INAUGURAL – DISSERTATION

submitted to the

Combined Faculty of Mathematics, Engineering and Natural Sciences

of

Ruprecht–Karls–University

Heidelberg

for the degree of

Doctor of Natural Sciences

put forward by

Vahdaneh Kiani, M.Sc.

from

Tehran, Iran

Date of oral exam:.....

Deformable Image Registration for Image-Guided Adaptive Radiation Therapy (IGART) Based on Massive Parallelism and Real-Time Scheduling

Advisors: Professor Dr. Holger Fröning, Professor Dr. Oliver Jäkel

To my mother.

Acknowledgements

This work would not have been possible without the support and encouragement of many wonderful individuals.

I am deeply indebted to my mother for her consistent support and encouragement, which kept me going even though we were miles away from each other. Her journey to Germany, overcoming four years of visa obstacles, brought much-needed solace and inspiration during my most challenging times. Her endless love and support are the foundation of all my achievements.

Words cannot express my gratitude to my primary advisor, Prof. Dr. Holger Fröning, for his unwavering guidance and support throughout my research period. His mentorship was invaluable in navigating the challenges of my work, offering both scientific and personal assistance. From a scientific perspective, our meetings and conversations were instrumental in providing inspiration and encouragement, and his enthusiasm in assisting with the research project is greatly appreciated. On a personal level, I will always be grateful for the support he showed, particularly when he and my colleagues were by my side for hours at the emergency hospital following my injury during a squash game. I would like to express my sincerest gratitude to my co-supervisor, Prof. Dr. Oliver Jäkel, and Dr. Kristina Giske from DKFZ for their invaluable guidance and support throughout my research. Furthermore, I extend my gratitude to Prof. Dr. Martin Frank, my TAC member, for dedicating his input and time to this project.

Heartfelt thanks to my father, my brother, Vahid and my sister-in-law, Parnian for their invaluable support and encouragement. I am deeply thankful to Vahid and Parnian for taking care of me during my post-surgery period. Their love and support will always be cherished.

I would like to express my deepest gratitude to my wonderful landlords, Dr. Peter and Brigitte Schneider, for their kindness and support, particularly during

my post-operative recovery period.

Special thanks to HIDSS4Health – Helmholtz Information and Data Science School for Health, for their support. The writing retreat in Kloster Neustadt was a pivotal moment in my thesis journey. I am grateful to the steering committee, Prof. Dr. Michael Gertz, Prof. Dr. Ralf Mikut, Prof. Dr. Klaus Maier-Hein and the coordination office team, Dr. Ines Reinartz, Dr. Nicole Merkle, and Kathrin Brunk, for their support.

Many thanks to my friend, Dr. Christoph Klein, for his assistance during his time at ZITI, where he provided valuable support with scientific matters. Additionally, heartfelt thanks to my dear friends Mohammad Sajjadi, Dr. Nooraldeen Tarade, Dr. Artemis Mohseni, and all my other friends who have contributed in various ways during this journey.

I am grateful to my valued former colleagues, Dr. Kazem Shekofte, Dr. Lorenz Braun, Daniel Barley, Bernhard Klein, Hendrick Borras, Yannick Emonds, Dr. Günther Schindler, Dr. Jonas Dann, Kevin Stehle, Dr. Cornelius Bauer, Ama Katseena Yawson and Alexandra Walter, for their willingness to assist me. Their encouragement made my path easier. I also appreciate Andrea Seeger, the ZITI institute secretary, for her consistent administrative support.

I would also like to thank Prof. Dr. Andreas Mang at the University of Houston, along with his PhD students, Malte Brunn and Dr. Naveen Himthani, for their assistance with various inquiries about CLAIRE.

Looking back on my master's studies, I am indebted to Dr. Midia Reshadi, my master's advisor, and Prof. Dr. Amir Masoud Rahmani, my course advisor, for inspiring me to pursue further academic advancement.

Finally, I would also like to acknowledge that this work was performed on the HoreKa supercomputer funded by the Ministry of Science, Research and the Arts Baden-Württemberg and by the Federal Ministry of Education and Research.

In closing, I am truly appreciative of everyone who has supported me intellectually and emotionally, helping me maintain my determination and mitigate the impact of major difficulties and setbacks, and who has accompanied me through this chapter of my life.

Abstract

Image-guided adaptive radiation therapy (IGART) significantly enhances modern radiotherapy by adapting to patient movements and anatomical changes in real-time. However, the high computational demands of the technology prevent its deployment on standard computing systems. Furthermore, the requirement for re-calculating doses in real-time, which allows for immediate treatment adjustments in response to these variations, further complicates its implementation.

Given these substantial computational challenges, with optimization processes constituting the majority of execution time, this study focuses on the image registration (IR) task within the IGART application. IR, which relies on iterative optimization, is a time-intensive process that is essential for accurate image alignment and analysis.

Due to the deformable nature of anatomical changes during treatment, deformable image registration (DIR) is crucial for addressing variations in the shape and size of internal organs between the initial and adaptive planning images. This study specifically targets the registration of three-dimensional computed tomography (CT) lung scans, which are among the most deformable and complex medical datasets due to their elasticity and the influence of respiratory motion. However, our method is adaptable to images from various regions of interest and can extend beyond the medical imaging domain.

Accelerating DIR is essential for efficient and timely medical procedures given its high computational demands. While multi-GPU implementations offer significant potential for DIR acceleration, challenges such as high communication overhead and increased complexity have made such configurations impractical for real-time applications.

To overcome these challenges, this study proposes a novel solution named CLAIRE-ROP (constrained large deformation diffeomorphic image registration - rapid overlapped partitioning-based). This framework leverages multi-GPU

systems to accelerate DIR without increasing programming complexity, aiming to enable rapid and precise registration, thus facilitating real-time adaptation in IGART. Our approach employs a partitioning scheme that divides lung images into multiple partitions, thereby enabling the individual registration of each partition without compromising accuracy. Our method has successfully registered images from the largest publicly available lung dataset (512×512×136) in under 0.5 seconds, achieving a Dice score of 0.991.

This work represents a significant advancement in lung image registration techniques, facilitating more efficient DIR in clinical applications. Currently, our achieved registration time on the DIR-Lab dataset is the fastest among all published DIR methods for 4DCT, encompassing both deep learning and optimization-based approaches. Notably, our results demonstrate that our approach maintains competitive registration accuracy.

Zusammenfassung

Die bildgestützte adaptive Strahlentherapie (IGART) stellt eine wesentliche Weiterentwicklung der modernen Strahlentherapie dar, da sie eine Anpassung der Strahlendosis an die individuellen Bewegungen des Patienten sowie anatomische Veränderungen ermöglicht. Die hohen Anforderungen an die Rechenleistung dieser Technologie verhindern jedoch ihren Einsatz auf Standard-Computersystemen. Zudem erweist sich die Umsetzung als anspruchsvoll, da die Dosis in Echtzeit neu berechnet werden muss. Dies ermöglicht eine unmittelbare Anpassung der Therapie, um auf Schwankungen zu reagieren.

In Anbetracht der signifikanten rechnerischen Herausforderungen, bei denen Optimierungsprozesse den Großteil der Ausführungszeit beanspruchen, konzentriert sich diese Studie auf die Aufgabe der Bildregistrierung (IR) innerhalb der Anwendung IGART. Die IR, die auf einer iterativen Optimierung basiert, ist ein zeitintensiver Prozess, der für eine präzise Bildausrichtung und -analyse unabdingbar ist.

Aufgrund der Verformbarkeit anatomischer Veränderungen während der Behandlung ist die deformierbare Bildregistrierung (DIR) von entscheidender Bedeutung, um Schwankungen in Form und Größe innerer Organe zwischen den ursprünglichen und den adaptiven Planungsbildern auszugleichen. Die vorliegende Studie fokussiert sich auf die Registrierung von dreidimensionalen Computertomografien (CT) der Lunge. Aufgrund ihrer Elastizität und des Einflusses von Atembewegungen gehören CT-Aufnahmen der Lunge zu den am stärksten verformbaren und komplexesten medizinischen Datensätzen. Die Anwendbarkeit unserer Methode beschränkt sich jedoch nicht auf die medizinische Bildgebung, sondern kann auch auf andere Bereiche übertragen werden.

Die Beschleunigung von DIR ist angesichts des hohen Rechenaufwands für effiziente und zeitnahe medizinische Verfahren von entscheidender Bedeutung. Multi-GPU-Implementierungen bieten zwar ein beträchtliches Potenzial

für die DIR-Beschleunigung, jedoch haben Herausforderungen wie ein hoher Kommunikations-Overhead und eine erhöhte Komplexität solche Konfigurationen für Echtzeitanwendungen bisher unpraktikabel gemacht.

Die Bewältigung der dargestellten Herausforderungen erfolgt in dieser Studie mittels des neuartigen Lösungsansatzes CLAIRE-ROP (constrained large deformation diffeomorphic image registration – rapid overlapped partitioning-based). Das Framework nutzt Multi-GPU-Systeme zur Beschleunigung von DIR, ohne die Komplexität der Programmierung zu erhöhen. Dadurch wird eine schnelle und präzise Registrierung ermöglicht, was eine erleichterte Echtzeitanpassung in IGART zur Folge hat. Unsere Methode hat erfolgreich Bilder aus dem größten öffentlich verfügbaren Lungen-Datensatz (512×512×136) in weniger als 0,5 Sekunden registriert und dabei einen Dice-Score von 0,991 erreicht.

Diese Arbeit stellt einen bedeutenden Fortschritt in der Registrierung von Lungenbildern dar und ermöglicht ein effizienteres DIR in klinischen Anwendungen. Derzeit ist die von uns auf dem DIR-Lab-Datensatz erreichte Registrierungszeit die schnellste unter allen veröffentlichten DIR-Methoden für 4DCT, einschließlich Deep-Learning- und optimierungsbasierter Ansätze. Insbesondere zeigen unsere Ergebnisse, dass unser Ansatz die konkurrenzfähige Genauigkeit der Registrierung beibehält.

Contents

CHAPTER 1

Introduction

The objective of radiation therapy (RT) is to minimize doses to the surrounding healthy tissues while delivering clinically effective doses to the target. Modern high-precision RT is governed by the trade-off between tumor control and sparing of healthy tissue. Prior to irradiation, a treatment plan is individually tailored to the disease condition and the anatomy of the patient based on imaging scans. An illustrative example of a radiation treatment plan can be seen in Figure 1.1.[1]

However, based on extensive clinical experience, patients who are undergoing RT may experience notable anatomical changes due to organ deformation or respiratory motion, which can result in the deterioration of the optimized dose distribution. Some contemporary approaches to radiotherapy have been developed that rely extensively on medical imaging to establish a spatial correlation between the target area and the surrounding healthy tissue. In image-guided radiotherapy (IGRT), the use of various in-room imaging techniques allows for enhanced accuracy. These techniques provide clinicians with a valuable opportunity to evaluate the treatment process during a course of treatment by detecting the target position and then applying online geometric corrections to the patient position, a process based on coordinate transformation.

Another concept in RT is image-guided adaptive radiation therapy (IGART), which provides physicians with the opportunity to perform a more complex correction that is accomplished through replanning based on the actual target

[1] https://www.elekta.com/products/radiation-therapy/versa-hd/

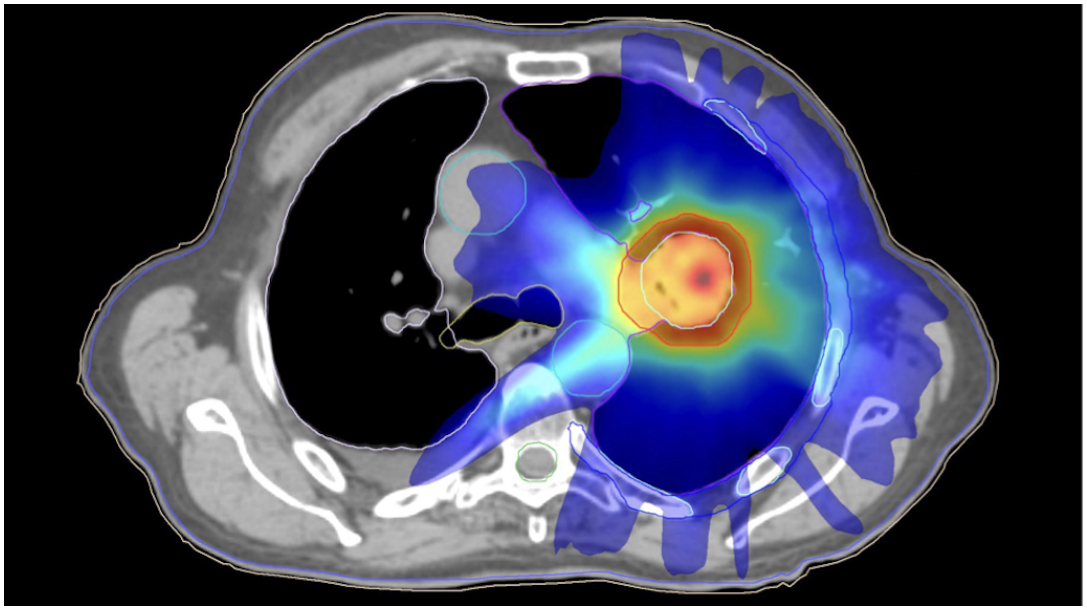


Figure 1.1 Example of radiation treatment plan. The red areas indicate regions receiving the prescribed radiation dose, while the green and blue areas receive lower doses. The contours outline the target volumes and critical structures to ensure effective treatment while minimizing exposure to healthy tissue.

contours. This allows for the modification of an initial radiation therapy treatment plan to adapt to the changes in the target area or healthy organs during a course of radiation therapy. A general schematic of IGART is shown in Figure 1.2.

As can be seen, this adaptive treatment necessitates a series of distinct, computationally demanding steps, frequently involving iterative optimization. However, such approaches are constrained in terms of computational and accuracy performance.

In this study, we concentrate on the image registration (IR) task, which plays a pivotal role in the stages of IGART. As can be observed, the process is based on iterative optimization and accounts for a significant portion of the total execution time. Given that anatomical changes during treatment are predominantly deformable, deformable image registration (DIR) plays a vital role in RT, as it enables the precise alignment of treatment plans with the target structures within the patient's body. This is especially crucial in instances where there is organ motion or deformation, as these can impact the precision of radiation delivery and increase the risk of damage to healthy tissues in the vicinity of the target. DIR techniques can address these concerns and improve the accuracy of treatment planning and delivery. Furthermore, real-time DIR

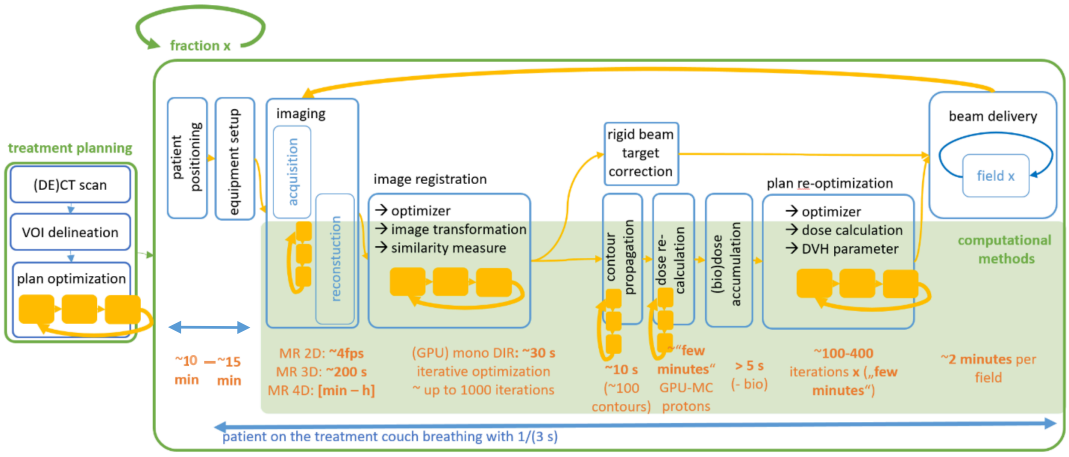


Figure 1.2 General scheme of IGART workflow. The yellow symbols represent steps with considerable computational load due to the necessity of iterative optimization. The orange color is used to summarize the calculation times for the current best-in-class methods in RT. Without image registration, at beam delivery, anatomy changes due to organ deformation or respiratory motion cause deterioration of the optimized dose distribution. This figure is outlined in the PhD work plan.

can facilitate adjustments to the treatment plan during treatment, which may lead to improved patient outcomes. While achieving high accuracy is imperative in the context of radiotherapy, the demand for efficient and rapid deformable image registration remains a necessity across various domains of application.

Despite the considerable success of DIR methods in medical imaging, particularly in achieving high registration accuracy for tasks like radiotherapy planning and lung image analysis, they necessitate extensive optimization calculations and task-specific parameter tuning. This, in turn, requires significant computing resources and time, which constrains clinical applications that are particularly time-sensitive, such as IGART.

Several studies have proposed the use of DIR methods to achieve a reduction in processing time through the utilization of different accelerators, including GPUs. These studies can be referenced as follows: [13, 33, 31, 52, 53, 64, 74, 84]. Additionally, multiple authors have proposed a multi-GPU-based DIR framework to further accelerate registration times [78, 43, 34, 14]. Nevertheless, none of these methods has achieved real-time DIR.

In recent years, there has been a growing interest in deep learning (DL) methods, with numerous DL-based approaches being applied to DIR (see, for example, references [22, 23, 25, 54, 26, 36, 35, 73]). These approaches can

significantly reduce the computational time required for DIR. However, the issue of model generalization remains a significant challenge, as DL models typically require task-specific training for each medical application. It remains uncertain how these models will perform when applied to unobserved clinical datasets or transferred to different applications.

To achieve real-time registration, we thus consider the most promising approach for accelerating DIR, which involves employing a multi-GPU system. Specifically, we select a multi-GPU version of the **c**onstrained **la**rge deformation diffeomorphic **i**mage **re**gistration (CLAIRE) framework, as detailed in reference [14]. CLAIRE is particularly well-suited for our needs because it ensures that the computed transformation is a smooth diffeomorphism, which is essential for applications like radiotherapy, where precision is paramount. Although originally designed for brain imaging, this study demonstrates the potential of CLAIRE as a robust framework for lung registration.

While CLAIRE effectively addresses large-scale imaging problems, it is not yet capable of achieving real-time DIR for realistic data sizes. Furthermore, in the case of strong scaling, where additional processors are incorporated without expanding the problem size, the runtime markedly increases due to the high communication requirements, thereby severely limiting scalability.

In this study, we present a novel method to address the limitations of CLAIRE, notably reducing the time required for real-time registration. The objective is to identify the optimal partitioning scheme for dividing lung images into multiple parts, thereby enabling the implementation of separate image registration for each partition. This partitioning strategy eliminates the necessity for communication between GPUs, as each GPU operates independently on its assigned partition, thereby reducing the overall processing time. As no dependencies exist among the different partitions, multiple partitions can be registered simultaneously and fully in parallel. This parallelism significantly accelerates the registration process in comparison to a single registration on a multi-GPU-based approach. This approach facilitates straightforward scalability. As the number of available GPUs increases, we can seamlessly adapt by assigning more partitions to utilize the additional computational power effectively. Furthermore, this method is distinguished by a notable advantage in customizing the lung area, allowing for tailored registration settings and parameters for each individual partition.

Throughout this study, we focus on registration of 3D CT lung scans, which

are among the most deformable and complex medical datasets due to their elasticity and the influence of respiratory motion during breathing. However, our method can register images of different regions of interest (ROIs) and extends beyond the medical imaging domain.

Our study includes an empirical analysis that investigates the impact of partition size and the number of GPUs on registration accuracy and time. It also provides a flexible and adaptable framework that can be applied to various image registration tasks beyond CLAIRE. Overall, our approach enables faster image analysis with a wide range of potential applications. It is particularly beneficial for large datasets and computationally intensive registration tasks. This work makes the following contributions:

1. Frameworks review, selection, and characterizing: We perform a comprehensive review of the state-of-the-art in DIR to identify the most promising existing frameworks for further development. Following this review, we conduct an extensive parameter search to determine the optimal execution settings for the selected framework, specifically for lung datasets. This approach ensures that the most suitable method is selected and configured for maximum performance and applicability for the remainder of this work [Chapters 3 and 4].

2. Improved scalability: We enhance the scalability of the selected DIR framework. By implementing an optimal partitioning scheme and dedicating GPUs to specific partitions, we minimize communication overhead between GPUs, significantly improving processing performance [Chapter 5].

3. Comprehensive evaluation: We provide a detailed, end-to-end comparison of our method against the state-of-the-art DIR methods, independent of hardware targets. Our evaluation includes both pre- and post-processing stages, demonstrating that our method not only reduces execution time but also outperforms existing approaches. Additionally, we compare it with DL-based DIR methods in terms of registration time, highlighting its efficiency [Chapter 6].

4. Released as open source: In keeping with our commitment to collaboration and transparency, we have made our framework available as open-source software. This initiative in intended to facilitate further advancements in

the field of lung image registration[1].

In the course of this research, we presented the following paper at the 2024 8th International Conference on Medical and Health Informatics (ICMHI 2024) in Yokohama, Japan. This presentation received the award for best presentation at the conference.

- Vahdaneh Kiani, Oliver Jäkel, Holger Fröning, "CLAIRE-ROP: Rapid Partitioning-based Deformable Image Registration on Multi-GPU Accelerator". In: *ICMHI '24: Proceedings of the 2024 8th International Conference on Medical and Health Informatics,* pp. 1–12, DOI: https://10.1145/3673971.3673983[2].

In summary, this work explores challenges and solutions in multi-GPU implementation for fast and robust lung image registration using the CLAIRE framework. The CLAIRE-ROP solution, with its partitioning scheme, effectively improves processing time without sacrificing accuracy. This work consists of a total of seven chapters. Chapter 2 serves as an introduction to the concepts of medical deformable image registration and accelerated computing. Chapter 3 provides an overview of related research in this field, focusing on studies aimed at accelerating DIR methods through various accelerators. In Chapter 4, we characterize CLAIRE with the objective of optimizing parameters for the lung dataset. We then proceed to compare it with Advanced Normalization Tools (ANTs), a well-known DIR method, and explore the limitations of CLAIRE. Chapter 5 presents our partitioning-based DIR method, developed to address the limitations identified in CLAIRE. A discussion of the overall execution time is presented in Chapter 6. This chapter also provides a brief overview of DL-based methods, with a particular focus on time-related aspects that are directly comparable to our own work. Finally, Chapter 7 serves as an outlook to this thesis, offering insights into potential future avenues of research.

[1]The code is accessible via https://github.com/UniHD-CEG/CLAIRE-ROP

[2]Available online at http://camps.aptaracorp.com/ACM_PMS/PMS/ACM/ICMHI2024/12/d9e8d643-2f06-11ef-8182-16bb50361d1f/OUT/icmhi2024-12.html

CHAPTER 2

Background

2.1 Introduction

Lung cancer is the second most commonly diagnosed cancer and remains the leading cause of cancer-related deaths [55]. RT represents a crucial treatment for lung cancer, employing high-energy radiation to eliminate cancer cells and halt their uncontrolled growth. However, the movement of lung tumors during natural expansion and contraction with each breath presents a significant challenge. Without proper image registration, these respiratory-induced anatomical changes can result in compromised dose distribution during irradiation [30]. In Figure 2.1, we show the intensity difference between two distinct respiratory phases: the end-inhalation phase and the end-exhalation phase, from an axial perspective. This visualization highlights the necessity for a deformable registration step.

In RT, efficient treatment planning is essential; however, the need for rapid processing also stems from the time-sensitive nature of RT procedures. The rapid processing of DIR facilitates real-time adaptive treatment and image-guided interventions, enabling immediate feedback and adjustments during procedures. This capability significantly enhances treatment precision and patient safety, highlighting the importance of timely responses within the radiotherapy workflow [41].

Among the various approaches aimed at accelerating DIR, GPUs are the most compelling choice. Since DIR is a data parallel task, with the deformation vectors

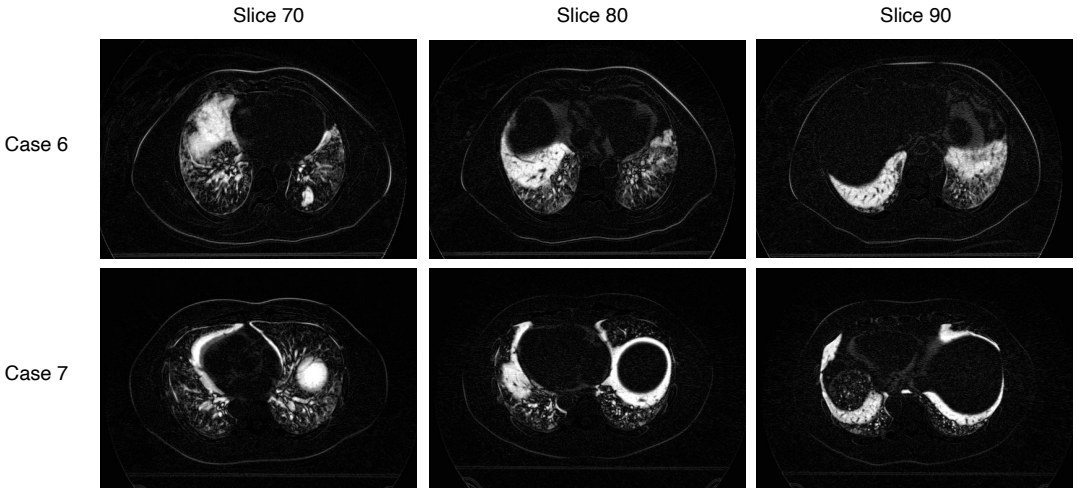


Figure 2.1 Axial view intensity difference slices between end-inhalation and end-exhalation phases, exemplifying the necessity for deformable registration, are illustrated by Cases 6 and 7 from the DIR-Lab 4DCT dataset [15]. DIR-Lab is one of the most cited public datasets for studies on 4DCT chest image registration.

at individual voxels able to be computed almost independently at each iteration step, GPUs are particularly well-suited for this application. Nevertheless, the accomplishment of real-time deformable registration computations on a single GPU remains a significant challenge, despite recent advancements in GPU technology. To address this challenge, a few studies have employed multiple GPUs, distributing the computational load across multiple GPUs to enhance parallelism. Adding further GPUs increases the potential for parallel processing but also introduces greater complexity in the interactions between the GPUs. Consequently, a multi-GPU utilization setup is less practical when dealing with clinical datasets. The concept of overlapping communication and computation serves as a technique to mitigate the impact of communication delays. However, applying overlapping techniques can be challenging for some mathematical tasks, such as interpolation and Fast Fourier Transforms (FFTs), which are the most significant contributors to the computational cost of deformable registration. These tasks involve data dependencies that complicate the process or prevent overlapping to a large extent.

2.2 Fundamentals of Image Registration

An image registration algorithm is typically comprised of three principal components. These comprise (i) a transformation model, (ii) a similarity metric, and (iii) an optimization method. A transformation model seeks to find the optimal alignment by deforming the image content to enhance the degree of similarity between the two images. The similarity metric serves to evaluate the efficiency of the registration process. The degree of similarity is determined by computing the correspondence between the two images.

Once an appropriate similarity metric and transformation model have been selected for the specific image registration problem, an optimization method should be applied as the final step to obtain the transformation parameters that yield the best registration.

Let F(x) represent the reference image set, and M(x') represent the template image set. In this context, x and x' denote the spatial coordinates in each image set, respectively. The transformation function, designated as T(x', β) is responsible for mapping the template image to the reference image, where β represents the parameters of the transformation. The objective of image registration is to reduce the discrepancy between F(x) and the transformed template image M(T(x', β)). The transformation function is defined as the sum of the local position vector in the template image, x', and the displacement vector, u(x', β). Therefore, the optimal image registration between the two image sets can be described as follows: The reference image set, F(x), is equal to the template image set, M(T(x',β)), which is equal to M(x'+u(x', β)). For further details, please refer to reference [14].

The typical image registration process can be illustrated in Figure 2.2. The iterative optimization process is continued until a stopping criterion is met. This criterion may be based on the change in the similarity metric, the number of iterations, or other factors. Once the optimization process has reached a point of convergence, the final transformation is applied to one of the images to align it with the other. This may entail non-linear warping, with the degree of deformation varying according to the algorithm and the characteristics of the images in question. After registration, the quality of the alignment is often evaluated through the utilization of diverse metrics, to ascertain the efficacy of the registration process.

The number of parameters required to determine the transformation T is dependent upon its form, which in turn is dependent upon the anatomical site, clinical application, and the imaging modalities involved.

The transformation T can range from a relatively simple form, as in rigid body transformations involving six parameters (three for translation and three for rotation), which is known as rigid image registration (RIR), to a more complex form. RIR is particularly well-suited to scenarios where no anticipated anatomical changes are present. In scenarios necessitating augmented flexibility, such as for accommodating scaling, shearing, and plane reflection, the transformation can be augmented to encompass nine or twelve parameters, representing affine transformations.

In instances where spatial variance is present, the transformation can possess degrees of freedom that are as extensive as three times the number of voxels present in the source dataset. This approach, known as deformable image registration (DIR), entails the use of a distinctive displacement vector for each voxel within the source dataset. DIR becomes a crucial tool when dealing with changes in anatomy due to factors such as organ deformation or respiratory motion, which RIR is unable to accommodate. However, the accuracy of DIR algorithms is difficult to ascertain in clinical settings due to the lack of a definitive ground truth.

In contrast to RIR, these spatially variant vector fields are typically constrained by a regularization function. This function ensures that the transformations are both anatomically and physiologically realistic, limiting unrealistic movements and producing a smooth deformation field. For instance, it may classify a region as bone, thereby restricting the extent of deformation in that area. This method ensures that the transformations adhere to the realistic constraints of human anatomy and physiology.

DIR algorithms are commonly utilized in medical imaging for a multitude of applications, including the alignment of pre- and post-operative scans, the tracking of organ motion, and numerous others. The iterative optimization process is the key to achieving accurate and reliable registration in these applications.

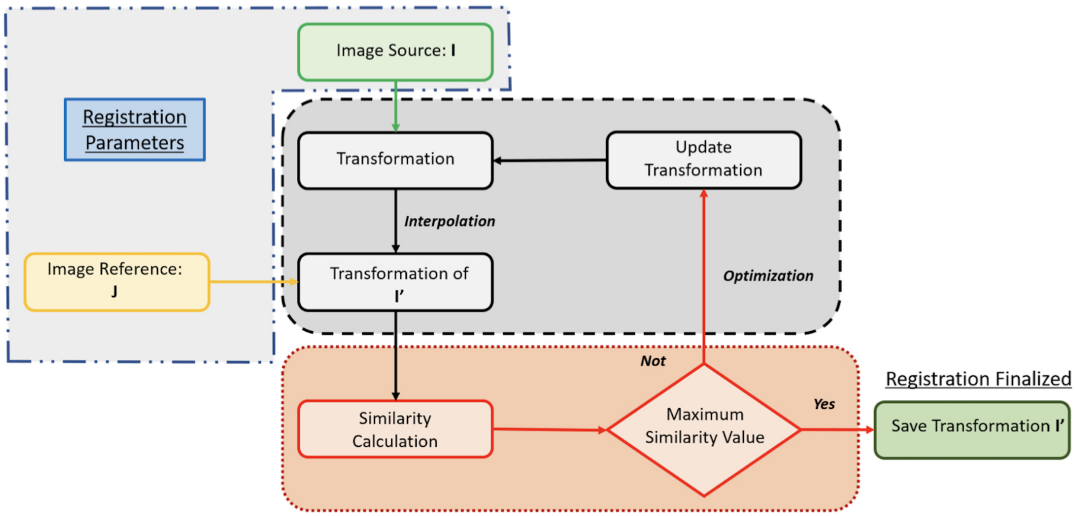


Figure 2.2 A typical DIR workflow comprises three main components: (i) a transformation model, (ii) an objective function, and (iii) an optimization method, as referenced in [4].

2.3 Specialized Image Registration for Lung Analysis

In RT, DIR is of particular importance when registering images of disparate ROIs, such as the lungs. Deformation due to respiratory motion can significantly impact the accuracy of radiation delivery, underscoring the necessity for precise registration.

Accurate lung registration is crucial in the context of RT treatment planning, as it enables the precise targeting of tumors while minimizing the risk of damage to surrounding healthy tissues. However, respiratory motion can result in substantial deformation of the lung tissue, thereby rendering accurate registration a challenging endeavor. To address this challenge, several DIR algorithms have been developed that can account for respiratory motion and other sources of deformation in the lung tissue. In general, accurate lung registration using DIR techniques is of paramount importance in RT. Nevertheless, it can be time-consuming due to the complex nature of the lung tissue and the necessity for precise alignment despite respiratory motion.

2.3.1 Anatomy of the Lung

The lungs are paired, pyramid-shaped organs that are integral to the respiratory system. They are connected to the trachea by the right and left bronchi. The lungs are positioned within the thoracic cavity, with the diaphragm serving as their lower boundary. From an anatomical perspective, the lungs can be divided into two distinct entities: the right lung and the left lung. Each lung is characterized by a specific set of lobes and unique anatomical features. Despite their similarities, the anatomy of the right and left lungs exhibits asymmetry. The right lung is composed of three lobes: the superior, or right upper lobe (RUL), the middle, or right middle lobe (RML), and the inferior, or right lower lobe (RLL). In contrast, the left lung is composed of two lobes: the superior, or left upper lobe (LUL), and the inferior, or left lower lobe (LLL) [17]. Figure 2.3 illustrates the delineation of lobe areas within a single CT image slice from axial, coronal, and sagittal perspectives. This segmentation was performed using an online platform, Totalsegmentator [82], and then rendered in a 3D visualization tool, 3D Slicer [24], to illustrate the spatial arrangement and extent of each lobe.

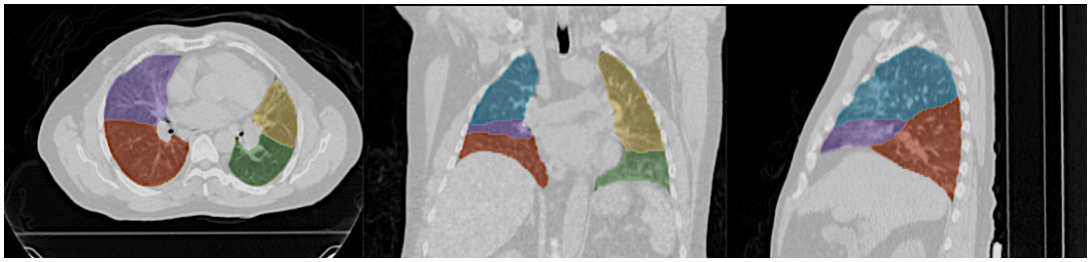


Figure 2.3 Segmentation and 3D visualization of the pulmonary lobes from the DIR-Lab dataset CT scans for Case 6.

2.3.2 CT Imaging Techniques

Modern RT employs a range of imaging data for treatment planning, delivery, and monitoring. Computed tomography (CT) imaging is primarily utilized in the context of treatment planning, whereby detailed three-dimensional (3D) anatomical and physical models of patients are created. This facilitates the optimization of beam arrangement, enhances the accuracy of dose calculation, and improves patient positioning.

Magnetic resonance imaging (MRI) serves to supplement CT by providing exceptional contrast for soft tissue and offering insight into physiological and

metabolic processes. Moreover, nuclear medicine imaging techniques, such as positron emission tomography (PET) and single photon emission computed tomography (SPECT), provide dynamic insights into physiological and metabolic processes, including glucose metabolism and DNA synthesis [44].

CT imaging employs X-ray beams to capture three-dimensional pixel intensities within the human body. The process involves the generation of high-energy electron beams from a heated cathode, which are then accelerated towards the anode to produce X-rays. The X-rays traverse the body's tissues and are detected on the opposite side. Dense tissues, such as bones, absorb more radiation compared to softer tissues, like fat. Consequently, areas with low absorption, including air-filled spaces within the lungs, are displayed in black on the images, while dense tissues that absorb more X-rays are displayed as white regions. This contrast in display color reflects the varying levels of X-ray absorption in different tissues.

During CT scanning, patients are positioned on a table that is moved through a ring-shaped scanner. X-ray beams are directed through the body from various angles, thereby producing a series of thin cross-sectional images, or "slices." Figure 2.4 illustrates an example of a CT lung scan comprising these two-dimensional slices. These are subsequently digitally reconstructed into a 3D representation of the ROI.

Three-dimensional lung CT scans are an essential imaging modality in the field of medicine, particularly in the context of pulmonary diagnostics and treatment planning. CT scans encompass three spatial dimensions: length (X-axis), width (Y-axis), and depth (Z-axis), offering a comprehensive volumetric view of lung anatomy. The quantity of slices (Z-axis) in CT lung images varies based on patient specifics and the intended imaging protocol or clinical indication. Figure 2.5 illustrates a volumetric CT lung image, accompanied by three perspective views: axial, coronal, and sagittal.

2.3.3 Dynamic Imaging with 4DCT

Four-dimensional computed tomography (4DCT) represents an advanced imaging technique that is integral to radiotherapy treatment planning. It captures time-varying 3D image sequences of the thorax throughout the respiratory cycle, thereby providing essential insights into lung movement during breathing [79].

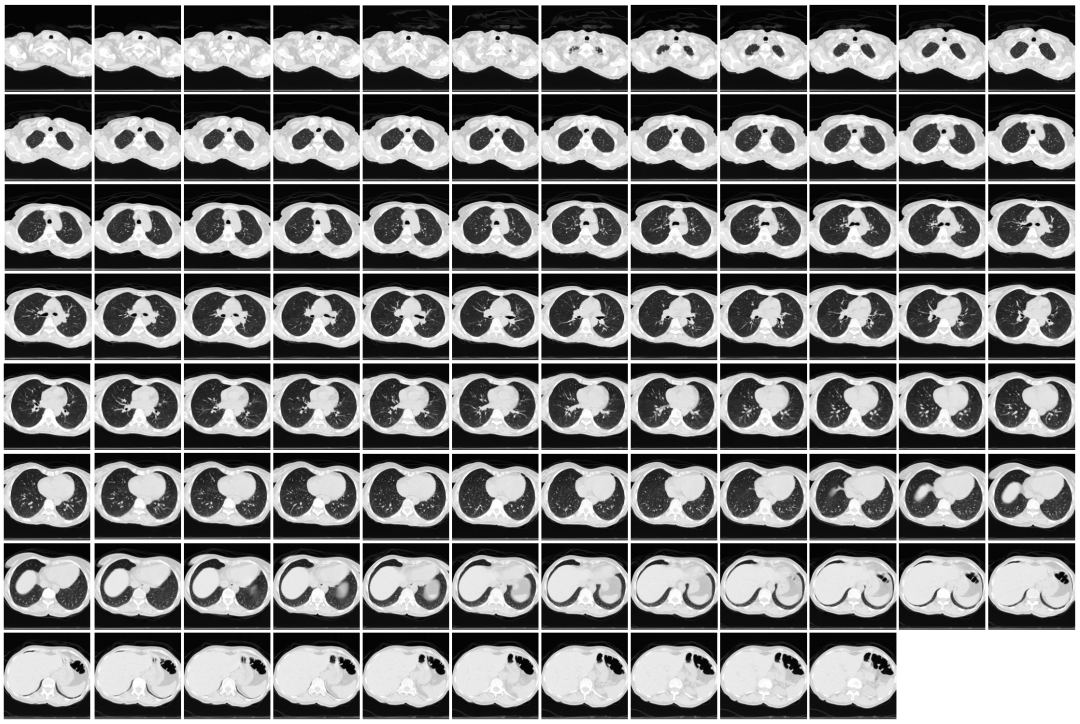


Figure 2.4 Axial slices of a 4DCT lung scan from the DIR-Lab Case 1 at the end of inhalation phase (T00) [16].

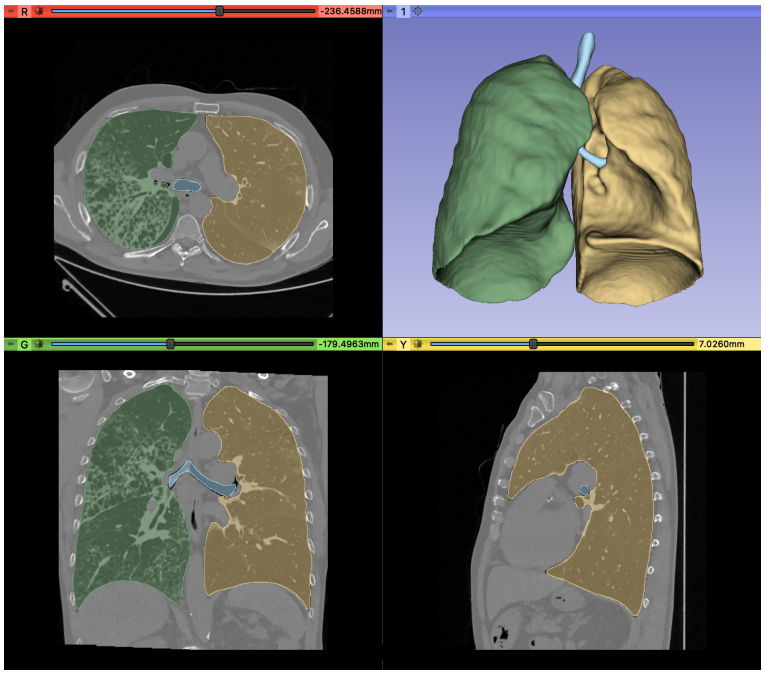


Figure 2.5 Example of a volumetric CT lung image, created using the DemoChestCT/DemoLungMasks module in 3D Slicer.

To quantitatively analyze this motion and related physiological information from 4DCT images, DIR is essential.

2.3.4 Dataset for Lung Studies

Our study makes use of the DIR-Lab 4DCT dataset, which is a highly regarded resource in the field of image registration research[1]. This dataset comprises ten 4D lung image cases, each consisting of ten 3D CT phases (T00 to T90) that capture various stages of the respiratory cycle.

In the present study, we have classified the DIR-Lab 4DCT dataset into two distinct categories based on image size and detail. The initial five cases (Case 1 to Case 5), designated as the "small dataset", have been specifically tailored to focus on the thorax region. The images have been cropped and resampled to in-plane dimensions of 256 $\times$ 256, as detailed in references [16, 15], allowing for a concentrated analysis of the thoracic area, which is crucial for accurately evaluating lung motion during the respiratory cycle. In contrast, the last five cases (Case 6 to Case 10) are considered as our "large dataset". These cases retain their original in-plane dimensions of 512 $\times$ 512, thereby providing a more extensive view of the thoracic cavity.

In our evaluation, we selected phase T00 (end of inhalation) as the reference image and phase T50 (end of exhalation) as the template image. These phases were chosen due to their significant representation of lung deformation (Figure 2.6), which facilitates a comprehensive assessment of registration accuracy.

2.3.5 Metrics for Evaluating Lung Image Registration

In order to ascertain whether the lung's deformation field is indeed compensating for respiratory motion effectively, a number of metrics may be employed. Among these, target registration error (TRE) and dice score are the most commonly used for evaluating DIR accuracy. TRE, calculated as the mean distance between corresponding landmarks in reference and template images post-registration, provides a direct measure of displacement error at specific anatomical points. Although intuitively appealing, the use of TRE is constrained by its dependence on the selection of landmarks, which can be subjective and may not fully encompass the entire image volume. Furthermore, the involvement of experts in

[1] http://www.dir-lab.com/

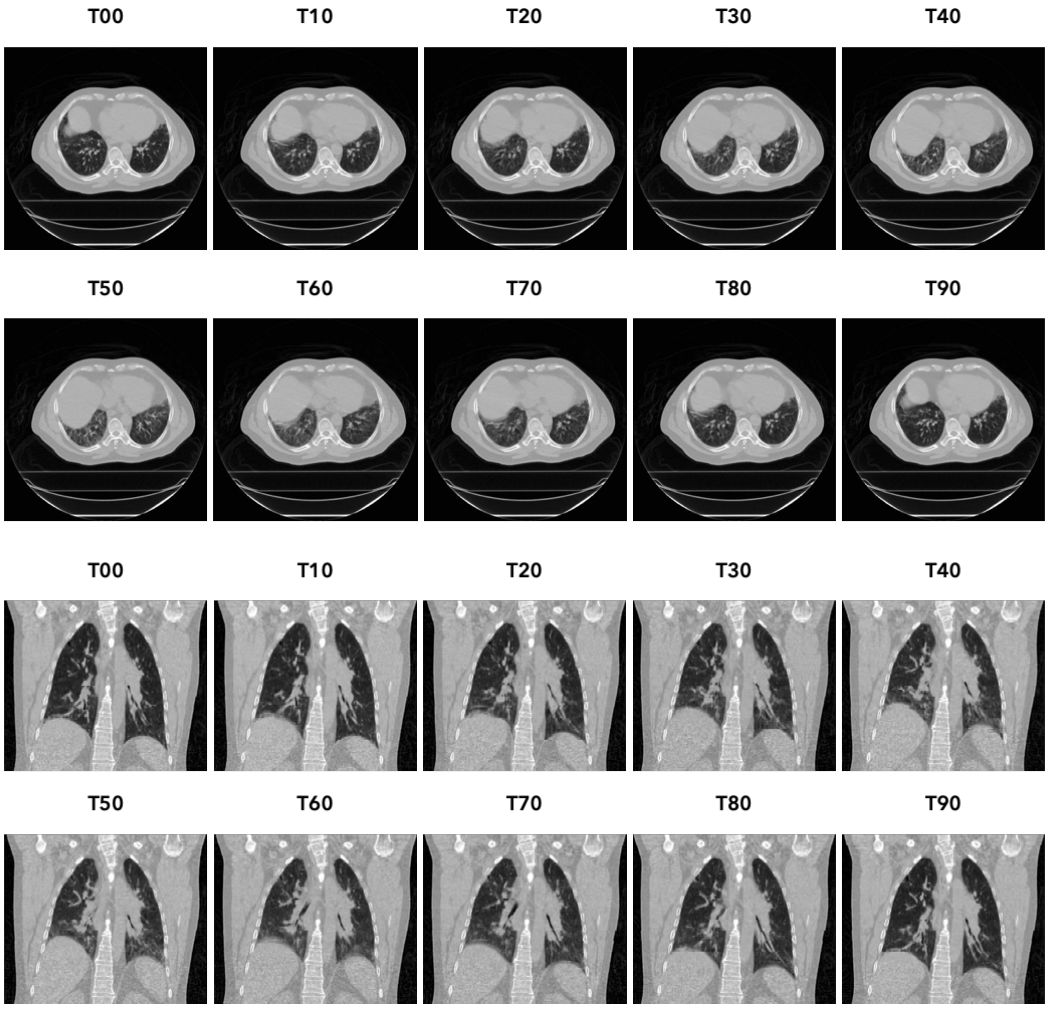


Figure 2.6 Axial (slice 81) and coronal (slice 260) views from T00 to T90 for Case 6. The most significant differences in lung deformation are observed between T00 and T50.

defining reference points adds to the complexity of this process. In contrast, the Dice score, which qualities the degree of overlap between two datasets, offers a more comprehensive assessment of registration quality, particularly in the context of volumetric analyses. In order to calculate the Dice score between two segmentations, A and B, the following equation is used: $\frac{2 \cdot |A \cap B|}{|A| + |B|}$. This score can range from 0, indicating no overlap, to 1, indicating perfect overlap. However, it is susceptible to variation in the size of the ROI and lacks specificity in localizing errors, as cited in reference [66].

In this study, we calculated the Dice score for each registration method to assess the degree of overlap between the reference and registered binary masks. Furthermore, we evaluate each lung lobe individually to ensure a comprehensive and reliable assessment of registration quality across diverse anatomical and functional regions. This approach allows for the successful registration to be identified not only globally but also within each specific lung lobe.

Binary masks are crucial for this analysis and are images where each pixel is either assigned a value of 0 or 1. In this context, binary masks are derived from grayscale CT images and represent ROI (such as lung tissues) where the value of 1 indicates the presence of the structure, and 0 indicates its absence. These masks are used to quantify and analyze the accuracy of the registration process.

2.4 Accelerated Computing

Accelerated computing involves the use of specialized hardware to significantly accelerate specific types of workloads. This approach is particularly crucial for tasks that are computationally demanding and can exceed the processing capabilities of traditional central processing units (CPUs) due to their serial execution nature. Accelerated computing relies on the use of specialized hardware components, including Graphics Processing Units (GPUs), and Field-Programmable Gate Arrays (FPGAs). One of the fundamental characteristics of accelerated computing is the ability to perform parallel processing. In contrast to CPUs, which typically execute tasks sequentially, accelerators are capable of processing multiple tasks concurrently. This parallelism is particularly beneficial for tasks that can be decomposed into smaller, independent subtasks. Accelerated computers often utilize a heterogeneous computing architecture, wherein multiple processor types, including CPUs and various accelerators, operate in unison to achieve optimal

performance and energy efficiency across a diverse range of applications. This approach leverages the unique capabilities of each processor type to enhance overall system efficiency[1].

2.4.1 Utilizing GPUs for Computational Acceleration

GPUs are the most widely utilized accelerators, initially developed as coprocessors for applications with high graphical demands. However, a significant shift occurred with Nvidia's introduction of compute unified device architecture (CUDA) and the Tesla architectures in 2006, which enabled these devices to be utilized for general-purpose computing. The family of GPUs designed for data center and high-performance computing (HPC) applications has seen a consistent release of new architectures, each named after notable innovators. The GPUs designated as Tesla, Fermi, Kepler, Maxwell, Pascal, Volta, Turing, and Ampere represent a series of architectural advancements.

The fundamental design of a GPU enables data parallelism, which involves the execution of numerous threads. These threads are divided into a number of thread blocks, which are then executed in parallel on streaming multiprocessors (SM). Each SM in a GPU is capable of supporting the concurrent execution of hundreds of threads, with each thread being able to process a relatively small part of the data (e.g., one pixel or voxel). In general, a GPU contains multiple SMs, which allows for the concurrent execution of thousands of threads on a single GPU.

As a result, GPUs have undergone a significant transformation, evolving from specialized processors to versatile parallel computing engines. This has enabled groundbreaking solutions across a multitude of fields, including scientific research and artificial intelligence [60].

2.4.2 Advancements with Multi-GPU Systems

In multi-GPU systems, GPUs are interconnected using technologies such as NVLink, which enables them to communicate and share data seamlessly[2]. This configuration offers several advantages over a single GPU setup, including:

[1] https://blogs.nvidia.com/blog/2021/09/01/what-is-accelerated-computing/
[2] https://medium.com/gpgpu/multi-gpu-programming-6768eeb42e2c/

- **Problem domain size:** Some datasets are too large to fit into the memory of a single GPU.
- **Increased processing power:** Multi-GPU systems combine the computational power of multiple GPUs, significantly enhancing overall processing capabilities.
- **Scalability:** As workload demands grow, additional GPUs can be added to the system, providing scalable performance improvements.
- **Parallelism:** Multi-GPU setups leverage parallel processing across multiple GPUs, distributing workloads efficiently and reducing processing times.
- **Redundancy and fault tolerance:** Multi-GPU configurations can provide redundancy and fault tolerance by distributing computations across multiple devices, reducing the risk of system failure due to hardware issues.
- **Specialized workloads:** Certain tasks, such as DL training and complex simulations, can benefit greatly from parallel processing across multiple GPUs, allowing for faster completion times and increased model accuracy.

In general, multi-GPU systems provide enhanced computational power, scalability, and efficiency, rendering them well-suited for a diverse array of demanding computational tasks across domains such as deep learning, scientific research, and HPC. In this study, we leverage the capabilities of multi-GPU acceleration to enhance the efficiency of the registration process. We envision a generic concept that facilitates the deployment of DIR computational tasks on a multi-GPU system.

Although multi-GPU systems offer a number of advantages, they also present a number of challenges and considerations during the design and deployment process. These include issues related to scalability, communication overhead, and software optimization. These aspects are explained in greater detail as follows:

- **Software optimization:** Software optimization involves tuning and refining the framework, including algorithms, libraries, and frameworks to maximize performance and efficiency on multi-GPU systems. Software optimization strategies may include the parallelization of algorithms to take advantage of GPU parallelism, optimizing memory access patterns to minimize latency and maximize throughput, and leveraging GPU-specific features and libraries for accelerated computation.

- **Scalability limitations:** Scalability limitations refer to challenges in effectively utilizing additional GPUs as the system grows in size. In practice, achieving ideal linear scalability can prove challenging due to various factors, such as diminishing returns observed in parallel efficiency, the overhead incurred by synchronization, and the contention for shared resources. Strategies for addressing scalability limitations may involve optimizing workload distribution, minimizing communication overhead, and employing efficient parallel algorithms that fully utilize all available resources.
- **Communication overhead:** Communication overhead refers to the time and resources consumed by inter-GPU communication, including data transfer and synchronization operations. In multi-GPU systems, it can become a significant bottleneck, particularly when transferring large volumes of data between GPUs or synchronizing computation across multiple devices. To mitigate communication overhead, techniques such as overlapping communication with computation, optimizing data transfer patterns, and minimizing unnecessary synchronization points can be employed. Additionally, the use of high-speed interconnects like NVLink can reduce latency and bandwidth limitations.

Through careful design, optimization, and strategic planning, developers can effectively address the challenges associated with multi-GPU systems, thereby achieving notable performance improvements for a range of computational tasks. This work is concerned with overcoming the aforementioned challenges, thus maximizing the potential of multi-GPU systems to enhance the accuracy and efficiency of lung image registration. The potential for software optimization is explored in Chapter 4, while the scalability limitations and communication overhead are discussed in Chapters 5 and 6.

We evaluate the effectiveness of parallelization through strong scaling, which measures the impact of the number of processors utilized. In contrast to weak scaling, which involves increasing both the number of processors and the problem size, strong scaling involves increasing the number of processors while keeping the problem size constant. The strong scaling speedup is defined as the ratio of the time required to complete a task of size M with a single GPU, $t(1, M)$, to the time required to complete the same task with N processing components,

$t(N, M)$:

$$Speedup = \frac{t(1,M)}{t(N,M)}$$

To assess the registration time, we report the results obtained on the HoreKa system[1]. HoreKa comprises 167 nodes, each equipped with four NVIDIA A100 GPUs with 40 GB RAM and two Intel Xeon Platinum 8368 CPUs with 76 cores each, resulting in a total of 152 cores per node. Further details about HoreKa are provided in Section 2.6.

2.5 Integration of MPI and CUDA for Optimized Performance

CUDA, developed by NVIDIA, is a parallel computing platform that leverages the power of GPUs to achieve significant performance gains. In contrast, the message passing interface (MPI) is a widely used standard application programming interface (API) in HPC for communication between distributed processes, enabling applications to scale across multi-node clusters. While CUDA is effective in exploiting parallelism on a single machine, combining it with MPI offers numerous advantages, particularly in large-scale computing environments:

- **Handling large data sizes:** The combination of MPI and CUDA allows for the resolution of problems involving data sets that exceed the capacity of a single GPU's memory.
- **Reducing compute time:** The approach markedly reduces the time required for computations that would otherwise take an excessively long time on a single node.
- **Enhancing MPI applications:** Incorporating GPU acceleration into existing MPI applications can significantly enhance performance.
- **Scaling across nodes:** The integration permits the effective scaling of a single-node, multi-GPU application across multiple nodes within a cluster.

In standard MPI programs, communication between processes typically involves passing pointers to host memory. However, in instances where MPI is

[1] https://www.scc.kit.edu/dienste/horeka.php

integrated with CUDA, there arises a necessity to facilitate direct communication of data from GPU memory buffers. This requirement presents challenges in non-CUDA-aware MPI implementations, where data must first be copied from the GPU to host memory before it can be sent to another process. The additional step of copying data introduces a significant amount of overhead, which in turn reduces the overall efficiency of the communication process, as illustrated in the code snippet below.

To address this inefficiency, CUDA-aware MPI implementations have been developed, allowing direct communication between GPU memory buffers across different processes. By eliminating the intermediate step of copying data to host memory, CUDA-aware MPI optimizes the communication process, resulting in improved performance. In addition to CUDA-aware MPI, technologies such as NVIDIA GPUDirect P2P (peer-to-peer) further enhance communication efficiency. GPUDirect P2P enables direct data transfers between GPUs without involving the host CPU, particularly when the GPUs are resided on the same node. When combined with CUDA-aware MPI, this allows MPI ranks executing on the same host (e.g., MPI rank 0 and MPI rank 1) to achieve highly efficient direct transfers of data between their respective GPU memories. This results in significant performance enhancements in multi-GPU applications where intra-node communication is prevalent.

The advantages of integrating CUDA-aware MPI and GPUDirect P2P are particularly evident in HPC environments, where minimizing communication overhead is vital for scaling applications across large clusters. By leveraging these technologies, developers can achieve accelerated data exchanges, reduced latency, and enhanced computational efficiency[1].

```
1  // MPI rank 0
2  cudaMemcpy(s_buf_h, s_buf_d, size, cudaMemcpyDeviceToHost);
3  MPI_Send(s_buf_h, size, MPI_CHAR, 1, 100, MPI_COMM_WORLD);
4
5  // MPI rank 1
6  MPI_Recv(r_buf_h, size, MPI_CHAR, 0, 100, MPI_COMM_WORLD, &status);
7  cudaMemcpy(r_buf_d, r_buf_h, size, cudaMemcpyHostToDevice);
```

[1] https://developer.nvidia.com/blog/introduction-cuda-aware-mpi/

Table 2.1 Overview of the hardware features of accelerator nodes.

Feature	HoreKa Green
No. of nodes	167
CPUs	Intel Xeon Platinum 8368
CPU Sockets per node	2
CPU Cores per node	76
CPU Threads per node	152
Cache L1	64K (per core)
Cache L2	1 MB (per core)
Cache L3	57 MB (shared, per CPU)
Main memory	512 GB
Accelerators	4x NVIDIA A100-40
Memory per accelerator	40 GB
Local disks	960 GB NVMe SSD
Interconnect	InfiniBand HDR

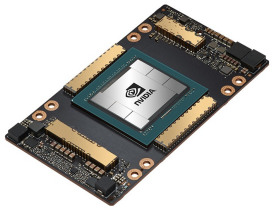
2.6 Computational Infrastructure: The HoreKa Supercomputer Cluster

HoreKa is a distributed memory parallel computer consisting of hundreds of individual servers called nodes. Each node is equipped with two Intel Xeon processors, a minimum of 256 GB of local memory, local NVMe SSD disks, and two high-performance network adapters. All nodes are interconnected via InfiniBand 4X HDR interconnect, which offers extremely fast data transfer rates and low latency. Additionally, two substantial parallel file systems are linked to HoreKa. Each node operates on Red Hat Enterprise Linux (RHEL) 8.x as its operating system. A suite of open-source software components, including Slurm[1], has been installed on top of this operating system.

HoreKa features three types of nodes: standard nodes (HoreKa Blue), accelerated nodes with GPUs (HoreKa Green), and nodes equipped with NVIDIA H100 GPUs (HoreKa Teal). To conduct our research, we employed the use of HoreKa Green. The comprehensive technical specifications for HoreKa Green are presented in Table 2.1. Furthermore, a comparison of the CPU and GPU capabilities within HoreKa is illustrated in Table 2.2.

[1] https://slurm.schedmd.com

Table 2.2 Hardware comparison of Intel Ice Lake Xeon and NVIDIA Ampere A100 in the HoreKa system, highlighting key specifications for accelerator nodes.

Intel Ice Lake Xeon	**NVIDIA Ampere A100**
	
10 nm, 38 Cores, 8x DDR4	7 nm, 6912 Cores, 6x HBM2
Optimized for general purpose	Optimized for highest performance
Average floating point performance	High floating point performance
Large memory with medium throughput	Small, fast memory
External interconnect (InfiniBand)	Internal NVLink mesh for multi-GPU

CHAPTER 3

State of the Art

In this section, we review studies that have attempted to accelerate DIR using various hardware and software solutions, with particular emphasis on lung and brain datasets. These organs are critical ROIs in medical imaging, and given the distinctive attributes of medical images, it is imperative to assess DIR frameworks specifically for each ROI. By analyzing the strengths and limitations of existing frameworks, we can identify the most suitable one to address our research question. To that end, we provide a rationale for selecting the CLAIRE framework by discussing relevant studies in the context of the current literature.

The present study focuses on research related to large-deformation diffeomorphic metric mapping (LDDMM) methods, which are essential for ensuring smoothness, invertibility, and the preservation of topology during image registration. LDDMM ensures that transformations applied to images remain smooth, preventing abrupt changes or distortions to the underlying anatomical structures. Additionally, it guarantees that each transformation is invertible, ensuring that the original image can be recovered from the transformed one. This is an essential feature in applications where comparisons between original and deformed images are often required. Furthermore, LDDMM preserves the topological characteristics of anatomical structures, preventing the formation of holes or merging of distinct regions, which is particularly important for maintaining anatomical integrity in medical applications.

Advanced LDDMM algorithms deliver high accuracy; however, computing diffeomorphisms to map one image to another is computationally expensive. This

places a significant burden on computational resources, necessitating the acceleration of these algorithms to make them practical for time-sensitive applications. One such application is IGART, where rapid and precise image registration is crucial for adapting treatment plans to changes in patient anatomy, thereby enhancing treatment efficacy and patient safety [49].

Research Question: *In IGART, DIR is a highly non-linear and ill-conditioned inverse problem. Given the extensive computational demands required for accurate DIR, how can we achieve significant acceleration while maintaining registration accuracy?*

This research question underscores the challenges of achieving an optimal balance between speed and accuracy in DIR, particularly within the context of IGART, where real-time processing is crucial. The following sections review various approaches from previous studies and explain how these insights informed our selection of the CLAIRE framework.

3.1 CPU-Based Implementations

Over the past decade, a wide variety of methods for robust image registration have been proposed. Popular CPU implementation packages for diffeomorphic registration, such as Demons [80], Elastix [45], advanced normalization tools (ANTs) [6, 8], diffeomorphic anatomical registration through exponentiated lie algebra algorithm (DARTEL) [5], are publicly available. These packages use multithreading for parallelism, largely based on kernels implemented in the insight toolkit (ITK) package. In general, higher map resolution leads to better registration quality but also increases the complexity of the optimization problem due to a greater number of degrees of freedom. However, these works do not take advantage of distributed memory parallelism, making them less scalable to full resolution. As a result, most codes resort to subsampling to handle large images.

CLAIRE, released alongside the work in [59], features a distributed-memory implementation of an effective solver for constrained large deformation diffeomorphic image registration, utilizing MPI for parallelization. The data are partitioned based on pencil decomposition, with each partition assigned to an MPI task. This enables CLAIRE to target applications of unprecedented scale without requiring data downsampling. Further details of CLAIRE's parallel implementation can be found in references [29, 58]. The authors have demonstrated

CLAIRE's ability to address registration problems involving clinically relevant data sizes, achieving impressive results. By utilizing 20 cores on a standard computer node, CLAIRE completes registration tasks within two to four minutes, ensuring data fidelity. Notably, they accomplished a registration time nearly competitive with the Demons algorithm on the same system while achieving superior registration quality.

Limitation: Although these methods have demonstrated remarkable efficacy in terms of registration accuracy, as evidenced by the cited references [37, 70, 27], they often require a considerable amount of time for execution, even when utilizing high-end multicore CPUs. The computational intensity of DIR renders CPU-based processing impractical for clinical applications.

The challenge is further compounded by the highly non-linear and ill-posed nature of image registration problems, which result in ill-conditioned inversion operators. Consequently, image registration can take several minutes on multicore CPUs. As clinical workflows for multi-center population studies, which require thousands of registrations, become more prevalent, the execution time for each registration becomes increasingly critical. Reducing the runtime to seconds could shorten the duration of clinical studies from weeks to just a few days.

Solution: In recent years, GPUs have been employed in a wide range of applications, achieving remarkable success in accelerating computationally demanding tasks. Given the substantial computational complexity of DIR, particularly with optimization processes that dominate execution time, a GPU-based computing framework is an ideal choice. GPUs are equipped with numerous processing elements designed for pixel-level processing, making them exceptionally effective for handling the computational demands of DIR.

In the following sections, we will examine studies that have utilized single or multiple GPUs as accelerators to understand their impact on performance enhancements in DIR.

3.2 Single GPU Implementations

Two popular packages that leverage GPU acceleration are NiftiReg [63] and Plastimatch [71]. However, it should be noted that neither of these packages supports scalable LDDMM algorithms, and no scaling studies have been reported.

Popular software packages that utilize GPU acceleration for LDDMM registration include Demons [21, 80], DARTEL [77], deformetrica [11], and python for computational anatomy (PyCA)[1]. The runtime reported for Demons is approximately 60 seconds on a Quadro FX 1400 for a dataset of size 128^3. Deformetrica reports registration times of 102 seconds and 202 seconds for two variants of the GPU implementation on an image size of $181 \times 217 \times 181$, executed on an Nvidia Quadro M4000. The GPU variant of PyCA reports a runtime of 648 seconds for $229 \times 193 \times 193$ neuroimaging datasets on an Nvidia TitanX (Pascal) [85].

A recent development involves porting CLAIRE to a single-node, single-GPU implementation, as detailed in reference [13]. CLAIRE deploys highly optimized, mixed-precision GPU kernels for the evaluation of scattered-data interpolation and replaces FFT-based first-order derivatives with optimized eighth-order finite differences (FD). It can register 256^3 clinical images in under 6 seconds on a single NVIDIA Tesla V100. Comparisons with PyCA and Deformetrica reveal that CLAIRE achieves superior registration quality and faster processing speed. Specifically, CLAIRE demonstrated superior performance on three neuroimaging datasets with grid sizes of 256^3, achieving results approximately an order of magnitude better and up to 30 times faster than PyCA. Additionally, Deformetrica was found to be slower than PyCA. Overall, CLAIRE provides over 20 times speedup compared to the CPU version and more than 30 times speedup over existing GPU implementations, establishing it as a robust image registration framework.

Another GPU-accelerated LDDMM implementation, FastReg [32], reports a runtime of approximately 35 seconds for neuroimaging data on a GeForce RTX 2080Ti, achieving an average Dice score of around 0.67. These results are significantly lower than those obtained by CLAIRE.

In the context of analyzing lung data, several studies have employed GPU-accelerated DIR methods, as referenced in the literature [74, 52, 53, 84, 31]. These studies demonstrated enhanced computational performance compared to CPU setups, with registration accuracy remaining consistent. For example, in reference to the DIR-Lab dataset (Cases 1 to 4, with an average size of $\times$ 256 $\times$ 102), the authors of [84] reported an impressive runtime of around 6 seconds, marking the fastest completion time within this dataset. However, a significant challenge

[1] https://bitbucket.org/scicompanat/pyca

in these studies is the lack of publicly available implementations, hindering the replication and validation of their results.

Furthermore, a GPU implementation of the Demons algorithm is detailed in [33, 64], demonstrating a significant increase in processing speed compared to its CPU implementation. For five sets of pulmonary 4DCT images, with an average size of $256 \times 256 \times 100$, the GPU-based DIR computation required approximately 7 to 11 seconds, as detailed in [33].

Limitation: Although utilizing GPUs for computation in DIR applications significantly accelerates the process, the registration remains time-consuming. Single GPU-based DIR implementations still fall short of meeting the speed requirements for time-sensitive applications.

Solution: To further enhance parallelism and reduce computation time, leveraging multiple GPUs offers a promising solution. Distributing the workload across several GPUs allows for the utilization of their combined computational power, significantly speeding up the process. By efficiently managing computational tasks in DIR, this approach can better meet the time-critical demands of applications such as IGART.

3.3 Multi-GPU Implementations

Despite the need for higher computational throughput and the availability of several software packages for LDDMM, there is limited work on multi-GPU implementations tailored for large-scale applications. A multi-GPU implementation of the LDDMM approach from [42] is presented in references [43, 34]. Additionally, reference [78] features a multi-GPU implementation of the LDDMM approach introduced in [77].

These studies investigate the potential applications of DIR frameworks in various image processing problems that require substantial computational resources and large datasets, particularly in brain imaging. To satisfy these demands, they propose multi-node, multi-GPU implementations, focusing on atlas construction from multiple image volumes. Although they optimize computational throughput on a single GPU, their approach emphasizes data parallelism, whereby multiple input images are loaded and processed simultaneously on different GPUs. The solver in [43] completes its runtime in approximately 12 seconds on a single NVIDIA Quadro FX5600 when processing a dataset of 256^3. The

multi-GPU implementations in these studies are designed to process multiple images simultaneously rather than focusing on single image registration.

In contrast, CLAIRE introduces a multi-node multi-GPU framework specifically optimized for high computational throughput in single registration problems, as detailed in [14, 39]. This implementation employs direct device communication through CUDA-aware MPI for fundamental computational kernels, including interpolation for solving transport equations, high-order finite difference operators for differentiation, and FFTs. This approach minimizes communication between the host and the device.

Several key optimizations for multi-GPU architectures include:

- Replacing FFT-based (spectral) first-order derivative evaluations with an 8th-order FD scheme, enhancing accuracy and reducing communication.
- Using texture-based Lagrange polynomial third-order interpolation instead of spline interpolation (which was superior on a single GPU) to minimize GPU-to-GPU communication.
- Combining cuFFT within nodes with a 2D slab decomposition—a method that partitions data into smaller slices for parallel processing—alongside an optimized all-to-all communication scheme across nodes.

In DIR, the solver typically incorporates a preconditioner to enhance performance. A preconditioner is a mathematical tool that improves the convergence rate of iterative solvers by transforming the original problem into a more easily solvable form, facilitating quicker convergence in iterative methods. One widely used approach is the preconditioned conjugate gradient (PCG) method, which extends the standard conjugate gradient method by integrating a preconditioner to enhance efficiency in finding solutions. In the multi-GPU implementation of CLAIRE, a novel preconditioner based on a zero-velocity approximation of the Hessian operator was introduced. The Hessian operator is a matrix that describes the curvature of the image data, providing information on how the image intensity changes in different directions. The zero-velocity approximation simplifies this by assuming that the velocity is negligible at certain points, which helps in estimating the deformations more efficiently. This new preconditioner reduces the number of PCG iterations required within a Gauss–Newton–Krylov scheme, leading to a runtime reduction of up to 2.5 times compared to the previous version of CLAIRE. Their optimized memory footprint enables solving

larger problems on a single GPU, addressing problems of unprecedented scale. Combined, these improvements result in up to a 70% speedup compared to the results presented in [13] on a single GPU.

In their smallest run, they registered 256^3 resolution images in an average of 5 seconds. The largest run involved the registration of 2048^3 resolution images, comprising 25 billion unknowns—approximately 152 times larger than the largest problem solved by state-of-the-art GPU implementations. This run utilized 64 nodes with 256 GPUs on TACC's Longhorn system. The accuracy achieved was consistent with prior CLAIRE results [59, 13]. Notably, the highest Dice score of 0.86 was achieved by utilizing na01 as the reference image and na02 as the template image from the NIREP dataset, compared to a score of 0.56 before registration.

A comprehensive summary of all CLAIRE-related publications can be found in [12].

3.4 Discussion

The research on multi-GPU-based image registration for RT applications has been relatively limited due to the complex challenges associated with data partitioning, memory allocation, and data movement. The absence of a unified approach to address these issues has resulted in a lag in the advancement of clinical RT applications compared to the rapid developments in computer science.

The multi-GPU implementation of CLAIRE has demonstrated robust performance in brain image registration, outperforming other frameworks. Importantly, CLAIRE is available as open-source software, which facilitates its broad use and further development.

In recent years, there has been a notable increase in the size of medical images. A decade ago, structural MR images of the human brain typically had voxel sizes of $2 \times 2 \times 2\ mm^3$, which were considered state-of-the-art at that time. Today, voxel sizes smaller than $1 \times 1 \times 1\ mm^3$ are common, resulting in data volumes that are approximately one order of magnitude larger. The field of microscopy has also seen significant growth, with gigabytes of high-resolution imaging data generated through techniques such as 3D imaging via tissue clearing [86]. Consequently, CLAIRE has become the optimal framework for efficiently handling such large-scale problems.

In light of these considerations, the subsequent chapter will assess the potential for effectively applying CLAIRE to lung images. Lung images differ substantially from brain images in terms of their characteristics, necessitating this focused evaluation. Addressing complex nonlinear deformations in lung image registration may present unique challenges, which will be explored in further detail.

CHAPTER 4

Benchmarking CLAIRE and ANTs: Accuracy and Performance Analysis

4.1 Introduction

The previous chapters have highlighted the adaptability and efficacy of multi-GPU implementation of the CLAIRE framework in handling the complexities of brain image registration. This robust performance, coupled with superior results over other open-source frameworks, prompts further investigation into its potential for lung image registration, which presents its own unique challenges due to the distinct nature of lung images.

In this chapter, we will begin by examining the formulation, dependencies on other tools, and methodological approach of the CLAIRE framework, outlining its core algorithms and computational dependencies. We will then discuss the metrics used to evaluate the effectiveness of the DIR, specifically within the context of lung images, to ensure a comprehensive understanding of the evaluation process. The contributions of this chapter are outlined as follows:

- **Parameter tuning:** Given the promising results of multi-GPU DIR for brain images based on CLAIRE, this section is concerned with assessing whether this method can also be applied to lung images, which differ substantially in their characteristics from brain images. To this end, we consider an extensive parameter search for optimal execution.

- **Assessing the accuracy and the registration time:** We report performance and accuracy for single- and multi-GPU experiments, comparing both metrics to a well-chosen baseline.
- **Scalability:** Given the vast inherent amount of parallelism in optimization-based DIR, we afterwards investigate the strong scaling behavior of these DIR methods and show how time scales with (1) the number of processors, and (2) image dimension (problem size).

4.2 Overview of CLAIRE

CLAIRE is a high-performance $C/C++$ software tool designed for three-dimensional diffeomorphic image registration based on velocity fields. It employs formulations related to LDDMM methods, which excel at handling significant deformations. LDDMM operates on the principle of diffeomorphic transformations to ensure smooth and invertible mappings between images. Such transformations are critical as they ensure that the registered images retain their geometric properties, providing a continuous and reversible path from one image to another throughout the registration process [10], [75]. Although LDDMM methods are computationally intensive due to their infinite-dimensional nature and the highly non-linear and ill-posed characteristics of the registration problem, CLAIRE has been optimized for advanced computing architectures. It extends its performance beyond multicore CPU systems, leveraging the computational power of multi-node, multi-GPU configurations. This optimization enables CLAIRE to deliver faster and more efficient image registration capabilities, making it suitable for handling complex registration tasks in modern medical imaging applications.

4.2.1 Formulation of the CLAIRE Framework

Like most existing algorithms for deformable registration, CLAIRE iteratively optimizes a transformation based on a similarity measure. The deformation map $y(x)$ is parameterized through a smooth, stationary velocity field $v(x)$. It uses an optimal control technique to solve the optimization problem by seeking $v(x)$ that produces the desired spatial transformation on a three-dimensional rectangular domain $\Omega := [0, 2\pi)^3 \subset \mathbb{R}^3$ with periodic boundary conditions on $\partial\Omega$. Let x denote the spatial coordinate; $(x_1, x_2, x_3)^T \in \mathbb{R}^3$, and $m(x,t)$ is the state

variable that transports intensities of m_0. The optimization problem with two images $m_0(x)$ (template image, image to be deformed) and $m_1(x)$ (reference image) can be stated as:

$$\underset{v,m}{\text{minimize}} \quad \frac{1}{2}\int_\Omega (m(x,1)-m_1(x))^2\,dx + \frac{\beta}{2} reg(v) \tag{4.1}$$

subject to

$$\begin{aligned} \partial_t m(x,t) + v(x)\cdot\nabla m(x,t) &= 0 \qquad \text{in} \quad \Omega\times(0,1], \\ m(x,t) &= m_0(x) \qquad \text{in} \quad \Omega\times\{0\} \end{aligned} \tag{4.2}$$

The first term in (4.1) measures image similarity between the deformed template image $m(x,1)$ and the reference image $m_1(x)$. They consider a squared L^2-distance, without loss of generality. The second term in (4.1) is a regularization term with a regularization trade-off parameter $\beta > 0$ to ensure the smoothness of the velocity field. This, in turn, ensures that the resulting geometric transformation of the template image retains the properties of the diffeomorphism. The transport equation (4.2) describes the geometric transformation of the template image $m_0(x)$. A reduced-space Gauss–Newton–Krylov method is used to solve (4.1).

4.2.2 System Requirements

CLAIRE utilizes a suite of software packages, each contributing essential functionalities to its image registration capabilities. For linear algebra operations, CLAIRE employs the portable, extensible toolkit for scientific computation library (PETSc), a powerful and scalable tool in scientific computing, along with PETSc's toolkit for advanced optimization (TAO) package, which specializes in the optimization of nonlinear problems[1]. For GPU-accelerated computations, CLAIRE integrates CUDA[2] and thrust[3], along with cuFFT for FFTs[4]. Additionally, it uses zlib[5] for the compression and decompression of data, adding an extra layer of efficiency in handling large imaging datasets.

[1] https://petsc.org/release/manual/manual.pdf
[2] https://developer.nvidia.com/cuda-downloads
[3] https://docs.nvidia.com/cuda/thrust/index.html
[4] https://docs.nvidia.com/cuda/cufft/index.html
[5] http://zlib.net

Input/Output operations are managed using niftilib, which allows for effective reading and writing of medical imaging data[1]. In the realm of parallel computing and communication, CLAIRE is compatible with IBM Spectrum MPI, which offers GPU support through CUDA-aware MPI, particularly beneficial for the multi-GPU version[2]. However, it is important to note that in our executions, we utilize Open MPI, a widely adopted MPI implementation known for its high performance and robustness[3]. The source code is accessible on GitHub and can be downloaded from https://github.com/andreasmang/claire. This repository also cites all the necessary packages, providing a comprehensive resource for users to fully leverage CLAIRE.

4.2.3 Data Specifications

The authors present findings using the Non-Rigid Image Registration Evaluation Project (NIREP) dataset, as referenced in [20]. This dataset comprises 16 T1-weighted MRI brain scans from different individuals (labeled na01 to na16), which have been rigidly aligned. Each of these scans has an image size of 256 $\times$ 300 $\times$ 256 voxels, providing a substantial and consistent basis for assessing non-rigid image registration techniques. Each dataset is annotated with a label map that identifies 32 gray matter regions (ground truth segmentations).

4.2.4 Evaluation Metrics

This section is dedicated to exploring the diverse range of metrics employed in CLAIRE, shedding light on their application and relevance. We will also discuss our specific usage of these metrics, offering insights into how they contribute to the robust assessment and enhancement of image registration processes.

4.2.4.1 Dice Score Analysis

To evaluate the precision of the registration outcomes, brain registration accuracy is quantified utilizing the Dice score, which is calculated for each of the 32 individual labels. This coefficient is computed for the union of all gray matter labels associated with the data sets. This 'union' method ensures that the evaluation of

[1] http://niftilib.sourceforge.net
[2] https://www.ibm.com/us-en/marketplace/spectrum-mpi
[3] https://docs.open-mpi.org/en/v5.0.x/

registration efficacy is not disproportionately influenced by high performance in only a few regions, thereby providing a balanced and comprehensive assessment of registration quality.

In line with this methodology, our study employs a comparable approach for lung image analysis. We initiate this process by segmenting each lobe using an automated segmentation tool. Subsequently, we compute the Dice score for the segmentation of each lobe in comparison to a predefined ground truth. Our analysis proceeds in a lobe-specific manner, calculating the Dice score independently for each lobe. Finally, an average Dice score is derived to represent the overall accuracy across the entire lung region.

4.2.4.2 Analysis of Relative Mismatch

Relative mismatch is a pivotal metric utilized within CLAIRE for assessing the accuracy of the image registration process, offering essential insights into the success of the transformations applied [13]. The computation of this metric is given by the following formula:

$$\text{Relative mismatch} = \frac{\|m(\cdot,1)-m_1\|_2}{\|m_1-m_0\|_2}$$

Here, $m_0(x)$ is the template image, $m_1(x)$ is the reference image, $m(x,1)$ is the transformed template image. The transformation is determined by the forward problem described in equation (4.2). The notation $||\cdot||_2$ indicates the Euclidean (L2) norm, providing a measure of the magnitude of the difference between the images before and after the transformation. Essentially, the relative mismatch provides a measure of how closely the registered image (transformed template) matches the reference image, relative to their initial dissimilarity.

In our study, we also evaluate this metric, which is an automated aspect of the CLAIRE framework's functionality.

4.2.4.3 Techniques in Data Visualization

Visualization plays a crucial role in understanding and evaluating image registration outcomes. Effective visualization transforms complex data into a more comprehensible and actionable format, enabling deeper insights and more informed decision-making.

The CLAIRE framework employs a range of visualization techniques, including the depiction of mismatch, visualization of the velocity field $v(x)$, mapping of the

determinant of the deformation gradient, and representation of the deformed grid to illustrate the in-plane components of the deformation map $y(x)$. Figure 4.1 showcases these exemplary results.

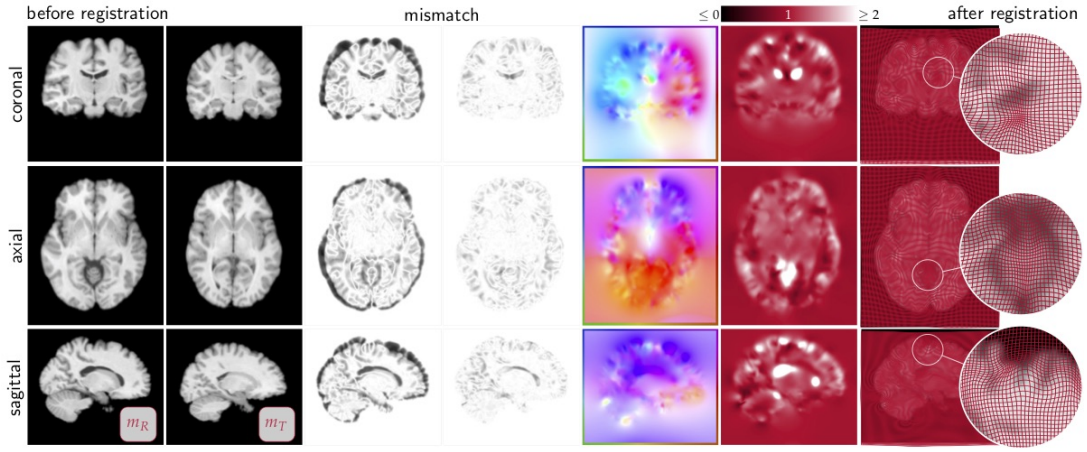


Figure 4.1 Exemplary results using the NIREP dataset, with na10 as the template image and na01 as the reference image. The three columns on the left display the original data. The four columns on the right show a map of the orientation of $v(x)$, a map of the determinant of the deformation gradient (with the color bar shown at the top), and a deformed grid illustrating the in-plane components of $y(x)$, respectively [59].

In our study, we utilized techniques that include color-coding regions of mismatch and generating overlay images, which compare the results of registration with the original images. We leveraged tools such as 3D Slicer and medical image processing, analysis, and visualization (MIPAV), which provide advanced features for visualizing disparities and overlays in medical imaging [24],[61]. Furthermore, by leveraging the post-processing features of CLAIRE, we visualized the evolution of the deformed images at various stages of the registration process, until the solver reached convergence.

4.3 Optimizing CLAIRE's Parameters

To determine the optimal configuration for the lung dataset, we performed extensive parameter tuning. This section is structured into four subsections:

- **Optimization methods explored** - Discusses the various optimization techniques we evaluated to enhance the performance of CLAIRE.

- **Determining the optimal regularization parameter** - Focuses on the process of finding the best regularization parameter for accurate and stable results.
- **Image orientation effects** - Examines how different image orientations impact the registration outcomes.
- **Summary of parameter tuning outcomes** - Summarizes the results of our parameter tuning efforts, highlighting the key findings and their implications for lung image registration.

Each subsection delves into specific aspects of the optimization process, contributing to the overall goal of fine-tuning CLAIRE for lung image registration.

4.3.1 Optimization Methods Explored

CLAIRE employs a Gauss–Newton–Krylov solver, which is globalized using an Armijo line search to determine the new iteration [59]. The framework utilizes three distinct methods to execute its solver and address the optimization problem:

- **Grid continuation:** This method adopts a multi-resolution approach. It begins by solving the problem at a lower resolution and progressively refines the resolution of images, thereby enhancing the solution's accuracy.
- **Scaling continuation:** Scaling, alternatively known as continuation in the smoothness of images, involves a multiscale strategy. It incrementally smooths images, which aids in simplifying and then gradually solving the optimization problem.
- **Parameter continuation (PC):** This scheme involves solving the registration problem for a sequence of decreasing values for β. For each new value, the velocity obtained in the previous step serves as the initial estimate for the Gauss-Newton-Krylov solver. In other words, the inversion is run iteratively to convergence, using a sequence of decreasing regularization parameters, until the target regularization parameter is reached.

They reported that the PC scheme achieves the best performance, yielding a speedup of $4\times$ to $6\times$ compared to a full solution [59]. For an in-depth discussion of these methods, see [57].

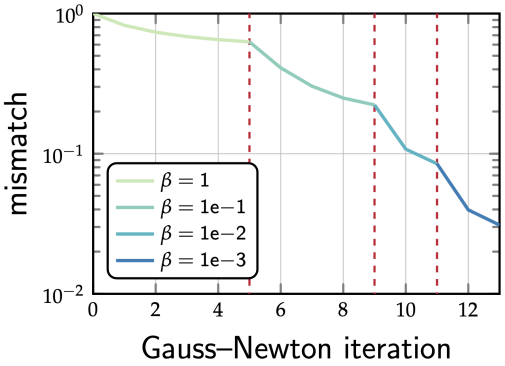


Figure 4.2 Convergence results for the CLAIRE parameter continuation scheme. Exemplary result for brain dataset: Registration of na11 to na01 (256 × 300 × 256) [59]. The figure illustrates the reduction of the mismatch versus the cumulative number of Gauss–Newton iterations, with individual levels indicated by vertical dashed lines.

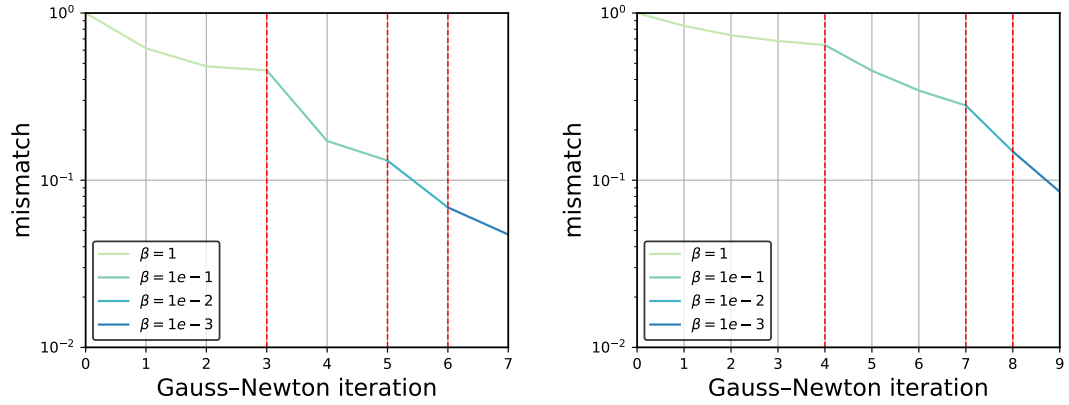
They showcased the trend of the mismatch for different levels of the PC scheme in Figure 4.2, which displays exemplary convergence results. For detailed results and analysis, refer to [59].

Observation: In our study, the implementation of a PC scheme proved effective with our dataset. Figure 4.3 displays exemplary convergence results, clearly showing that the solver required only four levels of iteration to converge in two different cases. Specifically, in Case 6, the number of iterations at each level was 4, 3, 1, and 1, respectively, while in Case 1, the iterations were 3, 2, 1, and 1.

4.3.2 Determining the Optimal Regularization Parameter

We aim to find a trade-off between maximizing assessment metrics, specifically the Dice score which reflects registration accuracy, and minimizing computational time. To achieve this, we conduct experiments that vary the regularization parameter to identify a value that remains consistent across the entire dataset. In Figure 4.4, we present our exploration of regularization parameters to achieve consistency across both small and large datasets in lung image registration.

Observation: Our findings indicate that a target parameter value of $\beta = 5e{-}3$ yields the highest efficacy across all examined datasets. This value represents an increase from the parameter $\beta = 5e{-}4$ suggested in [14].



(a) Analysis of Case 1 (256 × 256 × 94) **(b)** Analysis of Case 6 (512 × 512 × 128)

Figure 4.3 Convergence results for the CLAIRE parameter continuation scheme, showcased with lung datasets.

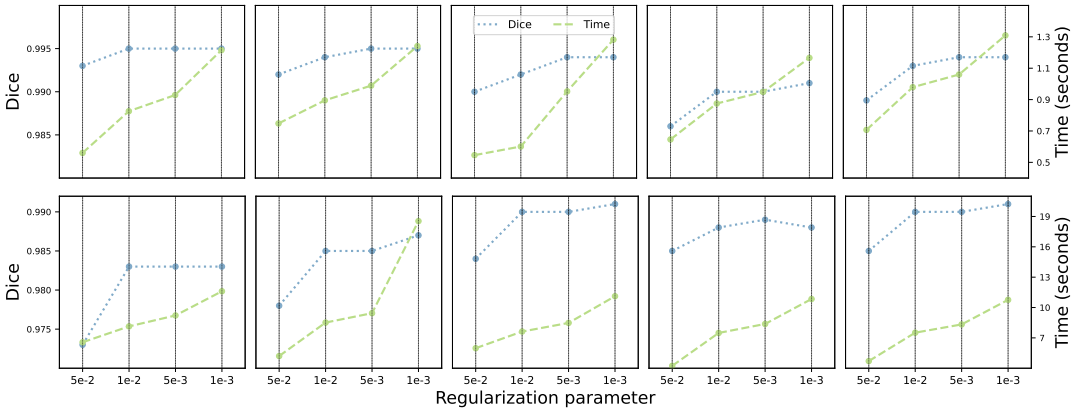


Figure 4.4 Optimizing regularization parameters for lung dataset consistency (Case 1 to Case 10). During our experimentation, we adjusted the regularization parameter to explore different values ($\beta = 5e{-}2$, $1e{-}2$, $5e{-}3$, $1e{-}3$) to find an optimal value that would be consistent across all datasets. Our goal is to find a balance between the Dice score and time. Through our investigations, we identified $5e{-}3$ as the optimal value for this trade-off.

4.3.3 Image Orientation Effects

The orientation of geometry relative to the coordinate frame can significantly affect performance, particularly in terms of memory ordering and computational efficiency. This is particularly relevant in medical imaging, where images can vary greatly in terms of orientation and patient positioning.

Ensuring consistent orientation and alignment before initiating the registration process is crucial. Variations in orientation can impact how data is organized in memory, affecting access patterns and computational speed. Moreover, alignment consistency is essential for the accuracy and efficacy of the registration, as it influences how image data is interpreted and processed. Proper alignment ensures that the computational algorithms function optimally, minimizing errors and enhancing the reliability of the results.

In this study, we evaluate the effects of varying image orientations on the accuracy and runtime of DIR. To assess the robustness of registration, we applied different orientation adjustments to the images. Our findings underscore that specific orientations can significantly influence the accuracy and performance of the registration outcomes.

Observation: Our findings indicate that employing the Right-Anterior-Superior (RAS) convention, along with the right/left (RL), anterior/posterior (AP), and superior/inferior (SI) orientation formats (Figure 4.5), significantly improves the efficiency of lung image registration. Although it may not yield the absolute fastest processing times, it performs better than many other conventions and achieves acceptable speeds. Moreover, adopting RAS as the preferred orientation convention offers several key advantages, including compatibility with a wide array of analytical and visualization tools specifically designed for this orientation. A notable benefit of using RAS is its alignment with the orientation used in neuroimaging informatics technology initiative (NIfTI) images, which predominantly employ the RAS convention. This compatibility is particularly beneficial, as it eliminates the need for complex data manipulation, thereby enhancing the usability of the CLAIRE framework, which is designed to work seamlessly with NIfTI image formats.

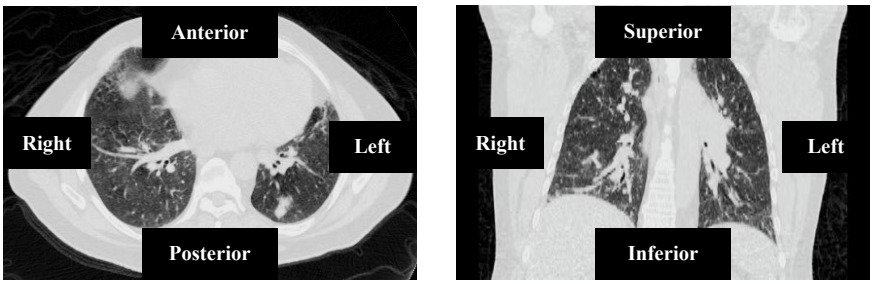


Figure 4.5 Optimal orientation formats for lung image registration: axial and coronal views (4DCT Case 6 -T00).

4.3.4 Summary of Parameter Tuning Outcomes

In summary, our research indicates that the implementation of the PC scheme for solving the optimization problem, coupled with a finely-tuned regularization parameter set at $\beta = 5\text{e}{-3}$, and the adoption of the RAS orientation convention, has been finely optimized across all image sets in the Dir-Lab dataset. This tailored approach has proven to be highly effective in enhancing the accuracy and efficiency of our lung image registration processes. Regarding other critical parameters, such as the distance measure and optimization method, we have observed that adhering to the guidelines proposed by the original authors leads to optimal outcomes. More details are provided in Appendix A. A significant finding from our study indicates that the conventional approach in deformable registration—starting with an initial affine transformation for global alignment before proceeding to more complex deformable transformations— is not necessary for lung datasets. This discovery notably simplifies the registration process by eliminating a commonly required preliminary phase, streamlining the workflow in deformable registration.

4.4 Comparative Analysis: CLAIRE vs. Deformable ANTs

In this section, we conduct a comparative analysis of the accuracy and performance between CLAIRE's single-GPU implementation and a carefully selected baseline—deformable ANTs—using a lung dataset. To ensure a fair and equitable comparison, we emphasize that extensive efforts were made to implement and fine-tune the ANTs package. This optimization was crucial for identifying the

configuration necessary to achieve the highest level of accuracy in registering our datasets. The details of this optimization process will be discussed in section 4.4.2.

4.4.1 Experimental Setup

This section details the essential steps in our workflow, crucial for the effective processing and analysis of the Dir-Lab dataset. It encompasses converting image formats to be compatible with our tools, tuning parameters for optimal performance, enhancing image clarity for precise segmentation, and evaluating the segmentation. Each step is integral to ensuring both the accuracy and efficiency of our study.

- **Image format conversion:** The images in the Dir-Lab dataset were originally provided in the Pinnacle TPS raw $*$.img format. To ensure compatibility with our suite of software tools — CLAIRE, ANTs, and TotalSegmentator [82] — we performed a format conversion process. This task was carried out within a Matlab environment, leveraging the image processing toolbox along with tools for NIfTI and ANALYZE image formats [1]. This essential conversion step ensured that the images were in formats suitable for the processing and analysis requirements of our study.
- **Parameter tuning:** As detailed in the previous sections, we conducted a thorough parameter tuning process. This involved adjusting and optimizing various parameters to enhance performance and accuracy.
- **Enhancing clarity of deformed images:** The initial deformed images outputted by CLAIRE exhibit blurriness, making them unsuitable for processing with TotalSegmentator, an automatic segmentation tool. To address this, we developed a script that forms a core component of our workflow. This script manipulates deformation map fields to reposition each point of a template image to new coordinates based on these fields. We obtain these deformation maps from CLAIRE using the `-defmap` option, which indicates transformations across three dimensions. Advanced interpolation techniques, implemented via the SciPy library [2], facilitate this mapping.

[1] https://www.mathworks.com/matlabcentral/fileexchange/8797-tools-for-nifti-and-analyze-image

Our script also utilizes NiBabel[1] for handling NIfTI neuroimaging data formats and optionally employs SimpleITK for further image processing tasks [87]. Consequently, this process yields a deformed image with significantly improved clarity and reduced blurriness. The resulting image quality is thus enhanced, making it compatible for segmentation with TotalSegmentator. This step is crucial as it ensures accurate image segmentation and analysis in the later stages of our workflow.

- **Segmentation:** We utilized TotalSegmentator to generate binary masks, leveraging its capacity for automatic and robust segmentation of major anatomical structures in body CT images. This tool operates using a DL segmentation model from the nnU-Net framework, a U-Net–based system designed to autoconfigure hyperparameters based on dataset characteristics [40]. With its demonstrated high accuracy, TotalSegmentator has shown superior performance, achieving a Dice score of 0.943 across a diverse range of clinical data, including cases with major abnormalities [82]. It is also compatible with NIfTI format images and supports both CPU and GPU environments.

  In our study, we opted for the model trained at a 1.5 mm resolution due to its superior accuracy. In contrast, the 3-mm resolution model, while yielding acceptable outcomes, showed less precision in border definition. Since segmentation was conducted as a post-registration step in our methodology, we did not primarily focus on concerns such as runtime, GPU memory, and RAM requirements. For the segmentation, we obtained individually segmented lung lobes to assess their respective accuracy. For a comprehensive evaluation of the entire lung, we combined these segmented lobes into a single binary mask, which was then utilized to evaluate the overall accuracy.

- **Segmentation evaluation:** After obtaining the deformed image and the ground truth segmentations, we computed the Dice score to evaluate registration accuracy. This calculation was performed using the overlap measure filter provided by SimpleITK, which facilitates precise and reliable assessment of the overlap between the segmented and the ground truth images.

[1] https://nipy.org/nibabel/

4.4.2 Overview of Deformable ANTs

Advanced Normalization Tools (ANTs) is a versatile software suite designed primarily for image registration and normalization, along with other image processing tasks, with a particular focus on medical imaging. Built largely on kernels implemented in the ITK package, ANTs excels due to its robust and accurate image registration algorithms, which are widely recognized for their effectiveness [6, 7, 76].

ANTs is a prominent tool used extensively in multimodal neuroimaging and lung imaging. It supports various image formats, including NIfTI, and offers a range of transformation models with adjustable complexity, regularization, and degrees of freedom. The spectrum of these models begins with simple translations, extends through rigid and/or affine transformations, and concludes with the most complex and versatile model: the diffeomorphic symmetric normalization (SyN). The feature of SyN lies in its optimization and integration of time-varying velocity fields, which leads to registration that is symmetric and accurate, as well as invertible. These attributes make the SyN model highly appealing, prompting us to investigate its potential for delivering robust registration.

Given these capabilities, ANTs stands out as a robust and suitable framework for image registration, comparable with CLAIRE in terms of registration accuracy. Moreover, our choice of ANTs as the baseline method is influenced by its widespread application in medical imaging, notably in RT planning and image-guided surgery, underscoring its relevance and utility in diverse clinical contexts.

In our study to determine the most effective ANTs configuration, we employed the `antRegistrationSyN` script, with a particular focus on two essential aspects[1]:

- **Masks:** Utilizing masks in ANTs allows for targeted registration on specific ROIs, enhancing accuracy by excluding irrelevant areas and increasing computational efficiency. We analyzed results with and without the use of masks.
- **Transform type:** Our exploration centered on the 's' (rigid + affine + deformable SyN) and 'so' (deformable SyN only) transform types.

[1] https://github.com/ANTsX/ANTs/blob/master/Scripts/antsRegistrationSyN.sh

Table 4.1 Comparative analysis of Dice scores for various transform types in ANTs registration, evaluated with and without the use of masks.

Dataset	s/woM	s/wM	so/woM	so/wM
Case1	0.991	0.993	0.990	0.989
Case2	0.989	0.992	0.989	0.988
Case3	0.990	0.992	0.989	0.988
Case4	0.986	0.989	0.985	0.987
Case5	0.985	0.991	0.985	0.987
Case6	0.971	0.983	0.968	0.973
Case7	0.974	0.985	0.975	0.975
Case8	0.979	0.987	0.976	0.981
Case9	0.981	0.987	0.980	0.984
Case10	0.984	0.987	0.979	0.982
Average	**0.983**	**0.986**	**0.982**	**0.983**

Other parameters, such as the similarity measurement (neighborhood cross-correlation), were kept at their default settings.

Observation: Our observations show that only the 's' and 'so' transform types achieved acceptable performance levels. The 's' transform was found to be more accurate, despite having a slightly longer runtime, which was considered negligible due to its higher accuracy. Additionally, using a mask significantly improved both runtime and accuracy. These results are detailed in Table 4.1 and Table 4.2.

The extended registration times observed for ANTs are largely attributable to its CPU-based implementation. Despite this, accuracy remains a paramount consideration in our evaluations. ANTs consistently achieved Dice scores across various configurations that meet or often exceed clinical acceptability standards. Notably, for the 's' transform type with a mask, the Dice scores reached an impressive 0.986, indicating the highest degree of accuracy.

In the context of RT applications, it is crucial to acknowledge that Dice scores ranging from 0.8 to 0.9 are clinically acceptable, as evidenced in various studies [46, 47, 62]. For instance, a study involving 12 lung cancer patients reported average Dice scores consistently above 0.8 [47]. Similarly, another study assessing three different CT thorax datasets found remarkably high mean Dice scores for both right and left lungs, at 0.960 [62].

This optimal configuration of ANTs, identified through our analysis, will be

Table 4.2 Comparative analysis of registration times (second) for various transform types in ANTs registration, evaluated with and without the use of masks.

Dataset	s/woM	s/wM	so/woM	so/wM
Case1	155	113	145	107
Case2	171	130	155	124
Case3	167	124	152	120
Case4	163	115	151	110
Case5	167	118	155	116
Case6	874	519	856	530
Case7	1061	614	1049	625
Case8	901	543	864	401
Case9	883	505	845	517
Case10	838	516	814	503
Average	**538**	**329**	**519**	**316**

compared with CLAIRE in the following section for a comparative evaluation.

4.4.3 Comparative Performance Evaluation

In Table 4.3, we present the performance comparison between the single GPU-based CLAIRE and ANTs. To ensure fairness, we utilized the optimal configuration of ANTs for the most accurate registration, as detailed earlier. Our evaluation considers scenarios both with and without the use of binary masks in the registration process. Interestingly, while ANTs shows improved performance in both time and accuracy with the inclusion of a mask, CLAIRE demonstrates a contrasting behavior. Specifically, using a mask with CLAIRE reduces the registration time but at the cost of decreased accuracy. Consequently, we calculated the Dice difference and a speedup factor, defined as the ratio of the registration time of ANTs with a mask to that of CLAIRE without a mask.

Remarkably, CLAIRE achieves equivalent or higher accuracy and significantly faster registration times. Specifically, it shows a speedup factor of 120 on a single GPU system for the small dataset and 61 for the large dataset when compared to ANTs' CPU-based implementation. This comparison highlights CLAIRE's efficiency and accuracy in the challenging task of nonlinear lung image registration, affirming its substantial value in medical imaging and research contexts.

Table 4.3 Comparative analysis of CLAIRE versus ANTs registration methods, with and without masks: Assessing registration time in seconds and the Dice score, where higher values indicate better performance. The Speedup factor and Dice difference are calculated by comparing CLAIRE without mask (w/oM) and ANTs with mask (wM), which are the optimal configurations. CLAIRE is implemented on a single GPU, whereas ANTs is implemented on a CPU. The results highlight CLAIRE's efficiency in medical image registration.

Dataset	Dimension	Dice (before)	CLAIRE (w/oM)		CLAIRE (wM)		ANTs (w/oM)		ANTs(wM)		Speedup factor	Dice difference
			Time	Dice	Time	Dice	Time	Dice	Time	Dice		
C1	$256 \times 256 \times 94$	0.951	0.93	0.995	0.67	0.989	155	0.991	113	0.993	122	+0.002
C2	$256 \times 256 \times 112$	0.948	0.98	0.994	0.74	0.990	171	0.989	130	0.992	133	+0.002
C3	$256 \times 256 \times 104$	0.936	0.96	0.993	0.68	0.988	167	0.990	124	0.992	129	+0.001
C4	$256 \times 256 \times 99$	0.917	1.02	0.990	0.71	0.983	163	0.986	115	0.989	113	+0.001
C5	$256 \times 256 \times 106$	0.934	1.13	0.993	0.80	0.987	167	0.985	118	0.991	104	+0.002
Average		**0.937**	**1.00**	**0.993**	**0.72**	**0.987**	**165**	**0.988**	**120**	**0.991**	**120**	**+0.002**
C6	$512 \times 512 \times 128$	0.864	9.19	0.983	4.20	0.978	874	0.971	519	0.983	56	+0.000
C7	$512 \times 512 \times 136$	0.892	9.42	0.985	4.18	0.978	1061	0.974	614	0.985	65	+0.000
C8	$512 \times 512 \times 128$	0.893	8.48	0.989	5.13	0.986	901	0.979	543	0.987	64	+0.002
C9	$512 \times 512 \times 128$	0.913	8.40	0.988	3.40	0.980	883	0.981	505	0.987	60	+0.001
C10	$512 \times 512 \times 120$	0.906	8.32	0.990	4.18	0.983	838	0.984	516	0.987	62	+0.003
Average		**0.901**	**8.76**	**0.987**	**4.22**	**0.981**	**911**	**0.978**	**539**	**0.986**	**61**	**+0.001**

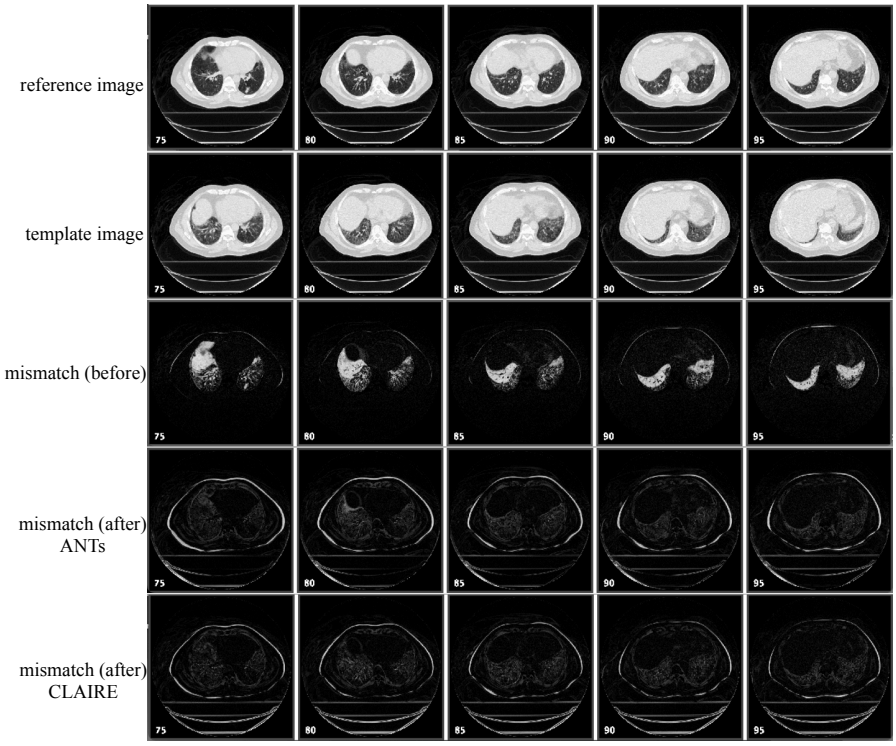


Figure 4.6 Visualization of registration results for CLAIRE and ANTs: Visual representations for Case 6, featuring slices 75, 80, 85, and 90 to showcase the registration results. The top-to-bottom sequence shows the input images (the reference and template images) and their differences before registration, where black areas indicate a small residual, and white areas indicate a large residual. This is followed by the difference visualizations after registration as achieved with CLAIRE, and ANTs. These visual comparisons underscore the effectiveness of CLAIRE in achieving high registration accuracy.

We further compared CLAIRE and ANTs by employing visualization techniques. This comparison is illustrated in Figure 4.6. The figure presents axial slices of Case 6 to compare the registration mismatches using ANTs and CLAIRE. Overall, both tools demonstrate a high level of registration accuracy. Notably, in slice 80, there is evidence of marginally better registration accuracy with CLAIRE as opposed to ANTs. This suggests that CLAIRE may more precisely handle the nuances of lung tissue deformation, at least in this particular slice.

We conducted two additional experiments to assess the Dice score in greater detail: one focusing on individual lung lobes and the other on the right and left lungs separately. The results of these experiments, particularly for Case 6, are detailed in Table 4.4.

The table reveals that the overall Dice score for all individual lobes combined

is lower than the Dice score for the entire lung, both before and after registration. This discrepancy arises because the Dice score for smaller regions like individual lobes is more sensitive to minor misalignment and errors. Small inaccuracies in boundary alignment can significantly impact the Dice score in these smaller regions, while the same inaccuracies may have a less pronounced effect on the entire lung due to its larger size. Furthermore, the variability in size and complexity of individual lobes, especially those smaller or irregularly shaped, poses additional challenges in achieving accurate registration, resulting in lower individual Dice scores compared to the more uniform whole lung area. However, the table confirms that both experiments yielded Dice scores within acceptable ranges, demonstrating that the CLAIRE framework performs reliably across various scales and complexities of anatomical structures.

Table 4.4 Comparison of Dice score before and after registration for Case 6: This table shows results from two assessments: (1) individual lung lobes and (2) right/left lungs.

Mask area	Dice (Before)	Dice (After)
Lung Lobes		
LLL	0.791	0.915
LUL	0.864	0.953
RLL	0.719	0.968
RML	0.692	0.957
RUL	0.883	0.977
Right/Left Lung		
RL	0.858	0.988
LL	0.872	0.983

In Figure 4.7, we showcase an overlay visualization generated using 3D Slicer, illustrating the precise alignment achieved through the registration process using CLAIRE. To validate our results, we selected Case 6 from the Dir-Lab dataset, which is characterized by significant lung deformations, making it a particularly challenging and informative case for evaluating image registration algorithms [26]. The figure displays 33 out of the 128 slices from this case, specifically chosen to prominently feature the lung region, providing a comprehensive view of the registration effectiveness in these critical areas.

4.5 Scalability of CLAIRE: Evaluating performance Across Dataset Sizes

In the previous section, we demonstrated the robustness of CLAIRE compared to ANTs, with a focus on registration times on a single GPU. The results indicated that while effective, the registration process is not yet suitable for real-time applications. In this section, we extend our evaluation of CLAIRE's multi-GPU capabilities. We aim to assess its strong scaling support and to identify any limitations and potential areas for improvement.

4.5.1 Experimental Setup for Scalability Testing

To further analyze CLAIRE's performance, we incorporated two key tools in our experiments:

- **NVIDIA Nsight Systems**: This performance analysis tool is crucial for optimizing applications on systems equipped with NVIDIA GPUs. It provides a detailed system-wide view, helping developers identify bottlenecks in both CPU and GPU operations. This tool is essential for enhancing the efficiency of applications across various domains such as gaming, HPC, and AI, ensuring optimal utilization of hardware resources[1].
- **NVIDIA's Tools Extension SDK (NVTX)**: This C-based API allows for detailed annotations of events and code segments within applications. Its Push/Pop range feature creates a stack that facilitates profiling by marking specific code ranges. The Pop call is automatically linked to the most recent Push on the same thread, which greatly simplifies the tracking and evaluation of complex, multithreaded applications[2].

4.5.2 Evaluation of Strong Scaling Support

While single GPU results for CLAIRE are noteworthy, it still encounters challenges when dealing with computationally intensive tasks in DIR, emphasizing the critical need for faster registrations. Multi-GPU setups excel in this context due to their parallel processing capabilities, proficiently distributing workloads across

[1] https://developer.nvidia.com/nsight-systems
[2] https://docs.nvidia.com/nvtx/index.html

Table 4.5 CLAIRE strong scaling results using 1, 2, 4, and 8 GPUs: In certain datasets, the solver failed to achieve convergence when scaling from 1 GPU to 2, 4, or 8 GPUs. Overall, CLAIRE showed poor scaling. Registration times are in seconds.

Dataset	1 GPU	2 GPUs	4 GPUs	8 GPUs
Case 1	0.93	1.02	-	-
Case 2	0.98	1.07	0.80	-
Case 3	0.96	1.08	0.81	-
Case 4	1.02	-	-	-
Case 5	1.13	1.14	-	-
Average	**1.00**	**1.07**	**0.80**	-
Case 6	9.19	8.78	5.01	28.65
Case 7	9.42	8.81	5.01	28.63
Case 8	8.48	7.97	4.55	26.09
Case 9	8.40	7.87	4.41	25.78
Case 10	8.32	8.02	4.55	26.43
Average	**8.76**	**8.29**	**4.71**	**27.11**

multiple GPUs to reduce computation time and meet the need for increased efficiency.

We investigated the strong scaling behavior of CLAIRE and have presented the results in Table 4.5. The data shows that CLAIRE does not scale effectively with an increase in the number of GPUs. Specifically, in our small dataset Case 4 failed to converge with 2 GPUs, and Case 1, Case 2, and Case 4 failed to converge with 4 GPUs. Additionally, none of the small datasets converged when using 8 GPUs. For large dataset, expanding beyond 4 GPUs introduced significant scalability issues due to the increased off-node communication overhead associated with additional nodes, which substantially prolonged communication times.

To gain deeper insight, we performed a profiling analysis. In this analysis, we report the fraction of time spent on computation time (CUDA kernels) and data transfer time (memory operations). Figure 4.8 shows that CLAIRE significantly reduces computation time, but its scalability suffers from the high communication cost. This highlights that the communication overhead becomes a bottleneck and dominates the overall execution time. Note that the Dice score remains consistent regardless of the number of GPUs used.

To gain a deeper understanding of CUDA kernel and communication times, we utilized NVTX ranges to distinctly mark the durations of kernel and communication times. Figure 4.9 presents an exemplary result for Case 6. Our observations indicate that the majority of the solver's runtime is consumed by the three primary computational kernels: FFTs, interpolation (IP), and FD.

As the number of GPUs increases, there's a noticeable decrease in computation time across all kernels. This is indicative of the tasks benefiting from the parallel processing capabilities of multiple GPUs. However, the increasing proportion of communication time (especially in FFT and IP kernels) as the number of GPUs increases becomes a limiting factor in scalability. As observed, using 2 GPUs, the additional communication costs for strong scaling could not be compensated by the reduced computations per GPU. In our experiment, utilizing up to four GPUs within a single node provided optimal scaling for our clinically relevant problem size. However, while this configuration offered the best scalability achievable with the CLAIRE framework, it still fell short of ideal strong scalability levels.

Given the significant communication overheads in multi-GPU setups, it would be valuable to investigate whether these overheads arise from data transfer, synchronization issues, or other factors. Optimizing data transfer and minimizing synchronization delays could lead to performance improvements. Although using multiple GPUs offers clear benefits, the relative increase in communication time suggests potential scalability limits. A deeper understanding of the FFT and IP tasks could provide insights into why they behave differently with respect to computation and communication times.

- **Interpolation method:** The formula for interpolating at an off-grid query point $\mathbf{x} = (x_1, x_2, x_3)$ is given by

$$f(\mathbf{x}) = \sum_{i=0}^{d} \sum_{j=0}^{d} \sum_{k=0}^{d} f_{ijk} \phi_i(x_1) \phi_j(x_2) \phi_k(x_3),$$

  where f_{ijk} represents the function value at a grid point, d denotes the polynomial order, and ϕ_l, $l = 0, \ldots, d$, are the Lagrange polynomial basis functions.

  CLAIRE employs four different schemes of interpolation. In our implementation, GPU-TXTLIN [14], which employs NVIDIA's texture memory for trilinear interpolation, has proven most efficient. It significantly re-

duces computational overhead by only evaluating 8 neighboring grid points compared to 64 in a typical cubic Lagrange interpolation. This method is further optimized by leveraging NVIDIA's texture units that provide fast access and interpolation capabilities, improving both performance and accuracy as evidenced by consistent Dice scores.

Multi-GPU interpolation kernel distribution: The interpolation task in CLAIRE is distributed across multiple GPUs in a structured multiphase process:

In the setup phase, CLAIRE initializes various parameters, including the maximum allocation size and whether CUDA-aware MPI is used. Memory is allocated for necessary arrays and buffers on both the host and device. This setup includes allocating memory for query points, interpolation results, and communication buffers, and preparing arrays for storing query points, processor indices, and other necessary information.

For data and query point preparation, the global grid is divided into smaller sub-grids, or slabs, each assigned to a different processor, thus parallelizing the workload across multiple GPUs. Each processor identifies its neighboring processors to manage boundary conditions efficiently. CLAIRE determines which query points belong to which processors based on their coordinates, facilitating the distribution of query points to the appropriate processors.

A custom comparison function sorts the query points based on their spatial coordinates. The sorting of query points improves cache efficiency and memory access patterns.

In the scatter phase, CLAIRE distributes the query points among processors using the sorted query points to determine the appropriate processor for each point. `thrust::copy_if` is used to copy only the relevant query points to the appropriate processors, with `thrust::make_zip_iterator` handling the (x, y, z) coordinates together. MPI's `MPI_Alltoall` and `MPI_Isend/Irecv` functions are used for sending and receiving query points among processors, ensuring each processor has the necessary data to perform local interpolations.

During the interpolation phase, CLAIRE performs local interpolation

using GPU-accelerated functions, utilizing ghost cells in arrays to manage boundary conditions effectively. Ghost cells contain data from neighboring subdomains, allowing for accurate computation near the boundaries of each subdomain and reducing communication overhead by storing necessary boundary information locally.

After local interpolation, the interpolated results are collected and redistributed to the corresponding processors using MPI, ensuring each processor receives the correct interpolated values for its query points. `thrust::copy_if` and other Thrust functions manage the data efficiently on the GPU. This process efficiently performs parallel interpolation by combining CUDA for GPU acceleration and MPI for inter-process communication.

By leveraging slab decomposition, we ensure efficient data distribution, effective boundary handling, and optimized interpolation across multiple GPUs. This approach significantly minimizes communication overhead and maximizes computational efficiency, making it highly suitable for large-scale grid-based computations.

Challenges: Despite leveraging CUDA-aware MPI for efficient data handling between GPUs and host devices, and utilizing the Thrust library to effectively allocate query points across GPUs—thus eliminating the need for host-side computation—significant challenges remain. These challenges include communication overheads and complex memory access patterns. The non-uniform spatial distribution of query points intensifies these difficulties, leading to imbalances across different MPI ranks. This complicates the determination of the optimal distribution of query points for local processing or for dispatch to other MPI ranks via `scatter_mpi_buffer`, which can result in costly scattered memory accesses. Furthermore, the inherent data dependencies within the semi-Lagrangian scheme prevent the overlapping of communication and computation, adding another layer of complexity to the process.

- **FFT method:** FFT is a highly efficient algorithm for computing the Discrete Fourier Transform (DFT) and its inverse. This transformation is crucial for converting signals from their original domains—typically time or space—into the frequency domain [83, 89]. In the CLAIRE framework,

FFT is instrumental in two key operations. Firstly, it facilitates the application of the regularization operator in the spectral domain, and its inverse, through two FFT operations in spectral methods. Secondly, FFT is utilized in the implementation of the preconditioner, which also involves two FFT operations. By leveraging the simplicity of operations in the frequency domain, FFT not only accelerates computational processes but also effectively manages the complexity inherent in advanced modeling and algorithmic challenges. These capabilities are vital for achieving high-precision registration in practical applications such as medical imaging. The efficiency of FFT is crucial for tasks that require rapid and accurate image alignment, making it an indispensable tool in the spectral methods and preconditioning processes utilized in CLAIRE.

Multi-GPU FFT kernel distribution: In CLAIRE, a multi-GPU 3D FFT implementation leverages cuFFT, enhanced by a 2D slab decomposition strategy for performance optimization. Spatial decomposition is managed along the outermost dimension (x_1), while the spectral domain is aligned with the x_2 dimension, ensuring memory continuity along x_3. This setup reduces misaligned memory accesses during communications and transpositions. The FFT process entails three sequential steps for both forward (real-to-complex) and inverse (complex-to-real) transformations, executed as batched 2D and 1D FFTs on the x_2-x_3 planes and x_1 dimension, respectively. Inter-GPU communication is streamlined through CUDA-aware MPI, and for data volumes over approximately 500 KB, it switches to GPU-optimized peer-to-peer routines. This adjustment is particularly beneficial in systems with up to four GPUs interconnected by NVLink, significantly boosting efficiency.

Challenges: FFT operations, particularly vital for tasks like image registration, involve complex mathematical processes and significant data communication. The Cooley-Tukey algorithm, a standard FFT approach, uses a 'butterfly' data exchange pattern. This approach divides the data set into smaller segments, processes FFTs independently on each segment, and then merges the results, requiring extensive inter-process communication in parallel computing settings.

On a single GPU, the challenge of communication is mitigated as all

data resides locally, enabling efficient computation handling by cuFFT without external data transfers. Numerous advancements have been made to accelerate FFT computations, including leveraging Tensor Cores in NVIDIA GPUs to speed up the FFT computation, but these enhancements often focus on 1D or 2D FFTs [3, 51, 68]. Furthermore, these improvements are primarily designed for single-GPU setups and do not fully tackle the complexities found in multi-GPU environments.

In multi-GPU systems, managing data distribution and communication overhead introduces significant challenges, especially for 3D FFTs. The requirement for all-to-all collective operations—such as data gathering, scattering, and synchronization—amplifies communication demands. Efficiently implementing FFTs in such distributed settings necessitates strategic planning and optimization to mitigate excessive communication overhead, which could negate the benefits of parallel processing. The computational complexity of FFTs, coupled with the extensive data communication they require, presents notable challenges. Load imbalances across ranks limit scalability. Furthermore, due to inherent data dependencies, communication can only be overlapped with the local transpose operation. Moreover, the frequent synchronization and data exchanges necessary for FFT computations mean that communications between GPUs on different nodes introduce considerable delays. These delays are primarily due to the lower bandwidth and higher latency of network interfaces compared to direct GPU interconnects like NVLink. These factors can significantly impact the performance and efficiency of FFT processes in distributed GPU environments.

The profiling results highlight the impact of communication patterns on FFT computation performance. Figure 4.10 presents a profiling snapshot taken during the execution of the FFT kernel within the CLAIRE application, using two GPUs.

In the timeline, multiple memory copy operations (`Memcpy DtoD`, `Memcpy PtoP`) indicate data transfers within the same GPU (Device to Device) and between GPUs (Peer to Peer). The "FFT Gaussian filter" tasks identified in the timeline represent the actual FFT computations, detailed using NVTX ranges to enhance clarity. Entries labeled `MPI_Waitall` highlight points

where the process is on hold, awaiting the completion of all non-blocking MPI operations.

The timeline reveals a marked contrast in the activity levels between GPU_0 and GPU_1. GPU_0 demonstrates a higher level of continuous activity, suggesting efficient task engagement with minimal idle periods. Conversely, GPU_1 exhibits longer idle times, which may indicate delays due to data transfers or synchronization needs. This disparity in activity levels between the two GPUs underscores a significant load imbalance, with GPU_0 processing consistently and efficiently, while GPU_1 experiences underutilization. This imbalance directly impacts the overall computational efficiency, pointing to potential areas for optimization in resource allocation and task distribution.

Figure 4.7 CLAIRE registration results for 4DCT Case 6: The left image shows an overlay of the reference image (green) and the template image (magenta) before registration. The right image displays the overlay of the reference and template images after registration, demonstrating the alignment achieved. Viewpoint: axial.

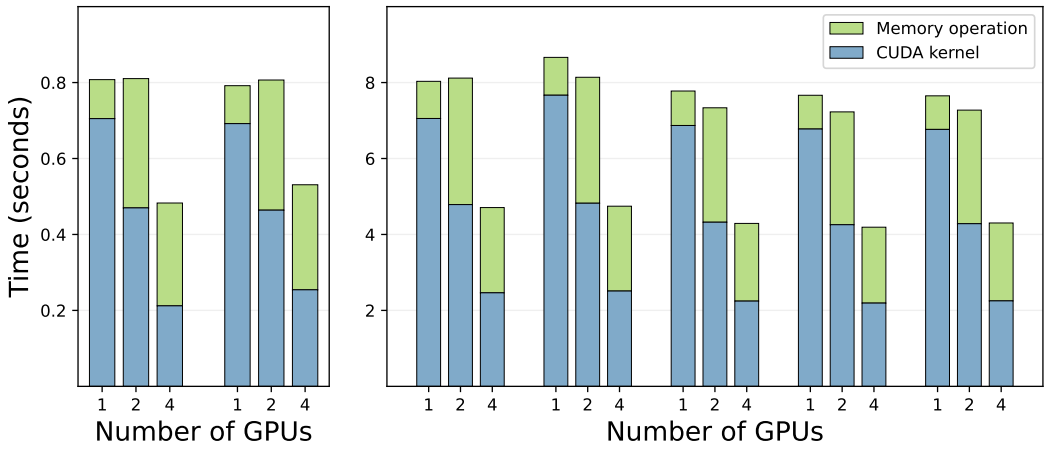


Figure 4.8 CLAIRE strong scaling profiling results using 1, 2, and 4 GPUs: A breakdown of time spent on CUDA kernel execution and memory operations. The chart on the left shows the results for the small dataset (Cases 2 and 3), and the chart on the right shows the results for the large dataset (Cases 6 to 10). Note that the time differs from the time reported in Table 4.5, which was the total runtime of the entire solver. These results highlight data transfer time as a potential bottleneck.

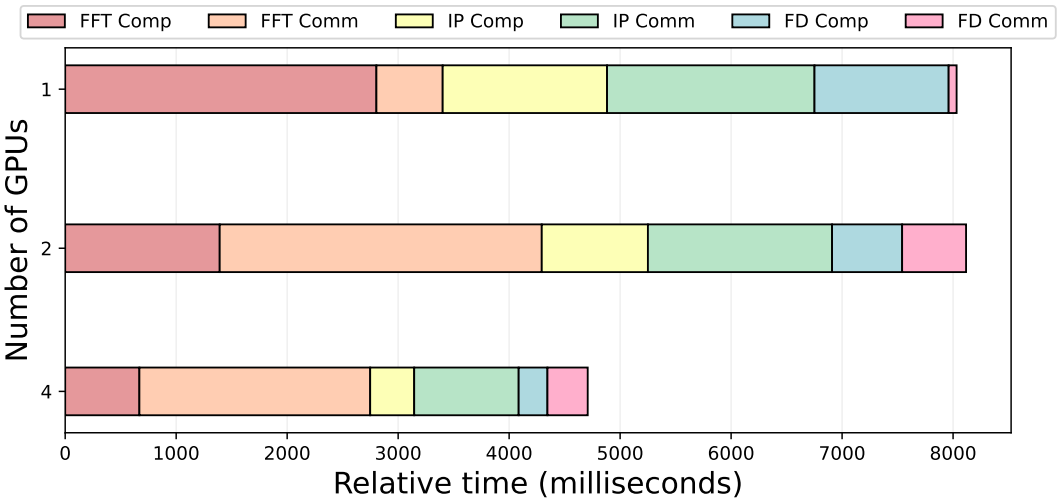


Figure 4.9 Strong scaling breakdown for Case 6: This figure shows the relative distribution of runtime across individual kernels and communication processes in a multi-GPU framework. It highlights that the runtime is predominantly occupied by communications, with FFT communication being the most significant factor.

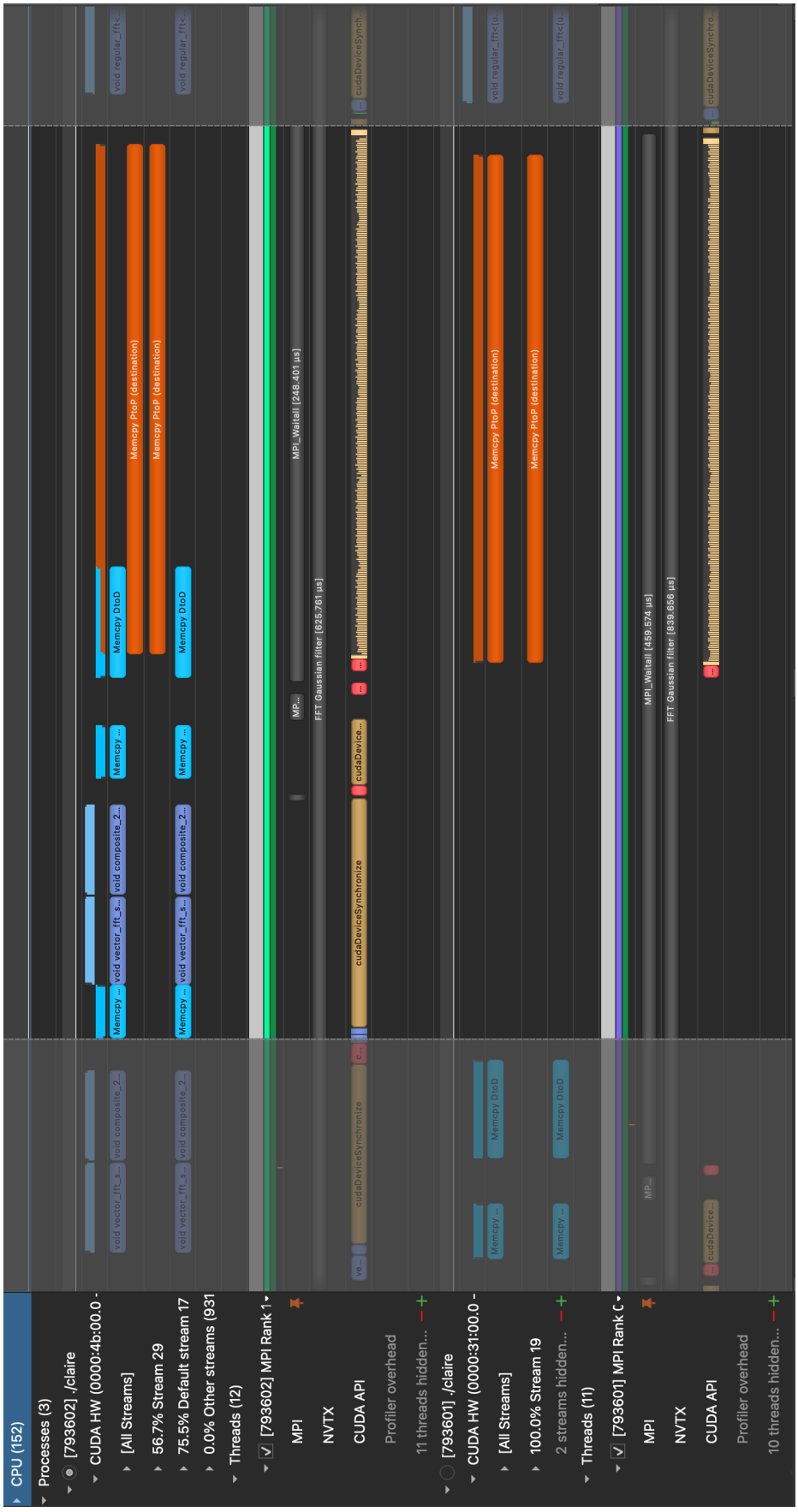


Figure 4.10 Profiling snapshot from NVIDIA Nsight Systems showing CLAIRE using two GPUs with MPI CUDA-aware: This figure illustrates the load imbalance between GPU_0 and GPU_1 during FFT computation, highlighting differences in memory transfer and processing times.

In [14], it is shown that the limitations of the CLAIRE multi-GPU framework are notably mitigated when handling large-scale problems. The implementation is particularly effective for very large datasets, achieving a grid size of 1024^3 across 32 GPUs on 8 nodes for the NIREP dataset, and an even more substantial 2024^3 grid across 256 GPUs on 64 nodes for synthetic data. These grid dimensions significantly surpass the capabilities of single GPU setups, highlighting the framework's enhanced ability to process large datasets efficiently.

However, while the framework scales effectively for large synthetic datasets, scalability issues persist for real-world datasets, which typically involve smaller problem sizes. In these scenarios, the communication overhead necessary for coordinating between GPUs often outweighs the computational benefits derived from distributing the workload.

4.5.3 Discussion on Scalability Findings

In this work, we addressed the challenges of scaling the CLAIRE framework across multiple GPUs. A critical issue is determining the optimal number of GPUs for different problem sizes to ensure efficient scalability. This challenge arises from the significant trade-offs between computation and communication overhead, which vary with both the problem size and the number of GPUs.

Our findings show that while using four GPUs is optimal for a lung dataset with dimensions $512 \times 512 \times 128$, a larger brain dataset with dimensions $512 \times 512 \times 512$ requires up to 64 GPUs. This highlights the scalability issues and underscores the importance of precise GPU configuration concerning the specific problem size.

The scalability of CLAIRE's multi-node, multi-GPU implementation notably suffers from high communication costs associated with clinically relevant problem sizes. In particular, the runtime of FFTs is significantly affected by communication overhead due to the required all-to-all collective operations. To address these issues and move towards achieving *real-time* capabilities, it is crucial to significantly reduce data transfer times. In the next chapter, we will outline our strategy to enhance scalability. This strategy involves tackling GPU interaction challenges by leveraging data parallelism within the image registration domain, with the goal of optimizing both efficiency and performance.

CHAPTER 5

Rapid Overlapped Partitioning-based Deformable Image Registration

5.1 Introduction

In the previous chapter, we examined the scalability challenges associated with the CLAIRE framework, particularly in the context of clinically relevant image sizes that result in considerable communication overhead.

In the standard CLAIRE framework, computational kernels are distributed across multiple GPUs but operate within a single, unified registration process. This method requires consistent data sharing and synchronization across GPUs, which ensures coherence but introduces substantial communication overhead. The data dependencies across GPU boundaries complicate the system's scalability, especially in multi-node environments, as discussed in the previous chapter.

This chapter introduces rapid overlapped partitioning-based DIR (CLAIRE-ROP), a solution designed to address scalability issues within the CLAIRE framework. Unlike the traditional multi-GPU model that distributes computational tasks across nodes, CLAIRE-ROP employs a data parallelism approach. This method involves strategically partitioning the image data and distributing these partitions across multiple GPUs. Each GPU independently registers its assigned image partition, thereby reducing the communication overhead and complexities associated with the original CLAIRE framework, where computational kernels were shared across a multi-node, multi-GPU environment.

This novel approach facilitates data management, markedly improving both scalability and efficiency in the image registration process. Additionally, our methodology has been extended to conserve GPU resources, resulting in the development of CLAIRE-ROP-EX (Efficiency Extended).

5.2 Methodology

Image registration is an inherently multi-steps process, typically encompassing three main stages: pre-processing, processing, and post-processing. Pre-processing is the initial step where input images are prepared for registration. This includes tasks such as noise reduction, image normalization, segmentation, and cropping to ensure images are optimally conditioned for accurate registration. Processing involves the actual registration of the images, where the spatial relationships between the images are estimated. This step includes computing deformation fields to align the template images with the reference images. Post-processing is the final step, focusing on refining the registered images and correcting any errors or artifacts that may have occurred during the previous stages. This often involves further segmentation and thorough quality assessment to ensure the integrity of the registration results.

Reflecting this structured approach, our methodology comprises:

- **Pre-processing:** We introduce a standardized partitioning technique designed to accommodate anatomical variations in lung images, which can vary widely in size and shape.
- **Processing:** Our partitioning technique facilitates parallel processing of individual registrations, significantly enhancing efficiency. In this crucial step of DIR, deformation fields are computed to align each partitioned template image with the corresponding partitions of the reference image.
- **Post-processing:** After completing individual registrations, the deformed partitions are carefully merged to create a unified and comprehensive deformed image.

These steps collectively enhance the effectiveness of our deformable image registration framework, ensuring that it not only maintains high accuracy but also exhibits improved scalability and performance.

5.2.1 Related Work

Distributed multi-GPU applications leverage data partitioning in HPC environments to efficiently distribute workloads across multiple processors or GPUs. This strategy is crucial for computationally intensive tasks, such as large-scale image analysis, 3D rendering, and deep learning. By processing each partition independently or in parallel, systems can greatly enhance efficiency and enable more detailed analysis at the local level. This parallel processing approach drastically reduces the time required for tasks like image analysis [14, 38, 9].

In CLAIRE, as discussed in previous chapters, both interpolation and FFT kernels leverage data partitioning strategies such as slab decomposition. These operations make use of the Thrust and cuFFT libraries to distribute computations efficiently across GPUs.

For example, [38] presents a framework that aligns with standard GPU programming paradigms while being designed to scale for large-scale problems. This framework allows GPU kernels to handle varying data volumes across numerous GPUs and nodes, introducing mechanisms for distributed kernel launches and managing distributed arrays via partitioning methods. These techniques ensure a transparent distribution of tasks and data across multiple GPUs, leading to a balanced workload distribution. The system was evaluated using eight different CUDA kernels to represent a range of computational workloads.

Furthermore, [9] discusses the design of a multi-GPU encryption framework that adopts a data-parallelism approach using partitioning methods to distribute the workload evenly across available GPUs. This method involves partitioning polynomials stored in matrices, utilizing matrix partitioning strategies like column or row partitioning based on access patterns.

These studies demonstrate how strategic data partitioning can substantially enhance the scalability and efficiency of complex computational tasks across multiple GPUs in distributed environments. However, as noted in previous chapters, achieving scalability in such systems poses significant challenges.

In the specific context of CLAIRE, as detailed in earlier discussions, the framework employs advanced data partitioning to distribute computational loads. Despite these efforts, scalability issues persist. Similar to other distributed GPU systems, the framework encounters challenges related to communication overhead and memory contention as the number of GPUs increases. For example, the

study in [9], despite its innovative partitioning strategies, encounters scalability issues when extending its partitioning strategies to a larger number of GPUs. This is due to the increased communication and memory contention, which limit its performance beyond eight GPUs.

This underscores a common challenge in distributed GPU systems, where the benefits of parallel processing can be offset by the overhead introduced by inter-GPU communication and data management. These insights emphasize the need for ongoing development in partitioning techniques and communication optimization to fully exploit the potential of multi-GPU environments for large-scale computational tasks.

Image partitioning is a fundamental technique in image processing and computer vision, where an image is divided into multiple segments or regions, often referred to as partitions, tiles, patches, or chunks. This method is widely applied across various applications to facilitate specific analysis tasks, boost performance, or address computational constraints. The primary goal of image partitioning is to simplify or enhance the analysis of complex images by breaking them into smaller, more manageable sections. For example, [88] discusses an adaptive non-uniform rectangular partitioning algorithm that enhances JPEG image compression at low bit rates. This algorithm dynamically adjusts partition sizes based on texture complexity, improving image quality and reducing artifacts. Similarly, [1] explores sophisticated partitioning structures for video coding, such as extended quad-tree and asymmetric ternary-tree partitioning, demonstrating the versatility and effectiveness of these strategies in refining media compression standards.

In the context of medical imaging, segmentation is a widely used technique for analyzing specific ROIs within larger scans, contrasting with partitioning, which primarily enhances computational efficiency for large datasets. Segmentation involves isolating precise anatomical structures, such as the heart, lungs, and bones in chest CT scans, allowing for the independent assessment of each area. This is critical for diagnostics, as it provides healthcare professionals with detailed visualizations of individual anatomical regions. Segmentation facilitates the diagnosis, monitoring, and treatment of various conditions by enabling accurate localization and assessment of diseases, as evidenced by the cited references [56, 18, 72, 67].

Unlike partitioning, which is generally used to enhance computational effi-

ciency for large datasets, segmentation has a direct impact on clinical outcomes by precisely identifying critical anatomical and pathological features within medical images. For instance, it is used to differentiate healthy tissue from tumors [56, 18], diagnose respiratory diseases [67], or identify lesions associated with conditions like COVID-19 in lung CT scans [72]. Segmented images provide vital information for accurate diagnosis, treatment planning, and disease monitoring. However, due to the complexity of the process and the detailed analysis required, segmentation is often time-consuming and typically performed offline.

In our research, we employ image partitioning techniques within the medical imaging domain to improve the parallel processing capabilities of the DIR framework. By distributing image partitions across multiple GPUs, we aim to reduce the communication overhead that was a significant bottleneck in the original CLAIRE framework, which relied on distributed kernel launches for computational tasks across GPUs.

5.2.2 Pre-processing Step

In Section 5.2.2.1, we describe our partitioning technique and highlight the limitations of dividing lung images into multiple partitions due to their distinctive anatomical structure. Section 5.2.2.2 presents a potential solution to prevent interdependencies among data splits: the use of halo-based partitioning, which employs overlapping regions in the partitioning process. Finally, Section 5.2.2.3 outlines the potential for resource conservation without compromising performance.

5.2.2.1 Image Partitioning

In this section, we begin our methodology by examining the diversity within the Dir-Lab dataset. We encounter two distinct pixel dimensions: 256×256 for Case 1 to Case 5 and 512×512 for Case 6 to Case 10, representing the X and Y dimensions. The Z dimension, indicating the number of slices, varies based on the patient, specific imaging protocol, or clinical indications. This variability is typical in CT imaging and reflects the depth of the imaged volume and its clinical requirements. Therefore, we employ a consistent 2D axial viewpoint for image partitioning to ensure uniform handling of the X and Y dimensions across all images.

To achieve an optimal workload balance among GPUs, careful consideration of the configuration layout is essential. Our empirical analysis concludes that a mixed approach, combining both column-wise and row-wise partitioning, is the most effective method for balancing the workload across the GPUs.

The optimal layout begins with partitioning the image data into major columns reflecting significant anatomical divisions, such as the separation of the left and right lungs. Each column is then subdivided into rows that represent different upper or lower sections, drawing inspiration from the anatomical layout of the lung's lobes. Finally, column-wise partitioning within each of these four broad partitions results in a total of eight smaller partitions.

To identify the optimal number of partitions for each dataset, we establish a threshold designed to maintain accuracy levels that are comparable to those of the baseline methods, namely ANTs and CLAIRE. Specifically, we aim to determine the maximum number of partitions that can achieve the fastest registration time while maintaining registration accuracy across various data sizes.

By adhering to an optimal partition layout, which forms a fundamental aspect of our decision-making process, we ensure the most rapid registration possible without compromising accuracy. This layout considers critical factors such as the number of available GPUs, the size of the problem, and the time sensitivity of the application. Based on these considerations, CLAIRE-ROP is configured to employ one of three distinct partitioning layouts: 2, 4, or 8 partitions. The results will be provided in Section 5.3.2.

This flexible partitioning framework allows CLAIRE-ROP to dynamically adapt to varying computational requirements, ensuring optimal performance across a broad range of medical images.

5.2.2.2 Boundary Effects Handling

Handling image partitions separately may introduce challenges in maintaining data coherency and consistency, particularly at the boundaries where data might be missing. Thus, careful attention must be given to how partitions are defined and how boundary conditions are managed. The clinical impact of these missing areas varies significantly by application. In contexts such as RT planning or detailed disease assessment, precise alignment of the lung area is crucial for effective treatment planning. Even minor omissions at the boundaries can lead to inadequate tumor alignment, significantly impacting treatment outcomes.

To mitigate these effects, we have developed a partitioning technique that ensures comprehensive data coverage, especially at the boundaries of partitions. This method involves creating overlapping partitions to ensure that no critical features are missed during the registration process. Overlaps are achieved by sharing regions between adjacent partitions, creating a seamless transition across boundaries.

To determine the optimal overlap size, we conducted experiments by incrementally adding pixels to the edges of partitions. We tested additions of 2, 4, 8, and 16 pixels to identify the most effective range for our computations. These additional pixels, also known as halo cells or ghost cells, are essential for guaranteeing that computations on the edges of partitions have access to necessary neighboring data, thereby enhancing the accuracy and efficiency of our processes. Our findings indicate that an addition of 8 pixels to the partition edges ensures the inclusion of all critical lung regions while only necessitating a slight increase in registration time. Detailed experimental results are provided in Section 5.3.1.

By employing this overlapping approach, we create well-controlled partitions that maintain comprehensive coverage of lung tissue throughout the registration process, ensuring both accuracy and efficacy in clinical applications. Figure 5.1 presents an illustrative example of our partitioning scheme, featuring an 8-pixel overlap. While overlaps are applied to all edges, it is important to note that the presence of lung tissue within vertical overlaps can vary across different datasets. For instance, in Case 6 depicted here, there is no lung tissue within the vertical borders - a characteristic consistent across all slices in this case. Conversely, observations from Case 8 underscore the critical importance of vertical overlaps. To ensure a standardized approach that is applicable to each image set, we systematically include overlaps for both horizontal and vertical borders. This strategy not only enhances the robustness of our registration process but also accommodates variations in lung tissue visibility, ensuring comprehensive coverage across all parts of the image.

5.2.2.3 Conserving Computational Resources

As detailed in Section 5.2.2.1, column-wise partitioning is employed in the final phase of the partitioning process to subdivide the images into a total of eight partitions. This layout was selected on the grounds that it offers the potential to

enhance efficiency by potentially excluding the registration of edge partitions. The rationale behind this approach is that, as the number of partitions increases, edge partitions are more likely to contain minimal lung tissue, which could result in unnecessary registration processes.

This section introduces CLAIRE-ROP-EX, an extension of CLAIRE-ROP, designed to enhance efficiency by incorporating metrics such as mask creation, mask partitioning, and lung area calculation. To determine whether edge partitions can be omitted from the registration process, we first create the template mask and then partition it in the same manner as the images.

We then analyze each mask to count the number of binary "1"s, which indicate the presence of lung tissue. By establishing a threshold for these binary values, we assess the amount of lung area in each partition. Depending on the desired accuracy—determined by the sensitivity of the application—users can decide whether to skip registration for specific partitions. This method involves a trade-off between conserving computational resources and maintaining accuracy.

In our experiments, we skip registrations only for edge partitions that do not contain any lung area to ensure the highest accuracy. "Edge partitions" refer specifically to those located at the right and left sides of the image. By first examining the content of these edge partitions and determining if they lack significant lung tissue, we register only the inner partitions. This selective registration approach helps conserve computational resources, which is particularly advantageous for large datasets.

5.2.3 Processing and Post-Processing Steps: Registrations and Integration of Deformed Partitions

Building on our partitioning approach, we perform separate registrations for each partition, carefully adjusting to accommodate overlaps. These registrations are conducted independently, allowing parallel execution and fully leveraging the processing power of multiple GPUs. This method effectively eliminates the GPU interactions that were previously identified as communication bottlenecks. Consequently, direct GPU-to-GPU communication is unnecessary, and the use of an NVLink GPU interconnect is not required for optimal performance.

Upon completing the individual registrations, we integrate the deformed partial images to construct the final, comprehensive deformed image. This

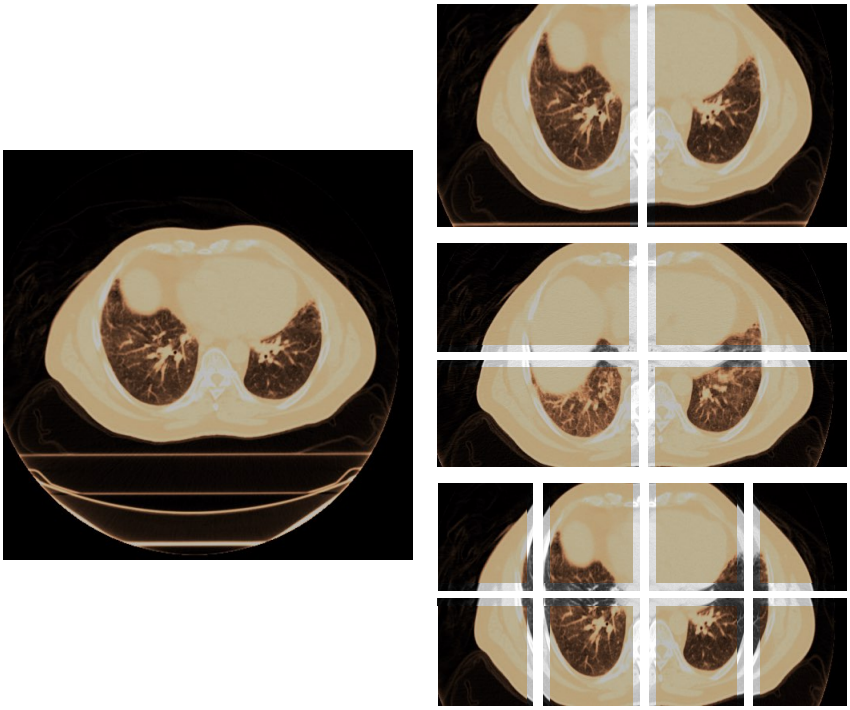


Figure 5.1 Example partitioning scheme demonstrating column-wise, row-wise, and column-wise partitioning to divide the image into 2, 4, and 8 partitions, respectively. Overlapping is used to ensure that information from neighboring regions is taken into account during the registration process to avoid loss of lung tissue (Case 6 - Slice 78).

integration process is carefully managed to ensure that all partitions are seamlessly merged, maintaining the coherence and integrity of the overall image.

In our approach, we employed a partitioning technique tailored to the number of available GPUs, with each partitioned image set assigned to a separate GPU. To minimize communication overhead, we also explored an alternative partitioning strategy where images are divided based on the number of nodes, rather than assigning each partition to an individual GPU. This strategy could streamline data management and reduce the communication burden between GPUs by grouping partitions across nodes. For instance, partitioning images into right and left sections (as in the first layout) and executing separate DIR processes for each partition across four GPUs within a single node could effectively eliminate off-node communication. This approach addresses a major source of overhead identified in the CLAIRE framework. However, as discussed in the previous chapter, on-node communication remains a significant challenge, particularly for operations like FFTs. Despite the potential benefits of reduced on-node communication, the ongoing overhead from off-node operations led us to adopt a partitioning layout configuration based on the number of available GPUs. This approach aligns more closely with our objective to optimize performance and advance real-time DIR.

The CLAIRE-ROP pipeline is outlined in Algorithm 1, and Figure 5.2 illustrates the pipeline for a scenario with 8 GPU resources. In this setup, our objective is to achieve the fastest possible registration time. To accomplish this, we utilize all available GPU resources by partitioning the images into eight distinct partitions.

To the best of our knowledge, this work represents the first instance of employing a partitioning approach to accurately address large misalignment in medical images. This innovative method significantly enhances processing speed while maintaining the accuracy required for clinical applications.

5.3 Experimental Results

This section presents the outcomes of our experimental evaluations of the CLAIRE-ROP framework. The results are divided into two key subsections aimed at validating the efficiency and effectiveness of our approach.

Algorithm 1: CLAIRE-ROP

Input : m_r, m_t, (n, k, z), o, N- reference image, moving image, original image set size, overlap, number of available GPUs

Output: m_d- deformed moving image

$original\_image\_size \leftarrow (n, k, z)$
$image\_set \leftarrow [m_r, m_t]$
$dataset \leftarrow (n, k) \leq (256, 256)$ **?** $small$ **:** $large$
/* Pre-processing */
for $image$ **in** $image\_set$ **do**
  $partition\_size \leftarrow (\frac{n}{2} + o, k, z)$
  $partitioned\_images\_2p\ [image] \leftarrow PartitionImage.execute\ (image, partition\_size)$
  if $2 \leq N < 4$ **then**
    $partitioned\_images\ [image] \leftarrow partitioned\_images\_2p\ [image]$
    break // Break from pre-processing with 2 partitions for images in image set
  end
  $partitioned\_image\_size \leftarrow (n_1, k_1, z_1)$
  for $image$ **in** $partitioned\_images\_2p$ **do**
    $partition\_size \leftarrow (n_1, \frac{k_1}{2} + o, z_1)$ // Update partition size
    $partitioned\_images\_4p\ [image] \leftarrow PartitionImage.execute\ (image, partition\_size)$
    if $4 \leq N < 8$ **or** $dataset = small$ **then**
      $partitioned\_images\ [image] \leftarrow partitioned\_images\_4p\ [image]$
      break // Break from pre-processing with 4 partitions for images in image set
    end
    $partitioned\_image\_size \leftarrow (n_2, k_2, z_2)$
    for $image$ **in** $partitioned\_images\_4p$ **do**
      $partition\_size\_e \leftarrow [(\frac{n_2}{2} + o, k_2, z_2)]$ // Edge partitions
      $partition\_size\_i \leftarrow [(\frac{n_2}{2} + 2o, k_2, z_2)]$ // Inner partitions
      $partitioned\_images\_8p\ [image] \leftarrow PartitionImage.execute\ (image, partition\_size\_i, partition\_size\_e)$
    end
    partitioned_images $[image] \leftarrow partitioned\_images\_8p\ [image]$
  end
end
/* Processing */
for $i \leftarrow 0$ **to** $N - 1$ **do**
  $Allocate\ (partitioned\_images\ [m_r](i)\ and\ partitioned\_images\ [m_t](i))\ in\ GPU\_i$
  /* Execute CLAIRE in parallel on GPU i */
  $registered \leftarrow CLAIRE.execute\ (partitioned\_images\ [m_r](i), partitioned\_images\ [m_t](i))$
  $deformed\_images.append\ (registered)$
end
/* Post-processing */
for $deformed\_image$ **in** $deformed\_images$ **do**
  $crop \leftarrow CropImage.execute\ (deformed\_image)$
  $deformed\_cropped\_images.append\ (crop)$
end
$m_d \leftarrow TileImage.execute\ (deformed\_cropped\_images)$
return m_d

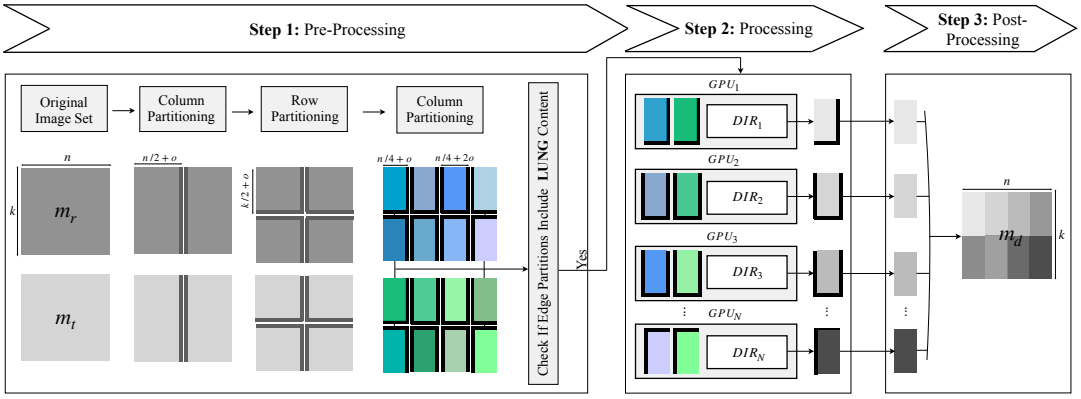


Figure 5.2 An example of the CLAIRE-ROP workflow. 'N', 'm_r', 'm_t', and 'm_d' denote the number of available GPUs, the reference image, the template image, and the deformed image, respectively. 'n' corresponds to the X-dimension size, 'k' to the Y-dimension size, and 'o' indicates the overlap. In this example, we assumed that the edge partitions contain lung contents, so we utilized all 8 GPUs. In Step 1, the partitioning method is applied. Step 2 involves performing parallel registrations, and Step 3 leads to obtaining the complete deformed template image.

5.3.1 Assessing Optimal Overlap Pixels

In this section, we investigate various configurations for overlap pixels to identify the optimal amount required to achieve the best registration performance. Table 5.1 presents the performance of CLAIRE-ROP across different border overlap settings.

Accuracy: The findings indicate that the size of the overlap has no impact on the Dice scores. These scores remain consistently high across all overlap configurations (2, 4, 8, and 16 pixels), suggesting that the quality of image alignment is maintained regardless of overlap size. All scores exceed 0.98, indicating nearly perfect registration between the template and reference images. This consistency highlights the effectiveness of the overlapping technique in ensuring comprehensive image registration without losing critical details.

Time: Although registration times vary slightly with different overlap pixel counts, no clear trend of increase or decrease is observed. This variability may be influenced by factors such as computational load at processing time or specific characteristics of each dataset.

The observation that larger borders can lead to faster results, despite an increase in image size, may seem counterintuitive. However, this can occur if

the additional border pixels reduce the number of iterations needed for convergence. Tight borders may create edge artifacts or require more iterations for the algorithm to achieve the correct alignment. By providing a larger overlap, the algorithm gains more contextual information, which can facilitate quicker convergence, even if each iteration takes slightly longer due to the increased image size.

Notably, in cases such as Case 6 and Case 7, where larger variations in registration time are observed, the registration times do not necessarily increase with a higher number of overlap pixels. For instance, in Case 6, the registration time with 4 pixels of overlap is significantly higher than with 16 pixels. This suggests that factors like initialization, algorithmic efficiency, and computational resource utilization at different stages may also influence the registration times.

The decision to use an 8-pixel overlap strikes a balance between ensuring high-quality registration and managing computational efficiency. While the registration times for an 8-pixel overlap are generally not the fastest, they remain competitive and are close to the lowest times observed across the datasets.

Figure 5.3 illustrates an integrated deformed image for Case 6, both with and without adding extra pixels to the borders. The seamless integration of the deformed images, particularly when overlaps are applied, confirms the theoretical and empirical findings presented in Table 5.1. Adding overlap ensures no critical information is lost, thereby supporting the integrity of the clinical assessments or treatment planning derived from these images. This demonstrates the practical benefits of using an optimal overlap in the registration process.

In summary, based on our dataset, an 8-pixel overlap provides a reasonable compromise between speed and accuracy. However, this choice is empirical, and the optimal value may vary with different datasets or algorithms. While other overlap values might perform marginally better in specific cases, an 8-pixel overlap represents a practical and balanced option for general applications across various scenarios.

5.3.2 Assessing Performance

Following the assessment of optimal overlap settings, we conduct a comparative analysis of CLAIRE-ROP's performance against the traditional CLAIRE framework. This evaluation focuses on key metrics such as the speedup and Dice

Table 5.1 Performance of the CLAIRE-ROP across 4 partitions, illustrating registration times in seconds for various overlap pixel configurations: 2, 4, 8, and 16.

Dataset	2 pixels	4 pixels	8 pixels	16 pixels	Dice
Case 1	0.31	0.30	0.32	0.31	0.995
Case 2	0.35	0.34	0.35	0.36	0.995
Case 3	0.28	0.27	0.32	0.33	0.993
Case 4	0.38	0.37	0.37	0.40	0.991
Case 5	0.35	0.33	0.35	0.39	0.993
Case 6	0.90	1.36	0.89	0.95	0.983
Case 7	1.05	1.24	1.04	1.13	0.987
Case 8	1.25	1.02	1.19	1.07	0.992
Case 9	0.77	0.73	0.75	0.81	0.988
Case 10	0.78	0.77	0.75	0.81	0.990

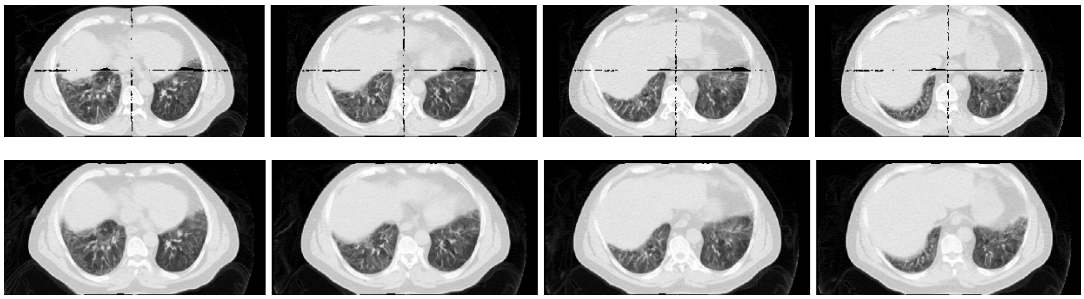


Figure 5.3 Registration visualization results for Case 6, showing four different slices, using four partitions. The top row displays the deformed image without additional border pixels, while the bottom row includes an 8-pixel border for overlap.

score to quantify improvements. Additionally, we provide visualization results to demonstrate CLAIRE-ROP's enhanced alignment capabilities, substantiating its superiority in handling complex registration tasks with increased efficiency and accuracy.

In Table 5.2, we show that partitioning the small dataset into up to 4 partitions and the large dataset into up to 8 partitions maintains accuracy while providing good scalability as resources are increased.

For the small dataset, we divided it into four partitions, each measuring

136×136 pixels. This size includes a core of 128×128 pixels with an 8-pixel overlap on each side, resulting in the total dimension of 136×136 pixels per partition. For the large dataset, we created four edge partitions of 136×136 pixels and four inner partitions of 144×136 pixels, with both dimensions accounting for an 8-pixel overlap along the common borders.

We also evaluated CLAIRE-ROP with further partitioning, dividing the image into up to 16 partitions. For the small dataset, using 16 GPUs allowed us to reduce the minimum partition size to 64×64 pixels. The results demonstrated that accuracy was maintained, and registration times for individual partitions were faster. However, the anticipated scalability improvements with 16 GPUs did not occur with the reduced partition sizes.

In Case 4, the overall registration time remained unchanged compared to the 8-GPU configuration. To investigate this, we examined the time required to complete each registration on the assigned GPUs. With 8 GPUs, the registration times for individual partitions were 189, 197, 202, 232, 238, 246, 249, and 258 milliseconds. In comparison, with 16 GPUs, the registration times for the partitions were 99, 138, 140, 142, 142, 144, 150, 157, 160, 163, 183, 193, 194, 213, 222, and 260 milliseconds. These results reveal that, despite the expectation of faster overall registration with 16 GPUs, the improvement in processing time for most partitions was offset by one partition that took significantly longer. This discrepancy led to no overall reduction in the total registration time for the entire image.

For the large dataset, using 16 GPUs allowed us to reduce the partition size to 128×64 pixels. The results showed that accuracy was maintained or improved, and faster registration times were achieved. However, the speedup gains showed diminishing returns with the increase in the number of GPUs.

Hence, when selecting the number of partitions, it is crucial to balance partition size, scalability, speedup, and the number of available resources. Excessively small partitions may result in inefficient GPU utilization, as they may not fully leverage the available computational power. Further discussion will be provided in the section 5.3.2.3.

5.3.2.1 Accessing Registration Time: Comparison with CLAIRE

To ensure a fair comparison between CLAIRE-ROP and the traditional CLAIRE framework, we selected configurations that offered a balance between registration

time and accuracy, as summarized in Table 5.3. The registration times were extracted from Tables 4.5 and 5.2, with particular attention given to scalability performance.

For the small dataset, CLAIRE achieved its best registration time using a single GPU, while CLAIRE-ROP performed optimally with 4 GPUs. For the large dataset, CLAIRE achieved its best registration time using 4 GPUs, while CLAIRE-ROP performed optimally with 8 GPUs. Although CLAIRE-ROP achieved the fastest time with 16 GPUs, we selected the 8-GPU configuration for CLAIRE-ROP due to its better scalability. With CLAIRE-ROP, the small dataset was registered in an average of 0.34 seconds across four partitions, achieving a speedup factor of 2.9. For the large dataset, registration with eight partitions was completed in an average of 0.46 seconds, resulting in a significant speedup factor of 10.6. Crucially, these performance enhancements were achieved without compromising the Dice scores, thus maintaining accuracy alongside improved efficiency.

Table 5.2 Comparative analysis of CLAIRE versus CLAIRE-ROP using 2, 4, 8, and 16 partitions. Note that running CLAIRE on 2 nodes with 8 GPUs introduces excessive communication overhead, making it non-scalable beyond 4 GPUs. Registration times are listed in seconds.

Dataset	2 GPUs				4 GPUs				8 GPUs				16 GPUs	
	CLAIRE		CLAIRE-ROP-2P		CLAIRE		CLAIRE-ROP-4P		CLAIRE		CLAIRE-ROP-8P		CLAIRE-ROP-16P	
	Time	Dice	Time	Dice	Time	Dice	Time	Dice	Time	Dice	Time	Dice	Time	Dice
C1	1.03	0.995	0.60	0.995	-	-	0.32	0.995	-	-	0.20	0.991	0.13	0.991
C2	1.06	0.994	0.64	0.995	0.80	0.994	0.35	0.995	-	-	0.24	0.991	0.21	0.991
C3	1.07	0.994	0.57	0.993	0.82	0.994	0.32	0.993	-	-	0.21	0.990	0.16	0.990
C4	-	-	0.65	0.991	-	-	0.37	0.991	-	-	0.26	0.987	0.26	0.987
C5	1.26	0.994	0.61	0.993	-	-	0.35	0.993	-	-	0.23	0.988	0.20	0.988
C6	8.84	0.981	1.38	0.983	5.01	0.981	0.89	0.983	28.65	0.981	0.56	0.983	0.43	0.983
C7	8.82	0.984	1.38	0.985	4.98	0.984	1.04	0.987	28.63	0.984	0.54	0.986	0.44	0.989
C8	7.97	0.989	1.39	0.991	4.62	0.989	1.19	0.991	26.09	0.989	0.46	0.991	0.31	0.992
C9	7.87	0.988	1.23	0.988	4.44	0.988	0.75	0.987	25.78	0.988	0.37	0.987	0.26	0.987
C10	7.94	0.990	1.27	0.990	4.56	0.990	0.75	0.990	26.43	0.990	0.36	0.989	0.29	0.989

5.3.2.2 Accessing Resource Conservation: CLAIRE-ROP-EX

In CLAIRE-ROP-EX, our objective was to optimize computational resource usage while maintaining registration accuracy, specifically by evaluating whether edge partitions could be excluded from registration. We assessed the lung content in edge partitions for our large dataset using 8 GPUs, with the results detailed in Table 5.4.

For cases such as Case 9 and Case 10, we observed that the edge partitions contained no lung tissue. Consequently, we concentrated on registering only the inner partitions. This approach allowed us to reduce GPU usage from 8 to 4 while achieving the same performance gains, thereby optimizing resource allocation.

In Case 6, the lung content in the right edge partitions was minimal, with percentages as low as 0.05% and 0.10%. Although these low percentages suggest that skipping registration for these partitions might be feasible, our method only skips registrations where the lung content is 0%, thus ensuring that accuracy is prioritized. However, depending on the application's sensitivity, alternative thresholds and decisions may be applied.

5.3.2.3 Accessing Registration Time: Strong Scaling Speedup

We evaluated the strong scalability of CLAIRE-ROP, as illustrated in Figure 5.4. This figure contrasts the actual speedup achieved with the theoretical ideal as the number of GPUs increases. The ideal speedup line represents a perfect linear increase in performance with additional GPUs, serving as a benchmark for evaluating the real-world performance of our system.

To ensure precise and reliable strong scaling calculations, we standardized the problem size by cropping large images to exclude background elements that do not contribute to the core computation tasks for CLAIRE. This optimization is seamlessly integrated into the CLAIRE-ROP's partitioning strategy.

The green dashed line shows the speedup for a small dataset. It trends upwards, indicating that speed increases with more GPUs but at a rate below the ideal. The blue dotted line represents the speedup for a large dataset. The slope is steeper compared to the small dataset, suggesting better scalability.

The graph demonstrates that CLAIRE-ROP enhances scalability compared to the traditional single-GPU CLAIRE framework, especially for larger datasets,

Table 5.3 Comparison of the registration times for each method on small and large datasets, along with the speedup factor.

Small	CLAIRE(1GPU)		CLAIRE-ROP(4GPU)		Speedup
dataset	Time	Dice	Time	Dice	factor
Case 1	0.93	0.995	0.32	0.995	2.9
Case 2	0.98	0.994	0.35	0.995	2.8
Case 3	0.99	0.993	0.32	0.993	3.0
Case 4	1.02	0.990	0.37	0.991	2.8
Case 5	1.13	0.993	0.35	0.993	3.2
Average	**1.04**	**0.993**	**0.34**	**0.993**	**2.9**
Large	CLAIRE(4GPU)		CLAIRE-ROP(8GPU)		Speedup
dataset	Time	Dice	Time	Dice	factor
Case 6	5.01	0.981	0.56	0.983	9.0
Case 7	4.98	0.984	0.54	0.986	9.2
Case 8	4.62	0.989	0.46	0.991	10.0
Case 9	4.44	0.988	0.37	0.987	12.1
Case 10	4.56	0.990	0.36	0.989	12.6
Average	**4.65**	**0.986**	**0.46**	**0.987**	**10.6**

Table 5.4 Lung content percentage for edge partitions with 8 partitions.

Dataset	RU_e	RL_e	LU_e	LL_e
Case 6	0.05%	0.10%	0.70%	1.72%
Case 7	1.63%	1.39%	1.07%	1.02%
Case 8	0.88%	0.15%	0.79%	0.21%
Case 9	0.00%	0.00%	0.00%	0.00%
Case 10	0.00%	0.00%	0.00%	0.00%

as detailed in Table 4.5. Notably, the steep incline of the blue line at higher GPU counts indicates that CLAIRE-ROP's partitioning strategy becomes increasingly efficient with the growing dataset size.

However, with 16 GPUs, scalability begins to level off, achieving a speedup of approximately 7.85 for the large dataset, which falls short of the ideal speedup of 16. This suggests that smaller partitions might not always deliver the expected speedups, especially if the computational tasks are not evenly distributed or if there is limited contextual information available for accurate processing. Smaller partitions, required for distributing workloads across more GPUs, could lead to inefficient GPU utilization, especially if the computational tasks do not scale linearly with more partitions. Smaller partitions, necessary for distributing workloads across more GPUs, may lead to inefficient GPU utilization if the computational tasks do not scale linearly with the number of partitions. Additionally, increasing the number of GPUs demands more resources, which may not be feasible in all clinical settings.

To gain further insight, Table 5.5 reports the minimum and maximum execution times for configurations with 2, 4, 8, and 16 GPUs for large dataset. These results highlight performance variations across different partitions when using CLAIRE-ROP. As observed, there are considerable differences in registration times achieved with different partitions, particularly in Case 6 and Case 7. In these instances, a time difference of up to 260 milliseconds between registering two different partitions at the 16-GPU configuration was noted. Such discrepancies can significantly impact overall performance, as the overall process must wait for the slowest GPU execution before proceeding with the integration of results.

The results emphasize the critical need to balance partition sizes based on

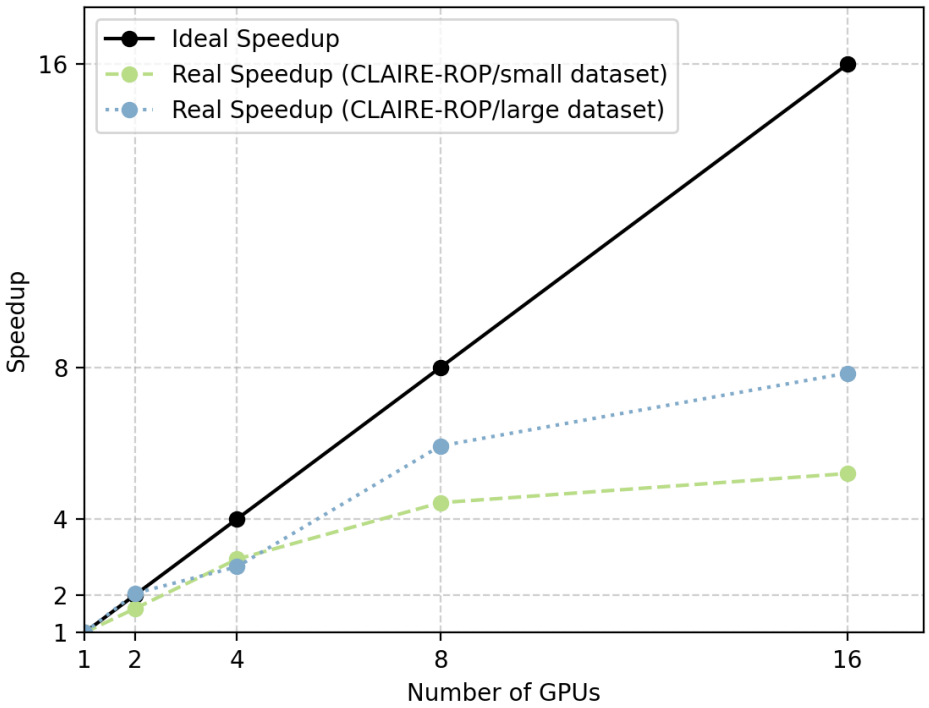


Figure 5.4 Strong scaling speedup factor in CLAIRE-ROP: The speedup factor for each dataset is calculated as the average across all five images. Our method demonstrates reasonable scalability on both datasets up to 8 GPUs.

dataset characteristics, and computational resources. Optimal partitioning is essential to ensure that no single GPU becomes a bottleneck due to a disproportionately large workload. This strategy, which will be detailed in Chapter 9, involves adjusting partition sizes dynamically based on the computational complexity of each region, helping to distribute the workload more evenly across GPUs and potentially improving overall scalability.

Although real speedups deviate from the ideal, the scalability improvements are notable given the computational intensity and data dependency of the tasks involved in image registration. These advancements can significantly reduce computation times, leading to faster clinical workflows and enabling more timely interventions where medical DIR is critical. This progress is particularly vital for real-time applications such as IGART, where rapid image processing directly impacts treatment efficacy and patient outcomes.

Table 5.5 Minimum and maximum registration times (in seconds) for registering large dataset partitions in parallel, with 2, 4, 8, and 16 GPU variations.

Dataset	2 GPUs		4 GPUs		8 GPUs		16 GPUs	
	min	max	min	max	min	max	min	max
Case 6	1.38	1.31	0.77	1.07	0.36	0.55	0.17	0.43
Case 7	1.34	1.38	0.74	1.04	0.36	0.54	0.18	0.44
Case 8	1.32	1.39	0.69	1.19	0.30	0.46	0.17	0.31
Case 9	1.23	1.45	0.64	0.75	0.31	0.37	0.18	0.26
Case 10	1.21	1.27	0.67	0.75	0.32	0.36	0.17	0.29

5.3.2.4 Accessing Registration Accuracy: Visualization Results

We provide visualization results that confirm the framework's capability to handle complex registration tasks efficiently, highlighting its ability to achieve robust registration with notably enhanced speed. Figure 5.5 displays the visualization results using white binary masks of the lung region, computed using TotalSegmentator, for Case 6 across six distinct slices (65, 70, 75, 80, 85, 90). The top row shows the masks for the reference image, the middle row for the template image, and the bottom row for the deformed image. The binary mask results indicate a slight decrease in accuracy at the borders of the lungs in slices from 75 to 85. It is important to note that the accuracy of these measurements heavily depends on the segmentation tool used.

Visual evaluation using Figure 5.6, which does not rely on segmentation, shows a more accurate alignment. This figure employs four partitions for registration and demonstrates overlays of the reference image with the template image both before and after registration. This is particularly evident in slice 78, where the template image is significantly deformed, exhibiting a notable initial discrepancy. This slice also falls within the range identified as less accurate when using binary masks for evaluation. This visualization demonstrates perfect alignment after registration, contrasting the results showcased with binary masks. Segmentation can introduce artifacts or inaccuracies that skew the perceived alignment when assessed with binary masks. Direct overlays, by contrast, allow for a straightforward, unaltered comparison, potentially providing a more accurate visual representation of how

well the images align.

As we reported in Table 5.2, the Dice score for the small dataset decreased when using 8 partitions. For example, in Case 5, the Dice score dropped from 0.995 to 0.991. To provide a visual comparison, Figure 5.7 displays the differences in the deformed images for Case 5. The figure confirms that the differences between using 2 partitions and 8 partitions are minimal and mainly visible at the edges of the lung regions, as shown by the overlaid green and magenta sections. The critical lung edges show slight variations, indicating that while there are differences, they are not substantial.

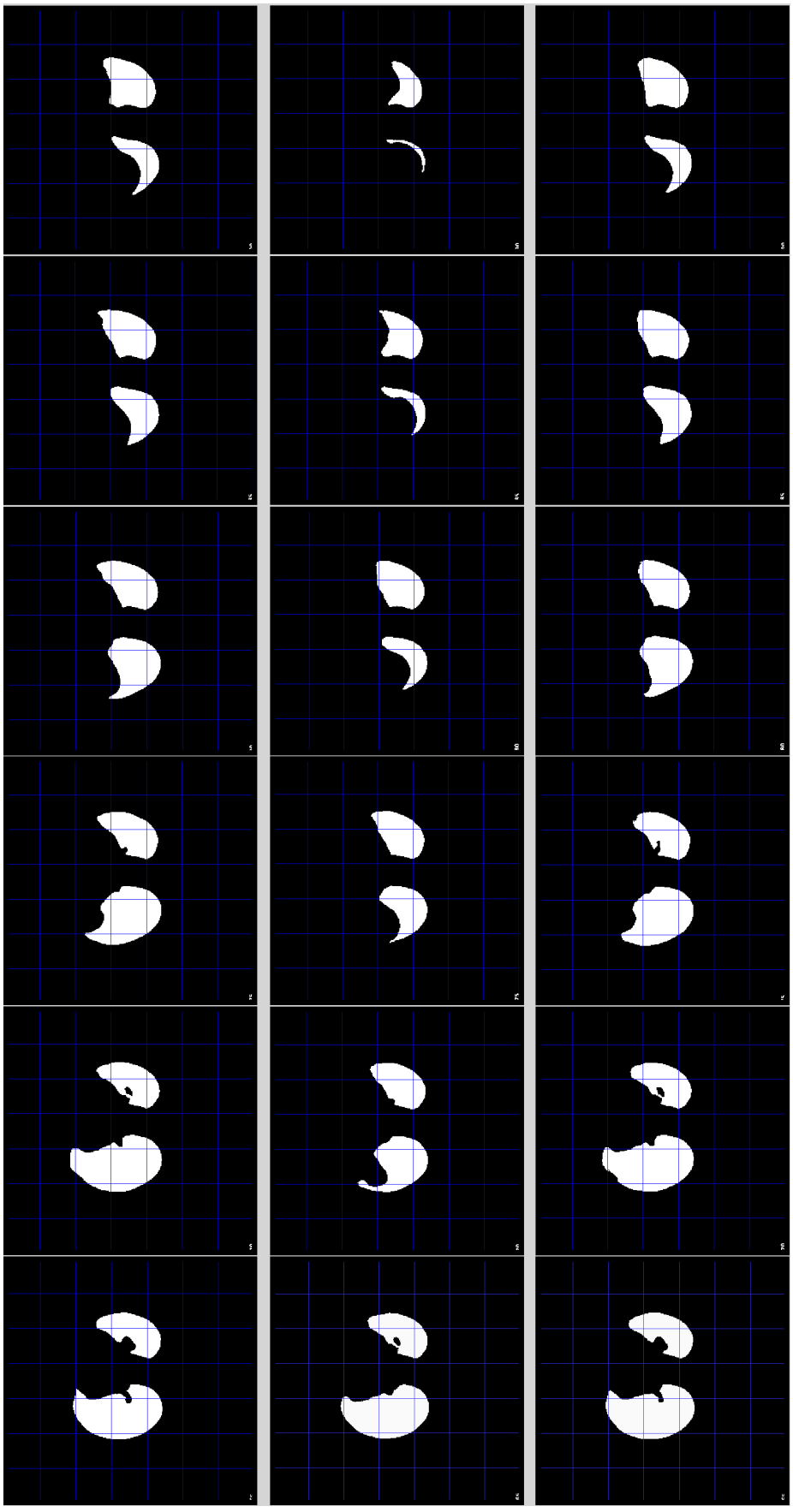


Figure 5.5 Binary masks for 4DCT Case 6, featuring slices 65, 70, 75, 80, 85, and 90. The first row displays the binary mask of the reference image; the middle row shows the binary mask of the template image; and the bottom row presents the integrated deformed image after registration. The registration was conducted using four partitions.

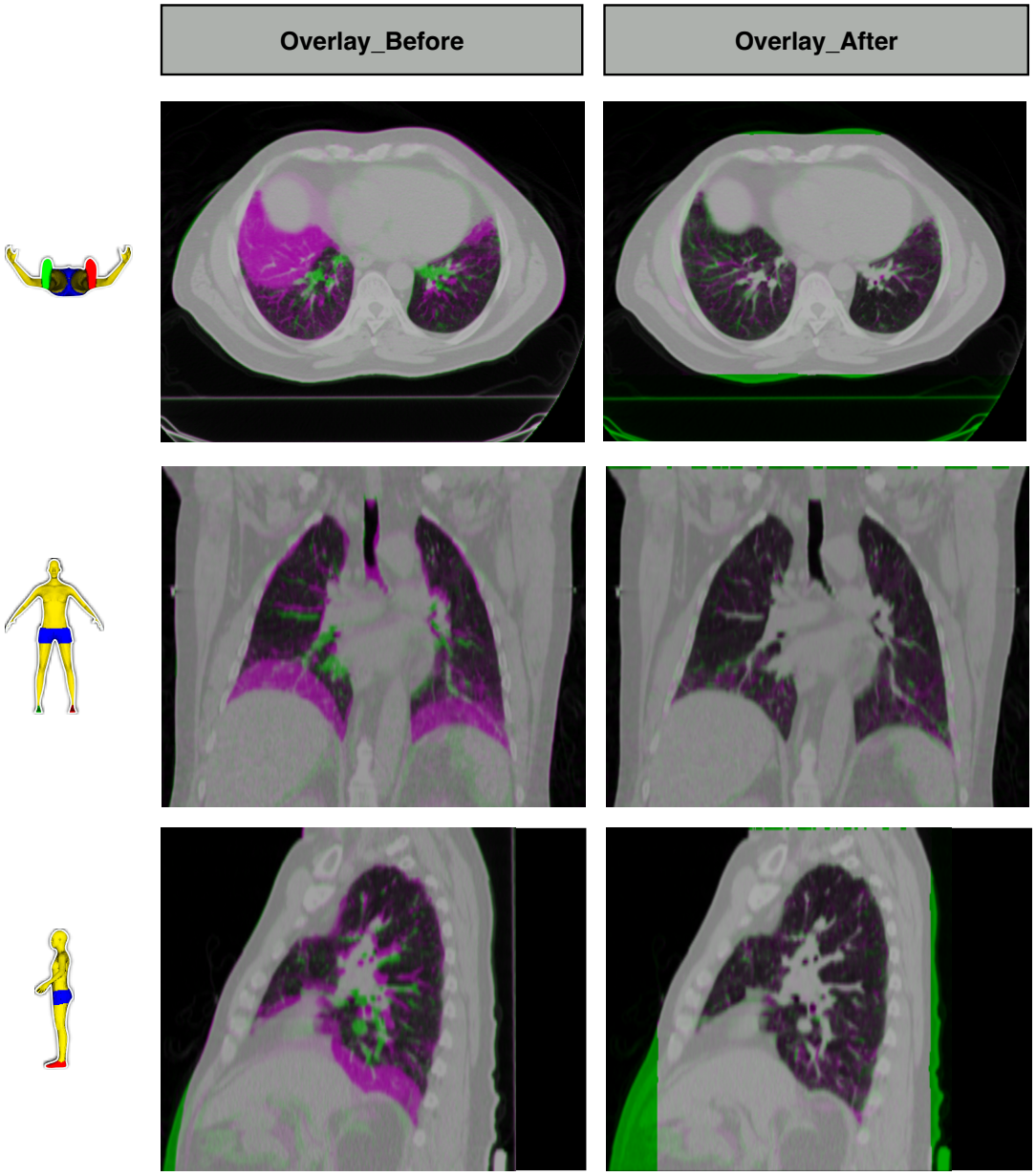


Figure 5.6 Registration results for 4DCT Case 6 with 4 partitions. The left column shows the overlays of the reference and template images before registration, while the right column displays the results after registration. The results are shown in all viewpoints. Image slice position: 78.

As reported in Table 5.2, the Dice scores showed a slight increase in the 16 GPU configuration for Case 7 and Case 8. The 16-partition configuration (using 16 GPUs) showed greater accuracy in capturing subtle details compared to configurations with fewer partitions, such as those with 8 or 4 GPUs.

In Figure 5.8, we present the overlay of deformed images obtained with 2 and 16 partitions for Case 7 from an axial perspective. The most prominent

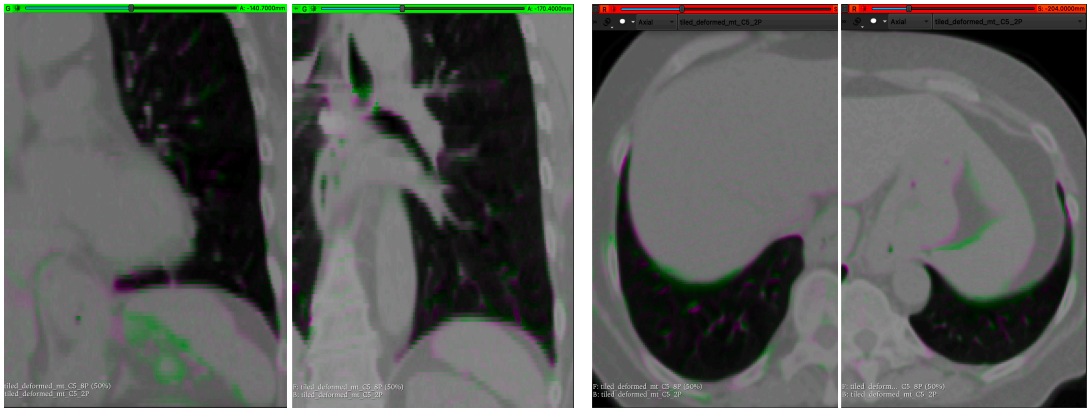


Figure 5.7 Overlay of deformed images achieved with 2 and 8 partitions for Case 5 from axial and coronal point of view.

differences are observed in the lower lung regions, where the increased precision of the 16-partition configuration is clearly visible.

While the Dice scores for 2, 4, and 8 partitions remained consistent, our analysis highlights slight differences in specific slices, particularly in the lower regions of the lung. In Figure 5.9, we display the overlay of deformed images obtained with 2, 4, 8, and 16 partitions for Case 8, focusing on these specific slices. As illustrated, increasing the number of partitions correlates with higher accuracy in capturing these subtle details.

The observed improvements for Case 7 and Case 8 are likely due to the finer granularity of the partitioning process, which facilitates more precise alignment of complex anatomical structures, particularly in regions where the lung is just beginning to become visible.

5.4 Summary

In this chapter, we introduced CLAIRE-ROP, an enhanced version of the CLAIRE framework, specifically designed to improve scalability and reduce computational overhead. In contrast to CLAIRE, which involves significant communication and struggles to scale across multiple nodes, CLAIRE-ROP minimizes communication by allowing each GPU to work independently on localized data partitions. This design approach simulates the experience of programming a single GPU, freeing developers from the complexities associated with network communication, memory management, and data transfers.

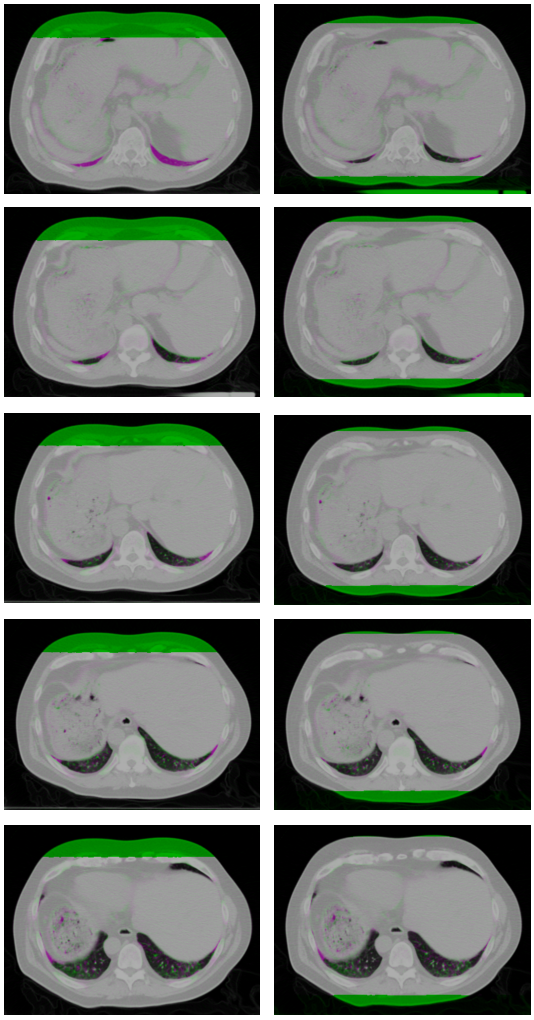


Figure 5.8 Overlay of deformed images achieved with 2 and 16 partitions for Case 7, viewed axially. The increased accuracy of the 16-partition configuration on the right is clearly visible in the selected slices.

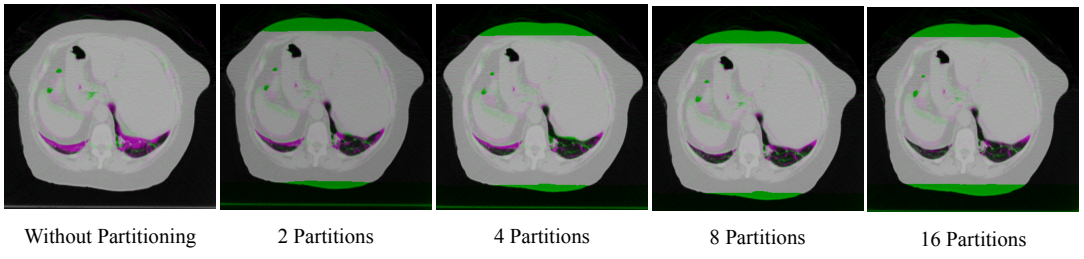


Figure 5.9 Overlay of deformed images achieved with 2, 4, 8, and 16 partitions for Case 8, viewed axially. A slice is selected where the largest differences are observed, illustrating that an increasing number of partitions results in more accurate registration.

The advantages of CLAIRE-ROP are particularly evident in environments where inter-GPU communication significantly hampers performance. By effectively managing the boundaries between partitions, we ensure the seamless integration of the separately registered images. The validation and testing phases were instrumental in confirming that no artifacts or discontinuities compromised the registration accuracy at the partition boundaries.

The findings of this study indicate that CLAIRE-ROP effectively addresses the scalability challenges inherent to the traditional CLAIRE framework. The acceleration of the computation process is achieved while maintaining high registration accuracy across distributed computing environments. Furthermore, in some cases, the accuracy of registration was enhanced with an increasing number of partitions.

Looking ahead, further investigation into scalability bottlenecks, particularly for larger datasets that present unique challenges, will be essential. Addressing these issues is crucial for advancing the capabilities of distributed image registration frameworks. Potential future directions may include developing more sophisticated load-balancing algorithms or exploring new GPU technologies that provide better interconnectivity and reduced communication overhead. These advancements could enhance the framework's ability to handle increasingly complex and large-scale registration tasks in real-time applications.

CHAPTER 6

Evaluation

6.1 Optimization-based Deformable Image Registration

6.1.1 Introduction

Evaluating DIR methods necessitates a comprehensive understanding of the total execution time, which encompasses pre-processing, processing, and post-processing stages. These steps, integrated with CLAIRE-ROP, were discussed in detail in the previous chapter. This chapter analyzes the overhead associated with each of these stages for the methods: ANTs, CLAIRE, CLAIRE-ROP, and CLAIRE-ROP-EX.

6.1.2 Comparative Analysis: CLAIRE-ROP vs. Optimization -based Methods

CLAIRE-ROP integrates partitioning into the pre-processing step, performs individual registrations for partitioned image sets during processing, and merges these partitions in the post-processing step. In CLAIRE-ROP-EX, an additional pre-processing step generates masks, partitions these masks, and calculates the lung content in each partitioned mask.

Table 6.1 summarizes the time taken for all three steps for different methods in Case 1 (small dataset) and Case 6 (large dataset). The pre-processing step includes time for mask generation and partitioning of image sets and masks, while the post-processing step involves cropping partitions and merging them. Key points of comparison include:

- **ANTs:** This method has the highest pre-processing time, particularly in Case 6, due to extensive mask generation. The processing time is also significantly longer compared to other methods, resulting in a substantial total execution time.
- **CLAIRE:** The traditional CLAIRE method reports no pre-processing time reported, as it assumes complete image input. Its processing time is considerably faster than ANTs, but it lacks the partitioning and merging efficiencies demonstrated in CLAIRE-ROP.
- **CLAIRE-ROP:** This method shows minimal pre-processing time (0.02 seconds in Case 1 and 0.05 seconds in Case 6) and post-processing time (0.08 seconds in Case 1 and 0.14 seconds in Case 6). This efficiency is achieved through automated partitioning and merging processes, resulting in a relatively small additional computational load.
- **CLAIRE-ROP-EX:** This variant includes additional pre-processing (mask generation) and is tailored for large datasets, where resource conservation is crucial. Despite the increased pre-processing time (92 seconds), the processing time remains comparable to CLAIRE-ROP, and the total execution time is still significantly lower than ANTs.

Table 6.1 Comparison of pre-processing, processing, post-processing, and total times (seconds) across different methods. CLAIRE-ROP shows a slight increase in computational load during the pre- and post-processing steps but achieves real-time image registration time.

DIR Method	Pre-processing			DIR	Post-processing		Total Time
	Mask	Partitioning	Mask partitioning		Crop	Merge	
Case 1							
ANTs	33	-	-	113.00	-	-	146
CLAIRE	-	-	-	0.93	-	-	0.93
CLAIRE-ROP	-	0.02	-	0.32	0.08	0.00	0.42
Case 6							
ANTs	92	-	-	519.00	-	-	611
CLAIRE	-	-	-	5.00	-	-	5.00
CLAIRE-ROP	-	0.05	-	0.56	0.14	0.03	0.78
CLAIRE-ROP-EX	92	0.05	0.07	0.56	0.14	0.03	93

6.1.3 Discussion

Overall, while CLAIRE-ROP introduces a minor computational load during pre- and post-processing, it maintains high efficiency. It significantly reduces pre-processing and post-processing times compared to ANTs and registration time compared to CLAIRE. Although CLAIRE-ROP-EX increases pre-processing time due to mask generation, it offers computational savings for large datasets through selective registration of edge partitions. Notably, two of the five large DIR-Lab datasets allowed for this selective registration, further optimizing resource usage. Importantly, all pre- and post-processing steps in both methods are fully automated, eliminating the need for manual intervention and further contributing to the overall efficiency of our methods.

6.2 DL-based Deformable Image Registration

6.2.1 Introduction

Deep learning (DL) methods for medical image registration have seen a remarkable surge in development over the past few years. Unlike optimization-based methods that use iterative optimization based on image similarity, DL-based methods use machine learning algorithms to train and directly estimate a transformation from two input images. This approach can potentially enhance speed, as it predicts transformations without requiring optimization steps. In recent years, several studies have demonstrated that DL-based methods can achieve higher accuracy and efficiency than their optimization-based counterparts. However, they also face several challenges and limitations. For instance, DL methods heavily depend on the availability and quality of training data, and they may struggle to generalize across images from different scanners or anatomical structures.

In this chapter, we explain how our framework overcomes the drawbacks of optimization-based methods. We then evaluate DL-based methods, particularly focusing on the time-related aspects, demonstrating the superiority of our framework over DL-based approaches in terms of speed. The primary aim of this discussion is to outline the comparative techniques utilized in recent research, with a particular emphasis on reporting results using the DIR-Lab 4DCT dataset as a standardized benchmark for assessment.

6.2.2 Comparative Analysis: CLAIRE-ROP vs. DL-based Methods

DL-based studies often highlight certain drawbacks of optimization-based methods for DIR [54, 26]. Some common challenges include:

- Parameter tuning: Optimization-based methods usually require careful parameter tuning to achieve optimal performance. This often necessitates multiple experiments and evaluations to identify the most effective parameter sets.
- Computational intensity: The iterative nature of optimization-based DIRs makes them computationally intensive, particularly for large image datasets. This demand stems from repetitive spatial filtering on deformation vector fields during optimization. Although various strategies exist for registering 4DCT DIR, the runtime is often unsatisfactory. In contrast, DL-based methods offer a significant advantage in terms of speed.

However, it is important to emphasize that CLAIRE, as an optimization-based method, does not exhibit these drawbacks:

- When applying CLAIRE to CT lung images, we observed a low sensitivity to reduction in β, aligning with results discussed in [13, 14]. CLAIRE demonstrates mostly β-independence, as illustrated in Chapter 4, Figure 4.4, where accuracy shows only slight degradation within the range from 1e$-$2 to 1e$-$3. To streamline the process, we performed parameter tuning for lung images only once, identifying optimal values that strike a trade-off between accuracy and registration time.
- In [14], the authors introduced a preconditioner to keep the number of iterations as small as possible in CLAIRE. When applied to the 4DCT dataset, this approach achieved fast convergence, requiring only 7 to 9 iterations, and this remained consistent across different resolution levels.

Although DL-based methods have made notable advancements, existing research has often prioritized registration accuracy over computational speed [25]. Nevertheless, it is essential to establish benchmarks for both registration accuracy and computational time in image registration. Despite the notable progress made through these advanced techniques, we underscore a key advantage

Table 6.2 Comparison of DL-based DIR methods on the 4DCT DIR-Lab dataset.

Reference	Network image size	Different dataset for testing?	Code available?	Supervision type	**Runtime**
[54]	$272 \times 240 \times 96$	Yes	No	Unsupervised	0.48
[26]	NA	Yes	No	Unsupervised	60
[36]	NA	No	Yes	Unsupervised	90
[35]	$192 \times 160 \times 192$	No	Yes	Unsupervised	2
[73]	$256 \times 256 \times 103$	Yes	Yes	Supervised	2.2
[22]	$128 \times 128 \times 128$	Yes	No	Supervised	0.58

of CLAIRE-ROP — exceptional speed. To illustrate our findings, Table 6.2 presents a comparative analysis, highlighting the registration times of various methods. This table also serves as a concise reference for readers seeking a summarized comparison.

6.2.3 Discussion: Evaluating DL-based DIR Performance, Challenges, and Future Prospects

Most reported runtimes deviate significantly from the real-time registration, as shown in studies such as [26, 36]. The most promising method, which is comparable to our results, is presented in [54]; however, the corresponding code for reproducing the results is not accessible. A key limitation of this method is that the network can only accept small images in current implementations, constrained by the available GPU RAM required to keep the entire network in memory. This is a common issue with DL-based methods. As shown in Table 6.2, images are typically cropped to a rough bounding box of the lungs to facilitate training in all these studies. In contrast, our approach preserves most of the thoracic area. We achieved an average registration time of 0.48 seconds for the large dataset, featuring image dimensions of 512×256 in the X and Y dimensions, while maintaining an unchanged number of slices in the Z dimension.

Two supervised methods are presented in [73, 22]. The challenge with supervised transformation prediction methods arises from the need for known ground truth transformations during network training, making it difficult to artificially generate realistic transformations.

In [22], the reported result of 0.58 seconds aligns closely with our results. However, like many studies, the code remains inaccessible. Furthermore, their method requires resizing images to $128 \times 128 \times 128$, a dimension considered impractically small.

To validate the methodology and assess generalization to different data, [54, 26, 73, 22] employed the trained network to register expiration images to inspiration images from a distinct set of test data — the DIR-Lab dataset. However, a consistent observation in all DL-based studies is the need to carefully select training data to comprehensively address potential future registration tasks. Moreover, significant effort is required for data preparation, and in particular the time spent on pre- or post-processing is often omitted in the reports. In [22], they showed that registration accuracy improves marginally on a similar dataset. However, this reported accuracy improvement was observed under conditions where the training and testing datasets were identical, which may not fully reflect performance with different datasets. Specifically, when the same dataset was used for both training and testing, the average target registration error was 1.52 ± 0.85. In contrast, when trained on different datasets, the error increased to 1.86 ± 1.17.

Building upon these limitations, the study by [23] introduces an innovative one-shot registration method for tracking periodic motion in 3D and 4D datasets without requiring extensive training data. However, it is worth mentioning that this approach comes with the drawback of prolonged computational time, reported as 26 minutes.

Given these challenges, DL-based methods are rapidly advancing and hold promise for future advancements in DIR. Despite this potential, these methods are currently more theoretical and not yet fully practical for widespread clinical or real-world applications. Challenges related to data requirements, computational demands, and algorithmic complexities still hinder their immediate use. However, with ongoing advancements in computational hardware and algorithmic efficiency, there is optimism that DL-based DIR methods will soon overcome these limitations, and provide more robust and versatile solutions for complex imaging tasks.

6.3 Summary

Our comparative evaluation demonstrates that CLAIRE-ROP and CLAIRE-ROP-EX significantly enhance processing efficiency in comparison to traditional methods. Additionally, both approaches are fully automated. Although DL-based methods demonstrate potential for increased speed and accuracy, they currently encounter practical limitations that restrict their broader adoption. It is noteworthy that our framework achieves the fastest registration times for the DIR-Lab dataset among all published 4DCT DIR methods, both DL and non-DL-based, while maintaining competitive registration accuracy. This places our approach at the forefront of deformable image registration solutions.

CHAPTER 7

Outlook

This chapter highlights the broader impact of the proposed technique and methodology, as well as potential future directions.

7.1 Customized DIR

In this study, the CLAIRE-ROP framework demonstrated significant improvements in multi-GPU image registration, particularly in reducing communication overhead, enhancing load balancing, and improving scalability. However, challenges persist in effectively utilizing all GPUs, especially when workloads are not evenly divisible or when computational demands vary significantly across image partitions.

As a potential direction for future research, a dynamic load-balancing strategy can be explored to address these challenges.

This strategy would involve adjusting partition sizes based on their content and computational complexity, ensuring a more balanced workload distribution across GPUs. The approach could include a detailed mask analysis to evaluate lung areas within each partition, guiding the adjustment of partition sizes according to specific workload characteristics.

This dynamic load-balancing approach aims to enhance both efficiency and effectiveness in the registration process. By optimizing partition sizes according to their computational demands, it may improve scalability and allow for more

effective utilization of additional GPUs. For example, while scalability was limited with 16 partitions, as discussed in Chapter 5, this strategy could enable better management of smaller partition sizes and more efficient use of GPU resources.

The optimization process would involve calculating residuals between the reference and template masks, partitioning these differential masks using the homogeneous partitioning layout described in Chapter 5, and computing the lung area percentage for each partition. This strategy mirrors CLAIRE-ROP-EX's approach to conserving resources; however, in this case, the residual masks would be partitioned instead of the template mask, refining the partitioning strategy better to capture the dynamics between the reference and template images. Building on these observations, a dynamic partitioning algorithm may be designed to adjust partition sizes according to workload characteristics.

This workload-aware methodology would optimize partition sizes and improve GPU efficiency by selectively processing partitions based on data volume. Partitions with data volumes below a predefined threshold can be excluded from registration, thereby conserving computational resources. Additionally, the approach could be tailored to the desired result accuracy by omitting partitions with minimal lung volume. This strategy seeks to maximize GPU resource utilization while preserving analysis integrity where it matters most.

7.2 Adaptive Partitioning-Based Registration for Radiotherapy Aligned with Dose Distribution Plan

In the context of RT, our partitioning-based method introduces a significant advancement by incorporating an adaptive registration technique that aligns with individualized dose distribution plans. Figure 7.1 illustrates the variation in dose distribution for seven patients, providing visual evidence of the benefits of customizing the partitioning approach based on each patient's unique anatomical features and treatment plans. This adaptive partitioning algorithm enables fine-tuning of DIR, leading to more accurate and effective treatment plans.

The methodology supports the customization of parameters based on tumor location and specific anatomical characteristics, which enhances accuracy and

potentially accelerates the registration process. By optimizing the DIR process within the IGART application, higher accuracy is achieved in critical regions, while controlled compromises are permitted in less critical areas. This focus on key anatomical details aims to improve the overall performance of the registration process.

A key element of this adaptive tuning is adjusting the regularization parameter based on the tumor's location within the lung. For example, increasing the regularization parameter in the right lung when the tumor is located in the left lung encourages the registration algorithm to favor smoother deformations in the right lung. This balance between the smoothness of the deformation field and accuracy ensures optimal results.

Overall, this approach integrates clinical context with dose distribution plans, tailoring parameter adjustments within the partitioning technique. By optimizing DIR to meet the specific anatomical and treatment requirements of each patient in IGART, this methodology holds the potential to improve therapeutic outcomes.

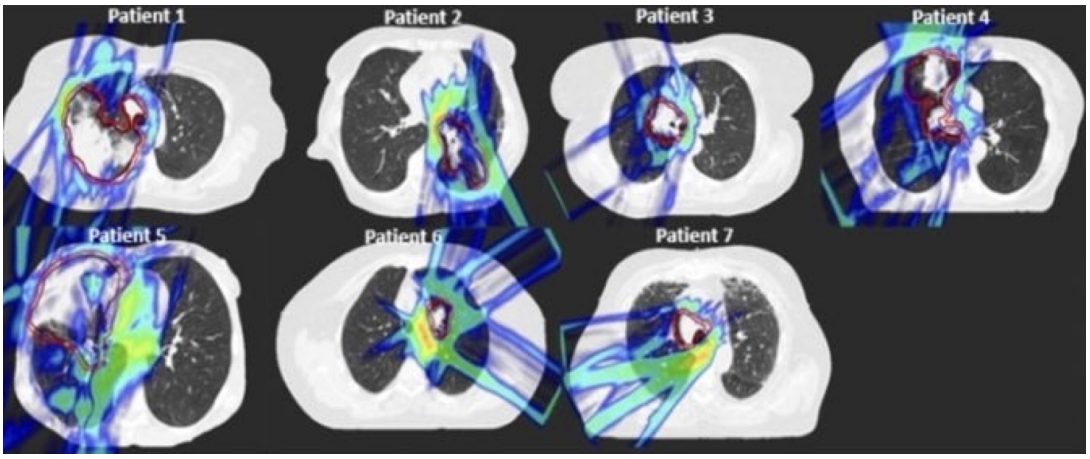


Figure 7.1 Dose distribution plans across seven patients [65].

7.3 Enhancing Computational Efficiency with Multi-Instance GPUs

Future developments may explore the possibility of leveraging multi-instance GPU (MIG) support to boost computational efficiency. Introduced with the NVIDIA Ampere architecture, MIG enables the dynamic partitioning of GPU resources, allowing for tailored memory allocation and compute resources for each

task. The MIG functionality on the NVIDIA A100 Tensor Core GPU ensures reliable performance for up to seven concurrent jobs on the same GPU, with each instance isolated in terms of compute, memory, and bandwidth. This capability facilitates optimal GPU resource allocation for each job, thereby eliminating interference between tasks[1] [19].

MIG technology is widely adopted in machine learning (ML) research due to its ability to meet the diverse and demanding computational requirements of these workloads. ML experiments vary significantly in resource needs, and MIG can facilitate precise resource allocation to address these demands efficiently [50, 69]. Furthermore, recent NVIDIA GPUs use a combination of MIG for coarse-grained physical partitioning and Multi-Process Service (MPS) for finer-grained logical partitioning. MPS is designed to enable multiple processes to share a single GPU while ensuring efficient resource management. This hierarchical approach maximizes resource utilization and improves performance by providing both physical and logical isolation of computing tasks [69].

CLAIRE-ROP can similarly benefit from this technology. Registering smaller image partitions may lead to underutilization of GPU SM cores, resulting in resource wastage. Instead of assigning an entire GPU to each partition, we allocate each partition to a separate GPU instance, with adjustable memory allocations. This strategy enables more granular control over resource use, ensuring that each partition is processed efficiently without wasting GPU capacity. By carefully allocating GPU resources for each registration task, the registration of the entire image can be optimized.

However, the effective implementation of this strategy requires a thorough characterization of MIG slices and workload distribution for each registration task. In [50], the authors characterized MIG slices across various application domains, including computer vision, language modeling, speech recognition, and scientific computing. They found that different ML models exhibit varying sensitivities to MIG slicing. Depending on the specific workload, using a partial MIG slice instead of a full A100 GPU can result in either minimal or significant performance degradation.

[1] https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html

7.4 Adapting CLAIRE-ROP for Multimodal Image Registration

Multimodal registration of medical images has become a recognized tool in clinical practice for fusing information from various imaging techniques across multiple medical fields, including oncology, cardiology, and neurology. This advancement facilitates the integration of diverse imaging modalities, such as CT, MRI, and PET, thereby enhancing holistic analysis and improving accuracy in clinical diagnostics and treatment planning. The focus is on developing and implementing algorithms that can effectively address the complexities associated with aligning different types of medical imaging data, ultimately enhancing the versatility and utility of the IR framework in various medical applications [28, 81, 48].

However, the involvement of multiple imaging modalities presents significant challenges and increases computational intensity. Although CLAIRE has added support for new distance measures to tackle multimodal registration challenges, this aspect has not been sufficiently explored.

Future work in multimodal image registration presents promising opportunities, particularly in the context of tumor imaging. Expanding the focus to include such scenarios would enrich the scope and applicability of this research. This extension could address more intricate and clinically relevant challenges in medical imaging, enhancing the utility of CLAIRE-ROP in diverse diagnostic and treatment settings. Exploring multimodal registrations, especially in oncology, could yield critical insights and advancements in the precise detection, characterization, and monitoring of tumors.

7.5 Closing Remarks

Modern radiation therapy applications require a planning process that is both time-efficient and accurate. Given that anatomical changes during treatment are often deformable, DIR plays a crucial role in radiation therapies by enabling optimal alignment of anatomy between two images.

In this dissertation, a real-time DIR framework was presented for use in radiation therapy applications, utilizing a multi-GPU architecture to accelerate

processing. The proposed accelerated registration method demonstrates considerable potential as an effective solution for tackling registration challenges. By markedly reducing communication overhead, it enhances overall performance and facilitates more efficient medical imaging processes. Nevertheless, enhancing the efficacy of medical image registration is subject to particular circumstances and is influenced by a multitude of factors, including clinical and research requirements concerning accuracy and computational complexity.

Looking to the future, it is essential to continue exploring innovative methodologies and technologies that enhance the performance and applicability of DIR across various medical contexts. Embracing advancements in computational resources, such as multi-instance GPUs and dynamic load-balancing strategies, will optimize image registration processes. Additionally, integrating multimodal imaging techniques can provide richer data for analysis, improving diagnostic accuracy and treatment planning.

By maintaining a strong focus on research and development, this work can address the complex challenges associated with DIR, potentially leading to improved patient outcomes in radiotherapy and other medical applications. Ongoing collaboration with clinical experts will help ensure that the methodology aligns with real-world requirements, facilitating effective implementation in diverse healthcare settings.

Bibliography

[1] Meng Wang et al. “Extended Coding Unit Partitioning for Future Video Coding”. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 2931–2946. DOI: 10.1109/TIP.2019.2955238.

[2] Pauli Virtanen Ralf et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[3] Sultan Durrani et al. “Accelerating Fourier and Number Theoretic Transforms using Tensor Cores and Warp Shuffles”. In: *30th International Conference on Parallel Architectures and Compilation Techniques (PACT), Atlanta, GA, USA* (2021), pp. 345–355. DOI: 10.1109/PACT52795.2021.00032.

[4] Natan Andrade, Fabio Augusto Faria, and Fábio Augusto Menocci Cappabianco. “A Practical Review on Medical Image Registration: from Rigid to Deep Learning based Approaches”. In: *Conference on Graphics, Patterns and Images (SIBGRAPI)* (Jan. 2019), pp. 463–470. DOI: 10.1109/SIBGRAPI.2018.00066.

[5] John Ashburner. “A fast diffeomorphic image registration algorithm”. In: *NeuroImage* 38.1 (Oct. 2007), pp. 95–113. DOI: 10.1016/j.neuroimage.2007.07.007.

[6] B. B. Avants et al. “Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain”. In: *Medical Image Analysis* 12 (Feb. 2008), pp. 26–41. DOI: 10.1016/j.media.2007.06.004.

[7] Brian B. Avants, Nick Tustison, and Hans Johnson. “Advanced Normalization Tools (ANTS) Release 2.x”. In: *Insight j* 2.365 (July 2009), pp. 1–35.

[8] Brian B. Avants et al. “A reproducible evaluation of ANTs similarity metric performance in brain image registration”. In: *NeuroImage* 54.3 (Feb. 2011), pp. 2033–2044. DOI: 10.1016/j.neuroimage.2010.09.025.

[9] Ahmad Al Badawi et al. “Multi-GPU Design and Performance Evaluation of Homomorphic Encryption on GPU Clusters”. In: *IEEE Transactions on Parallel and Distributed Systems* 32.2 (Feb. 2021), pp. 379–391. DOI: 10.1109/TPDS.2020.3021238.

[10] M. Faisal Beg et al. “Computing large deformation metric mappings via geodesic flows of diffeomorphisms”. In: *International Journal of Computer Vision* 61.2 (Feb. 2005), pp. 139–157. DOI: 10.1023/B:VISI.0000043755.93987.aa.

[11] Alexandre Bône et al. “Deformetrica 4: An open-source software for statistical shape analysis”. In: *Proc International Workshop on Shape in Medical Imaging* (Nov. 2018), pp. 3–13. DOI: 10.1007/978-3-030-04747-4_1.

[12] Malte Brunn et al. “CLAIRE: Constrained Large Deformation Diffeomorphic Image Registration on Parallel Computing Architectures”. In: *Journal of Open Source Software* 6.61 (May 2021). DOI: 10.21105/joss.03038.

[13] Malte Brunn et al. “Fast GPU 3D diffeomorphic image registration”. In: *J. Parallel Distrib. Comput.* 149 (Mar. 2021), pp. 149–162. DOI: 10.1016/j.jpdc.2020.11.006.

[14] Malte Brunn et al. “Multi-Node Multi-GPU Diffeomorphic Image Registration for Large-Scale Imaging Problems”. In: *Int Conf High Perform Comput Netw Storage Anal.* (Nov. 2020), pp. 1–17. DOI: 10.1109/sc41405.2020.00042.

[15] Edward Castillo et al. “Four-dimensional deformable image registration using trajectory modeling”. In: *Phys. Med. Biol.* 55 (Jan. 2010), pp. 305–327. DOI: 10.1088/0031-9155/55/1/018.

[16] Richard Castillo et al. “A framework for evaluation of deformable image registration spatial accuracy using large landmark point sets”. In: *Phys. Med. Biol.* 54 (Mar. 2009), pp. 1849–1870. DOI: 10.1088/0031-9155/54/7/001.

[17] Raheel Chaudhry, Adekunle E. Omole, and Bruno Bordoni. *Anatomy, Thorax, Lungs*. Treasure Island (FL): StatPearls Publishing, 2024.

[18] Yang Chen et al. “Segmentation of lung nodules based on a refined segmentation network”. In: *Med Phys.* 51.4 (Apr. 2024), pp. 2759–2771. DOI: 10.1002/mp.16900.

[19] Jack Choquette et al. “3.2 The A100 Datacenter GPU and Ampere Architecture”. In: *2021 IEEE International Solid-State Circuits Conference (ISSCC)* (2021), pp. 48–50. DOI: 10.1109/ISSCC42613.2021.9365803.

[20] Gary E Christensen et al. “Introduction to the non-rigid image registration evaluation project, in Proc Biomedical Image Registration”. In: *Biomedical Image Registration, Third International Workshop, WBIR 2006.* 4057 (July 2006), pp. 128–135. DOI: 10.1007/11784012_16.

[21] Nicolas Courty and Pierre Hellier. “Accelerating 3D Non-Rigid Registration using Graphics Hardware”. In: *International Journal of Image and Graphics* 8.1 (2008), pp. 1–18.

[22] Koen A. J. Eppenhof and Josien P. W. Pluim. “Pulmonary CT Registration Through Supervised Learning With Convolutional Neural Networks”. In:

IEEE Trans Med Imaging 38.5 (May 2019), pp. 1097–1105. DOI: 10.1109/TMI.2018.2878316.

[23] Tobias Fechter and Dimos Baltas. "One Shot Learning for Deformable Medical Image Registration and Periodic Motion Tracking". In: *IEEE Trans Med Imaging* 39.7 (July 2020), pp. 2506–2517. DOI: 10.1109/TMI.2020.2972616.

[24] Andriy Fedorov et al. "3D Slicer as an Image Computing Platform for the Quantitative Imaging Network". In: *Magnetic Resonance Imaging* 30.9 (Nov. 2012), pp. 1323–1341. DOI: 10.1016/j.mri.2012.05.001.

[25] Yabo Fu et al. "Deep learning in medical image registration: a review". In: *Int Conf High Perform Comput Netw Storage Anal.* 65.20 (Oct. 2020), pp. 1–17. DOI: 10.1088/1361-6560/ab843e.

[26] Yabo Fu et al. "LungRegNet: An unsupervised deformable image registration method for 4D-CT lung". In: *Med. Phys.* 47.4 (Apr. 2020), pp. 1763–1774. DOI: 10.1002/mp.14065.

[27] Yabu Fu et al. "An adaptive motion regularization technique to support sliding motion in deformable image registration". In: *Med Phys.* 45.2 (Jan. 2018), pp. 735–747. DOI: 10.1002/mp.12734.

[28] Simone Garzia et al. "Three-Dimensional Multi-Modality Registration for Orthopaedics and Cardiovascular Settings: State-of-the-Art and Clinical Applications". In: *Sensors (Basel)* 24.4 (Feb. 2024). DOI: 10.3390/s24041072.

[29] Amir Gholami et al. "A framework for scalable biophysics-based image analysis". In: *Proc ACM/IEEE Conference on Supercomputing* 19 (2017), pp. 1–13. DOI: 10.1145/3126908.3126930.

[30] Carri K. Glide-Hurst et al. "Adaptive Radiation Therapy (ART) Strategies and Technical Considerations: A State of the ART Review From NRG Oncology". In: *Int J Radiation Oncol Biol Phys* 109.4 (2021), pp. 1054–1075. DOI: 10.1016/j.ijrobp.2020.10.021.

[31] Lun Gong et al. "Nonrigid Image Registration Using Spatially Region-Weighted Correlation Ratio and GPU-Acceleration". In: *IEEE J. Biomed. Health Inform.* 23.2 (Mar. 2019), pp. 766–778. DOI: 10.1109/JBHI.2018.2836380.

[32] Daniel Grzech et al. "FastReg: Fast Non-Rigid Registration via Accelerated Optimisation on the Manifold of Diffeomorphisms". In: *ArXiv* abs/1903.01905 (Mar. 2019). DOI: 10.48550/arXiv.1903.01905.

[33] Xuejun Gu et al. "Implementation and evaluation of various demons deformable image registration algorithms on a GPU". In: *Phys Med Biol* 55 (Jan. 2010), pp. 207–219. DOI: 10.1088/0031-9155/55/1/012.

[34] Linh Ha et al. "Multiscale unbiased diffeomorphic atlas construction on multi-GPUs". In: *GPU Computing Gems Emerald Edition* (2011), pp. 771–791. DOI: 10.1016/B978-0-12-384988-5.00048-6.

[35] Lasse Hansen and Mattias P. Heinrich. "GraphRegNet: Deep Graph Regularisation Networks on Sparse Keypoints for Dense Registration of 3D Lung CTs". In: *IEEE Trans Med Imaging* 40.9 (Sept. 2021), pp. 2246–2257. DOI: 10.1109/TMI.2021.3073986.

[36] Louis D. van Harten, Jaap Stoker, and Ivana Išgum. "Robust deformable image registration using cycle-consistent implicit representations". In: *IEEE Trans Med Imaging* 43.2 (Feb. 2024), pp. 784–793. DOI: 10.1109/TMI.2023.3321425.

[37] Mattias P. Heinrich et al. "MRF-based deformable registration and ventilation estimation of lung CT". In: *IEEE Trans Med Imaging* 32.7 (July 2013), pp. 1239–1248. DOI: 10.1109/TMI.2013.2246577.

[38] Stijn Heldens et al. "Lightning: Scaling the GPU Programming Model Beyond a Single GPU". In: *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)* (2022), pp. 492–503. DOI: 10.1109/IPDPS53621.2022.00054.

[39] Naveen Hitmani et al. "CLAIRE—Parallelized Diffeomorphic Image Registration for Large-Scale Biomedical Imaging Applications". In: *Journal of Imaging* 8.9 (Sept. 2022). DOI: 10.3390/jimaging8090251.

[40] Fabian Isensee et al. "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature methods". In: *Proceedings 14th IEEE Symposium on Computer-Based Medical Systems.* 18.2 (2021), pp. 203–211. DOI: 10.1038/s41592-020-01008-z.

[41] Xun Jia, Peter Ziegenhein, and Steve B. Jiang. "GPU-based High-Performance Computing for Radiation Therapy". In: *Phys Med Biol* 59.4 (Feb. 2014), pp. 151–182. DOI: 10.1088/0031-9155/59/4/R151.

[42] S. Joshi et al. "Unbiased diffeomorphic atlas construction for computational anatomy". In: *NeuroImage* 23 (Sept. 2004), pp. 151–160. DOI: 10.1016/j.neuroimage.2004.07.068.

[43] Ha Linh K. et al. "Fast parallel unbiased diffeomorphic atlas construction on multi-graphics processing units". In: *Proceedings of the 9th Eurographics conference on Parallel Graphics and Visualization* (Mar. 2009), pp. 41–48. DOI: 10.2312/EGPGV/EGPGV09/041-048.

[44] Brock KK et al. "Use of image registration and fusion algorithms and techniques in radiotherapy: Report of the AAPM Radiation Therapy Committee Task Group No. 132". In: *Med Phys.* 44.7 (July 2017), pp. 42–76. DOI: 10.1002/mp.12256.

[45] Stefan Klein et al. "elastix: A Toolbox for Intensity-Based Medical Image Registration". In: *IEEE Trans Med Imaging* 29.1 (Jan. 2010), pp. 196–205. DOI: 10.1109/TMI.2009.2035616.

[46] Akila Kumarasiri et al. "Deformable image registration based automatic CT-to-CT contour propagation for head and neck adaptive radiotherapy

in the routine clinical setting". In: *Med Phys.* 41.12 (Dec. 2014). DOI: 10.1118/1.4901409.

[47] Akila Kumarasiri et al. "Evaluation of Image Registration Accuracy for Tumor and Organs at Risk in the Thorax for Compliance With TG 132 Recommendations". In: *Adv Radiat Oncol.* 4.1 (Sept. 2018), pp. 177–185. DOI: 10.1016/j.adro.2018.08.02.

[48] L Lafitte et al. "Accelerating multi-modal image registration using a supervoxel-based variational framework". In: *Phys Med Biol* 63.23 (Nov. 2018). DOI: 10.1088/1361-6560/aaebc2.

[49] Donghoon Lee et al. "Seq2Morph: A deep learning deformable image registration algorithm for longitudinal imaging studies and adaptive radiotherapy". In: *Med Phys.* 50.2 (Feb. 2023), pp. 970–979. DOI: 10.1002/mp.16026.

[50] Baolin Li et al. "Characterizing Multi-Instance GPU for Machine Learning Workloads". In: *2022 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (2022), pp. 724–731. DOI: 10.1109/IPDPSW55747.2022.00124.

[51] Binrui Li, Shenggan Cheng, and James Lin. "tcFFT: A Fast Half-Precision FFT Library for NVIDIA Tensor Cores". In: *IEEE International Conference on Cluster Computing (CLUSTER), Portland, OR, USA* (2021), pp. 1–11. DOI: 10.1109/Cluster48925.2021.00035.

[52] Min Li et al. "GPU-accelerated Block Matching Algorithm for Deformable Registration of Lung CT Images". In: *Proc IEEE Int Conf Prog Inform Comput (PIC)* (2015), pp. 292–295. DOI: 10.1109/PIC.2015.7489856.

[53] Yixun Liu et al. "A GPU-Based Method in Recovering the Full 3D Deformation Field Using Multiple 2D Fluoroscopic Views in Lung Navigation". In: *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)* (2016), pp. 294–297. DOI: 10.1109/ISBI.2016.7493267.

[54] Jiayi Lu et al. "Lung-CRNet: A convolutional recurrent neural network for lung 4DCT image registration". In: *Med. Phys.* 48.12 (Dec. 2021), pp. 7900–7912. DOI: 10.1002/mp.15324.

[55] Tao Lu et al. "Trends in the incidence, treatment, and survival of patients with lung cancer in the last four decades". In: *Cancer Manag Res.* 11 (Jan. 2019), pp. 943–953. DOI: 10.2147/CMAR.S187317.

[56] Malathi M et al. "Segmentation of CT Lung Images Using FCM with Active Contour and CNN Classifier". In: *Asian Pac J Cancer Prev. 2022 Mar* 23.3 (2022), pp. 905–910. DOI: 10.31557/APJCP.2022.23.3.905.

[57] Andreas Mang and George Biros. "An inexact Newton–Krylov algorithm for constrained diffeomorphic image registration". In: *SIAM Journal on Imaging Sciences* 8.2 (2015), pp. 1030–1069. DOI: 10.1137/140984002.

[58] Andreas Mang, Amir Gholami, and George Biros. "Distributed-memory large deformation diffeomorphic 3D image registration". In: *SC '16: Pro-*

ceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (Nov. 2016), pp. 842–853. DOI: 10.1109/SC.2016.71.

[59] Andreas Mang et al. "CLAIRE: A distributed-memory solver for constrained large deformation diffeomorphic image registration". In: *SIAM J. Sci. Comput.* 41 (Oct. 2019), pp. C548–C584. DOI: 10.1137/18m1207818.

[60] John Cheng Max and Grossman Ty McKercher. *Professional CUDA C Programming*. John Wiley and Sons, Inc. 10475 Crosspoint Boulevard Indianapolis, IN 46256, 2014. ISBN: 978-1-118-73932-7.

[61] M. J. McAuliffe et al. "Medical Image Processing, Analysis and Visualization In Clinical Research". In: *Proceedings 14th IEEE Symposium on Computer-Based Medical Systems.* (2002), pp. 381–386. DOI: 10.1109/CBMS.2001.941749.

[62] Ross McCall et al. "Anatomical contouring variability in thoracic organs at risk". In: *Medical Dosimetry* 41.4 (Oct. 2016), pp. 344–350. DOI: 10.1016/j.meddos.2016.08.004.

[63] Marc Modata et al. "Fast free-form deformation using graphics processing units". In: *Computer Methods and Programs in Biomedicine* 98.3 (2010), pp. 278–284. DOI: 10.1016/j.cmpb.2009.09.002.

[64] Pınar Muyan-Ozcelik et al. "Fast Deformable Registration on the GPU: A CUDA Implementation of Demons". In: *Int. J. Comput. Appl.* (2008), pp. 223–233. DOI: 10.1109/ICCSA.2008.22.

[65] Lena Nenoff et al. "Deformable image registration uncertainty for inter-fractional dose accumulation of lung cancer proton therapy". In: *Radiother Oncol.* 147 (June 2020), pp. 177–185. DOI: 10.1016/j.radonc.2020.04.046.

[66] Lena Nenoff et al. "Review and recommendations on deformable image registration uncertainties for radiotherapy applications". In: *Phys Med Biol.* 68.24 (Dec. 2023), pp. 1–40. DOI: 10.1088/1361-6560/ad0d8a.

[67] Michael Osadebey et al. "Three-stage segmentation of lung region from CT images using deep neural networks". In: *BMC Med Imaging.* 21.1 (July 2021), pp. 1–19. DOI: 10.1186/s12880-021-00640-1.

[68] Louis Pisha and Lukasz Ligowski. "Accelerating non-power-of-2 size Fourier transforms with GPU Tensor Cores". In: *EEE International Parallel and Distributed Processing Symposium (IPDPS), Portland, OR, USA* (2021), pp. 507–516. DOI: 10.1109/IPDPS49936.2021.00059.

[69] Urvij Saroliya et al. "Hierarchical Resource Partitioning on Modern GPUs: A Reinforcement Learning Approach". In: *2023 IEEE International Conference on Cluster Computing (CLUSTER)* (2023), pp. 185–196. DOI: 10.1109/CLUSTER52292.2023.00023.

[70] Alexander Schmidt-Richberg et al. “Estimation of slipping organ motion by registration with direction-dependent regularization”. In: *Med Image Anal.* 16.7 (Jan. 2012), pp. 150–159. DOI: 10.1016/j.media.2011.06.007.

[71] J A Shackleford, N Kandasamy, and G C Sharp. “On developing B-spline registration algorithms for multi-core processors”. In: *Physics in Medicine and Biology* 55.21 (Oct. 2010), pp. 6329–6351. DOI: 10.1088/0031-9155/55/21/001.

[72] Moiz Khan Sherwani et al. “Lesion segmentation in lung CT scans using unsupervised adversarial learning”. In: *Med Biol Eng Comput.* 60.11 (Nov. 2022), pp. 3203–3215. DOI: 10.1007/s11517-022-02651-8.

[73] Hessam Sokooti et al. “3D Convolutional Neural Networks Image Registration Based on Efficient Supervised Learning from Artificial Deformations”. In: *eess.IV* (2019). DOI: 10.48550/arXiv.1908.10235.

[74] Nazanin Tahmasebi et al. “Real-Time Lung Tumor Tracking Using a CUDA Enabled Nonrigid Registration Algorithm for MRI”. In: *IEEE J Transl Eng Health Med* 8 (Apr. 2020), pp. 1–8. DOI: 10.1109/JTEHM.2020.2989124.

[75] ALAIN Trouvé. “Diffeomorphism groups and pattern matching in image analysis”. In: *International Journal of Computer Vision* 28.3 (1998), pp. 213–221. DOI: 10.1023/A:1008001603737.

[76] Nicholas J. Tustison et al. “Large-scale evaluation of ANTs and FreeSurfer cortical thickness measurements”. In: *NeuroImage 99* (Oct. 2014), pp. 166–179. DOI: 10.1016/j.neuroimage.2014.05.044.

[77] Pedro Valero-Lara. “A GPU Approach for Accelerating 3D Deformable Registration (DARTEL) on Brain Biomedical Images”. In: *Proc European MPI Users’ Group Meeting* (2013), pp. 187–192. DOI: 10.1145/2488551.2488592.

[78] Pedro Valero-Lara. “Multi-GPU acceleration of DARTEL (early detection of Alzheimer)”. In: *2014 IEEE International Conference on Cluster Computing (CLUSTER)* (2014), pp. 346–354. DOI: 10.1109/CLUSTER.2014.6968783.

[79] Jef Vandemeulebroucke et al. “Spatiotemporal motion estimation for respiratory-correlated imaging of the lungs”. In: *Medical physics* 38.1 (Jan. 2011), pp. 166–178. DOI: 10.1118/1.3523619.

[80] Tom Vercauteren et al. “Diffeomorphic demons: Efficient non-parametric image registration”. In: *NeuroImage* 45 (Mar. 2009), S61–S72. DOI: 10.1016/j.neuroimage.2008.10.040.

[81] Chengjia Wang, Guang Yang, and Giorgos Papanastasiou. “Unsupervised image registration towards enhancing performance and explainability in cardiac and brain image analysis”. In: *Sensors (Basel)* 22.6 (Mar. 2022). DOI: 10.3390/s22062125.

[82] Jakob Wasserthal et al. "TotalSegmentator: Robust Segmentation of 104 Anatomic Structures in CT Images". In: *Radiol Artif Intell* 5.5 (June 2023). DOI: 10.1148/ryai.230024.

[83] Jing Wu, Joseph JaJa, and Elias Balaras. "An Optimized FFT-Based Direct Poisson Solver on CUDA GPUs". In: *IEEE Transactions on Parallel and Distributed Systems* 25.3 (Mar. 2014), pp. 550–559. DOI: 10.1109/TPDS.2013.53.

[84] Peng Xue et al. "Lung Respiratory Motion Estimation Based on Fast Kalman Filtering and 4D CT Image Registration". In: *IEEE J. Biomed. Health Inform.* 25.6 (June 2021), pp. 2007–2017. DOI: 10.1109/JBHI.2020.3030071.

[85] Xiao Yang† et al. "Quicksilver: Fast Predictive Image Registration – a Deep Learning Approach". In: *NeuroImage* 158 (Sept. 2017), pp. 378–396. DOI: 10.1016/j.neuroimage.2017.07.008.

[86] Xiao Yanga et al. "Quicksilver: Fast predictive image registration – A deep learning approach". In: *NeuroImage* 158 (Mar. 2017), pp. 378–396. DOI: 10.1016/j.neuroimage.2017.07.008.

[87] Ziv Yaniv et al. "SimpleITK Image-Analysis Notebooks: a Collaborative Environment for Education and Reproducible Research". In: *J Digit Imaging.* 31.3 (2018), pp. 290–303. DOI: 10.1007/s10278-017-0037-8.

[88] Yumo Zhang, Zhanchuan Cai, and Gangqiang Xiong. "A New Image Compression Algorithm Based on Non-Uniform Partition and U-System". In: *IEEE Transactions on Multimedia* 23 (2021), pp. 1069–1082. DOI: 10.1109/TMM.2020.2992940.

[89] Zhicheng Zhao and Yaqun Zhao. "The Optimization of FFT Algorithm Based with Parallel Computing on GPU". In: *IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference(IAEAC 2018)* (2018), pp. 2003–2007. DOI: 10.1109/IAEAC.2018.8577843.

APPENDIX A

Bash and Embedded Python Scripts

Developed Code

Our DIR workflow employs a combined Bash and Python script, specifically designed for execution in an HPC environment using Slurm job scheduling. This script manages the partitioning, processing, and registration of medical imaging data, leveraging SimpleITK to enhance computational efficiency and scalability.

Pre-processing Step

The pre-processing step is managed by ‘Partitioning.sh‘ and includes three main sections:

1. **Partitioning (run_section_1=1):** This section uses a Python script to divide input images into smaller, manageable sections or "partitions". The partitioning process is controlled by environment variables and Slurm parameters, allowing flexible adjustment of GPU resources and overlap settings. Key parameters include:

    - **SLURM_GPUS:** Specifies the number of GPUs to be used. The partitioning logic adjusts dynamically based on the available GPU count, allowing for 2, 4, 8, or 16 partitions.

- **OVERLAP_VALUE:** Defines the overlap between partitions to ensure continuity between adjacent partitions.

Images are partitioned by dividing them along the X and Y axes according to the number of available GPUs, using SimpleITK's '*ExtractImageFilter*' to crop images based on specified indices and crop sizes.

2. **Mask Partitioning and Lung Coverage (run_section_2=0):** This section handles partitioning masks and calculating lung coverage percentages in the images. If this step is enabled and the lung coverage percentage is 0, edge partitions are skipped in the Multi-Image Registration (MIR) step.
3. **Proceed to Multi-Image Registration (MIR) (run_section_3=1):** After partitioning, the pipeline progresses to the image registration stage, where different Slurm batch jobs are conditionally executed based on the number of GPUs.

Processing and Post-processing Steps

The processing and post-processing stages are managed by determining which partitions to process based on the 'result' variable. If 'result' equals 0 (as determined by the partitioning bash script), only the inner partitions are processed; otherwise, all partitions are handled.

1. **Running CLAIRE and Get the Wrapped Images (run_section_4=1):** The process 'MIR.sh' begins by running two key functions:

   - `run_registration_defmap`: Generates deformation maps using the CLAIRE registration tool for each partition.
   - `run_registration_time`: Measures CLAIRE-ROP registration time with CLAIRE.

   The following bash code snippet runs the CLAIRE tool, balancing accuracy, computational efficiency, and the physical plausibility of transformations as discussed in Chapter 4:

```
mpirun ./claire -mt "$DATA/$Partition_mt.nii.gz" -mr "$DATA/
    $Partition_mr.nii.gz" -regnorm h1s-div -maxit 50 -krylovmaxit
    100 -precond invreg -iporder 1 -betacont "$betacont" -beta-div
     1e-04 -diffpde finite -x "$DATA/${defmap_name}_" -defmap
```

Key CLAIRE Parameters:

- $-mt$ and $-mr$**:** Paths to the moving and reference images, provided as '$*$.nii.gz' files.
- $-regnorm\ h1s-div$**:** Regularization norm for the velocity field, with $h1s-div$ controlling the divergence with a penalty parameter of $-beta-div$.
- $-maxit$ 50**:** Maximum number of Newton iterations for the optimization process.
- $-krylovmaxit$ 100**:** Maximum number of Krylov iterations.
- $-precond\ invreg$**:** Sets the preconditioner to the inverse regularization operator.
- $-iporder$ 1**:** Defines the interpolation order (1 for linear interpolation).
- $-beta-div\ 1e-04$**:** Sets the weight for divergence, ensuring a volume-preserving transformation.
- $-betacont\ 5e-3$**:** Controls the smoothness of the deformation field.
- $-diffpde\ finite$**:** Uses a finite differences scheme for the gradient and divergence operators.
- $-defmap$**:** Indicates that the output should include deformation maps, essential for generating the wrapped partitions.

Finally, a python script, as detailed in Chapter 3, is executed to perform image warping using the deformation maps. The script leverages the '*interpn*' function from 'SciPy' to interpolate the moving image on transformed coordinates, producing the deformed image.

2. **Cropping and Merging (run_section_5=1):** This section crops the deformed images and merges them into a single image. SimpleITK's '*CropImageFilter*' is used for cropping, and '*TileImageFilter*' is em-

ployed to merge the images into a composite, with the layout specified by '*sitk.SetLayout*' (e.g., a '4×2' grid for an 8-partition layout).

3. **Padding (run_section_6=1):** Padding is applied to the tiled images to ensure they match the reference image dimensions, preparing them for further processing.

4. **Generating Masks (run_section_7=1):** In this section, masks are generated and combined to focus on specific ROIs, such as the lungs. The '*TotalSegmentator*' tool is used for automated segmentation, and '*totalseg_combine_masks*' combines different segmentations into a single mask.

```
TotalSegmentator -i $DATA/img.nii.gz -o $DATA/mask_img
totalseg_combine_masks -i $DATA/mask_img -o $DATA/mask_img.nii
    .gz -m lung
```

5. **Computing Dice Score (run_section_8=1):** This script computes the Dice score, a measure of similarity between two segmentation masks, using SimpleITK's '*LabelOverlapMeasuresImageFilter*'. The results are stored in a NumPy array and printed for both the original and deformed images, allowing evaluation of registration accuracy.

ANTs

We used the 'antsRegistrationSyN.sh' script from the ANTs toolkit with specific flags to perform image registration. The details of the flags and their implications are discussed in Chapter 4.

Key ANTs Parameters:

- $-d$ 3**:** Image dimension (3D registration of a single volume).
- $-f$**:** Reference image.
- $-m$**:** Template image.
- $-x$**:** Mask for the reference image space.
- $-o$**:** Output prefix for all output files.
- $-n$ 56**:** Number of threads.

- $-t\ s$: Transform type (rigid + affine + deformable Syn, 3 stages).

```
antsRegistrationSyN.sh -d 3 -f $DATA/mr.nii.gz -m $DATA/mt.nii.gz -x
    $DATA/mask_mr.nii.gz -o $PWD/ -n 56 -t s
mv Warped.nii.gz $DATA/mt_deformed_DANT.nii.gz
```

Baseline Experimental Setup

1. Libraries and versions:
    - PETSc/3.14.2 (configured with:`-with-cuda=1`,`-use-gpu-aware-mpi`)
    - Open MPI/4.0.0
    - NIfTI/2.2.0
    - CUDA/11.8
    - zlib/1.2.13
    - GCC/10
    - CMake/3.18.1
    - Python/3.7
    - SciPy/1.8
    - NiBabel/4.0.0
2. GitHub repositories for applications, tools, and dataset URL:
    - CLAIRE: https://github.com/andreasmang/claire
    - ANTS: https://github.com/ANTsX/ANTs
    - CLAIRE-ROP: https://github.com/UniHD-CEG/CLAIRE-ROP
    - TotalSegmentator(V1): https://github.com/wasserth/TotalSegmentator
    - SimpleITK: https://github.com/SimpleITK/SimpleITK
    - NVTX: https://github.com/NVIDIA/NVTX
    - Dataset: http://dir-lab.com

Evaluation

1. Visualization: Our primary focus is on visualizing Case 6 due to its significant deformations, which provide the most pronounced insights into the algorithm's behavior and effectiveness.
2. Accuracy and precision of timings: To ensure reliable performance metrics, we execute each dataset multiple times. No outliers were observed in the recorded data.
3. Image roles: In our analysis, $T00$ is used as the reference image and $T50$ as the moving image. We also experiment with reversing these roles to evaluate the impact on results and performance. Similar results were achieved in both configurations.