Aus dem Lehrstuhl für Computerunterstützte Klinische Medizin

der Medizinischen Fakultät Mannheim

(Direktor: Prof. Dr. rer. nat. Lothar R. Schad)

Automatic interventional Imaging for the Parametrisation of Vascular Structures

Inauguraldissertation
zur Erlangung des Doctor scientiarum humanarum (Dr. sc. hum.)
der
Medizinischen Fakultät Mannheim
der Ruprecht-Karls-Universität zu Heidelberg

vorgelegt von

Christian Tönnes, M.Sc.

aus

Rüsselsheim

2023

Dekan: Prof. Dr. med. Sergij Goerdt
Referent: Prof. Dr. rer. nat. Frank G. Zöllner

Improvements for Medical Imaging

Robotic computer tomography systems offer a wide range possibilities for non-circular and patient specific trajectories. These could reduce metal artifacts, improve the image quality or increase the available space for the practitioners. But every trajectory needs to be calibrated to reconstruct the CT image. There are two main approaches, an offline calibration uses a special phantom that has to be imaged before the patient and the online calibration where the trajectory is calibrated with the images acquired from using a prior CT image.

In chapter 5, a method for calibrating arbitrary Cone-Beam Computed Tomography (CBCT) trajectories (FORCASTER) is developed. It detects and matches feature points in simulated projections and acquired projections. These paired feature points are then used to correct the projection parameters by directly calculating the correction factor or with a minimization algorithm, which is also presented in this part. To simulate projections a prior image of the object is needed, but it is also shown that this calibration functions if the prior image has some differences compared to the object that is currently imaged. The FORCASTER algorithm achieves similar calibration results when compared to state-of-the-art algorithms.

The following chapter 6 introduces FORCAST-EST, an extension for the FORCASTER algorithm, which estimates the starting parameters for the calibration. This allows the calibration of trajectories that come without positional and rotational data for the individual projections. It simulates projections in a grid and then compares every acquired projection to the grid, first in a coarse search and then in the surrounding of the best matches. The estimates are sufficient to calibrate the trajectory with a similar accuracy as FORCASTER using the regular starting parameters recorded by the CBCT system.

In chapter 3 of this thesis, an algorithm for detecting arteries in dynamic contrast enhanced MRI images is presented. Through thresholding, flood fill and morphological operations, a mask for the artery at the time point where the initial wave of contrast agent arrives is created. The process is deterministic and independent of the user experience. Since the annotation of the artery has a strong impact on the calculated perfusion values, helps this algorithm with reproducibility and comparability of perfusion imaging.

The chapter 4 presents a web-based MRI image generator for teaching students. It allows the students to choose different sequences and set the relevant parameters, and they can then see how it changes the resulting MRI image. The software runs completely inside the browser and is open source[1].

The presented work show algorithms that help in the imaging process by improving reproducability and comparability or in reconstructing CBCT volumes.

[1] https://github.com/ChristianToennes/VirtMRI

Verbesserungen für die medizinische Bildgebung

Robotische Computertomographen bieten eine Vielzahl von Möglichkeiten für nicht-zirkuläre und patientenspezifische Trajektorien. Diese können Metallartefakte reduzieren, die Bildqualität verbessern oder den im Interventionsraum zur Verfügung stehenden Platz vergrößern. Aber jede Trajektorie muss kalibriert werden bevor das CT-Bild rekonstruiert werden kann. Es gibt zwei Hauptansätze: die Offline-Kalibrierung, bei der ein spezielles Phantom verwendet wird, und die Online-Kalibrierung, in der die Trajektorie mit einem vorherigen CT Bild des Patienten kalibriert wird.

Im Kapitel 5 wird eine Kalibrierung für beliebige CBCT-Trajektorien entwickelt. Er erkennt und gleicht Merkmalspunkte in simulierten Projektionen und erworbenen Projektionen ab, um die Differenz zu minimieren, die dann die korrekten Parameter für die Projektion ergibt. Dies erfordert ein vorheriges Bild des Objekts, aber es wird auch gezeigt, dass die Kalibrierung auch dann funktioniert, wenn das vorherige Bild mehrere Unterschiede im Vergleich zum Objekt aufweist, wenn es mit der zu kalibrierenden Trajektorie abgebildet wurde.

Im darauffolgenden Kapitel 6 wird eine Erweiterung des FORCASTER-Algorithmus vorgestellt, die die Startparameter für die Kalibrierung schätzt. Dies ermöglicht die Kalibrierung von Trajektorien, die ohne Positions- und Rotationsdaten für die einzelnen Projektionen vorliegen. Der Algorithmus simuliert Projektionen in einem Gitter und vergleicht dann jede aufgenommene Projektion mittels markanten Punkten mit diesem Gitter, erst einer groben Schrittweite, dann in den Umgebung zu den besten Funden. Diese Schätzungen reichen aus, um die Trajektorie mit einer ähnlichen Genauigkeit zu kalibrieren wie FORCASTER mit den regulären Startparametern, die vom CBCT-System aufgezeichnet werden.

Kapitel 3 dieser Arbeit stellt ein Algorithmus zur Erkennung von Arterien in dynamischen kontrastverstärkten MRT-Bildern vor. Durch Schwellenwertbildung, Flutung und morphologische Operationen wird eine Maske für die Arterie erstellt. Der Prozess ist deterministisch und unabhängig von der Erfahrung des Anwenders. Da die Annotierung der Arterie einen starken Einfluss auf die berechneten Perfusionswerte hat, hilft dieser Algorithmus der Reproduktion und der Vergleichbarkeit von Perfusionsbildgebung.

In Kapitel 4 wird ein webbasiertes MRT für die Ausbildung von Studenten vorgestellt. Die Student*innen können verschiedene Sequenzen auswählen und die entsprechenden Parameter einstellen, um dann festzustellen wie sich das MRT-Bild verändert. Die Software läuft vollständig innerhalb des Browsers und ist quelloffen[2].

Die vorgestellte Arbeit zeigt Algorithmen, die bei der Bildgebung helfen, indem sie die Reproduzierbarkeit und Vergleichbarkeit oder die Rekonstruktion von CBCT-Bildern verbessern.

[2] https://github.com/ChristianToennes/VirtMRI

Preface

This is a cumulative thesis consisting of four journal publications.

Chapter 3: C. Tönnes, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. Schad and F. Zöllner. Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer. Magn. Reson. Imaging, 75, pp.116-123 (2021), doi: 10.1016/j.mri.2020.09.009

For this paper I developed, implemented and evaluated the Algorithm. The data annotation was annotated by Dr. Janssen and me in cooperation. The manuscript was written by me, reviewed and corrected with the help of the other authors.

Chapter 4: C. Tönnes, C. Licht, L. Schad and F. G. Zöllner. VirtMRI: A tool for teaching MRI. J. Med. Syst. Submitted

For this paper I developed, implemented, designed the complete system and wrote the manuscript. The manuscript was reviewed and corrected with the help of the other authors.

Chapter 5: C. Tönnes, T. Russ, L. Schad and F. Zöllner. Feature-based CBCT Self-Calibration for Arbitrary Trajectories. Int J CARS, 2022, 17 (11), pp.2151-2159. doi: 10.1007/s11548-022-02645-9

For this paper I developed, implemented and evaluated the algorithm; acquired and preprocessed the data; wrote the manuscript. The manuscript was reviewed and corrected with the help of the other authors.

Chapter 6: C. Tönnes and F. G. Zöllner, Feature-Oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST. Appl. Sci. 2023, 13, 9179. https://doi.org/10.3390/app13169179

For this paper I developed, implemented and evaluated the algorith; acquired and preprocessed the data; wrote the manuscript. The manuscript was reviewed and corrected with the help of the other authors.

Darstellung der Eigenleistung der Doktorandin/des Doktoranden

bei kumulativen Dissertationen

Name der Doktorandin/des Doktoranden Christian Tönnes

Titel der Dissertation
Automatic interventional Imaging for the Parametrisation of Vascular Structures

Betreut durch Frank Zöllner

☒ Ich möchte eine kumulative Dissertation einreichen und bitte den Promotionsausschuss zu prüfen, ob die vorgeschlagenen Publikationen quantitativ und qualitativ ausreichen, um die Anforderungen an eine kumulative Dissertation zu erfüllen.

☐ Der Promotionsausschuss hat zuvor geprüft, ob meine Publikationen für eine kumulative Dissertation geeignet sind, und dies ist eine abschließende Übersicht über die in meiner kumulativen Dissertation enthaltenen Publikationen.

1. Liste der peer-reviewed Publikationen, die in die kumulative Dissertation aufgenommen werden. Geben Sie für jede Publikation eine vollständige Liste der Autoren, den Titel, die Zeitschrift, den Impact Factor der Zeitschrift an und ob das Manuskript zur Veröffentlichung angenommen wurde, sich nach der Begutachtung in Überarbeitung befindet oder eingereicht wurde und zur Begutachtung ansteht. Geteilte Erstautorenschaften sollten deutlich angegeben werden. Bitte geben Sie auch an, ob es sich bei der Publikation um einen Original-Forschungsbericht, einen Review oder eine andere Art von Artikel handelt.

Publikation 1: C. Tönnes, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. Schad and F. Zöllner. Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer. Magn. Reson. Imaging, 75, pp.116-123 (2021), doi: 10.1016/j.mri.2020.09.009
Original-Forschungsbericht, Published, Impact Factor: 3.13

Publikation 2: C. Tönnes, T. Russ, L. Schad and F. Zöllner. Feature-based CBCT Self-Calibration for Arbitrary Trajectories. Int J CARS, 2022, 17 (11), pp.2151-2159. doi: 10.1007/s11548-022-02645-9
Original-Forschungsbericht, Published, Impact Factor: 3.421

Publikation 3: C. Tönnes and F. Zöllner. Feature-oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST. Appl. Sci., 2023, https://doi.org/10.3390/app13169179
Original-Forschungsbericht, Published, Impact Factor: 2.7

2. Zusammenfassung des Beitrags der Doktorandin/des Doktoranden zu der in jedem Manuskript berichteten Arbeit

Arbeitsschritte	Publikation 1	Publikation 2	Publikation 3
Konzeption (%)	80	100	100
Literaturrecherche (%)	100	100	100
Ethikantrag (%)	0	-	-
Tierversuchsantrag (%)	-	-	-
Datenerhebung (%)	50	100	100
Datenauswertung (%)	90	100	100
Ergebnisinterpretation (%)	100	100	100
Verfassen des Manuskripttextes (%)	90	95	95
Revision (%)	100	100	100
Geben Sie an, welche Abbildungen/ Tabellen aus Ihrer Doktorarbeit entstanden sind.			
Geben Sie im Einzelnen an, welche Daten/Zahlen/Tabellen auf Forschungsergebnissen von anderen beruhen.			

3. Die Mindestanzahl der Publikationen, die für eine publikationsbasierte kumulative Dissertation erforderlich sind, ist in den "Ausführungsbestimmungen zu publikationsbasierten Dissertationen" festgelegt. Im Falle einer gemeinsamen Erstautorenschaft oder einer Letztautorenschaft begründen Sie bitte unten, warum die Veröffentlichung einer einzelnen Erstautorenschaft gleichgestellt werden soll.

4. Ich bestätige hiermit, dass dies eine wahrheitsgetreue Darstellung des Beitrags der Doktorandin/des Doktoranden zu den aufgeführten Publikationen ist.

Unterschrift der Doktorandin bzw. des Doktoranden

Unterschrift der Betreuerin bzw. des Betreuers

Contents

List of Figures

List of Tables

List of Listings

List of Acronyms

CBCT Cone-Beam Computed Tomography. v, vi, 1–3, 12, 82, 84

CT Computed Tomography. 2, 3, 86

DCE-MRI Dynamic Contrast Enhanced MRI. 82, 86

FORCAST Feature ORiented Calibration of Arbitrary Scan Trajectories. 82

FORCAST-EST FORCAST-parameter ESTimator. 82

FORCASTER Feature ORiented Calibration of Arbitrary Scan Trajectories with Enhanced Reliability. 82

HU Hounsfield Unit. 12

MRI Magnetic Resonance Imaging. 2, 3, 86

1 Introduction and Outline

1.1 Motivation

Imaging technology offers many benefits to medical interventions, but also several challenges. Two of these challenges are metal artifacts from instruments or implants that are introduced during the procedure and the limited available space, so every device has to contest for it. The Cone-Beam Computed Tomography (CBCT) is uniquely suited for these, it occupies just a little space and due to its high flexibility in position and motion it can acquire images with reduced metal artifacts.

A CBCT system consist of an X-Ray source and a digital detector, commonly fixed to a C-Arm. The C-Arm can be moved and rotated around the table with a high degree of freedom, which allows an easy acquisition of X-Rays from different directions. Also, these systems can acquire X-Ray images continuously as a fluoroscopy, e.g. useful for heart catheter interventions where the flow of contrast agent through the coronary arteries is observed. The third image type CBCT can acquire are 3D images. Here, the C-Arm is rotated in a circle around the patient while many X-Ray images are acquired. Then the 3D image can be reconstructed from these images. Since these systems are often used in interventional settings the images often contain metal objects, while not a problem for 2D images in the reconstructed 3D image the metal creates artifacts. There are different strategies to remove metal artifacts, one is to adapt the trajectory on which the images are acquired. The CBCT is due to its flexible movement, uniquely suited for this. The new trajectories are called task-based, free, or arbitrary trajectories. The CBCT allows not only to simply tilt the standard circular trajectory it can follow any path, as long as the surrounding has enough space for it, even acquiring projections at totally random spots. Another use-case for arbitrary trajectories is the reduction of radiation. If only one slice is relevant for the task, then a trajectory that only creates a sharp image in the required slice but nowhere else (a tomosynthesis), can achieve the task with reduced radiation.

These techniques aren't used in current systems, there is a problem with adapting trajectories to the current task. To reconstruct an image from the projections acquired with a trajectory, this trajectory has to be calibrated. The robotic systems that control the C-Arm are unable to accurately follow the desired path and therefore the reconstructed image is incorrect. But the deviance of the system is always the same, so the actual trajectory of the CBCT is measured and then used for reconstruction. This is called calibration and can be done in multiple ways. Image a calibration phantom with well known properties where for every projection the actual acquisition position and orientation

can be calculated from the position of the visible objects. Another way is to use a prior image of the patient and then find the parameters by comparing the acquired projection image to simulated projections. The former one is called offline calibration, the latter online calibration. For offline calibration you have to calibrate the trajectory before you want to use it. Whereas, online calibration allows calibrating the trajectory with just the images that should be used to reconstruct the image and a previously acquired Computed Tomography (CT) image. Therefore, online registration offers the opportunity to create CBCT trajectories on the fly, then calibrate and reconstruct the 3D volume without removing the patient, but we are not there, yet. Online calibration needs a lot of time to find the parameters and it is not feasible to wait for an hour until the image is calibrated and reconstructed. Here, research is needed to reduce the runtime, but also to increase the accuracy and robustness.

Acquiring images so a physician can look at them is a tremendous help for medicine, but it is not enough for evidence based medicine. We want images, that can be quantitatively compared to images from other devices, hospitals or times. This is especially a problem for Magnetic Resonance Imaging (MRI) images where the measured values aren't even reproducible on the same machine with the same patient. It gets worse if a physician has to annotate structures inside a noisy and low resolution Magnetic Resonance Imaging (MRI) image, which are then used to calculate quantitative parameters, for example the perfusion. Different physicians will annotate the structures differently and then the calculated parameters also change. Algorithms offer the chance to reproducible annotate structures. Every time a deterministic algorithm processes the same image, it will create the same annotation. So when recalculating, the parameters will stay the same, even if the original annotations were not saved or lost. An algorithmic annotation also makes results of different physicians more comparable, since the experience of the physician is no longer relevant for the quality of the annotation. Thus, reproducibility and quantitative imaging goes hand in hand with algorithm development.

MRI systems allows the use of many different sequences with adjustable parameters, that can generate a lot of different image contrasts. But it is not possible to try different options on a real scanner, because the acquisition requires too much time. Here, simulation or virtualization of the imaging process can help. Allowing physicians to test different parameters or sequences beforehand and find the one with the image contrast they want.

In this thesis, I will present two algorithms for calibrating CBCT trajectories and one that segments arteries to be used in the parametrization of perfusion calculation. Furthermore, I present a side project that can help with teaching MRI to students.

1.2 Outline

This thesis is written cumulatively. The chapter 3 introduces an algorithm for finding the arteries in contrast enhanced MRI images. In 5 and 6 two algorithms for calibrating CBCT trajectories are presented. The chapter 4 is a little detour about a web-based simulator for teaching MRI sequences to students. All of these four mentioned chapters consist of published and peer-reviewed scientific papers.

The theoretical background for the above described chapters can be found in 2. It gives an overview of the basics for MRI and CT imaging, then goes further into calibration of CT trajectories. Feature finding and matching, used by the algorithms in 5 and 6, is explained and the relevant algorithm shortly described.

In chapter 3 an algorithm for finding the arteries in dynamic contrast enhanced MRI images is described. From the found arteries, the arterial input function can be extracted to be then used in perfusion calculations. To segment the arteries, the algorithm first finds the time point where the contrast agent appears and then uses classical algorithmic filters to segment the pixels belonging to the artery.

A side project is the MRI simulator shown in chapter 4, a web-based simulator which can be used in the common modern browsers. The purpose is to teach students how different MRI sequences and parameters influence the resulting images. It is completely browser based and calculates everything within the browser, the server only provides the static files and data sets needed for the simulation.

Chapter 5 is about an algorithm to calibrate CBCT trajectories. It is based on the FORCAST[3] algorithm and improves it in several parts. To calibrate a trajectory, each individual projection is separately registered to a prior CT image. For this, feature points are detected and then matched to feature points from forward projections of the prior image. Several metrics are minimized to find the correct position of the projection.

A further improvement of the FORCASTER algorithm is shown in chapter 6, where the need for an initialization is removed. The initial parameters needed are estimated by comparing the projection with a coarse grid of projections around the prior image, and then refined until the estimates are close enough until a calibration algorithm like the one described in chapter 5 can successfully calibrate the trajectory.

A summary of this thesis and the four papers can be found in chapter 7.

The outlook in chapter 8 contain future plans and development options for the presented algorithms and software.

1.3 Citation of Previous Publications

Several chapters of this thesis have already been published or are currently submitted for publication. The citations for these chapters are:

Chapter 3: C. Tönnes, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. Schad and F. Zöllner. Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer. Magn. Reson. Imaging, 75, pp.116-123 (2021), doi: 10.1016/j.mri.2020.09.009

Chapter 4: C. Tönnes, C. Licht, L. Schad and F. G. Zöllner. VirtMRI: A tool for teaching MRI. J. Med. Syst. Submitted

Chapter 5: C. Tönnes, T. Russ, L. Schad and F. Zöllner. Feature-based CBCT Self-Calibration for Arbitrary Trajectories. Int J CARS, 2022, 17 (11), pp.2151-2159. doi: 10.1007/s11548-022-02645-9

Chapter 6: C. Tönnes and F. G. Zöllner, Feature-Oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST. Appl. Sci. 2023, 13, 9179. https://doi.org/10.3390/app13169179

2 Theoretical Background

2.1 Medical Imaging

Medical Imaging encompasses several technologies, in this thesis only magnetic resonance imaging and computed tomography are used. The following section explains shortly how MRI and CT function. It is followed by an introduction to detecting feature points in images and

2.1.1 Magnetic Resonance Imaging

MRI is based upon the behavior of spins inside a magnetic field. To simplify it until it is technically wrong but understandable for humans, a spin is a tiny compass. A standard MR image measures the spins of single protons, so the spin of hydrogen atoms.[4] It is often shortened to 1H MRI or proton MRI and this spin has two possible states, $\pm 1/2$. In theory all Atoms with spin states unequal to zero could be used for imaging, but for biological tissue 1H is the most common due to its abundance in water and fat. Another interesting atom is sodium imaging, or 23Na imaging. It has a spin number of 3/2 and therefore can have the spin states of $\pm 3/2$, ± 1, and $\pm 1/2$. This allows not only to create image that show how much sodium is present, but also if this sodium is free or bound to a molecule, which can give information about the cell functions.

Regardless of the measured atom cores, the signal acquisition follows the same principle. When a magnetic field B_0 is present, the spins tend to orient themselves parallel to it, or antiparallel. The parallel orientation has a slightly lower energy and is therefore preferred. This energy difference and the ratio of parallel to antiparallel is proportional to the magnetic field. Because of quantum mechanical shenanigans, the spins don't actually decide on one orientation, since we are unable to measure the signal of a single spin. Instead, we measure the sum of all spins inside one voxel, which create a magnetization M_0, that is parallel and proportional to the magnetic field. This magnetization alone is not enough to create images, but fortunately this netto magnetization vector is also rotating around the magnetic field direction. If M_0 is oriented along the magnetic field, we cannot measure it, but we can push it out of its orientation with a high-frequency radio signal. Then the magnetization vector will rotate around the magnetic field, and therefore it induces a currency in a correctly oriented coil. The rotation is called precession and can also be observed if you push a spinning top out of its resting position. The behavior of Spins is comparable to such a spinning top, and it will slowly return to its resting position, parallel to the magnetic field. At this point it is important to

mention, that an atom core having a spin does not mean this core is spinning, it just behaves as if it was spinning.

The high-frequency radio signal needs to have the same frequency as the precession of the spins that should be rotated. This precession frequency, called the Larmor frequency, depends on the magnetic field B_0 and the gyromagnetic ratio γ, see also equation 2.1. All spins with a different Larmor frequency are not affected by the radio signal, and thus only the rotated spins generate a measurable signal.

$$\omega_0 = -\gamma B_0, \tag{2.1}$$

To describe the magnetization further, a few definitions are needed. In the following, we will only consider the magnetization vector of a single voxel. Then we will divide it into two parts, M_z describes the magnetization along the magnetic field or longitudinal magnetization and M_{xy} is the magnetization perpendicular to it, the lateral magnetization. Because the spins are constantly precessing around the magnetic field, a further subdivision into M_x and M_y will only complicate the equations but won't enhance the understanding of the principles.

Now, the equation 2.2 describes how the magnetization of our spins return to equilibrium after we have rotated them by 90°. The constant T_1 is unique to every tissue and substance and is 63% $(1 - \frac{1}{e})$ of the time needed for the magnetization to turn back, parallel to the magnetic field. This return of magnetization to the equilibrium $M_{z,eq}$ is called T1 relaxation or spin-lattice relaxation.

$$M_z(t) = M_{z,eq} - (M_{z,eq} - M_z(0))\, e^{-\frac{t}{T_1}}, \tag{2.2}$$

A second kind of relaxation happens to the magnetization in x, y direction M_{xy}, which is perpendicular to the magnetic field and to the magnetic component M_z. This relaxation is described by equation 2.3 and is called T2 relaxation or spin-spin relaxation. The name already indicates it, this relaxation is caused by interaction between spins. Neighboring spins interact with each other and get out of phase. This dephasing of individual spins leads to a reduction of the total magnetization vector in the transversal plane M_{xy}. It's important to note, that this reduction in magnetization is not equal to the increase in the longitudinal magnetization M_z. The reduction is due to spins getting out of sync, while the other is due to spins realigning with the magnetic field.

$$M_{xy}(t) = M_{xy}(0)\, e^{-\frac{t}{T_2}}, \tag{2.3}$$

The T_1 and T_2 relaxation are both dependent on the tissue and therefore very interesting for medical applications. There is a third relaxation which is independent of the tissue, the T_2^* relaxation, or free induction decay (FID). In an ideal MRI device the magnetic field has the same strength everywhere,

but this is not true in reality with imperfect coils, a patient in our scanner, different temperatures, materials with magnetic properties, etc..., therefore differs the Larmor frequency slightly for each spin. Since our HF radio pulse is also imperfect and has a certain, adjustable, bandwidth we can turn all these spins, even though they have different Larmor frequencies, but in the transversal plane they will rotate with different speeds. A good representation of this is a race between Achilles and a tortoise. The Achilles is faster and will run around the stadium faster than the tortoise, and the combined signal will be reduced and finally vanish completely. For a mathematical explanation of the relaxation see equation 2.4.

$$M_{xy}(t) = M_{xy}(0)\, e^{-\frac{t}{T_2^*}} \tag{2.4}$$

This T_2^* leads directly to the first and most basic MR imaging technique, called Spin Echo[5]. Since this relaxation depends only on approximately static things, like the temperature, objects and manufactured imperfections in the MRI device which do not change much over time, we can reverse those. Imagine the race between the Achilles and the tortoise, after a certain time both will turn back and run the same distance towards the starting point, they each run back with the same speed they had when running forward, so they reach the start at the same time. For our spins that means, if we manage to turn them by 180° they will run backwards until they are at the starting point, at which they all had the same phase and we can measure the full M_{xy} magnetization. So we have to first rotate the spins by 90° into the transversal plane, then we wait some time $\frac{TE}{2}$ and rotate everything by 180°. A specific order of rotations, activations of gradient magnetic fields, waiting times and acquisition of the signal is called a Sequence, in our case it is the Spin Echo Sequence. The spins will take the same time $\frac{TE}{2}$ to reach a synchronized phase again. After the time TE, called the Echo Time, we will have an echo of the original signal at $t = 0$. Depending on the time we wait, this echo signal will be reduced based on the T_2 relaxation and equation 2.3. Therefore, the Spin Echo Sequence can be used to measure T_2 weighted images. Depending on the echo time, TE the contrasts between different tissues will change. Another parameter for the Spin Echo Sequence is the repetition time TR, the time to wait until the next 90° degree rotation is made, and the sequence starts over. This time should be larger than the T_1 time, otherwise the 90° rotation will not rotate the maximum magnetization $M_{z,eq}$, but only the part $M_z(TR)$ that has relaxed. For this more complex case, see equation 2.5.

$$M_{xy}(t) = (1 - \ e^{-\frac{TR}{T_1}})\, e^{-\frac{t}{T_2^*}} \tag{2.5}$$

Apart from images that use the T_2 sequence for tissue contrast, there is the Inversion Recovery Sequence to create image contrasts based on the T_1 relaxation times. Here we add a 180° rotation before the Spin Echo Sequence. The time before the first inversion and the 90° rotation is the inversion time

TI. To reduce the impact of T_2 relaxation, the TE time should be very short. The mathematical relation can be seen in equation 2.6.

$$M_{xy} = (1 - 2\, e^{-\frac{TI}{T_1}} + e^{-\frac{TR}{TI}})\, e^{-\frac{TE}{T_2}} \tag{2.6}$$

These two very basic sequences are not everything possible with MRI devices. Another way to refocus the spins after their inevitable discord is using the gradient magnetic fields. This sequence, called Gradient Recalled Echo, does not need a 180° radio pulse, instead it inverts one of the gradient magnets.

$$M_{xy} = M_0\, e^{-\frac{TE}{T_2^*}} \frac{sin\theta \left(1 - e^{-\frac{TR}{T_1}}\right)}{1 - cos\theta\, e^{-\frac{TR}{T_1}}} \tag{2.7}$$

Gradient Magnetic Fields haven't been covered so far. They have multiple purposes, one is their use in refocusing spins, as seen previously, another use is for 3D imaging. As stated previously, we measure the sum of all spins in our object. This is not desired, the sum of all spins inside a body gives little information usable for diagnosis concerning specific regions. A way to differentiate the measured signals into all three space dimensions is needed. Previously the relation of the Larmor frequency to the strength of the magnetic field was described, this is elemental for the next part. Also elemental is the addition of three distinct magnetic fields B_x, B_y, B_z, one for each dimension and all perpendicular to each other. To get three different information for every signal, we have to encode these three dimensions into our signal. All encodings will use the gradient fields and the Larmor frequency.

To show this encoding the simplest sequence, Spin Echo, will be used. First, we will select one slice out of the volume, only this slice will be rotated by the first 90° HF signal. Therefore, only spins in this slice will be part of the measured signal. This reduces our problem to a two-dimensional one.

For the next two encodings we have to use Fourier transformations, one is implicit, the other explicit. Starting with the easy one, the explicit Fourier transformation or frequency encoding. When measuring a signal we acquire it over time, this allows us to decompose it into individual frequencies using the Fourier transformation. These frequencies correspond to different areas where the Larmor frequency differs. Using the gradient magnetic fields, we can add a magnetic field strength gradient to force a certain difference in frequency along a defined vector. In this way, we can resolve our signal along one dimension.

What remains is encoding the final dimension. The Fourier transformation returns a complex signal, the amplitude for each frequency and the phase. The frequency encoding uses the amplitude, so for the last dimension we have to use the phase information. The phase encoding happens between the 180° HF pulse and the acquisition, we create a magnetic field, perpendicular to the one used for frequency encoding and to the one used for slice selection, to change the Larmor frequencies of the spins. Before the signal is measured, this gradient field is turned off. It results in the spins along this direction having a

specific phase. Since our gradient field is linear, which leads to a linear increase in the precession frequency, the resulting phase shift between spins can be described with a sinus function. Changes in contrast along this direction with the same frequency as the frequency of the phase shifts is enhanced, and all other frequencies are suppressed. Repeating the measurement with different phase encoding frequencies allows us to differentiate signals along this axis.

The measurements are recorded in the so called K-Space. In the center of this space are the low frequencies, and towards the edges the frequencies raise. The direction of all frequencies is from their point in the K-Space towards the center point. For our example, the K-Space consists of a stack of two-dimensional planes. Every plane consists of lines that are measured at once and then decomposed using the Fourier transformation. The phase encoding gives the second dimension for our measured lines. No phase encoding gradient equals to no difference in phases, so we save our recorded line through the center. With increasing phase encoding gradients, the phases diverge more and the distance to the center increases, since the frequency to describe the difference in phases increases. This is also the reason why this phase encoding is an indirect Fourier transformation. We don't perform the actual transformation, but use physical processes to extract the individual frequencies.

After measuring the K-Space with our three-dimensional encoding, slice selection, phase encoding, and frequency encoding, we perform an inverse Fourier transformation to calculate the actual MRI image. If the K-Space is acquired in the way we described above, we use a two-dimensional inverse Fourier transformation. There are also more advanced MRI sequences that acquire the K-Space fully in three dimensions, then a three-dimensional Fourier transformation is needed.

One of the mentioned sequences and all the three-dimensional localization have to be combined to create a three-dimensional MRI image. There are also a multitude of other sequences that can be combined with the described dimensional localizations, and also a plethora of truly three-dimensional sequences that mix localization and signal generation.

2.1.2 Computed Tomography

Computed tomography generates three-dimensional images based upon the absorption of X-rays. X-Ray images of an object are acquired from many different angles, and from these the three-dimensional image is reconstructed. A simplified version of this process is described in the following paragraph.

The X-Rays are generated inside a Coolidge tube also known as Hot Cathode Tube ([6]). The cathode is heated with the filament voltage and electrons exit the filament to form a cloud around the cathode. From the cathode these electrons get accelerated towards the anode by an acceleration voltage. When an electron hits the anode, it gets slowed down and the kinetic energy is released as a photon. This radiation is called Bremsstrahlung and the energy of these photons is a spectrum that is limited by the applied acceleration voltage.

An electron accelerated with the current U has an energy of U electron volts eV (Equation 2.8). Electron volts can be converted to the joule, the SI unit for energy, but electron volts is more convenient.

$$E = e \cdot U, \tag{2.8}$$

Additionally, the electrons will excite the atoms of the anode which will create photons with very specific energies, called characteristic radiation, when returning to an unexcited state. The combination of these two radiations is the X-ray radiation that will exit the tube ([6]). At the exit of the tube is a metal plate to filter out the photons with a low energy, these can't penetrate the human body and would therefore contribute nothing to the image but still damage the tissue.

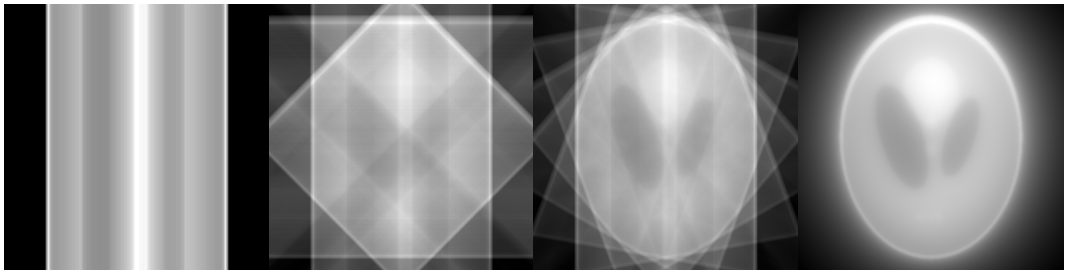
When photons traverse a material, a part will be exponentially absorbed. This depends on the material and the energy of the photons and is described by the absorption coefficient μ. As a general rule, the absorption coefficient is higher for denser materials and materials with a higher atomic number and lower for photons with a higher energy. The absorption also depends on the depth of the traversed material d. When traveling through multiple materials, the x-ray beam will be absorbed by the integral of all the individual absorption coefficients. The Beer-Lambert law (Equation 2.9) describes how the initial intensity of the x-ray radiation I_0 is reduced. This equation is simplified, since the absorption coefficient also depends on the energy of the photons, but for computed tomography this is ignored. The newer photon counting CTs have a limited ability to measure the energy of photons and the more complex equation is used. A standard CT cannot measure the energy of photons, and therefore it is only possible to calculate an average absorption coefficient for all energies.

$$I = I_0 e^{-\int \mu(x,y) \mathrm{d}s} \tag{2.9}$$

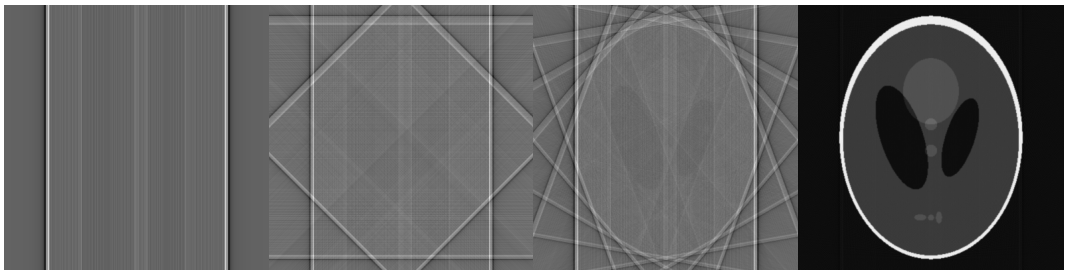
The measured signal is I in equation 2.9, to get the absorption along one x-ray beam we have to divide by the initial radiation intensity I_0 and calculate the negative natural logarithm (equation 2.10).

$$p = -\ln\left(\frac{I}{I_0}\right) = \int \mu(x,y)\, ds \tag{2.10}$$

With this equation, we now have the absorption along all our measured angles. With these we could create a bunch of equations in which we sum up the values of each voxel a ray travels through, weighted by the length of the ray inside the voxel, and equal it with the measured value. If we have enough measurements, we could solve this set of equations to get an CT image. A



(a) Unfiltered back projections.



(b) Filtered back projections using a ramp filter.

Figure 2.1: CT reconstruction using 1, 4, 10 and 600 projections.

normal CT volume has $512 * 512$ voxel per slice, so we would need to measure our object from 262144 different angles, which is unfeasible, it would take too long and create too much radiation damage. Instead, the back projection algorithm is used. It is the inverse operation to taking an x-ray image, which is a forward projection since it goes forward through the object and integrates all values. The back projection takes the value p and smears it equally across the volume in the direction of the x-ray beam. This is repeated for all projections and will result in a reconstructed image of the object. The process is shown in figure 2.1 (a). The more projections from different directions are used, the better the image, but even when using many projections, the resulting image is blurry. This happens if the measured values are used directly. They have, the problem, that low frequencies are measured in a higher proportion than high frequencies. To remedy this, the low frequencies have to be suppressed, this can be done by applying a ramp filter in the frequency domain. This simplest filter is just a multiplication with the absolute value of the frequency (equation 2.11 divided by the maximum frequency. The maximum frequency we can measure is constrained by Nyquist–Shannon sampling theorem, therefore all frequencies above this maximum are noise and should be discarded.

$$\phi(f) = \begin{cases} \frac{|f|}{f_{max}} & , |f| < f_{max} \\ 0 & , else \end{cases} \tag{2.11}$$

After applying this filter to the projections, the back projection algorithm will create an image with sharp edges and no blurring (Fig. 2.1 b).

In the next step, the attenuation values μ from the reconstructed CT image

are normalized using the attenuation of water μ_{Water} and air μ_{Air} (eq. 2.12), this gives results in Hounsfield Unit (HU). With this normalization, the CT value of air is -1000 and water has a value of 0.

$$HU = \frac{\mu - \mu_{\text{Water}}}{\mu_{\text{Water}} - \mu_{\text{Air}}} \cdot 1000 \tag{2.12}$$

Trajectory Calibration

To get a good reconstruction for every projection, the exact position and orientation is needed. For this, the trajectory the CBCT uses has to be calibrated. This can be done in multiple ways, one is using a calibration phantom. A calibration phantom is precisely crafted where the position of every part is well known, and the parts are easily distinguishable from the background in a projection. After acquiring the projections on each of those the parts of the phantom are marked and because the positions are known the parameters of a projection can be calculated. Of course, the calibration phantom is build so it looks different from different angles. This offline calibration is highly precise, but there are two requirements. First, the used trajectory has to be run twice, once with the phantom and once with the patient. Secondly, the CBCT system needs the ability to perform the trajectory twice with only minimal deviations. The second requirement is easily fulfilled by modern CBCT systems.

Another way to calibrate a trajectory is online calibration. Here a prior image is acquired using an already calibrated trajectory, then the new trajectory is acquired. The projections from the new trajectory are then calibrated using the prior image as a reference. This allows to easily create new trajectories, use them and immediately calibrate and reconstruct an image. But on the other hand, it needs a prior image. Similar to registration tasks, the accuracy depends on how close the prior image is to the actual image and how close the starting parameters are to the actual parameters.

2.1.3 Feature Points

Feature points are significant points in an image, that can be used to describe or understand that image. Widely used algorithms are Speeded-Up Robust Features (SURF)[7] or Scale-Invariant Feature Transform (SIFT)[8]. Both use a multiscale representation of the input image. They create this scale space by convolution with a Gaussian kernel and increasing standard deviation. This creates increasingly blurred images. Then these images are used to detect the significant points of the input image.

In this thesis, feature points are detected using the AKAZE[9] algorithm, which is an accelerated version of KAZE (jap. for "wind")[10]. Similar to SURF and SIFT, it uses a multiscale representation, but instead of a Gaussian filter it uses a nonlinear diffusion function. While the Gaussian blurs noise and edges equally, this nonlinear blurring function preserves strong edges and

Figure 2.2: Multiscale image representation for Gaussian in the top row and the nonlinear diffusion function used by AKAZE in the bottom row. Notice, how the nonlinear diffusion preserves strong edges between the objects.

blurs only within the regions. An example of this difference can be seen in Figure 2.2.

It is clearly visible, that the multiscale representation generated by AKAZE is preserving the high-contrast edges in the image while the low-contrast edges withing regions are removed.

The features are then detected by calculating the Hessian matrix for every point in the image and finding extremas. For every found feature point, AKAZE creates a binary description vector. This vector contains information about the gradients around the point. The algorithms in this thesis use the rotation invariant version of this descriptor.

Since the descriptor is binary, it can be compared using the Hamming distance. This distance counts the number of different bits. A few quick examples: 101 and 111 have a Hamming distance of one; 100 and 011 a Hamming distance of three. For the descriptors of the feature points, a low Hamming distance corresponds to a similar distribution of gradients around the feature point. This means, that a low distance equals a similar distribution of gradients and these points look similar in the images.

To find matching parts in two images, first detect feature points and extract the descriptors. Then every feature point from one is compared to every other feature point from the other image. For the binary descriptor, the comparison is done using the Hamming distance. Using this, there will be a matching feature point for every point, but obviously this means many of these matching pairs are very poor matches.

Therefore, the ratio filter developed by Lowe et al.[8] is commonly used. Here, for every feature point the two points with the lowest Hamming distance are selected and the point with the lower distance is only considered a good match if the distance is less than 0.7 times the distance of the second-best match. The used ratio can be adapted, in this thesis it is between 0.7 and 0.8.

Matching feature points can then be used for registration or object tracking. An example of matching the entrance to the complete tower is shown in figure 2.3. In addition to Lowe's ratio filter, the matches were also filtered by using Random sample consensus (RANSAC). This algorithm divides all found matches into inliers (good matches) and outliers. The door, signs and the big crack in the wall were all successfully matched between the two images. It is very common, that only few feature points will have a match. In this example there are 25 good matches, but the upper image has 1442 feature points and the lower one has 39315.

Since the coordinates of each point is known and matched to a coordinate in the other image, it is easy to get a transformation matrix between these two images.

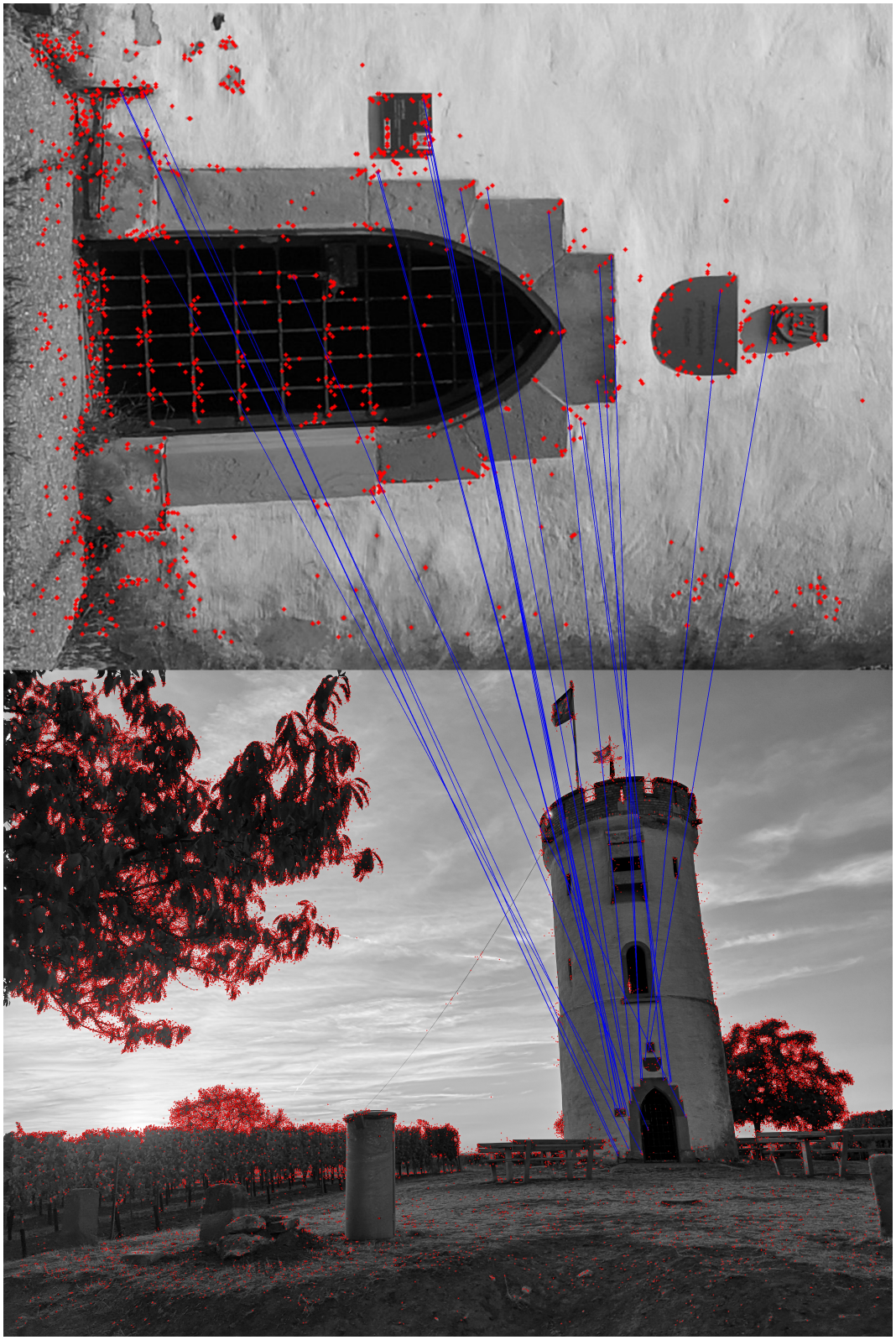


Figure 2.3: Matching a photo of the entrance of the tower to the photo showing the complete tower and surrounding. Detected feature points are shown as red dots, matching points are connected with a blue line.

3 Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer

Christian Tönnes[1], Sonja Janssen[2], Alena-Kathrin Schnurr[1], Tanja Uhrig[1], Khanlian Chung[1], Lothar R. Schad[1], and Frank Gerrit Zöllner[1]

[1]Computer Assisted Clinical Medicine, Mannheim Institute for Intelligent Systems in Medicine, Medical Faculty Mannheim, University Heidelberg, Mannheim, Germany

[2]Department for Clinical Radiology and Nuclear Medicine, Medical Faculty Mannheim, University Heidelberg, Mannheim, Germany

3.1 Abstract

Development of a deterministic algorithm for automated detection of the Arterial Input Function (AIF) in DCE-MRI of colorectal cancer.
Using a filter pipeline to determine the AIF region of interest. Comparison to algorithms from literature with mean squared error and quantitative perfusion parameter K^{trans}.
The AIF found by our algorithm has a lower mean squared error (0.0022 ± 0.0021) in reference to the manual annotation than comparable algorithms. The error of K^{trans} ($21.52 \pm 17.2\%$) is lower than that of other algorithms.
Our algorithm generates reproducible results and thus supports a robust and comparable perfusion analysis.

3.2 Introduction

Quantitative perfusion calculation[11][12] can be used to measure the response of rectal cancer to treatment[13][14][15]. A physician needs to annotate the artery and tumor region in DCE-MRI images to perform a quantification of the perfusion parameters from the DCE-MRI time series data. Manual annotations however, suffer from high inter- and intra-user variability[16][17].

Furthermore, the task of annotating images is time consuming. Due to this, in the current clinical workflow, the arterial input function (AIF) and tumor region of interest (ROI) is only annotated on a single slice. Depending on the slice, which can be chosen freely by the practitioner, the resulting perfusion parameters may have even higher variations. Therefore, there is a need

of reliable, fast and automated AIF determination in quantitative perfusion imaging.

Apart from deep learning approaches to segment [18] the artery several classical algorithms to find the AIF were proposed. Murase et. al. used a half automatic fuzzy c-means clustering[19] to detect the AIF in DCE MRI images of the brain. Parker et al.[20] introduced a semi-automatic algorithm to find the AIF in the lung, prostate and brain region, where the user has to select a slice and then the algorithm find the 5% of voxels with the highest peak value within the first 20 seconds. A semi-automatic algorithm - the authors claim it is fully automatic, but the user has to select a slice - for DCE MRI brain images using k-means clustering was developed by Mouridsen et. al.[21]. K-Means clustering is not deterministic because the initial cluster centers are chosen randomly. Chen et. al. published [22] a pipeline with height, slope and 2D/3D blob filter together with a region growing to select the artery in the breast, brain, liver and prostate region. This algorithm is not fully automatic because the user has to change the parameters to get a optimal result. From Peruzzo et. al. [23] comes an algorithm for the brain, that fits a gamma variate function and then uses a clustering algorithm to iteratively reduce the found voxels to six. Another algorithm for detecting the AIF in the prostate region using the gamma variate function and a clustering algorithm is by Zhu et. al. [24]. An algorithm by Shi et. al. [25], developed on datasets of rat kidneys and human heads, also uses a clustering algorithm, after filtering voxel by the area under the curve and the peak value. Recently Tabbara et. al. published an algorithm to find the AIF in DCE MRI images of the brain. The algorithm uses k-means clustering to find the arteries and then a priority-flooding to segment the brain areas, that are supplied by each found artery[26].

For a reproducible and robust perfusion calculation the algorithm to find the AIF has to be deterministic. A deterministic algorithm cannot depend on user interaction or random initialization. Selecting a slice or using k-means clustering lead to not reproducible results.

In short the criteria for our algorithm. It has to be deterministic and fully automatic. If the algorithm only uses the images as input, if it runs without user interaction or random number generators, it will generate reproducible results. The algorithm should annotate the artery in the full 3D Volume and annotate only the main artery trunk.

In this paper we propose an algorithm that is deterministic and fully independent of the user and user input. Therefore it can generate reproducible results and still reaches human performance.

3.3 Methods

3.3.1 Image Data

We did a retrospective study on 40 patients with colorectal cancer. All images were acquired by physicians during normal clinical assessment of patients with colorectal cancer. The data was partly used in previous papers evaluating quantitative perfusion calculation [27][28].

All patients received a DCE-MRI at 3T-(Magnetom Trio or Magnetom Skyra, Siemens Healthineers, Erlangen, Germany). DCE-MRI was performed using a 3D TWIST sequence with parameters TR/TE/FA = $3.6ms/1.44ms/15$, matrix size = 192×144, FOV = $260 \times 158mm^2$, slice thickness = $3.6mm$, 20 to 32 slices, and parallel imaging with a GRAPPA factor of 2 [27]. Images were either acquired in axial plane or tilted in direction of the coronal plane to cover the tumor as best as possible. Each data set consists of 70 volumes, which were continuously acquired over the time of 5.2 to 7.55 minutes.

The pre-processing consisted of re-sampling all images to the same dimension and then subtracting the first image of every time series from each following image to extract the change of intensity relative to the baseline without tracer agent. The pre-processed images were used as input for all algorithms.

3.3.2 Manual AIF determination

Manual AIF determination was performed using MeVisLab[1] and a self developed annotation workflow. In this workflow the region of interest was delineated in consensus with an experienced radiologist using a graphic tablet. All annotations were done in 3D. It took 2.5 hours to annotate all tumors and also 2.5 hours for all arteries, i. e. on average 5 minutes to annotate a single tumor and 2.5 minutes for a single artery.

3.3.3 Algorithm development

The physical and physiological properties of arteries and blood flow are well known[29]. For our algorithm we used the shape of arteries. In the region of the rectal colon the arteries are slightly curved tubes, parallel to the colon. This leads to near circular representations on the image slices. Another property concerns the concentration of trace agent. Over time the concentration of tracer agent in the artery $c_a(t)$ behaves similar to a gamma variate function. It has a steep increase to a high peak value and a slightly slower fall afterwards. This increase will be the first one in the time series and it also has the highest peak value.

Based on these artery properties multiple steps were implemented to filter the input images. We developed a total of 9 steps which will be describe in detail in the following. These steps are combined to a pipeline which results in

[1] https://www.mevislab.de/

a binary mask for the 3D volume. Every step creates a 4D binary mask and a timestep index, these are given to the next step together with the preprocessed image.

1st Step: Select 1% of the brightest voxel
The first step is to select the voxel with the highest intensity in the first 15 timesteps. The average arrival time step for the contrast agent was 8.5 ± 1.3, therefore a cutoff at the 15th timestep should suffice even for patients with a low cardial output. Because our images are already preprocessed and show only the change of intensity in contrast to the first image, this will discard the background voxels and will only select the artery, tumor and noise or movement artifacts. The bladder is inside the visible region and would be the region with the highest concentration, but the tracer agent will not arrive there within the first 15 timesteps, depending on the sequence that was after 65 to 96 seconds.

2nd Step: Binary Opening
Selected structures may be connected with just a few voxel that only have a high intensity due to partial volume effects. These connections are eliminated by using the morphological operation of a binary opening.

$$Image \circ B = (Image \ominus B) \oplus B \tag{3.1}$$

Our structuring element B is a 1x3x3x3x1 matrix containing a sphere shape $B_{0,x,y,z,0} = 1 | x = 1 \vee y = 1 \vee z = 1$.

3rd: Find timestep with the most peak values.
First the timestep of the peak value for every voxel within the already selected mask is determined. Then all voxels with peak values of 0 or less and those with a timestep, that is not in the first quarter of the time series are discarded. Afterwards select the timestep which contains most of these peak values. From the previous mask only the volume at the found timestep is retained, the rest is replaced with zeros.

$$timestep = \arg\max_t \left(\sum_{x,y,z} \arg\max(image(x,y,z,t'))_{t'} == t \right) \tag{3.2}$$

4th: Blob & Tube Filter
This step gets its mask from the 2nd step, so it essentially runs in parallel to step 3. First we use a connected component analysis to find objects in the mask. We used a connectivity of 18. The found objects are then processed separate from each other. An object is sliced along the longitudinal direction

and the bounding box on every slice is compared to the content.

$$r = \frac{ObjectVoxelCount}{max(BoundingBox_x, BoundingBox_y)^2} \tag{3.3}$$

If a square, defined by the longest side of the bounding box, is filled by less then 60% or more than 80% with this object, then this object is not considered roundish and discarded.

5th Step: Find timestep with most peak values and compare to result from Step 3
Sometimes step 4 throws away enough objects to change the timestep with the most peak values, but the new timestep might not be the correct one. So another timestep is calculated with the results from step 4 and then compared to the one from step 3A. The timestep that is smaller but still above five is chosen. Then the mask from 3B is cropped to this timestep.

6th Step: Fit gamma variate function
The mask from the one timestep found in the previous step is used on all timesteps to extract the intensity values over time for every voxel. A connected component analysis is used to extract found objects form the mask. The intensity values of each object is averaged. This results in one intensity curve for each object. To these intensity curve we then fit a gamma variate function.

$$GVT(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \tag{3.4}$$

The selection criteria for the bounds of α are defined similar to Zhu et. al.[24].

$$\alpha_{lower} = \frac{\log \frac{c(t_{last})}{c(t_{peak})}}{1 + \log \frac{t_{last}-t_0}{t_{peak}-t_0} - \frac{t_{last}-t_0}{t_{peak}-t_0}} \tag{3.5}$$

$$\alpha_{upper} = t_{peak} - t_0 \tag{3.6}$$

All object with a fitted alpha outside these bounds are discarded.

7th Step: Erosion and Dilation
To remove very small objects an morphological erosion is performed, afterwards a dilation will restore the other objects to the previous size.

8th Step: Region Growing
To refine the boundaries of the objects a flood fill region growing is used. For every object the brightest voxel is used as a seed point. The threshold for the flood fill algorithm is determined by the average value of the object minus one standard deviation.
The threshold for the flood fill is set to the average minus one standard devi-

ation of upper 0.8 quantile of the object voxels.

$$\begin{aligned} threshold = avg\,(voxels >= quantile_{0.2}(voxels)) \\ -std\,(voxels >= quantile_{0.2}(voxels)) \end{aligned} \tag{3.7}$$

9th Step: Select two largest objects
As the arteries are two big structures on the left and right halve of the image volume, the last step simply selects the largest object in both halves. These two objects are used as the ROI for the AIF.

Implementation
The algorithm and its steps were implemented in Python 3.5 using the python packages numpy and scipy.

3.3.4 Evaluation

We performed three distinct experiments. First, all combinations, without changing the order, of the substeps were evaluated. Then, we compare our algorithm to algorithms from literature. In the last experiment we compute quantitative perfusion parameters to compare the algorithm for the clinical use case of tumor evaluation.

1st Experiment
These steps were evaluated in every possible combination to determine which step improve the result and which steps have no or a detrimental impact. The combination of 9 steps, without changing the order, results in a total of $2^8 = 512$ combinations of steps.

To compare these versions of the algorithm two metrics were employed. The overlap of the manual ROI and the ROI found by the algorithm is evaluated using the Dice-Sørensen Coefficient[30][31]. The AIF generated from the ROI is compared by calculating the mean squared error with the AIF from the manual annotation.

2nd Experiment
We compared our algorithm to some already published algorithms. Algorithms that were reimplemented are originally from Parker et. al. [20], Chen et. al. [22] and Shi et. al. [25]. In addition to this, we used an erosion and dilation of the manual annotation, furthermore three 2D slices, extracted at the middle, the lower quarter and upper quarter of the volume in the longitudinal direction. "Lower" is the slice of the first quartile and "Upper" is the 3rd quartile. "Middle" is the middle slice and "Middle+-3" are the slices three slices up or down from the middle. In our case every image consists of 20 slices, starting at slice number 1, so these slices are number 5, 7, 10, 13 and 15. Those modifications of the manual annotations were made to better compare the algorithms to the current medical practice of only annotating on a single

slice.

The comparison of these algorithms to our algorithm after optimization and the manual annotation uses the same metrics introduced above in the evaluation of the pipeline steps, namely the Dice-Sørensen-Coefficient and the mean squared error of the AIF curve obtained from the manual annotation and the AIF curve from the algorithm defined ROI.

3rd Experiment
For the third experiment want to compare the automatic AIF detection algorithms with respect to quantitative perfusion calculation. Therefore, the found AIFs were employed to calculate $K^{trans}[1/min]$ using the 2-compartment uptake model (2CUM)[32][11]. The comparison then used the difference between the transfer constant K^{trans} calculated with the AIF algorithm and the K^{trans} calculated with the manual annotation. We have chosen K^{trans} as our comparison parameter, because it can be used to measure the tumor response to therapy[14][13]. The 2CUM model also provides the plasma flow $F_p[ml/min/100ml]$, extraction fraction of tracer agent $E[\%]$, the plasma volume $v_p[ml/100ml]$, and the mean transit time $MTT[min]$ which are also included in the evaluation to reflect the effect of different AIF selections to all model parameters.

3.4 Results

1st Experiment
The evaluation of the 512 combinations of substeps resulted in the combination of six steps: Select brightest 1% (1), Binary Opening (2), Blob and Tube filter (4), Gamma variate Filter (6), Flood fill region growing (8), Select two largest structures (9). Figure 3.1 shows a flow chart of the proposed algorithm DisLAIF (Discover Local AIF) algorithm and a 3D rendering of depicting the results of each of the algorithms processing steps. Briefly, the algorithm selects 1% of the voxels with the highest value, then connecting voxel are removed with a binary opening. The next step filters structures which are not formed like a tube or smaller than a third of the volume height. Afterwards the gamma variate function is fitted to each structure and unsuited ones are discarded. To refine the borders of each structure a flood fill region growing is used. The last step selects the largest structure on the left and right side of the volume, resulting in the AIF.

2nd Experiment
Figure 3.2 shows the Dice coefficients obtained by the different algorithms. Here DisLAIF (0.66 ± 0.09) has a higher dice coefficient than the other algorithms (Shi: 0.37 ± 0.27; Parker: 0.0009 ± 0.0015; Chen: 0.0003 ± 0.0008).

For one example case are the resulting masks of one slice displayed in Figure 3.3. The manual annotation is shown in green, shown in red are the results

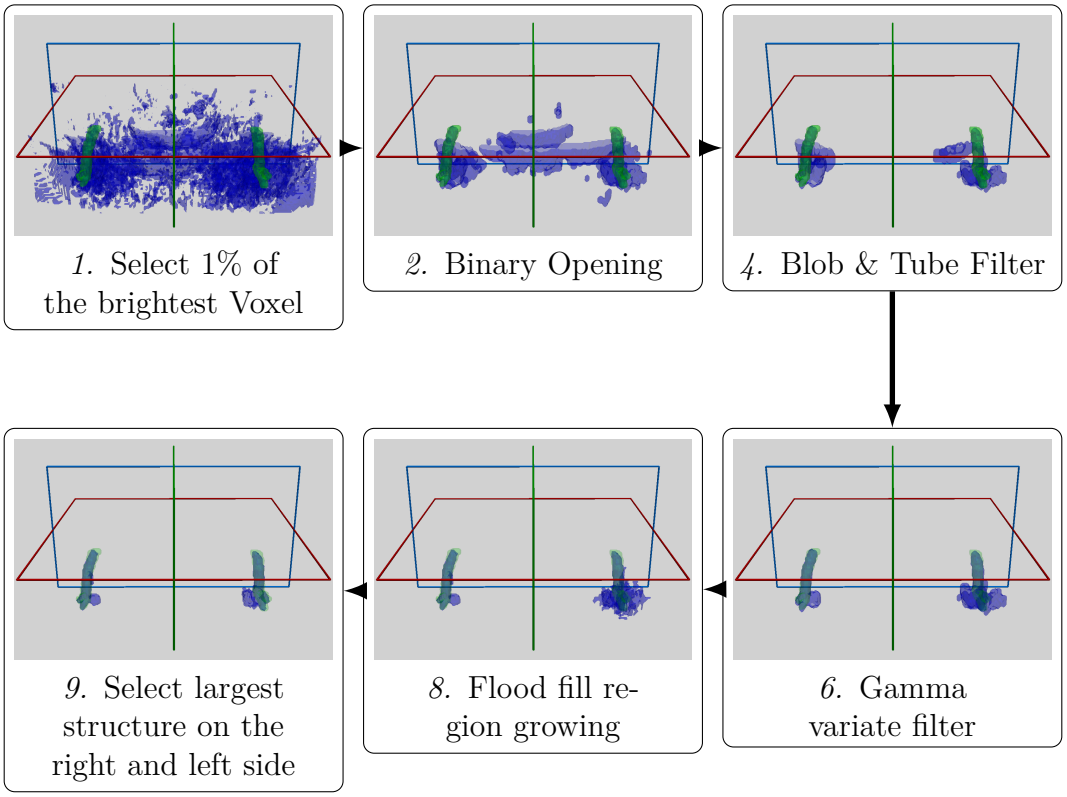


Figure 3.1: Flowchart of the optimized AIF detection filter pipeline (DisLAIF) with a representative 3D rendering to show the result after the step. Each image is the sum of all timesteps and the image data of patient 16 was used.

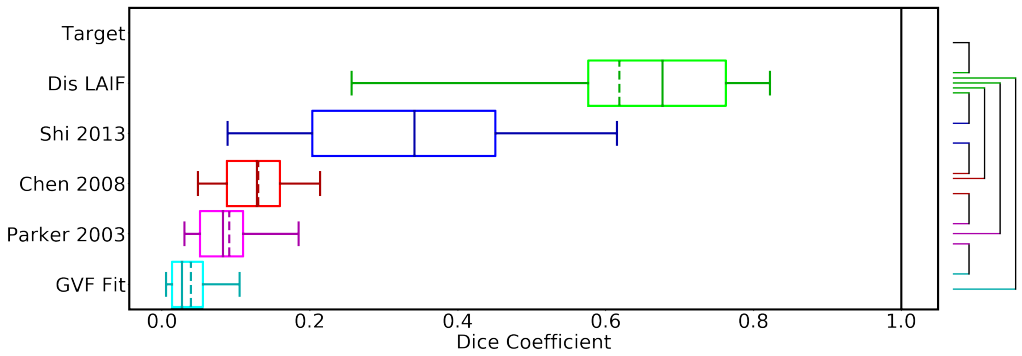
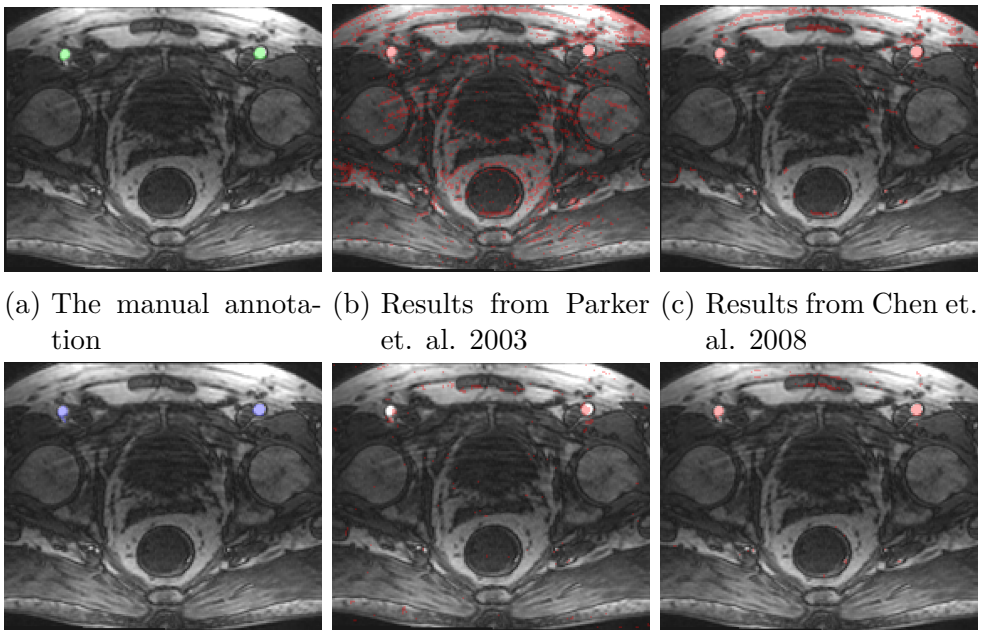


Figure 3.2: Dice coefficients for the different algorithms. The dotted line in this and all following bot plots denotes the mean/average and the solid line is the median value. The brackets indicate a significant ($p<0.05$) difference. Only significance in reference to Dis LAIF was calculated and, if significant, shown.



(a) The manual annotation (b) Results from Parker et. al. 2003 (c) Results from Chen et. al. 2008

(d) Results from Dis LAIF (e) Results from Gamma Variate Fit (f) Results from Shi et. al. 2013

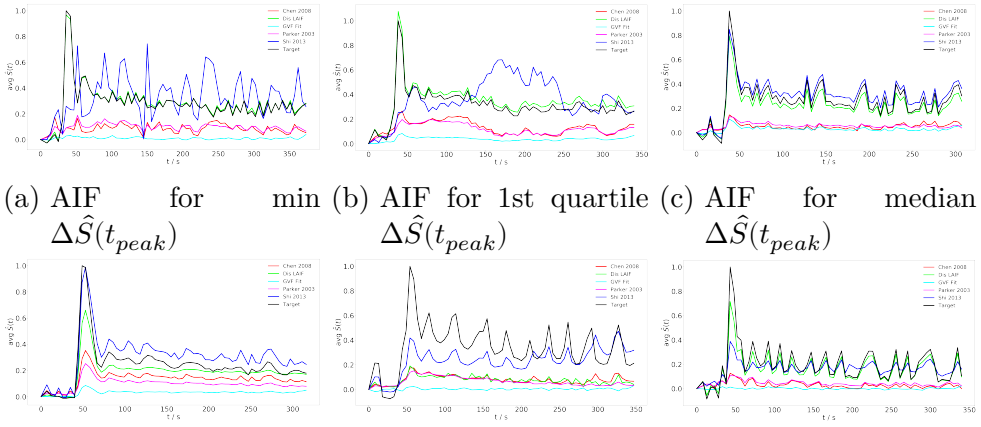
Figure 3.3: The resulting AIF masks on a representative slice and patient for different approaches for one representative slice of patient 16.

of the reference algorithms and blue is our algorithm Dis LAIF. Simple algorithms, like the ones from Parker et. al. and Chen et. al., select many voxels that are outside the artery. The Gamma Variate Fit selects few voxel outside the artery, but also only a part of the artery. The results from Shi et. al. covers the artery and only few additional voxels outside the artery. Dis LAIF finds the artery correctly, but also includes arteries branching off the main trunk.

	Mean Error compared to Manual Annotation			Median Error		
AIF Algorithm	Δt_{peak}	$\Delta\hat{S}(t_{peak})$	ΔAUC	Δt_{peak}	$\Delta\hat{S}(t_{peak})$	ΔAUC
Dis LAIF	**0.050 ±0.218**	**0.270 ±0.230**	**17.268 ±12.568**	**0.000**	**0.195**	**13.281**
Shi	12.825 ± 19.597	0.369 ± 0.377	18.307 ± 14.026	1.000	0.320	14.535
Chen	10.600 ± 20.386	0.691 ± 0.169	24.253 ± 14.476	1.000	0.735	21.560
Parker	2.200 ± 2.638	0.748 ± 0.151	24.257 ± 15.152	1.000	0.776	21.290
GVF Fit	22.100 ± 27.435	0.920 ± 0.044	27.164 ± 16.276	3.000	0.926	24.238

Table 3.1: Error of the different AIF algorithms in comparison to the manual annotation. Shown are the difference of the peak timestep, the difference of the peak value and the difference of the area under the curve. Lowest errors are printed in bold.

Comparing the mean squared error of the generated AIF to the other algo-



(a) AIF for min $\Delta\hat{S}(t_{peak})$ (b) AIF for 1st quartile $\Delta\hat{S}(t_{peak})$ (c) AIF for median $\Delta\hat{S}(t_{peak})$

(d) AIF for 3rd Quartile $\Delta\hat{S}(t_{peak})$ (e) AIF for max $\Delta\hat{S}(t_{peak})$ (f) AIF for average $\Delta\hat{S}(t_{peak})$

Figure 3.4: Several AIF curves generated by the algorithms for different patients. Selected curves show the error range of our algorithm. Based on the error of the peak value the selected curves show the patients where the error is the minimum (a), 1st quartile (b), median (c), 3rd quartile (d), maximum (e) and closest to the average (f)

rithms DisLAIF has the lowest error (0.0022 ± 0.0021) of all algorithms ($p < 0.05$) (cf. Fig. 3.5). The algorithm by Shi et. al., which was developed for finding the AIF in the brain has only slightly larger errors (0.0084 ± 0.0151) than DisLAIF. Chen et. al. (0.0188 ± 0.0116) and Parker et. al. (0.0191 ± 0.0118) have similar errors ($p > 0.05$) which are significantly higher than the errors of Dis LAIF and Shi et. al..

Table 3.1 shows the errors of the AIF algorithm for the normalized peak value $\hat{S}(t)$, the timepoint of the peak value, and the area under the curve in comparison to the manual annotation. It shows, that our algorithm has the lowest average and median difference in comparison to the manual annotation.

$$\hat{S}(t) = \frac{S(t) - S(0)}{S_{target}(t_{peak})} \tag{3.8}$$

In Figure 3.4 the resulting AIF curve for different patients is shown. The shown patients are the ones where the error of the peak value ($\Delta\hat{S}(t_{peak})$ is the minimum (0.033), 1st quartile (0.077), median (0.195), 3rd quartile (0.357), maximum (0.808) and average (0.27) error.

3rd Experiment

The Figure 3.6 shows the percentage difference in the calculated K^{trans} perfusion parameter when the AIF generated by the algorithm is used in contrast

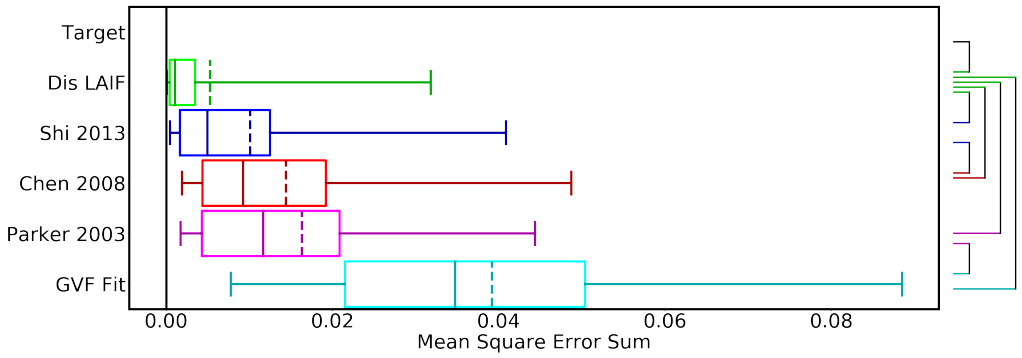


Figure 3.5: Mean squared error sum of AIF from the different algorithms compared to the AIF from the manual annotation. The brackets indicate a significant ($p<0.05$) difference. Only significance in reference to Dis LAIF was calculated and, if significant, shown.

	Mean Error (%)				
AIF Algorithm	F_p	K^{trans}	E	v_p	MTT
Dis LAIF	**21.8 ±21.1**	**21.5 ±20.9**	**7.6 ±7.1**	**42 ±130**	394.3 ± 1225.2
Shi	30.8 ± 27.2	28.9 ± 24.2	9.2 ± 9.4	2086 ± 3420	281.7 ± 825.5
Chen	552.9 ± 541.8	865.9 ± 675.9	64.8 ± 78.5	1532 ± 3105	**169.6 ±349**
Parker	885 ± 1397	1314 ± 1339.9	130.9 ± 222.1	3521 ± 5745	274 ± 644.6
GVF Fit	435 ± 399.1	3691.3 ± 13087.2	814.7 ± 3503.2	5612 ± 6131	6001.6 ± 31768

Table 3.2: Error of the calculated perfusion parameters of the evaluated algorithms in reference to the perfusion parameters calculated with the manual annotation. The values are the average error in percent.

to the manual, target, annotation.

$$Error\%(X,Y) = \frac{|Y-X|}{X} * 100 \tag{3.9}$$

The algorithms by Chen, Parker and the simple GVF Fit truncated in the graph, because with a median Difference of more than 700% and average difference higher than 1300% the graph would not adequately show the small differences between the algorithms with a low error. The Figure shows that the algorithms Dis LAIF ($21.08\% \pm 15.07$) and the one from Shi et. al. ($28.87\% \pm 24.20$). The algorithms by Chen et. al. and Parker et. al. have an average error of 866% and 1313%, that is approximately 40 to 60 times higher than Dis LAIF. Further perfusion parameters provided by the compartment tissue uptake model are shown in table 3.2.

The computation time needed for our algorithm is, with an average of 15 seconds ($\pm 2.3s$), slightly higher than Shi (6.6 ± 0.5), Chen (5.8 ± 0.3) and Parker (1.7 ± 0.04). The gamma variate needed 11 seconds (± 2.3).

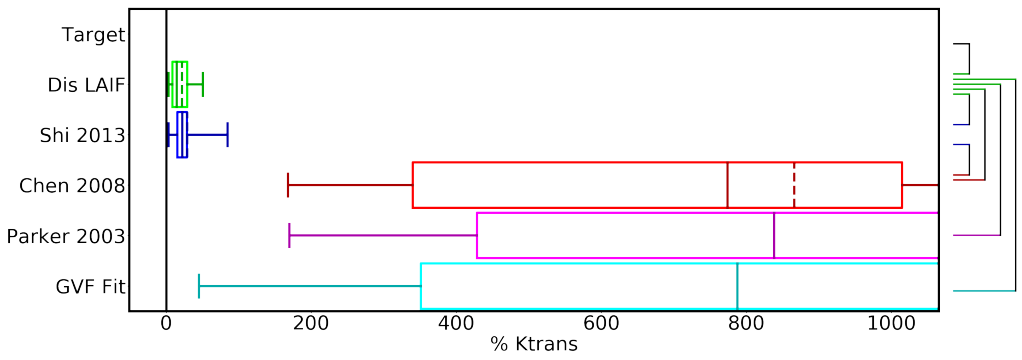


Figure 3.6: The difference in percent of the K^{trans} parameter computed using the compartment tissue uptake model with the manual annotation and the output from different algorithms. The brackets indicate a significant ($p<0.05$) difference. Only significance in reference to Dis LAIF was calculated and, if significant, shown.

3.5 Discussion

Manual annotation of the AIF is a time consuming task, with inter- and intra-user variability in the results [16]. But for the quantification of the perfusion the calculation requires a stable AIF selection to be reproducible. We proposed the algorithm DisLAIF which can fully-automatic select the AIF in a deterministic way. This allows reproducible calculation of the AIF and therefore reproducible perfusion calculation.

Other algorithms have previously been proposed to solve this problem, but our results show, that our algorithms has a significantly ($p<0.05$) lower mean squared error than the algorithms from literature. The error of the K^{trans} perfusion parameter is significantly ($p<0.05$) lower than all algorithms. This is mostly due to more complex filtering, which incorporate steps taken from the previous algorithms. The higher dice coefficient can partly be explained, because the other algorithms were not developed to segment the complete artery and only select enough voxel to reconstruct the AIF.

Parker et. al.[20] uses simple thresholding and the 95% quantile to select the voxel for the AIF. This will select voxel only based on the signal strength regardless of their position and time curve (see Figure 3.3). In the paper by Parker et. al. they used this algorithm to find the AIF in a single, user-selected slice and not in a 3D volume. In a 3D volume the artery is smaller in comparison to the total count of voxel than in an optimally chosen 2D slice. Therefore the selection of the top 5% will select more voxel than is necessary and useful.

The algorithm by Chen et. al. begins similar to the one by Parker et. al., but it also considers the slope of the initial uptake and the 2D and 3D shape of found regions. This drastically reduces the falsely selected voxel. But it still

selects more fat and tumor than arterial voxel and thus the extracted AIF is similar to the AIF by Parker et. al..

Shi et. al. combines thresholding to remove the background and then calculates the contrast agent curves. Based on these curves only voxels with a high peak and a big area under the curve (AUC) are selected. Because the contrast agent curves only contain the change of intensity over time tissue that has a constant high signal has a small AUC and is discarded. So this algorithm selects only few fat voxels.

In our algorithm we used parts of these algorithms and combined them with new aspects. Our shape filter also considers the size of objects and discards small regions. This will reduce remove the selection of noise. Our gamma variate filter not only checks one slope but the general shape of the function. This allows our algorithm to better discern between the main arteries, small arteries and, other tissue like fat and tumor. Because our dataset contains image series with different timings and the single volumes are not registered to each other, it shows a robustness of our algorithm to temporal resolution and motion artifacts. It is expected, that our algorithm has a higher computation time, as it has an overall higher complexity with more filter steps than the compared algorithms. Despite this impact on the computation time the algorithm is still faster than a human reader, who needs approximately 300 seconds per data set.

Our ground truth was annotated by humans and therefore the normal human error apply. A better dataset with error free ground truth could be compiled by using artificial intelligence to generate MRI images from a body phantom[33] and then introduce the AIF by using the population based AIF from Parker et. al.[34].

Our algorithm has several limitations. First, it was developed on images with transversal slice orientation and arteries lying in perpendicular direction to the slices. When transferring our algorithm to other applications, either the blob filter needs to be adapted or, if possible the 3D volume needs to be reformated to a transversal orientation. Furthermore we segment two arteries from the images. This is due to the fact, that it is hard to detect the tumor feeding artery, as this depends on the location of the tumor. Gaa et. al.[27] showed that not taking both arteries into account hampers perfusion quantification. When transferring our algorithm to an application with a single artery, e.g. the abdominal artery in renal perfusion quantification, the last programming step needs to be adapted. Further research towards a broader field of applications is planned.

The pipeline architecture allows to easily facilitate these changes while reusing the majority of unchanged steps. Using deep learning ("AI"), for finding the AIF was discarded, for the sake of understanding the selection. A neural network would be able to learn to find the AIF and the results would be deterministic and reproducible, but it is not possible to easily adapt the neural network to find a different artery. A neural network, once it is trained, is

inflexible and needs to be retrained for a new application, such as an artery in a different part of the body. Also the decisions a neural network makes to select the artery are not observable. The inner working is hidden in the neuron weights. Here a classic algorithm has the advantage of being understandable.

3.5.1 Conclusion

In conclusion, our algorithm adequately solves the aforementioned problems. It can find the AIF ROI with an accuracy that is comparable to a human practitioner, but it is deterministic and generates reproducible results. As a future step, incorporation into perfusion analysis software [35] is warranted to benefit from user independent and reproducible AIF selection and to further automate the process of perfusion analysis in DCE-MRI. In the long term this can help translating quantitative DCE-MRI into clinical routine to the benefit of the patient.

Acknowledgements

This research project is partly supported of the Research Campus M2OLIE funded by the German Federal Ministry of Education and Research (BMBF) within the Framework “Forschungscampus: public-private partnership for Innovations” under the funding code 13GW0388A.

Conflict of Interest

The authors declare no conflict of interest.

4 VirtMRI: A tool for teaching MRI

Christian Tönnes[1,2], Christian Licht[1,2], Lothar R. Schad[1,2], and Frank G. Zöllner[1,2]

[1]Computer Assisted Clinical Medicine, Medical Faculty Mannheim, Heidelberg University, 68167 Mannheim, Germany

[2]Mannheim Institute for Intelligent Systems in Medicine, Medical Faculty Mannheim, Heidelberg University, 68167 Mannheim, Germany

4.1 Abstract

Magnetic resonance image formation is not trivial and remains a difficult subject for teaching. Therefore, we saw an urgent need to facilitate teaching by developing a practical and easily accessible MR image generator. Due to the increasing interest in X-nuclei MRI, sodium image generation is also offered. The tool is implemented as a web application that is compatible with all standard desktop browsers and is open source. The user interface focuses on the parameters needed for the creation and display of the resulting images. Available MR sequences range from the standard Spin Echo and Inversion Recovery over steady-state to conventional sodium and more advanced single and triple quantum sequences. Additionally, the user interface has parameters to alter the resolution, the noise, and the k-space sampling. Our software is free to use and specifically suited for teaching purposes.

4.2 Introduction

Teaching (medical) students about MRI is a balancing act between quantum physics and understandable application. In our medical school, medical students are taught very compressed about the physics of MRI systems at the start of the first semester, and they have a short seminar where a table-top MRI device and a program for generating MRI images are used to show how sequences work and contrasts are generated. A second longer seminar with the same table-top MRI and program is in the fourth year. So far, we have used the software by Hackländer et al. [36] for our teaching. It is a Java program that enables students to test different sequences and the influence of various parameters. The software also supports noise addition, k-space manipulations, and motion artifacts. A disadvantage is that students can access it only during class and hence, we saw an urgent need to develop a tool that is remotely accessible. To the best of our knowledge, the only MRI image generator for teaching that is able to easily solve the accessibility problem published in the

last decade is by Treceño-Fernández et al. [37]. This system is web-based and therefore, could be made accessible over the internet. It also allows students to test different sequences, set the parameters, add different types of noise, manipulate k-space, and use different B0 inhomogeneities. This tool performs all the calculations exclusively on the server, which leads to high server load and bandwidth usage. Therefore, it is suitable for a class setting using the local network, but probably does not scale well if large groups of students access the web page simultaneously over the internet. In addition to this, Treceño-Fernández et al. focus more on the usage of MRI devices and matched their workflow and user interface to those of real MRIs, while the tool presented here aims to demonstrate differences between sequences and the resulting images. Apart from these two MR image generators, there are a number of simulators published in recent years that run on the local computer[38, 39, 40, 41]. Those simulators are mostly developed for researchers or physics and engineering students. For non-technical students like medical students installing programs or Java, using Matlab, simulators that only run on selected operating systems or complicated interfaces that need in-depth knowledge about Bloch equations or sequences make these simulators inaccessible. We present here a different approach to a web-based image generator that performs all computations on the client to eliminate the scaling problem and has a lean user interface.

4.3 MRI Generation

For our system, the main goal was to create a teaching tool that is compact, usable across many platforms, intuitive, and with minimal load on the web server.

4.3.1 Requirements

The work for the server should be minimal, which is realized by performing all computations by the client's device. This requires a system with low computational overhead so that users can run the tool on smartphones or tablets. Programming languages such as HTML, CSS, and JavaScript were used so that the program can be run on any platform with a modern browser. Therefore, the main objective was to make it independent of having a specific operating system, 3rd party software or device. Hence, the following prioritization list has been derived to guide the development of the presented software:

1. low server load
2. remote accessibility
3. cross-platform
4. low resource usage

5. convenient and clear GUI

6. small file sizes

4.3.2 Functions

MRI enables users to create different contrasts between tissues by exploiting different magnetization properties. Therefore, we included multiple sequences, and besides standard hydrogen (^{1}H), sodium (^{23}Na) imaging was also included.

Currently, the system supports six basic sequences for ^{1}H MRI, Spin Echo, Inversion Recovery, and more advanced sequences such as spoiled gradient echo.

The symbols used in the following equations are explained in table 4.1

Spin Echo[5][42]

$$S \propto pd * e^{\frac{-TE}{T2}} * (1 - e^{\frac{-TR}{T1}}) \tag{4.1}$$

Inversion Recovery[43][44]

$$S \propto pd * (1.0 - 2.0 * e^{\frac{-TI}{T1}} + e^{\frac{-TR}{TI}}) * e^{\frac{-TE}{T2}} \tag{4.2}$$

Spoiled Gradient Echo[45][46]

$$S \propto pd * \frac{(1 - e^{\frac{-TR}{T1}}) * sin(FA)}{1 - e^{\frac{-TR}{T1}} * cos(FA)} * e^{\frac{-TE}{T2*}} \tag{4.3}$$

Also included are three steady-state sequences. These are common sequences available on commercial MRI scanners and provide contrasts different from the previous three sequences.

Balanced Steady-State Free Precession (True FISP/FIESTA/Balanced FFE)[47][48]

$$S \propto pd * \frac{(1 - e^{\frac{-TR}{T1}}) * sin(FA)}{1 - (e^{\frac{-TR}{T1}} - e^{\frac{-TR}{T2}}) * cos(FA) - e^{\frac{-TR}{T1}} * e^{\frac{-TR}{T2}}} * e^{\frac{-TE}{T2}} \tag{4.4}$$

Postexcitation Refocused Steady-State Precession (FISP/GRASS, fast MPGR/FFE)[48]

$$S \propto \ pd * tan(\frac{FA}{2}) * e^{\frac{-TE}{T2*}} * \left(1 - (e^{\frac{-TR}{T1}} - cos(FA)) * f(TR, T1, T2, FA)\right)$$
$$f(TR, T1, T2, FA) = \sqrt{\frac{1 - e^{\frac{-2*TR}{T2}}}{(1 - e^{\frac{-TR}{T1}} * cos(FA))^2 - e^{\frac{-2*TR}{T2}} * (e^{\frac{-TR}{T1}} - cos(FA))^2}} \tag{4.5}$$

Preexcitation Refocused Steady-State Precession (PSIF/SSFP/T2-FFE)[48]

$$S \propto pd * tan(\frac{FA}{2}) * e^{\frac{-TE}{T2}} * \Big(1 - (1 - e^{\frac{-TR}{T1}} * cos(FA)) * f(TR, T1, T2, FA)\Big) \tag{4.6}$$

For ^{23}Na imaging, we implemented the signal equation for ^{23}Na, enabling the creation of conventional ^{23}Na MR images. However, sodium is a quadrupole in its nature and thus, exhibits multi-quantum properties. Under certain conditions, one can observe besides the Single Quantum (SQ) also Triple Quantum (TQ) signal, which could provide a richer tissue sodium characterization. Hence, we implemented CRISTINA[49] so we can generate three ^{23}Na images: conventional, single-, and triple-quantum. The single- and triple-quantum images can be further used to calculate the ratio of triple- to single-quantum signal.

^{23}Na Signal [50]

$$\begin{aligned} S \propto\ & (na_{vol} - vol) * mm * (1 - e^{\frac{-TR}{T1}}) * (0.6 * e^{\frac{-TE}{T2f}} + 0.4 * e^{\frac{-TE}{T2s}}) \\ & + vol * na_{mm} * (1 - e^{\frac{-TR}{T1}}) * e^{\frac{-TE}{T2fr}} \end{aligned} \tag{4.7}$$

Single Quantum Spin Echo[49]

$$S_{sq} \propto \frac{1}{|TEs|} \sum_{TE \in TEs} mm * (e^{\frac{-TE+\tau_1}{T2s}} + e^{\frac{-TE+\tau_1}{T2f}}) * sin(FA) \tag{4.8}$$

Triple Quantum Spin Echo[49]

$$S_{tq} \propto \frac{1}{|TEs|} \sum_{TE \in TEs} mm * (e^{\frac{-TE}{T2s}} - e^{\frac{-TE}{T2f}}) * (e^{\frac{-\tau_1}{T2s}} - e^{\frac{-\tau_1}{T2f}}) * e^{\frac{-\tau_2}{T2s}} * sin(FA)^5 \tag{4.9}$$

TQ/SQ Spin Echo

$$S \propto \frac{S_{tq}}{S_{sq}} \tag{4.10}$$

For all these functions, the user can change the used parameters. The parameters are mostly the echo time, the repetition time, or the flip angle. For most sequences, we give an estimate of the acquisition time that a real MRI device would need.

Furthermore, subsampling with different interpolation modes, Gaussian noise, simple k-space manipulation, and 2D or 3D Fourier transform is supported.

For undersampling of the k-space we give a choice of three schemes: Random, density-adapted Pseudo-Random, and Regularly spaced. Random means

	Symbol	Description	Comment
	S	Measured signal strength	proportional to real signal
Tissue Parameters	pd	Proton Density	only ^{1}H imaging
	T1	T1 Relaxation Time	
	T2	T2 Relaxation Time	only ^{1}H imaging
	T2*	T2* Relaxation Time	only ^{1}H imaging
	T2f	T2 Time, fast component	only ^{23}Na imaging
	T2s	T2 Time, slow component	only ^{23}Na imaging
	mm	sodium concentration	in mmol/ml
	vol	fraction of extracellular sodium	
	na_{vol}	voxel fraction containing sodium	fixed to 0.7
	na_{mm}	sodium in water	fixed to 140mmol/ml
	T2fr	T2 Time for free sodium	fixed to 60ms
Sequence Parameters	TE	Echo Time	set by user in milliseconds
	TR	Repetition Time	set by user in milliseconds
	TI	Inversion Time	set by user in milliseconds
	FA	Flip Angle	set by user in degree
	τ_1	Time between 1st and 2nd RF pulse	set by user in milliseconds
	τ_2	Time between 2nd and 3rd RF pulse	set by user in milliseconds

Table 4.1: Explanation for the symbols used in the signal equations.

an arbitrary decision to include or discard a voxel in k-space, yielding non-Cartesian k-space trajectories. The other two schemes retain or discard complete phase-encoding lines in k-space, representing Cartesian trajectories. Regularly spaced means for a 50% sampling fraction f_s that every second line is measured, for 33% every third. The condition for measuring a line y is shown in equation 4.11. If the condition for $measure(y)$ is true, then the line y is measured; otherwise it will be dropped. The line numbering y starts at 1.

$$measure(y) = \begin{cases} \frac{\text{ceil}(y \cdot f_s)}{f_s} - y & < 1 \quad f_s < 0.5 \\ \frac{\text{ceil}(y(1-f_s))}{(1-f_s)} - y & \geq 1 \quad f_s \geq 0.5 \end{cases} \tag{4.11}$$

A commonly used sampling scheme is the density-adapted pseudo-random sampling, which keeps the full k-space center, and the probability to drop a line increases with the distance from the center. This is a common sampling scheme for compressed sensing [51]. We always keep a fraction f_{in} 10% in the center of the complete k-space. Then a random number is generated at each line and compared to a linearly decreasing threshold. The parameters for this threshold are chosen so that the resulting sampling rate is the selected sampling rate. The calculation for these parameters is shown in equation 4.12. In these equations, Dim_y is the total number of y lines.

$$
\begin{aligned}
b &= \begin{cases} \frac{f_s - f_{in}}{0.5\,(1-f_{in})} - 1 & f_s < 0.5 \\ 2 - \frac{f_s - f_{in}}{0.5\,(1-f_{in})} & f_s \geq 0.5 \end{cases} \\
a &= \frac{b}{1-f_{in}} \\
\Psi(y) &= \begin{cases} 2 * y/Dim_y & y/Dim_y < 0.5 \\ 2 * (1 - y/Dim_y) & y/Dim_y \geq 0.5 \end{cases} \\
measure(y) &= r_{random} \leq -a\,(\Psi(y) - f_{in}) + b
\end{aligned}
\tag{4.12}
$$

4.3.3 Architecture

The architecture can be viewed on two levels. There is a server-client architecture to deliver the web app to the browser. Here we use static files which can be served by every standard web server. This project is based on the SimpleHttpServer included in Python 3.

The web page uses the Model-View-Controller pattern and offloads the computation to a worker thread. It only connects to the server to load a data set. After that, all computation and data handling is performed within the browser. The view is written in HTML and CSS using the CSS files from the Bootstrap project. Some responsive behavior, e.g. calculating the needed time for a scan, is calculated in JavaScript. The controller uses JavaScript and most of the computation is written in both JavaScript and c/WebASM. An exception is the FFT, where we use the KissFFT project, which is only written in c and then compiled with emscripten to WebASM. This is done to speed up the computations. We purposely did not write everything in c/WebASM so that an interested user can simply open the web developer tools and follow the computation with the built-in debugger. The WebASM version of the image creation process has a faster runtime and is therefore set as the default computation backend.

4.3.4 Data sets

Each data set consists of multiple 3D arrays for the different parameter maps. For 1H MRI that includes: T1, T2, T2*, and proton density. For ^{23}Na, the parameters are T1, T2 fast, T2 slow, sodium density, and extracellular volume fraction. Every array has a size of 256x256x256 voxels and was generated using published head phantoms [2][52][53][54]. The data sets generated using Aubertbroche et al.[2][52] and Holmes et al.[53] are available for 3T and 1T ^{1}H and 3T ^{23}Na MRI and the data set generated from Alfano et al.[54] is 1T and 1.5T ^{1}H MRI. The phantoms we used consisted of segmentation masks for different tissue types. We used these to generate the parameter maps by simply inserting the values for each parameter found in the literature (Tables 4.2). These maps were then resampled to 256x256x256 voxels.

Parameters for 1.5T ^{1}H[52]				
Tissue	T1 [ms]	T2 [ms]	T2* [ms]	PD
Background	0	0	0	0
CSF	2569.0	329	58	1
Grey Matter	833	83	69	0.86
White Matter	500	70	61	0.77
Fat	350.0	70.0	58	1
Muscle	900.0	47	30	1
Muscle / Skin	569.0	329	58	1
Skull	0	0	0	0
Vessels	2569.0	329	0	1
Dura Mater	2569.0	329	58	1
Bone Marrow	500.0	70	61	0.77

Parameters for 3T ^{1}H[55][56][57]				
Tissue	T1 [ms]	T2 [ms]	T2* [ms]	PD
Background	0	0	0	0
CSF	4163.0	329	58	1
Grey Matter	1445	83	66	0.86
White Matter	791	75	53.2	0.77
Fat	346	68	58	1
Muscle	1420	44	30	1
Muscle / Skin	371.0	133	58	1
Skull	0	0	0	0
Vessels	1984.4	275.0	0	1
Dura Mater	2569.0	329	58	1
Bone Marrow	365	133	61	0.77

Parameters for 3T ^{23}Na[58][59]					
Tissue	T1 [ms]	T2 slow [ms]	T2 fast [ms]	Extracellular fraction	Sodium [mmol]
Background	0	0	0	0	0
CSF	50	60	60	1	140
Grey Matter	30	60	2	0.21	55
White Matter	30	60	2	0.17	45
Fat	10	50	4	0.2	0
Muscle	25.2	30	2	0.2	20
Muscle / Skin*	25.2	30	2	0.2	20
Skull	0	0	0	0	0
Vessels	38.4	20	3	1	150
Dura Mater*	10	50	4	0.2	0
Bone Marrow*	10	50	4	0.2	0

Table 4.2: Parameters used for 1.5T ^{1}H, 3T ^{1}H and 3T ^{23}Na images. Parameter names are in analogy to [2]. *: Parameters were not found, approximated with values for fat/muscle

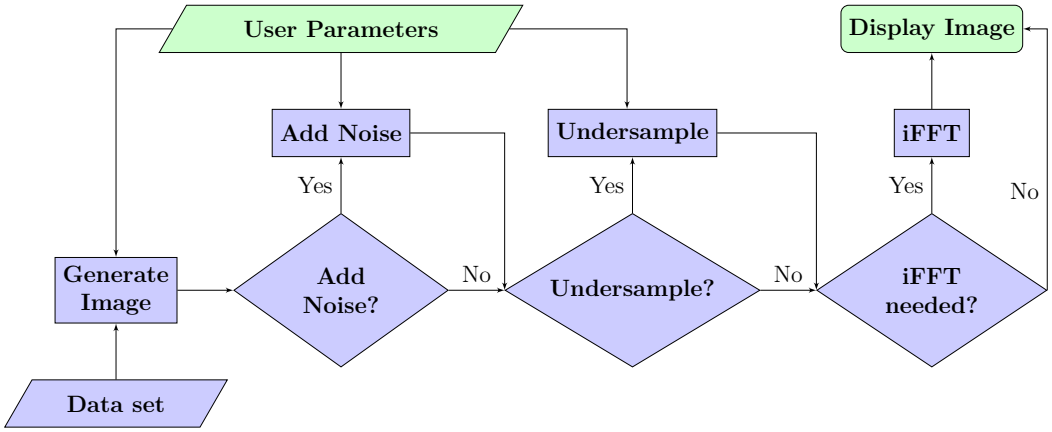


Figure 4.1: Image generation pipeline. The green boxes are part of the front end and the blue ones are in the back end.

4.3.5 Image generation Pipeline

Our generation process (Fig. 4.1) is quite straightforward. MR images are computed in the image domain using the equations 4.1 to 4.10 for every voxel and followed by a Fourier transform to calculate the k-space. For added noise, random numbers, chosen from a Gaussian distribution, are added to each value in the k-space and if undersampling is activated, the k-space is filtered using the selected sampling scheme to remove a configurable percentage of the total lines prior to the inverse Fourier transformation. If the k-space was modified, an inverse Fourier transform is used to compute the final image to be displayed.

4.3.6 GUI

The user interface is written using HTML and CSS. The base CSS files are from the bootstrap project (version 5)[60], a toolkit to build web frontends. The dark Gruvbox[61] scheme was chosen for the color theme. When the user opens the web page, they first have to choose a data set. After loading the data set, the input fields for general parameters and sequences become visible (Fig. 4.2). The link ”Dataset source” next to the drop-down box always links to the webpage of the selected data set, where the input files for each data set can be downloaded. An MRI sequence can be selected by clicking on the corresponding tab, which also visualizes the specific parameters for this sequence. The parameters for each sequence are independent, e.g. changing the Echo Time in Inversion Recovery does not change the Echo Time for Spin Echo. Only the selected and visible parameters are used for a sequence, except for the ’^{23}Na TQ/SQ’ sequence, which uses the parameters of the ’^{23}Na SQ’ and ’^{23}Na TQ’ tabs. The field ’Total Measuring Time’ provides an approximation of the time required to conduct the selected MRI experiment.

The general parameters are in an accordion menu and can be expanded or collapsed as needed. In the screenshot, the menu ”General Parameters”

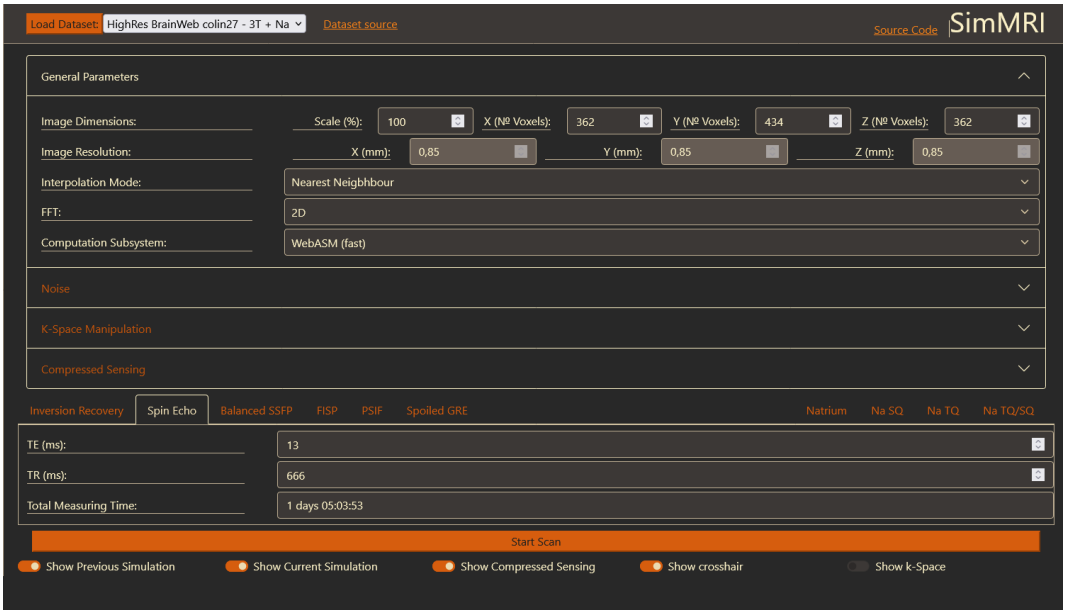


Figure 4.2: The GUI after loading a data set. General parameters are stated at the top and sequence specific parameters are found at the bottom.

is expanded and the menus for noise and compressed sensing are collapsed. Collapsed menus have a different font color to emphasize that they can be expanded. We chose this to signify that all general parameters are always used for image generation, except for the sequence parameters, only the ones on the currently active and visible tab are used. However, displaying all parameters at once creates an overloaded interface, so the user has the option to collapse them. The button with the label ”Start Scan” starts the generation process. The computed images will be displayed below (Fig. 4.3). The toggle buttons allow the user to select which images should be displayed and to show or hide the respective k-space.

Every image is displayed in a four-panel view. The top left corner contains a transversal, the top right a sagittal, and the bottom left a coronal slice. The bottom right quarter is either the k-space or a 3D view of the current slices, which also allows for rotation of the view.

After generating a second image, the user can now decide to view both of them next to each other (Fig. 4.4) or only one of them. Fig. 4.4 shows a comparison of two Spin Echo images with different TE values. Additionally, the crosshair has been hidden and the 3D view is replaced with the respective k-spaces.

The user can interact with the other sections of the image by holding the mouse button and moving it, which changes the center and width of the window. The slice can be changed with the mouse wheel. Both the windowing and the slice can also be selected using the input fields below the image. When a 23Na is created, the selection fields for the windowing are replaced

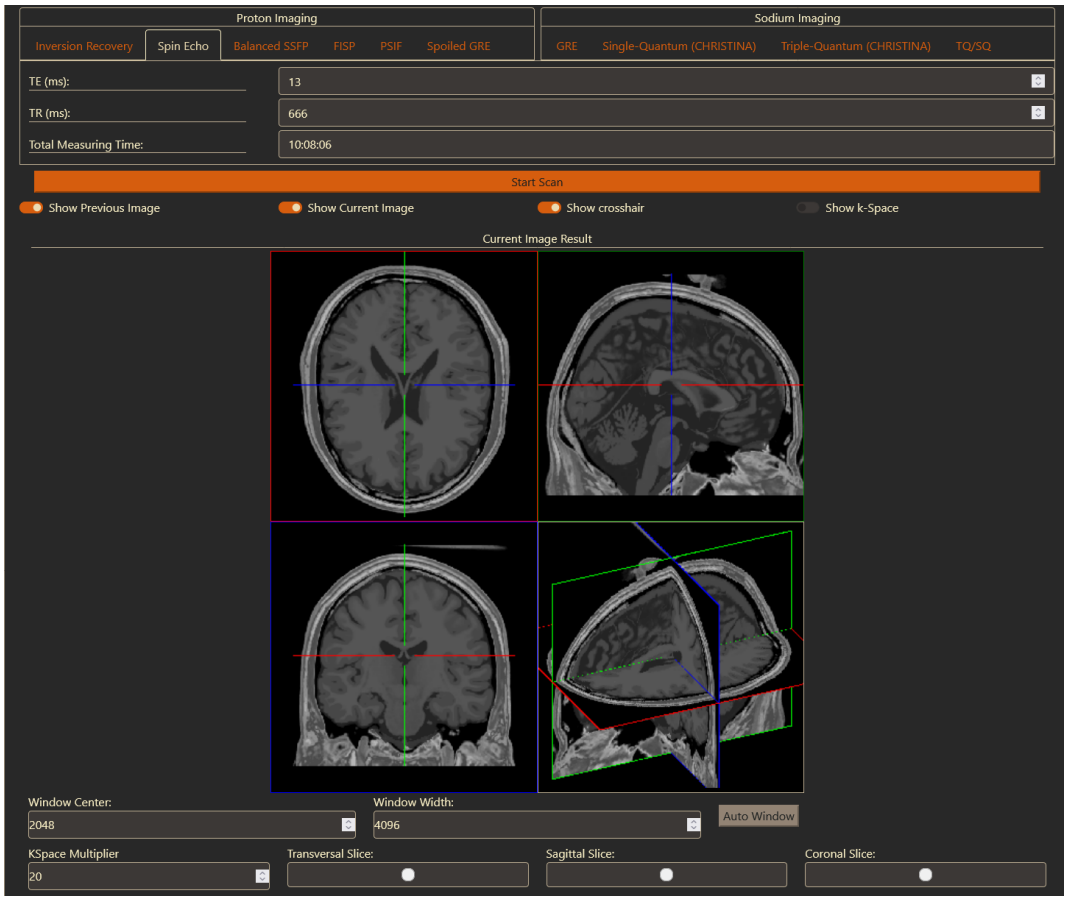


Figure 4.3: Resulting signal computation for a Spin Echo acquisition based on Eq. 4.1. The Viewer shows a slice from the transversal, sagittal, and coronal planes and then these three slices in a 3D view. Control panels to manipulate and navigate through the images are found beneath the images.

Figure 4.4: Generation of a second Spin Echo image. On the left-hand side is the previous Spin Echo image from Fig. 4.3 and on the right side the new Spin Echo image is shown. The crosshair has been turned off and the 3D view replaced with the k-space.

by a color bar. Slice selection for the image and the k-space is partly synchronized. Scrolling through one image also scrolls through the other visible images, enabling users to compare images with different acquisition parameters and sequences. Slice selection using the input boxes below the image is not synchronized. This allows a user to set every image to a different slice. In that case, the scrolling is still synchronized, but the offset between the slices is kept until one image is at the first or last slice.

To generate an image with a lower resolution, the value of the scale field has to be changed. The interpolation mode then decides how to calculate the voxel value. Possible options are to use the nearest neighbor or to average over all voxels in the data set that would be within the virtual image voxel.

4.4 Results

We will focus on the impact of changing the general parameters, resolution, interpolation mode, 3D vs. 2D FFT, computational subsystem, and noise (Table 4.3). For all parameters, the JavaScript version was much slower than WebASM. Chrome was always slower than Firefox when using WebASM, but Chrome was faster most of the time when using the JavaScript version. Generating an image using only the nearest point of the data set to the center of a voxel is faster than averaging over all data points inside the voxel. Reducing the size of the generated image also reduces the computation time, since the Fourier transformations are quicker and they take up a big share of the total computation time. When noise is added to the k-space, an additional inverse Fourier transform is required to obtain the noisy image, which, as expected, increases the running time.

Spin Echo and Inversion Recovery images generated with the here proposed software are shown in Fig. 4.5. The first four rows show generated images using Spin Echo and different echo time, ($TE_1 = 0.1ms$,$TE_2 = 13ms$,$TE_3 = 42ms$,$TE_4 = 121ms$). Furthermore, two Inversion Recovery images are shown with different inversion times chosen to suppress White Matter ($TI_1 = 600ms$) and Grey Matter ($TI_2 = 993ms$).

The steady-state sequences are shown in Figure 4.6. Similar to the previous figure, different parameters are used in each row and the same three slices of the head are shown.

Figure 4.7 shows the Spin Echo sodium sequences in addition to single and triple quantum imaging. These images are generated with a reduced resolution to better resemble state-of-the-art for sodium imaging in reality. The Spin Echo images are downsampled to an isotropic voxel size of 4mm and the single/triple quantum images have a voxel size of 16mm.

Noise	XxYxZ	Interpolation	FFT	Compute Subsystem	Runtime Firefox [s]	Runtime Chrome [s]
No	256x256x256	Nearest	3D	JavaScript	6.85 ± 0.70	**5.85 ±0.12**
				WebASM	**2.98 ±0.12**	4.16 ± 0.26
			2D	JavaScript	6.57 ± 0.69	**6.03 ±0.81**
				WebASM	**3.13 ±0.49**	4.19 ± 0.46
		Average	3D	JavaScript	7.78 ± 0.81	**7.50 ±1.05**
				WebASM	**3.08 ±0.17**	4.29 ± 0.44
			2D	JavaScript	7.66 ± 0.54	**7.51 ±1.00**
				WebASM	**3.16 ±0.22**	4.31 ± 0.48
No	256x256x64	Nearest	3D	JavaScript	1.66 ± 0.14	**1.52 ±0.19**
				WebASM	**0.77 ±0.05**	1.07 ± 0.13
			2D	JavaScript	1.65 ± 0.10	**1.54 ±0.20**
				WebASM	**0.74 ±0.04**	1.06 ± 0.09
		Average	3D	JavaScript	3.83 ± 0.23	**2.91 ±0.33**
				WebASM	**1.51 ±0.09**	2.34 ± 0.25
			2D	JavaScript	4.09 ± 0.44	**2.94 ±0.40**
				WebASM	**1.68 ±0.28**	2.36 ± 0.27
Yes	256x256x256	Nearest	2D	JavaScript	11.38 ± 2.07	**10.40 ±1.37**
				WebASM	**4.65 ±0.51**	7.12 ± 0.97
		Average	2D	JavaScript	**11.99 ±1.57**	12.20 ± 1.83
				WebASM	**4.58 ±0.51**	7.48 ± 1.13
	256x256x64	Nearest	2D	JavaScript	**2.72 ±0.31**	2.77 ± 0.52
				WebASM	**1.20 ±0.13**	1.86 ± 0.29
		Average	2D	JavaScript	4.92 ± 0.68	**4.26 ±0.84**
				WebASM	**1.94 ±0.20**	3.19 ± 0.51

Table 4.3: Runtimes for several parameter combinations using a Spin Echo sequence (TE: 23, TR: 666). All parameter combinations are computed with the slow JavaScript and the faster WebASM version. The runtimes are averaged over 10 runs, on a PC with Intel i5-6500 CPU and 64GB RAM. Maximum RAM used by the Browsers: Firefox 1.8GB, Chrome 1.9GB.

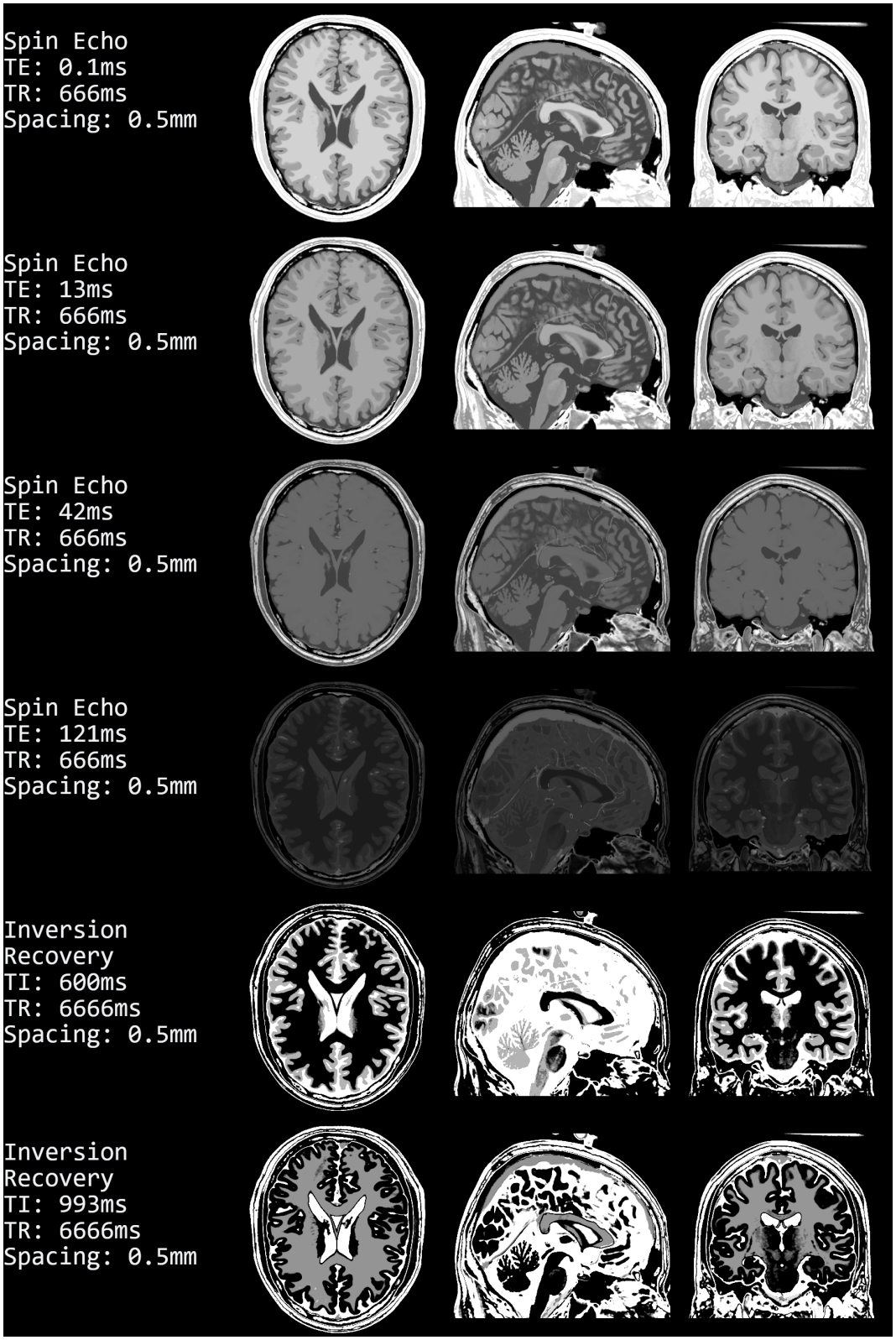


Figure 4.5: Generated images using the Spin Echo sequence (Eq. 4.1) with different Echo Times and Inversion Recovery sequence (Eq. 4.2) with different inversion times. In all rows, the shown slices are the middle slice in the transversal, sagittal, and coronal planes.

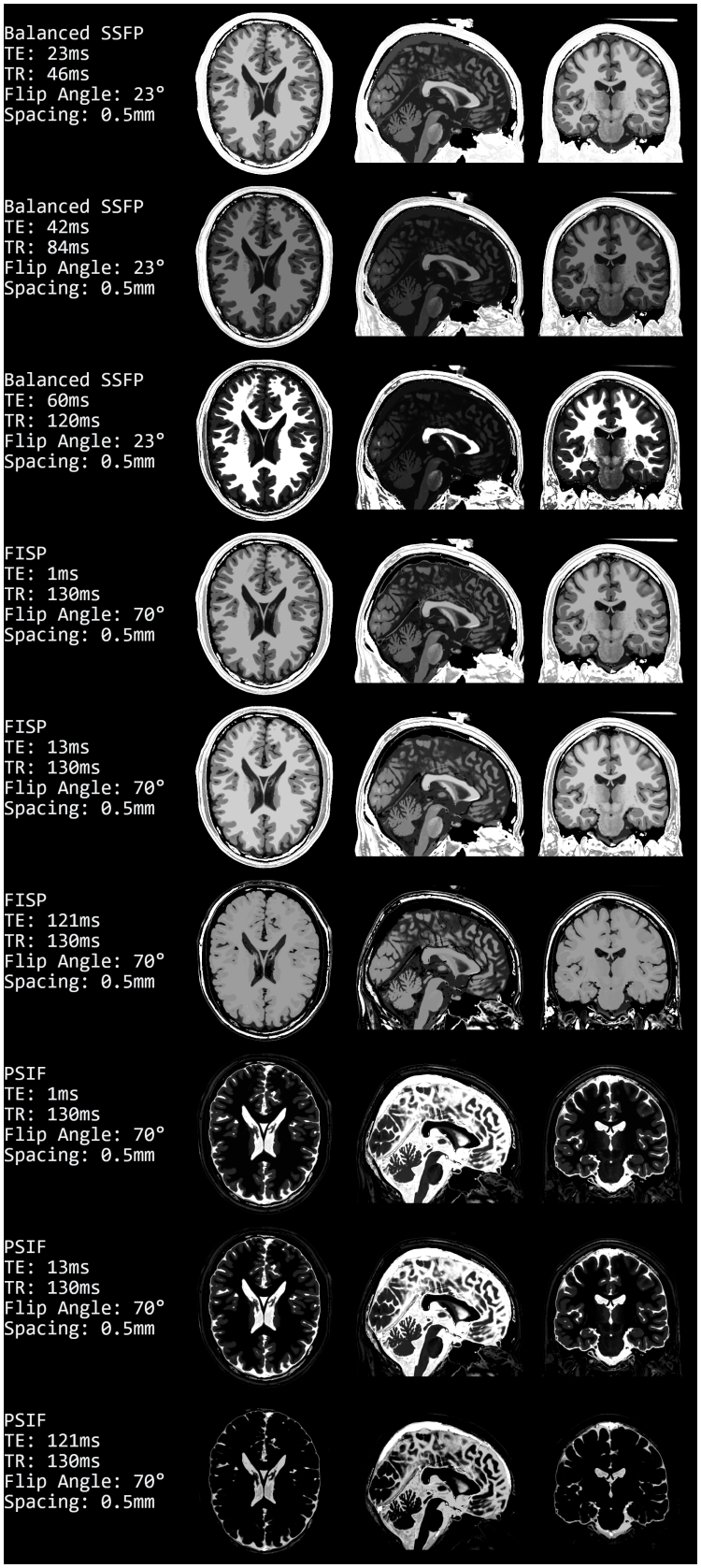


Figure 4.6: Images computed using the steady-state sequences balanced SSFP (Eq. 4.4), FISP (Eq. 4.5), and PSIF (Eq. 4.6).

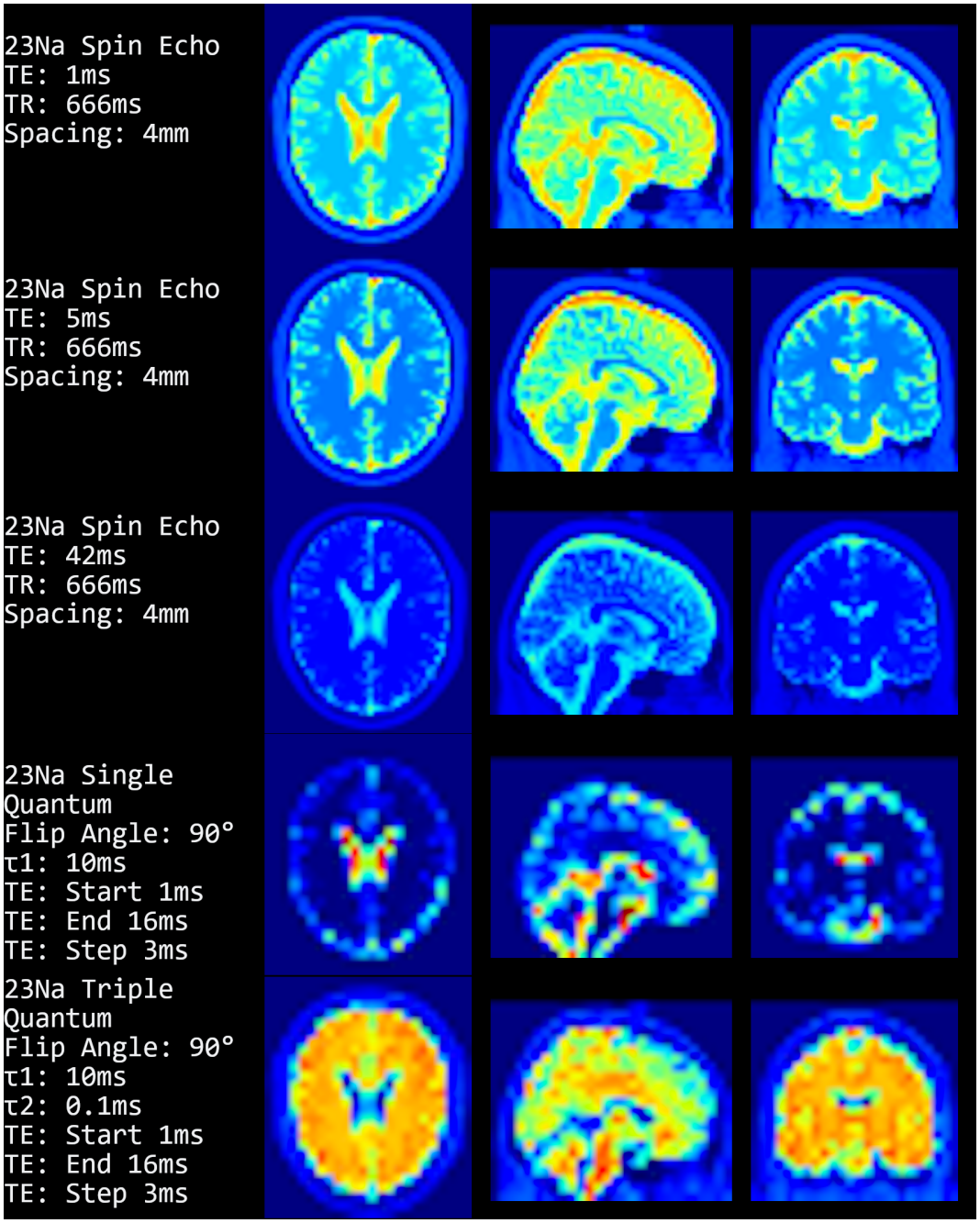


Figure 4.7: Images were computed using the equations for the sodium sequences. The top three rows show ^{23}Na Spin Echo (Eq. 4.7), then follows a single quantum (Eq. 4.8) and a triple quantum (Eq. 4.9) image. The voxel size and spacing for the single/triple quantum images is 16mm.

4.5 Discussion

In summary, we have presented a web-based image generator designed for teaching. The software allows the generation of signals based on different sequences and the demonstration of the influence of the parameters. It supports a vast amount of sequences ranging from standard proton Spin Echo to more advanced sodium sequences. Further, the system contains the options to add noise, change image resolution, and k-space undersampling with different strategies. Secondly, a user-friendly interface was developed that eases usage. Additionally, the software can run on a wide range of devices, which is attributed to the fact, that the software was developed as a web-based application. Lastly, the server management and costs are reduced since we have only static files. No computation is necessary on the server, and static files can be distributed to many clients without much effort. Deploying to a new server is also simple, just copy the "wwwroot" folder from the GitHub repository [62], this contains all the required source and data files.

Our tool started as a small piece of software, but with added functions, it increased in size and computational cost. While still able to be used on smartphones, a user has to wait for some time until the computation process is finished. One solution to reduce computational cost would be to detect mobile devices like a smartphone and then provide a reduced version of the web page, e.g. only allowing nearest-neighbor interpolation and 2D Fourier transformation. Another solution, which we implemented already, is to code the computationally intense algorithms in c and then compile them to WebASM. This makes the computation process less transparent because interested users would not be able to simply open the web debugger (which can be accessed in most major browsers by pressing F12) and look at how the program runs inside their browser. On the other hand, we see that the WebASM version only takes 50% of the time required by the JS version to calculate an image. This will allow a user to choose between the slow, but debuggable, JavaScript and the fast, oblique WebASM version.

Future work will focus on including parallel imaging and compressed sensing. Both are implemented in modern MRI devices and are quite interesting.

The added noise and image artifacts are quite basic. So far, the user can only select Gaussian noise. A possible artifact we could add without much hassle is B0 homogeneity by extending the image creation process with the inclusion of a static homogeneity map. The movement of the patient would be somewhat more difficult. To include a single and fast movement of the complete patient, the interpolation grid could be shifted and rotated during the computation. This would require computing the images and k-spaces twice, and then merging these k-spaces so that the points captured before and after the motion are from the corresponding k-space. While this is not a perfect representation of patient movement, it should be a usable approximation and starting point for more complex movements. The flexible interpolation grid

required for the proposed patient motion artifacts could also be used for other purposes, such as changing the orientation of the slices. Setting the slice orientation could be done by simply changing values for the rotation in several input boxes. But we think this would not be intuitive and a better approach is a 3D view, similar to what Treceño-Fernández et al. implemented.

Other tools focus on having a GUI that resembles a real MRI machine. However, we focused on convenience in regard to usage and accessibility, which was the reason to neglect the implementation of a scanner related interface. The workflow for acquiring images on a real MRI is beyond the scope of this software.

In conclusion, we have presented a web-based image generator for a wide range of MR sequences that is scalable, cross-platform, and freely available.

Declarations

Ethical Approval

not applicable

Competing interests

The authors declare no competing interests.

Authors' contributions

Christian Tönnes wrote the main manuscript text, created the figures and programmed the software. Christian Licht advised and reviewed all aspects of this work and software, including but not limited to: compressed sensing, sodium mri, sequence equations, etc. Frank Zöllner and Lothar Schad supervised our work, gave advice, and reviewed the manuscript.

Funding

This research project is partly supported of the Research Campus M2OLIE funded by the German Federal Ministry of Education and Research (BMBF) within the Framework "Forschungscampus: public-private partnership for Innovations" under the funding code 13GW0388A.

5 Feature-based CBCT Self-Calibration for Arbitrary Trajectories.

Christian Tönnes[1], Tom Russ[1], Lothar R. Schad[1], and Frank G. Zöllner[1]

[1]Computer Assisted Clinical Medicine, Mannheim Institute for Intelligent Systems in Medicine, Medical Faculty Mannheim, University Heidelberg, Mannheim, Germany

5.1 Abstract

Purpose: Development of an algorithm to self-calibrate arbitrary CBCT trajectories which can be used to reduce metal artifacts. By using feature detection and matching we want to reduce the amount of parameters for the BFGS optimization and thus reduce the runtime.
Methods: Each projection is 2D-3D registered on a prior image with AKAZE feature detection and brute force matching. Translational misalignment is calculated directly from the misalignment of feature positions, rotations are aligned using a minimization algorithm that fits a quartic function and determines the minimum of this function.
Evaluation: We did three experiments to compare how well the algorithm can handle noise on the different degrees of freedom. Our algorithms are compared to Broyden–Fletcher–Goldfarb–Shanno (BFGS) minimizer with Normalized Gradient Information (NGI) objective function, and BFGS with distance between features objective function using SSIM, nRMSE, and the Dice coefficient of segmented metal object.
Results: Our algorithm (Feature ORiented Calibration for Arbitrary Scan Trajectories with Enhanced Reliability (FORCASTER)) performs on par with the state-of-the-art algorithms (BFGS with NGI objective). nRMSE: FORCASTER=0.3390, BFGS+NGI=0.3441; SSIM: FORCASTER=0.83, BFGS+NGI=0.79; Dice: FORCASTER=0.86, BFGS+NGI=0.87.
Conclusion: The proposed algorithm can determine the parameters of the projection orientations for arbitrary trajectories with calibration quality comparable to state-of-the-art algorithms, but faster and with higher tolerance to errors in the initially guessed parameters.

5.2 Introduction

Arbitrary trajectories can be used to reduce metal artifacts[63] or cone beam artifacts[64], change field of view[65], and to reduce needed projections[66]. For the quality of these CBCT images, the exact position and rotation at which each projection was acquired is essential. Even though, modern engineering produces machines which can detect their position with a high accuracy, this accuracy is still not sufficient for an artifact-free image. For circular trajectories several algorithms have already been developed [67][68][69], these algorithms use properties specific to circular trajectories which gives them a significant speed advantage over our proposed algorithm, but it also means they are not usable for arbitrary trajectories. Other calibration methods use phantoms consisting of several metal balls [70][71][72]. Here the phantom is imaged and then the trajectory can be calibrated using geometric analysis. Only after these two steps the trajectory can be used for the intended image acquisition. This does not work for trajectories that are created on the fly for the current patient and situation, or when the imaging system cannot accurately reproduce the same trajectory. For the calibration of completely arbitrary trajectories only a few papers are published. Ouadah et. al. uses normalized gradient information as the objective function for a Broyden–Fletcher–Goldfarb–Shanno (BFGS) minimization [73]. Chung et al. [3] uses BFGS minimization with an object function based on the distance of Speeded Up Robust Features (SURF)[7] features in simulated forward projections and the acquired images. Both algorithms need multiple hours for a calibration run. Furthermore, both algorithms are evaluated on a regular CBCT image of the same object, which is acceptable for experimental settings, but not for clinical routine examinations. The calibration algorithms has to work with a prior image that is older and differs from the current image.

5.3 Methods

For arbitrary trajectories the calibration algorithm cannot use any inter-image consistency conditions for the projection parameters. Every projection is pairwise independent, and relations, like being next to each other or on the opposite site, are not known and cannot be assumed to exist. We will however, assume that source and detector never change their position relative to each other, e.g. because they are mounted on a c-arm. This leads to a 2D-3D registration for every single projection. Such a registration typically consists of an optimization (also called minimization) algorithm and an objective function. In the approach by Ouadah et. al.[73] or Chung et al.[3] they use the optimization algorithm BFGS to minimize an objective function. This objective function evaluates all projection parameters at the same time and gives an estimate for the correctness, with lower values meaning that the parameters are closer to the correct values. We, instead, propose using different objective functions,

one for each parameter. This approach allows us to create objectives, that are sensitive towards change in only one of the parameters.

5.3.1 Projection and Optimization Parameters

In this paper, we use three 3D vectors to describe the position and orientation of a projection (Fig. 5.1). The vector $\vec{d}$ points to the middle of the detector and $\vec{u}$, $\vec{v}$ contain the direction and distance from the center of one detector element to the center of neighbour elements on the left and top. This definition is equal to the vectors $\vec{d}$, $\vec{u}$ & $\vec{v}$ used by the Astra toolbox[74] to define cone beam geometries.

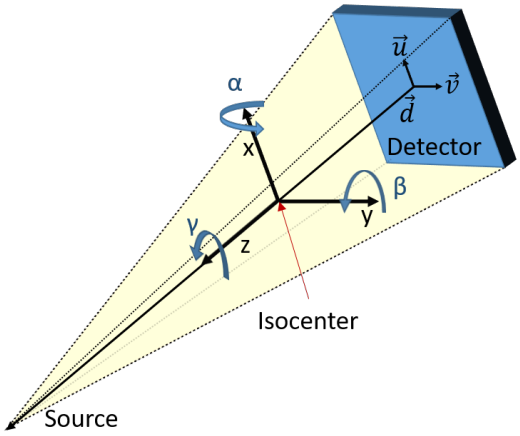


Figure 5.1: Overview of the coordinate system, parameters and degrees of freedom.

On these vectors we have the six common degrees of freedom in three dimensional space, that is three translations along the cartesian axes and three rotations around these axes. These optimization parameters use a coordinate system where the x- and y-axis point in the direction of $\vec{u}$ and $\vec{v}$. The z axis is then given by the direction of the cross product $\vec{u} \times \vec{v}$ and points towards the source. Therefore, the translations and rotations all depend on the current orientation of the projection.

With this coordinate system a movement along the x- or y-axis corresponds to simple horizontal or vertical shifts of the pixel values in the projection. A rotation around the x- or y-axis results in points moving horizontally or vertically. Movement in the z-direction zooms the image in or out and rotation around the z-axis rotates the projection without any other change.

5.3.2 Feature Points Matching

The algorithmic parts shared by all of our investigated algorithms are feature detection and matching. Features are detected in the real image p_{ri} and a

simulated image p_{si} using the Accelerated KAZE (AKAZE) algorithm [9]. This algorithm detects features and computes a descriptor for each feature. Then the features from one image can be matched to the ones from the other image by comparing the feature descriptors using the hamming distance. The hamming distance is the number of different elements in two vectors of equal length. For every feature in one image we will find the two features in the other image with the lowest hamming distance. These are then used to perform the ratio check described by Lowe et. al. [8] which will discard wrongly matched features. Furthermore, we discard matches if two or more points in one image are matched to the same point in the other image. Also discarded are matches with larger distances than one standard deviation plus the mean distance of all matched points (already excluding multiply matched points). Now we have a set of points $\vec{p_{si}} \in \Omega$ in the simulated images and a function to match them to points $\vec{p_{ri}}$ in the real, acquired, image.

$$\vec{p_{ri}} = \Psi(\vec{p_{si}})$$
$$\vec{p_{si}} = \Psi^{-1}(\vec{p_{ri}})$$

5.3.3 Correcting Shifts

First, we will present the method for correcting shifts along the X- and Y-axis (Listing 1). We calculate the shift along the x or y axis of every pair of points in the real image and the simulated image. For a perfect matching image this shift would be zero. So we simply move the simulated image by the median detected shift.

```
def correctXY(cur, iterations):
    for i in range(iterations):
        projs = ForwardProjection(cur)
        Ψ, Ω = FindFeatures(projs, real_acquisitions)
        diff = [Ψ(p⃗) − p⃗ for p⃗ in Ω]
        med = median(diff, axis=0)

        xdir = cur[1]; ydir = cur[2]
        cur[0] += med[0] * xdir
        cur[0] += med[1] * ydir
    return cur
```

Listing 1: Calibration function for shifts in x&y direction.

For correcting shifts along the z-axis our function calculates the pairwise distance between points within each image and then uses the median ratio of these distances multiplied with the distance between source and iso-center for the shift along the z-axis (Listing 2). This ignores misalignment in x or y

direction and only considers the magnification. If in both images all distances have the same length the median ratio is one and no zooming is necessary.

```
def correctZ(cur, iterations):
    for i in 1 ..its:
        proj = ForwardProjection(cur)
        Ψ, Ω = FindFeatures(proj, real_acquisitions)

        dist_sim  = [ ‖  p⃗1  −  p⃗2  ‖2 for p⃗1, p⃗2 in Ω ]
        dist_real = [ ‖ Ψ(p⃗1) − Ψ(p⃗2) ‖2 for p⃗1, p⃗2 in Ω ]
        scale = median(dist_real/dist_new) - 1

        xdir = cur[1]; ydir = cur[2]
        zdir = cross_product(ydir, xdir)
        zdir = zdir / ‖zdir‖2
        cur[0] += dist_source_origin * scale * zdir
    return cur
```

Listing 2: Calibration function for shifts in z direction.

With these functions we can correct the misalignment of the isocenter position. First for the x and y directions then the z direction and another time for x and y directions. Because we have noisy data and use the median to have less influence from outliers we need multiple calls to both functions.

5.3.4 Correcting Rotations

Secondly, we have to correct the rotations. In contrast to the shift correction we haven't found a trivial algorithm, instead we needed an optimizer and a suitable objective function. Despite trying to find objective functions which are specific to each rotation the best results were achieved by measuring the mean euclidean distance between matching feature points.

This objective function is too noisy for a simple minimization with an off-the-shelf BFGS optimizer. So, to minimize this objective we developed a simple function. We evaluate the objective at multiple points and then fit a quartic function to these values. The smallest root within the bounds of the used points is the minimized parameter value (Listing 3). We will call this quartic-fitting trajectory alignment function (QUT-AF) during the rest of this paper. A few iterations with decreasing range for the input parameters sufficient for the calibration of the rotational parameters.

For our objective we included another filter for the matched features: Only features present in all images are used. This reduces the noise of the objective function.

5.3.5 Full Algorithm

The two previously described algorithms are interwoven to perform the calibration of all parameters. First we use the functions for correcting shifts, in the order xy-z-xy, with three iterations each. Then we use the minimizer for the rotations and between every iteration we do a fast shift correction with only one iteration. After all iterations of the rotation calibration we have a final correction for shift parameters. The full code of our algorithm "Feature ORiented Calibration for Arbitrary Scan Trajectories with Enhanced Reliability" (FORCASTER) is shown in listing 4.

5.3.6 Image Data

We acquired two CBCT short scans on an Artis Zeego (Siemens Healthineers, Erlangen, Germany) of a lumbar spine phantom. In the first scan a needle was inserted, the second scan had an additional large metal object and the needle position was changed slightly. An axial, saggital and coronal slice of these two images is shown in figure 5.2. We use the first CBCT image, containing only a needle, as our prior image. We will register the projections from the second CBCT scan to this image.

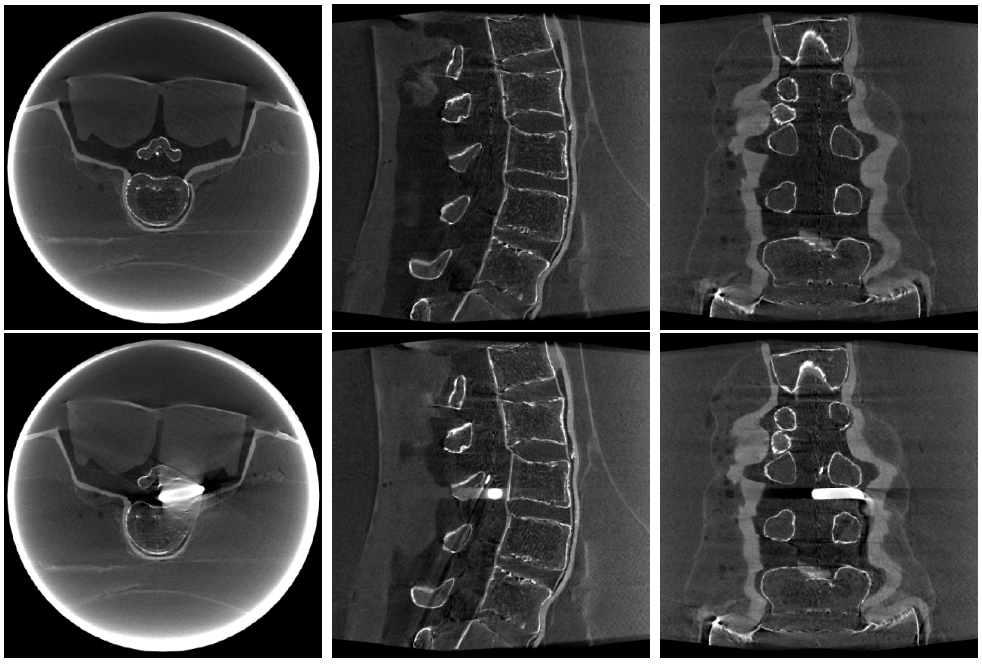


Figure 5.2: Upper Row: 1st CBCT used as prior. Bottom Row: 2nd CBCT, projections used in calibration

5.3.7 Evaluation

We evaluated FORCASTER using three experiments and compared it to state-of-the-art algorithms from literature. One of these is the BFGS minimization using the distance between matched feature points [3], the other one is BFGS minimization using the normalized gradient information as the objective function[73]. Additionally, we have two mixed algorithms where we correct the translational errors with our algorithm and then use BFGS with our feature based objective and NGI for the rotations. We also test a variant of our FORCASTER algorithm using NGI as an objective for the QUT-AF.

For the gradient used by the BFGS algorithm we use a numerical 3-step approximation and run the BFGS multiple times with diminishing step sizes (Table 5.1). For the algorithms where we mix BFGS optimization with our algorithm for correcting translations we will run a translation correction at the start and after every BFGS run.

	Parameter	1st run	2nd run	3rd run*
Our Objective	Rotations	0.25°	0.025°	
	Translations	2	1	
NGI Objective	Rotations	0.25°	0.05°	0.01°
	Translations	3	2	1

Table 5.1: Step sizes for the gradient approximations. *only for NGI objective

For each BFGS run the stop conditions are an iteration limit of 50 and a gradient norm of less than 10^{-5}.

1st Experiment

The first experiments consist of calibrating a trajectory where only the translational parameters are noisy. To create this trajectory we shifted the initial trajectory in x- and y-direction by an random amount of pixels from an uniform distribution with the bounds of -10 to +10. The zoom factor is chosen from an uniform distribution using the interval from 0.95 to 1. This disturbed trajectory is then calibrated by the different algorithms. The randomization seed is constant, so all algorithms are initialized with the same trajectory. The projections are taken from the second CBCT and the prior image is also the second CBCT.

2nd Experiment

In the second experiment we add noise to the rotational and the translational parameters. The noise for rotation parameters are sourced from an uniform distribution of -2° to +2°. Similar to the 1st experiment all calibration algorithms are initialized with the same noisy trajectory. We use the second CBCT as our prior image and calibrate the projections of the same CBCT.

3rd Experiment

Finally, in the 3rd experiment we take the projections from the second CBCT, add noise to all parameters, and then calibrate them using the 1st CBCT as our prior image.

Metrics

We use two metrics for comparison of the results. The first is the structural similarity index (SSIM)[75] evaluated an the projection images. The second is the normalized root mean square error (NRMSE) evaluated on the projections. As a third metric we segment part of the big metal object with simple thresholds and then calculate the Dice-Sørensen-Coefficient. For this we reconstruct the images using the FDK algorithm from the astra toolbox. The segmentation is performed by finding the voxel with the highest value in the rough area where the object should be and then using a threshold at half this maximum value. Afterwards a morhpological opening with a 3x3x3 kernel and a connected component analysis is used to get the segmentation.

System Specification

Our algorithms were run on a system with an Intel Core i7-4790K, 32 GB RAM, NVIDIA GeForce RTX 2070 SUPER. Due to each projection being independent from the others, all these algorithms can be easily parallelized. We use as many parallel processes as the CPU has logical cores, so 8 for the i7. We use python 3.7.6 and the packages: astra-toolbox 1.9.9.dev[74], scipy 1.6.1, skimage 0.18.3, numpy 1.17.4, opencv 4.5.1-dev.

5.4 Results

1st Experiment

The results from the first experiment, where we only had translational noise to calibrate, are in table 5.2. We have seen in this experiment, that after three iterations of each correction step no further corrections are made. So whenever we mention our translation correction algorithm it will be three iterations of x,y correction, three times z correction and three times x,y correction. The BFGS optimizer with NGI optimizer performed very poorly in all metrics. We therefore added another run where the translational noise is reduced by halve to a pixel shift of -5 to +5.

Algorithm	SSIM	NRMSE	Dice	Runtime [hh:mm]
Our Algorithm	1.00	0.0017	1.00	00:16
BFGS (Our Objective)	0.96	0.0287	0.99	05:30
BFGS (NGI Objective)	0.71	0.2456	0.82	07:20
BFGS (NGI Objective)*	0.91	0.0794	0.97	05:13

Table 5.2: Results for the 1st experiment. *reduced translational noise

2nd Experiment

In table 5.3 are the results for the 2nd experiment. Here we can see that our minimizer performs equally good with our objective and the NGI objective. Furthermore we can see in all three metrics, that our minimizer performs better than the algorithms based on the BFGS optimizer.

Algorithm	SSIM	NRMSE	Dice	Runtime [hh:mm]
FORCASTER	0.97	0.0237	0.99	03:14
FORCASTER (NGI Objective)	0.96	0.0322	0.99	01:16
Mixed BFGS (Our Objective)	0.91	0.0559	0.98	06:47
Mixed BFGS (NGI Objective)	0.98	0.0224	0.99	02:47
BFGS (Our Objective)	0.89	0.0606	0.97	13:01
BFGS (NGI Objective)	0.67	0.2491	0.94	11:32
BFGS (NGI Objective)*	0.90	0.0644	0.96	13:43

Table 5.3: Results for the 2st experiment. *: reduced translational noise

3rd Experiment
In the 3rd experiment the images were calibrated on the 1st CBCT and compared to the 2nd CBCT, therefore all metrics (Table 5.4) are slightly worse than those of the 2nd experiment. Still, it shows similar results. Our algorithm is on par with the one from literature. BFGS using our feature distance objective performs worse than the NGI objective for calibrating the rotations but better in relation to the translations. In Figure 5.4 are images of the reconstructions for the calibration done by our minimizer and the Mixed BFGS with NGI objective. The selected slices are the same as in Fig. 5.2.

The runtime for the full BFGS algorithms are, as expected, very high. The NGI objective always performed much faster than our objective unrelated to the underlaying optimizer. The fastest algorithm was the one using our translation correction using feature matching and the QUT-AF minimizer with the NGI objective. This algorithm needed only 10% of the time the state-of-the-art algorithm of BFGS+NGI took. In figure 5.3 the NRMSE is plotted over the iterations. The mixed and FORCASTER algorithms all start with the same steps, therfore they have the same steep decline at the first iteration. The total number of iterations for every algorithm can also be seen in the table 5.4. It shows, that the total number of iterations is only slightly decreased when moving the translation correction out of the BFGS minimization.

5.5 Discussion

We have shown, that FORCASTER can achieve an accuracy that is comparable to the state of the art for the problem of arbitrary task-basked trajectory calibration. Even if we use a prior image, that has significant changes in contrast to the projections, we can successfully calibrate the trajectory. We deem this to be an important property for online calibration algorithms if task-based trajectories should be integrated into clinical practice.

Furthermore, our results show, that it is possible to separate the optimization of rotations and translations without an impact to the calibration performance. The mixed BFGS algorithms had a slightly lower NRMSE than the full BFGS algorithms. The FORCASTER algorithm, going one step further

Algorithm	SSIM	NRMSE	Dice	Runtime [hh:mm]	Iterations **
Prior Difference	0.86	0.3380	0.88	Ø	
FORCASTER	0.83	0.3390	0.86	02:59	9
FORCASTER (NGI Objective)	0.84	0.3390	0.87	01:03	9
Mixed BFGS (Our Objective)	0.79	0.3406	0.87	09:21	24
Mixed BFGS (NGI Objective)	0.85	0.3387	0.88	03:47	120
BFGS (Our Objective)	0.79	0.3410	0.87	20:54	31
BFGS (NGI Objective)	0.63	0.3743	0.82	10:23	124
BFGS (NGI Objective)*	0.79	0.3441	0.87	10:36	69

Table 5.4: Results for the 3st experiment and the difference of the prior image to the calibrated image. *: reduced translational noise; **: iterations of FORCASTER and BFGS not comparable due to different minimzing algorithms.

and optimizing the parameters serially, but with six loops, also achieved a lower NRMSE than a full BFGS.

One obvious problem with feature matching are wrong matches. Most mismatched features are eliminated by Lowe's ratio check. From the remaining matches 5% are still incorrect. These are removed by the outlier and double match filters. In figure 5.5 the matches discarded by the filtering steps and the remaining matches are displayed for one projection.

Using the distance between features for the objective function given to a BFGS minimizer gave poor results when calibrating rotations. This is probably due to the noisiness of this objective. Our approach with QUT-AF of fitting a quartic function is more robust than the 2-point or 3-point numerical derivative used for BFGS, but calculating this derivative with more points increases the computational cost and thus further slows down the minimization.

Even though our algorithm is, with a runtime of 3 hours, faster than a BFGS minimization (10h), its long runtime is still a problem that needs to be solved. One approach could be by leveraging the pair-wise independence of each projection and use more parallel processing. This could be done on a high-performance cluster or on a GPU with enough memory. Alternatively, developing a faster objective function for the calibration of the rotations might improve processing speed. Here our results show, that the mixed approach of a feature-based objective for translations and the NGI objective for rotations is three times faster than our algorithm.

A way to estimate the needed rotation from two simulated projections, similar to how we estimated the needed translation, would speed up the calibration immensely. Removing the translational minimization from the BFGS optimizer saved more than 10 hours of computation. Here the matched features give a plethora of information on what changes between slightly rotated projections which is hopefully enough for a simple and fast algorithm.

Also the feature-based approach showed a high tolerance to wrongly guessed

start parameters. An adaption to further remove the need for an accurate initial guess would help with images that do not have attached positions. This is the case for continuous acquisition on an Artis Zeego System. Only the starting position is exported to the DICOM but not the positions of all subsequent frames. In conclusion we have presented a viable approach to calibration which uses techniques that are in this context not well explored but are interesting for further research.

In conclusion we have shown, that feature based calibration is a contender to the state of the art calibration algorithms. With equal quality, but shorter runtime and higher robustness to wrong start parameters.

Supplementary information

If your article has accompanying supplementary file/s please state so here.

Please refer to Journal-level guidance for any specific requirements.

Acknowledgments

This research project is partly supported of the Research Campus M2OLIE funded by the German Federal Ministry of Education and Research (BMBF) within the Framework “Forschungscampus: public-private partnership for Innovations” under the funding code 13GW0388A.

Declarations

The authors declare no conflict of interest.

```
def correctRotation(cur, axis, width, count):
     s = linspace(-width, 0, count)+[0]+linspace(0, width, count)

    dvec  = [applyRotation(cur,  , axis) for  in  s]
    projs = ForwardProjections(dvec)

    for i in 1 ..| s|:
        Ψ_i, Ω_i = FindFeatures(projs[i], real_acquisitions)

    # Only use points found in all projections
    shared_points = ⋂_i Ψ^-1(Ω_i)
    Ω_i = Ψ_i(shared_points)

    value_i  = [‖ p⃗ − Ψ_i(p⃗) ‖_2 for p⃗ in Ω_i]
    values  = [mean(value_i) for i in 1 ..| s|]

    values = (values-min(values)) / (max(values)-min(values))
    # fit to quartic function (p_4 * x^4 + p_3 * x^3 + p_2 * x^2 + p_1 * x + p_0)
    poly  = numpy.polyfit( s, values, 4)
    # find roots of 1st derivative of polynom
    dpoly = numpy.polyder(poly)
    roots = real(numpy.roots(dpoly))
    # ignore roots outside of our input area
    roots = [ r for r in roots if -width   r   width ]
    if |r| == 0:
    # if no roots are found use median of
    # the five smallest objective values
        midpoints = argsort(values)[:5]
        min_  = median( s[midpoints])
    else:
        min_root = argmin(numpy.polyval(poly, roots))
        min_  = real(roots[min_r])

    cur = applyRotation(cur, min_ , axis)

    return cur
```

Listing 3: Our minimizer (QUT-AF) for calibration of rotations.

```
# cur are three vectors: the translation,
# detector orientation in x, and y direction
iterations = 3 # correct shifts with 3 iterations each
cur = correctXY(cur, iterations)
cur = correctZ(cur, iterations)
cur = correctXY(cur, iterations)

for width,count in [(2,9),(1.5,9),(1,9),(0.5,9),(0.25,9),(0.1,9)]:
# (2,9) means: 9 points are used in range from [-2° to 0°)
#            + 0° + 9 points in range (0° to 2°] ->
#  -2, -1.77, -1.55, -1.33, -1.11, -0.88, -0.66, -0.44, -0.22,
#   0,  0.22,  0.44,  0.66,  0.88,  1.11,  1.33,  1.55,  1.77, 2
    # to fit the quartic function
      = correctRotation(cur, 0, width, count) # x-axis
      = correctRotation(cur, 1, width, count) # y-axis
      = correctRotation(cur, 2, width, count) # z-axis
    cur = applyRotation(cur,  ,  ,  ) # apply rotations
    iterations = 1 # correct shifts between iterations
                   # but only with 1 iteration to save time
    cur = correctXY(cur, iterations)
    cur = correctZ(cur, iterations)
    cur = correctXY(cur, iterations)

iterations = 3 # final correction of shifts
cur = correctXY(cur, iterations)
cur = correctZ(cur, iterations)
cur = correctXY(cur, iterations)
return cur
```

Listing 4: Full calibration algorithm: Feature ORiented Calibration for Arbitrary Scan Trajectories with Enhanced Reliability (FORCASTER)

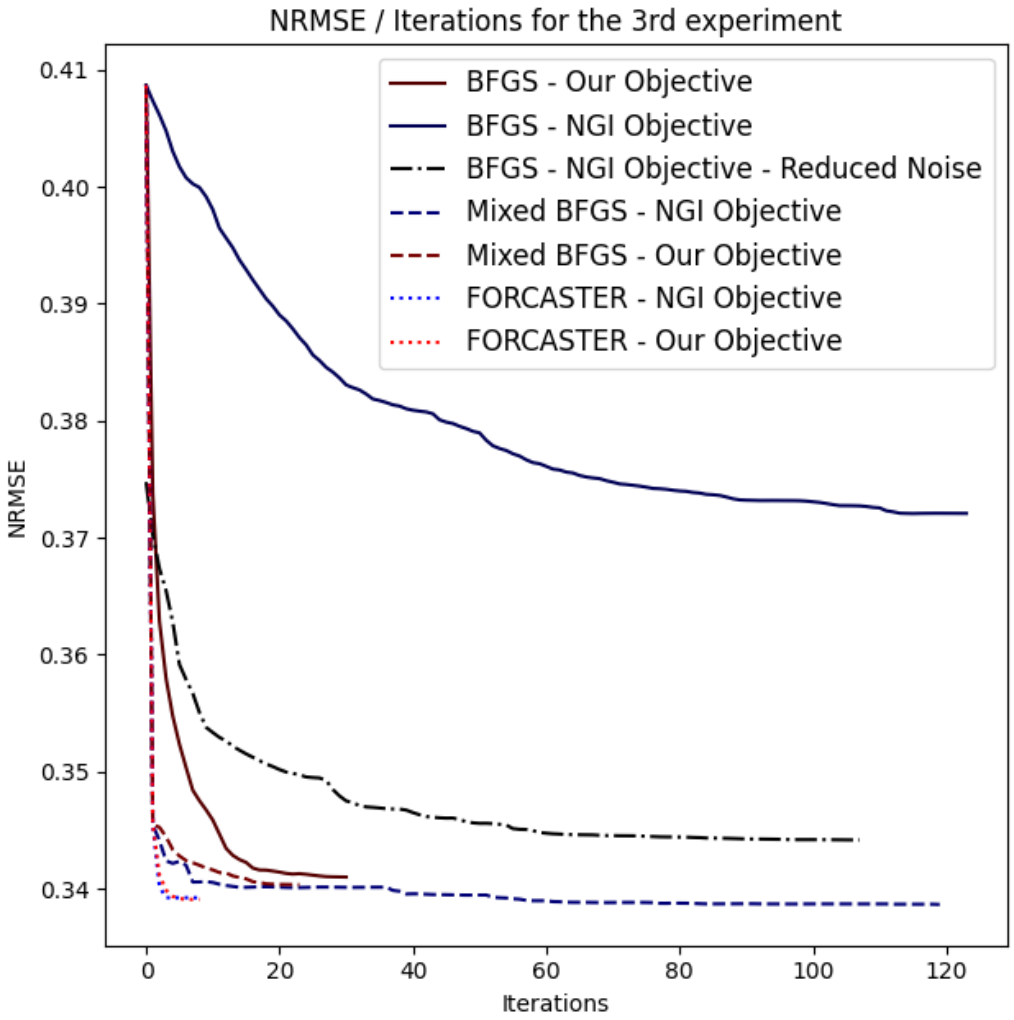


Figure 5.3: The NRMSE plotted over the iterations for the algorithms and data used in the 3rd experiment.

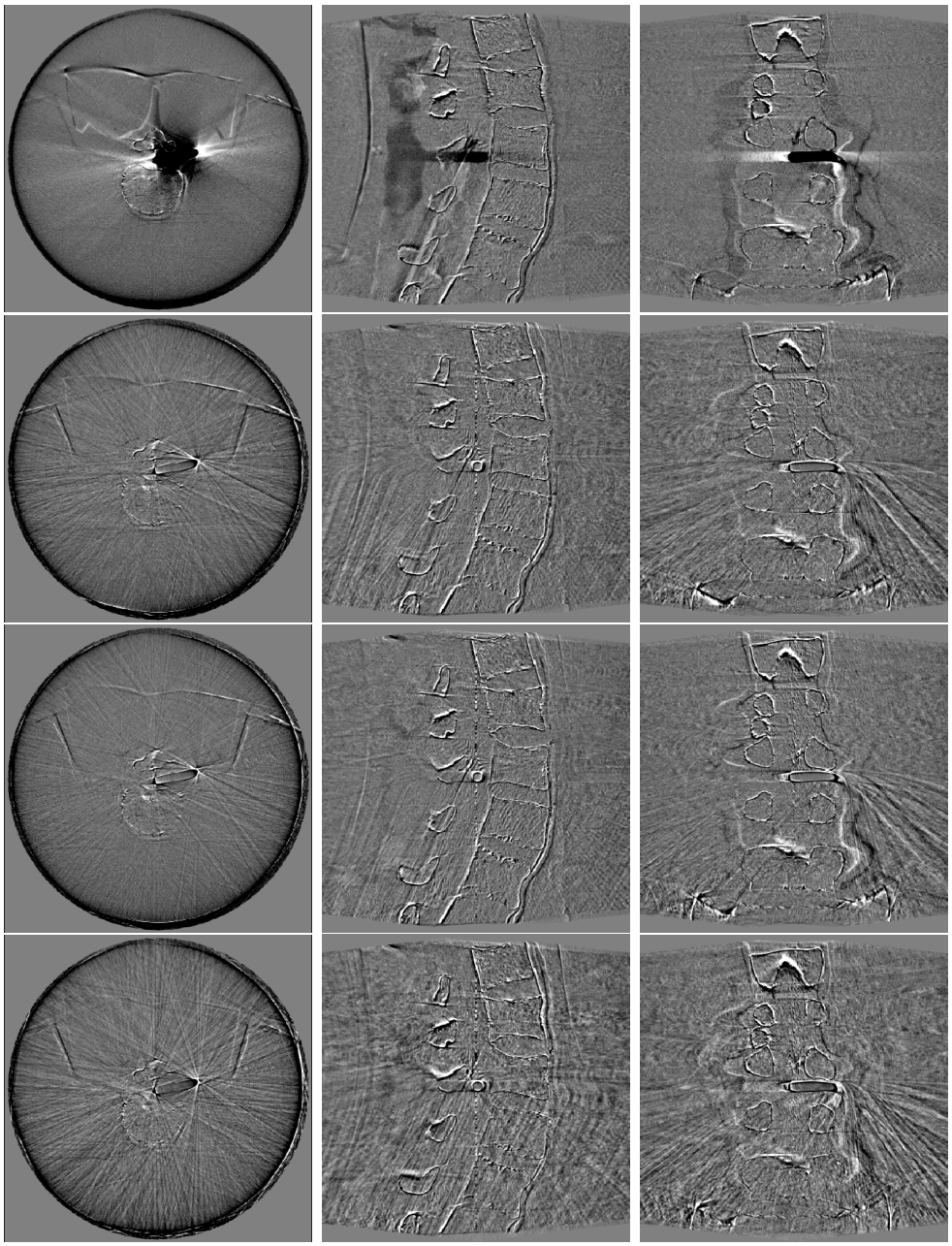


Figure 5.4: Error between reconstructions from Experiment 3 and the actual image. First row: error of the prior image. Second row: error of FORCASTER. Third row: error of mixed BFGS with NGI. Bottom row: error of BFGS with NGI (reduced translational noise)

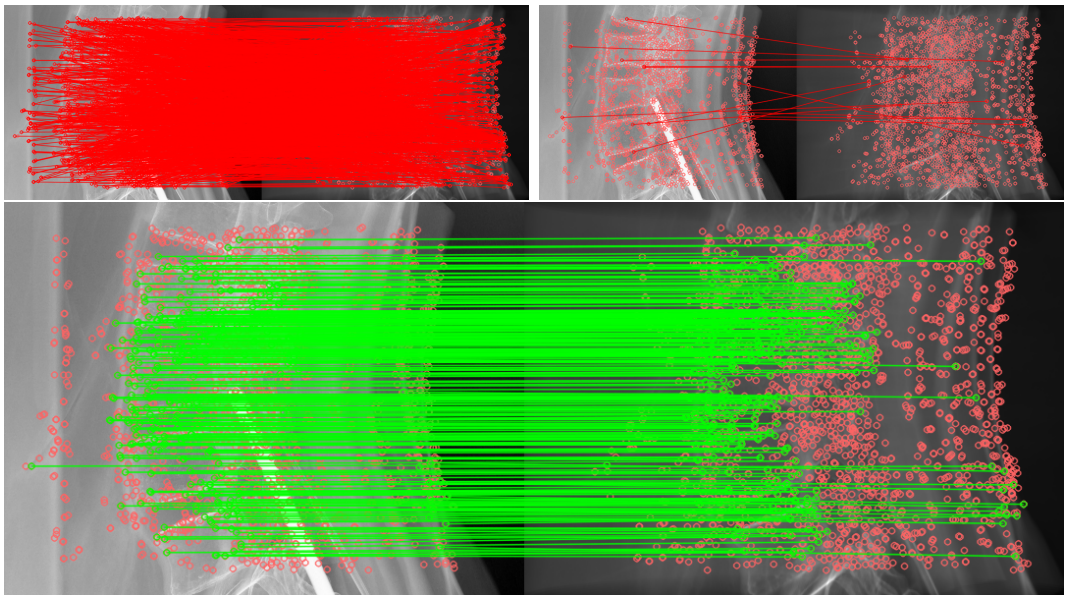


Figure 5.5: Top Left: Matches discarded by Lowe's ratio check. Top Right: Matches discarded by double match and outlier filter. Bottom: Remaining matches after filtering.

6 Feature-oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST

Christian Tönnes[1,2] and Frank G. Zöllner[1,2]
[1]Computer Assisted Clinical Medicine, Medical Faculty Mannheim, Heidelberg University, 68167 Mannheim, Germany
[2]Mannheim Institute for Intelligent Systems in Medicine, Medical Faculty Mannheim, Heidelberg University, 68167 Mannheim, Germany

6.1 Abstract

Background: For the reconstruction of Cone-Beam CT volumes, the exact position of each projection is needed; however, in some situations, this information is missing. **Purpose:** The development of a self-calibration algorithm for arbitrary CBCT trajectories that does not need initial positions. **Methods:** Projections are simulated in a spherical grid around the center of rotation. Through using feature detection and matching, an acquired projection is compared to each simulated image in this grid. The position with the most matched features was used as a starting point for a fine calibration with a state-of-the-art algorithm. **Evaluation:** This approach is compared with the calibration of nearly correct starting positions when using FORCASTER and CMA-ES minimization with a normalized gradient information (NGI) objective function. The comparison metrics were the normalized root mean squared error, structural similarity index, and the dice coefficient, which were evaluated on the segmentation of a metal object. **Results:** The parameter estimation for a regular Cone-Beam CT with a 496 projection took 1:26 h with the following metric values: NRMSE = 0.0669; SSIM = 0.992; NGI = 0.75; and Dice = 0.96. FORCASTER with parameter estimation took 3:28 h with the following metrics: NRMSE = 0.0190; SSIM = 0.999; NGI = 0.92; and Dice = 0.99. CMA-ES with parameter estimation took 5:39 h with the following metrics: NRMSE = 0.0037; SSIM = 1.0; NGI = 0.98; and Dice = 1.0. **Conclusions:** The proposed algorithm can determine the parameters of the projection orientations for arbitrary trajectories with enough accuracy to reconstruct a 3D volume with low errors.

6.2 Introduction

To reconstruct a Cone-Beam Computed Tomography image, several X-ray images are required and must be input into a reconstruction algorithm. This

algorithm must know at which position and orientation each X-ray image was made. Several methods have been developed to solve this problem, and they come in two broad categories, offline and online calibration.

Offline calibration uses phantoms with markers, which are often small metal beads that are imaged for each projection where the position and orientation can be calculated [72, 71, 76, 77, 78, 79, 80]. The phantom is imaged with a trajectory, and the correction factors are calculated. Then, these correction factors are used when reconstructing the images from this trajectory. This requires a dedicated run of the trajectory with the phantom.

Online calibration uses the acquired projections and then uses prior information about the imaged object [81, 3, 1, 82], or it minimizes a cost function that is defined on the reconstructed image [69, 83, 84, 85]. A review of current approaches was published by Hatamikia et al. [66]. This approach does not need a dedicated phantom nor an extra run of the trajectory for the calibration, but it does have other constraints. Some use a prior image on which the projections are registered when using 2D–3D registration algorithms, others require the trajectory to have a specific, often circular, shape.

Using a 2D–3D registration of the acquired projections on a prior image allows for the calibration of fully arbitrary trajectories [73], and the registration is faster if the initial parameters are close to the actual parameters. The position reported by the CBCT system is usually close enough for a quick and good calibration with state-of-the-art algorithms. But, if this information is not available (e.g., portable C-Arms, continuous acquisition/fluoroscopy mode of the Artis Zeego), other means of obtaining these initial parameters are required.

One option is to track the C-Arm externally with inertia sensors [86, 87]. These sensors can be attached to the C-Arm and, after calibrating the sensors, they track the inertia in all three dimensions. With this inertia data and a known starting position, the movement of the C-Arm can be calculated.

Also possible is the use of 3D cameras [88, 89]. Here, the position of the C-Arm is observed by tracking optical markers through using either a camera attached to the X-ray source or two external cameras.

This paper presents another option that uses a prior image to simulate forward projections from different angles, and then uses feature matching to find the one that fits bests for each acquired projection. In this way, the parameters for each projection can be estimated.

6.3 Materials and Methods

6.3.1 Projection and Optimization Parameters

The algorithm uses an intrinsic coordinate system that is bound to the detector plane. The three vectors to define the position and orientation of the acquired image are shown in Figure 6.1. The vector $\vec{d}$ points from the middle of the

detector to the source, and the length is the distance between the detector and the source. The vectors $\vec{u}$ and $\vec{v}$ describe the direction and spacing of the detector elements, respectively, and they point from the center of one detector element to the center of the left/top neighbor. The size of each vector is the spacing between elements.

The three unit vectors $\vec{x}$, $\vec{y}$, and $\vec{z}$ that make up the coordinate system are parallel to the vectors $\vec{u}$, $\vec{v}$ and $\vec{d}$, respectively, and the point of origin is the isocenter, which is the center of the CT image.

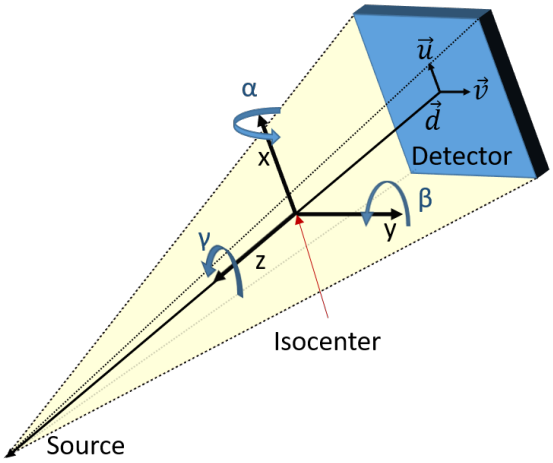


Figure 6.1: Overview of the coordinate system, parameters, and the degrees of freedom [1].

These vectors are also used for all movements and rotations. There are three rotations, one around each of the vectors x, y, and z, and three translations that are also along these vectors.

6.3.2 Feature Points Matching

The algorithm depends on feature points; these are found within each image through using the AKAZE [90] algorithm. The parameters used for AKAZE were as follows—threshold: 0.0005, four Octaves, and five Octave Layers. AKAZE also generates a description vector for each feature point, and these descriptors can be used to compare to points through using the Hamming distance. To find the matching features between images, the Hamming distances between all feature descriptors of one image to all descriptors from the other image are calculated. Then, for every feature in the calibration image, the features with the lowest Hamming distance d_1, and the one with the second lowest distance d_2 are selected. On these two distances, Lowe's ratio test [8] is applied. This compares the distance with a ratio r, and this is performed in order to check that the smaller distance is much smaller than the second best

match ($d_1 < r * d_2$). When the test succeeds, the feature point with the smaller distance is used to form a matching pair of feature points. If the distances do not comply with this test, no matching pair was found.

Afterward, all found pairs are filtered by finding the ones that match to the same feature point and those are removed.

The last step involves discarding pairs where the Euclidean distance between the points in the matched pairs is more than one standard deviation from the mean distance of all pairs.

6.3.3 Algorithm

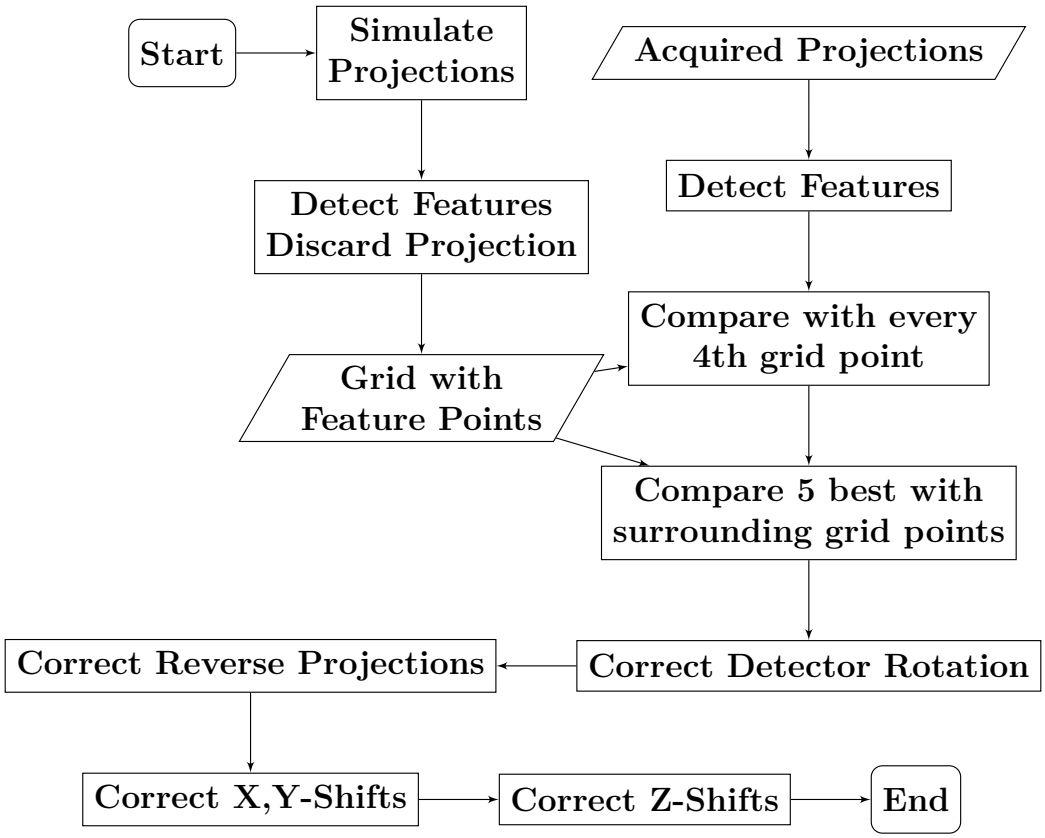


Figure 6.2: Pipeline of the estimation algorithm. Rectangles are processes, trapezoids store data.

An overview of the estimation process is in Figure 6.2, and this will be explained in more detail in the following section.

The algorithm is initiated with the acquired images, a prior CT, and geometry information about the size of the detector, as well as the distance to the source and to the isocenter. The first step of the algorithm is to generate simulated projections in a regular grid around the center of the prior CT image. The grid has 95 points each for the rotations around the x- and y-axis,

which results in a grid with a spacing of 3.8°. Further, three different detector rotations are used, 0°, 120°, and 240°. On each of these images, the AKAZE algorithm detects features and extracts the feature descriptors. The simulated projections are then deleted, and only the feature points, descriptors, and projection rotations (α, β, and γ) are saved. They are calculated once and then used for all further calibration.

To find the approximate position of an acquired image, the algorithm first detects the features. They are then matched with each set of features from the simulated grid projections and the matched feature points are counted. To save time, the algorithm operates along this grid with a step size of four, and it then selects the five grid points with the most matched feature points. Next, the grid points surrounding these five points are compared to the acquired image in the same way. For the grid point that has the most matched pairs, the projection rotations are returned. These are the current approximation of the rotations $\hat{\alpha}_1$, $\hat{\beta}_1$, and $\hat{\gamma}_1$ (Fig. 6.3 (a)).

This selected grid position is most likely the closest to the target position, but it can also be a projection from the opposite side. These projections, simulated from the wrong side, will be corrected later.

Before that, the detector rotation is approximated. The average value of the feature point coordinates is used as the center point; this is performed separately for the real and simulated image. Then, the coordinates of the feature points are converted to polar coordinates by using the averaged center point as the zero point. The angle for each feature point in the simulated image is subtracted from the angle of the matching feature point in the real image. The median of these differences is the new approximated detector rotation $\hat{\gamma}_2$. Listing 5 shows this in pseudocode.

A similar approach is taken to correct the projections taken from the opposite direction. Four projections are simulated with different rotation parameters, as well as the approximate rotations with a detector rotation of 180° ($\hat{\gamma}_2 + \pi$). This is conducted from the opposite side that rotates around the x-axis ($\hat{\alpha}_1 + \pi$) and around the y-axis ($\hat{\beta}_1 + \pi$). For these four projections, features are detected and matched, and the projection with the lowest mean Euclidean distance between the matched points is then used.

The result of this first step is a rough calibration. As such, in the next step, this rough calibration is further refined. The translational misalignment is corrected using the method described by Tönnes et al. [1]. The median Euclidean distance between the matching points is used to move the projection in the x and y directions. The z-translation is corrected by calculating the distance between the feature points within each image, and by then dividing the distances of one image by those of the other image results in the zoom factor. This ratio and the distance between the source and the isocenter are multiplied to give the new distance.

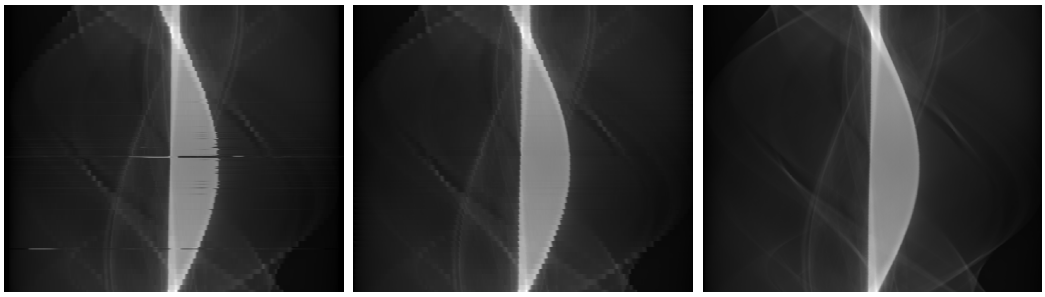
After correcting the translations along the x-, y- and z-axis, the previously described procedure that is used to correct the detector rotation is applied

```
def approximate_detector_rotation(current_parameters):
    # simulate projection and track features
    simulated_projection = ForwardProjection(current_parameters)
    simulated_feature_points =
     ↪  trackFeatures(simultaded_projection, real_projection)
    # calculate center point
    simulated_mid = mean(real_feature_points, axis=0)
    real_mid = mean(simulated_feature_points, axis=0)
    sim_points = simulated_feature_points - simulated_mid
    real_points = real_feature_points - real_mid
    # calculate angle
    angles = (arctan2(sim_points[:,0], sim_points[:,1])
              -arctan2(real_points[:,0], real_points[:,1])) *
               ↪  180.0/PI
    angles[angle<-180] += 360
    angles[angle>180] -= 360
    detector_angle = median(angles)
    # test in which direction to rotate
    proj = ForwardProjection(applyRotation(current_parameters,
     ↪  0,0,-detector_angle))
    points = trackFeatures(proj, real_projection)
    diffn = points - real_feature_points
    proj = ForwardProjection(applyRotation(current_parameters,
     ↪  0,0,+detector_angle))
    points = trackFeatures(proj, real_projection)
    diffp = points - real_feature_points
    if sum( abs(diffn) ) < sum( abs(diffp) ):
        return -detector_angle
    else:
        return detector_angle
```

Listing 5: Approximation function for the detector rotation.

once more (Fig. 6.3 (b)).

The resulting parameters can then be used to run a state-of-the-art calibration algorithm and fully calibrate the trajectory (Fig. 6.3 (c)).

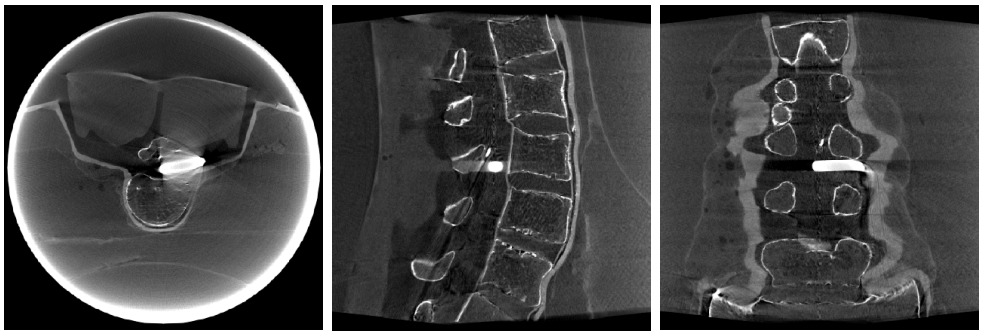


(a) (b) (c)

Figure 6.3: The sinograms for the different steps in the algorithm. From left to right is as follows: (**a**) coarse estimate, (**b**) refined estimate, and (**c**) the refined estimate with FORCASTER when using the NGI objective.

6.3.4 Image Data

In this paper, the data from Tönnes et al. [1] were used, which were obtained from a CT scan of a lumbal spine phantom with an inserted metal object. The reconstructions can be seen in Figure 6.4.



(a) (b) (c)

Figure 6.4: CBCT used as a prior image. (**a**) Transversal slice. (**b**) Sagittal slice. (**c**) Coronal slice.

Furthermore, a sinusoidal trajectory, acquired shortly after the abovementioned CT scan, is used. The phantom is not moved in between. The sinusoidal trajectory is acquired in a step-and-shoot mode, which means moving the C-Arm to each of the 161 positions on this trajectory, and then acquiring a single X-ray image with the standard protocol called “P16_DR_L” at 70 keV and with the mAs controlled by the Artis Zeego System.

The third trajectory is acquired using the continuous acquisition mode and by moving the C-Arm during acquisition. This one has the problem that was mentioned in the introduction in that it does not contain positional information for the individual frames. This trajectory is an arc around the object tilted by 28°, with 70 keV and 30 frames per s. The exposure time and tube current are managed by the Artis Zeego System; furthermore, the average pulse width is 3.5 ms, with an average current of 35 mA. It consists of 666 individual projections.

All three sinograms are shown in Figure 6.5. Since the sinograms are three-dimensional, only two slices are shown, and each of them is cut through the center of all individual projections, both horizontally and vertically.

6.3.5 Evaluation

To evaluate the quality of the estimator, the estimated parameters were used as inputs for the FORCASTER [1] algorithm and the algorithm by Oudah et al. [73] (which uses a CMA-ES minimizer with the normalized gradient information (NGI) as the objective function).

The calibrated trajectories are then reconstructed with the FDK algorithm, which is part of the astra toolbox [74]. The images become cropped to the field of view, and no further post-processing is performed. The calibrated parameters are also used to generate a forward projection using the prior image; this simulated sinogram is compared to the simulated forward projections of the state-of-the-art calibration algorithm. The continuous acquisition sinogram is compared to the acquired data since there are no correctly calibrated parameters.

Metrics

The structural similarity index (SSIM) [75] (Equation (1)) is the normalized gradient information (NGI) [91], and the normalized root mean squared error (NRMSE) (Equation (2)) are evaluated on the projections that are simulated with the parameters from the calibrated trajectory in comparison to the forward projections of the reference calibration.

$$SSIM(x,y) = \frac{2\mu_x\mu_y(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{6.1}$$

$$NRMSE(x,y) = \frac{RMSE(x,y)}{||x||_F} \tag{6.2}$$

$$RMSE(x,y) = \sqrt{\frac{1}{N}\sum_i^N (x_i - y_i)^2} \tag{6.3}$$

In this equation, μ_x is the mean value of x; σ_x the standard deviation; c_1 and c_2 are the constants; N : This is changed to be italics format to keep

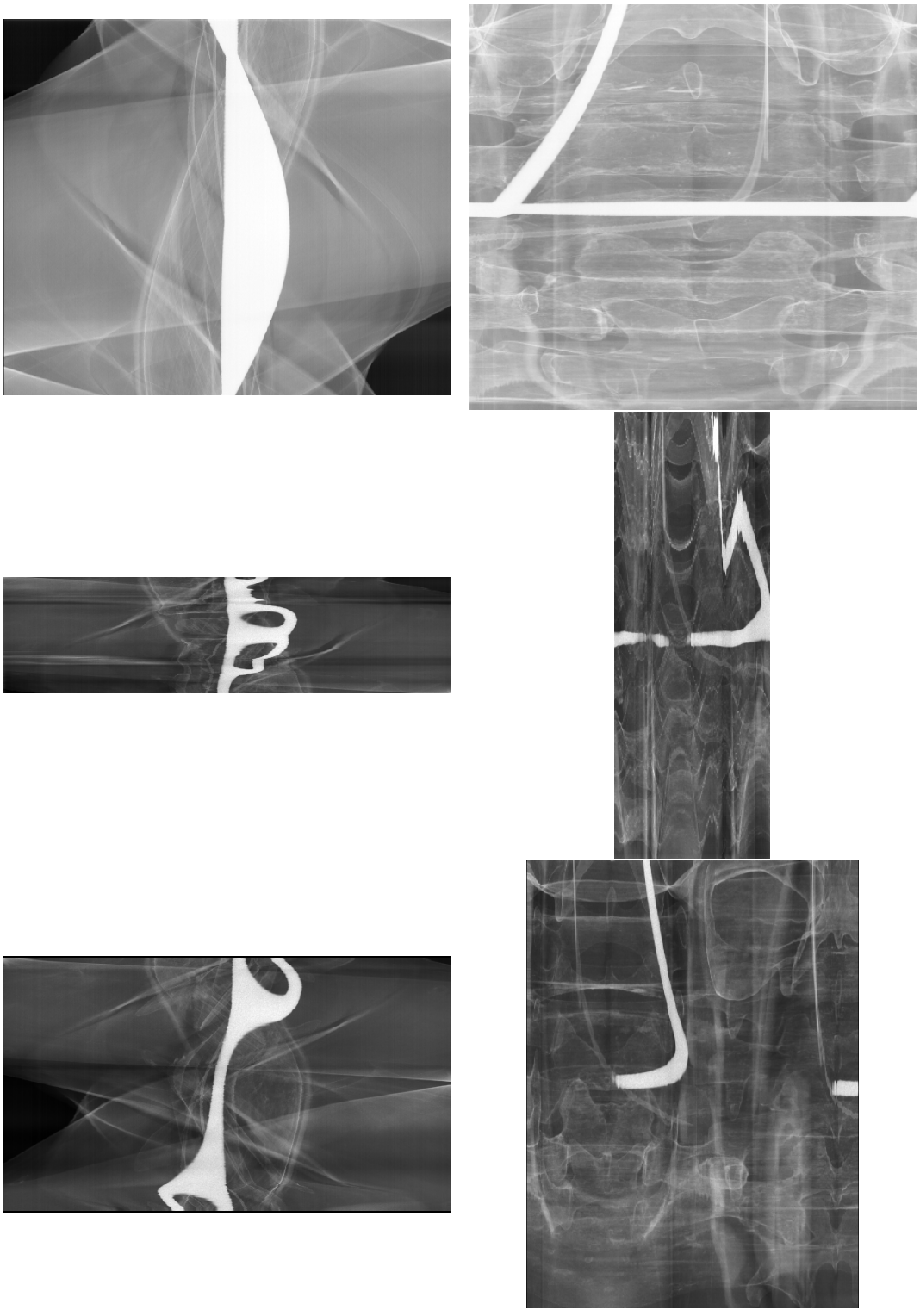


Figure 6.5: Two slices through the acquired three-dimensional sinograms. The top row is the standard CBCT, the middle row is the sinusoidal trajectory, and the bottom row is a continuous acquisition of a circular arc of 400°.

consistent with the equation, please confirm. Same below. is the number of voxels; and $||x||_F$ is the Frobenius norm of x.

The reconstructions obtained after calibration and cropping to the point of view are also compared through using the same metrics. Additionally, the vertical part of the large metal object in the center of the phantom is segmented, and the Dice coefficient on the segmentations is then calculated. All metrics are applied to each 2D slice of the images and then averaged.

System Specifications

The algorithms were run on a system with an AMD Ryzen 9 7900X CPU, 128 GB RAM, and NVIDIA GeForce RTX 2070 SUPER. Due to each projection being independent of the others, the algorithm can be easily parallelized. In this paper, ten parallel processes were used. Python version 3.9.9 the following packages was used: astra-toolbox 2.0 [74], scipy 1.7.3, skimage 0.19, numpy 1.21.4, and opencv 4.5.4.

6.4 Results

The results obtained from comparing the sinograms are in Table 6.1. The similarity index and NGI are low for the coarse estimate, but they increased significantly after refining. When using the FORCASTER algorithm with the estimated positions, the SSIM and NRMSE were comparable to the ones reported by Tönnes et al. [1].

Similar to the results from the sinograms, the metrics evaluated on the reconstructions, shown in Table 6.2, showed a significant improvement in the refined estimate over the coarse one. Here, the NRMSE and Dice functions, after estimating and applying FORCASTER, were also comparable to the ones previously reported. There was no significant difference ($p = 0.56$) between the Dice values of the two reconstructions when using the state-of-the-art algorithms and refined estimates; however, the other metrics—NGI, SSIM, and NRMSE—were significantly better for the CMA-ES calibration.

The reconstructed images are shown in Figure 6.6. The top row contains the reconstruction after only performing the rough estimate. There were obvious artifacts visible, which can be seen in the left column. These artifacts are regularly spaced and rotated around the center in the plane of the acquisition trajectory. Several edges were reconstructed twice with a slight offset. In the second row, the refined estimate was used for reconstruction. The radial artifacts are still there, but the double edges are now consolidated. Finally, the reconstruction without these radial artifacts is in the bottom row.

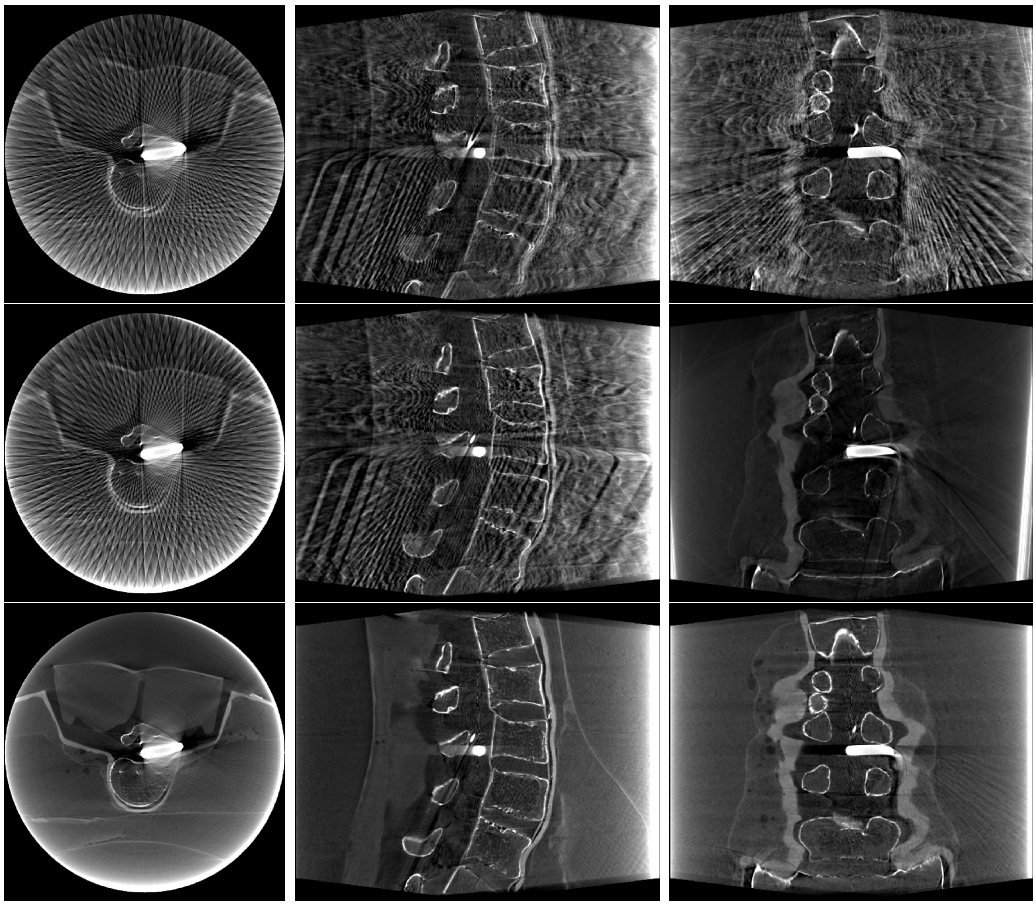


Figure 6.6: Reconstructions of the different steps of the algorithms. From top to bottom is as follows: Coarse estimate, Refined estimate and Refined estimate + CMA-ES when using the NGI objective.

Algorithm	Runtime [hh:mm]	NGI	SSIM	NRMSE
Coarse Estimate	01:06	0.397 *	0.944 *	0.2327 *
Refined Estimate	01:26	0.753	0.992	0.0669
Est. + FORCASTER	03:28	0.919 *	0.999 *	0.0190 *
Est. + CMA-ES & NGI	05:39	0.983 *	1.000 *	0.0037 *

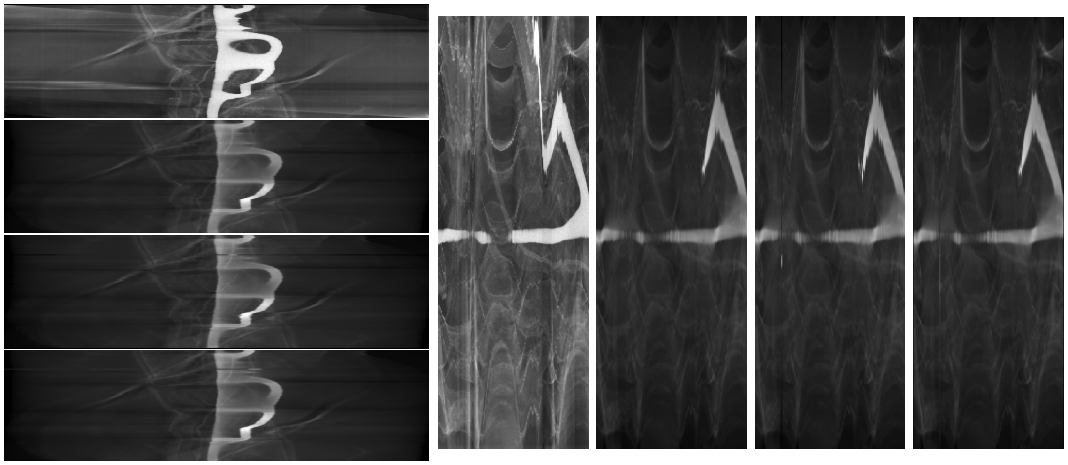
Table 6.1: Results for the metrics evaluation on the CBCT sinogram. The two bottom rows used the refined estimate as the input for the FORCASTER algorithm, and this was achieved once with the NGI objective and once with the objective when based on feature points. The values marked with an * had significant ($p < 0.05$) changes compared to the refined estimate.

Algorithm	NGI	SSIM	NRMSE	Dice †
Coarse estimate	0.372 *	0.290 *	0.7898 *	0.60 *
Refined estimate	0.525	0.360	0.5485	0.96
Est. + FORCASTER	0.766 *	0.677 *	0.1627 *	0.99 *
Est. + CMA-ES & NGI	0.944 *	0.979 *	0.0295 *	1.00 *

Table 6.2: Results for the evaluation of the CBCT reconstructions. The dice † : Please consider replacing this symbol with another one (e.g., **) to distinguish from *. This applies also to Table 5. function was evaluated on a segmentation of the large metal object in the phantom. Values marked with an * had significant ($p < 0.05$) changes compared to the refined estimate.

Figure 6.7 shows the sinograms for the sinusoidal trajectory. The calibrated sinograms are the forward projections of the prior CT image. Because this image does not contain the complete object, these two sinograms are missing structures in comparison to the acquired images. One easily spotted difference is in column (b) in the top right quadrant, which is missing a high-contrast object. In the sinogram obtained with estimated starting parameters, i.e., in the bottom row in column (a) and the right image in column (b), there is a distortion visible at the top (column (a)) and left edge (column (b)). This is more pronounced in the sinograms that were calibrated with FORCASTER than the ones calibrated with CMA-ES. This is a projection where the algorithm did not estimate the starting parameters close to the correct ones, and where the calibration did not succeed. This projection is shown in Figure 6.8 alongside the acquired image and the projection from the correctly calibrated parameters.

The values obtained from evaluating the metrics are shown in Table 6.3. Similar to the calibration of the CBCT trajectory, the metrics improved with every step. The final calibration produced lower results than the CBCT calibration, which is in line with the distortions visible in the sinogram. The runtime was much shorter because the sinusoidal trajectory contained only 161



(a) (b)

Figure 6.7: Two slices out of the sinogram of the sinusoidal trajectory from the acquired data. The FORCASTER calibration used nearly accurate starting parameters. The calibration with refined estimates and FORCASTER, as well as the refined estimates with CMA-ES and NGI objective are shown. The left column (**a**) is ordered from top to bottom as follows: acquired data, simulated projections from the calibration with starting parameters, simulated projections from the FORCASTER calibration with estimated starting parameters, and CMA-ES calibration with the estimated parameters. The right columns (**b**) have the same order, also from left to right.

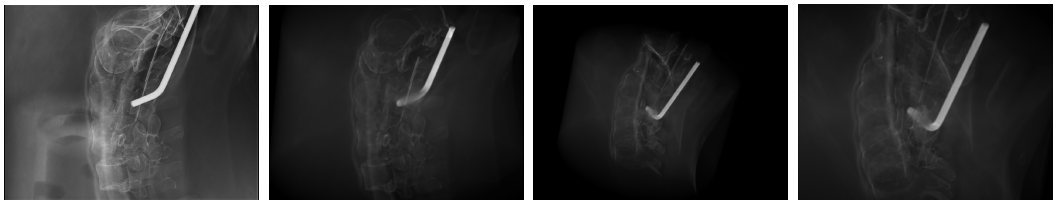


Figure 6.8: Example projections at a point where the parameter estimation fails. From left to right is as follows: acquired projection, simulated projection from the calibration with starting parameters, simulated projection from the calibration with estimated starting parameters and FORCASTER, and the projections from estimated parameters with CMA-ES calibration.

projections in comparison to the 496 contained in the CBCT trajectory.

The results for the continuous arc trajectory are shown in Table 6.4. Because there was no correct calibration for this trajectory (since there were no starting parameters for the individual projections), the metrics were not evaluated on the simulated projections but were instead evaluated on the acquired images. The metric values were much lower there than for the ones for the two

Algorithm	Runtime [hh:mm]	NGI	SSIM	NRMSE
Coarse estimate	00:24	0.259 *	0.885 *	0.4513 *
Refined estimate	00:27	0.662	0.992	0.0851
Est. + FORCASTER	00:54	0.836*	0.997	0.0369 *
Est. + CMA-ES & NGI	01:50	0.955 *	0.999 *	0.0106 *

Table 6.3: Results of the evaluation on the sinusoidal trajectory. Values marked with an * had significant ($p < 0.05$) changes compared to the refined estimate.

previous trajectories. Still, they significantly improved with each step, and there were no significant differences in the SSIM ($p = 0.5$) and NRMSE ($p = 0.9$) metrics between the two state-of-the-art algorithms.

For comparison, the calibrations for the other two trajectories on the acquired images were evaluated and are included in the results located in the lower part of the table.

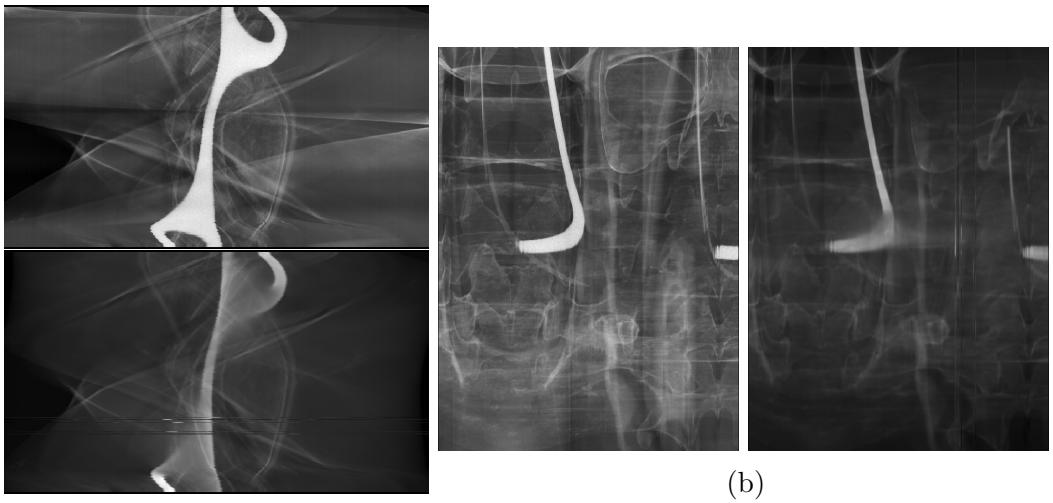
Algorithm	Runtime [hh:mm]	NGI	SSIM	NRMSE
Coarse estimate	01:26	0.123*	0.532*	0.7019*
Refined estimate	01:31	0.205	0.626	0.6359
Est. + FORCASTER	03:43	0.240*	0.641*	0.6283*
Est. + CMA-ES & NGI	07:23	0.259 *	0.643*	0.6284*
Correct cal. CBCT traj.	00:35	0.294	0.377	0.7894
Est. + FORCASTER CBCT traj.	03:28	0.286	0.392	0.7817
Correct cal. sinus traj.	00:10	0.194	0.486	0.7291
Est. + FORCASTER sinus traj.	00:37	0.183	0.481	0.7311

Table 6.4: Results of the evaluation on the continuous arc trajectory. Here, the forward projection is compared to the acquired data, and the evaluation results for the other two trajectories are also included if the forward projections from the correct calibration were compared to the acquired data. Values marked with an * had significant ($p < 0.05$) changes compared to the refined estimate.

In Figure 6.9, the acquired sinogram and the simulated sinogram that used the estimated parameters, which was also calibrated by the CMA-ES with the NGI objective, are shown. Similar to the sinusoidal trajectory, there were some projections where the estimated parameters were wrong. These can be seen in the lower half of column (a), and on the right of column (b). Apart from these few slices, no major misalignment can be seen.

The reconstructed image that used the continuous arc and the calibration is in Figure 6.10. The image was reconstructed using the FDK algorithm from the astra toolbox without any postprocessing. The images show that the object was reconstructed with just a few artifacts (which came from miscalibration). The results of the metrics are in Table 6.5. Since there exists no correct calibration and reconstruction of this set of projections, the image was compared

with the image obtained by reconstructing the CBCT trajectory. The NGI, SSIM, and NRMSE metrics all increased significantly with each step. There was no significant increase for the Dice value in the reconstruction with refined estimates and the reconstruction after applying the two calibration algorithms. Still, the high Dice score shows that the metal object was reconstructed with a good quality and at the correct position. The SSIM and NGI metrics were not as high as the results from the calibration of the CBCT trajectory. In addition, the misaligned projections generated a few artifacts.



(a) (b)

Figure 6.9: Two slices out of the arc trajectory sinogram from the acquired data, as well as from the calibration with refined estimates and CMA-ES with the NGI objective. The left column is (**a**) ordered from top to bottom as follows: acquired data, and the simulated projections from the calibration with estimated starting parameters. The right columns (**b**) have the same order, also from left to right. Since there were no positional data reported by the Artis Zeego System, there was no calibration without estimated parameters conducted.

6.5 Discussion

This paper describes the FORCAST-EST algorithm, which is able to approximate the initial parameters for the calibration of arbitrary CBCT trajectories. Based on the literature review, this algorithm is the first estimator that uses only online calibration techniques. The results have shown that the estimated parameters are close enough to the correct parameters, such that state-of-the-art algorithms can successfully calibrate the trajectory. The reconstructed images are of comparable quality to those that have been reconstructed after calibration with close-to-accurate starting positions. The Dice score achieved

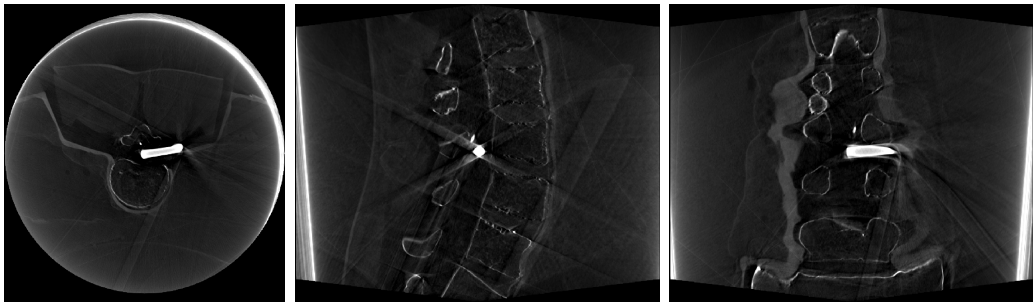


Figure 6.10: Reconstructions of the continuous arc when using the proposed estimator, as well as CMA-ES with the NGI objective and the FDK algorithm from the astra toolbox.

Table 6.5: Metric results for the arc reconstruction. The dice † was evaluated on a segmentation of the large metal object in the phantom in comparison to the segmentation of this object in the reconstructed image from the regular CBCT trajectory. Values marked with an * had significant ($p < 0.05$) changes compared to the refined estimate.

Algorithm	**NGI**	**SSIM**	**NRMSE**	**Dice †**
Coarse estimate	0.353 *	0.360 *	0.3567 *	0.00 *
Refined estimate	0.446	0.418	0.2379	0.82
Est. + FORCASTER	0.510 *	0.473 *	0.2114 *	0.84
Est. + CMA-ES & NGI	0.535 *	0.535 *	0.1958 *	0.84

by both algorithms was very high ≥0.99, which means the metal object in the center was reconstructed correctly. The CBCT calibrations also produced very high SSIM values (>0.999), such that they are very close to the reference calibration. These metric values are comparable to those previously reported for the FORCASTER calibration [1]. The estimation, therefore, does not reduce the quality of the reconstructed images. A comparison to the calibration algorithm was developed by Oudah et al. [73], and this was meant to be used in this paper. However, this was not possible since the calibration algorithm uses metrics that require specific objects in the measured phantom, and these were not included in the object scanned for this work.

For the CBCT trajectory, the refined estimate needed about one and a half hours, while together with the CMA-ES algorithm it took five and a half hours for the calibration. As such, the estimation was less than 30% of the total runtime; furthermore, this difference was even higher for the arc calibration, where the estimation only contributed 20%.

Further, in contrast to the approaches by Grzeda et al. or Lemammer et al. [86, 87], no modification of the CBCT-Device was necessary to estimate the param-

eters. If the modification of the C-Arm is possible and allowed, the addition of more sensors might be the preferred option.

Removing the dependency on roughly accurate starting parameters comes with the cost of a higher computation time. In the experiments, there was a steep increase in the time when compared to the calibration that had its starting parameters provided by the CBCT device. Still, every projection was optimized separately from one another, and they also independently used the calibration algorithm. Therefore, parallelization was easily implemented. The initial generation of the grid with projections can also be parallelized, but this was not implemented since the data can be saved and reused for the next time the calibration algorithm is run. This is possible as it only depends on the prior image, and the same grid can be used for the calibration of different trajectories around the object.

One caveat are projections where the acquired image and simulation differ too much. This can happen for projections that are too far out of the plane of the original CT trajectory. The acquired projections contain parts of the object that are not in the prior CT because the phantom is larger than the CT volume. This makes it harder to find enough matching feature pairs. Therefore, the noise of the miss-matched features on one projection might outweigh the low count of the correctly matched points on the correct projection; as such, the estimated parameters would then be incorrect. During the development, it became evident that the parameters like the ratio of Lowe's ratio test had a high impact on how far the projections would be miscalibrated. Further exploration of the different parameter combinations, or other ways through which to filter out wrong matches, might improve this algorithm. The filtering system for matched feature points could also be replaced with the one used by Yang et al. [82] in their calibration algorithm.

More development should be conducted to reduce the runtime. The implementation relies heavily on the CPU, but the graphic card could reduce the needed computation time drastically, since graphic cards are designed for parallel computing.

6.6 Conclusions

In conclusion, this paper presents an algorithm that is able to roughly calibrate a trajectory without initial parameters, and it is accurate enough for state-of-the-art algorithms to further refine the produced image, as demonstrated when using CMA-ES with NGI and the FORCASTER algorithm.

Supplementary information

If your article has accompanying supplementary file/s please state so here.

Please refer to Journal-level guidance for any specific requirements.

Acknowledgments

This research project is partly supported of the Research Campus M2OLIE funded by the German Federal Ministry of Education and Research (BMBF) within the Framework "Forschungscampus: public-private partnership for Innovations" under the funding code 13GW0388A.

Declarations

The authors declare no conflict of interest.

7 Summary

The first part of this thesis is about MRI with two different focus points.

One is the algorithm for segmenting arteries in Dynamic Contrast Enhanced MRI (DCE-MRI) images. This segmentation works without user input and is a necessary step for reproducible quantitative MRI. Small changes in the segmentation mask can have a big impact on the result of further processing[92]. An automatic and deterministic segmentation helps with reducing the variance of quantitative measurements.

The second paper presents a web application to generate MRI images with different sequences and parameters. This, while not helping directly during an intervention or for image processing, allows medical students and physicians to get an idea of how images will look like for the given sequence and parameters.

In the second part, two algorithms for the calibration of CBCT trajectories are presented. The first (Feature ORiented Calibration of Arbitrary Scan Trajectories with Enhanced Reliability (FORCASTER)) is a further development of the Feature ORiented Calibration of Arbitrary Scan Trajectories (FORCAST)[3] algorithm and has improvements to speed and reliability. This algorithm can calibrate arbitrary trajectories and needs only a prior image and a set of initial parameters. Arbitrary trajectories can be used to reduce metal artifacts or radiation. The second algorithm (FORCAST-parameter ESTimator (FORCAST-EST)) uses the same principles as the other two and can estimate the calibration parameters. It only needs a prior image and can therefore be used to estimate the initial parameters for regular calibration algorithms. Together, these two allow the use of arbitrary trajectories acquired with any X-Ray or CBCT system.

A detailed summary of the four scientific studies presented in chapters 3 to 6 is following.

Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer,
Int J Comput Assist Radiol Surg, doi: 10.1016/j.mri.2020.09.009

In chapter 3 an algorithm for segmenting the abdominal arteries was introduced. The motivation was to make quantitative perfusion calculation more reproducible, which reduces the impact the experience of the physician has on the result of the calculation.

The algorithm works on DCE-MRI images acquired over time, the first acquired image is subtracted from every other image to remove the background.

Then, the algorithm uses nine steps to find and refine the artery segmentation. It starts by selecting the 1% brightest voxels in the first 15 time steps. Then a binary opening separates structures only connected by a few voxels. The third step is finding the time step where the majority of the selected voxels have their highest value. This time step is now used for further segmentation. In the fourth step, the structures in the segmentation mask are labeled and for every structure the percentage they fill of their bounding box is determined. If the filled area is between 60 and 80% the structure is considered roundish, all other structures are removed. Afterward, step three is repeated and if it results in an earlier time step than the one found in step three, it is chosen as the new segmentation time step. Step six fits a gamma variate function to each structure in the segmentation mask. Structures where the parameters are out of bound are discarded. Step seven removes very small structures, and in step eight a flood fill algorithm is applied to each structure. Finally, the largest structure in the left half and right half of the image is selected as the segmentation of the arteries.

To evaluate the accuracy, the output from the algorithm is compared to manual segmentations using the dice coefficient. Here, the presented algorithm reached higher coefficients than all the compared state-of-the-art algorithms. Together with manual annotations of the tumor, the perfusion parameters were calculated and compared. Again, the presented algorithm had lower errors in all parameters than the other algorithms.

VirtMRI: A tool for teaching MRI,
submitted J. Med. Syst., 19.05.2021

The chapter 4 presents a web-based tool for teaching MRI to non-technical and medical students. It allows students to generate MRI images using different sequences and parameters within their own web-browser. The system supports several sequences for the standard proton MRI and also some sodium MRI sequences. The user can set the sequence specific parameters, but also general parameters for image resolution, noise level and undersampling. Generated images are displayed in a 4-pane view, showing a transversal, coronal and sagittal slice. The fourth pane can be changed between the k-space and a 3D-view of the three visible slices. The display also supports windowing of the image. The last two generated images can be displayed side by side to easily compare the differences.

The image generation uses the solution to the Bloch equation for each sequence to calculate the signal response. The equations are evaluated on a dataset consisting of 3D parameter maps for T1, T2, proton density, etc.. The datasets are generated by using published head phantoms where every tissue is filled with the appropriate values. The calculation is performed completely within the browser using WebASM for lower computation time.

Feature-based CBCT Self-Calibration for Arbitrary Trajectories,
Int J Comput Assist Radiol Surg, doi: 10.1007/s11548-022-02645-9

Chapter 5 introduces the FORCASTER algorithm for self-calibration of CBCT trajectories. Calibration of trajectories is necessary to properly reconstruct a CT image. The proposed algorithm uses online-calibration, so it does not need a special phantom and an additional calibration run, only the acquired data and a prior image. Different methods are used to calibrate the image in the three lateral and the three rotational dimensions. The coordinate systems x- and y-axis are in-plane of the detector, with the z-axis pointing towards the source. To correct a shift in x- and y-direction, features are detected and matched between the real and simulated image. Then the parameters are shifted by the median Euclidean distance of the coordinates for matching pairs. The shift in z-direction brings the object closer or farther away from the detector and is therefore a zoom operation. To find the z-correction, all distances between feature points in one image are divided by the distance of the matching points in the other image. The difference to 1 is multiplied with the source-detector-distance to give the z-correction. Each rotation is corrected using a minimizer QUT-AF developed in that chapter. Several projections with slightly changing angles are simulated, and the median Euclidean distance between matching feature points is calculated. A quartic function is fitted to those values, and the minimum is the corrected rotational parameter. All those methods are repeatedly applied to the acquired projection.

The resulting trajectory parameters are of comparable quality as the parameters found by state-of-the-art algorithms. The proposed algorithm is several times faster and has a higher tolerance to deviating starting parameters.

Feature-oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST,
submitted Appl. Sci.

In chapter 6 an algorithm is presented, that can estimate the acquisition parameters for the projections in a CBCT trajectory. This estimation is needed in the case where the CBCT system does not provide a position and orientation for each projection, but the calibration algorithm needs to be initialized with starting parameters.

The algorithm uses a prior image of the object to generate projections in a grid of possible rotations around the object, it uses a grid step size of about 4° or 95 projections for a full rotation. Then the AKAZE algorithm is applied to get feature points for each projection. The projections themselves are discarded. To estimate the parameters for an acquired projection the features are extracted and then compared to the features from the previously generated grid. To increase the speed in a first round the algorithm only compares them to the every fourth grid point, then the five best matching grid points are selected. In a second step the surrounding grid points of these five selected ones

are compared to the features from the acquired projection. The best matching grid point is taken as the coarse estimate.

A refinement of the coarse estimate is using the correction methods for lateral shifts introduced in 5, additionally a new correction for the detector rotation is introduced. Here the Cartesian coordinates of matched feature points are converted to polar coordinates. The median difference of the angular value for each pair of matching feature points is the corrective value for the detector rotation. Each refinement step is repeated several times.

The evaluation of the estimated parameters show, that state-of-the-art calibration algorithms are able to correctly calibrate several different trajectories using these estimates.

8 Outlook

In chapter 3 an algorithm for detecting arteries in DCE-MRI images was presented. The algorithm was compared to previously published algorithms and from these the best algorithm was one that was designed to find arteries in contrast enhanced brain CTs. Therefore, it would be interesting to see if how well my algorithm performs for finding arteries in images of other parts of the body, when using other MRI sequences or in CT images.

For quantitative MRI, more research could be done for deterministic algorithms to segment structures. Time is a major constraint for physicians, so algorithms so hand drawn segmentations are minimalistic. Segmentation algorithms could support this process by quickly and accurately segmenting complex structure. Further calculations, like blood perfusion, can then be easier replicated and are independent of the physician's experience.

VirtMRI

The MR image generator has many possible extensions. Researchers are constantly developing new sequences to measure new signals. So, many additional MR sequences could be added to the system.

New phantoms for different body parts would also be interesting to add. This addition is quite straight-forward since it does not require any change to the program itself, only finding a suitable dataset with the required tissue segmentations. One idea for this was to use digital phantoms if the license allows it.

Also, a real simulation of the Bloch equations should at some point be implemented. This allows for more realistic images that are based on real simulation. But such an implementation requires a lot of work and time.

Further development is also needed for the mobile interface. While the system looks good on a Desktop PC, it is not quite as usable on smartphones. The small display requires a drastically different user interface layout, and the touch display offers different usage options for the image viewer.

Currently under development is the inclusion of the BART[93] toolkit. This toolkit provides many methods for reconstructing MRI images, including some for parallel imaging and compressed sensing. The main problem is correctly cross-compiling the fortran code to WebAssemply, which is not yet supported by the compiler.

CBCT Calibration

Chapters 5 and 6 describe CBCT calibration algorithms. Further development should focus on speed. In their current implementation, these algorithms work, but are only usable in research settings where a few hours of runtime is acceptable. For real word usage, the runtime has to be much shorter.

Another possible research direction is accuracy. In this aspect, it does not reach the same performance as the CMA-ES minimizer. But this increased accuracy should not be accompanied by a large increase of the runtime.

The parameter estimator FORCASTEST needs some more fine-tuning to remove the last few wrongly calibrated projections.

Besides calibrating the trajectory, there is also the task of finding a trajectory. A good trajectory would be one, that reduces the radiation, reduces artifacts within the important area and is fast. More conditions could be included, limits to radiation for certain structures, limits to the C-Arm movement, continuous acquisition, ... This is a very complex, multidimensional optimization process. Research into this also needs to consider the runtime of the system and the usability for physicians during an intervention. Especially important is the user interface, since all the conditions are dependent, and improving one will decrease others.

Bibliography

[1] C. Tönnes, T. Russ, L. R. Schad, and F. G. Zöllner, "Feature-based cbct self-calibration for arbitrary trajectories," *International Journal of Computer Assisted Radiology and Surgery*, vol. 17, no. 11, pp. 2151–2159, 2022.

[2] B. Aubert-Broche, M. Griffin, G. Pike, A. Evans, and D. Collins, "Twenty new digital brain phantoms for creation of validation image data bases," *IEEE Transactions on Medical Imaging*, vol. 25, no. 11, pp. 1410–1416, 2006.

[3] K. Chung, L. R. Schad, and F. G. Zöllner, "Tomosynthesis implementation with adaptive online calibration on clinical c-arm systems," *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 10, pp. 1481–1495, 2018.

[4] V. P. Grover, J. M. Tognarelli, M. M. Crossey, I. J. Cox, S. D. Taylor-Robinson, and M. J. McPhail, "Magnetic resonance imaging: principles and techniques: lessons for clinicians," *Journal of clinical and experimental hepatology*, vol. 5, no. 3, pp. 246–255, 2015.

[5] E. L. Hahn, "Spin echoes," *Phys. Rev.*, vol. 80, pp. 580–594, Nov 1950.

[6] R. Schofield, L. King, U. Tayal, I. Castellano, J. Stirrup, F. Pontana, J. Earls, and E. Nicol, "Image reconstruction: Part 1–understanding filtered back projection, noise and image acquisition," *Journal of cardiovascular computed tomography*, vol. 14, no. 3, pp. 219–225, 2020.

[7] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer Vision – ECCV 2006*, pp. 404–417, 2006.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] P. F. Alcantarilla and T. Solutions, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Patt. Anal. Mach. Intell*, vol. 34, no. 7, pp. 1281–1298, 2013.

[10] P. F. Alcantarilla, A. Bartoli, and A. J. Davison, "Kaze features," in *Computer Vision – ECCV 2012* (A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, eds.), (Berlin, Heidelberg), pp. 214–227, Springer Berlin Heidelberg, 2012.

[11] S. P. Sourbron and D. L. Buckley, "Tracer kinetic modelling in MRI: estimating perfusion and capillary permeability," *Phys Med Biol*, vol. 57, no. 2, pp. R1–33, 2012.

[12] S. P. Sourbron and D. L. Buckley, "Classic models for dynamic contrast-enhanced MRI," *NMR in Biomedicine*, vol. 26, no. 8, pp. 1004–1027, 2013.

[13] M. J. Gollub, D. H. Gultekin, O. Akin, R. K. Do, J. L. Fuqua, M. Gonen, D. Kuk, M. Weiser, L. Saltz, D. Schrag, K. Goodman, P. Paty, J. Guillem, G. M. Nash, L. Temple, J. Shia, and L. H. Schwartz, "Dynamic contrast enhanced-mri for the detection of pathological complete response to neoadjuvant chemotherapy for locally advanced rectal cancer," *European Radiology*, vol. 22, no. 4, pp. 821–831, 2012.

[14] J. S. Lim, D. Kim, S.-E. Baek, S. Myoung, J. Choi, S. J. Shin, M.-J. Kim, N. K. Kim, J. Suh, K. W. Kim, and K. C. Keum, "Perfusion mri for the prediction of treatment response after preoperative chemoradiotherapy in locally advanced rectal cancer," *European Radiology*, vol. 22, pp. 1693–1700, Aug 2012.

[15] M. Ciolina, D. Caruso, D. De Santis, M. Zerunian, M. Rengo, N. Alfieri, D. Musio, F. De Felice, A. Ciardi, V. Tombolini, and A. Laghi, "Dynamic contrast-enhanced magnetic resonance imaging in locally advanced rectal cancer: role of perfusion parameters in the assessment of response to treatment," *La radiologia medica*, vol. 124, pp. 331–338, May 2019.

[16] M. Cutajar, I. Mendichovszky, P. Tofts, and I. Gordon, "The importance of aif roi selection in dce-mri renography: Reproducibility and variability of renal perfusion and filtration," *European Journal of Radiology*, vol. 74, no. 3, pp. 154 – 160, 2010.

[17] M. S. Davenport, T. Heye, B. M. Dale, J. J. Horvath, S. R. Breault, S. Feuerlein, M. R. Bashir, D. T. Boll, and E. M. Merkle, "Inter- and intrarater reproducibility of quantitative dynamic contrast enhanced MRI using TWIST perfusion data in a uterine fibroid model," *Journal of Magnetic Resonance Imaging*, vol. 38, no. 2, pp. 329–335, 2013.

[18] A. S. Lundervold and A. Lundervold, "An overview of deep learning in medical imaging focusing on mri," *Zeitschrift für Medizinische Physik*, vol. 29, no. 2, pp. 102 – 127, 2019. Special Issue: Deep Learning in Medical Physics.

[19] K. Murase, K. Kikuchi, H. Miki, T. Shimizu, and J. Ikezoe, "Determination of arterial input function using fuzzy clustering for quantification of cerebral blood flow with dynamic susceptibility contrast-enhanced mr imaging," *Journal of Magnetic Resonance Imaging*, vol. 13, no. 5, pp. 797–806, 2001.

[20] G. J. M. Parker, A. Jackson, J. C. Waterton, and D. L. Buckley, "Automated arterial input function extraction for T1-weighted DCE-MRI," in *Proceedings of the International Society for Magnetic Resonance in Medicine* (ISMRM, ed.), vol. 11, pp. 0–0, 2003.

[21] K. Mouridsen, S. Christensen, L. Gyldensted, and L. Østergaard, "Automatic selection of arterial input function using cluster analysis," *Magnetic Resonance in Medicine*, vol. 55, no. 3, pp. 524–531, 2006.

[22] J. Chen, J. Yao, and D. Thomasson, "Automatic determination of arterial input function for dynamic contrast enhanced MRI in tumor assessment," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2008* (D. Metaxas, L. Axel, G. Fichtinger, and G. Székely, eds.), (Berlin, Heidelberg), pp. 594–601, Springer Berlin Heidelberg, 2008.

[23] D. Peruzzo, A. Bertoldo, F. Zanderigo, and C. Cobelli, "Automatic selection of arterial input function on dynamic contrast-enhanced mr images," *Computer Methods and Programs in Biomedicine*, vol. 104, no. 3, pp. e148 – e157, 2011.

[24] S. G. Yingxuan Zhu, Ming-Ching Chang, "Automated determination of arterial input function for DCE-MRI of the prostate," in *Proc.SPIE* (Proc.SPIE, ed.), vol. 7963, pp. 7963 – 7963 – 8, 2011.

[25] L. Shi, D. Wang, W. Liu, K. Fang, Y.-X. J. Wang, W. Huang, A. D. King, P. A. Heng, and A. T. Ahuja, "Automatic detection of arterial input function in dynamic contrast enhanced mri based on affinity propagation clustering," *Journal of Magnetic Resonance Imaging*, vol. 39, no. 5, pp. 1327–1337, 2013.

[26] R. Tabbara, A. Connelly, and F. Calamante, "Multi-stage automated local arterial input function selection in perfusion mri," *Magnetic Resonance Materials in Physics, Biology and Medicine*, Nov 2019.

[27] T. Gaa, W. Neumann, S. Sudarski, U. I. Attenberger, S. O. Schönberg, L. R. Schad, and F. G. Zöllner, "Comparison of perfusion models for quantitative t1 weighted dce-mri of rectal cancer," *Scientific Reports*, vol. 7, no. 1, p. 12036, 2017.

[28] S. Sudarski, T. Henzler, T. Floss, T. Gaa, M. Meyer, H. Haubenreisser, S. O. Schoenberg, and U. I. Attenberger, "Variability and reproducibility of 3rd-generation dual-source dynamic volume perfusion ct parameters in comparison to mr-perfusion parameters in rectal cancer," *Scientific Reports*, vol. 8, no. 1, p. 6868, 2018.

[29] F. Calamante, "Arterial input function in perfusion MRI: A comprehensive review," *Progress in Nuclear Magnetic Resonance Spectroscopy*, vol. 74, pp. 1 – 32, 2013.

[30] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[31] T. Sørensen, *A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons*, vol. 5 of *Biologiske skrifter*. I kommission hos Ejnar Munksgaard, 1948.

[32] C. Pradel, N. Siauve, G. Bruneteau, O. Clement, C. de Bazelaire, F. Frouin, S. Wedge, J. Tessier, P. Robert, G. Frija, and C. Cuenod, "Reduced capillary perfusion and permeability in human tumour xenografts treated with the vegf signalling inhibitor zd4190: an in vivo assessment using dynamic mr imaging and macromolecular contrast media," *Magnetic Resonance Imaging*, vol. 21, no. 8, pp. 845 – 851, 2003.

[33] T. Russ, S. Goerttler, A.-K. Schnurr, D. F. Bauer, S. Hatamikia, L. R. Schad, F. G. Zöllner, and K. Chung, "Synthesis of ct images from digital body phantoms using cyclegan," *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, pp. 1741–1750, Oct 2019.

[34] G. J. Parker, C. Roberts, A. Macdonald, G. A. Buonaccorsi, S. Cheung, D. L. Buckley, A. Jackson, Y. Watson, K. Davies, and G. C. Jayson, "Experimentally-derived functional form for a population-averaged high-temporal-resolution arterial input function for dynamic contrast-enhanced mri," *Magnetic Resonance in Medicine*, vol. 56, no. 5, pp. 993–1000, 2006.

[35] F. G. Zöllner, M. Daab, S. P. Sourbron, L. R. Schad, S. O. Schoenberg, and G. Weisser, "An open source software for analysis of dynamic contrast enhanced magnetic resonance images: Ummperfusion revisited," *BMC Medical Imaging*, vol. 16, no. 1, p. 7, 2016.

[36] T. Hackländer and H. Mertens, "Virtual mri: A pc-based simulation of a clinical mr scanner1," *Academic Radiology*, vol. 12, no. 1, pp. 85–96, 2005.

[37] D. Treceño-Fernández, J. Calabia-del Campo, M. L. Bote-Lorenzo, E. G. Sánchez, R. de Luis-García, and C. Alberola-López, "A web-based educational magnetic resonance simulator: Design, implementation and testing," *Journal of Medical Systems*, vol. 44, no. 1, p. 9, 2019.

[38] H. Benoit-Cattin, G. Collewet, B. Belaroussi, H. Saint-Jalmes, and C. Odet, "The simri project: a versatile and interactive mri simulator," *Journal of Magnetic Resonance*, vol. 173, no. 1, pp. 97–115, 2005.

[39] T. Stöcker, K. Vahedipour, D. Pflugfelder, and N. J. Shah, "High-performance computing mri simulations," *Magnetic Resonance in Medicine*, vol. 64, no. 1, pp. 186–193, 2010.

[40] F. Liu, J. V. Velikina, W. F. Block, R. Kijowski, and A. A. Samsonov, "Fast realistic mri simulations based on generalized multi-pool exchange tissue model," *IEEE Transactions on Medical Imaging*, vol. 36, no. 2, pp. 527–537, 2017.

[41] J. E. Wilhjelm, J. Duun-Henriksen, and L. G. Hanson, "A virtual scanner for teaching fundamental magnetic resonance in biomedical engineering," *Computer Applications in Engineering Education*, vol. 26, no. 6, pp. 2197–2209, 2018.

[42] W. Perman, S. Hilal, H. Simon, and A. Maudsley, "Contrast manipulation in nmr imaging," *Magnetic Resonance Imaging*, vol. 2, no. 1, pp. 23–32, 1984. Second Annual Meeting of the Society for Magnetic Resonance Imaging.

[43] L. E. Drain, "A direct method of measuring nuclear spin-lattice relaxation times," *Proceedings of the Physical Society. Section A*, vol. 62, p. 301, may 1949.

[44] P. Moran, N. Kumar, N. Karstaedt, and S. Jackels, "Tissue contrast enhancement: Image reconstruction algorithm and selection of ti in inversion recovery mri," *Magnetic Resonance Imaging*, vol. 4, no. 3, pp. 229–235, 1986.

[45] A. Haase, J. Frahm, D. Matthaei, W. Hanicke, and K.-D. Merboldt, "Flash imaging. rapid nmr imaging using low flip-angle pulses," *Journal of Magnetic Resonance (1969)*, vol. 67, no. 2, pp. 258–266, 1986.

[46] B. Hargreaves, "Rapid gradient-echo imaging," *Journal of Magnetic Resonance Imaging*, vol. 36, no. 6, pp. 1300–1313, 2012.

[47] A. Oppelt, R. Graumann, H. Barfuss, H. Fischer, W. Hartl, and W. Schajor, "Fisp: eine neue schnelle pulssequenz fuer die kernspintomographie," *Electromedica*, vol. 54, no. 1, pp. 15–18, 1986.

[48] G. B. Chavhan, P. S. Babyn, B. G. Jankharia, H.-L. M. Cheng, and M. M. Shroff, "Steady-state mr imaging sequences: Physics, classification, and clinical applications," *RadioGraphics*, vol. 28, no. 4, pp. 1147–1160, 2008.

[49] M. A. U. Hoesl, L. R. Schad, and S. Rapacchi, "Efficient 23na triple-quantum signal imaging on clinical scanners: Cartesian imaging of single and triple-quantum 23na (cristina)," *Magnetic Resonance in Medicine*, vol. 84, no. 5, pp. 2412–2428, 2020.

[50] Y. Qian, A. Panigrahy, C. M. Laymon, V. K. Lee, J. Drappatz, F. S. Lieberman, F. E. Boada, and J. M. Mountz, "Short-t2 imaging for quantifying concentration of sodium (23na) of bi-exponential t2 relaxation," *Magnetic Resonance in Medicine*, vol. 74, no. 1, pp. 162–174, 2015.

[51] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse mri: The application of compressed sensing for rapid mr imaging," *Magnetic Resonance in Medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[52] B. Aubert-Broche, A. C. Evans, and L. Collins, "A new improved version of the realistic digital brain phantom," *NeuroImage*, vol. 32, no. 1, pp. 138–145, 2006.

[53] C. J. Holmes, R. Hoge, L. Collins, R. Woods, A. W. Toga, and A. C. Evans, "Enhancement of mr images using registration for signal averaging," *Journal of Computer Assisted Tomography*, vol. 22, no. 2, 1998.

[54] B. Alfano, M. Comerci, M. Larobina, A. Prinster, J. P. Hornak, S. E. Selvan, U. Amato, M. Quarantelli, G. Tedeschi, A. Brunetti, and M. Salvatore, "An mri digital brain phantom for validation of segmentation methods," *Medical Image Analysis*, vol. 15, no. 3, pp. 329–339, 2011.

[55] J. Z. Bojorquez, S. Bricq, C. Acquitter, F. Brunotte, P. M. Walker, and A. Lalande, "What are normal relaxation times of tissues at 3 t?," *Magnetic Resonance Imaging*, vol. 35, pp. 69–80, 2017.

[56] J. P. Wansapura, S. K. Holland, R. S. Dunn, and W. S. Ball Jr., "Nmr relaxation times in the human brain at 3.0 tesla," *Journal of Magnetic Resonance Imaging*, vol. 9, no. 4, pp. 531–538, 1999.

[57] A. M. Peters, M. J. Brookes, F. G. Hoogenraad, P. A. Gowland, S. T. Francis, P. G. Morris, and R. Bowtell, "T2* measurements in human brain at 1.5, 3 and 7 t," *Magnetic Resonance Imaging*, vol. 25, no. 6, pp. 748–753, 2007. Proceedings of the International School on Magnetic Resonance and Brain Function.

[58] G. Madelin, J.-S. Lee, R. R. Regatte, and A. Jerschow, "Sodium mri: Methods and applications," *Progress in Nuclear Magnetic Resonance Spectroscopy*, vol. 79, pp. 14–47, 2014.

[59] G. Madelin, J. Babb, D. Xia, and R. R. Regatte, "Repeatability of quantitative sodium magnetic resonance imaging for estimating pseudo-intracellular sodium concentration and pseudo-extracellular volume fraction in brain at 3 t," *PLOS ONE*, vol. 10, pp. 1–15, 03 2015.

[60] "Bootstrap · the most popular html, css, and js library in the world.." `https://getbootstrap.com/`. Accessed: 2023-04-24.

[61] "morhetz/gruvbox-contrib: Ports of the gruvbox colorscheme." `https://github.com/morhetz/gruvbox-contrib`. Accessed: 2023-04-24.

[62] "Christiantoennes/virtmri: Virtmri: Generate images for different mri sequences in your browser. written in javascript and c compiled to webasm."

`https://github.com/ChristianToennes/VirtMRI`. Accessed: 2023-04-24.

[63] G. J. Gang, J. H. Siewerdsen, and J. W. Stayman, "Non-circular ct orbit design for elimination of metal artifacts," *Medical imaging 2020: physics of medical imaging*, vol. 11312, p. 27, 2020.

[64] E. A. Pearson, S. Cho, C. A. Pelizzari, and X. Pan, "Non-circular cone beam ct trajectories: A preliminary investigation on a clinical scanner," *IEEE Nuclear Science Symposuim Medical Imaging Conference*, pp. 3172–3175, 2010.

[65] M. Herbst, F. Schebesch, M. Berger, R. Fahrig, J. Hornegger, and A. Maier, "Improved trajectories in c-arm computed tomography for non-circular fields of view," *Proceedings of the Third International Conference on Image Formation in X-ray Computed Tomography, edited by F. Noo (Salt Lake City, UT, 2014)*, pp. 274–278, 2014.

[66] S. Hatamikia, A. Biguri, G. Kronreif, T. Russ, J. Kettenbach, and W. Birkfellner, "Short scan source-detector trajectories for target-based cbct," *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, pp. 1299–1302, 2020.

[67] D. Gross, U. Heil, R. Schulze, E. Schoemer, and U. Schwanecke, "Auto calibration of a cone-beam-ct," *Medical Physics*, vol. 39, no. 10, pp. 5959–5970, 2012.

[68] Y. Kyriakou, R. M. Lapp, L. Hillebrand, D. Ertel, and W. A. Kalender, "Image-based online correction of misalignment artifacts in cone-beam CT," *Medical Imaging 2009: Physics of Medical Imaging*, vol. 7258, pp. 610 – 619, 2009.

[69] J. Muders and J. Hesser, "Stable and robust geometric self-calibration for cone-beam ct using mutual information," *IEEE Transactions on Nuclear Science*, vol. 61, no. 1, pp. 202–217, 2014.

[70] F. Stopp, A. J. Wieckowski, M. Käseberg, S. Engel, F. Fehlhaber, and E. Keeve, "A geometric calibration method for an open cone-beam ct system," *12th International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*, pp. 106–109, 2013.

[71] K. Yang, A. L. C. Kwan, D. F. Miller, and J. M. Boone, "A geometric calibration method for cone beam ct systems," *Medical Physics*, vol. 33, no. 6Part1, pp. 1695–1706, 2006.

[72] L. Von Smekal, M. Kachelrieß, E. Stepina, and W. A. Kalender, "Geometric misalignment and calibration in cone-beam tomography: Geometric misalignment and calibration in cone-beam tomography," *Medical physics*, vol. 31, no. 12, pp. 3242–3266, 2004.

[73] S. Ouadah, J. W. Stayman, G. J. Gang, T. Ehtiati, and J. H. Siewerdsen, "Self-calibration of cone-beam CT geometry using 3d–2d image registration," *Physics in Medicine and Biology*, vol. 61, pp. 2613–2632, mar 2016.

[74] W. van Aarle, W. J. Palenstijn, J. Cant, E. Janssens, F. Bleichrodt, A. Dabravolski, J. D. Beenhouwer, K. J. Batenburg, and J. Sijbers, "Fast and flexible x-ray tomography using the astra toolbox," *Opt. Express*, vol. 24, pp. 25129–25147, Oct 2016.

[75] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[76] C. Mennessier, R. Clackdoyle, and F. Noo, "Direct determination of geometric alignment parameters for cone-beam scanners," *Physics in Medicine & Biology*, vol. 54, p. 1633, feb 2009.

[77] M. W. Jacobson, M. D. Ketcha, S. Capostagno, A. Martin, A. Uneri, J. Goerres, T. D. Silva, S. Reaungamornrat, R. Han, A. Manbachi, J. W. Stayman, S. Vogt, G. Kleinszig, and J. H. Siewerdsen, "A line fiducial method for geometric calibration of cone-beam ct systems with diverse scan trajectories," *Physics in Medicine & Biology*, vol. 63, p. 025030, jan 2018.

[78] M. Ferrucci, P. Heřmánek, E. Ametova, E. Sbettega, M. Vopalensky, I. Kumpová, D. Vavřík, S. Carmignato, T. Craeghs, and W. Dewulf, "Measurement of the x-ray computed tomography instrument geometry by minimization of reprojection errors—implementation on experimental data," *Precision Engineering*, vol. 54, pp. 107–117, 2018.

[79] M. Ferrucci, R. K. Leach, C. Giusca, S. Carmignato, and W. Dewulf, "Towards geometrical calibration of x-ray computed tomography systems—a review," *Measurement Science and Technology*, vol. 26, p. 092003, aug 2015.

[80] A. Zechner, M. Stock, D. Kellner, I. Ziegler, P. Keuschnigg, P. Huber, U. Mayer, F. Sedlmayer, H. Deutschmann, and P. Steininger, "Development and first use of a novel cylindrical ball bearing phantom for 9-dof geometric calibrations of flat panel imaging devices used in image-guided ion beam therapy," *Physics in Medicine & Biology*, vol. 61, p. N592, oct 2016.

[81] S. Ouadah, J. W. Stayman, G. Gang, A. Uneri, T. Ehtiati, and J. H. Siewerdsen, "Self-calibration of cone-beam CT geometry using 3D-2D image registration: development and application to tasked-based imaging with a robotic C-arm," *Medical Imaging 2015: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 9415, pp. 336 – 342, 2015.

[82] S. Yang, Y. Han, L. Li, X. Xi, S. Tan, L. Zhu, M. Liu, and B. Yan, "Geometric parameter self-calibration based on projection feature matching for x-ray nanotomography," *Applied Sciences*, vol. 12, no. 22, 2022.

[83] C. Kim, C. Jeong, M. jae Park, B. Cho, S. Y. Song, S. wook Lee, and J. Kwak, "A feasibility study of data redundancy based on-line geometric calibration without dedicated phantom on Varian OBI CBCT system," in *Medical Imaging 2021: Physics of Medical Imaging* (H. Bosmans, W. Zhao, and L. Yu, eds.), vol. 11595, p. 115952H, International Society for Optics and Photonics, SPIE, 2021.

[84] J. Zhang, B. He, Z. Yang, and W. Kang, "A novel geometric parameter self-calibration method for portable cbct systems," *Electronics*, vol. 12, no. 3, 2023.

[85] L. Zheng, S. Luo, L. Chen, and S. Luo, "Self-geometric calibration of circular cone beam CT based on epipolar geometry consistency," in *Medical Imaging 2019: Physics of Medical Imaging* (T. G. Schmidt, G.-H. Chen, and H. Bosmans, eds.), vol. 10948, p. 109482J, International Society for Optics and Photonics, SPIE, 2019.

[86] V. Grzeda and G. Fichtinger, "C-arm rotation encoding with accelerometers," *International Journal of Computer Assisted Radiology and Surgery*, vol. 5, no. 4, pp. 385–391, 2010.

[87] I. Lemammer, O. Michel, H. Ayasso, S. Zozor, and G. Bernard, "Online mobile c-arm calibration using inertial sensors: a preliminary study in order to achieve cbct," *International Journal of Computer Assisted Radiology and Surgery*, vol. 15, no. 2, pp. 213–224, 2020.

[88] M. M. Mitschke, N. Navab, and O. Schuetz, "Online geometrical calibration of a mobile C-arm using external sensors," in *Medical Imaging 2000: Image Display and Visualization* (S. K. Mun, ed.), vol. 3976, pp. 580 – 587, International Society for Optics and Photonics, SPIE, 2000.

[89] S. Sorensen, M. Mitschke, and T. Solberg, "Cone-beam ct using a mobile c-arm: a registration solution for igrt with an optical tracking system," *Physics in Medicine & Biology*, vol. 52, no. 12, p. 3389, 2007.

[90] P. F. Alcantarilla and T. Solutions, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Patt. Anal. Mach. Intell*, vol. 34, no. 7, pp. 1281–1298, 2011.

[91] Y. Otake, A. S. Wang, J. W. Stayman, A. Uneri, G. Kleinszig, S. Vogt, A. J. Khanna, Z. L. Gokaslan, and J. H. Siewerdsen, "Robust 3d–2d image registration: application to spine interventions and vertebral labeling in the presence of anatomical deformation," *Physics in Medicine & Biology*, vol. 58, p. 8535, nov 2013.

[92] C. Tönnes, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. R. Schad, and F. G. Zöllner, "The impact of variations in annotation on the aif and perfusion parameter," in *4th Conference on Image-Guided Interventions*, 2019.

[93] M. Uecker, F. Ong, J. I. Tamir, D. Bahri, P. Virtue, J. Y. Cheng, T. Zhang, and M. Lustig, "Berkeley advanced reconstruction toolbox," *Proc. Intl. Soc. Mag. Reson. Med*, vol. 23, no. 2486, 2015.

9 Publications

Journal Papers

- **C. Tönnes**, C. Licht, L. Schad and F. G. Zöllner. VirtMRI: A tool for teaching MRI. J. Med. Syst. Submitted Dec 2022

- **C. Tönnes** and F. G. Zöllner, Feature-Oriented CBCT Self-Calibration Parameter Estimator for Arbitrary Trajectories: FORCAST-EST. Appl. Sci. 2023, 13, 9179. https://doi.org/10.3390/app13169179

- D. Bauer, J. Rosenkranz, A. Golla, **C. Tönnes**, I. Hermann, T. Russ, G. Kabelitz, A. Rothfuss, L. Schad, J. Stallkamp and F. Zöllner, Validation of an oligometastatic disease diagnosis workow using an abdominal phantom. Med Phys, 2022, 49 (7), pp.4445-4454.

- **C. Tönnes**, T. Russ, L. Schad and F. Zöllner. Feature-based CBCT Self-Calibration forArbitrary Trajectories. Int J CARS, 2022, 17 (11), pp.2151-2159. doi: 10.1007/s11548-022-02645-9

- A.-K. Golla, **C. Tönnes**, T. Russ, D. F. Bauer, M. F. Frölich, S. J. Diehl, S. O. Schönberg, M. Keese, L. R. Schad, F. G. Zöllner and J. S. Rink, Automated Screening for Abdominal Aortic Aneurysm in CT Scans under Clinical Conditions Using Deep Learning. Diagnostics, 11(11), 2131, doi.org/10.3390/diagnostics11112131, 2021

- D. F. Bauer, T. Russ, B. I. Waldkirch, **C. Tönnes**, W. P. Segars, L. R. Schad, F. G. Zöllner and A.-K. Golla, Generation of annotated multimodal ground truth datasets for abdominal medical image registration. International Journal of Computer Assisted Radiology and Surgery, 16, pp.1277–1285, doi: 10.1007/s11548-021-02372-7, 2021.

- A.-K. Golla, D. F. Bauer, R. Schmidt, T. Russ, D. Nörenberg, K. Chung, **C. Tönnes**, L. R. Schad, and F. G. Zöllner, Convolutional Neural Network Ensemble Segmentation with Ratio-based Sampling for the Arteries and Veins in Abdominal CT Scans. IEEE Transactions on Biomedical Engineering, 68(5), pp.1518-1526, doi: 10.1109/TBME.2020.3042640, 2021.

- **C. Tönnes**, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. Schad and F. Zöllner. Deterministic Arterial Input Function selection in DCE-MRI for automation of quantitative perfusion calculation of colorectal cancer. Magn. Reson. Imaging, 75, pp.116-123 (2021), doi: 10.1016/j.mri.2020.09.009

Conference Abstracts

- **C. Tönnes**, C. Licht, L. R. Schad and F. G. Zöllner, SimMRI – A web-based MR Image Simulator for easy accessible MRI teaching, Proc. Intl. Soc. Mag. Reson. Med., Toronto, Canada, 2023.

- T. Russ, W. Wang, A.-K. Golla, D. F. Bauer, M. Tivnan, **C. Tönnes**, Y. W. Ma, T. Reynolds, S. Hatamikia, L. R. Schad, F. G. Zöllner, G. J. Gang, and J. W. Stayman, Fast CBCT reconstruction using convolutional neural networks for arbitrary robotic C-arm orbits, SPIE Medical Imaging, San Diego, USA, 2022.

- D. F. Bauer, C. Ulrich, A.-K. Golla, T. Russ, **C. Tönnes**, J. Leuschner, M. Schmidt, L. R. Schad and F. G. Zöllner, Image reconstruction using end-to-end deep learning for low-dose CT. Proc. Computer Assisted Radiology and Surgery, Munich, Germany, 2021.

- T. Russ, W. Wang, A.-K. Golla, D. F. Bauer, M. Tivnan, **C. Tönnes**, Y. W. Ma, T. Reynolds, S. Hatamikia, L. R. Schad, F. G. Zöllner, G. J. Gang, J. W. Stayman, Fast Reconstruction of non-circular CBCT orbits using CNNs. Proc. Fully3D Congress, Online, 2021.

- **C. Tönnes**, S. Janssen, A. Schnurr, T. Uhrig, K. Chung, L. Schad and F. Zöllner, Filter-pipeline based algorithm to find the AIF in DCE-MRI images for perfusion calculation of rectal cancer, Proc. Intl. Soc. Mag. Reson. Med. , Virutal Meeting, 2020 28, p.3381.

- **C. Tönnes**, S. Janssen, A. Schnurr and F. Zöllner, Exploration of deep learning approaches for the segmentation of colorectal cancer in DCE-MRI images, Proc. ESMRMB Congress, Virutal Meeting, 2020, pp.173-174.

- **C. Tönnes**, S. Janssen, A. Schnurr, U. Tanja, K. Chung, L. Schad and F. Zöllner. The impact of variations in annotation on the AIF and perfusion parameter, 4th Conference on Image-Guided Interventions, 2019.

10 Curriculum Vitae

Name **Christian Tönnes**

Date of Birth **1. June 1990**

Place of Birth **Rüsselsheim**

Nationality **German**

Education

2019 - today **Doctoral Candidate: Dr. sc. hum.** Ruprecht Karl University of Heidelberg
Grade: tba
Topic: Automatic interventional Imaging for the Parametrisation of Vascular Structures
Supervisor: Prof. Dr. rer. nat. Frank G. Zöllner

2017 - 2019 **Master of Science: Computer Science** Hochschule Mannheim
Grade: 1.7
Master's thesis: Region Selection for Arterial Input Function Determination and Segmentation of Colorectal Tumors in DCE-MRI Images for Perfusion Calculation

2012 - 2017 **Bachelor of Science: Medical Computer Science** Ruprecht Karl University of Heidelberg
Grade: 2.0
Bachelor's thesis: Grafisch unterstützte Konfiguration von Suchanfragen auf einer Patientenfalldatenbank

2010 - 2012 **Bachelor of Science: Informatik** Technische Universität Kaiserslautern
Grade: 5.0

1999 - 2009 **Abitur** Georg-Forster-Gesamtschule Wörrstadt Grade: 2.6

Experience

2009 **Zivildienst** Johanniter-Unfall-Hilfe e.V. Regionalverband Rheinhessen

2010 – 2023 **Rettungssanitäter** Johanniter-Unfall-Hilfe e.V. Regionalverband Rheinhessen

11 Acknowledgements

First, I want to thank Prof. Dr. Lothar R. Schad for the opportunity to start my PhD thesis in his chair.

I take this opportunity to express my gratitude to Prof. Dr. Frank Zöllner for his supervision and guidance for this thesis.

Many thanks to all current and former members of the CKM. The coffee breaks in front of house 8 or in the seminar room were always a welcome distraction and were filled with exciting conversations.

Special thanks go to my old colleges Tom, Dominik and Alena for their advice and help with any problem. For the constructive feedback for my presentations and papers.

Last but not least, I want to thank my family for all the support and encouragement.