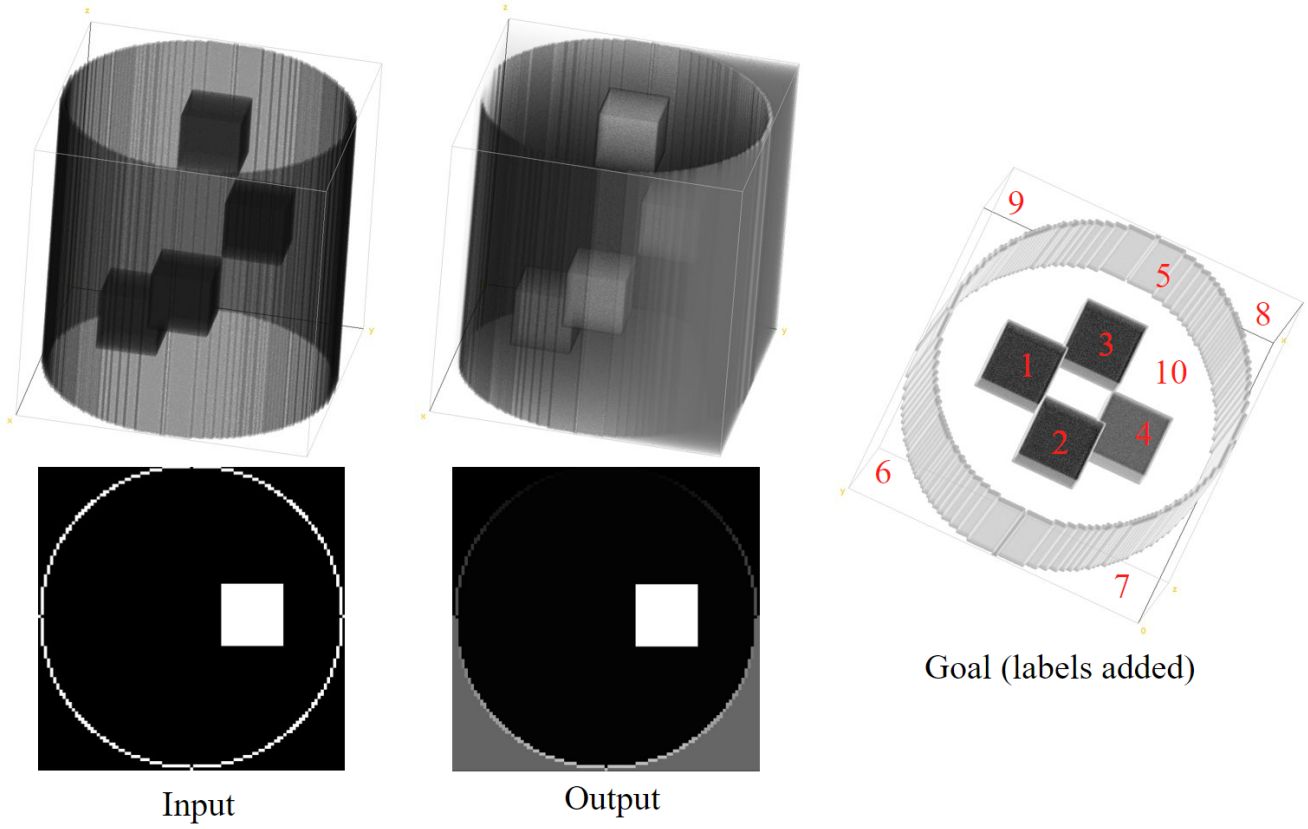


Figure 15: Example result for a connected-component labelling algorithm (size: $99 \times 99 \times 99$).

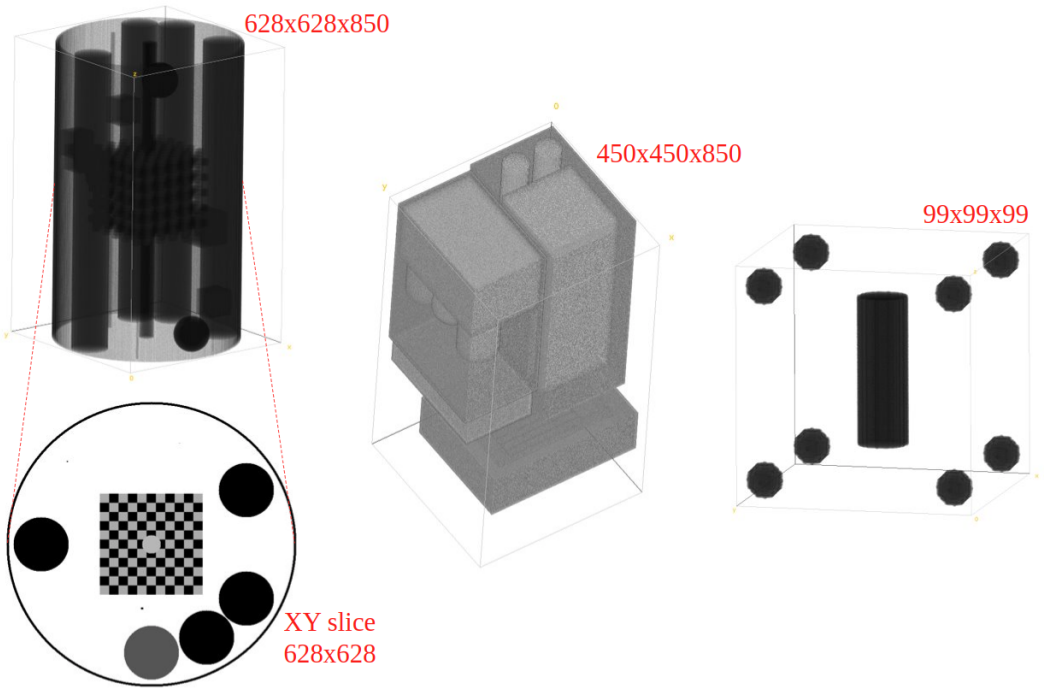


Figure 16: Example simulated DECT image inputs for connected-component labelling.

The quality of the output for any image is independent of the algorithm used because the input image uses categorical data, the material ID. When the algorithm determines whether a voxel is apart of a component with it's neighbouring voxels, there is no discrepancy, they either share the the same material ID or they don't. Therefore, the outputs look identical to the inputs and are more useful at an abstract level in the code for the imaging team at Micro-X, such as singling components of a certain material.

One example result is shown in Figure 15. The result shows that all components in the image maintain their shape. This is true regardless of the input. The figure also displays the goal of the algorithm to better show what the algorithm is doing. At the code level, all the voxel values are an integer that is unique to the component it is in.

In Figure 15, it is observed that the corners along the edges that are parallel to baggage scanner cylindrical bin of the resulting image are labelled as separate components. This is because the cylindrical bin touches the edge of the image on each side. This causes each edge section to be disconnected. As expected and also demonstrated in the goal in Figure 15, this causes the CCL algorithms to label them as separate components.

Multiple simulated DECT images were used when testing and comparing the results for the graph in Figure 17. Example inputs are shown in Figure 16. Although that when the CCL algorithms decide whether a voxel belongs in a certain component, there is discrepancy, it was still important to test the implementations using different combinations of components of different shapes and sizes.

4.1.2 Segmentation Time

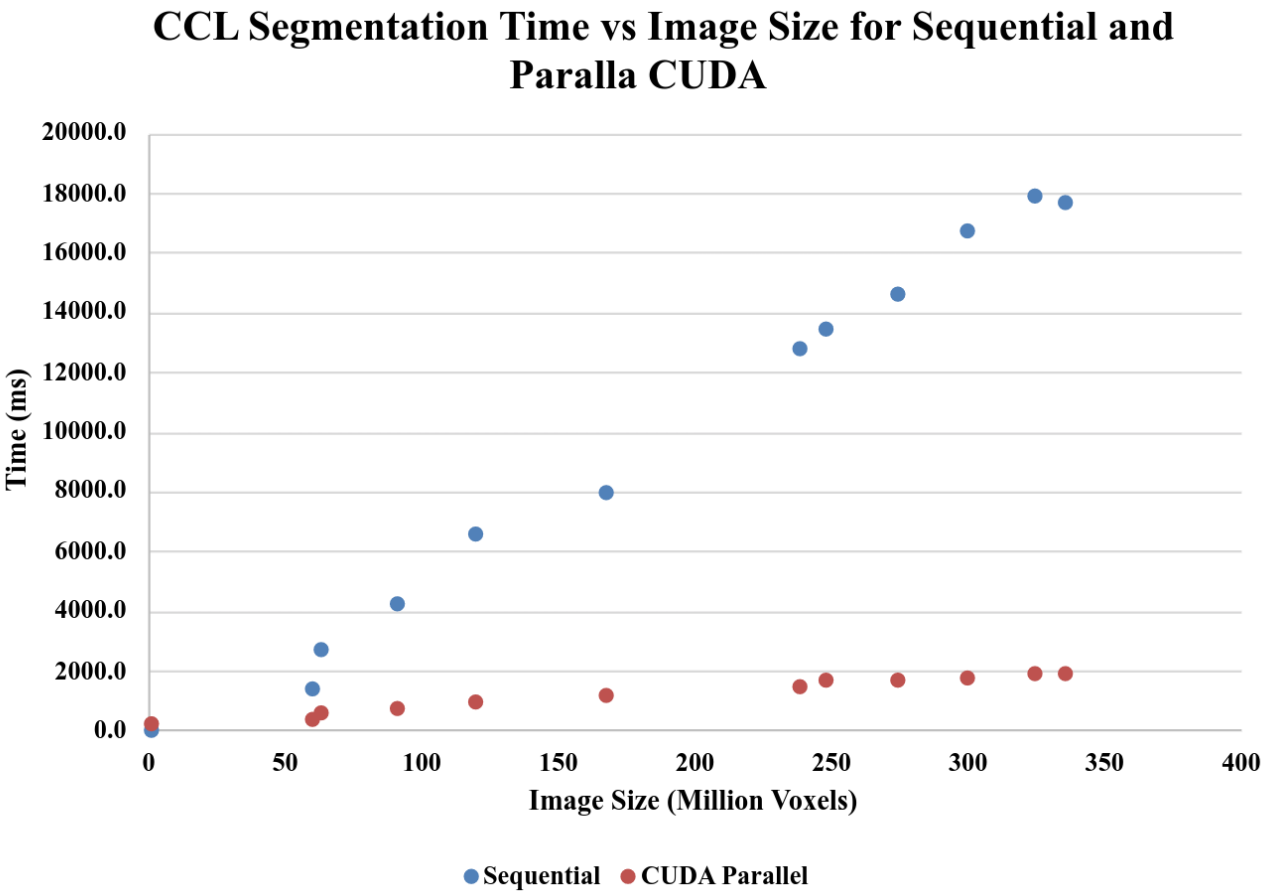


Figure 17: Connect-component labelling segmentation time vs image size for sequential and parallel CUDA.

Observing the graph in Figure 17 shows that the GPU accelerated parallel CUDA CCL implementation has a substantial increase in performance over the non-GPU sequential CCL implementation. The sequential CCL results show that the relationship between the segmentation time and image size is almost linear. The scatter points for the parallel CUDA implementation is also linear. However, the time does not increase drastically. This is expected as the GPU assesses all pixels in parallel using the threads on the processor. The slight increase in the segmentation time is due to the overhead of transferring the DECT image data memory from the host (CPU) to the device (GPU). From Figure 17, it can be seen the the GPU accelerated CCL algorithm has a substantial performance advantage over the sequential algorithm and the larger the DECT material ID image is, the more of an advantage the parallel CUDA CCL algorithm has.

It should be noted for the smallest image size on Figure 17, 970299 ($99 \times 99 \times 99$), the segmentation time for the sequential implementation is actually slightly faster that the GPU CCL. This is due to overhead required by the GPU having a substantial affect on the performance relative to the actual time spent finding components in the image. The next smallest image has a size of 59691250 voxels ($265 \times 265 \times 850$) and the parallel CUDA CCL is 73% faster than the sequential counterpart. This percentage continues to increase. The largest image tested has 335226400 voxels ($628 \times 628 \times 850$) and the parallel CUDA is 89% faster. Furthermore, the most probable DECT scan size at Micro-X

has the same dimensions, $628 \times 628 \times 850$. Each voxel is $1mm^3$. The graph clearly shows that for the larger images, the GPU implementation is much faster.

4.2 Minimal Spanning Tree

The results for MST implementations are split into three categories, results for simulated data, test reconstructions and segmentation time. Results will include a 2D slice from the 3D image underneath.

4.2.1 Simulated Data

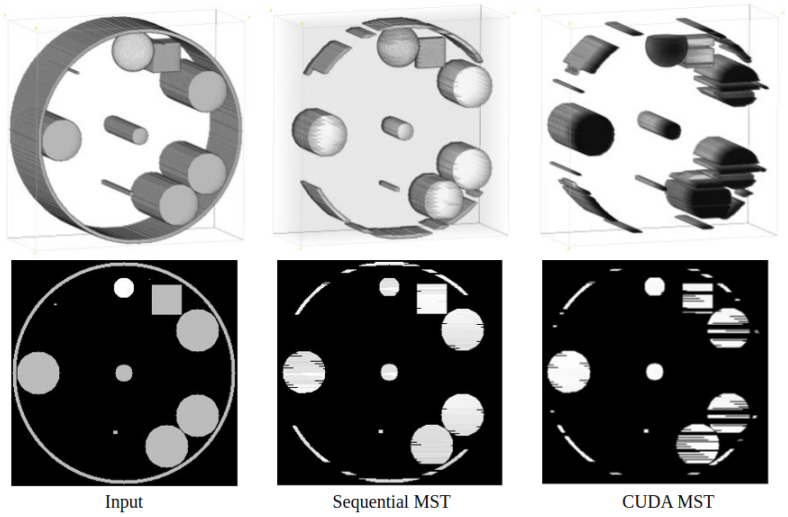


Figure 18: Simulated image sequential and parallel CUDA minimal spanning tree segmentation results (size: $265 \times 265 \times 100$*).*

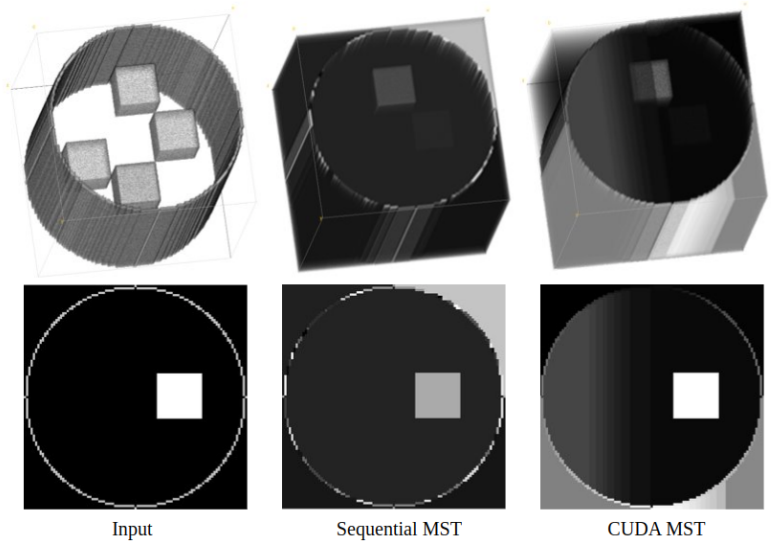


Figure 19: Simulated image sequential and parallel CUDA minimal spanning tree segmentation results (size: $99 \times 99 \times 99$*).*

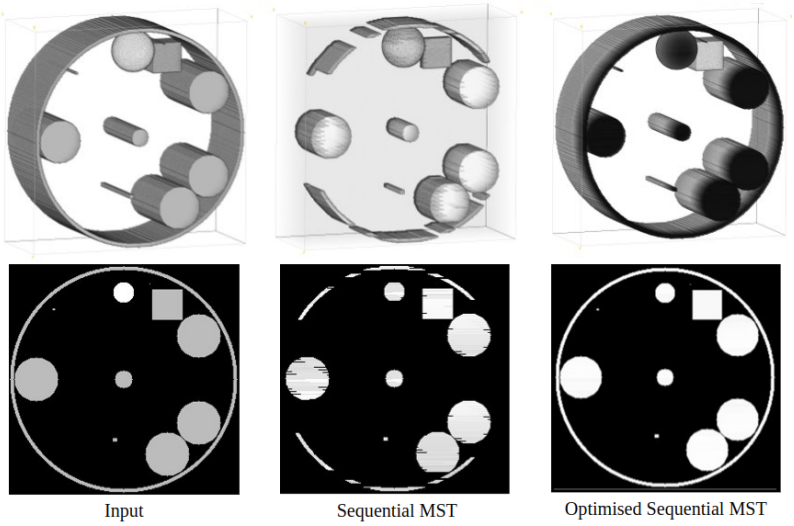


Figure 20: Simulated image sequential and optimised sequential minimal spanning tree segmentation results (size: $265 \times 265 \times 100$*).*

For both the sequential and parallel CUDA MST implementations on the simulated DECT images show information is lost, even despite the images having certain shapes and high contrast edges. These are shown in Figures 18 and 19. In Figure 18, it can be seen that the majority of the outer cylindrical bin and sections in the inner objects are missing. Figure 19 has a better result as the outer cylindrical bin and inner cubes are represented without missing information. However, on the CUDA MST output, the volume space (outside and inside cylindrical bin) is divided up in smaller segments when it should be one segment.

In contrast to the sub-optimal performance from the sequential and parallel CUDA MST, the optimised version of the sequential MST performed substantially better. The test with the simulated data in Figure 20 shows that all information was included in the 3D and 2D slice images, unlike the sequential counterpart. However, it appears all three implementations have a gradient effect on the segmented target objects, which is seen on the 3D representations.

4.2.2 Test Reconstructions

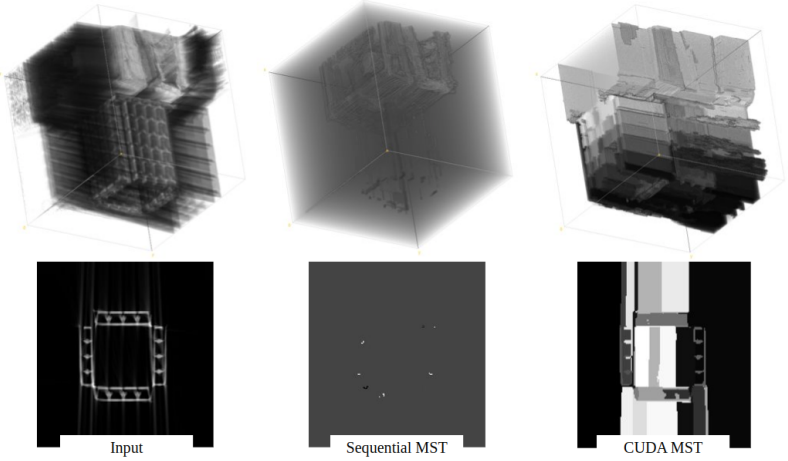


Figure 21: Sequential and parallel CUDA minimal spanning tree segmentation result (size: $200 \times 200 \times 200$*).*

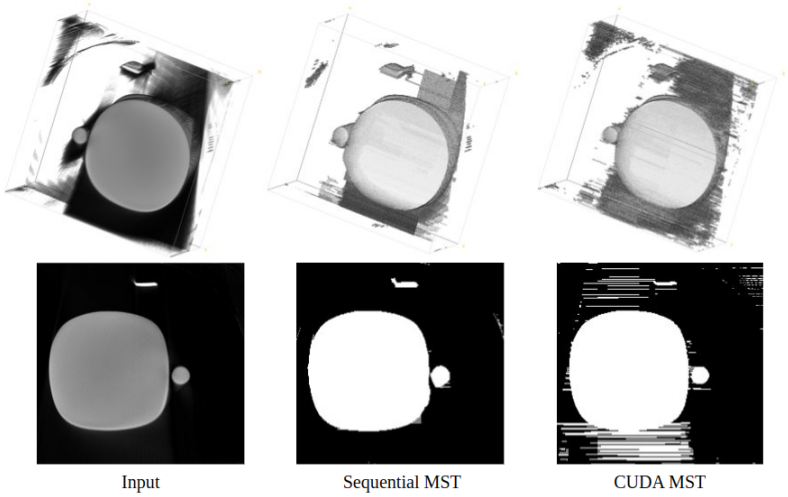


Figure 22: Sequential and parallel CUDA minimal spanning tree segmentation result (size: $250 \times 250 \times 100$*).*

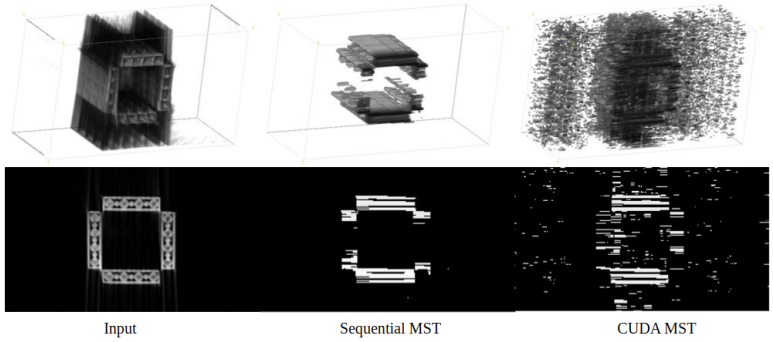


Figure 23: Sequential and parallel CUDA minimal spanning tree segmentation result (size: $280 \times 160 \times 100$*).*

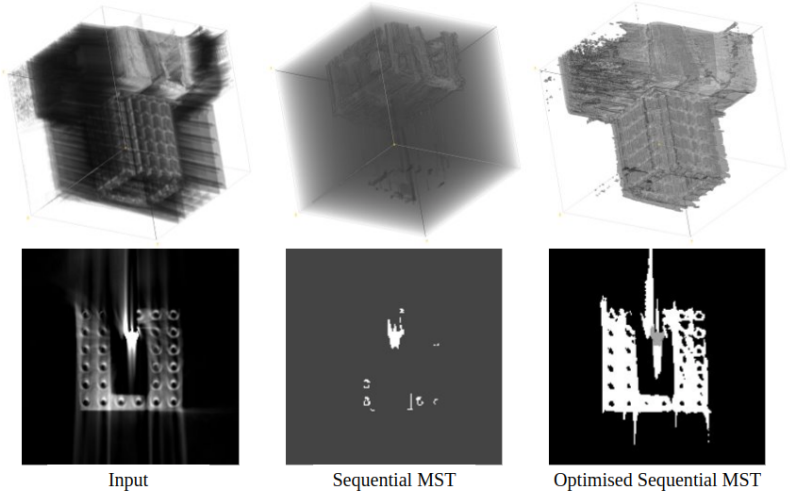


Figure 24: Sequential and optimised sequential minimal spanning tree segmentation results (size: $200 \times 200 \times 200$*).*

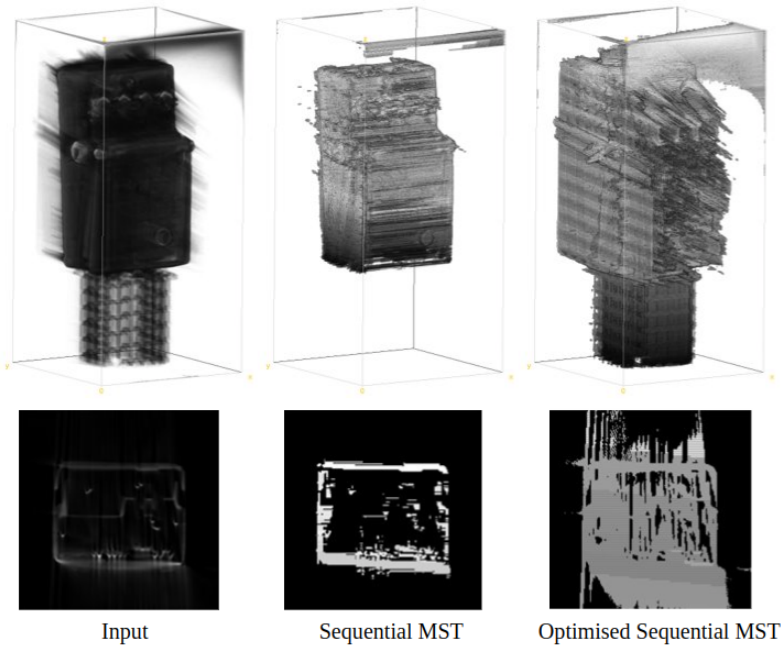


Figure 25: Sequential and optimised sequential minimal spanning tree segmentation results (size: $200 \times 200 \times 420$*).*

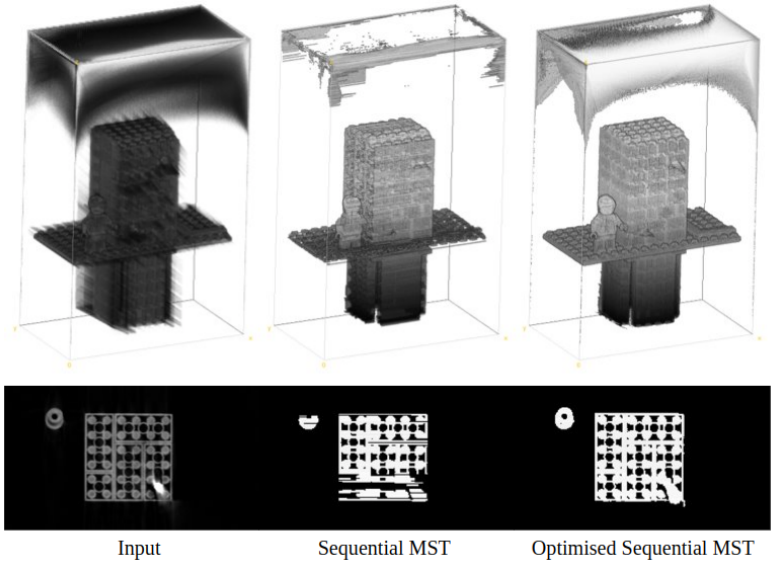


Figure 26: Sequential and optimised sequential minimal spanning tree segmentation results (size: $280 \times 160 \times 448$*).*

The sequential and parallel CUDA MST implementations on the test reconstructions have shown lackluster results. In Figure 21, the sequential MST has omitted the majority of the information defining the shape of the Lego statue. The CUDA MST is really influenced by the streaking and artifacts and includes too much information. Each of the artifacts (which can be see best in the slice) have their own segment in the image. For the CUDA MST, this pattern continues in Figure 22. The sequential MST, however, does a better job at only segmenting the piece of material and does well at omitting just the streaking and artifacts. This is very apparent in the slice of the output. However, in Figure 23, the outputs for both sequential and CUDA MST have similar characteristics to those in Figure 21. The sequential MST fails to include all of the Lego structure, where the CUDA MST includes a lot of the streaking and artifacts. Overall, both the

sequential and CUDA MST have similar lackluster results and have a difficulty segmenting in the presence of the streaking and artifacts.

The same characteristics from the simulated data in Figure 20 continues when the optimised sequential MST was tested on the test reconstructions. All figures (Figures 24, 25 and 26) show that the implementation includes much more information. In Figure 24, the target Lego structure had a metal ball bearing placed inside. This has a higher attenuation then the Acrylonitrile Butadiene Styrene (ABS) plastic. This means it will have a substantially different greyscale value on the CT image. Therefore, it should be a trivial task to be differentiated from each other using segmentation. The optimised sequential MST actually segmented the two materials from each other (ball bearing shown in grey). The regular sequential MST included the ball bearing but lost all information of the Lego. These results are best seen in the slice images in Figure 24.

Furthermore in Figure 25, the sequential MST failed to include any of the Lego stand holding up the target (electronic guitar pedal), whilst the optimised sequential MST include both. However, the optimised sequential MST does include more artifacts and streaking than desired.

The segmentation quality increase when using the optimised implementation is again demonstrated in Figure 26. The regular sequential MST achieves it's best results in this test. However, the optimised sequential MST clearly shows a better image of the Lego structure well outlined in the segmentation result, whereas the regular implementation still has some information missing.

Despite the optimised sequential MST having better segmentations and performing much faster than the regular sequential and parallel CUDA implementations, it falls into the same pattern of the results containing a very evident gradient. This is visible in all of the MST figures. In terms of the segmentation outputs, the gradient throughout the slices means that each slice is a different segment. This bug is a by-product of converting 2D implementations into 3D implementations. This will be noted when handing the toolkit to the imaging team at Micro-X as an issue that would need to be fixed.

4.2.3 Segmentation Time

The graph in Figure 27 shows that despite the parallel CUDA implementation using multi-threading on the GPU to accelerate the algorithm, it still performs worse than the regular sequential MST. The parallel CUDA has an 33% slower average segmentation time. It can be seen that the optimised sequential MST significantly outperforms the regular sequential MST and the parallel CUDA MST. The data shows that the optimised sequential MST has an average segmentation time decrease of 88.5% over the regular sequential MST and an average of 92.8% decrease over the parallel CUDA. For the largest image of 8 million voxels (200x200x200), the optimised sequential variant doesn't pass the 20 second mark. Whereas the sequential and parallel CUDA pass the 1 minute and 2 minute mark, respectively.

The results from the simulated data, reconstruction data and the segmentation time graph clearly show that the best performing implementation is the optimised sequential MST.

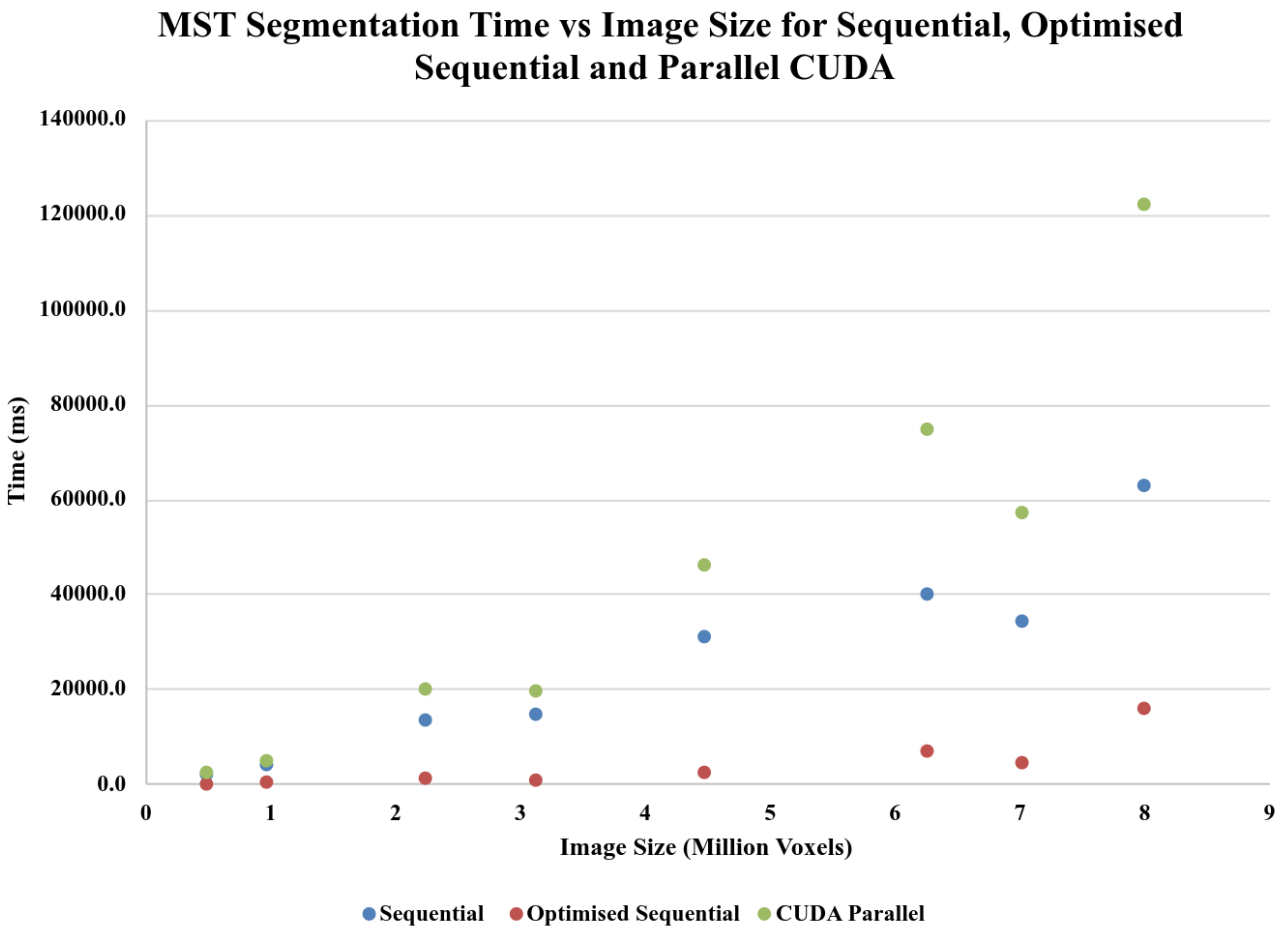


Figure 27: Minimal spanning tree segmentation time vs image size for sequential, optimised sequential and parallel CUDA.

5 | Discussion

In this section, the results from Chapter 4 are compared to literature. Furthermore, the section will discuss the influence GPU optimisation has on the DECT baggage scanner image segmentation toolkit. Finally the limitations will be discussed, which focuses on the hardware and data.

5.1 GPU Optimisation

From the literature review, all studies that utilised the GPU for multi-threaded parallel optimisation were able to achieve a dramatic speed increase. For example Zhuge et al. [25], Dalvand et al. [28] and Sui et al. [27] achieved more than 10× speed improvement with little to no segmentation degrading. Following the pattern from the literature, the parallel CUDA CCL implementation had a dramatic increase in speed as discussed in Section 4.1.2. However, the CUDA MST performed much worse than even the non-optimised sequential MST.

A potential reason for the CUDA MST implementation having worse performance even though it utilises multi-threading is that the implementations are of different algorithms. The base of the CUDA MST algorithm is Boruvka's MST instead of Kruskal's MST. However, this itself would not have a large enough influence to have the CUDA implementation perform significantly slower. The CUDA implementation, which is from Ganin [46], actually analyses twice as many edges as the sequential MST from Motwani [67]. Voxel connectivity (pixel connectivity) is the reason for this. The sequential MST implementation only creates one edge between voxels. It uses the 6-connectivity to find neighbours for voxels. However, the algorithm recognises that if all six possible neighbours are used, there would be two edges between each voxel. For example say voxel a and voxel b are neighbours. If a edge was made for all six of their neighbours, there would be two edges between a and b. Therefore Motwani's implementation only finds three of the six possible neighbours, resulting in only one edge between voxels. Ganin's implementation requires all six of the 6-connectivity neighbours of each voxel to have an edge as the Thrust API functions, such as inclusive scan, require all outgoing edges from each voxel. As there are less edges to connect and merge components together, the result would contain over-segmentation.

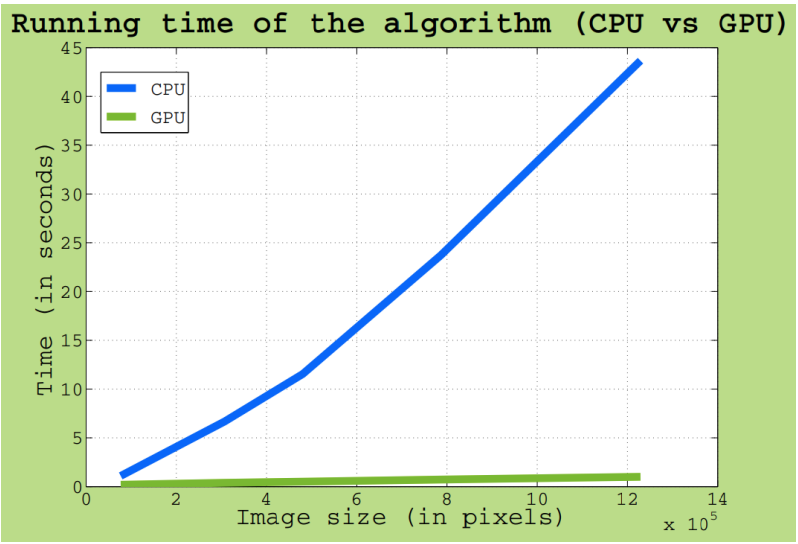


Figure 28: Parallel CUDA minimal spanning tree performance increase graph from Ganin et al. [46]

The performance increase from the parallel CUDA implementation from Playne et al. [34] supports the findings from the corresponding study. However, despite the parallel CUDA MST implementation by Ganin [46] performing slower in the toolkit than the sequential MST, Ganin achieved faster segmentation times than the corresponding sequential counterpart. The graph from Ganin's study is shown in Figure 28. The graph shows the dramatic performance increase from the CUDA MST implementation over the sequential MST. These results may be because the Playne et al. implementation was proposed to be used for both 2D and 3D images. In fact, the open-source code had a 3D version available, which was used for the toolkit. The parallel CUDA implementation by Ganin had to be converted from 2D to 3D, which resulted in more edges per pixel/voxel. Also, the sequential MST counterpart in Ganin's study most likely analysed six edges for every voxel as well, instead of three of the six as Motwani's sequential MST implementation did. A better comparison for Ganin's parallel CUDA MST would be to implement it's exact sequential counterpart. However, for the sake of the toolkit, having Motwani's faster regular and optimised sequential MST implementation is better for the Micro-X imaging team.

5.2 Comparing Segmentation Results to Literature

From the literature review, two stand out studies are going to be compared to the MST results achieved in this thesis. These are the results from Mouton et al. and Wiley et al. [4, 10]. Mouton et al.'s study investigated an initial coarse segmentation for material-based CT baggage scans approach that was used in conjunction with other popular segmentation algorithms. Wiley et al.'s study proposed a CT image segmentation approach from baggage scans as well but was not dual-energy or material-based.

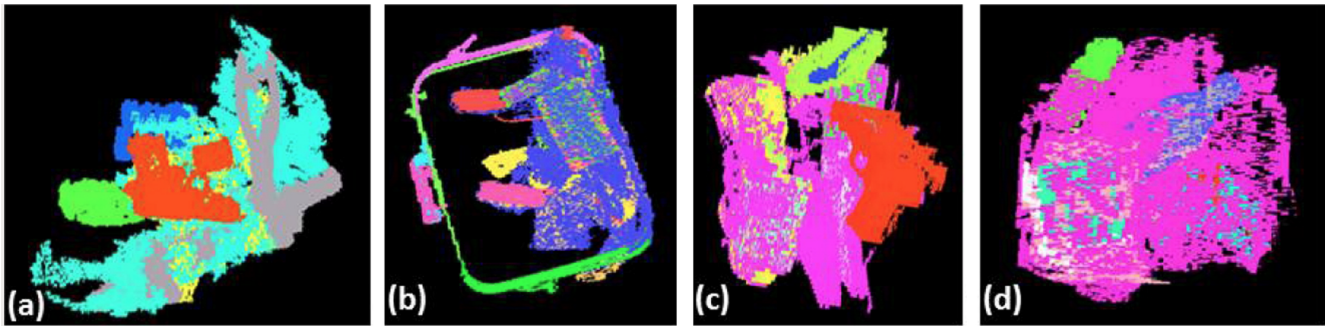


Figure 29: Example CT segmentation from Mouton et al. of two suitcases.
[4]

The segmentation images shown in Figure 29 are from Mouton et al.'s study [4]. These outputs are only a few from many in the study. However, most of the segmentations (which have the most optimal results) from the study were applied to DECT images that had has undergone preprocessing to remove streaking and artifacts. The segmentation outputs in Figure 29 were the result of segmenting images that were not preprocessed, similar to the CT images used in this thesis. The outputs show that the images also lost information during the segmentation. This is shown in the gaps and holes in the objects. Though, Mouton et al.'s methods were able to differentiate multiple objects in the DECT scans. The most differentiating the MST methods

achieved in the toolkit was the metal ball bearing and ABS Lego in Figure 24. However, comparing the performance of the segmentation time, the methods in Mouton et al.'s study almost reached three minutes. All of the optimised sequential MST segmentation times were below 20 seconds. Though, it is unknown what the size of the images were in Mouton et al.'s study.

Wiley et al.'s study achieved high quality segmentation results for their CT baggage images [10]. The example results, shown in Figure 30, show that the segmentation algorithm could successfully segment each of the different parts of the tripod. The candle in the figure shows that there was some over-segmentation as the candle's outer surface has been split up until two parts (shown in green and orange). Nevertheless, the segmentation quality is much better then what was achieved by the MST algorithms in this toolkit. However, the MST algorithms in this toolkit perform much faster. Even the slower parallel CUDA MST, which had it's highest time at around 2 minutes with an image of 8 million voxels, is extremely quick compared to Wiley et al.'s 40 minute average [10].

Figure 30: Example CT segmentation from Wiley et al. of a tripod (left) and candle (right).

[10]

The MST algorithms in this toolkit have all performed faster than the the DECT and CT implementations by Mouton et al. and Wiley et al. [4, 10]. However, the segmentation quality of the targets in those studies is much higher. Although Mouton et al.'s results are affected by artifacts and streaking as well, they still have a higher segmentation quality. It is up to interpretation on which is more important, segmentation speed or quality. However, for the use case of baggage scanner, both are extremely important.

5.3 Limitations

When developing and testing the segmentation toolkit, there were a number of limitations encountered. This section discusses the data and hardware limitations for this project.

5.3.1 Data

Although, the simulated data was an extremely valuable asset when developing and testing the CCL and MST algorithms, the test reconstructions provided by Micro-X left more to be desired. The quality of the reconstructions contained significant amounts of streaking and artifacts in all of the images. These characteristics had an obvious affect on the MST segmentation algorithms. Additionally, only three reconstructions were provided, restricting the depth in which the MST algorithms could be tested. These issues presented a significant limitation when testing the toolkit.

Furthermore, the test reconstructions had not been converted to the material ID values as the

software at Micro-X responsible for this is still in development. This essentially makes the reconstruction images greyscale CT images instead of dual-energy CT images. Therefore, all of the CCL algorithms that relied on the categorical data, could only use the simulated data. This restricts the degree in which the CCL algorithms can be analysed.

A very significant limitation in regards to the data is that there was no ground truth for both the simulated data and the test reconstructions. This resulted in no performance metrics to analyse the quality of the segmentations. Through the literature review, more specifically at Section 2.5.2, it was found that most if not all (for MST at least), studies used some sort of performance metric to assess the quality. This also allows studies to compare the segmentation quality achieved with with studies in the field. An example of possible method of a quantitative analysis to measure segmentation quality are described by Howes et al. [11]. In the study, Howes et al. used pixel-based metrics which include Area-Under-The-Curve (AUC) and Dice Similarity Coefficient (DSC). This allowed comparisons between the proposed methods and other CT segmentation methods in field. The discussion of creating ground truth for CT images prior to testing the segmentation algorithms is discussed in Chapter 7.

Although no quality assessment performance metrics were used, the time to segment the images was measured. In the context of a checkpoint system at an airport, the segmentation time is extremely crucial [5]. Although, the graphs for CCL and MST allowed for a comprehensive comparison between the algorithms within each group, it should be noted that the segmentation time does not entirely dependent on the image size. This is evident from the MST segmentation time graph in Figure 27. All segmentation times for an image with 7 million voxels were slower than an image with 6.25 million voxels when using the same algorithm. As this result is consistent among all of the MST algorithms, it shows that the segmentation time also depends on the characteristics of the image, including image dimensions, component shapes, sizes and the number of components. Despite this, the graphs used in this report show enough information to understand the suitability of each algorithm for the toolkit.

5.3.2 Hardware

Other limitations that were encountered when developing this toolkit were hardware limitations. Micro-X's baggage scanner will have very capable hardware. This includes two NVIDIA RTX A6000 GPUs [72]. These GPUs are intended for professional use cases such as computer-aided design (CAD), computer-generated imagery (CGI), digital content creation (DCC). They have a 48 GB memory capacity each for a total of 96 GB in the Micro-X baggage scanner. The dedicated GPU in the laptop that was used to develop the toolkit has only 4 GB. Not only does the A6000 GPUs have substantially more memory, but that also have much faster computational speeds. Therefore, although fast CCL and MST speeds have been achieved, it should be noted that the speeds that would be a achieved by these algorithms in the context of the baggage scanner would be much faster.

One significant limitation encountered when testing the toolkit, was that many of the images required more memory than was available. This issue would only occur for the parallel CUDA

MST algorithm. All the MST algorithm information, such as the edge and voxel arrays, are stored on the GPU (device) whilst is it executing. Because of this, many of the reconstruction images had to be reduced to a smaller size. This involved removing a section of the image. This is evident in the results, such as Figures 22 and 23. On the other hand, the sequential MST, which executes on the main CPU uses the computer RAM. The laptop used has 32GB of RAM. This means that the sequential (and optimised sequential) MST algorithm was able to segment much larger images. This is shown in Figure 26. The figure shows that the entire object is encapsulated in the image. When comparing the sequential, optimised sequential and parallel CUDA MST algorithm times, the same smaller sized images were used. But when comparing the segmentation quality of the sequential and optimised sequential, the larger full-sized images could be used.

Despite having to use smaller images when testing the MST algorithm because of the GPU size, being able to change the size of the images allowed the algorithms to be compared with more images of different sizes. This is shown on Figure 27. The graph shows more than three image sizes, despite only three reconstruction images provided.

6 | Conclusions

The aim of this industry honours thesis was to create a dual-energy CT image segmentation toolkit for a baggage scanner at Micro-X. The toolkit needed to contain several algorithmic implementations that are appropriate for an airport baggage checkpoint system. This required fast high quality segmentations.

The toolkit developed for this project contains four different algorithms to segment the DECT images provided by Micro-X. Two connected-component labelling algorithms are aimed at the material ID information that can be obtained using the dual-energy technology. The other two algorithms, minimal spanning trees, resort to segmenting the greyscale values of the image, a more common approach in the field. Despite using four algorithms, five different implementations are included in the toolkit as one of the MST algorithms was able to be optimised for the specific use case of a baggage scanner. Within the sets of CCL and MST algorithms, multi-threaded parallel processing using the Graphics Processing Unit was investigated, a technology that will be a available on the Micro-X Miniature Baggage Scanner. Whilst being presented limited data to analyse the implementations, the CCL results support and the MST results contradict the optimisation abilities found in literature when using a GPU. Though, as discussed, there a several possible reasons to why the MST algorithm did not benefit from the GPU. The MST implementations still performed faster than those found in literature, but failed to achieve the same segmentation quality.

The use of dual-energy to determine the material at each CT voxel has many benefits in regards to threat detection in baggage scans. Despite the technology being popular in explosive detection, the literature for segmentation for DECT is scarce in comparison to other segmentation studies for 2D and 3D baggage scans. No studies were found that used the material ID to segment the image creating a novelty aspect to the project as the CCL implementations used the material IDs. This thesis successfully collected and implemented several possible algorithms that Micro-X can use to segment the DECT scans in the baggage scanner. Though the MST results show obvious influence from artifacts, as Micro-X continue to improve their reconstruction images, the MST implementations can be tweaked to better suit the Micro-X imaging team.

7 | Future Work

The work in this thesis shows the potential that using the material ID and connected-component labelling has in terms of segmentation time. GPU optimisation has substantial benefits when using a CCL algorithm. When Micro-X has successfully converted a greyscale CT image to material ID, there is the possibility to use a CCL technique as an initial coarse segmentation prior to using a minimal spanning tree technique which would then use the greyscale information to complete the segmentation. The combination of both the CCL and MST algorithms would allow the CCL algorithm's fast speeds to segment the bulk of the image and then rely on common established segmentation techniques to finish the segmentation. Studies such as Howes et al. [11] successfully used a initial presegmentation on CT images to achieve speed increases, but didn't use CCL as a means of initial segmentation and the CT images were of knee bones rather than airport baggage.

Future work to further investigate dual-energy CT segmentation for the Micro-X baggage scanner include improvements that would minimise the limitations present in this study. Acquiring more DECT images from Micro-X that have both the greyscale and material ID information would allow for a more comprehensive analysis on the algorithms. It would also make it a more feasible option to investigate obtaining or creating ground truth for the images if they have less streaking and artifacts. More improvements and future work include obtaining a reputable dataset such as the ALERT dataset that is specifically designed for DECT segmentation [59]. This would allow segmentation results to be directly compared to results in literature.

This honours project has created a foundation for a future PhD project at Flinders University in collaboration with Micro-X. The project will investigate threat detection and object identification for dual-energy CT scans from the Micro-X baggage scanner. The segmentation toolkit developed in this project can be used to differentiate each object in the scans before using the learning techniques for object identification. More DECT reconstructions will be required from Micro-X, or the ALERT dataset will be obtained.

References

[1] Muhammet Baştan. Multi-View Object Detection in Dual-Energy X-ray Images. *Machine Vision and Applications*, 26(7-8):1045–1060, 11 2015.

[2] Andre Mouton and Toby P. Breckon. A Review of Automated Image Understanding within 3D Baggage Computed Tomography Security Screening. *Journal of X-Ray Science and Technology*, 23(5):531–555, 9 2015.

[3] Limor Martin, Ahmet Tuysuzoglu, W. Clem Karl, and Prakash Ishwar. Learning-Based Object Identification and Segmentation Using Dual-Energy CT Images for Security. *IEEE Transactions on Image Processing*, 24(11):4069–4081, 11 2015.

[4] Andre Mouton and Toby P. Breckon. Materials-Based 3D Segmentation of Unknown Objects from Dual-Energy Computed Tomography Imagery in Baggage Security Screening. *Pattern Recognition*, 48(6):1961–1978, 6 2015.

[5] Sameer Singh and Maneesha Singh. Explosives Detection Systems (EDS) for Aviation Security. *Signal Processing*, 83:31–55, 2003.

[6] Micro-X. Micro-X Miniature Baggage Scanner. https://micro-x.com/products/miniature-baggage-scanner/, Sept 2021. Accessed: 30 Mar 2022.

[7] Caijing Miao, Lingxi Xie, Fang Wan, Chi Su, Hongye Liu, Jianbin Jiao, and Qixiang Ye. SiXray: A Large-scale Security Inspection X-ray Benchmark for Prohibited Item Discovery in Overlapping Images. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:2114–2123, 6 2019.

[8] R. M. Haralick and L. G. Shapiro. Image Segmentation Techniques. *Computer Vision, Graphics, and Image Processing*, 29(1):100–132, 1 1985.

[9] Vojtěch Jarník. On a Certain Problem of Minimization. *Práce Moravské přírodovědecké společnosti 6, fasc. 4*, pages 57–63, 1930.

[10] David Wiley and Deboshmita Ghosh. Automatic Segmentation of CT Scans of Checked Baggage Checkpoint. Technical report, Stratovan, 2012.

[11] Michael Howes, Mariusz Bajger, Gobert Lee, Francesca Bucci, and Saulo Martelli. Texture Enhanced Statistical Region Merging with application to Automatic Knee Bones Segmentationfrom CT. *DICTA 2021 - 2021 International Conference on Digital Image Computing: Techniques and Applications*, 2021.

[12] Transportation Security Administration. TSA awards $781.2 million to procure additional CT X-Ray scanners for airport checkpoints. https://www.tsa.gov/news/press/releases/2022/03/18/tsa-awards-7812-million-procure-additional-ct-x-ray-scanners-airport, March 2022. Accessed: 25 July 2022.

[13] Taimur Hassan, Samet Akçay, Mohammed Bennamoun, Salman Khan, and Naoufel Werghi. Tensor Pooling-Driven Instance Segmentation Framework for Baggage Threat Recognition. *Neural Computing and Applications 2021 34:2*, 34(2):1239–1250, 9 2021.

[14] Domingo Mery, Vladimir Riffo, Uwe Zscherpel, German Mondragón, Iván Lillo, Irene Zuccar, Hans Lobel, and Miguel Carrasco. GDXray: The Database of X-ray Images for Nondestructive Testing. *Journal of Nondestructive Evaluation*, 34(4):1–12, 11 2015.

[15] R. Paranjape, M. Sluser, and K. Runtz. Segmentation of Handguns in Dual Energy X-ray Imagery of Passenger Carry-on Baggage. In *Canadian Conference on Electrical and Computer Engineering*, volume 1, pages 377–380. IEEE, 1998.

[16] Thorsten R.C. Johnson, Bernhard Krauß, Martin Sedlmair, Michael Grasruck, Herbert Bruder, Dominik Morhard, Christian Fink, Sabine Weckbach, Miriam Lenhard, Bernhard Schmidt, Thomas Flohr, Maximilian F. Reiser, and Christoph R. Becker. Material Differentiation by Dual Energy CT: Initial Experience. *European Radiology*, 17(6):1510–1517, 6 2007.

[17] Yuanyuan Liu, Jianping Cheng, Zhiqiang Chen, and Yuxiang Xing. Feasibility Study: Low-Cost Dual Energy CT for Security Inspection. In *IEEE Nuclear Science Symposium Conference Record*, pages 879–882, 2009.

[18] Qian Wang and Toby P. Breckon. Contraband Materials Detection Within Volumetric 3D Computed Tomography Baggage Security Screening Imagery. *Proceedings - 20th IEEE International Conference on Machine Learning and Applications, ICMLA 2021*, pages 75–82, 2021.

[19] Najla Megherbi, Toby P. Breckon, and Greg T. Flitton. Investigating Existing Medical CT Segmentation Techniques within Automated Baggage and Package Inspection. *Optics and Photonics for Counterterrorism, Crime Fighting and Defence IX; and Optical Materials and Biomaterials in Security and Defence Systems Technology X*, 8901:89010L, 10 2013.

[20] Frederick Kelcz, Peter M. Joseph, and Sadek K. Hilal. Noise Considerations in Dual Energy CT Scanning. *Medical Physics*, 6(5):418–425, 9 1979.

[21] Ali Saglam and Nurdan Akhan Baykan. Sequential Image Segmentation based on Minimum Spanning Tree Representation. *Pattern Recognition Letters*, 87:155–162, 2 2017.

[22] Pei-Gee Ho. *Image Segmentation*. InTechOpen, 4 2011.

[23] John Cheng, Max Grossman, and Ty McKercher. *Professional CUDA C Programming*. John Wiley & Sons, 2014.

[24] Denis Demidov, Karsten Ahnert, Karl Rupp, and Peter Gottschling. Programming CUDA and OpenCL: A Case Study Using Modern C++ Libraries. *SIAM Journal on Scientific Computing*, 35(5), 12 2012.

[25] Ying Zhuge, Yong Cao, Jayaram K. Udupa, and Robert W. Miller. Parallel Fuzzy Connected Image Segmentation on GPU. *Medical Physics*, 38(7):4365–4371, 7 2011.

[26] Erik Smistad, Thomas L. Falch, Mohammadmehdi Bozorgi, Anne C. Elster, and Frank Lindseth. Medical Image Segmentation on GPUs – A Comprehensive Review. *Medical Image Analysis*, 20(1):1–18, 2 2015.

[27] Haigang Sui, Feifei Peng, Chuan Xu, Kaimin Sun, and Jianya Gong. GPU-accelerated MRF Segmentation Algorithm for SAR Images. *Computers & Geosciences*, 43:159–166, 6 2012.

[28] Mehran Dalvand, Abdolhossein Fathi, and Arezoo Kamran. Flooding Region Growing: A New Parallel Image Segmentation Model based on Membrane Computing. *Journal of Real-Time Image Processing*, 18(1):37–55, 2 2021.

[29] Patrick Nigri Happ, Raul Queiroz Feitosa, Cristiana Bentes, and Ricardo Farias. A Region-Growing Segmentation Algorithm for GPUs. *IEEE Geoscience and Remote Sensing Letters*, 10(6):1612–1616, 2013.

[30] Vibhav Vineet, Pawan Harish, Suryakant Patidar, and P Narayanan. Fast Minimum Spanning Tree for Large Graphs on the GPU. In *Proceedings of the Conference on high performance graphics 2009*, HPG '09, pages 167–171. ACM, 2009.

[31] Seongjin Park, Jeongjin Lee, Hyunna Lee, Juneseuk Shin, Jinwook Seo, Kyoung Ho Lee, Yeong Gil Shin, and Bohyoung Kim. Parallelized Seeded Region Growing using CUDA. *Computational and Mathematical Methods in Medicine*, 2014, 2014.

[32] Stefano Allegretti, Federico Bolelli, Michele Cancilla, and Costantino Grana. Optimizing GPU-Based Connected Components Labeling Algorithms. *IEEE 3rd International Conference on Image Processing, Applications and Systems, IPAS 2018*, pages 175–180, 7 2018.

[33] Stefano Allegretti, Federico Bolelli, and Costantino Grana. Optimized Block-Based Algorithms to Label Connected Components on GPUs. *IEEE Transactions on Parallel and Distributed Systems*, 31(2):423–438, 2 2020.

[34] Daniel Peter Playne and Ken Hawick. A New Algorithm for Parallel Connected-Component Labelling on GPUs. *IEEE Transactions on Parallel and Distributed Systems*, 29(6):1217–1230, 6 2018.

[35] Michael B. Dillencourt, Hanan Samet, and Markku Tamminen. A General Approach to Connected-Component Labeling for Arbitrary Image Representations. *Journal of the ACM (JACM)*, 39(2):253–280, 4 1992.

[36] Niu Lianqiang, Chen Xin, Peng Min, and Zhang Gang. Connected Components Labeling based on Union-Find Operations applied to Connected Branches. *Journal of intelligent & fuzzy systems*, 32(5):3739–3748, 2017.

[37] Lifeng He, Yuyan Chao, Kenji Suzuki, and Kesheng Wu. Fast Connected-Component Labeling. *Pattern recognition*, 42(9):1977–1987, 2009.

[38] Lifeng He, Xiwei Ren, Qihang Gao, Xiao Zhao, Bin Yao, and Yuyan Chao. The Connected-Component Labeling Problem: A Review of State-of-the-Art Algorithms. *Pattern Recognition*, 70:25–43, 10 2017.

[39] Yisleidy Linares Zaila, Marta L.Baguer Díaz-Romañach, and Manuel González-Hidalgo. A Graph Based Segmentation Strategy for Baggage Scanner Images. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8563 LNCS:81–93, 2014.

[40] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2):167–181, 9 2004.

[41] Jan Wassenberg, Wolfgang Middelmann, and Peter Sanders. An Efficient Parallel Algorithm for Graph-Based Image Segmentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5702 LNCS:1003–1010, 2009.

[42] Maksim Privalov and Maksim Kazantsev. CT Images Segmentation Algorithm Based on Graph Cuts. *2018 International Multi-Conference on Industrial Engineering and Modern Technologies, FarEastCon 2018*, 2018.

[43] R. C. Prim. Shortest Connection Networks And Some Generalizations. *Bell System Technical Journal*, 36(6):1389–1401, 1957.

[44] Joseph B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48, 2 1956.

[45] Guan Wei Wang, Chun Xia Zhang, and Jian Zhuang. Clustering with Prim's Sequential representation of Minimum Spanning Tree. *Applied Mathematics and Computation*, 247:521–534, 11 2014.

[46] NVIDIA Yaroslav Ganin. Efficient Segmentation Tress on the GPU. https://on-demand.gputechconf.com/gtc/2009/posters/P0496_Efficient_Segmentation_Trees_by_Ganin.pdf, 2012. Accessed: 15 Sept 2022.

[47] Yll Haxhimusa and Walter Kropatsch. Segmentation Graph Hierarchies. In *Structural, Syntactic, and Statistical Pattern Recognition*, pages 343–351, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[48] Lihong Guo, Yueyun Liu, Yu Wang, Yuping Duan, and Xue-Cheng Tai. Learned Snakes for 3D Image Segmentation. *Signal Processing*, 183:108013, 2021.

[49] Tianwei Ni, Lingxi Xie, Huangjie Zheng, Elliot K. Fishman, and Alan L. Yuille. Elastic Boundary Projection for 3D Medical Image Segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:2104–2113, 12 2018.

[50] Zhengqin Li and Jiansheng Chen. Superpixel Segmentation Using Linear Spectral Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2015.

[51] Xujiong Ye, Gareth Beddoe, and Greg Slabaugh. Automatic Graph Cut Segmentation of Lesions in CT using Mean Shift Superpixels. *International Journal of Biomedical Imaging*, 2010, 2010.

[52] Amal Farag, Le Lu, Holger R. Roth, Jiamin Liu, Evrim Turkbey, and Ronald M. Summers. A Bottom-Up Approach for Pancreas Segmentation Using Cascaded Superpixels and (Deep) Image Patch Labeling. *IEEE Transactions on Image Processing*, 26(1):386–399, 1 2017.

[53] Shu Yen Wan and William E. Higgins. Symmetric Region Growing. *IEEE Transactions on Image Processing*, 12(9):1007–1015, 9 2003.

[54] Zheng Lin, Jesse Jin, and Hugues Talbot. Unseeded Region Growing for 3D Image Segmentation. *VIP '00: Selected papers from the Pan-Sydney workshop on Visualisation*, 02:31–37, 2000.

[55] Yu Ke Chen, Xiao Ming Wu, Ken Cai, and Shan Xin Ou. CT Image Segmentation based on Clustering and Graph-Cuts. *Procedia Engineering*, 15:5179–5184, 2011.

[56] Matthias Krueger, Patrice Delmas, and Georgy Gimel'Farb. Active Contour Based Segmentation of 3D Surfaces. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5303 LNCS(PART 2):350–363, 2008.

[57] Leo Grady, Vivek Singh, Timo Kohlberger, Christopher Alvino, and Claus Bahlmann. Automatic Segmentation of Unknown Objects, with application to Baggage Security. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7573 LNCS(PART 2):430–444, 2012.

[58] Jianli Ding, Yuanxiang Li, Xing Xu, and Lingling Wang. X-ray Image Segmentation by Attribute Relational Graph Matching. *International Conference on Signal Processing Proceedings, ICSP*, 2, 2006.

[59] Carl Crawford. Awareness and Localization of Security-Related Threats (ALERT): Segmentation of Objects from Volumetric CT Data. Technical report, DHS Center of Excellence at Northeastern University, Boston, Massachusetts, 12 2011.

[60] Parisa Babaheidarian and David Castañón. Joint Segmentation and Material Recognition in Dual-Energy CT Images. In *IS and T International Symposium on Electronic Imaging Science and Technology*, pages 30–35. Society for Imaging Science and Technology, 2017.

[61] Qian Wang and Toby P. Breckon. On the Evaluation of Semi-Supervised 2D Segmentation for Volumetric 3D Computed Tomography Baggage Security Screening. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2021-July. Institute of Electrical and Electronics Engineers Inc., 7 2021.

[62] Seemeen Karimi, Xiaoqian Jiang, Pamela Cosman, and Harry Martz. Flexible Methods for Segmentation Evaluation: Results from CT-based Luggage Screening. *Journal of X-Ray Science and Technology*, 22(2):175–195, 2014.

[63] Andre Mouton, Toby P. Breckon, Greg T. Flitton, and Najla Megherbi. 3D Object Classification in Baggage Computed Tomography Imagery using Randomised Clustering Forests. In *2014 IEEE International Conference on Image Processing, ICIP 2014*, pages 5202–5206. Institute of Electrical and Electronics Engineers Inc., 1 2014.

[64] Taimur Hassan, Samet Akçay, Mohammed Bennamoun, Salman Khan, and Naoufel Werghi. Unsupervised Anomaly Instance Segmentation for Baggage Threat Recognition. *Journal of Ambient Intelligence and Humanized Computing*, 1:1–12, 7 2021.

[65] Luc Vincent, Luc Vincent, and Pierre Soille. Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.

[66] Wikipedia. Connected-Component Labelling. https://en.wikipedia.org/wiki/Connected-component_labeling, Sept 2022. Accessed: 15 Sept 2022.

[67] Mohit Motwani. Implementing Efficient Graph Based Image Segmentation with C++. https://iammohitm.github.io/Graph-Based-Image-Segmentation/, 2020. Access: 15 Sept 2022.

[68] Otakar Boruvka. About a Certain Minimal Problem. *Práce Moravské přírodovědecké společnosti 5, fasc. 3*, pages 37–58, 1926.

[69] Bjarne Stroustrup. *A Tour of C++*. C++ In-Depth Series. Addison-Wesley, Upper Saddle River, NJ, 2014.

[70] International Organization for Standardization. ISO/IEC 14882:2003 Programming languages — C++. https://www.iso.org/standard/38110.html, 2003. Access: 15 Sept 2022.

[71] National Institute of Standards and Technology. X-Ray Mass Attenuation Coefficients. https://www.nist.gov/pml/x-ray-mass-attenuation-coefficients, 2004. Access: 15 Sept 2022.

[72] NIVIDA. NVIDIA RTX 6000 Ada Generation Graphics Card. https://www.nvidia.com/en-au/design-visualization/rtx-6000/, 2022. Access: 15 Sept 2022.

[73] Khaled Y Elbanna, · Bahar Mansoori, · Achille Mileto, Patrik Rogalla, · Luís, and S Guimarães. Dual-Energy CT in Diffuse Liver Disease: Is there a Role? *Abdominal Radiology*, 45:3413–3424, 2020.

A | Segmentation Time Data

	Dimensions				Times (ms)		
	X	Y	Z	Total	Sequential	Optimised Sequential	Parallel CUDA
Test 1	99	99	50	490050	2004.0	129.0	2246.0
	99	99	99	970299	3745.0	331.0	3992.0
	200	200	200	8000000	64282.0	16361.0	122049.0
	250	250	50	3125000	14919.0	1013.0	19738.0
	250	250	100	6250000	40855.0	6830.0	74744.0
	265	265	100	7022500	32961.0	4208.0	57292.0
	280	160	50	2240000	13324.0	995.0	19555.0
	280	160	100	4480000	31983.0	2520.0	46170.0
Test 2	99	99	50	490050	2136.0	137.0	3062.0
	99	99	99	970299	3845.0	329.0	5424.0
	200	200	200	8000000	61475.0	15701.0	121999.0
	250	250	50	3125000	14710.0	944.0	19717.0
	250	250	100	6250000	37381.0	6745.0	74756.0
	265	265	100	7022500	35570.0	4097.0	57304.0
	280	160	50	2240000	13550.0	1000.0	20039.0
	280	160	100	4480000	31335.0	2462.0	46045.0
Test 3	99	99	50	490050	1987.0	126.0	2284.0
	99	99	99	970299	3993.0	335.0	4485.0
	200	200	200	8000000	62077.0	15493.0	122046.0
	250	250	50	3125000	13703.0	931.0	19728.0
	250	250	100	6250000	40525.0	6902.0	74751.0
	265	265	100	7022500	35841.0	4465.0	57323.0
	280	160	50	2240000	13561.0	1115.0	20042.0
	280	160	100	4480000	30726.0	2411.0	46157.0
Test 4	99	99	50	490050	2293.0	143.0	3084.0
	99	99	99	970299	3895.0	331.0	5432.0
	200	200	200	8000000	65305.0	17035.0	122038.0
	250	250	50	3125000	14207.0	950.0	19744.0
	250	250	100	6250000	39755.0	7361.0	74734.0
	265	265	100	7022500	32776.0	4250.0	57279.0
	280	160	50	2240000	13465.0	958.0	19959.0
	280	160	100	4480000	30630.0	2363.0	46035.0
Test 5	99	99	50	490050	1970.0	133.0	2327.0
	99	99	99	970299	3871.0	333.0	5099.0
	200	200	200	8000000	61355.0	15639.0	122043.0
	250	250	50	3125000	14898.0	982.0	19723.0
	250	250	100	6250000	42155.0	6611.0	74758.0
	265	265	100	7022500	33436.0	4349.0	57279.0
	280	160	50	2240000	13944.0	958.0	20024.0
	280	160	100	4480000	31278.0	2411.0	46132.0
Average	99	99	50	490050	2078.0	133.6	2600.6
	99	99	99	970299	3869.8	331.8	4886.4
	200	200	200	8000000	62898.8	16045.8	122035.0
	250	250	50	3125000	14487.4	964.0	19730.0
	250	250	100	6250000	40134.2	6889.8	74748.6
	265	265	100	7022500	34116.8	4273.8	57295.4
	280	160	50	2240000	13568.8	1005.2	19923.8
	280	160	100	4480000	31190.4	2433.4	46107.8

Table 2: Minimal spanning tree segmentation time results.

	Dimensions					Times (ms)			
	# Components	X	Y	Z	Total	Sequential - 6	Sequential - 18	Sequential - 26	Parallel CUDA
Test 1	84	99	99	99	970299	13.0	22.0	31.0	219.0
	1337	265	265	850	59691250	1351.0	2034.0	2522.0	360.0
	13	530	530	225	63202500	2722.0	4099.0	5027.0	589.0
	134	530	530	325	91292500	4026.0	6002.0	7301.0	750.0
	735	530	530	425	119382500	6303.0	9698.0	11739.0	841.0
	1337	530	530	850	238765000	12320.0	19202.0	22063.0	1378.0
	1337	568	568	850	274230400	14128.0	21586.0	25266.0	1667.0
	1337	594	594	850	299910600	16716.0	24252.0	28892.0	1769.0
	1337	618	618	850	324635400	17015.0	26349.0	31444.0	1853.0
	735	628	628	425	167613200	7833.0	11841.0	14235.0	1107.0
	1337	628	628	628	247673152	13450.0	19460.0	23144.0	1692.0
	1337	628	628	850	335226400	17535.0	26071.0	30342.0	1936.0
Test 2	84	99	99	99	970299	22.0	22.0	26.0	220.0
	1337	265	265	850	59691250	1412.0	2215.0	2775.0	369.0
	13	530	530	225	63202500	2700.0	4264.0	5014.0	588.0
	134	530	530	325	91292500	4348.0	6372.0	7813.0	743.0
	735	530	530	425	119382500	6432.0	9948.0	11391.0	918.0
	1337	530	530	850	238765000	13139.0	19348.0	20493.0	1455.0
	1337	568	568	850	274230400	14394.0	21679.0	25376.0	1567.0
	1337	594	594	850	299910600	16658.0	24549.0	27325.0	1735.0
	1337	618	618	850	324635400	17500.0	26219.0	29657.0	1863.0
	735	628	628	425	167613200	8148.0	12230.0	14768.0	1248.0
	1337	628	628	628	247673152	12839.0	19263.0	22684.0	1643.0
	1337	628	628	850	335226400	17361.0	25038.0	29966.0	1969.0
Test 3	84	99	99	99	970299	15.0	26.0	31.0	227.0
	1337	265	265	850	59691250	1453.0	2111.0	2603.0	370.0
	13	530	530	225	63202500	2693.0	4041.0	4966.0	580.0
	134	530	530	325	91292500	4287.0	6456.0	7940.0	779.0
	735	530	530	425	119382500	6323.0	9403.0	11954.0	921.0
	1337	530	530	850	238765000	12577.0	20013.0	22400.0	1481.0
	1337	568	568	850	274230400	14046.0	20628.0	25274.0	1623.0
	1337	594	594	850	299910600	16551.0	24131.0	28172.0	1775.0
	1337	618	618	850	324635400	17956.0	25776.0	31055.0	1840.0
	735	628	628	425	167613200	8061.0	12013.0	14530.0	1228.0
	1337	628	628	628	247673152	13758.0	19335.0	22652.0	1675.0
	1337	628	628	850	335226400	17882.0	26251.0	30952.0	1963.0
Test 4	84	99	99	99	970299	15.0	23.0	31.0	236.0
	1337	265	265	850	59691250	1394.0	2298.0	2817.0	390.0
	13	530	530	225	63202500	2693.0	4194.0	5076.0	592.0
	134	530	530	325	91292500	4177.0	6169.0	7538.0	749.0
	735	530	530	425	119382500	6104.0	10033.0	12216.0	915.0
	1337	530	530	850	238765000	12982.0	19501.0	23027.0	1399.0
	1337	568	568	850	274230400	15200.0	22172.0	24490.0	1579.0
	1337	594	594	850	299910600	16432.0	23622.0	28787.0	1803.0
	1337	618	618	850	324635400	19624.0	26408.0	31419.0	1934.0
	735	628	628	425	167613200	7838.0	11577.0	14188.0	1198.0
	1337	628	628	628	247673152	13643.0	18881.0	22876.0	1693.0
	1337	628	628	850	335226400	18286.0	24258.0	28549.0	1929.0
Test 5	84	99	99	99	970299	13.0	21.0	24.0	224.0
	1337	265	265	850	59691250	1451.0	2063.0	2606.0	367.0
	13	530	530	225	63202500	2789.0	4100.0	5018.0	578.0
	134	530	530	325	91292500	4212.0	6153.0	7507.0	741.0
	735	530	530	425	119382500	6082.0	9353.0	11597.0	967.0
	1337	530	530	850	238765000	12956.0	18226.0	22740.0	1387.0
	1337	568	568	850	274230400	15421.0	20943.0	24914.0	1911.0
	1337	594	594	850	299910600	17160.0	24165.0	29914.0	1796.0
	1337	618	618	850	324635400	17284.0	24182.0	30006.0	1922.0
	735	628	628	425	167613200	8067.0	12301.0	14829.0	1187.0
	1337	628	628	628	247673152	13341.0	19572.0	23385.0	1733.0
	1337	628	628	850	335226400	17132.0	24172.0	29639.0	1841.0
Average	84	99	99	99	970299	15.6	22.8	28.6	225.2
	1337	265	265	850	59691250	1412.2	2144.2	2664.6	371.2
	13	530	530	225	63202500	2719.4	4139.6	5020.2	585.4
	134	530	530	325	91292500	4210.0	6230.4	7619.8	752.4
	735	530	530	425	119382500	6592.0	10143.4	12454.8	978.4
	1337	530	530	850	238765000	12794.8	19258.0	22144.6	1420.0
	1337	568	568	850	274230400	14637.8	21401.6	25064.0	1669.4
	1337	594	594	850	299910600	16703.4	24143.8	28618.0	1775.6
	1337	618	618	850	324635400	17875.8	25786.8	30716.2	1882.4
	735	628	628	425	167613200	7989.4	11992.4	14510.0	1193.6
	1337	628	628	628	247673152	13406.2	19302.2	22948.2	1687.2
	1337	628	628	850	335226400	17639.2	25158.0	29889.6	1927.6

Table 3: Connected-component labelling segmentation time results.

Note all connectivities are included (6, 18 and 26) for sequential. 6-Connectivity was used when comparing to parallel CUDA.

B | Material Chemical Composition from Dual-Energy

The dual-energy information in the CT scan allows the chemical composition of the target material to be discovered. A useful graph from Elbanna et al. [73] demonstrates how the graphs developed from different energy levels differentiate the two materials. The graph, in Figure 31, shows the X-Ray absorption for Iodine and Bone. Both of these materials have different X-Ray absorption levels are 80kV and 120 kV. The unique graph for each material can be identified with only the two energies. The use of a lookup table, called a Dual-Energy Index (DEI), can provide the material identification given an the X-Ray absorption at two different energy levels. Example of a lookup table from the National Institute of Standards and Technology (NIST) shown in Figure 32 [71].

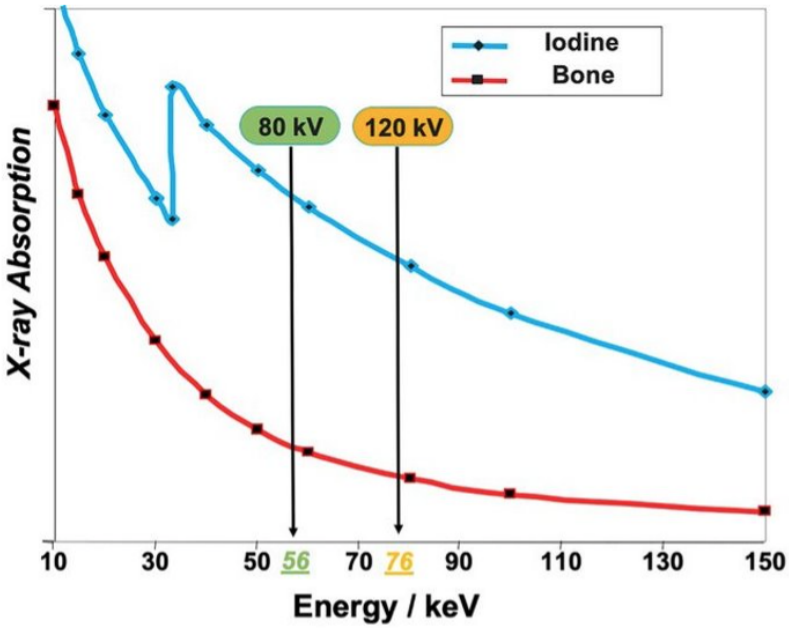
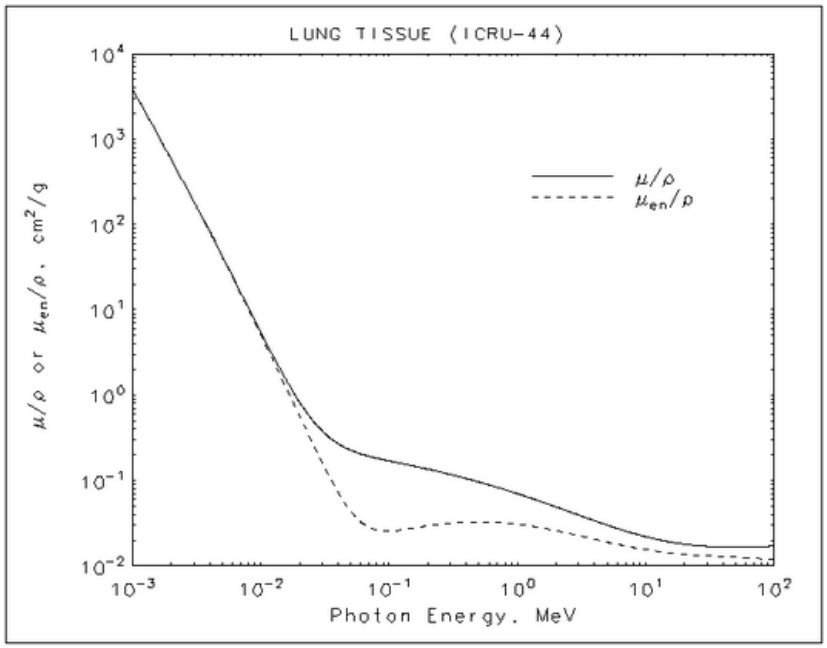


Figure 31: X-Ray absorption vs energy keV for iodine and bone from Elbanna et al. [73]



Lung Tissue (ICRU-44)
HTML table format

	Energy (MeV)	μ/ρ (cm^2/g)	μ_{en}/ρ (cm^2/g)
	1.00000E-03	3.803E+03	3.791E+03
	1.03542E-03	3.469E+03	3.459E+03
	1.07210E-03	3.164E+03	3.155E+03
11 K	1.07210E-03	3.176E+03	3.167E+03
	1.50000E-03	1.283E+03	1.280E+03
	2.00000E-03	5.746E+02	5.727E+02
	2.14550E-03	4.707E+02	4.690E+02
15 K	2.14550E-03	4.752E+02	4.732E+02
	2.30297E-03	3.883E+02	3.865E+02
	2.47200E-03	3.170E+02	3.154E+02
16 K	2.47200E-03	3.226E+02	3.206E+02
	2.64140E-03	2.668E+02	2.650E+02
	2.82240E-03	2.204E+02	2.189E+02
17 K	2.82240E-03	2.248E+02	2.229E+02
	3.00000E-03	1.888E+02	1.870E+02
	3.60740E-03	1.103E+02	1.091E+02
19 K	3.60740E-03	1.125E+02	1.110E+02
	4.00000E-03	8.306E+01	8.181E+01
	5.00000E-03	4.296E+01	4.210E+01
	6.00000E-03	2.497E+01	2.431E+01
	8.00000E-03	1.058E+01	1.010E+01
	1.00000E-02	5.459E+00	5.067E+00

Lung Tissue (ICRU-44)
ASCII format

	Energy (MeV)	μ/ρ (cm^2/g)	μ_{en}/ρ (cm^2/g)
	1.00000E-03	3.803E+03	3.791E+03
	1.03542E-03	3.469E+03	3.459E+03
	1.07210E-03	3.164E+03	3.155E+03
11 K	1.07210E-03	3.176E+03	3.167E+03
	1.50000E-03	1.283E+03	1.280E+03
	2.00000E-03	5.746E+02	5.727E+02
	2.14550E-03	4.707E+02	4.690E+02
15 K	2.14550E-03	4.752E+02	4.732E+02
	2.30297E-03	3.883E+02	3.865E+02
	2.47200E-03	3.170E+02	3.154E+02
16 K	2.47200E-03	3.226E+02	3.206E+02
	2.64140E-03	2.668E+02	2.650E+02
	2.82240E-03	2.204E+02	2.189E+02
17 K	2.82240E-03	2.248E+02	2.229E+02
	3.00000E-03	1.888E+02	1.870E+02
	3.60740E-03	1.103E+02	1.091E+02
19 K	3.60740E-03	1.125E+02	1.110E+02
	4.00000E-03	8.306E+01	8.181E+01
	5.00000E-03	4.296E+01	4.210E+01
	6.00000E-03	2.497E+01	2.431E+01
	8.00000E-03	1.058E+01	1.010E+01
	1.00000E-02	5.459E+00	5.067E+00
	1.50000E-02	1.721E+00	1.423E+00
	2.00000E-02	8.316E-01	5.740E-01
	3.00000E-02	3.815E-01	1.635E-01
	4.00000E-02	2.699E-01	7.286E-02
	5.00000E-02	2.270E-01	4.393E-02
	6.00000E-02	2.053E-01	3.282E-02
	8.00000E-02	1.826E-01	2.625E-02
	1.00000E-01	1.695E-01	2.550E-02
	1.50000E-01	1.493E-01	2.748E-02
	2.00000E-01	1.359E-01	2.945E-02
	3.00000E-01	1.177E-01	3.167E-02

Figure 32: Example Dual-Energy Index (DEI) lookup table for Lung Tissue from NIST. [71]

C | Gantt Chart

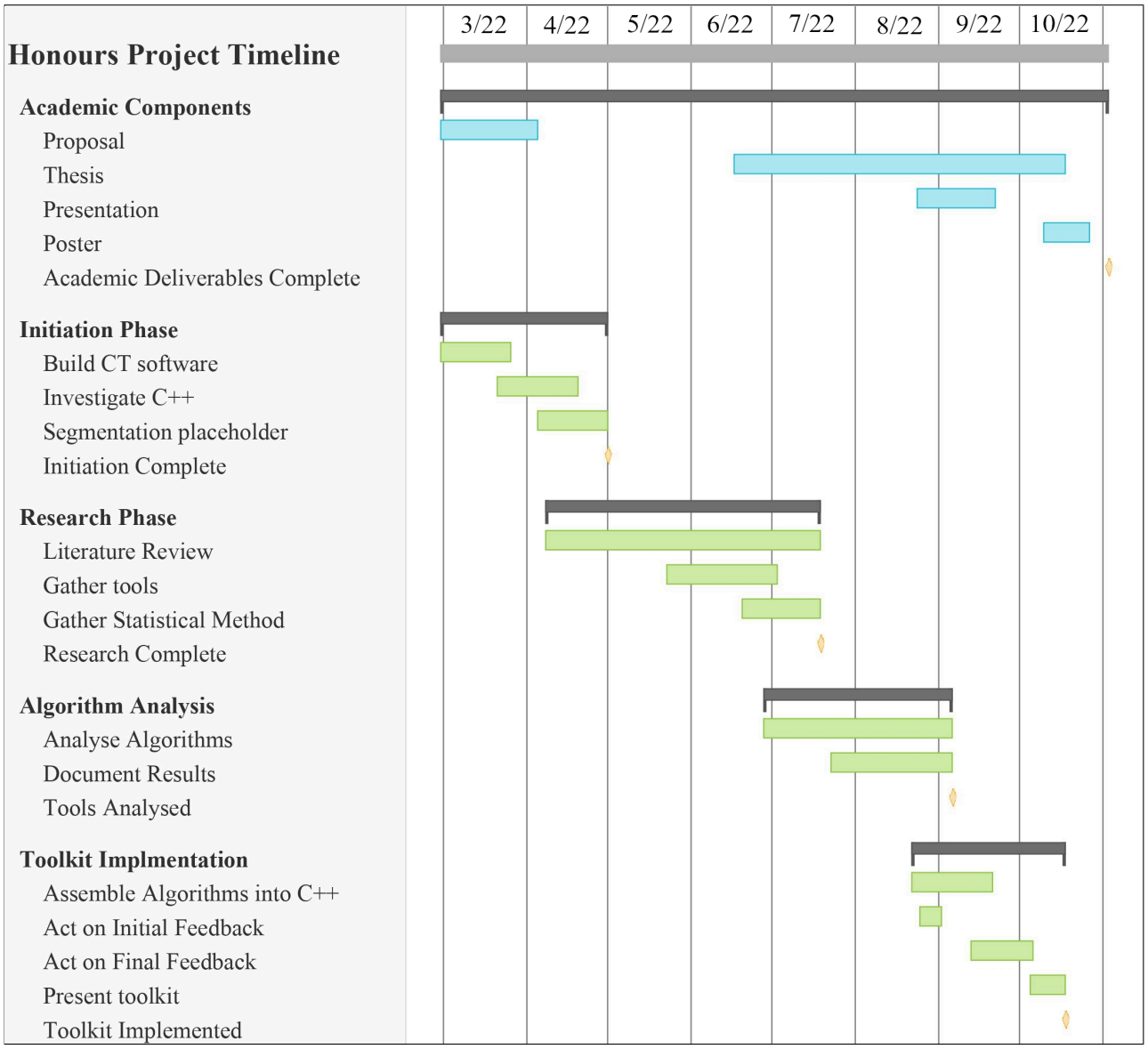


Figure 33: Project Gantt chart.