FROM THE INSTITUTE FOR SIGNAL PROCESSING
OF THE UNIVERSITY OF LÜBECK
DIRECTOR: PROF. DR.-ING. ALFRED MERTINS

Methods for Motion Compensation in Magnetic Resonance Imaging

DISSERTATION

for Fulfillment of Requirements
for the Doctoral Degree
of the University of Lübeck

from the Departement of Computer Sciences
and Technical Engineering

Submitted by
Tim J. Parbs
from Thedinghausen

Lübeck, 2023

First referee:	Prof. Dr.-Ing. Alfred Mertins
Second referee:	Prof. Dr.-Ing. Erhardt Barth
Date of oral examination:	16.04.2024
Approved for printing:	18.04.2024

Abstract

Magnetic resonance imaging is a vital component of modern medical diagnostics. However, its physical signal acquisition process causes the modality to be highly susceptible to motion artifacts caused by patient movement. The resulting image corruption from voluntary or involuntary movements during a scan often results in the tomography to be unusable and a need to repeat the procedure. In this thesis, several autofocusing algorithms aiming to approximate the motion solely based on captured image data are presented, together with algorithms restoring sharp images from motion corrupted data.

An efficient multi scale parameter path search algorithm is proposed to register partial MRI scans highly affected by rigid motion artifacts. By designing the method to estimate motion in a multi-resolution scheme, the computational cost can be kept manageable. In combination with an image recovery scheme leveraging image sparsity, the algorithm is shown to successfully register motion and restore images even for large motion amplitudes.

Furthermore, a novel deep learning convolutional attention layer is presented as part of an augmented U-Net architecture for image registration. Two image processing tasks are tackled using the attention layer. First, a deep learning approach is presented incorporating the layer. The network is trained on an augmented real-world dataset containing cardiac cine MRI measurements of multiple patients. Using a dedicated loss function, the results show that the network natively learns to output diffeomorphic registrations. The performance of the network is compared to state-of-the-art deep registration and numeric registration methods and outperforms both.

Afterwards, the task of deep image registration is extended to sequences of data sampled during an MRI scan. Several deep network architectures are evaluated for this novel task. Extensive tests on real-world data and comparison to numeric methods not only show that deep registration outperforms numeric methods when faced with frequency subsampled data, but that the application of the novel convolution layer improves registration performance.

Zusammenfassung

Die Magnetresonanztomografie ist eine essentielle Methode der modernen Bildgebung und Diagnostik. Der physikalische Prozess, welcher zur Erzeugung und Messung von Bildern genutzt wird, macht die Methode jedoch anfällig für Bewegungsartefakte. In dieser Arbeit werden mehrere Autofocusing-Ansätze vorgestellt, die allein auf Basis der gemessenen Bildrohdaten die den Artefakten zugrunde liegende Bewegung schätzen und die Bildqualität verbessern.

Ein effizienter Algorithmus zur Parametersuche in Bewegungsmodellen wird als Ansatz zur Schätzung von rigiden Patientenbewegungen vorgestellt. Die Methode nutzt einen mehrstufigen Verfeinerungsprozess, um gleichzeitig eine mehr als pixelgenaue Auflösung der Bewegung zu erreichen und die benötigte Rechenzeit niedrig zu halten. In Kombination mit einer Bildschätzung auf Basis von konvexer Optimierung und unter Ausnutzung der Spärlichkeit der Bilder wird gezeigt, dass der Ansatz selbst für starke Bewegungsartefakte die Bildqualität deutlich erhöht.

Weiterhin werden künstliche neuronale Netze als ein Lösungsansatz für die Bildregistrierung entwickelt und eingesetzt. Weiterhin wird ein neuartiger faltungsbasierter Attention-Mechanismus vorgestellt. Die entwickelte Schicht wird als Teil von erweiterten U-Net-Architekturen eingesetzt, um zwei Bildverarbeitungsprobleme zu bearbeiten. Ein Ansatz zur elastischen Bildregistrierung mittels maschinellen Lernens und unter Verwendung der entwickelten Struktur wird präsentiert und auf Cine-MRT-Daten der Herzbewegung mehrerer Patienten trainiert. Eine dafür angepasste Fehlerfunktion ermöglicht dem Netzwerk die Erzeugung diffeomorpher Registrierungen zu erlernen. Die vorgestellte Architektur erzeugt im direkten Vergleich mit State-of-the-Art Netzwerken und numerischen Verfahren verbesserte Registrierungsergebnisse.

Dieser Ansatz wird für die MRT-spezifische Problemstellung weiterentwickelt und zur Registrierung von Frequenzunterräumen medizinischer Bilder eingesetzt, welche in einer Simulation gewöhnlicher MRT-Sequenzen gemessen werden. Verschiedene Netzwerkarchitekturen werden für diese neuartige Fragestellung erprobt, und es wird gezeigt, dass Netzwerke, welche die entwickelte Attention-Schicht einsetzen, die besten Ergebnisse erzielen.

Contents

1 Introduction

Magnetic resonance imaging (MRI) is a non-invasive tomography modality which has become indispensable for modern medical diagnostics. After its inception in the middle of the last century, the method quickly gained acceptance and worldwide relevance. According to recent data [3], 158 MRI scans are conducted in Germany yearly and per 1000 inhabitants[1], almost putting the country at the top for scans per capita.

In contrast to comparable techniques, such as computed tomography scans, MRI is able to offer a higher contrast in soft tissue and allows easier differentiation between different tissue types. This makes MRI a suitable tool for accurate diagnosis in organs such as brain, heart or lungs as well as ligaments and cartilage. Combined with the methods ability to create tomographies in any arbitrary plane or volume in the patient body, this also makes the modality the method of choice for imaging delicate peripheral joints and their connective tissues.

Furthermore, MRI is not reliant on potentially harmful ionizing radiation. Instead, images are produced through a combination of the application of multiple strong external magnetic fields and the transmission of precisely controlled electromagnetic pulses through the subject. Due to the nature of the creation and acquisition of the signals carrying image information, MRI relies on large hardware which is expensive to operate. Furthermore, its measurements take

[1] Report from the Organisation for Economic Co-operation and Development from 2021

a comparatively long time, often requiring upwards of twenty minutes for the acquisition of a single image.

Aside from patient comfort inside the narrow scanner bore, this poses problems as MRI scans are highly susceptible to subject motion occurring during image acquisition. Even slight movements of less than a centimetre can degrade the quality of the tomography to the point where it is no longer useful for diagnosis, leading to a necessity of repeated scans.

All MRI scans of living patients are affected by motion artifacts to some degree, as perfect motionlessness is not achievable. In many regions of interest such as extremities or joints, patient compliance and preparation can help minimise their influence by simply immobilising or not moving the body part. While sedation and fixation can limit the motion of extremities more effectively, the solution they offer is far from ideal. Both methods are especially problematic for children or patients affected by health issues or claustrophobic tendencies, which are the groups unlikely to be able to lie perfectly still for the duration of a scan.

With both patient compliance and interventional methods yielding imperfect and situational solutions, other approaches to the problem of patient motion have been pursued which allow for some limited movement during the scan. These solutions are either *prospective* or *retrospective*, with prospective methods measuring and compensating for motion during the scan itself.

Retrospective motion compensation on the other hand aims to minimise motion artifacts after the acquisition by aligning measured data. Here, methods which incorporate separate measurements of motion exist as well, for example by using respiratory belt signals to filter out scan data exhibiting strong relative motion. Drawbacks of these methods include the reliance of external hardware and typically prolong the scan duration.

This work on the other hand is focussed on the subset of retrospective MRI motion compensation which does not require additional measurements and aims to reconstruct a full image from the data acquired during a regular scan. These *autofocusing* or *blind* motion compensation methods have a number of advantages for both the radiologist and the scan subject, as they require no extra hardware and can shorten the time to acquire a full image.

Two different perspectives of this topic are taken, in which blind motion estimation is either tackled by energy minimisation or by using novel deep learning models to register images directly.

This work is separated into 7 main chapters. After the problem definition and contribution summary of the current chapter, Chapter 2 will introduce the physical mechanism of MR imaging and the subset of MR techniques as well as the discrete sampling model used in this thesis.

Chapter 3 then introduces the basics of deep learning, which are later used to formulate two of the approaches towards registration. A short summary of the history of artificial neural networks is presented together with an overview of current techniques and a detailed description of the parts of machine learning theory relevant for this thesis. In Chapter 4, theoretical and practical aspects of motion estimation and convex image recovery are introduced which form the basis of later models.

In the following chapters, the models developed within the scope of this thesis are presented. In Chapter 5, a model for registering bulk motion is introduced, which is able to correctly approximate motion with sub-pixel accuracy. The approach is evaluated using extensive simulations.

In Chapter 6, the deep learning approaches developed for image registration and autofocusing are presented. Section 6.3 introduces a novel attention mechanism for machine learning models, in which all necessary embeddings are created using convolutional operations. The novel layer is used to build a deep image registration architecture in Section 6.6, which is extended in Section 6.7 to register sequences of frequency subspaces as they might occur during motion afflicted MRI. Both machine learning approaches are trained and evaluated using a real-world dataset containing scans from the cardiac region of several dozens of patients.

Chapter 7 finally gives a brief summary of the thesis, its methods and their results, followed by an overview of open questions and further fields of research.

1.1 Contribution

In this work, two radically different approaches to autofocusing and image registration in MRI are offered, with one being a more traditional autofocusing algorithm and the other leveraging deep learning to approximate motion. The three main advancements offered in this thesis can therefore be grouped into either of these two categories.

In Chapter 5, a blind retrospective motion compensation algorithm for MRI first published in 2019 [1] is discussed. This approach significantly reduces motion artifacts from images corrupted by rigid patient motion without requiring additional data or modifying scan parameters. The algorithm estimates motion by means of an efficient parameter search for a given motion model. The motion estimate is leveraged for enhanced reconstruction of an improved image in a convex recovery scheme. Motion estimation and image reconstruction are run sequentially in an alternating fashion. The size of the rather large motion parameter space is restricted by additionally incorporating a multi-scale approach, which gradually refines the search space of the estimated parameters to a sub-pixel resolution. The presented solution offers robust image artifact reduction or removal even for highly corrupted MRI scans.

The two contributions to the field of deep learning and registration are introduced in Chapter 6.

In Section 6.6, the novel convolutional attention layer presented in Section 6.3 is incorporated in a deep registration algorithm as an extension to a conventional image processing architecture. The network is trained on real-world cardiac cine MRI images and compared to numeric registration methods and other networks trained for the same task. The presented attention layer improves objective registration quality for all evaluated cases.

In Section 6.7, the previous deep registration method is expanded into a novel deep learning task. Here, the problem of motion registration for multi-shot MRI sequences is tackled using a convolutional network architecture which again incorporates the previously presented attention cell. Extensive testing of the architecture shows that the inclusion of the developed attention mechanism again improves registration results.

2

Basics of Magnetic Resonance Imaging

Contents

Magnetic Resonance Imaging (*MRI*) measures the spatially varying concentration of specific atoms inside a solid object non-invasively using varying magnetic gradient fields and electromagnetic pulses. This imaging modality is used in modern medical diagnostics to create tomographic images from patients' bodies. In contrast to other tomographic imaging modalities, such as computed tomography, MRI uses no ionizing radiation which might create an additional health concern.

The next chapters will give a brief introduction to the physical process driving MRI data acquisition, the gradient sequences used to measure a tomography and its image reconstruction process.

2.1 Physical Background

MRI (and its material science brother, Nuclear Magnetic Resonance) is able to capture the spatial distribution $\rho(\boldsymbol{z})$ of certain atomic nuclei with an unchanging, intrinsic nuclear property called *spin*. Although there is no exact analogy, this property can be understood as similar to angular momentum in classical physics. Crucially for the medical application of MRI, hydrogen has a non-zero spin and is found in the fat and water making up the bulk of most tissues throughout the human body.

A non-zero spin causes nuclei to express a magnetic dipole. Although the strength of the magnetic moment is unchanging for a given nucleus, its direction normally is random due to thermal motion. This causes the net magnetic field of all the nuclei - and therefore the magnetisation of the object - to be zero. By applying an approximately homogenous strong external magnetic field $\mathbf{B}_0$, the nuclear dipoles align along the field, and the object is magnetised[1]. Conventionally, a coordinate system is applied in which the external magnetic field $\mathbf{B}_0$ points along the longitudinal z-axis.

Due to their spin, the nuclei of a single element precess along $\mathbf{B}_0$ with their specific *Larmor frequency* ω_L given by

$$\omega_L = \gamma \mathbf{B}_0. \tag{2.1}$$

In this Equation, γ is the gyromagnetic ratio of the element (for hydrogen, $\gamma = 267.52 \cdot 10^6\,\mathrm{rad\,s^{-1}\,T^{-1}}$).

Defining the three dimensional magnetisation of the object as

$$\boldsymbol{M}(t,\boldsymbol{z}) = \begin{bmatrix} M_x(t,\boldsymbol{z}) \\ M_y(t,\boldsymbol{z}) \\ M_z(t,\boldsymbol{z}) \end{bmatrix} \tag{2.2}$$

allows a description of the magnetisation as a spatially and time dependent vector field.

[1]Every nucleus spin can still inhabit different discrete spin states, their number dependent on a material-specific constant. Hydrogen atoms can inhabit two different spin states, 'with' or 'against' $\mathbf{B}_0$, with one state being slightly preferred. The difference in state population causes detectable magnetisation.

A short time after the dipoles are subjected to the magnetic field, the spins enter *equilibrium* and align with $\mathbf{B}_0$. The resulting longitudinal equilibrium magnetisation is denoted $M_z(t, \boldsymbol{z}) = M_0$.

By introducing a radio-frequency excitation pulse S_1 with a frequency of ω_L, the precession angle of the spin system can be manipulated. Applying the pulse in the xy-plane of the coordinate system flips the spin angle of the nuclei in that direction, with the flip angle being dependant on the length and strength of the pulse. As a consequence, this also rotates the magnetisation angle of the spin system. Crucially, the affected hydrogen spins are not only influenced in their spin precession angle, but are phase-aligned after the pulse as well. Receiver coils aligned in the xy-plane are able to capture the net amplitude and phase of the transversal component of the precessing magnetisation.

As worked out in e.g. [4] and assuming a homogenous magnetic field, the signal received at time t can be written as

$$s(t) \propto e^{-i\omega_L t} \int_V M_{xy}(t, \boldsymbol{z}) d^3\boldsymbol{z} \tag{2.3}$$

by denoting the time and spatially dependent transversal magnetisation as

$$\begin{aligned} M_{xy}(t, \boldsymbol{z}) &= \begin{bmatrix} M_x(t, \boldsymbol{z}) \\ M_y(t, \boldsymbol{z}) \end{bmatrix} && (2.4) \\ &\propto \rho(\boldsymbol{z}). && (2.5) \end{aligned}$$

The so-called carrier signal $e^{-i\omega_L t}$ which results from the excitation can be effectively removed by demodulation.

The received *induction signal* is short-lived, however, as interactions in the scanned material quickly cause the spin signal to diminish. Two effects are prevalent: the loss of transversal magnetisation due to a gradual return of the magnetisation vector to the longitudinal axis (T1 relaxation), and a loss of phase coherence (T2 relaxation).

The relaxation of the magnetisation is described by the well-known Bloch equation [5] as

$$\frac{d\boldsymbol{M}(t, \boldsymbol{z})}{dt} = \boldsymbol{M}(t, \boldsymbol{z}) \times \gamma \boldsymbol{B} + \begin{bmatrix} -M_{xy}(t, \boldsymbol{z})/T_2(\boldsymbol{z}) \\ (M_0(t, \boldsymbol{z}) - M_z(t, \boldsymbol{z}))/T_1(\boldsymbol{z}) \end{bmatrix} \tag{2.6}$$

where T_1 and T_2 are time constants describing the T1 or T2 relaxation of the material at location $\boldsymbol{z}$ and $\boldsymbol{B}$ is the overall magnetic field applied by the scanner. Specifically, $T_1(\boldsymbol{z})$ is the constant time the material at position $\boldsymbol{z}$ requires to regain 63% of its maximal longitudinal magnetisation M_0, while $T_1(\boldsymbol{z})$ is the constant time the material needs to lose 37% of its maximal transversal magnetisation after excitation.

Using the process outlined above, it would be possible to get a measurement of the overall spin - or hydrogen - concentration in the body. To resolve an image, an additional time-resolved linear (and again approximately homogenous) magnetic gradient field $\mathbf{B}_G$ is introduced. The overall magnetic field in the scanner at a time t is then given as

$$\mathbf{B}(\boldsymbol{z}, t) = \mathbf{B}_0 + \langle \boldsymbol{z}, \mathbf{B}_G(t) \rangle. \tag{2.7}$$

In effect, the resonant Larmor frequency of the system becomes spatially and time dependent:

$$\omega_L(\boldsymbol{z}, t) = \gamma \mathbf{B}(\boldsymbol{z}, t) \tag{2.8}$$

Several encoding techniques exploiting the spatial differences in precession frequency are applied in tandem to spatially resolve a tomographic image. For most scan modalities, $\mathbf{B}_G(t)$ is decomposed into a longitudinal *slice selection* component $\mathbf{B}_{Gz}(t)$ and a transversal component $\mathbf{B}_{Gxy}(t)$ so that

$$\mathbf{B}_G(t) = \begin{bmatrix} \mathbf{B}_{Gxy}(t) \\ \mathbf{B}_{Gz}(t) \end{bmatrix}, \quad \mathbf{B}_{Gxy}(t) = \begin{bmatrix} \mathbf{B}_{Gx}(t) \\ \mathbf{B}_{Gy}(t) \end{bmatrix}. \tag{2.9}$$

The next sections deal with the design of these gradients and their implications for MR imaging.

2.1.1 Slice selection

The application of the gradient field $\mathbf{B}_{Gz}(t)$ during excitation enables the design of a *selective* excitation pulse $S_{sl}(t)$ to specifically excite spins in a narrow precession frequency range and therefore a narrow longitudinal range. The shape of the radio frequency excitation pulse varies with application, but rectangular or gaussian pulse shapes are common.

As an example, a simple pulse design with amplitude K, a rectangular excitation profile and a frequency band of $\omega_S \pm \Delta\omega/2$ has the following characteristic in frequency and time domain:

$$S_{sl}(\omega) = \begin{cases} K, & \text{if } |\omega - \omega_S| < \Delta\omega/2 \\ 0, & \text{otherwise} \end{cases} \longleftrightarrow S_{sl}(t) = \frac{K\Delta\omega}{\pi} \operatorname{sinc}(\Delta\omega\, t)\, \mathrm{e}^{-j\omega_S t}. \quad (2.10)$$

Using the selective pulse in conjunction with a magnetic field gradient $\mathbf{B}_{Gz}$ would excite a slice of spins along the longitudinal z-axis centred around

$$z_S = \frac{\omega_S}{\gamma(\mathbf{B}_{Gz} + \mathbf{B}_0)}. \quad (2.11)$$

Its thickness is related to the bandwidth $\Delta\omega$ of the rectangular pulse, which means that thinner slice excitations require more stretched out excitation pulses. Since the sinc$(\cdot)$ does not have compact support, the pulse is truncated in practice, leading to a trade-off between pulse length, signal-to-noise ratio and resolution along the z-axis.

2.1.2 Fourier Encoding

Having excited a slice of the object, gradient fields orthogonal to $\mathbf{B}_{Gz}$ are used to encode the tomography of the object into the measured signal by precisely altering the precession frequency of the magnetisation in the transversal xy-plane of the body or object at specific points in time. The two separate, time-resolved magnetic gradient fields $\mathbf{B}_{GP}$ and $\mathbf{B}_{GF}$ are used to accomplish this, which together make up the transversal gradient as

$$\mathbf{B}_{Gxy}(t) = \mathbf{B}_{GF}(t) + \mathbf{B}_{GP}(t). \quad (2.12)$$

In the *phase encoding* direction, a linear magnetic gradient field $\mathbf{B}_{GP}(t)$ is applied for a short variable duration t_{PE} between slice excitation and signal recording. As the precession frequency in the phase encoding direction differs during t_{PE}, a phase offset is imposed onto the spins, which persists until the next excitation.

Additionally, in the *readout* direction, a similar linear magnetic gradient field $\mathbf{B}_{GF}(t)$ is applied while the signal is received. In effect, the precession frequency of the molecules in the excited slice now varies along the readout direction.

Signals along the frequency encoding direction can be separated in the signal induced in the measurement coils, since the frequency shift created by $\mathbf{B}_{GF}(t)$ can be demodulated. This is not possible for the phase encoding as only a single phase offset can be encoded into each frequency component. Therefore, the phase encoding process has to be repeated multiple times to cover the whole two-dimensional frequency plane. In combination, these properties allow Fourier encoding to spatially encode a single, continuos line in frequency space, which can be measured (almost) instantaneously.

While the frequency and phase encoding directions only have to span the transversal plane, they are often chosen as orthogonal.

2.1.3 The k-Space

Ignoring effects of relaxation, the time dependent signal received can be compactly expressed in relation to the underlying spin density and the gradient field by extending Equation (2.3) as

$$s(t) \propto \int_V \exp\left(-i\gamma \int_0^t \langle \boldsymbol{z}, \mathbf{B}_G(t') \rangle dt'\right) \rho(\boldsymbol{z}) d^3\boldsymbol{z}, \tag{2.13}$$

where the carrier signal is omitted.

Here, $t = 0$ denotes the time the excitation pulse flips the spin system and a signal is induced in the receiver coil.

Furthermore, this can be simplified by assuming that the selection slice thickness is negligible. This way, the signal from a given slice is solely influenced by $\mathbf{B}_{Gxy}(t)$. Defining

$$\boldsymbol{k} = \frac{\gamma}{2\pi} \int_0^t \mathbf{B}_{Gxy}(t') dt', \tag{2.14}$$

and inserting Equation (2.14) into Equation (2.13) yields the signal from the excited slice in a form

$$s(\boldsymbol{k}) \propto \int_V \exp\left(-i2\pi \langle \boldsymbol{r}, \boldsymbol{k} \rangle\right) \rho(\boldsymbol{x}) d^2\boldsymbol{x}. \tag{2.15}$$

Using this reformulation, it becomes obvious that the signal forms a Fourier pair with $\rho(\boldsymbol{x})$.

The substitution of Equation (2.14) yields the so-called *k-space* (first defined in [6] and [7]), a popular notation in MRI. The k-space abstracts the complicated

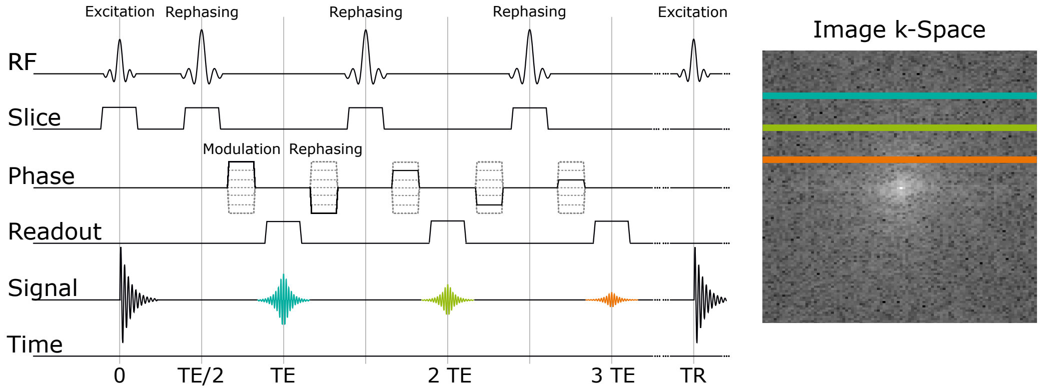


Figure 2.1: A simplified fast spin echo sequence. A single slice of the object is excited with an excitation pulse. The spin system is rephased, phase modulated using a phase encoding gradient and a single k-space line is frequency encoded and sampled at time TE. The system is rephased again, and the process is repeated for a different phase modulation, yielding a different k-space line. Note that this schematic is somewhat simplified and does not, for example, show the phase correction or spoiler gradients used in practice.

gradient integrals from the two-dimensional frequency space holding the information of the tomography image.

2.1.4 Pulse and Gradient Sequences

The pulse used to excite a single readout line in the object flips the hydrogen precession angles into the transversal plane. By doing so, two problems arise: in comparison to the measured signals, the radio-frequency pulses used to excite the spin system are of comparatively high energy. Thus, the simultaneous detection of the system's signal is impossible. Secondly, the system's response is very short, as the spins immediately begin to dephase and the signal produced magnetisation signal quickly decays. Therefore, another radio frequency pulse is introduced after a short time. The so-called rephasing pulse flips the direction of dephasing. This causes the system to return to phase coherence after an echo time of TE, when the signal can be acquired. This combination of flipping and rephasing pulses is commonly referred to as the *spin-echo* sequence.

Since the system is still in an excited state after signal acquisition, this process can be repeated multiple times for a single excitation pulse. Rephasing the signal multiple times, using different readout and phase encoding gradients and undoing the phase modulation after signal acquisition yields the *fast spin-echo* sequence. Using this measurement scheme, multiple lines in k-space can be acquired almost instantaneously.

Using all the techniques presented in this chapter, the k-space of multi-dimensional images can now be measured using a sequence of precisely timed gradient fields and radiofrequency pulses. As Fourier encoding allows measuring a line in k-space at once, the whole frequency space of the spin distribution in question is sampled by measuring multiple such lines. An example of part of a spin-echo-sequence is shown in Figure 2.1.

The timing of such a spin sequence has great influence on the captured image. After excitation, the spin system requires some time to relax and regain its longitudinal magnetisation. Tissues and materials with a short T1 relaxation time will be able to complete the relaxation process faster. Thus, if the repetition time of a sequence is chosen short enough, the difference in relaxation creates a contrast in the next excitation, with the more relaxed materials producing a stronger signal. As the T1 relaxation time differences then dominantly influence the measured image, this is known as T1-weighted imaging.

Choosing a long repetition time will negate this effect, as all tissues have enough time to regain their longitudinal magnetisation. Then, the image contrast will be dominated by differences in phase coherence, with areas losing this coherence faster appearing darker. This concept is therefore known as T2-weighted imaging.

Similarly, the weighting between the two relaxation times is influenced by the choice of the excitation pulse. For short, low-angle excitation pulses, the resulting signal will be T2-weighted, as the longitudinal magnetisation plays less of a role than phase coherence. The same is true for the opposite - larger flip angles lead to the T1-weighting to become dominant in the images.

2.2 Motion Artifacts in MRI

The motion observed in the various parts of the human body is highly diverse and can, in some settings, be of critical importance for medical diagnoses. Examinations of this type thus require very fast temporal sampling in its scanning modalities to accurately picture the object in motion. An extreme example is the analysis of fluid motion in blood vessels, for example in cardiac blood flow analysis[2]. In

[2]While this is traditionally the domain of specialised imaging techniques such as Doppler ultrasound imaging, it has recently been accomplished using phase-contrast MRI [8].

such cases, the characteristics of the motion or flow can be more important than the structural information from the tomographic image.

In most MRI applications, however, the structural information of the organic tissues is of far higher interest than any motion that might occur during a scan - such as when imaging regions of the brain, lungs or extremities. Here, a couple or peculiarities specific to the signal capturing process of MRI become relevant.

Since the acquisition of magnetic resonance signals is dependent on spin relaxation times, there is an inherent trade-off between measurement time and the quantity of recorded image data[3]. Thus, higher resolution scans which require more sampled data also require more scanning time.

Even when patient comfort in the confines of the scanner is ignored, the long scan times required for high-resolution MRI pose significant problems, as the modality is highly sensitive to patient motion. As described in the previous chapter, MRI acquires its signals on sequentially measured lines in a frequency space. Shifts in the patients anatomy during a scan will cause misalignment of the measured points. Since the MR signals are related to human-readable image information via the inverse Fourier transform, misalignments will result in image-wide artifacts, often corrupting the image beyond the point of clinical usefulness.

There are a number of ways to prevent motion artifacts in MRI scans. The first is simply *patient compliance* - asking the patient to lie still during the procedure. This is highly effective when measuring extremities of cooperative patients. Additionally, for especially high-resolution scans of sensitive tissue, mechanical fixation devices can be used, even though they may cause some patient discomfort. For uncooperative patients or subjects dealing with claustrophobia, sedatives are sometimes used as well, but pose their own problems due to limited medical supervision during the scan.

These methods only work on voluntary motion, however, and have no effect on involuntary physiological movements. Especially the thorax region is problematic in this regard, as in a live patient, several sources of complex motion overlap. The normal breathing motion causes the lungs to inflate and deflate and thereby shifts adjoining organs. Additionally, the constant peristaltic motion of the intestine and the non-rigid deformation of the beating heart all combine toward motion artifacts.

[3] The strength of the magnetic baseline and gradient fields $\mathbf{B}_0$ and $\mathbf{B}_G$ also influence the strength of the acquired signals, though they are constrained by scanner hardware.

2.3 Sampling Trajectories

With the spatial encoding described in Section 2.1, each partial scan (or *shot*) of an MRI measurement can be described as a discrete collection of coordinates in k-space (the *trajectory* of the partial measurement) and their corresponding coefficients, which are measured simultaneously. This section describes the trajectory used in this work, namely the PROPELLER trajectory introduced by Pipe in 1999 [9] as well as the Cartesian trajectory as its precursor. From this chapter on, the notions of spin density distributions and relaxation weightings are abstracted to an *image* to be recovered from scanned data.

A base assumption in this work (and almost all similar published research on the topic of registration and alignment) is that the image can be modelled as static during an acquisition. Accordingly, subject motion occurs almost entirely during the relaxation time of the MRI spin system between acquisitions. The reasoning is that the spin system takes far longer to reach equilibrium than the time spent measuring a single read-out line. According to [10] and [11], for example, the relaxation time in a conventional sequence is approximately 22 to 28 times longer than the time to acquire a single shot[4]. Following this assumption, each scan in a measurement sequence of N shots 'sees' one of N deformations of a continuos base image $\tilde{\boldsymbol{I}}$. Here, the nth scanned image is denoted $\tilde{\boldsymbol{I}}^{(n)}$.

Let $\boldsymbol{k} = [\boldsymbol{k}^{(1)}, \boldsymbol{k}^{(2)}, \ldots, \boldsymbol{k}^{(N)}]$ be the sequence of coordinate sets defining the trajectories of an MRI measurement consisting of N shots. Then, $\boldsymbol{k}^{(n)} = [\boldsymbol{k}^{(n,1)}, \boldsymbol{k}^{(n,2)}, \ldots, \boldsymbol{k}^{(n,M)}]$ is the partial sequence of M Cartesian coordinates to a two-dimensional k-space measured in the n-th shot of the MRI measurement, with each sampling point $\boldsymbol{k}^{(n,m)}$ being a 2-dimensional frequency coordinate. The partial measurement of the image observed at this time can then be expressed by

$$\boldsymbol{y}^{(n)} = \begin{bmatrix} y^{(n,1)} \\ y^{(n,2)} \\ \vdots \\ y^{(n,M)} \end{bmatrix}, \quad y^{(n,m)} = \int_V \mathcal{M}_{\boldsymbol{x}} \, \tilde{\boldsymbol{I}}^{(n)}(\boldsymbol{x}) \exp(-i2\pi \langle \boldsymbol{k}^{(n,m)}, \boldsymbol{x} \rangle) d^2\boldsymbol{x} + \epsilon, \quad (2.16)$$

[4]Forbes et al. [10] state a TR / TE of 2600/90 ms, Wang et al. [11] a TR / TE of 1600/70 ms, both for T2-weighted imaging using PROPELLER.

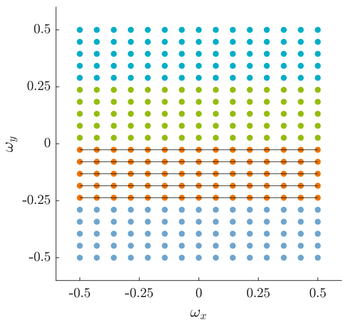


Figure 2.2: A sample Cartesian trajectory. Trajectories of the 4 partial acquisitions in k-space are shown in different colours, with dots denoting measurement points. For a single acquisition, the actual 'path' through k-space is shown as a black line.

which is a highly truncated Fourier series of the continous image and can be understood as a discretisation of the signal formulation from Equation (2.15).

In the above equation, ϵ denotes measurement noise and the operator $\mathcal{M}_{\boldsymbol{x}}$ represents additional modificators, like field inhomogeneities present during the measurement or coil sensitivity profiles for multi-coil measurements such as SENSE [12]. As the work of this thesis does not take into account either of those, $\mathcal{M}_{\boldsymbol{x}}$ and ϵ are neglected in later chapters and experiments.

Applying this notation, a single shot can be uniquely defined by the discrete k-space coordinates $\boldsymbol{k}^{(n)}$ and the measured complex coefficients stacked in $\boldsymbol{y}^{(n)}$. This is repeated for all N scans.

2.3.1 The Cartesian Trajectory

A basic MRI trajectory consists of sequentially scanning straight lines in k-space. This is called the *Cartesian trajectory*, which is shown schematically in Figure 2.2. The Fourier encoding introduced in Section 2.1.2 allows such lines to be captured together by spatially modulating frequencies in the k-space during measurement. In practice, readout lines have to be discretised into a number of sampled points. The phase encoding direction (which is orthogonal to the readout-direction in this trajectory) is sampled by repeating the measurement with different gradients. In modern MRI sequences, several such lines can be captured in one shot. In the case of non-undersampled MRI, this leads to a uniform sampling of k-space, from which a tomography image can be obtained by an inverse Fourier transform. The scanning raster for the trajectory is shown in Figure 2.2.

2.3.2 PROPELLER

A more recent development is the use of the *Periodically Rotated Overlapping Parallel Lines with Enhanced Reconstruction* (PROPELLER) trajectory [10]. PROPELLER was developed for head and thorax scans to enhance robustness towards rigid patient motion.

PROPELLER acquires frequency coefficients on a discretized and equispaced rectangular grid in k-space centred around its origin for each partial scan. In this modality, each partial scan is called a *blade*. Each shot captures a few frequency encoded lines, yielding a sampling which can be compared to a truncated Cartesian measurement of k-space. In contrast to Cartesian sampling, coverage is achieved by rotating the blade around the k-space origin between scans, resulting in a far denser sampling of low-frequency regions.

The rotation angle between consecutive blades is commonly chosen to be π/N. Alternatively, the *golden angle*[5] of 111.25° is employed, which aims to decorrelate motion between partial scans. Acquisition using the golden angle has been experimentally shown to be associated with better signal-to-noise ratio in MRI scans [13].

In accordance to Arfanakis et al. [14], sufficient sampling of the k-space for image reconstruction can be achieved in N shots by choosing

$$L = \left\lfloor \frac{C \cdot \pi}{2 \cdot N} \right\rfloor, \tag{2.17}$$

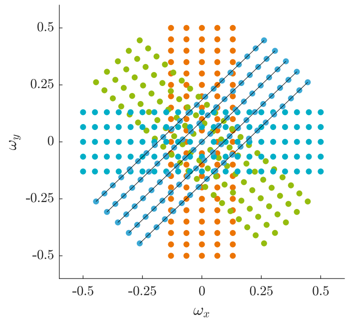
where L is the number of lines acquired in each blade, and C the number of coefficients sampled on each blade line. Equation (2.17) allows to calculate the optimal L, C or N, given that two of the values are known or already defined.

A sample PROPELLER scan trajectory (with 4 blades and L=5) is shown in Figure 2.3a as a scatter plot of the measured points in k-space[6]. Different blade acquisitions are shown in different colours.

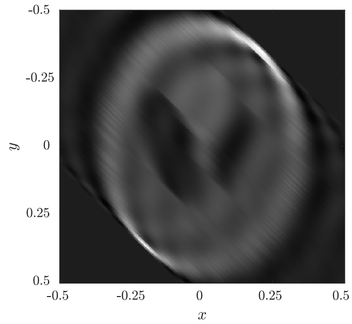
Since the scanning grids used to acquire PROPELLER blades are symmetric around the k-space origin (ignoring imperfections in the scan pattern), each scan

[5]The golden angle is a natural phenomenon observed in plant leaf growth. Each new leaf of certain plants is rotated with this angle in respect to the last leaf, which leads to an optimised layout for an arbitrary number of leafs.

[6]Note that the sampling shown would not suffice for acceptable reconstruction and is chosen for visualization purposes



(a) PROPELLER Trajectory



(b) Partial Image

Figure 2.3: a) A sample PROPELLER trajectory. Trajectories of the 4 blades in k-space are shown in different colours, with dots denoting measurement points. For a single blade, the actual 'path' through k-space is shown as a black line. b) The inverse Fourier transform of a single blade measurement yields a partial scan image (shown here for the Shepp-Logan phantom). Gibbs Ringing artifacts and significant blur is visible.

produces a real valued, low resolution image of the field of view. Of course, due to the hard band-limiting nature of the scan, significant blur and Gibbs ringing (see e.g. [15]) are present, oriented with the angle of the blade. Figure 2.3 shows such an image resulting from a single blade measurement of the Shepp-Logan phantom [16].

Since each blade overlaps on a central k-space region while yielding a real-valued image, this trajectory is a good candidate for motion correction algorithms. All experiments of this work use the PROPELLER trajectory with a varying number of blades.

2.3.3 Sampling Considerations and Linear Algebra Model

Since only a discrete number of coefficients are recorded in k-space, a discretised image can be reconstructed without further loss of information. Sampling considerations from signal processing theory apply here, which give boundaries for the number and distance of measured samples required for image reconstruction[7].

The sampled k-space relates to a discrete image by a discrete Fourier transform (DFT). In a Cartesian trajectory, let s_x equispaced phase encoding steps be captured with each measuring s_y samples on its readout line. This then implies reconstruction of a discrete image of size $s_x \times s_y$. The extend of the k-space sampling k_x^{max}, k_y^{max}

[7]Note that this is not related to the sampling required to acquire the induced magnetic resonance signal

signifies the maximal spatial frequencies of the continuos image captured by the measurement.

Let $\boldsymbol{I}$ be a flattened discretisation of $\tilde{\boldsymbol{I}}$ to s_p pixels, with s_x and s_y being the dimensions of the unflattened image and $s_p = s_x \cdot s_y$. Similarly, let $\boldsymbol{I}^{(n)}$ be one of N deformations of $\boldsymbol{I}$. This discretisation reduces the integral in Equation (2.16) to the sum

$$y^{(n,m)} = \sum_{j=1}^{s_p} \boldsymbol{I}^{(n)}(\boldsymbol{x}_j)\exp(-i2\pi\langle \boldsymbol{k}^{(n,m)}, \boldsymbol{x}_j\rangle) \tag{2.18}$$

where $\boldsymbol{x}_j$ is the j-th pixel location. This sum can be compactly expressed by the linear equation

$$\boldsymbol{y}^{(n)} = \boldsymbol{A}^{(n)}\boldsymbol{I}^{(n)} \tag{2.19}$$

using the matrix form of the Fourier sum to form a forward system matrix as

$$\boldsymbol{A}^{(n)} = \begin{bmatrix} a^{(1,n,1)}, & a^{(1,n,2)}, & \dots, & a^{(1,n,s_p)}) \\ a^{(2,n,1)}, & a^{(2,n,2)}, & \dots, & a^{(2,n,s_p)} \\ \vdots & \vdots & & \vdots \\ a^{(M,n,1)}, & a^{(s_p,n,2)}, & \dots & a^{(M,n,s_p)} \end{bmatrix}, \quad a^{(m,n,v)} = e^{-i2\pi\langle \boldsymbol{k}^{(n,m)}, \boldsymbol{x}_v\rangle}. \tag{2.20}$$

When ignoring motion and assuming a fully sampled Cartesian trajectory, the equation in Equation (2.18) is the DFT of an image, and a human-readable image can be recovered from k-space by the well-known inverse fast Fourier transform (iFFT).

In contrast to the fully sampled Cartesian trajectory, PROPELLER introduces some complications. Each blade can be accurately described as a rotated and truncated Cartesian sampling of k-space, from which a discrete image of size $L \times s_y$ with $s_y = C$ could be recovered via iFFT, even though such a reconstructed image would be of low resolution and affected by Gibbs ringing. Recovering a full image is less intuitive, as the collection of all captured points does not offer equispaced sampling of the k-space.

In PROPELLER and other, similarly non-equispaced k-space trajectories, the image is related to the sampling via the non-uniform Fourier transform of type 1 (NUDFT-I, also known as *nonuniform discrete Fourier transform*, or *NDFT*). The recovery of a static image involves solving

$$\boldsymbol{I} = \boldsymbol{A}^{-1}\boldsymbol{y}, \quad \text{with } \boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}^{(1)} \\ \boldsymbol{A}^{(2)} \\ \vdots \\ \boldsymbol{A}^{(N)} \end{bmatrix} \text{ and } \boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}^{(1)} \\ \boldsymbol{y}^{(2)} \\ \vdots \\ \boldsymbol{y}^{(N)} \end{bmatrix}. \tag{2.21}$$

This is exactly solvable as long as $\boldsymbol{A}$ is square, implying that the number of coefficients in both domains is the same, and non-singular - which is guaranteed if all sampling points are distinct. In most practical cases, $s_p \neq N \cdot M$, and a pseudo-inverse is used in place of the exact inverse.

Directly posing and inverting the measurement matrix has a complexity of $\mathcal{O}(s_p \cdot N \cdot M)$ and is not feasible for larger image sizes. However, fast algorithms exist for the NDFT transform, which are commonly known as *NFFT* in allusion to the FFT. NFFT either converts the non-uniform problem to a uniform grid before applying the FFT - a process known in MRI literature as regridding (for example in [17]). Another approach estimates a solution by an optimisation problem based on Equation (2.21). In this work, software libraries are used which efficiently invert the NDFT either using regridding operations [18], [19] or iterative methods [20], depending on the application.

The Shannon-Nyquist theorem [21] holds further implications for the sampling of k-space, in that it dictates boundaries for the size of the field of view of the continuos image. For a Cartesian trajectory, let Δk_x and Δk_y be the distances between sampled coefficients in k-space in readout and phase encoding directions. Then, the maximal field of view $\text{FOV}_{x,y}$ which can be captured by the scan without introducing aliasing artifacts is

$$\text{FOV}_{x,y} = \frac{1}{\Delta k_{x,y}}. \tag{2.22}$$

Interestingly, similar restrictions hold for non-equispaced sampling schemes: Following the results of Landau [22], [23], aliasing-free recovery of a field of view is possible for

$$\text{FOV}_{x,y} \leq \frac{1}{\Delta \bar{k}_{x,y}}, \tag{2.23}$$

where $\Delta \bar{k}_{x,y}$ is the mean sampling rate of k-space.

3
Machine Learning

Contents

Modern technology enables the acquisition of vast amounts of data for any given process. In many applications, designing a method to take advantage of this data is far more difficult. While it is comparatively easy to procure a large database of pictures of different animals, designing algorithms to reliably classify the animal depicted is very hard without a good way to adapt the process to the data distribution. Hand-crafted features offer very limited performance for such applications and would rely on the detection of simple shapes in the pictures.

An answer to the need of flexible, large-scale data processing comes in the form of Machine Learning (ML). ML - in a broad sense - consists of methods which are able to automatically recognize patterns in various forms of data, and use these patterns to solve a posed problem. Among many others, possible tasks could be detection problems, like analysing histological samples for signs of malignant growths, prediction problems like weather forecasting, or classification problems like the example stated above.

In most modern applications, ML algorithms utilize *artificial neural networks* (ANNs), which are directed graphs of mathematical functions with adaptable parameters. While ANNs can become arbitrarily large and complex, their building blocks are in most cases simple algebraic functions.

This chapter offers an overview of the building blocks, their connection and the adaptation process as it pertains to this thesis. Section 3.1 offers a brief overview of the background and biological motivation of the basics underlying ANNs, while Section 3.2 and Section 3.3 present the methods of adapting the networks to the underlying data. Section 3.4 gives an overview over the most common non-linear functions to connect blocks of an ANN, and Section 3.5 addresses the most common machine learning blocks in Computer Vision tasks and this thesis. Finally, Section 3.6 explains the attention mechanism - a recently developed technique from natural language processing for spatial feature extraction.

3.1 The Artificial Neuron and Dense Networks

The *artificial neuron* (AN) is the most basic building block inside a neural network. Its structure is inspired by biological neurons in animal neuron systems, which combine electrical inputs emitted by other, connected neurons. Additionally, these cells are able to *learn* a weighting of incoming signals, which is adapted based on the strength of past inputs according to to Hebb's rule [24]. As shown by McCulloch and Pitts in 1943 [25], directed graphs of idealized neurons can be used to model different logic gates and thresholding functions, and can therefore be utilized to carry out calculations. In the works of Rosenblatt [26], the AN was combined with the self-actualization rules described by Hebb to form the idea of a *perceptron*, using the adaptable behaviour of the AN in computational tasks.

An AN mimics the behaviour of a biological neuron's ability to make a binary decision (to fire a signal or not to fire a signal) depending on the signals inbound to its cell body. The AN considers an input vector $\boldsymbol{x}$ whose entries are weighted with adaptable scalar weights $\boldsymbol{\omega}$ and summed with a scalar bias b. The result is used to drive an activation function $\mathrm{a}(\cdot)$ which defines the output of the artificial

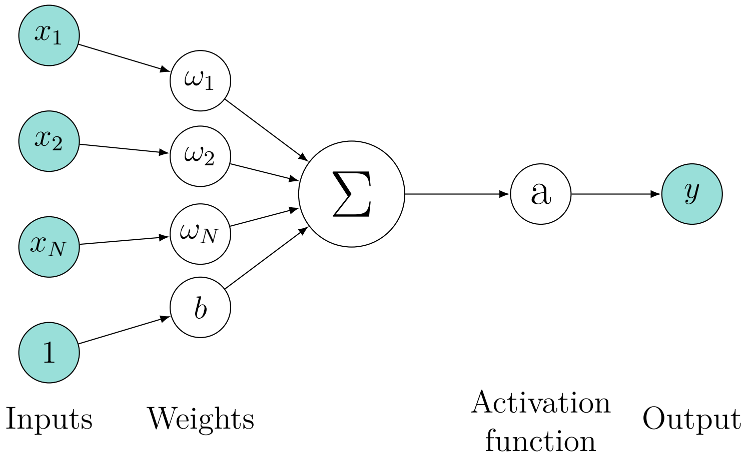


Figure 3.1: Visualisation of an artificial neuron with N inputs (x_1 to x_N), their associated weights (ω_1 to ω_N) and bias b.

cell. The scalar output y of a single-layer perceptron can be compactly defined based on input- and weighting vectors with

$$y = \mathrm{a}\left(\boldsymbol{\omega}^\intercal \boldsymbol{x} + b\right), \tag{3.1}$$

while Figure 3.1 shows this inner working schematically.

In the seminal research of Rosenblatt et al. [26], the perceptron was designed to be a binary classifier with a decision boundary $0 = \boldsymbol{\omega}^\intercal \boldsymbol{x} + b$. This is achieved by defining the activation function as the Heaviside step function

$$\mathrm{a}_H(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}, \tag{3.2}$$

resulting in an output $y = 1$ iff $\boldsymbol{\omega}^\intercal \boldsymbol{x} + b > 0$. Infamously, the perceptron is limited by its inability to model a non-linear separation of classes [27].

This failing directly led to the development of multilayer networks, which are guaranteed to be able to approximate any given underlying function as long as they consist of more than a single layer of perceptrons. [28][1].

Figure 3.2 shows a simple, four-layered feed-forward neural network. It is composed of an *input* layer introducing the inbound vector $\boldsymbol{x} \in \mathbb{R}^3$ into the network, two *hidden* layers producing intermediary steps and an *output* layer which produces

[1]Although the universal approximation theorem allows approximation of arbitrary functions, equalness might require infinite depth *or width* of the network.

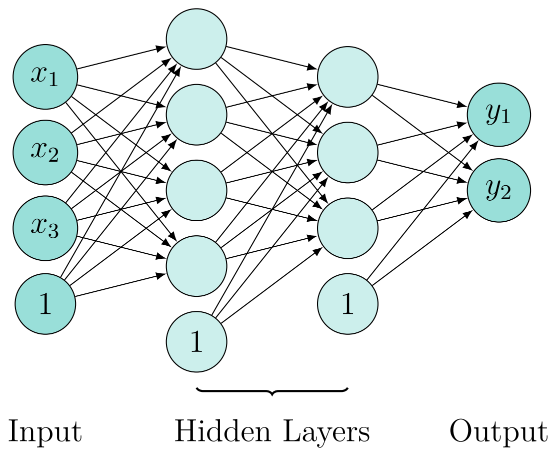


Figure 3.2: Visualization of a simple artificial neural network with two hidden layers, an input $\boldsymbol{x} \in \mathbb{R}^3$ and output $\boldsymbol{y} \in \mathbb{R}^2$.

the output vector $\boldsymbol{y} \in \mathbb{R}^2$. The hidden layers used in this example are known as *fully connected* (FC) layers, as each unit receives an input from all neurons in the previous layer, and passes information to all units in the next layer. The output of the hidden layers is usually not evaluated directly, but can be used for some advanced regularisation schemes. The activation function is no longer a thresholding operator, but can be chosen as any differentiable, bounded, non-linear function which is applied on each vector element individually.

Since each layer of the multilayer network is composed of stacked perceptrons, each unit in the layer produces an output of the form

$$x_k^{(j)} = \mathrm{a}\left(\sum_{n=1}^{N^{(j-1)}} \left[\omega_{k,n}^{(j)}\, x_n^{(j-1)}\right] + b_k^{(j)}\right), \tag{3.3}$$

where j is the index of the current layer, k is the index of a single unit in this layer, and $N^{(j-1)}$ is the number of neurons in the previous layer. This implies that the jth layer receives values from $N^{(j-1)}$ units of its predecessor. $x_k^{(j)}$ is the output value of the k-th unit in the j-th layer, which takes the values $x_1^{(j-1)}, x_2^{(j-1)}, ..., x_{N_{j-1}}^{(j-1)}$ as inputs.

The behaviour of a layer can be expressed in a single multiplication by grouping the weights of layer j into a matrix $\boldsymbol{W}^{(j)} \in \mathbb{R}^{N_j \times N_{j-1}+1}$ as

$$
\boldsymbol{W}^{(j)} = \begin{bmatrix} \omega_{1,1}^{(j)} & \omega_{1,2}^{(j)} & \dots & \omega_{1,N^{(j-1)}}^{(j)} & b_1^{(j)} \\ \omega_{2,1}^{(j)} & \omega_{2,2}^{(j)} & \dots & \omega_{2,N^{(j-1)}}^{(j)} & b_2^{(j)} \\ \vdots & & & & \\ \omega_{N^{(j)},1}^{(j)} & \omega_{N^{(j)},2}^{(j)} & \dots & \omega_{N^{(j)},N^{(j-1)}}^{(j)} & b_{N^{(j)}}^{(j)} \end{bmatrix} \tag{3.4}
$$

and the layer inputs into a vector $\boldsymbol{x}^{(j)} \in \mathbb{R}^{N_{j-1}+1}$ as

$$
\boldsymbol{x}^{(j-1)} = \begin{bmatrix} x_1^{(j-1)} \\ x_2^{(j-1)} \\ \vdots \\ x_{N^{(j-1)}}^{(j-1)} \\ 1 \end{bmatrix}. \tag{3.5}
$$

Using this notation, the output of layer j can be expressed with a function $f : \mathbb{R}^{N^{(j-1)}} \times \mathbb{R}^{N^{(j)}} \times \mathbb{R}^{N^{(j-1)}+1} \to \mathbb{R}^{N^{(j)}}$ as

$$
\begin{aligned} \boldsymbol{x}^{(j)} &= \mathrm{a}(\ \boldsymbol{W}^{(j)}\boldsymbol{x}^{(j-1)}\ ) \\ &= f(\boldsymbol{x}^{(j-1)}, \boldsymbol{W}^{(j)}). \end{aligned} \tag{3.6}
$$

Then, the whole network can be summarised succinctly by chaining the expressions for all available layers with

$$
\begin{aligned} \boldsymbol{y} =& \mathrm{a}(\boldsymbol{W}^{(3)}\mathrm{a}(\boldsymbol{W}^{(2)}\mathrm{a}(\boldsymbol{W}^{(1)}\boldsymbol{x}^{(0)}))) \\ =& f(\qquad f(\qquad f(\boldsymbol{x}^{(0)}, \boldsymbol{W}^{(1)}), \boldsymbol{W}^{(2)}), \boldsymbol{W}^{(3)}) \end{aligned} \tag{3.7}
$$

where $\boldsymbol{x}^{(0)}$ is the network's input appended with the scalar value of 1. Defining a set of available weights as $\boldsymbol{W} = \{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)}, \dots \boldsymbol{W}^{(M)}\}$, the entirety of a M-layered multilayer network can be expressed as

$$
\begin{aligned} \boldsymbol{y} &= \hat{f}(\boldsymbol{x}^{(0)}, \boldsymbol{W}) \\ &= f(f(\dots f(\boldsymbol{x}^{(0)}, \boldsymbol{W}^{(1)}), \dots, \boldsymbol{W}^{(M-1)}), \boldsymbol{W}^{(M)}). \end{aligned} \tag{3.8}
$$

3.2 Learning and Network Training

As previously mentioned, ANNs aim to approximate an unknown function by *learning* optimal values for their set of weights. The learning problem can be formulated as an optimisation problem for which a differentiable objective function (the *loss function*) is minimized. The optimal parameter set minimizing the loss function of a specific task strongly depends on the individual problem. The choice and design of the loss function therefore shapes the network output, and has to be carefully constructed to properly capture the difference of a given output to the optimal distribution. Let $\hat{\mathcal{L}}(\boldsymbol{W})$ be the loss of the network calculated for the weight set $\boldsymbol{W}$ and a given input distribution. The optimal set of weights is then given by

$$\boldsymbol{W}_{opt} = \arg\min_{\boldsymbol{W}} \hat{\mathcal{L}}(\boldsymbol{W}). \tag{3.9}$$

In applied machine learning, the true input distribution is unknown and has to be approximated using a finite number of data points. Therefore, a more discrete approach is required in place of the continuos formulation above. The set of available input data points with the associated ground truth - the known, optimal output value for any data point in the set - is contained in the *dataset* $\mathcal{D}$, which has to be representative for the underlying data distribution. Then, a gradient descent algorithm can be used to minimize the overall loss for $\mathcal{D}$, so that the network output approximates the true data distribution.

Depending on the data set, machine learning models can be trained in a *supervised* or *unsupervised* fashion. Whereas unsupervised learning requires no prior knowledge of an optimal output of the network for a given input point, supervised learning requires knowledge of the *ground truth*. In the use case of medical applications, the ground truth is created manually by a staff of experts. Supervised training therefore requires more specialised data sets, which are often created specifically for a given task.

In the beginning of the network fitting procedure, all weights of the network are instantiated using one of many initialization algorithms. Common strategies for shallow network include using random, Gaussian distributed values or uniform values. Deeper networks employ random values weighted according to local network topology [29], [30] to optimise the convergence rate of the used gradient descent algorithms.

With known ground truth $\boldsymbol{y}^{gt}$ and data points $\boldsymbol{x}$ in the data set, the optimisation of the weight set $\boldsymbol{W}$ is accomplished by minimizing a differentiable, scalar distance function $L(\hat{f}(\boldsymbol{x}, \boldsymbol{W}), \boldsymbol{y}^{gt})$ for each point in $\mathcal{D}$. The optimal weight set for $\mathcal{D}$ is found by

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{N} \sum_{n=1}^{N} L(\hat{f}(\boldsymbol{x}_n, \boldsymbol{W}), \boldsymbol{y}_n^{gt}) \tag{3.10}$$

$$\boldsymbol{W}_{opt} = \arg\min_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}). \tag{3.11}$$

Here, $\mathcal{L}(\boldsymbol{W})$ denotes the mean error of the network on the N available data points in $\mathcal{D}$ for the current weight configuration. With a small step size β, a gradient descent optimisation algorithm can be defined with

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \beta \frac{\partial \mathcal{L}(\boldsymbol{W})}{\partial \boldsymbol{W}}. \tag{3.12}$$

Because all parts of a network are designed to be differentiable[2], a gradient can be found for each trainable component. The gradient optimisation on the full data set is referred to as *batch gradient descent*. Using this formulation, the gradient descent is guaranteed to reach either a local minimum or a saddle point as long as the input data is bounded and the step size β is sufficiently small.

However, computing the gradient of the network is a highly memory intensive task even for a single point of data. Memory size constraints quickly become apparent when using batch gradient descent, especially when using graphic processing units for parallelization. *Stochastic gradient descent* (SGD) circumvents this limitation by updating the network weights using the gradient for a single data point, and iterating over all points in $\mathcal{D}$. Local gradients tend to lead to a noisy convergence, but move in the direction of the true gradient on average. They also can handle larger step sizes and therefore lead to faster convergence and avoid optimisation to local minima [31].

Modern computing hardware is highly optimised for parallelized calculations and enables the SGD to be sped up considerably by grouping several data points together. In *mini-batch gradient descent* (MGD), the sum of gradients for a small

[2]Some discontinuities might exist, like in $\mathrm{ReLu}(\cdot)$ or $\max(\cdot)$ functions, but this is unproblematic in application as the gradients are well-behaved in their proximity.

subset of the available training points is used in the update step. Since the update for the mini-batch can be calculated in parallel for its data points, this leads to a considerable increase in speed and smoothens the noisy local gradient estimates.

To smooth convergence even further, modern SGD and MGD algorithms not only use the local gradient, but choose an adaptive learning rate for each weight based on an approximated gradient momentum, accelerating the optimisation in common gradient directions. One of the most commonly used optimisation algorithms is the *adaptive moment estimation algorithm* (ADAM) [32]. ADAM and its extensions use estimates for first and second order moments, which has been shown empirically to lead to faster convergence compared to standard SGD and momentum-free variants [33].

In most cases, a network can be optimised to fit to the data ensemble it is presented with during training, but that does not necessarily imply the same performance on the true data distribution. In most cases, it is impossible to precisely assess the fit to the true distribution, but approximate methods exist. Commonly, the set of available data points is split randomly into a *training* subset $\mathcal{D}_{Train}$ and a *validation* subset $\mathcal{D}_{Val}$ so that $|\mathcal{D}_{Train}| \gg |\mathcal{D}_{Val}|$, $\mathcal{D}_{Train} \cap \mathcal{D}_{Val} = \emptyset$ and $\mathcal{D}_{Train} \cup \mathcal{D}_{Val} = \mathcal{D}$. The model is fitted only on $\mathcal{D}_{Train}$, and its performance is evaluated on $\mathcal{D}_{Val}$ in regular intervals during training. The performance on $\mathcal{D}_{Val}$ is used as an indicator for real world performance[3].

This splitting technique still assumes that both $\mathcal{D}_{Train}$ and $\mathcal{D}_{Val}$ are representative of the true distribution. A computationally more intensive, but more robust approach is *k-fold cross* validation. In k-fold cross validation, the data set is shuffled and split into k subgroups $\mathcal{D}_k$, each containing a $1/k$ fraction of the data points. A total of k models are fit for the given architecture, with the nth model being fit on $\mathcal{D}\backslash\mathcal{D}_n$ and evaluated on $\mathcal{D}_n$. The collected evaluation results for all k trained models form the overall performance for the architecture on the entire data set.

k-cross validation has the benefit that all data samples are a part of the validation subset once. This makes it possible to more robustly approximate the network architectures ability to generalise to unseen data. The obvious drawback is the k-fold increase in computation time in comparison to a single train-validation split.

[3]In practice, a separate *test* dataset is also often which is used to check model performance after the training process has concluded. This is omitted in this work.

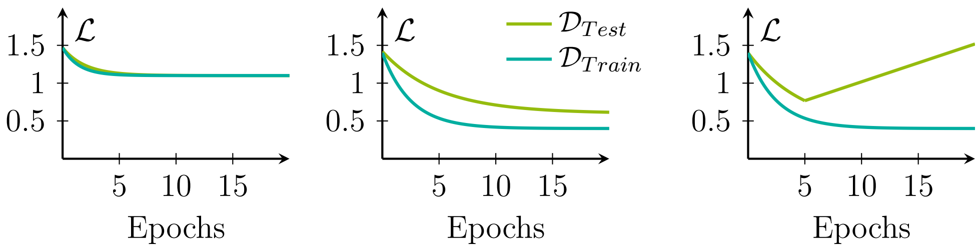


Figure 3.3: A graphic representation of network convergence. Left: Underfitting, middle: Healthy convergence over approx. 20 epochs, right: Overfitting after 5 epochs.

Independent from the chosen strategy, the training procedure is broken down into a number of *epochs*. During a single epoch, $\mathcal{D}_{Train}$ is shown to the network once, a weight gradient for all data points is calculated, applied by the chosen optimiser and the network performance is evaluated on $\mathcal{D}_{Val}$.

In this process, some design considerations may become apparent. Observing the empirical error on $\mathcal{D}_{Train}$ and $\mathcal{D}_{Val}$, the desired outcome is for both values to converge over a number of epochs. Typically, the performance measured on the training is slightly better. If the network design is too 'powerful' and has more trainable parameters than necessary to describe the data distribution, it might memorise the training data samples instead of fitting the distribution. This *overfitting* on the training data results in good performance on $\mathcal{D}_{Train}$, but an increased error or divergence on the unseen data points in $\mathcal{D}_{Val}$. If, on the other hand, the network is designed with too few parameters to fit the data distribution, no good approximation may be found. This so-called *underfitting* results in low performance in both training and validation phase. These behaviours are shown graphically in comparison to a suitable network size in Figure 3.3.

3.3 Backpropagation

As shown in Section 3.2, the objective error function of a neural network is a composition of different convex terms and can be minimized using various gradient descent algorithms. However, since the network topology varies greatly between approaches, and the number of parameters is typically large, analytic determination of the associated gradients is not viable. This motivated the development of an algorithmic way of calculating a gradient through arbitrary networks.

Since each part of the network is differentiable, the chain rule can be applied to compute the gradient through the network. This *automatic differentiation* [34] is the foundation of the *backpropagation* algorithm used in ANNs [35].

As shown in Equation (3.7), an N-layered network with input $\boldsymbol{x}$ and output $\boldsymbol{y}$ can be expressed as the chain of operations $\boldsymbol{y} = f^{(N)}(f^{(N-1)}(\dots f^{(1)}(\boldsymbol{x})\dots))$ using the shorthand $f^{(j)}(\boldsymbol{x}^{(j)}) = f(\boldsymbol{x}^{(j)}, \boldsymbol{W}^{(j)})$ to denote the j-th layer together with its hidden input $\boldsymbol{x}^{(j)}$.

Given a ground truth $\boldsymbol{y}^{gt}$ for $\boldsymbol{x}$, the gradient of the loss function $L(\boldsymbol{y}, \boldsymbol{y}^{gt})$ with respect to the weights of the previous network layer is found by applying the chain rule:

$$\frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial \boldsymbol{W}^{(N)}} = \frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial f^{(N)}} \cdot \frac{\partial f^{(N)}}{\partial \boldsymbol{W}^{(N)}} \tag{3.13}$$

Deriving for $\boldsymbol{W}^{(N-1)}$, the chain rule is used again to split the second term:

$$\frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial \boldsymbol{W}^{(N-1)}} = \frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial f^{(N)}} \cdot \frac{\partial f^{(N)}}{\partial \boldsymbol{W}^{(N-1)}} \tag{3.14}$$

$$= \frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial f^{(N)}} \cdot \frac{\partial f^{(N)}}{\partial f^{(N-1)}} \cdot \frac{\partial f^{(N-1)}}{\partial \boldsymbol{W}^{(N-1)}} \tag{3.15}$$

This process is repeated for the next layer:

$$\frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial \boldsymbol{W}^{(N-2)}} = \frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial f^{(N)}} \cdot \frac{\partial f^{(N)}}{\partial f^{(N-1)}} \cdot \frac{\partial f^{(N-1)}}{\partial \boldsymbol{W}^{(N-2)}} \tag{3.16}$$

$$= \frac{\partial L(\boldsymbol{y}, \boldsymbol{y}^{gt})}{\partial f^{(N)}} \cdot \frac{\partial f^{(N)}}{\partial f^{(N-1)}} \cdot \frac{\partial f^{(N-1)}}{\partial f^{(N-2)}} \cdot \frac{\partial f^{(N-2)}}{\partial \boldsymbol{W}^{(N-2)})} \tag{3.17}$$

and all subsequent layers in the network. Note that parts of the gradient term repeat, which can be exploited for efficiency using dynamic programming.

Through the repeated application of the chain rule the gradient in each layer is a product composition of derivatives from higher layers. This can pose a problem for networks with particularly deep structures. When the optimisation of a single layer of the network approaches a (local) minimum or a saddle point, the value of its gradient approaches zero. Since the gradient affected by this is still reused in the product chains of deeper layers, later gradients in the chain in turn vanish as well. This is known as the *vanishing gradients* problem [36].

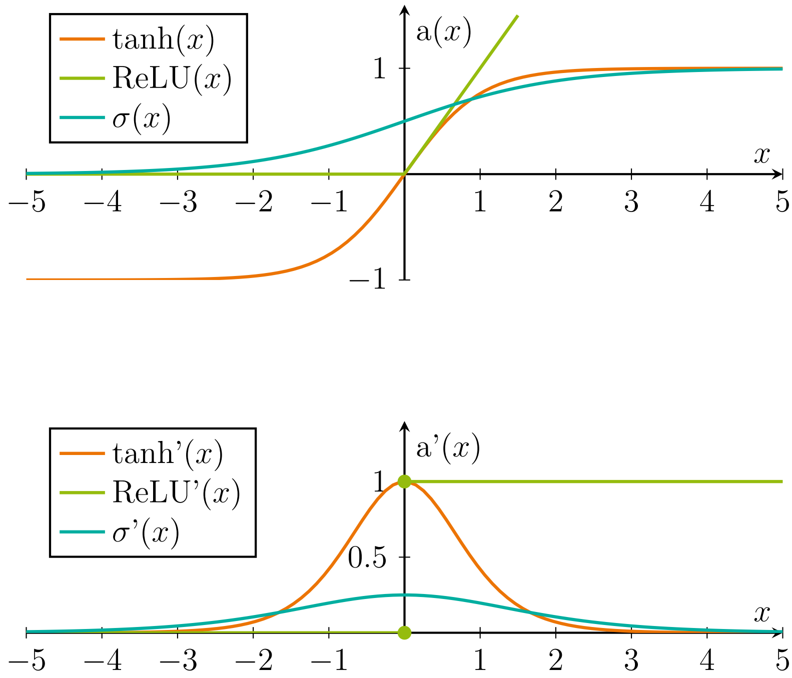


Figure 3.4: Three commonly used nonlinear activation functions for artificial neuronal networks (top), and their respective derivatives (bottom).

3.4 Nonlinearities

As mentioned in Section 3.1, a single-layered fully connected network is not able to model nonlinear functions. If the activation function a(·) is omitted or set to a linear term in the multilayer network function as described in Equation (3.7), the network would collapse, as the chain of linear matrix multiplications is a linear function itself. This motivates a(·) to be a non-linear function, which only has to be differentiable to work within the neural network framework.

There are many different functions available and in use for different problem settings. This section gives a short overview over the most commonly used activation functions, which are also shown grapically in Figure 3.4.

For training networks to estimate probabilites, it is common to use the sigmoid function $\sigma(x)$ or the hyperbolic tangent function $\tanh(x)$ [37]. Both functions are modeled after the biological neuron, which acts as a threshold and fires its activation potential when the sum of inbound potentials from connected cells is strong enough. However, both run into problems for high output values, as they and their associated gradients get squashed by the activation function.

For modern regression and especially image processing networks, the *rectified linear unit* (ReLU) function [38] or one of its variants is more commonly applied. It is unbounded and its gradient has a discontinuity at $x = 0$, but its ability to provide a strong gradient (for all positive values) and its computational simplicity make it suitable for many applications. These properties also made it the de-facto standard for convolutional networks[4].

In regards to network performance, the ReLU function and its variants beat all comparable network activation functions in terms of accuracy and convergence speed in image classification tasks [40]. Together with its prevalence in related research, its use in the networks in this work is motivated by these findings.

In non-regressesion tasks such as classification, the output of a network is often interpreted as a probability function. For example, the output of a network designed for a N-class discrimination problem can consist of a vector $\in \mathbb{R}^N$. Its value at position n is interpreted as the probability that the input belongs to class n. The Softmax function $\text{Softmax} : \mathbb{R}^N \to (0,1)^N$ normalizes its input values, for example the output of a linear embedding, and converts it to a valid probability distribution so that it sums to 1 and is positive semidefinite. As such, the function is an outlier in the set of activation functions, as it is not scalar, but vector-valued. The Softmax function is defined for an input vector $\boldsymbol{v}$ as

$$\text{Softmax}(\boldsymbol{v}) = \frac{\exp \boldsymbol{v}}{\sum \exp \boldsymbol{v}}, \tag{3.18}$$

where the exponential is computed for each element of $\boldsymbol{v}$. For matrices or tensors, the Softmax is conventionally computed for each row separately. The term 'softmax' describes the behaviour of the function: Values at (or near) the maximum value of the input are assigned much larger output probabilities by the exponential function. Typically, the output probability distribution is highly concentrated at one position, so that the function acts a sort of 'soft', differentiable argmax-function.

[4]It is possible for a network to suffer a so-called *Dead ReLU*. If the bias of a cell using ReLU turns strongly negative, the output of that cell - and its gradient - will be functionally disabled. A workaround is found in the (generally more computationally expensive) ReLU variants [39]. In this work, no dead ReLU-problems were identified, so the standard ReLU is used.

3.5 Convolutional Layers and Pooling Operations

In most image processing applications including this work, the data considered does not consist of vector-valued signals, but multi-dimensional images. For applications using medical image data, the network is tasked to process an at least two-dimensional greyscale image $\boldsymbol{I} \in \mathbb{R}^{s_x \times s_y \times 1}$. Up to this point, only fully connected layers were considered as trainable network structures. However, especially in the context of image processing, those FC layers bear two important problems:

- Since the size of the layer weight matrix depends on its input and output size, FC layers are unable to handle large scale input data, such as high-resolution images or audio signals. For example, a very simple image denoising network utilizing a single hidden FC layer for grayscale images $\boldsymbol{I}$ would incorporate a weight matrix $\boldsymbol{W} \in \mathbb{R}^{(s_x \cdot s_y) \times (s_x \cdot s_y)}$. Even on modern hardware, the memory requirements are therefore challenging for most image sizes.

- Inherently, FC layers are highly affected by shifts in input data. A translational shift of a single pixel in the input would be interpreted as a completely new sample, whereas these shifts are inconsequential for data types such as audio signals or images. This presents a high risk for overfitting, as the learned distribution would not generalize to the true distribution where such shifts occur naturally.

A solution comes in the application of *convolutional* operations in the network in place of the matrix-vector multiplication of the classic FC layer [38].

The intuition is once again borrowed from biology. Sensory cells in the eyes of higher animals are clustered together into so-called *receptive fields*, which activate one or few visual neurons [41]. This is understood to allow the visual nervous system to directly receive spatial features encoded by the 'wiring' of the light receptors. The optical receptive field allows for a kind of preprocessing before the visual information is analysed by higher brain functions, in which important shapes are recognized by the receptive fields regardless of their position in the seen image.

In a artificial neural network setting, this behaviour can be replicated by convolving the input image with a small learned weight kernel and adding a bias value to be able to model an offset shift. Together with the convolution operation, the learned weights form the *convolutional layer*. Convolutional layers are not only memory efficient (as the learned kernel is far smaller than a weight matrix of a fully connected layer) but also exhibit an inherent *shift invariance*. This way, a shifted input to a convolution causes a similarly shifted output. This for most image processing applications highly desirable quality is a key feature of convolutional layers.

As one might expect, the core idea of convolutional layers is the use of a convolution sum. Let the image or feature map $\boldsymbol{I} \in \mathbb{R}^{s_x \times s_x \times C_i}$ be the three-dimensional input to a convolutional layer, where s_x and s_y are the spatial dimensions and C_i the number of its channels. Convolutional layers use adaptable four-dimensional weight kernels $\boldsymbol{W}_c \in \mathbb{R}^{k \times k \times C_i \times C_o}$ with $k \ll s_x, s_y$ to compute a convolution $*$ as a sum of the form

$$\boldsymbol{I}_{out}[m,n,c] = \boldsymbol{W}_c * \boldsymbol{I} = \sum_{u=1}^{C_i} \sum_{v=1}^{s_x} \sum_{p=1}^{s_y} \boldsymbol{W}_c[u,v,p,c] \cdot \boldsymbol{I}[m-u, n-v, p]. \tag{3.19}$$

In this formulation, the area outside of the kernel support is assumed to be zero and the learned bias is omitted. This is a straightforward extension of a two-dimensional convolution - note that there is no shift in the third dimension of the input. In terms of image convolutions, which might be more familiar, each input channel is separately convolved with a filter kernel and the convolution results are summed up. The operation returns a two-dimensional output for a three-dimensional input and filter.

This process is repeated for C_o separate convolution kernels, which form the number of output channels.

The discrete convolution as defined above produces an output tensor $\boldsymbol{I}_{out}$ of size $(k + s_x - 1) \times (k + s_y - 1) \times C_o$. This work exclusively uses non-expansive convolutions, in which the convolution output is cropped to the original size of $s_x \times s_y \times C_o$ size. A graphic representation of this convolution is given in Figure 3.5, where $C_i = 1$, $C_o = 1$ for simplicity.

Since convolutions are differentiable in respect to image and kernel [42], the weights of the convolution layers can be trained with the backpropagation algorithm presented in Section 3.3.

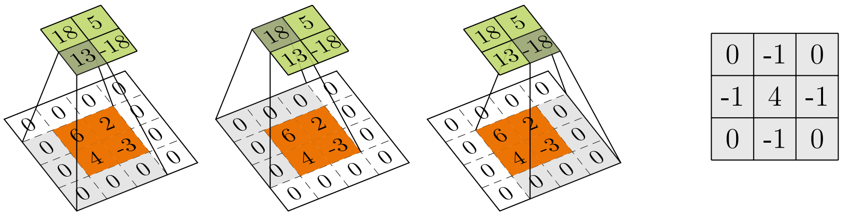


Figure 3.5: A non-expansive convolution operation using a $3 \times$ kernel (gray) over a $2 \times$ input (orange) produces a 2×2 output (green)

A network relying on convolutional layers ideally would be able to process large or coarse image structures, because in natural images, low-frequency features tend to carry a lot of important information. Using a single convolution operation to analyse such structures would require large kernels, which in turn would require more training.

Subsampling layers, such as the *max-pooling* layer, instead reduce the size of the incoming data. Max-pooling layers, in particular, aim to keep the most important information while reducing data size. Hence, they only keep the largest numerical value from small neighbourhoods in the data. The size of the neighbourhood as well as their distance to each other controls the behaviour of this layer. In this work, non-overlapping neighbourhoods of size 2×2 are used, resulting in a halving of input data size. This operation is used on each channel independently, preserving the number of channels of the feature map.

A graphic example of a max-pooling operation is given in Figure 3.6.

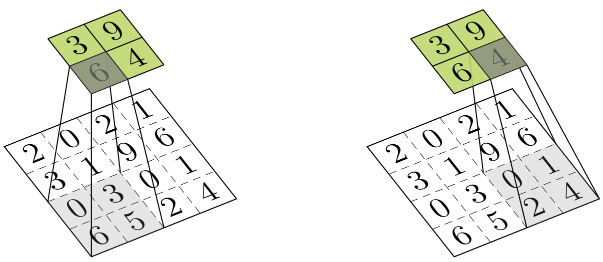


Figure 3.6: A 2×2 max-pooling operation on a single-channel feature map. The feature map size is reduced by a factor of 2 while its *important* aspects (measured by value) are kept.

To restore the size of the feature map after further processing, several options are available, such as nearest-neighbour interpolation. A commonly used data driven approach is to upsample the data and convolve with a learnable filter kernel to fill in the resulting gaps. This, in a machine learning context, is compactly expressed as a *transposed convolution*. Here, the feature map is upsampled according to a specified stride length and convolved with an appropriately sized filter. The result however is not an interpolation in the classical sense, as the used filter will not necessarily exhibit the required frequency domain behaviour.

A graphic example of the transposed convolution (again shown on a single feature map and channel, for simplicity) is shown in Figure 3.7.

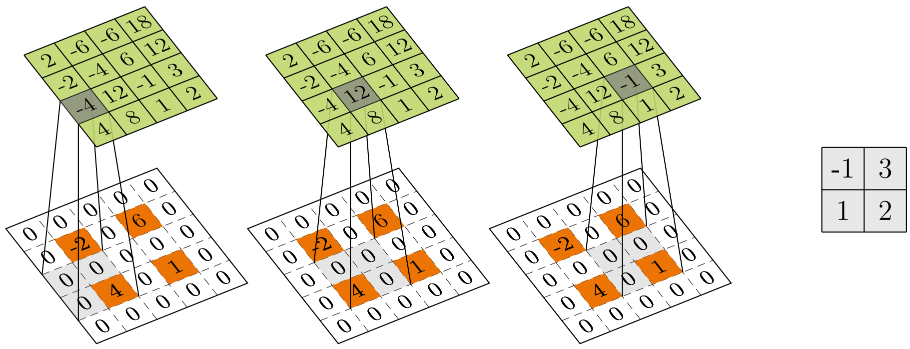


Figure 3.7: A transposed convolution using a 2 × 2 kernel over a 2 × 2 input using a stride of 2 results in a 4 × 4 output. This operation is the same as convolving an ↑ 2 upsampled and padded input with a 2 × 2 kernel. The used kernel is shown on the right.

Similar to convolutional layers, the operation of transposed convolution together with its learnable weight kernel is grouped into a *transposed convolutional layer*. These three building blocks of convolutional layer, max-pooling and transposed convolutional layer allows for the construction of fully convolutional networks for multi-dimensional signal processing. A simple end-to-end denoising network architecture is shown in Figure 3.8 while more advanced networks using these layers will be discussed in upcoming chapters.

3.6 Attention Mechanisms

A recent advance in neural network research is the so-called *attention* mechanism, which created a major performance improvement especially in natural language processing architectures.

In language processing, a foundational challenge is the encoding and decoding of variable-length sequences. Machine translation, for example, needs to convert arbitrarily long sentences from a source to a target language, with the sentence

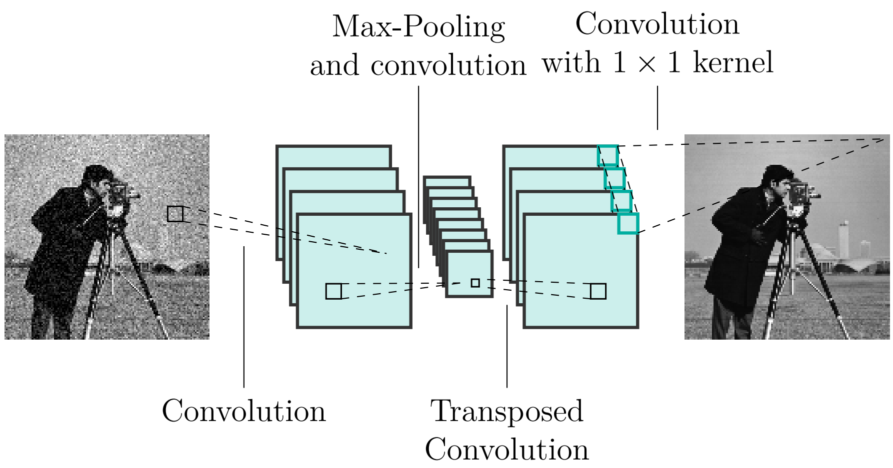


Figure 3.8: An example of a simplistic convolutional architecture for image denoising. A noisy image is fed into a convolutional layer with 4 channels, the feature maps are downsampled in a max-pooling layer (displayed here as a single operation) , and upsampled and filtered again in a transposed convolution layer. The resulting feature maps are reduced to a single-channeled image by a convolution with a 1×1 kernel and a output channel dimension of one.

length possibly being different in both languages. This task can therefore not depend on sentence length and must incorporate some form of contextual information to deal with grammatical differences and language quirks such as homonyms.

Before the advent of attention, recurrent networks were the state of the art for machine translation. In recurrent architectures, a state is preserved between sequential outputs, enabling the network to form spatial relationships among values in a sequence. This technique was incorporated into sequence-to-sequence networks, which formed the basis for early neural machine translation models [43], [44]. These *encoder-decoder* models incorporate separate recursive layers for encoding a source sentence into a latent space and decoding into a target language. Previously decoded words serve as additional inputs for the model.

However, there are a number of shortfalls in this approach. Since recurrent models are based on repeated application of a set of weights, they are highly sensitive to the scaling imposed by those weights. This makes recurrent models vulnerable to vanishing and exploding gradient problems. Additionally, even though they can effectively model spatially close-range relations, their long gradient paths in conjunction with difficult scaling make modeling long distance relationships in the input data difficult and intransparent. Sutskever et al. [43] for example note that the translation accuracy of their recurrent model improves by reversing the source sentence in respect to the target sentence, without being able to offer a conclusive explanation.

The attention mechanism remediates these problems by involving the spatial context of the entire input sequence into the encoding and decoding steps. First proposed as a way to mitigate problems in modelling arbitrary long sequences [45], it has since been used extensively in natural language processing, most famously in large language models (e.g. [46]–[48]). These networks specifically use the transformer architecture from Vaswani et al. [49].

Attention in machine learning is based on a different approach than conventional network layers. Attention, as defined by Bahdanau et al. [45] and Vaswani et al. [49], connects two sequences of input tokens by acting as a differentiable dictionary lookup. One sequence is encoded into dictionary *keys* and associated *values*, the other one into *queries*. A specific query is *matched* against all keys and the associated value is returned. The matching function typically consists of a scaled inner product between key and value entries, though other methods exist. Self-attention is a special case of attention in which the two input sequences are the same.

The concept of attention and self-attention as presented in [49] can be expressed as a combination of three different embeddings. Starting from two input sequences $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M]$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_M]$ with each $\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathbb{R}^{d_{in}} \forall 1 \leq i \leq M$, *Value* ($\boldsymbol{V}$), *Key* ($\boldsymbol{K}$) and *Query* ($\boldsymbol{Q}$) sequences are generated in a data driven process. Typically, this is implemented as a linear embedding using corresponding learned matrices $\boldsymbol{W}_V$, $\boldsymbol{W}_K$ and $\boldsymbol{W}_Q$ through

$$\boldsymbol{K} = \boldsymbol{W}_K \boldsymbol{X} \tag{3.20}$$

$$\boldsymbol{V} = \boldsymbol{W}_V \boldsymbol{X} \tag{3.21}$$

$$\boldsymbol{Q} = \boldsymbol{W}_Q \boldsymbol{Y}. \tag{3.22}$$

Here, $\boldsymbol{K}, \boldsymbol{Q} \in \mathbb{R}^{d_k \times d_{in}}$ and $\boldsymbol{V} \in \mathbb{R}^{d_{model} \times d_{in}}$, where d_k is the number of entries in the attention 'dictionary' of size d_{model} and d_{in} is the size of entries in the input sequences.

As described above, Keys and Queries are compared using a matching function to generate an alignment score. Several matching functions have been developed and are in use in different research fields and applications. This work focusses on the *scaled-dot product* attention, which captures the inner product of the vector embeddings contained in $\boldsymbol{Q}$ and $\boldsymbol{K}$ scaled by the number of embeddings as

$$\text{match}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{\boldsymbol{Q}\boldsymbol{K}^\intercal}{\sqrt{d_k}}. \tag{3.23}$$

To prevent an unchecked increase in parameter amplitude, the scores are scaled so that the row sum equals 1. The idea is to preserve the biggest value in each row, interpreted as the closest match between a key and query vector, but not forcing a winner-takes-it-all approach. This is compactly expressed by converting the matching score to a probability distribution with the softmax-function already discussed in Section 3.4:

$$\boldsymbol{S} = \text{Softmax}\left(\boldsymbol{Q}, \boldsymbol{K}\right). \tag{3.24}$$

Finally, a linear weighting of the attention score $\boldsymbol{S}$ and the embedded values $\boldsymbol{V}$ is performed to form the output of the attention head:

$$\boldsymbol{B} = \boldsymbol{S}\boldsymbol{V}. \tag{3.25}$$

The matrix $\boldsymbol{B} \in \mathbb{R}^{M \times d_{model}}$ is therefore a weighted sum of the values, scaled by a computed relevance score between entries in the input sequence.

In the transformer architecture, the attention mechanism is applied multiple times in each self-attention layer, forming *multi-head attention.* Multiple attention heads are computed in parallel for the input values $\boldsymbol{X}$ and $\boldsymbol{Y}$, which are commonly split into subsets to be processed by the separated self-attention functions and fused back together after each attention score has been calculated.

Let $\text{Att}(\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V})$ denote a single attention head. Multihead attention then combines N heads into a single output:

$$\text{multihead}(\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}) = \boldsymbol{W} \begin{bmatrix} \text{Att}_1(\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V})^\intercal \\ \text{Att}_2(\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V})^\intercal \\ \vdots \\ \text{Att}_N(\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V})^\intercal \end{bmatrix}. \tag{3.26}$$

The learned linear embedding expressed by the matrix $\boldsymbol{W} \in \mathbb{R}^{d_{model} \times (N \cdot M)}$ combines the output of all heads into a output of size $d_{model} \times d_{model}$. The usage of multiple attention heads is thought to allow the model to focus on multiple elements in the sequence at once, where a single, larger head might only be able to model a single important connection between sequence elements.

Since this attention model features no convolutional structures or recurrence, it is unable to use the order of the sequence. This is resolved by adding *positional encoding* to each entry in the input sequences [49], [50]. The encoding is designed as a deterministic, bounded function of the position in a sequence, allowing a network to incorporate the distance between input values.

While learning the positional encoding is a possibility, there seems to be no performance improvement in doing so over using static functions [49]. The transformer architecture uses two sinusoidal functions of varying frequencies, which are added to the values of each entry in a sequence and are designed to make it easy for the network to attend to relative positions.

4

Image Registration and Recovery

Contents

As previously mentioned, patient motion poses an important problem for MRI as a medical diagnosis modality. At its core, this thesis is concerned with ways to estimate the motion inherent to MRI scans from their partial measurements and with the recovery of sharp images from motion corrupted scans.

In this chapter, the equations governing motion estimation and convex image recovery as they are used in all models of this thesis are defined. Section 4.1 begins by defining the task of static image recovery as an energy minimization problem. Then, Section 4.2 does the same for motion estimation. Section 4.3 presents the motion compensated image recovery process followed in the methods presented in later chapters. In Section 4.4 and Section 4.5, the two motion parametrizations which are used as a framework of estimating motions in the registration approaches are presented.

4.1 Convex Image Recovery

Image recovery tasks, such as the creation of sharp tomographic images from the signals acquired in an MRI scan, can often be cast as minimization problems using the available measurement data. For this, an energy function is defined from the measurement configuration to map each possible solution to an energy value, with unfavourable solutions having an higher associated energy. Recovery then consists of finding the point in solution space which produces minimal energy. In many relevant applications including MRI, the problem is underdetermined or badly conditioned. Due to this, the energy functional commonly consists of a fidelity operator Ψ and an additional regularisation function $\mathcal{R}_{\boldsymbol{I}}$ to restrict the solution space.

The general form of such an optimisation problem then takes the form of

$$\boldsymbol{I}_{opt} = \arg\min_{\boldsymbol{I}} \Psi\left(\boldsymbol{A}\boldsymbol{I},\ \boldsymbol{y}\right) + \lambda_{\boldsymbol{I}}\mathcal{R}_{\boldsymbol{I}}(\boldsymbol{I}) \tag{4.1}$$

where $\boldsymbol{I}_{opt} \in \mathbb{R}^{s_p}$ is the flattened, two-dimensional discrete image of size $s_x \times s_y$ in question, with $s_p = s_x \cdot s_y$ being its number of pixels. $\lambda_{\boldsymbol{I}}$ denotes being the scalar Lagrange parameter of the regulariser. The system matrix $\boldsymbol{A} \in \mathbb{C}^{s_m \times s_p}$ encapsulates the model underlying the measurement, and $\boldsymbol{y} \in \mathbb{C}^{s_m}$ denotes the information available about the image in the measurement domain. In this formulation, s_m is the total number of measured k-space coefficients.

For an MRI measurement consisting of N partial scans, the system matrix and measurements are composed of

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}^{(1)} \\ \boldsymbol{A}^{(2)} \\ \vdots \\ \boldsymbol{A}^{(N)} \end{bmatrix} \quad \text{and} \quad \boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}^{(1)} \\ \boldsymbol{y}^{(2)} \\ \vdots \\ \boldsymbol{y}^{(N)} \end{bmatrix}, \tag{4.2}$$

where each partial system matrix $\boldsymbol{A}^{(n)} \in \mathbb{C}^{s_m \times s_p}$ describes the transformation of the image into measurement space, producing the signal $\boldsymbol{y}^{(n)}$. As such, all known scan parameters from the partial k-space scanning trajectories are used to form the overall system matrix $\boldsymbol{A}$.

Since the measurement configuration is known and can be approximately expressed as a linear embedding of the image into frequency space with the matrix $\boldsymbol{A}$, convex methods for image recovery are commonly employed. In the context of convex optimisation, the data fidelity function $\Psi : \mathbb{C}^{s_m} \times \mathbb{C}^{s_m} \to \mathbb{R}$ is used to quantify the error between a point in solution space and the observation and is chosen to be convex (and therefore continuously differentiable) and real valued. In many cases (and in this work) the square of the Euclidean distance function is used for the data fidelity term, so that $\Psi(\boldsymbol{a}, \boldsymbol{b}) = \|\boldsymbol{a} - \boldsymbol{b}\|_2^2$. This way, the optimisation problem in Equation (4.1) results in the regularised least-squares optimisation problem

$$\boldsymbol{I}_{opt} = \arg\min_{\boldsymbol{I}} \|\boldsymbol{AI} - \boldsymbol{y}\|_2^2 + \lambda_{\boldsymbol{I}} \mathcal{R}_{\boldsymbol{I}}(\boldsymbol{I}). \tag{4.3}$$

The regulariser $\mathcal{R}_{\boldsymbol{I}} : \mathbb{R}^{s_p} \to \mathbb{R}$ on the other hand is constructed to penalise unwanted characteristics of the image and might be not continuously differentiable. In the following chapter, the wavelet regularisation scheme employed in this thesis is motivated and outlined.

4.1.1 Wavelet Sparsity and Image Regularisation

The unregularised optimisation problem $\min_{\boldsymbol{I}} \|\boldsymbol{AI} - \boldsymbol{y}\|_2^2$ is, for most applications, ill-posed. In medical imaging, the number of measured points is normally significantly lower than the number of pixels in the image, so that $s_p \gg s_m$. Such problems are underdetermined and, while solutions exist, they are not unique.

Regularization constricts the solution space by introducing external knowledge about the structure of a desired solution to the optimisation problem. An example of such an implicit prior is the knowledge that the transforms of natural signals in certain domains are *sparse*. This means that the signal in the transformed domain consists mostly of zeroes, with only a few coefficients carrying almost all of its energy. Furthermore, while signals can be sparsely represented, the same is *not* true about the perturbations and measurement noise affecting them.

This observation forms the basis of compressive sensing (CS). In CS, a signal recovery problem is formulated: How many measurements of a signal are required for perfect recovery, given that a sparsifying transform for this signal is available? Research was able to show that under these circumstances, perfect recovery is possible with significantly less measurements than necessary for 'classical' signal recovery using the Shannon-Nyquist bounds.

While CS image recovery is of great interest for MR image recovery researchers, this thesis uses only some fundamental insights from CS theory and research. It

Trans. \ Coeffs.	Natural Images 1%	Natural Images 5%	Natural Images 10%	Natural Images 20%	ACDC 1%	ACDC 5%	ACDC 10%	ACDC 20%
Biorthogonal 4.4	80.41%	96.47%	98.54%	99.59%	88.32%	99.77%	99.92%	99.98%
Haar	78.89%	94.69%	97.61%	99.28%	92.44%	99.51%	99.81%	99.95%
Fejer-Korovkin 8	79.81%	95.69%	98.16%	99.45%	88.88%	99.71%	99.90%	99.98%
Daubechies 8	75.55%	94.76%	97.78%	99.34%	82.15%	99.65%	99.89%	99.98%
2D Fourier	83.18%	93.25%	96.05%	98.01%	97.75%	99.31%	99.64%	99.85%
Untransformed	12.25%	37.81%	56.79%	77.73%	4.89%	20.47%	35.53%	58.73%

Table 4.1: Results from energy compression experiments. Here, the mean energy percentage contained in a number of coefficients that is 1% to 20% of the number of pixels of the original image. The largest coefficients are kept, and the analysis is shown for different representations: 2D discrete Wavelet transforms with three mother wavelets (Biorthogonal 4.4, Haar, and Fejér-Korovkin (FK) 8), Daubechies 8, 2D Fourier transformation, and for comparison a 'regular', non-transformed image. The mean is calculated from each grayscale image contained in a natural image dataset [53] and ACDC [54] datasets. Note that each representation yields a significant compression in respect to a spatial image, with the representations compressing almost all energy into only 5% of the coefficients.

can be easily shown that natural images can be sparsely represented in a wavelet domain [51], [52], with their energy highly concentrated in a few coefficients. To exemplify this, Figure 4.1 contains results of energy concentration experiments on different representations and datasets. Here, the ACDC dataset further discussed in Section 6.4 was chosen to represent medical images. Furthermore, a dataset containing over 13000 natural images used in [53][1] was evaluated for comparison.

The results in the table show that on average, over 95% of the energy of an image is contained in 5% of the largest wavelet coefficients. This is true for both datasets.

The second insight from CS theory used in the optimisation concerns the choice of its norm. While the ℓ_0 pseudo-norm is the intuitive metric for sparseness of a signal, it is unsuitable for optimisation and indeed results in NP-hard problems [55], [56]. Relevant literature shows that the sparseness of the solution of an optimisation problem can instead be promoted by penalising the ℓ_1-norm of the sparse representation, allowing for convex optimisation or basis pursuit algorithms [55]. With these two factors, a suitable regulariser promoting the recovered images to be free of noise and imaging artifacts is

$$\mathcal{R}_{\boldsymbol{I}}(\boldsymbol{I}) = \|W\boldsymbol{I}\|_1, \tag{4.4}$$

[1]Downloaded from `https://www.kaggle.com/datasets/prasunroy/natural-images` on 15.04.2023

where W is a two-dimensional discrete wavelet forward transform. As shown above, the wavelet family used in the transform is of little consequence, as most families offer good energy compression. In further experiments, the Biorthogonal 4.4 (or *bior 4.4*) mother wavelet was chosen together with a symmetric image extension.

A similar regularisation function employed by multiple image processing methods is penalising the total variation of the image. In contrast to the above wavelet formulation, it is not based on a compressibility prior, but the observation that image noise and unwanted signal details cause strong gradients in the image. Therefore, a denoising term for an image $\boldsymbol{I}$ using this principle can be designed as its *total variation*:

$$\mathrm{TV}(\boldsymbol{I}) = \sum_{i,j} \sqrt{\left(I[i+1,j] - I[i,j]\right)^2 + \left(I[i,j+1] - I[i,j]\right)^2}. \tag{4.5}$$

Here, $I[i,j]$ is the image value at pixel position i, j. Penalising the total variation when finding an image through an optimisation procedure suppresses high frequency noise, such as image speckles, while leaving image edges crucial to the image information intact. As the penalisation term Equation (4.5) is not differentiable, primal-dual splitting methods have to be employed for its integration into optimisation tasks.

Preliminary research showed that the image artifacts arising from the MRI measurement are more effectively penalised using the wavelet regulariser formulated in Equation (4.4). The soft edges of the Gibbs ringing resulting from frequency subsampling do not tend to disappear when using total variation denoising, making it a poor candidate for regularisation of such images.

4.2 Motion Estimation: A Non-Convex Problem

In the beginning of this chapter, an optimisation problem was posed which recovered a static image from a number of frequency coefficients measured by MRI. However, up to this point, the formulation of this optimisation ignored the motion of the image. When recovering images from motion corrupted data, the minimization problem posed in Equation (4.1) is therefore unlikely to yield images of satisfactory quality. A necessary step towards sharp images is the approximation of a set of motion parameters, which can then be used to compensate for the motion in the measurements. Since the focus in this thesis is on blind motion compensation, no information on the motion is available initially.

Let's consider a more realistic setting for the scan setting, in which each of the N partial measurements would 'see' one of N images $\boldsymbol{I}^{(n)}$, $0 < n \leq N$.

As mentioned in earlier chapters, instantaneous shot acquisition allows each partial scan at time step n to be expressed as $\boldsymbol{y}^{(n)} = \boldsymbol{A}^{(n)}\boldsymbol{I}^{(n)}$, with a system matrix $\boldsymbol{A}^{(n)} \in \mathbb{C}^{s_m \times s_p}$ and the flattened image $\boldsymbol{I}^{(n)} \in \mathbb{R}^{s_p}$ with $s_p \gg s_m$.

The goal is then to approximate a set of motion parameters describing the state of the measured image at each time step in relation to a reconstruction target. Conventionally, the first image in the series ($\boldsymbol{I}^{(0)}$) is used as the target for this registration and is thus assumed to be motion free.

Let $\mathcal{T}_{\boldsymbol{\phi}^{(n)}}$ be an interpolation matrix parametrised by a number of motion coefficients in $\boldsymbol{\phi}^{(n)}$. In this formulation, the optimal parameter set would yield interpolation matrices so that

$$\mathcal{T}_{\boldsymbol{\phi}^{(n)}}\boldsymbol{I}^{(1)} = \boldsymbol{I}^{(n)} \;\; \forall\, 0 < n \leq N. \tag{4.6}$$

Optimally, an inverse $\bar{\boldsymbol{\phi}}^{(n)}$ to any parametrisation also exists, so that $\mathcal{T}_{\bar{\boldsymbol{\phi}}^{(n)}}\boldsymbol{I}^{(n)} = \boldsymbol{I}^{(1)}$. This is not a prerequisite for motion estimation, but required for its use in image recovery.

Assuming for a moment that the sharp image $\boldsymbol{I}$ is already known, the optimisation problem for the sequence of motion parameters $\boldsymbol{\phi}_{opt} = [\boldsymbol{\phi}_{opt}^{(1)}, \boldsymbol{\phi}_{opt}^{(2)}, \dots \boldsymbol{\phi}_{opt}^{(n)}]$ can be formulated as

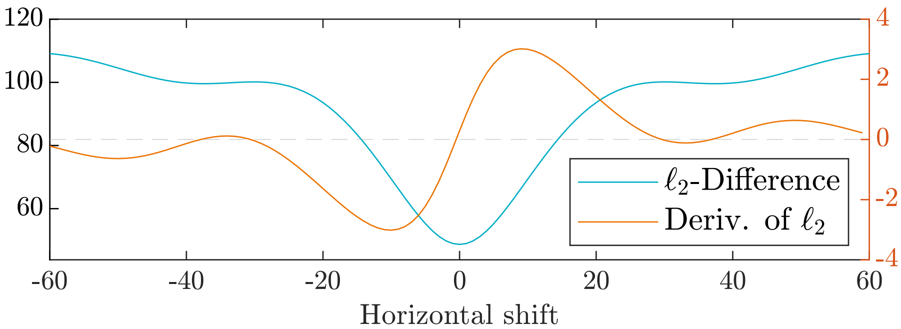
$$\boldsymbol{\phi}_{opt} = \arg\min_{\boldsymbol{\phi}} \sum_{n=1}^{N} \|\boldsymbol{A}^{(n)}\mathcal{T}_{\boldsymbol{\phi}^{(n)}}\boldsymbol{I} - \boldsymbol{y}^{(n)}\|_2^2 + \lambda_{\boldsymbol{\phi}}\mathcal{R}_{\boldsymbol{\phi}}(\boldsymbol{\phi}). \tag{4.7}$$

Here, $\mathcal{R}_{\boldsymbol{\phi}}$ is a regularisation function on the estimated motion parameters scaled with a Lagrange multiplier $\lambda_{\boldsymbol{\phi}}$. Intuitively, Equation (4.7) seeks the motion which explains the measurements, given a sharp image $\boldsymbol{I}$.

Since the time step between measurements is generally not large, the approximated motions are expected to be 'smooth', changing gradually without large jumps in magnitude. While not a strict necessity, most research approaches incorporate a motion regularisation term which promotes smoothness by, for example, simply

(a)

(b)



(c) ℓ_2-error and forward derivative

Figure 4.2: Example of non-convex behaviour in motion estimation. Two images ((a) and (b)), with (b) affected by Gibbs ringing, are registered by shifting (b) horizontally. (c) shows the squared difference between (a) and the shifted (b) as well as the derivative of the error (in blue). Note that the error function exhibits local minima, signified by a change in the sign of its derivative (shown in red).

penalising the difference in parametrisation between discrete time steps. A common formulation used in this work and others (for example in [57]) is a term of the form

$$\lambda_\phi \mathcal{R}_\phi(\boldsymbol{\phi}) = \lambda_\phi \| \sum_{n=2}^{N} \boldsymbol{\phi}^{(n-1)} - \boldsymbol{\phi}^{(n)} \|_1, \tag{4.8}$$

where the Lagrange parameter λ_ϕ weights the 'smoothness' of the approximated motion and the image reconstruction terms in Equation (4.7). An alternative approach used in Section 5.4 directly restricts the magnitude of difference between consecutive motion states.

Optimizing for a set of motion parameters bears a couple of problems. The choice of the underlying motion parametrisation fundamentally influences the optimisation procedure, as the model might not be differentiable in respect to the quality of the recovered image. Furthermore, there is little guarantee that the optimisation problem is convex, especially for larger deformations.

Figure 4.2 shows a simple example based on the Shepp-Logan phantom [16]. An image exhibiting horizontal Fourier ringing artifacts (as might arise from PROPELLER sampling in MRI) is matched with an artifact-free image using a motion model only permitting horizontal movement. Its motion parameter is sampled for a large range of motion and the ℓ_2 loss between the two images evaluated for each sample point. While the ℓ_2-term exhibits a clear global minimum[2], there are noticeable fluctuations outside a 'region of influence' of this minimum, with additional local minima at a shift of about ± 35 pixels.

As the solution to the problem in Equation (4.7) reached by gradient descent algorithms therefore depends on the choice of initial values of $\boldsymbol{\phi}$, approaches using convex optimisation for registration have to assume the underlying motion to be small.

4.3 Motion Compensated Image Reconstruction

The image reconstruction sketched in the beginning of this chapter does not yet incorporate the motion estimate found by Equation (4.7). Considering the motion as known, a motion operator can be included into the optimisation problem in Equation (4.1). The motion compensated quadratic image recovery problem (incorporating the wavelet regulariser from Section 4.1.1) then reads as

$$\boldsymbol{I}_{opt} = \arg\min_{\boldsymbol{I}} \frac{1}{2} \| \sum_{n=1}^{N} \boldsymbol{A}^{(n)} \mathcal{T}_{\boldsymbol{\phi}^{(n)}} \boldsymbol{I} - \boldsymbol{y}^{(n)} \|_2^2 + \lambda_{\boldsymbol{I}} \| W \boldsymbol{I} \|_1. \tag{4.9}$$

The intuition behind Equation (4.9) is similar to the one given for Equation (4.7). However, now the image $\boldsymbol{I}$ which explains the measurement given a motion parametrisation is sought.

Grouping the system and interpolation matrices as

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}^{(1)} & & & \\ & \boldsymbol{A}^{(2)} & & 0 \\ & & \ddots & \\ & 0 & & \boldsymbol{A}^{(N)} \end{bmatrix}, \quad \boldsymbol{T} = \begin{bmatrix} \mathcal{T}_{\boldsymbol{\phi}^{(1)}} \\ \mathcal{T}_{\boldsymbol{\phi}^{(2)}} \\ \vdots \\ \mathcal{T}_{\boldsymbol{\phi}^{(N)}} \end{bmatrix},$$

[2] Which, due to the influence of ringing, is not at or near 0.

the problem can be stated as

$$\boldsymbol{I}_{opt} = \arg\min_{\boldsymbol{I}} \frac{1}{2}\|\boldsymbol{ATI} - \boldsymbol{y}\|_2^2 + \lambda_{\boldsymbol{I}}\|W\boldsymbol{I}\|_1. \tag{4.10}$$

The optimisation problem in Equation (4.10) is in the form of an *Least Absolute Shrinkage and Selection Operator*(LASSO) [58] problem. While there are no known closed solutions to LASSO problems, iterative solutions do exist. The most common ones build on the method published by Tibshirani [58], with two examples being Gauss-Seidel iterative thresholding [59] or least angle regression [60]. In this work, a memory friendly iterative thresholding approach based on primal-dual splitting methods is used. The regularised optimisation problem is partitioned into two distinct projections.

Splitting objective function and regulariser yields the constrained optimisation problem

$$\begin{aligned} \boldsymbol{I}_{opt} &= \arg\min_{\boldsymbol{I}} f(\boldsymbol{I}) + \lambda_{\boldsymbol{I}}\mathcal{R}_I(\boldsymbol{v}) \quad \text{s.t. } \boldsymbol{I} - \boldsymbol{v} = 0 \\ f(\boldsymbol{I}) &= \|\boldsymbol{ATI} - \boldsymbol{y}\|_2^2 \end{aligned} \tag{4.11}$$

which introduces an auxillary variable $\boldsymbol{v}$ connecting the two problems.

Using the *Alternating Direction Method of Multipliers* (ADMM)[3] [61] framework and the proximal operator

$$\text{prox}_{f,\rho}(\boldsymbol{q}) = \arg\min_{\boldsymbol{I}} \left(f(\boldsymbol{I}) + \frac{1}{2\rho}\|\boldsymbol{I} - \boldsymbol{q}\|_2^2 \right), \tag{4.12}$$

the problem can be decomposed into an iterative algorithm consisting of the three steps

$$\begin{aligned} \boldsymbol{I}^{j+1} &= \text{prox}_{f,\rho}(\boldsymbol{v}^j + \boldsymbol{w}^j) \\ &= \arg\min_{\boldsymbol{I}} \frac{1}{2}||\boldsymbol{ATI} - \boldsymbol{y}||_2^2 + \frac{1}{2\rho}\|\boldsymbol{I} - \left(\boldsymbol{v}^j + \boldsymbol{w}^j\right)\|_2^2 && (4.13) \\ \boldsymbol{v}^{j+1} &= \text{prox}_{\mathcal{R}_I,\rho}(\boldsymbol{I}^{j+1} + \boldsymbol{w}^j) \\ &= W^{-1}\mathcal{S}_{\lambda_{\boldsymbol{I}}/\rho}\left(W\left(\boldsymbol{I}^{j+1} + \boldsymbol{w}^j\right)\right) && (4.14) \\ \boldsymbol{w}^{j+1} &= \boldsymbol{w}^j + \boldsymbol{I}^{j+1} - \boldsymbol{v}^{j+1}. && (4.15) \end{aligned}$$

[3]Also known as Douglas-Rachford-Splitting

In these equations, ρ is a relaxation parameter weighting the partial solutions and $\mathcal{S}$ is the proximal operator of the ℓ_1 norm. In most cases, the split variable $\boldsymbol{v}^0$ and the residual $\boldsymbol{w}^0$ can be initialized with zeroes.

For the ℓ_1-norm, the proximal operator has a closed solution with the soft shrinkage operator defined as

$$\mathcal{S}_b(\boldsymbol{a}) = [s_b(a_1), s_b(a_2), ...s_b(a_{N_a}))]^\intercal, \quad \boldsymbol{a} \in \mathbb{R}^{N_a} \tag{4.16}$$

$$s_b(a) = \operatorname{sign}(a)\max(|a| - b,\ 0). \tag{4.17}$$

Most of the complexity of the ADMM then comes from its image update step in Equation (4.13), as the solution to the proximal operator involves solving an unconstrained quadratic optimisation problem. For this, a variant of the conjugated gradient (CG) optimiser [62] was implemented in this work.

CG finds approximate solutions of $\boldsymbol{x}$ in linear problems of the form

$$\boldsymbol{b} = \boldsymbol{H}\boldsymbol{x}, \tag{4.18}$$

where the matrix $\boldsymbol{H}$ is required to be positive definite and symmetric. In case the matrix $\boldsymbol{H}$ does not fulfil these requirements, the CG can be applied to its normal equations by left-multiplying $\boldsymbol{H}^*$ with * denoting the Hermitian transpose. Then, the overall problem reads as

$$\boldsymbol{H}^*\boldsymbol{b} = \boldsymbol{H}^*\boldsymbol{H}\boldsymbol{x}. \tag{4.19}$$

The resulting method is known as the Conjugate Gradient method on the Normal Residual (CGNR).

To bring Equation (4.13) into a form solvable by CG or CGNR, the two quadratic norm terms are combined into

$$\boldsymbol{I}^{j+1} = \arg\min_{\boldsymbol{I}} \left\| \begin{bmatrix} \boldsymbol{AT} \\ 1/\sqrt{\rho}\, \mathbb{1}_{s_p} \end{bmatrix} \boldsymbol{I} - \begin{bmatrix} \boldsymbol{y} \\ 1/\sqrt{\rho}\, (\boldsymbol{v}^j + \boldsymbol{w}^j) \end{bmatrix} \right\|_2^2, \tag{4.20}$$

where $\mathbb{1}_{s_p}$ is an identity matrix $\in \mathbb{R}^{s_p \times s_p}$. Formulated this way, the exact solution would yield

$$\begin{bmatrix} \boldsymbol{AT} \\ 1/\sqrt{\rho}\, \mathbb{1}_{s_p} \end{bmatrix} \boldsymbol{I} = \begin{bmatrix} \boldsymbol{y} \\ 1/\sqrt{\rho}\, (\boldsymbol{v}^j + \boldsymbol{w}^j) \end{bmatrix}. \tag{4.21}$$

Since the matrix $\begin{bmatrix} \boldsymbol{AT} \\ 1/\sqrt{\rho}\,\mathbb{1}_{s_p} \end{bmatrix}$ is neither positive definite nor symmetric, multiplying its Hermitian transpose to both sides of Equation (4.21) yields an appropriate form which fulfils the criteria the CG requires. The linear equation approximately solved by CGNR then becomes

$$\left(\boldsymbol{T}^{\intercal}\boldsymbol{A}^{*}\boldsymbol{AT} + 1/\rho\,\mathbb{1}_{s_p}\right)\boldsymbol{I} = \boldsymbol{T}^{\intercal}\boldsymbol{A}^{*}\boldsymbol{y} + 1/\rho\,(\boldsymbol{v}^{j} + \boldsymbol{w}^{j}). \tag{4.22}$$

Since the large matrices involved are block diagonal and composed of operations for which fast solvers can be employed, the CGNR implementations used in this work do not explicitly form either matrix $\boldsymbol{A}$ or $\boldsymbol{T}$ or their transposeds.

4.4 Rigid Motion Models

An important motion modality in MRI is so-called *bulk* or *rigid motion*, in which the tomographed tissue moves as a single object. Among other regions of interest, this accurately describes for example the motion expected during brain scans, as the brain and its surrounding skull are for the purposes of tomography imaging (and disregarding minute pulsatile movements of e.g. blood vessels) *rigid*.

Hence, a model only describing translation and rotation of the image is sufficient to capture the motion of such structures. Since all images in this work are two-dimensional, the description of these motions is restricted to two dimensions as well, albeit the extension to 3D is straightforward and known from relevant literature.

A continuos image $\tilde{\boldsymbol{I}}$ indexed by $\boldsymbol{x}$ is translated by a vector of shift parameters $\boldsymbol{\phi}_{shift} \in \mathbb{R}^2$ with

$$\tilde{\boldsymbol{I}}_{shift}(\boldsymbol{x}) = \tilde{\boldsymbol{I}}(\boldsymbol{x} + \boldsymbol{\phi}_{shift}). \tag{4.23}$$

A rotation of the image of ϕ_{rot} radians around its centre can be expressed as a multiplication of the coordinate vector with a rotation matrix $\boldsymbol{R}_{\phi_{rot}}$ as

$$\tilde{\boldsymbol{I}}_{rot}(\boldsymbol{x}) = \tilde{\boldsymbol{I}}(\boldsymbol{R}_{\phi_{rot}}\boldsymbol{x}), \quad \boldsymbol{R}_{\phi_{rot}} = \begin{bmatrix} \cos(\phi_{rot}) & \sin(\phi_{rot}) \\ -\sin(\phi_{rot}) & \cos(\phi_{rot}) \end{bmatrix}. \tag{4.24}$$

Using the defintions of translation and rotation, the combined parametrisation for a rigidly moved image can be expressed as the vector

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi}_{shift} \\ \phi_{rot} \end{bmatrix}. \tag{4.25}$$

Both rotation and translation are affine and preserve all point-wise differences in the images. Another useful property for image registration is their straightforward invertibility and composition, as

$$\tilde{\boldsymbol{I}}_{shift}(\boldsymbol{x} - \boldsymbol{\phi}_{shift}) = \boldsymbol{I}(\boldsymbol{x}), \quad \tilde{\boldsymbol{I}}_{rot}(\boldsymbol{R}_{-\phi_{rot}}\boldsymbol{x}) = \tilde{\boldsymbol{I}}(\boldsymbol{x}) \tag{4.26}$$

holds. While this might seem obvious, it is not true for the motion description introduced in the coming section.

Another advantageous property of these transformations is that equivalent transforms can be applied in frequency space. Translation corresponds to a multiplication of the spectrum with a phase ramp, so that

$$\tilde{\boldsymbol{I}}(\boldsymbol{x} + \boldsymbol{\phi}_{shift}) \longleftrightarrow \hat{\boldsymbol{I}}(\boldsymbol{\omega})\exp(-i2\pi\langle\boldsymbol{\omega}, \boldsymbol{\phi}_{shift}\rangle), \tag{4.27}$$

where $\hat{\boldsymbol{I}}(\boldsymbol{\omega})$ is the two-dimensional Fourier transform of $\tilde{\boldsymbol{I}}(\boldsymbol{x})$.

A rotation in image space corresponds to a rotation in frequency space, yielding

$$\tilde{\boldsymbol{I}}(\boldsymbol{R}_{\phi_{rot}}\boldsymbol{x}) \longleftrightarrow \hat{\boldsymbol{I}}(\boldsymbol{R}_{\phi_{rot}}\boldsymbol{\omega}). \tag{4.28}$$

While the above is sufficient for the description of rigid motion in continuos images, rotations and non-integer translations require an additional spatial interpolation function for discrete images. This work generally chooses linear interpolation when transforming images in the spatial domain.

4.5 Vector Fields

In many cases, the rigid motion described by affine models does not suffice for correctly describing biological motion. Especially for pulsatile motion found in the heart and peristaltic colon motions, strictly preserving pointwise distances is too restrictive. If there is no known motion model capable of describing the motion in question, parameter-free motion descriptions have to be used. In this case, instead of a handful of descriptive parameters, a dense vector field is approximated.

Let $\boldsymbol{y}(\boldsymbol{x})$ be a vector field describing the motion of any point in position $\boldsymbol{x}$ between two images $\boldsymbol{I}_0$ and $\boldsymbol{I}_1$, so that $\boldsymbol{I}_1(\boldsymbol{y}(\boldsymbol{x})) \approx \boldsymbol{I}_0(\boldsymbol{x})$. Thus, the vector field $\boldsymbol{y}$ *registers* $\boldsymbol{I}_1$ to $\boldsymbol{I}_0$.

The motion vector field is further separated into two components: the static identity field $\boldsymbol{x}$ and the *deformation field* $\hat{\boldsymbol{y}}(\boldsymbol{x})$, so that

$$\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{x} + \hat{\boldsymbol{y}}(\boldsymbol{x}). \tag{4.29}$$

Thus, to connect the vector field formulation to registration to the equations in the previous sections, the parametrisation of motion in discrete two-dimensional images with s_p pixels using dense vector fields would yield the motion parameters

$$\boldsymbol{\phi} = \begin{bmatrix} \hat{\boldsymbol{y}}(1) \\ \hat{\boldsymbol{y}}(2) \\ \vdots \\ \hat{\boldsymbol{y}}(s_p) \end{bmatrix} \in \mathbb{R}^{s_p \times 2}. \tag{4.30}$$

While the rigid motion models from the previous section can be inverted by a change in their sign, the same is not true for arbitrary vector fields. Here, the inverse might be very different from the simple negation of the field, or may not exist at all.

In [63], a fixed-point scheme is proposed to handle this specific problem. Let $\boldsymbol{a}(\boldsymbol{x})$ be an invertible vector field, and $\boldsymbol{b}(\boldsymbol{x})$ denote its inverse, so that $\boldsymbol{a}(\boldsymbol{b}(\boldsymbol{x})) = \boldsymbol{x}$. Using the expansion of vector fields into deformation fields and identity, this becomes

$$\begin{aligned} \boldsymbol{a}(\boldsymbol{b}(\boldsymbol{x})) &= \hat{\boldsymbol{a}}(\boldsymbol{b}(\boldsymbol{x})) + \boldsymbol{b}(\boldsymbol{x}) && (4.31)\\ &= \boldsymbol{x} + \hat{\boldsymbol{b}}(\boldsymbol{x}) + \hat{\boldsymbol{a}}(\boldsymbol{x} + \hat{\boldsymbol{b}}(\boldsymbol{x})) && (4.32)\\ &= \boldsymbol{x} && (4.33)\\ \hat{\boldsymbol{b}}(\boldsymbol{x}) &= -\hat{\boldsymbol{a}}(\boldsymbol{x} + \hat{\boldsymbol{b}}(\boldsymbol{x})). && (4.34) \end{aligned}$$

Here, $\hat{\boldsymbol{a}}, \hat{\boldsymbol{b}}$ are again the deformation fields of the vector fields $\boldsymbol{a}, \boldsymbol{b}$.

Equation (4.34) shows the problem in vector field inversion: negation of $\hat{\boldsymbol{a}}$ yields the correct inversion field $\hat{\boldsymbol{b}}$ only in case the fields are very uniform - for example, for vector fields of rigid motion. In the general case, the influence of the (unknown) $\hat{\boldsymbol{b}}$ on the equation necessitates a different approach.

Interestingly, Equation (4.34) already defines a basic fixed point algorithm for the inversion. The deformation field of the inverse can be found iteratively through the algorithm shown in pseudo-code in Algorithm 1.

Algorithm 1 Fixed-point inversion of vector fields

procedure INVERSION($\hat{\boldsymbol{a}}(\boldsymbol{x}), \tau$)
  $\hat{\boldsymbol{b}}_0(\boldsymbol{x}) \leftarrow -\hat{\boldsymbol{a}}(\boldsymbol{x})$
  $n \leftarrow 0$
  while $\|\hat{\boldsymbol{b}}_n(\boldsymbol{x}) + \hat{\boldsymbol{a}}(\boldsymbol{x} + \hat{\boldsymbol{b}}(\boldsymbol{x}))\| > \tau$ **do**
    $\hat{\boldsymbol{b}}_{n+1}(\boldsymbol{x}) \leftarrow -\hat{\boldsymbol{a}}(\boldsymbol{x} + \hat{\boldsymbol{b}}^n(\boldsymbol{x}))$
    $n \leftarrow n + 1$
  return $\hat{\boldsymbol{b}}_n(\boldsymbol{x})$

In Algorithm 1, $\hat{\boldsymbol{a}}(\boldsymbol{x})$ is iteratively deformed by the current estimation of the inverse deformation. The iteration is stopped if the norm of the sum of deformation fields $\|\hat{\boldsymbol{b}}_n(\boldsymbol{x}) + \hat{\boldsymbol{a}}(\boldsymbol{x} + \hat{\boldsymbol{b}}(\boldsymbol{x}))\|$ is sufficiently close to 0, with a small threshold $\tau > 0$. A suitable starting value is $\hat{\boldsymbol{b}}^0(\boldsymbol{x}) = -\hat{\boldsymbol{a}}(\boldsymbol{x})$, which would be reached after one iteration if starting from $\hat{\boldsymbol{b}}^0(\boldsymbol{x}) = \boldsymbol{0}$.

A fixed-point algorithm of this form converges if the iterated field is bounded and a *contraction*, so that the inequalities

$$\|\hat{\boldsymbol{a}}(\boldsymbol{x})\| < \infty \tag{4.35}$$

$$\|\hat{\boldsymbol{a}}(\boldsymbol{q}) - \hat{\boldsymbol{a}}(\boldsymbol{p})\| \leq K\|\boldsymbol{q} - \boldsymbol{p}\| \quad \forall q, p \quad \text{for some } K < 1 \tag{4.36}$$

both hold.

Condition 4.35 is natively obtained for natural, bounded images. Ensuring that $\hat{\boldsymbol{a}}(\boldsymbol{x})$ is sub-1 Lipschitz and therefore a contraction as required by condition 4.36 is a bit more involved.

To fulfil the condition posited in Equation (4.36), the difference of the displacement of two points in the field has to be smaller than the distance of the points. In a practical sense, if the Lipschitz condition is broken for a point pair, there will be elements in the image whose orientation flips as an effect of the mapping, causing their area to become negative. While this is not impossible in a biological setting (an occlusion in the field of view might, for example, cause an orientation flip in the imaged area), its effect on the mapping make registering orientation flips undesirable. As occlusion can not be reversed by the application of deformation, such folding effects are irreversible in the context of image registration. Furthermore, the approach outlined in Section 6.7 requires the deformation field to be invertible.

A practical way to measure 'breaks' in the Lipschitz condition (and therefore orientation flips) in an estimated field utilizes the Jacobian of the deformation field

$J(\hat{\boldsymbol{a}})$ [64]. The determinant of the Jacobian of a two-dimensional deformation field $\hat{\boldsymbol{a}}(\boldsymbol{x})$ is defined as

$$\det(J(\hat{\boldsymbol{a}})) = \det \begin{vmatrix} \frac{d\hat{\boldsymbol{a}}_1(\boldsymbol{x})}{d\boldsymbol{x}_1} & \frac{d\hat{\boldsymbol{a}}_1(\boldsymbol{x})}{d\boldsymbol{x}_2} \\ \frac{d\hat{\boldsymbol{a}}_2(\boldsymbol{x})}{d\boldsymbol{x}_1} & \frac{d\hat{\boldsymbol{a}}_2(\boldsymbol{x})}{d\boldsymbol{x}_2} \end{vmatrix}, \tag{4.37}$$

where the dimension of field and deformation is indicated in the subscript. This determinant gives insight into the change in area an element in the image experiences during deformation. Specifically, the value of the Jacobian determinant of a vector field at a point p is non-zero if the mapping preserves orientation in a neighbourhood of p, turns negative on an orientation flip, and equals zero for element p having zero area after the deformation is applied.

Since the vector fields of interest in this thesis are the results of optimisation problems, the properties formulated above can often be guaranteed by appropriately restricting the solution space of the optimisation. In the context of this work, the required properties regarding invertibility can be established by ensuring that the found mapping is a *diffeomorphism*, making it a member of a Lie Group.

Lie groups, in a general sense, are smooth manifolds exhibiting a group structure which is also a smooth manifold. Thus, its members are diffeomorphisms, ensuring that for each member of the group, a (differentiable) inverse operation exists. Lie groups are defined by their *algebra* - their tangent space at the identity of the group. Each point on the group manifold can be uniquely generated by integrating a point in the tangent space. Using this method, an invertible vector field can be generated as a member of a Lie group by picking a point in its tangent space. Since the transformation from algebra to group is differentiable, many image registration routines relying on optimisation can be adapted to output invertible vector fields by optimizing over a Lie algebra space.

From a physical perspective, this reformulation of considering a vector field as a point on a Lie algebra can be interpreted as not looking for fields of particle flow, but for their instantaneous velocities at each point. Integrating the field of particle velocities will in turn yield a deformation field.

Definition 1. *Let $\mathfrak{G}$ be a Lie group and let $\boldsymbol{p}$ be a point in its Lie algebra $\mathfrak{g}$. Then $g(t)$ is the subgroup of $\mathfrak{G}$ defined by $\boldsymbol{p}$ and is given by*

$$\begin{aligned} \frac{\partial g(t)}{\partial t} &= g(t) \circ \boldsymbol{p} \\ g(0) &= e. \end{aligned}$$

Here, e is the neutral element of $\mathfrak{G}$ and $\circ$ denotes left multiplication of $g(t)$ to $\boldsymbol{p}$. The function $g(t)$ is the one parameter subgroup of $\mathfrak{G}$ with $\boldsymbol{p}$ being its infinitesimal generator. The Lie-group exponential $\exp(\boldsymbol{p})$ *is defined by the function $g(t)$ at $t = 1$.*

Considering Definition 1, the exponentiation denotes a general exponential map from the Lie algebra to a member of its group. In this thesis, both the members of the Lie algebra as well as the group members are vector fields, for which a compact definition of the exponential map is derived from an approach to the exponentiation of square matrices.

Considering for a moment that both the Lie group element and its generating element from the associated algebra are real square matrices, the exponential map connecting the two becomes the *matrix exponential*: for a matrix $\boldsymbol{B}$ from the Lie algebra, the Lie group member $\exp(\boldsymbol{B})$ can be calculated by the series

$$\exp(\boldsymbol{B}) = \sum_{k=0}^{\infty} \frac{1}{k!} \boldsymbol{B}^k, \tag{4.38}$$

in which $\boldsymbol{B}^0$ is defined as the identity matrix.

There are many ways to compute the matrix exponential (many of which are *dubious*, according to [65]), but the most common one is *scaling and squaring* [65]. This algorithm turns a "bad" approximation of the matrix exponential into a "good" one by using fundamental properties of the exponential.

As the name implies, scaling and squaring involves *scaling* a matrix $\boldsymbol{B}$ prior to it being exponentiated, and *squaring* the result to undo the scaling. This way, the matrix exponential becomes

$$\mathrm{e}^{\boldsymbol{B}} = \left(\mathrm{e}^{\boldsymbol{B}/2^j}\right)^{2^j}. \tag{4.39}$$

This formulation exploits the two findings: First, the truncation errors as well as the computational complexity of common series approximations of the matrix exponential scale with the spectral norm of the matrix in question as well as the spread of its Eigenvalues. Second, squaring a matrix is comparatively cheap, from a computational cost standpoint. These two properties are combined into the above scaling and squaring method, where a scaling integer k is chosen so that the approximation error and algorithmic complexity are sufficiently small and the result is calculated by squaring j times.

Equation (4.39) can be extended to velocity vector fields by extending the squaring operation with a composition operator $\circ$. For a given velocity vector field $\hat{\boldsymbol{y}}$, the squaring is defined as

$$\hat{\boldsymbol{y}}^2 = \hat{\boldsymbol{y}}(\boldsymbol{x}) \circ \hat{\boldsymbol{y}}(\boldsymbol{x}) = \hat{\boldsymbol{y}}(\boldsymbol{x} + \hat{\boldsymbol{y}}(\boldsymbol{x})). \tag{4.40}$$

An approximation for the exponential map can be created using the series in Equation (4.38) as a first-order Taylor series:

$$\exp(\hat{\boldsymbol{y}}(\boldsymbol{x}))^{1/2^j} \approx \boldsymbol{x} + \frac{\hat{\boldsymbol{y}}(\boldsymbol{x})}{2^j}, \tag{4.41}$$

for which the truncation error can again be controlled by the exponential scaling factor j. Then, the exponentiated vector field is calculated by the recursion

$$\exp(\hat{\boldsymbol{y}}(\boldsymbol{x}))^{1/2^{(k-1)}} = \exp(\hat{\boldsymbol{y}}(\boldsymbol{x}))^{1/2^k} \circ \exp(\hat{\boldsymbol{y}}(\boldsymbol{x}))^{1/2^k}. \tag{4.42}$$

Starting with $k = j$, Equation (4.42) yields the exponential map of $\boldsymbol{y}(\boldsymbol{x})$ as

$$\tilde{\hat{\boldsymbol{y}}}(\boldsymbol{x}) = \exp(\hat{\boldsymbol{y}}(\boldsymbol{x}))^{1/2^{(0)}} \tag{4.43}$$

after j recursive steps. The resulting field $\tilde{\hat{\boldsymbol{y}}}$ is invertible and has a positive Jacobian determinant at every position.

5

Discrete Model Based Motion Estimation

Contents

5.1 Problem Statement

For many parts of the human body, the expected motion can often be described very accurately using a parametrised description. The most commonly seen example is the description of bulk motion as encountered in scans of the human head. Here, a rigid motion model is sufficient to describe most of the observed movement. As the rigid motion model is differentiable, a gradient descent motion estimation algorithm can be formulated. This is a popular approach in relevant research, as laid out in Section 5.2.

Gradient descent based approaches exhibit two problems: They commonly struggle with larger movement, for which the motion estimation problem often becomes non-convex, and they rely on the differentiability of the motion model, preventing the use of more complex motion descriptions.

In this chapter, an autofocusing approach based on an efficient parameter search over a space of motion model parameters is presented as a method of autofocusing. As such a formulation is not dependent on the gradient of the motion model, it serves as a potential solution to both of the mentioned problems.

The approach presented in this chapter was initially published in the research paper titled ”Retrospective Blind MR Image Recovery with Parametrized Motion Models” [1] at the BVM 2019 and was developed further for this thesis. The method presented in this section represents one of the major contributions of this work.

5.2 State of the Art

As outlined in the previous chapters, the specific form of image registration employed in MRI faces particular difficulties, as its images are acquired in frequency space. Then, the motion between two images underlying two partial scans has to be estimated from their subspaces in k-space, which potentially do not overlap.

Many approaches exist to incorporate external signals into the measurement sequence or the estimation of parametrised deformations. The best known example might be the use of so-called navigator shots [66]–[68], in which a specific line or region in k-space is sampled additionally in each acquisition step. Robust motion correction is then calculated from the navigator measurements.

Another approach is the use of *gating* to help reduce motion artifacts in repeating movements. An external signal, for example acquired from a respiratory belt or an electrocardiogram is used to trigger each acquisition step in the same phase of the motion in question. Cardiac measurements are often gated to capture the end systole or end diastole of the heartbeat cycle, while the state of complete exhalation is used to gate lung tomographies.

In this work however, the focus is on *autofocusing* methods in which no external data is acquired next to the k-space samplings. In autofocusing methods, the motion is typically estimated by minimising an artifact metric or maximising an image quality measure. Autofocusing algorithms tend to be iterative solvers, and produce their solution by alternating estimation of an image and motion parameters from measured k-space data.

A popular example of such an iterative scheme applied to MRI reconstruction problems is GradMC. Starting out using a restriction to affine motion parametrisation [57], the algorithm was further enhanced to handle non-rigid motion by assuming a motion locality [69]. In the latter, non-rigid motion field is approximated through the use of locally rigidly moving image patches.

Taymourtash et al. [70] researched affine motion registration for fetal brain functional MR imaging for 4D volume reconstruction. In their approach, motion correction is again accomplished through convex optimisation using a total variation prior. Furthermore, they penalise the rank of the 4D volume, enforcing self-similarity between volume slices.

Recently, deep learning supported approaches were developed for autofocusing tasks, even though their use in MRI remains sparse.

Examples of schemes incorporating neural networks include the work by Kuzima et al. [71], who use a U-Net as a prior in a gradient descent algorithm, which optimises translation and rotation parameters to minimise a ℓ_1-norm based image similarity measure. The U-Net primarily acts as a denoiser and gradually removes artifacts from a noisy MR image.

Similarly, Wang et al. [72] propose an architecture based on a generative adversarial network which serves as a prior for a joint translation correction and super resolution task, specifically for super resolution cardiac segmentation.

Pan et al. [73] use an iterative reconstruction scheme without motion compensation in tandem with group-wise motion estimation and a warping network. Iterating image reconstruction and motion estimation again yields favourable results. A similar approach is used by Xu et al. [74], who use two transformer-inspired networks to estimate rigid motion and image features iteratively.

5.3 Alternating optimisation

As explained in the previous chapters, tools from convex optimisation are suitable to extract an image from known measurements and system matrices and a (correctly estimated) motion parameter approximation. However, the estimation of motion parameters using convex methods is limited by the non-convexity of the associated problem. Formally, Equation (4.1) and Equation (4.7) can be combined into a single optimisation problem with

$$\boldsymbol{I}_{opt}, \boldsymbol{\phi}_{opt} = \arg\min_{\boldsymbol{I},\boldsymbol{\phi}} \sum_{n=1}^{N} \|\boldsymbol{A}^{(n)} \mathcal{T}_{\boldsymbol{\phi}^{(n)}} \boldsymbol{I} - \boldsymbol{y}^{(n)}\|_2^2 + \lambda_I \mathcal{R}_I(\boldsymbol{I}) + \lambda_\phi \mathcal{R}_\phi(\boldsymbol{\phi}) \tag{5.1}$$

This combined expression inherits the combination of problems outlined in the previous chapters, as the problem is highly underdetermined and possibly non-convex in respect to the motion parameters. Thus, to the best of the authors knowledge, no approaches directly solving the problem posed in Equation (5.1) exist.

The strategy followed in most approaches involves alternating between image and motion approximation. The basic idea is that the motion artifacts arising in MRI inhibit the sparseness of the representation of an image in a suitable transform domain. Therefore, one can weaken the influence of the motion artifacts by regularising for a known image prior, as discussed in Section 4.1.1. This procedure will not eliminate motion artifacts, as a regularisation strong enough to do so would strongly damage image quality. However, adequate levels of regularisation allow to recover an image estimate which can be used for motion approximation. Incorporating the motion resulting from that optimisation in the image reconstruction then ideally yields an image of higher quality and with reduced motion artifacts. Iterating the two optimisations then converges to a sharp image and an accurate motion approximate.

In alternating approaches, the above optimisation problem is decomposed into the iterated set of sub-problems

$$\begin{aligned}
\boldsymbol{I}_{k+1} = f_{\mathrm{img}}(\boldsymbol{\phi}_k) \quad &= \arg\min_{\boldsymbol{I}} \sum_{n=1}^{N} \left\| \boldsymbol{A}^{(n)} \mathcal{T}_{\boldsymbol{\phi}_k^{(n)}} \boldsymbol{I} - \boldsymbol{y}^{(n)} \right\|_2^2 + \lambda_I \mathcal{R}_I(\boldsymbol{I}) && (5.2)\\
\boldsymbol{\phi}_{k+1} = f_{\mathrm{mot}}(\boldsymbol{I}_{k+1}) &= \arg\min_{\boldsymbol{\phi}} \sum_{n=1}^{N} \left\| \boldsymbol{A}^{(n)} \mathcal{T}_{\boldsymbol{\phi}^{(n)}} \boldsymbol{I}_{k+1} - \boldsymbol{y}^{(n)} \right\|_2^2 + \lambda_\phi \mathcal{R}_\phi(\boldsymbol{\phi}) && (5.3)\\
k \leftarrow k+1. &
\end{aligned}$$

Starting with the approximations $\boldsymbol{I}_0$ and $\boldsymbol{\phi}_0$, the two optimisation problems are iterated, each using the current approximate of the other.

As stated in previous chapters, Equation (5.2) can be tackled using convex optimisation. Since some parametrised motion models (including the rigid motion model introduced in Section 4.4) are differentiable, gradient descent methods are a popular approach to tackle Equation (5.3) as well. However, these methods commonly struggle when encountering stronger subject motion, since the problem is generally non-convex.

Thus, the method presented in this chapter employs a multi-stage motion parameter search algorithm which covers a large range of motion. This is coupled with an image recovery algorithm using convex methods in an alternating fashion to jointly estimate images and their underlying motion.

5.4 Grid-Search for Motion Parameters

Since it is not feasible to search the motion parameter space for all measurements of a multi-shot measurement simultaneously, the objective motion estimation problem in Equation (5.3) is split into a number of N smaller sub-problems for each time step by splitting the sum into its components. Minimizing the N partial problems then yields a solution for the overall problem. For a fixed image approximation $\boldsymbol{I}_{approx}$, the objective squared error in the n-th sub-problem caused by motion is defined by

$$\hat{f}^{(n)}\left(\boldsymbol{\phi}^{(n)}\right) = ||\boldsymbol{A}^{(n)}\mathcal{T}\left(\boldsymbol{\phi}^{(n)}, \boldsymbol{I}_{approx}\right) - \boldsymbol{y}^{(n)}||_2^2, \tag{5.4}$$

with $\boldsymbol{\phi}^{(n)}$ being the approximated parametrisation for the motion in measurement $\boldsymbol{y}^{(n)}$.

Using the N partial objective functions, the presented algorithm evaluates a search space of possible values for all $\boldsymbol{\phi}^{(n)}$ to find a set of motion parameters which minimizes each error term while not changing too rapidly, which in this approach is enforced using an explicit restriction on the considered motion parameters.

For this explicit enforcement, the search space for the overall optimisation problem is restricted to the space of motion sequences which are *admissible* by not changing rapidly over time.

The set of admissible motion sequences $\boldsymbol{\Xi}$ containing sequences which are sufficiently smooth can be defined as

$$\boldsymbol{\Xi} = \{\boldsymbol{\phi} \mid \alpha(\boldsymbol{\phi}^{(n)}, \boldsymbol{\phi}^{(n-1)}) = 0 \;\forall\, n \in \{2, \dots, N\}\}. \tag{5.5}$$

The set encompasses all sequences of motion parameters in which the motion states between any two time steps are not *too far apart*, assessed by a compatibility function $\alpha(\boldsymbol{\phi}_a, \boldsymbol{\phi}_b)$ between the two motion states $\boldsymbol{\phi}_a$ and $\boldsymbol{\phi}_b$. The compatibility function returns 0 if the difference is deemed acceptable, and $+\infty$ if the states are too far apart.

Let the motion parametrisation for a single time step be composed from M scalar values, so that $\boldsymbol{\phi}^{(n)} = [\phi^{(n,1)}, \phi^{(n,2)}, \dots \phi^{(n,M)}]$ and $|\phi^{(n,m)}| < b^{(m)} \approx G^{(m)} \ \forall \ 0 < n \leq N$, where $b^{(m)}$ is an unknown upper bound for the m-th parameter and $G^{(m)}$ is an approximation of $b^{(m)}$.

Then, the space of motion parameters can be sampled with a step size $\gamma^{(m)}$ to form a M-dimensional lattice

$$\boldsymbol{\Omega} = \boldsymbol{V}^{(1)} \times \boldsymbol{V}^{(2)} \times \dots \times \boldsymbol{V}^{(M)} \tag{5.6}$$

with $\boldsymbol{V}^{(m)} = \{-G^{(m)}, -G^{(m)} + \gamma^{(m)}, -G^{(m)} + 2\gamma^{(m)}, \dots, G^{(m)}\}$ and $\times$ being the Cartesian product. The parameters $G^{(m)}$ and $\gamma^{(m)}$ are crucial for the algorithm, as they define the maximum value for every parameter as well as its resolution and thereby the computational effort required by the method. Each point $\boldsymbol{\Omega}_v$ on the lattice corresponds to a single discrete motion state, with $0 < v \leq V$ and V being the total number of states on the lattice.

The objective error value given by Equation (5.4) is calculated for every partial measurement and every available motion state using the current image approximation and is collected into a matrix $\boldsymbol{Q} \in \mathbb{R}^{N \times V}$ with entries $Q^{(n,v)} = \hat{f}^{(n)}(\boldsymbol{\Omega}_v)$. A sequence of states on $\boldsymbol{\Omega}$ which minimises the objective function in Equation (5.3) is then found by solving a path search problem through $\mathbf{Q}$, so that the sequence is in $\boldsymbol{\Xi}$. Multiple path search algorithms are possible candidates, but here and in the original publication [1], a modified variant of the Viterbi algorithm is used.

The original Viterbi algorithm [75] is given a sequence of codes and tasked with finding the sequence of hidden states in a hidden Markov model that maximizes the posterior probability of generating the measurements. Hidden states are allowed to transition, with the transition probabilities as well as the probabilities that a given state will generate a measurement being predefined.

The path search algorithm incorporated in this method uses a variant of the Viterbi algorithm to find a sequence of motion states for a given multi-shot measurement. In this problem formulation, instead of maximizing the probability of a code sequence being emitted by a sequence of states as in the original method, a misalignment term resulting from a motion state sequence on the partial k-space measurements is minimized.

The state transition probabilities of the original Viterbi model are replaced with the compatibility function $\alpha(\cdot,\cdot)$ used to build the admissible set of states in Equation (5.5). The exact definition depends on the motion model, but the basic idea is to restrict the difference between sequential motion states. The compatibility function is used as a multiplicative term when evaluating state transitions. Since the modified Viterbi algorithm presented here aims to find a path causing minimal error, this multiplication effectively bars incompatible transitions from being considered. This serves as a form of regularisation, as it prohibits the algorithm from finding unfeasible movements which vary too quickly between the short measurement time steps.

Since the path search algorithm is designed to work in a multi-stage iterative framework, the result from prior iterations has to be considered when calculating the compatibility between states.

A path through the space of motion parameters which minimizes the overall error is found through the path search shown in Algorithm 2 in pseudo-code. There, $\tilde{\boldsymbol{\phi}}$ is the previous motion state estimated by the algorithm and $\bullet$ is an operation connecting two motion states, so that $\mathcal{T}(\boldsymbol{\Omega}_1 \bullet \boldsymbol{\Omega}_2, \boldsymbol{I}) = \mathcal{T}(\boldsymbol{\Omega}_1, \mathcal{T}(\boldsymbol{\Omega}_2, \boldsymbol{I}))$. The exact nature of this operation again depends on the chosen motion model.

This algorithm is able to find the optimal path in $\mathcal{O}(VN)$, once $\boldsymbol{Q}$ is calculated. The principle of the path search is illustrated in Figure 5.1. A greedy path search, which would start with the lowest possible error reachable in $n = 1$ and then propagate to the minimal error in a compatible motion state in $n = 2$ would offer results without having to explicitly calculate $\boldsymbol{Q}$. However, greedy methods would yield different, suboptimal results for the example given in Figure 5.1.

With $\boldsymbol{Q}$ being fairly expensive to set up, approximating parameters for a large region and with a high resolution would be infeasible using this approach. However, in most cases, the partial objective functions are 'smooth enough' for a small step size $\gamma^{(m)}$ so that local minima of the objective function in respect to a motion

Algorithm 2 (Modified) Viterbi path search

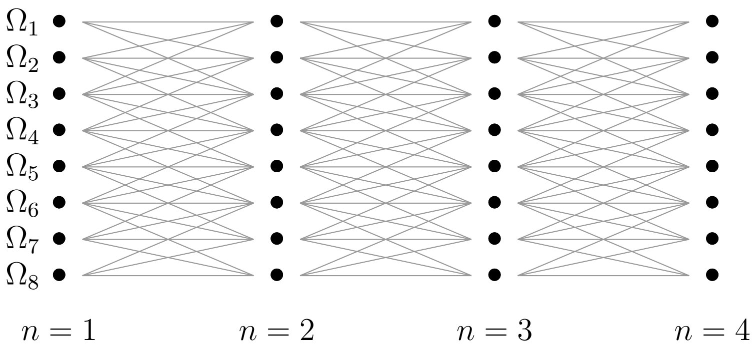
Inputs:
  $\boldsymbol{Q}, \boldsymbol{\Omega}, \tilde{\boldsymbol{\phi}}$
Initialize:
  $e^{(1,v)} \leftarrow Q^{(1,v)},\ v = 1, \ldots, V$ ▷ *Init. cumulative error matrix* $\boldsymbol{e}$
for n = 2 to N **do** ▷ *Iterate over all time steps*
  for v = 1 to V **do** ▷ *Iterate over all available grid positions*
    ▷ *Find the minimum error in the next time step from compatible motion states* ◁
    $i \leftarrow \arg\min_c \left(e^{(n-1,c)} + \alpha(\boldsymbol{\Omega}_v \bullet \tilde{\boldsymbol{\phi}}^{(n-1)}, \boldsymbol{\Omega}_c \bullet \tilde{\boldsymbol{\phi}}^{(n)})\right)$
    ▷ *Update cumulative error in* $\boldsymbol{e}$ ◁
    $e^{(n,v)} = e^{(n-1,i)} + Q^{(n,v)}$
    ▷ *Update path matrix for the current time step and motion state* ◁
    if $n\ =\ 2$ **then**
      $\boldsymbol{p}^{(n-1,v)} = \boldsymbol{\Omega}_i$
    else
      $\boldsymbol{p}^{(n-1,v)} = \left[p^{(n-2,v)},\ \boldsymbol{\Omega}_i\right]$
▷ *Return motion sequence associated with minimum cumulative error* ◁
$i \leftarrow \arg\min_c \left(e^{(N,c)}\right)$
$\boldsymbol{\phi}_{opt} = [p^{(N-1,i)}, \boldsymbol{\Omega}_i]$
output $\boldsymbol{\phi}_{opt}$

parameter are not missed on the sampled grid. Then, the approach can easily be extended to an iterative multi-resolution algorithm.

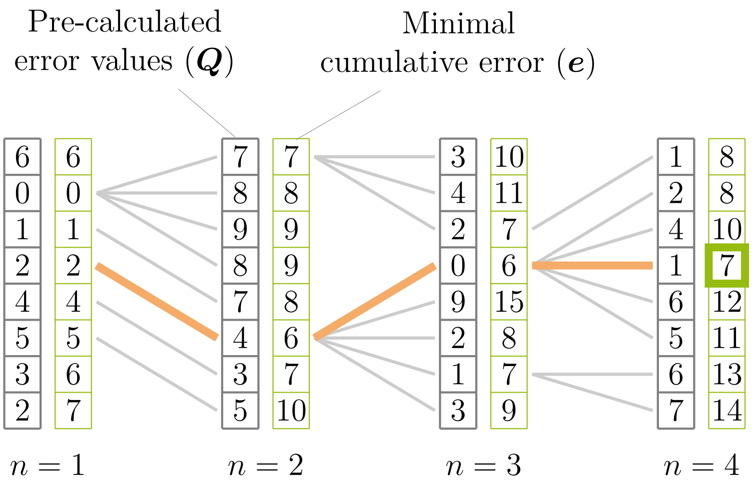
After an approximation for a motion sequence has been found by Algorithm 2, lowering the maximal search range $G^{(m)}$ and interval $\gamma^{(m)}$ for each parameter increases the resolution for the motion state lattice. Progressively refining $\boldsymbol{\Omega}$ and re-centring the lattice on the current approximation allows the approximation to reach subpixel accuracy while keeping the computational effort feasible.

Combining the path search algorithm for motion estimation with a motion compensated image recovery algorithm allows the definition of an efficient method for blind autofocusing. Alternating motion approximation and motion estimation and refining the search space between iterations allows for progressively improving image quality by the refinement of the motion approximation.

As in the original publication [1], this multi-resolution strategy with predefined



(a) Example: Allowed state transitions



(b) Example: Result of a (modified) Viterbi path search

Figure 5.1: Illustrations of the Viterbi path search algorithm employed for parameter searching. Figure 5.1a shows a possible setup of the path search algorithm. 8 discrete states $\Omega_1 ... \Omega_7$ are evaluated for 4 time steps. All possible transitions between motion states in the time steps are marked with a grey line. Here, the compatibility measure allows each state to transition to up to two states 'away' from itself. Figure 5.1b shows the result of a sample path search of the Viterbi algorithm for the setup from Figure 5.1a. Starting from $n = 2$, the algorithm iterates over all 8 available states and looks for the smallest cumulative error in the previous time step it transition to. After the end is reached, the best path is identified by choosing the smallest cumulative error. Each state in each time step is optimally reached by the path indicated by a grey line, with the overall best path being shown in orange.

starting values for maximal parameter search range $\boldsymbol{G}$ and step size $\boldsymbol{\gamma}$ is defined as in Algorithm 3. Here, the grid is refined by a factor of 2 between iterations, and the image $\boldsymbol{I}_{approx}$ used to evaluate the expression in Equation (5.4) is updated based on the current estimate of the motion parameters. This allows the multi-scale algorithm to iteratively sharpen the image and find improved motion sequence approximations.

Algorithm 3 encapsulates the alternating motion and image approximation. In each of the T stages, the values of the error matrix $\boldsymbol{Q}$ are calculated using the

Algorithm 3 Multi-resolution grid search motion parameter approximation

1: **Inputs:**
   $\boldsymbol{G}, \boldsymbol{\gamma}, \boldsymbol{A}, \boldsymbol{y}$
2: **Initialize:**
   $\boldsymbol{\phi}_1 \leftarrow \left[\boldsymbol{\phi}_1^{(1)}, \boldsymbol{\phi}_1^{(2)}, \dots \boldsymbol{\phi}_1^{(N)}\right],\ \boldsymbol{\phi}_1^{(i)} = \boldsymbol{0}^T,\ 0 < i \leq T$
   $\boldsymbol{I}_{approx} \leftarrow f_{img}(\boldsymbol{\phi}_t)$
3: **for** m = 1 to M **do** ▷ *Initialize parameter grid.*
4: $\quad \boldsymbol{V}^{(m)} \leftarrow \{-G^{(m)}, -G^{(m)} + \gamma^{(m)}, -G^{(m)} + 2\gamma^{(m)}, \dots, G^{(m)}\}$
5: $\boldsymbol{\Omega}^{(0)} \leftarrow \boldsymbol{V}^{(1)} \times \boldsymbol{V}^{(2)} \times \dots \times \boldsymbol{V}^{(M)}$
6: **for** k = 1 to K **do** ▷ *Iterate K times in total*
7: ▷ *Create error matrix $\boldsymbol{Q}$ using current image estimate.* ◁
8: **for** n = 1 to N **do**
9: **for** v = 1 to V **do**
10: $Q_{(n,v)} \leftarrow ||\boldsymbol{A}^{(n)}\mathcal{T}\left(\boldsymbol{\Omega}_v^{(m)} \bullet \boldsymbol{\phi}_k^{(n)},\ \boldsymbol{I}_{approx}\right) - \boldsymbol{y}^{(n)}||_2^2$
11: $\boldsymbol{\phi}_{k+1} \leftarrow \text{Viterbi}(\boldsymbol{Q}, \boldsymbol{\Omega}^{(m)}, \boldsymbol{\phi}_k) \bullet \boldsymbol{\phi}_k$
12: ▷ *Offset correction so that $\boldsymbol{\phi}_{k+1}^{(1)} = \boldsymbol{0}^T$* ◁
13: **for** n = 1 to N **do**
14: $\boldsymbol{\phi}_{k+1}^{(n)} \leftarrow \boldsymbol{\phi}_{k+1}^{(n)} \bullet \bar{\boldsymbol{\phi}}_{k+1}^{(1)}$
15: $\boldsymbol{I}_{approx} \leftarrow f_{img}(\boldsymbol{\phi}_{k+1})$
16: $\boldsymbol{\Omega}^{(m+1)} \leftarrow \boldsymbol{\Omega}^{(m)}/2$ ▷ *Refine grid between iterations*
17: $\boldsymbol{\phi}_{opt} \leftarrow \boldsymbol{\phi}_{k+1}$
18: $\boldsymbol{I}_{opt} \leftarrow f_{img}(\boldsymbol{\phi}_{k+1})$
19: **output** $\boldsymbol{\phi}_{opt}$

current image approximation estimated by $f_{img}(\boldsymbol{\phi}_t)$ and the motion model on all current grid positions. While being shown as serial computation, this process can be very efficiently parallelized.

A new motion estimate is then approximated by the path searching algorithm, taking into account past motion estimates. As only relative motion in the image is relevant to its artifacts, the algorithm might estimate an unwanted motion offset. To remedy this, the first image in the sequence is chosen as the motion-free target. This is enforced by shifting the estimate for each time step with the inverse of the motion for the first time step $\bar{\boldsymbol{\phi}}_{k+1}^{(1)}$ in line 14 in Algorithm 3.

After an iteration, the motion parameter grid is refined, and a new image estimate for the next iteration is found. The algorithm returns its final estimation of motion and image as $\boldsymbol{\phi}_{opt}$ and $\boldsymbol{I}_{opt}$, respectively.

5.5 Motion Model

The shortfall of the algorithm described in the previous section is that it scales exponentially with dimensionality of the motion model. This prevents its use in high-dimensional motion models such as dense vector field estimation. However, in contrast to gradient descent algorithms for the same task, it only depends on the forward transform of the motion model and does not require the model to be differentiable. In this work, the motion model is restricted to rotations and translations in both image directions. The motion approximated in each step n of the measurement is characterised by

$$\boldsymbol{\phi}^{(n)} = \begin{bmatrix} \boldsymbol{\phi}_{shift}^{(n)} \\ \phi_{rot}^{(n)} \end{bmatrix}, \tag{5.7}$$

using translation parameters $\boldsymbol{\phi}_{shift} \in \mathbb{R}^2$ and a single rotation parameter $\boldsymbol{\phi}_{rot} \in \mathbb{R}$ for each time step.

The deformation operator of this model is thus defined as a function of the image coordinates:

$$\mathcal{T}\left(I(\boldsymbol{x}^{(n)}), \boldsymbol{\phi}\right) = I\left(\boldsymbol{R}(\phi_{rot}^{(n)}) \cdot (\boldsymbol{x} + \boldsymbol{\phi}_{shift}^{(n)})\right).$$

Here, $R(\cdot)$ is the rotation matrix already defined in Equation (4.24).

As discussed in Section 4.4, both rotation and translation allow an equivalent operation in frequency space. Hence, the motion operation can be incorporated into $\boldsymbol{A}$ by directly modifying the known system matrix. This allows the definition of the combined operator $\boldsymbol{A}_{\boldsymbol{\phi}}^{(n)}$ for a given k-space trajectory and motion approximation at a time step n as

$$\boldsymbol{A}_{\boldsymbol{\phi}}^{(n)} = \begin{bmatrix} a_n^{(1,1)}, & a_n^{(1,2)}, & \dots, & a_n^{(1,s_p)} \\ a_n^{(2,1)}, & a_n^{(2,2)}, & \dots, & a_n^{(2,s_p)} \\ \vdots & \vdots & & \vdots \\ a_n^{(V,1)}, & a_n^{(V,2)}, & \dots & a_n^{(V,s_p)} \end{bmatrix} \tag{5.8}$$

$$a_n^{(m,v)} = \exp(-i2\pi\langle \boldsymbol{r}_n^{(m,v)}, \boldsymbol{x}_v\rangle + \langle \boldsymbol{k}^{(n,m)}, \boldsymbol{\phi}_{shift}\rangle) \tag{5.9}$$

$$\boldsymbol{r}_n^{(m,v)} = \boldsymbol{R}(\phi_{rot})\boldsymbol{k}^{(n,m)}. \tag{5.10}$$

In the above equation, the frequency coordinates in $k^{(n,m)}$ define the MRI sampling trajectory through k-space for the time step n.

Stacking the N matrices to an overall system matrix $\boldsymbol{A_\phi}$ allows to formulate the problem solved by the ADMM described in Section 4.3 as

$$\boldsymbol{I}_{opt} = \arg\min_{\boldsymbol{I}} \frac{1}{2}\|\boldsymbol{A_\phi I} - \boldsymbol{y}\|_2^2 + \lambda_{\boldsymbol{I}}\|W\boldsymbol{I}\|_1. \tag{5.11}$$

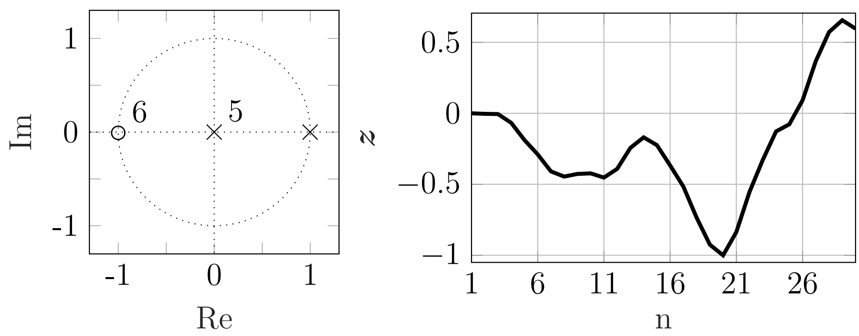
The transformations imposed by the combined motion and system matrices can be approximated through the use of an NFFT, as the motion model only effects a coordinate transform on the frequency space data and a phase shift. The transposed form of the transformation required by the CGNR is similarly formed by a coordinate transform and phase shift using the negative of the approximated motion before applying an inverse NFFT. Therefore, the reconstruction problem formulated in Equation (5.11) can be solved very efficiently for this motion model.

5.6 Test Setup

To evaluate this model, a PROPELLER MRI measurement with 30 acquisition blades was simulated. As base image, a simulated tomography of the human brain (from the Brainweb dataset [76]) with a size of 160×160 pixels was used. As the presented autofocusing algorithm is not data-driven, the effect of the underlying image content on the result is negligible for the chosen motion model, and all experiments are carried out using a single base image for easier comparison. The base image was padded with zeros prior to simulating motion as to prevent it from leaving the field of view. While the frequency space formulation of the rigid motion model is natively equipped to handle wrap-over artifacts as they might present in MRI images, the initial estimate of the image required for the algorithm would be affected by the aliasing. The padding extends the images to a size of 200×200 pixels.

Using the optimal PROPELLER blade height specified in Equation (2.17) results in 30 blades of size 200×10 sampled coefficients. Between acquisitions, the blade was rotated according to the golden angle of 111.65°.

Translational rigid movements as well as rotations were used to move the underlying images to corrupt the measured k-space, assuming a static object during acquisition of a single blade and movement only between two measurement steps. To simulate object motion, approximately smooth random motion curves were



(a) Z-plane representation

(b) Noise response

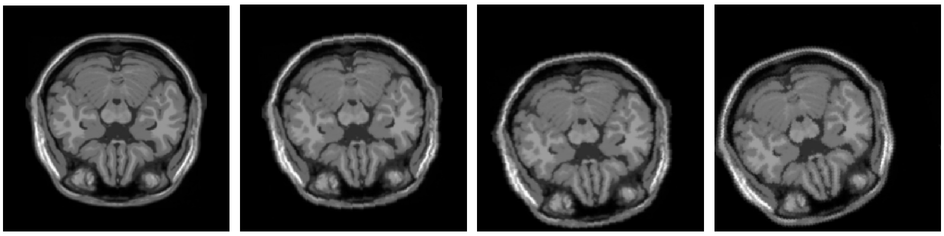
Figure 5.2: Z-plane representation and noise response of the ARMA filter model used to generate random sequences in the experiments. The filter exhibits a strong low-pass characteristic, resulting in smooth curves being generated as noise responses.

generated through the use of an Autoregressive Moving Average (ARMA) model. In this model, an approximately smooth signal $\boldsymbol{z} \in \mathbb{R}^M$ is defined by

$$z(n) = \sum_{i=1}^{a} \alpha_i z(n-i) + \sum_{j=1}^{b} \beta_j \eta(n-j+1) \tag{5.12}$$

where $\boldsymbol{\eta}$ is a zero-mean Gaussian noise signal of length M. Equation (5.12) corresponds to the application of a discrete filter with infinite impulse response characterised by the feedback parameters $\boldsymbol{\alpha} = \alpha_1, \alpha_2 ... \alpha_a$ and feedforward parameters $\boldsymbol{\beta} = \beta_1, \beta_2 ... \beta_b$. In these experiments, a short feedback path with $a = 1$, $\alpha_1 = 0.999$ and a feedforward filter with a lowpass characteristic and $b = 7$, $\boldsymbol{\beta} = [1, 6, 15, 20, 15, 6, 1]$ was chosen. The resulting filter exhibits a very strong low-pass characteristic, blocking high frequencies in the noise signal entirely. The z-plane representation of the filter and a sample Gaussian noise response are shown in Figure 5.2.

Using the this model, smooth random curves were generated for each of the three parameters of the simulation. To assess the capability of the algorithm to approximate different motion strengths, a number of experiments with different maximal translational and rotational amplitudes were conducted. Specifically, the maximal rotation amplitude was scaled to $0°, 1°, 5°, 10°$ and $20°$, and the maximal translation scaled to $0, 5, 10, 20$ and 30 pixels for both image directions. As stated in Section 5.5, rotation is applied before translation.



(a) $\boldsymbol{I}^{(1)}$ (b) $\boldsymbol{I}^{(10)}$ (c) $\boldsymbol{I}^{(20)}$ (d) $\boldsymbol{I}^{(30)}$

Figure 5.3: An example of the simulated motion used in the experiments. Here, 4 images from different time steps in a sequence afflicted by rotation and translation are shown.

Samples from a motion simulation are shown in Figure 5.3. Here, 4 time steps from the sequence are shown.

All combinations between rotation and translation amplitudes were evaluated 100 times with randomly generated motion, resulting in 2500 experiments.

After simulating the PROPELLER measurement, the multi-resolution method presented in Algorithm 3 is used to recover the base image and the motion curves solely from the simulated k-space data.

The motion model used for motion compensation approximates translational and rotational parameters. As in Section 5.5, a single motion state for a time step n is therefore defined by $\boldsymbol{\phi}^{(n)} = [\boldsymbol{\phi}_{shift}^{(n)}, \phi_{rot}^{(n)}]^\intercal$, with a translational component $\boldsymbol{\phi}_{shift}^{(n)} \in \mathbb{R}^2$ and a scalar rotation parameter $\phi_{rot}^{(n)}$.

In the original publication in [1], the motion models used for simulation and approximation incorporated only translational movement, therefore approximating only $\boldsymbol{\phi}_{shift}$ for each measurement step. For this thesis, the model was expanded to approximate rotational motions in each measurement as well. As such, the results in [1] are not exactly reproduced in this work. However, they are covered by the special case where the amplitude of rotational motion is set to 0.

For rigid motions, the composition operation $\bullet$ between motion states required in Algorithm 3 is parameter-wise addition. The indicator function α is chosen to be

$$\alpha(\boldsymbol{\phi}^{(n)}, \boldsymbol{\phi}^{(n-1)}) = \xi(\|\boldsymbol{\phi}_{shift}^{(n)} - \boldsymbol{\phi}_{shift}^{(n-1)} - \tau_{shift}\|_2) + \xi(\|\phi_{rot}^{(n)} - \phi_{rot}^{(n-1)}\|_2 - \tau_{rot}), \tag{5.13}$$

$$\xi(a) = \begin{cases} 0 \ \text{ if } a \leq 0 \\ +\infty \ \text{ else} \end{cases}$$

returning 0 if both the differences in translation and rotation between steps are within acceptable ranges defined by the control parameters τ_{shift} and τ_{rot}, and $+\infty$ if not. This assumes that translation and rotation are not correlated.

Preliminary numerical evaluations showed that the partial objective functions for each of the partial measurements are almost convex in a region around the true minimum. This allows the use of the multi-resolution approach described in Section 5.4 without optimising to a false local minimum. Figure 5.4 shows the function value of a partial function evaluated for a translation range of ± 30 pixels in both spatial dimensions, evaluated at a higher resolution than used in the experiments.

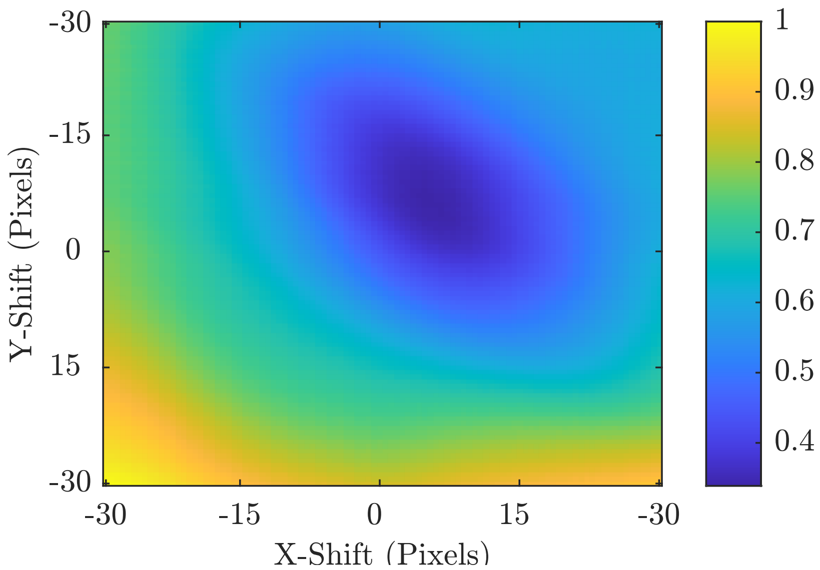


Figure 5.4: Objective function evaluated for translations in range ± 30 pixels. Values shown are normalised into range $[0, 1]$. The function surface is convex in this range, with a minimum at about $[7, 7]$.

In this setup, the maximum motion approximated by the algorithm as well as the resolution are freely chosen as input parameters to the multi-resolution method. With the motion model in use, the motion state grid spans three dimensions, so that both the grid size parameters $\boldsymbol{G}$ as well as the resolution parameters $\boldsymbol{\gamma}$ are $\in \mathbb{R}^3$. The first two values of $\boldsymbol{G}$ and $\boldsymbol{\gamma}$ control the maximal amplitude and resolution for the approximation of translational movements, and the third parameter of each controls the maximum amplitude and resolution of the approximation of rotation.

Starting with a maximum approximated motion of $G^{(1)} = G^{(2)} = 20$ px and step size $\gamma^{(1)} = \gamma^{(2)} = 2$ px as well as $G^{(3)} = 20°$ and $\gamma^{(3)} = 2°$, all parameters in $\boldsymbol{G}$ and

$\boldsymbol{\gamma}$ were halved between each of the $T = 7$ stages of the multi-resolution algorithm. The method can therefore reach a maximal resolution of 0.16 px and 0.16°.

The maximum allowed parameter distance used in Equation (5.13) was set dynamically, depending on the motion parameter grid size. Specifically, τ_{shift} was set to $0.2 \cdot G^{(1)}$, and τ_{rot} was set to $0.2 \cdot G^{(3)}$ in each stage.

As the motion compensated reconstruction method f_{img} required in the method to approximate progressively better images in each iteration, the ADMM method incorporating conjugate gradient descent introduced in Section 4.3 was used. A wavelet regularisation parameter $\lambda_I = 1$ and a 3 staged two-dimensional wavelet decomposition using the Biorthogonal 4.4 mother wavelet were chosen as priors, and the ADMM relaxation parameter ρ was set to 1. The parameters chosen for the image recovery algorithm were found empirically in preliminary experiments, with different choices ρ and wavelet families only marginally influencing the outcome of recovery. For performance reasons, a reconstruction with 5 ADMM iterations was used in each stage of the multi-resolution algorithm, and a longer one with 30 ADMM steps to reconstruct the final image. At convergence, the soft thresholding using the chosen wavelet transform compresses the image to 13% of its coefficients.

In this evaluation, two error functions are used to quantize registration and reconstruction results. One suitable metric is the ℓ_2-error, which was already used to define the optimisation problems f_{img} and f_{mot}. The scalar ℓ_2-error is defined as the squared Euclidean norm of the difference between two signals $\mathbf{a}$ and $\mathbf{b}$ of the same size as

$$\ell_2(\mathbf{a}, \mathbf{b}) = \frac{1}{2}\|\mathbf{a} - \mathbf{b}\|_2^2. \tag{5.14}$$

For images, the error is the sum of squares of element-wise brightness differences between the two inputs. The intuitive nature of the squared error, along with its continuos differentiability makes it a commonly used difference function for optimisation, among others in computer vision tasks.

The objective difference might be insufficient for the quantification of the difference between natural images. The ℓ_2 error is highly susceptible to image scaling and alignment differences, and small errors in either category may lead to large ℓ_2-error values. This might not accurately capture the desired properties of a registration, however. Medical images are made to be evaluated by a human observer, who will generally be indifferent to small translational offsets between images - as long as the pictured structures are visible clearly and free of artifacts.

This shift in perspective motivated the development of image difference functions which focus on human observers, for example the *Structural Image Similarity Measure* (SSIM) [77]. In contrast to the absolute error calculated by the ℓ_2-loss, the SSIM is a composite measure that aims to quantify a subjective change a human observer might experience between images without directly incorporating a human vision model. This change is formulated as a comparison of image luminosities and contrasts. It also features a correlation term, which aims to capture the structural information of the images. In contrast to the ℓ_2-loss, structural information as a concept assumes that spatially close pixels express information about the image structure which is important to human perception.

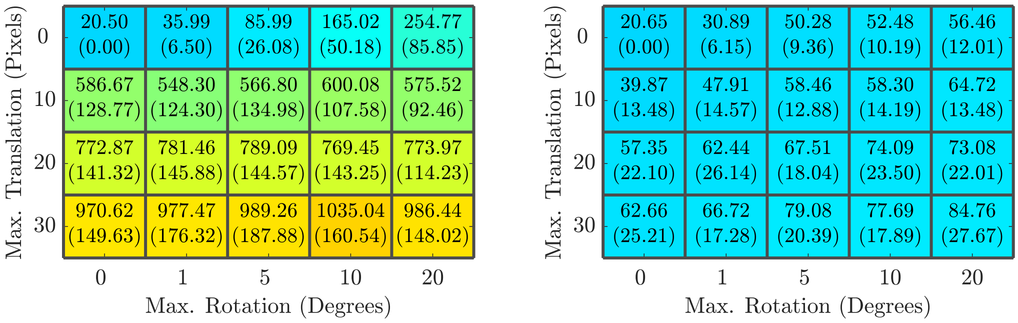
The SSIM for images $\mathbf{a}$ and $\mathbf{b}$ is defined by

$$\begin{aligned} \mathrm{SSIM}(\mathbf{a},\mathbf{b}) &= \underbrace{\frac{\sigma_{\mathbf{ab}}+c_1}{\sigma_{\mathbf{a}}\sigma_{\mathbf{b}}+c_1}}_{\text{Correlation}} \cdot \underbrace{\frac{2\bar{a}\bar{b}+c_2}{\bar{a}^2+\bar{b}^2}+c_2}_{\text{Mean luminance}} \cdot \underbrace{\frac{2\sigma_{\mathbf{a}}\sigma_{\mathbf{b}}+c_3}{\sigma_{\mathbf{a}}^2+\sigma_{\mathbf{b}}^2}+c_3}_{\text{Contrast}} \qquad (5.15)\\ \text{where } \bar{a} &= \frac{1}{s_p}\sum_{s=1}^{s_p} a_s, \qquad \sigma_{\mathbf{a}}^2 = \frac{1}{s_p-1}\sum_{s=1}^{s_p}(a_s-\bar{a})^2\\ \sigma_{\mathbf{ab}} &= \frac{1}{s_p-1}\sum_{s=0}^{s_p}\Big[(a_s-\bar{a})(b_s-\bar{b})\Big]\\ \text{and } \mathbf{a} &= [a_1,a_2,\dots,a_{s_p}],\mathbf{b}=[b_1,b_2,\dots,b_{s_p}] \end{aligned}$$

with s_p as the total number of pixels in the images.

The scalars c_1, c_2, c_3 are stabilizing values, which depend on the dynamic range of the input images. Since all images in this work are in the same dynamic range of $[0,1]$, the coefficients can be given explicitly as $c_1 = 10^{-4}$, $c_2 = 9\cdot10^{-4}$, $c_3 = 4.5\cdot10^{-4}$. The SSIM is bounded between -1 and 1, with a value of 1 indicating that $\mathbf{a}$ and $\mathbf{b}$ are equal and -1 denoting perfect anti-correlation of the images.

The ℓ_2-error and SSIM were evaluated between the unmoving base image $\boldsymbol{I}^{(1)}$ used in the simulations and reconstruction $\boldsymbol{I}_{opt}$ to quantify the recovery quality of image and motion. Both metrics are also evaluated on a recovery of the image using the same recovery algorithm but assuming $\boldsymbol{\phi}^{(n)} = \mathbf{0}$ for all steps, dubbed the *naive* recovery.



(a) ℓ_2-error (naive recovery) **(b)** ℓ_2-error (motion compensated recovery)

Figure 5.5: Effect of motion compensation on ℓ_2-error for all combinations of simulated translation and rotation strengths. In each cell, the mean ℓ_2-error between approximated image and ground truth as well as its standard deviation over all experiments is shown. For the ℓ_2 error, lower values (coloured blue) indicate higher image quality.

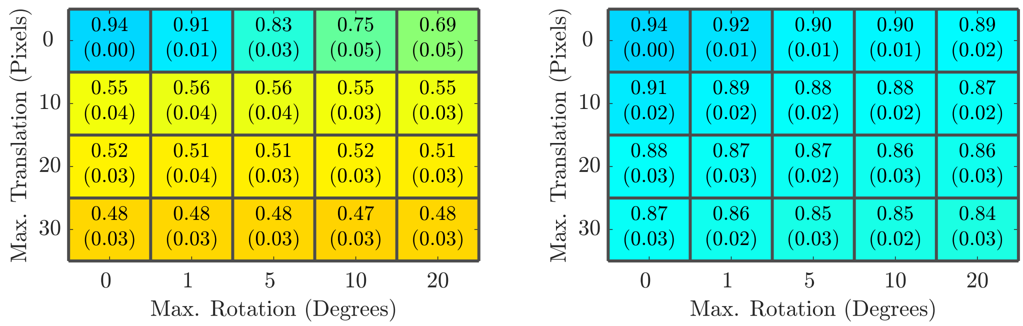
5.7 Results

The presented algorithm is able to recover sharp images even for measurements which are highly afflicted by motion. Even in reconstructions where initially, no anatomic structure was visible, the algorithm is able to produce subjectively good and sharp images.

Figure 5.5 shows the mean ℓ_2-error between base image and images recovered from the simulated measurements without and with motion compensation.

Since PROPELLER oversamples the central region of k-space while undersampling high frequency regions, a small ℓ_2-error is to be expected even when no motion is present, as seen in the evaluations of small motion amplitudes. Due to this undersampling, a perfect reconstruction with a zero ℓ_2 error is not achievable even for motion-less cases. However, in the presence of simulated motion, the ℓ_2-error is robustly decreased for all motion strengths, yielding near perfect results for smaller translational movements.

From the results, a very strong improvement in image quality can be seen for all motion strengths. For a purely translational motion (e.g. maximal amplitudes $\phi_{shift} = 30\text{px}$ and $\phi_{rot} = 0$ in the results), a reduction of more than 90% can be observed in the ℓ_2-error when using motion compensation in comparison to the naive case.



(a) SSIM (no motion compensation) **(b)** SSIM (with motion compensation)

Figure 5.6: Effect of motion compensation on SSIM for all combinations of simulated translation and rotation strengths. In each cell, the mean SSIM between approximated image and ground truth as well as its standard deviation over all experiments is shown. For the SSIM, higher values (coloured blue) indicate higher image quality.

When combining rotation and translation, the error in naive recovery caused by rotation seems to quickly overpower the error caused by translation - as seen in any given row in Figure 5.5a. This causes the difference in ℓ_2-error to be less drastic when comparing 'combined' motions. For example, motion compensating for maximal values of $\boldsymbol{\phi}_{shift} = 30$px, $\boldsymbol{\phi}_{rot} = 20°$ offers a slightly decreased relative reduction in the ℓ_2-error than motion compensating for maximal values of $\boldsymbol{\phi}_{shift} = 30$px, $\boldsymbol{\phi}_{rot} = 0°$.

In case no motion is present, approximation errors in the algorithm sometimes cause a slight additional error compared to the naive case. This results in the mean ℓ_2-error measured for experiments with $\boldsymbol{\phi}_{shift} = 0$px, $\boldsymbol{\phi}_{rot} = 0°$ to be slightly better in the naive case. This indicates that introducing an additional, more 'traditional' regularisation term associating a cost to the strength of approximated motion might be beneficial as well.

Figure 5.6 shows the mean SSIM between base image and naive reconstruction as well as between base image and motion compensated reconstruction. These results are qualitatively very similar to the mean ℓ_2-error results shown above, as application of the motion compensation algorithm robustly improves image quality. Of note is that the small approximation errors for the case where no motion is present, which lead to slightly worse ℓ_2-errors for the motion compensated case, do not seem to impact SSIM values.

For easier comparison, Figure 5.7 shows the difference in the two evaluated metrics between motion compensated and naive recovery. As shown in the figure,

Max. Translation (Pixels) \ Max. Rotation (Degrees)	0	1	5	10	20
0	0.15 (0.00)	-5.10 (6.49)	-35.72 (28.22)	-112.54 (50.80)	-198.32 (86.25)
10	-546.80 (128.29)	-500.39 (128.29)	-508.34 (136.63)	-541.77 (110.54)	-510.79 (95.45)
20	-715.52 (145.70)	-719.02 (144.24)	-721.58 (150.30)	-695.36 (146.22)	-700.89 (113.12)
30	-907.96 (153.80)	-910.75 (178.75)	-910.18 (187.82)	-957.35 (164.67)	-901.68 (155.03)

(a) Difference (ℓ_2-error)

Max. Translation (Pixels) \ Max. Rotation (Degrees)	0	1	5	10	20
0	-0.00 (0.00)	0.01 (0.01)	0.06 (0.04)	0.15 (0.05)	0.20 (0.05)
10	0.36 (0.05)	0.33 (0.05)	0.32 (0.05)	0.33 (0.04)	0.32 (0.03)
20	0.37 (0.04)	0.36 (0.04)	0.36 (0.04)	0.34 (0.05)	0.35 (0.04)
30	0.40 (0.05)	0.39 (0.04)	0.37 (0.04)	0.38 (0.04)	0.37 (0.05)

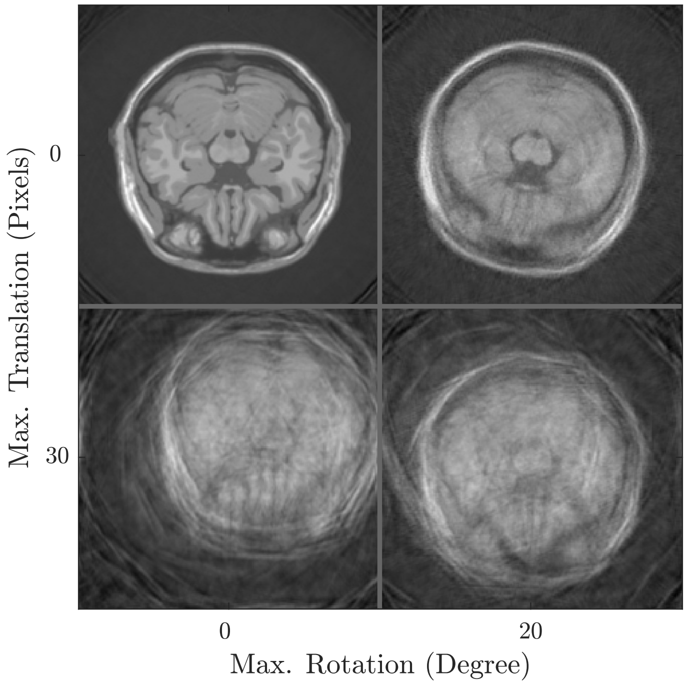
(b) Difference (SSIM)

Figure 5.7: Mean difference achieved when using motion compensation for a) mean ℓ_2-error and b) SSIM, calculated as the substraction of the value of the metric of the motion compensated image from the value for a reconstruction without motion compensation. Again, mean and standard deviation are shown for each combination of simulated motion amplitudes.

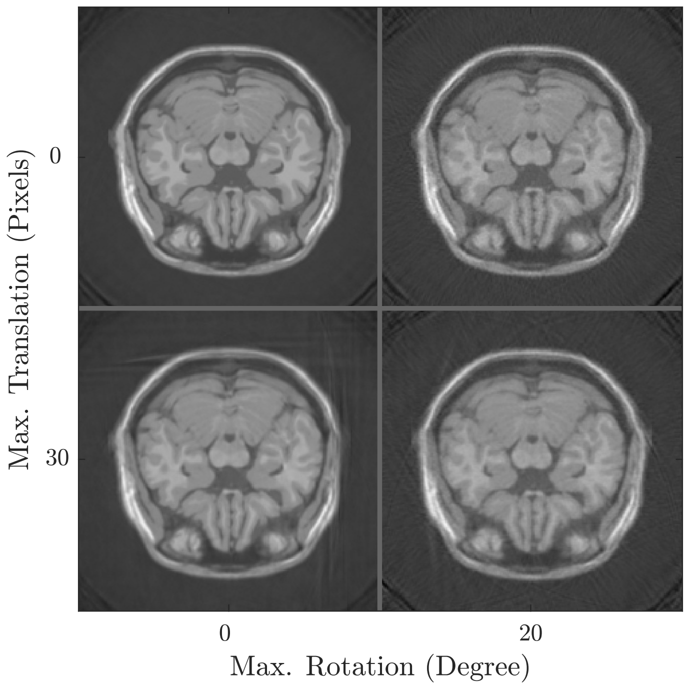
the presented motion compensation led to an increase in SSIM of up to 0.4 for very strong motion amplitudes.

The effect of motion compensation on image reconstruction is shown exemplary in Figure 5.8. Obviously, motion compensation drastically improves image quality, leaving only slight artifacts for very high translation amplitudes and blurring for strong rotation while no image details can be identified in the non-motion compensated reconstructions.

Using parallel computing and dynamically pre-computing the necessary NUDFT-I-transforms, the calculation of each state error matrix $\boldsymbol{Q}$ takes about 10 seconds on a conventional CPU. Each image recovery using the ADMM algorithm requires about 5 seconds. Including the path search algorithm, the presented motion compensated image recovery has a run time of about 2 minutes per image, which could be sped up further by a CUDA implementation.



(a) Without Motion Compensation



(b) Motion compensated

Figure 5.8: Effect of motion compensation on image reconstruction. Figure 5.8a shows the non-motion compensated reconstruction for simulations with translation amplitudes 0 and 30 pixels and rotation amplitudes of 0 and 20 degree, representing the extremes of the evaluated motion parameter range.

6

Dense Vector Field Based Motion Estimation

Contents

6.1 Problem Formulation

Up to this point, parameter-based motion descriptions were used to describe the movement afflicting an image during an MRI scan with the goal of incorporating the estimation in an enhanced recovery algorithm. However, for more complex real-world scenarios, model-based description of motion is insufficient as many organic structures inside a living body exhibit a complex motion profile.

The complex and highly variable nature of the motion expected in a scan of a human torso for example can not be suitably modelled by static, low-parametric models like the rigid model employed in Chapter 5. Other traditional registration algorithms, in which each image element is aligned by matching regions with similar appearance [78], [79], yield a parameter free approach and directly estimate dense vector fields to align images. However, they are typically very slow, as they do not take advantage from the parallelization capabilities of modern hardware. Furthermore, especially in the case of algorithms relying on equal-brightness constraints, they are very sensitive to image noise and are unable to handle occlusion artifacts in the images.

Since this thesis aims to solve a registration task between images resulting from partial MRI acquisitions, in which structured ringing artifacts caused by frequency space undersampling are unavoidable, these algorithms are not expected to perform too well.

To address the problem of direct dense vector field estimation especially for a medical context, the deep learning framework introduced in Chapter 3 is leveraged in this chapter to develop two neural network based image registration methods. The ability of networks to learn good representations of physiological motion vector fields was verified in a prior masters thesis at the Institute for Signal Processing [80].

This chapter will first present a novel convolutional attention mechanism used to model long-distance relationships in convolutional networks in Section 6.3. In Section 6.4, the dataset used to fit and evaluate the networks is shown and discussed, while Section 6.5 introduces the components of the loss functions used in the networks. The two network architectures which incorporate the new attention cell follow, with the research originally published at the EUSIPCO 2022 [2] being presented in Section 6.6.

In an extension to the published registration approach, this thesis also introduces a network architecture designed to predict a series of motion vector fields based on sampled k-spaces of an image series. While there are a small number of existing neural registration approaches, the task of jointly aligning a series of frequency subsampled images using neural networks and without motion models is novel. While not working iteratively and not natively including image reconstruction, this approach can still be categorised as autofocusing in the context of MRI related research. It's unpublished extension to series of frequency subsampled images is presented in Section 6.7.

6.2 State of the Art

Medical image registration tasks are starting to become another field of research where fruitful advances are being made by applying machine learning. Deep image registration uses artificial neural networks to produce deformation parameters or dense vector fields as a response to a reference image and a moved image. The output of the network then drives a transformation function to map the moved image to the reference image.

Since the space of motions for any given subject is limited, neural networks are a good fit for registration. As neural networks learn to fit their output to the data contained in their training sample set, they can learn efficient models of the motion occurring in their respective areas of deployment.

In the context of this work, *unsupervised* and *weakly supervised* training approaches are of particular interest. Both of these training schemes require no deformation ground truth, allowing more flexibility in choosing sets of data to optimise the network. While fully supervised approaches exist, datasets containing their required ground truths are rare.

Unsupervised training

In unsupervised registration, only the image data (of moved and reference image) are available during network training and testing. Instead of approximating a ground truth registration, the network aims to maximise a similarity measure between registered and reference image. The unsupervised training approach using a similarity function is schematically shown in Figure 6.1.

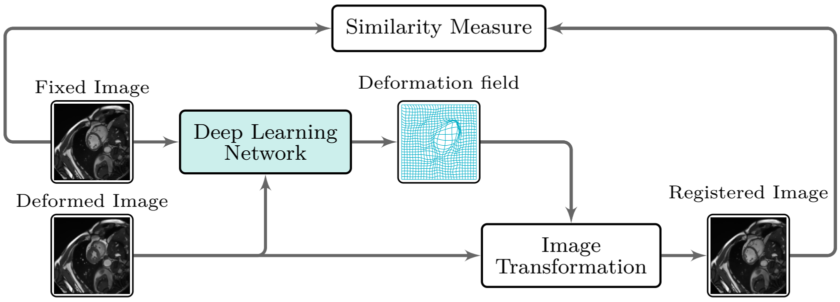


Figure 6.1: Flowchart for unsupervised deep registration. Only images are available in the training dataset. The network is tuned so that a similarity measure between a deformed image which is transformed by the network output and a reference image is maximised.

Balakrishnan et al. [81] published the most widely used unsupervised registration algorithm in 2019. VoxelMorph maps an input pair of images directly to a dense deformation field which is used to align the images. The mapping is parametrised by a U-Net architecture trained to minimise image distance based on pixel intensities. VoxelMorph performs similar to conventional image registration algorithms at a drastically increased speed.

Based on VoxelMorph, many approaches refined the original method. Dalca et al. [82] use a similar architecture to create a probabilistic mapping function between images. Their algorithm achieves similar performance to VoxelMorph, but guarantees diffeomorphic registration. Hoffmann et al. [83] train their variant solely on synthetic data, removing the reliance of VoxelMorph on an image dataset during training while outperforming the registration of the base network.

After the advent of attention supported networks with the Transformer architecture published by Vaswani et al. [49], networks leveraging attention and its transformer architecture also became popular in Computer Vision tasks, such as the Vision Transformer for image classification [84]. Based on this, several image regression tasks were approached using comparable networks. Zhu and Liu [85] employ an attention module on non-overlapping image patches, again as an extension to the VoxelMorph architecture. Their approach ensures invertibility by learning forward and backward transforms simultaneously, while offering improved performance. Wang et al. [86] use a Vision Transformer architecture for image registration, citing comparable accuracy to a U-Net trained for the same task.

Recently, research on networks leveraging attention mechanisms intensified not only for image classification but for image registration as well. Chen et al. [87]

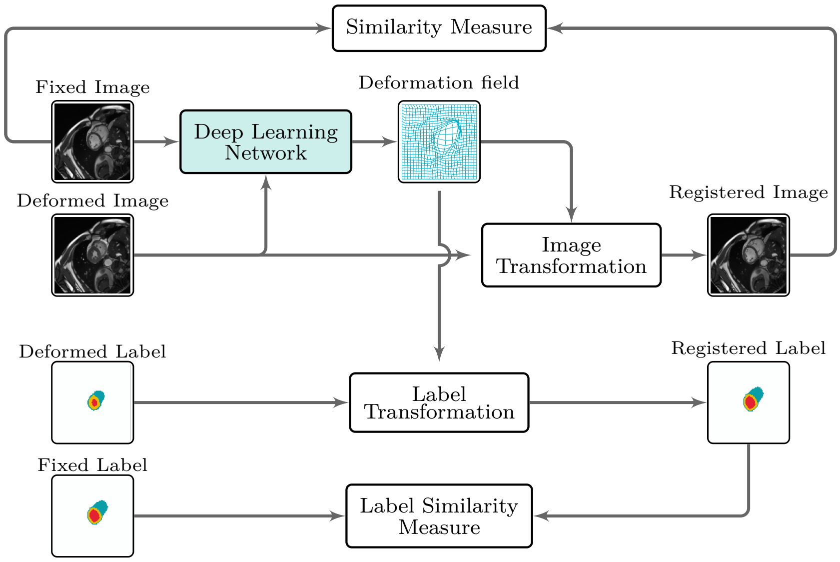


Figure 6.2: Flowchart for weakly supervised deep registration. Labels are available during training and used to drive a label similarity function (typically the Dice-Coefficient) which is maximised in tandem with the image similarity measure.

use a multi-scale network similar to a U-Net in conjunction with attention to infer deformation fields as a summation of several learned basis functions. Their network offers competitive performance compared to traditional registration algorithms and VoxelMorph.

Weakly supervised training

Several datasets contain some level of information pertaining to the deformation. While the actual ground truth deformation is absent in most cases, semantic label maps of the objects in the images or landmarks might be available. *Weakly supervised* training allows this data to be used during the training phase of the network. This approach is less common, as it requires a compatible dataset.

Weakly supervised training is depicted schematically in Figure 6.2 for a dataset containing semantic label maps of the images in question. The data shown in Figure 6.1 and Figure 6.2 originates from the ACDC dataset, which will be discussed in detail in Section 6.4.

Hu et al. [88] published an approach which uses a deep CNN to register images from three-dimensional MR and ultrasound measurements. Their dataset contained expert label maps and landmarks of a number of anatomical structures. Their network is trained to maximise the Dice coefficient between the image labels maps, using the landmarks to define a distance metric for network evaluation.

Xu et al. [89] use two U-Nets to learn a dense registration of two images and their semantic segmentation map in tandem. Their loss function incorporates terms quantising image similarity and anatomical similarity post-registration. They are able to show that both sub-networks exhibit improved accuracy by training them in tandem.

Hering et al. [90] use an ℓ_2-similarity term based on the semantic label maps in the ACDC dataset (described in Section 6.4) in conjunction with ℓ_2- and edge based similarity measures on the registered images to train a U-Net. They use a multilevel approach during training, in which every stage of the U-Net is encouraged to produce a valid image alignment. A refined version of their original approach integrates landmarks into the algorithm and achieves state-of-the-art accuracy for lung registration [91].

6.3 Convolutional Attention

Many modern algorithms for pixel-wise regression or classification use a U-Net [92] as their base architecture, as the inherent multi-level approach of the U-Net is well suited to analyse the overall structure as well as finer details of the input images. However, the inductive bias of the convolution layers which form the U-Net architecture make it difficult for the network to model long-distance relationships between image features, with the largest object a purely convolutional network can accurately model being dependent on its cumulative receptive field. For an N-layered convolutional network with kernel size k, the cumulative receptive field has a size of $k \cdot 2^N$, with long distance relationships exhibiting far longer gradient paths [93]. This problem has been tackled in recent research on image segmentation tasks, for example by using dilated convolution kernels [93], [94]. A recent approach is the use of attention schemes with fully connected embedding layers fed by flattened non-overlapping image patches [95], [96], a technique which also finds use in image classification and object detection [84].

While this enables top-of-the-class performance while lowering the computational load in comparison to conventional CNNs, the overall image structure is lost. This poses no problem for the image-wise classification task for which the architecture was developed, but is unsuitable for pixel-to-pixel tasks such as registration.

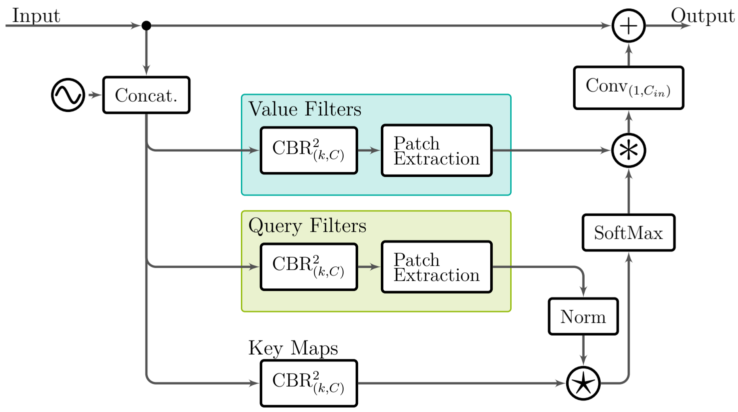


Figure 6.3: The residual attention cell. ⊗ is the positional encoding appended to the input channels.

In this chapter, a viable connection between convolutional networks and attention is presented. The novel attention layer represents the embeddings used in self-attention using convolutional operations.

To summarize the general idea of the attention mechanism described in Section 3.6, attention in a machine learning context aims to to generate a matching score as a measure of similarity between embeddings of its inputs. Attention in, for example, natural language processing calculates this measure based on embeddings of words of the input and output sentences and learns to measure contextual significance between parts of the sequence.

The attention mechanism developed for this registration task and presented in [2] aims to calculate this similarity between images or feature maps. Hence, it measures relevance between small image regions using convolutional operators. The aim is to combine the implicit bias of convolutional operations with the macroscopic structural analysis capabilities of the attention mechanism. The resulting cell is named the *Convolutive Self-Attention (CSA)*-layer[1] and is schematically shown in Figure 6.3.

As in most attention mechanisms, a positional encoding is used to supply spatial information to the embedding functions. For computer vision tasks, different positional encodings have been proposed, from scalar embeddings in which the additional information is the index of a patch in a raster order [84] to relative encodings between patch positions [97]. Since the exact choice of encoding does not seem to matter for the results in most applications [84], [98], in this work, a Cartesian

[1] In the original publication [2], the cell was named the *Residual Self-Attention (ReSAtt)* cell, foregoing the convolutional aspect.

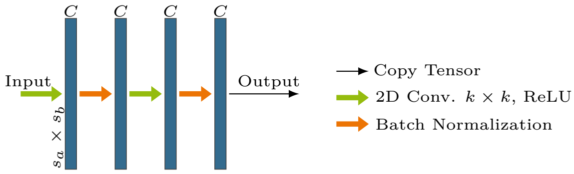


Figure 6.4: The CBR^2-layer used in every network and convolutional attention. A input feature map of size $s_a \times s_b \times C_{in}$ is processed in a series of convolutional layers with C kernels of size $k \times k$, batch normalization layers and ReLU-activation functions. Since the convolutions used are non-expansive, its output size will be $s_a \times s_b \times C$.

grid encoding is used. A hard coded, two-dimensional normalised Cartesian grid is concatenated to the feature map prior to patch extraction. This is not only lightweight and robust against different patch sampling schemes, but has also enabled coordinate transfer learning in convolutional neural networks in the past [99].

Where the attention mechanism used in the Transformer and Vision Transformer architectures uses the dot-product to calculate relevance between embedded token pairs, the CSA-layer uses cross-correlation between small data patches and feature maps. The result is roughly comparable to using the Vision Transformer to embed and process highly overlapping image patches, but using correlation and convolution operations in place of the fully connected embedding layers and matrix-vector operations used to drive scaled dot-product attention.

In the CSA-layer, Query ($\boldsymbol{Q}$), Key ($\boldsymbol{K}$) and Value ($\boldsymbol{V}$) tensors are created as embeddings of the input with appended positional encoding using learned 2D convolutional transformations. Specifically, each of the three tensors is created using a CBR^2 layer, which is a stack of two convolutions, batch normalisations and ReLU activation functions. The CBR^2 layer structure is also a building block of the U-Net-like architectures used in Section 6.6 and Section 6.7 and is pictured in Figure 6.4. Given an input feature map of size $s_a \times s_b \times C_{in}$, each of the three CBR^2-layers produces an embedding of size $s_a \times s_b \times C_{att}$. The number of channels used C_{att} as well as the convolution filter sizes k are freely chosen based on the desired complexity of the cell.

To calculate similarity between regions on Queries and Keys, patches are extracted from the Query tensor $\boldsymbol{Q}$. Let $R_{r,k_P}(\cdot)$ be an operator that extracts a neighbourhood of size $k_P \times k_P \times C_{att}$ centred around the pixel location r. Then,

$$\boldsymbol{q}_r = R_{r,k_P}(\boldsymbol{Q}) \tag{6.1}$$

is a single patch from the Query tensor centred around r. Collecting an ensemble of patches which adequately cover the feature map involves sampling r on a Cartesian grid with step size u, so that the patches extracted by R_{r,k_P} are distributed dense enough. If $k_P > u$ is chosen, this results in overlapping feature patches. Then, the patch tensor $\hat{\boldsymbol{Q}} = [\boldsymbol{q}_0, \boldsymbol{q}_1, ...\boldsymbol{q}_{F-1}]$ with $\hat{\boldsymbol{Q}} \in \mathbb{R}^{F\times k_P \times k_P \times C}$ is formed from the extracted regions. The total number of patches F can be inferred as

$$F = \left(\left\lfloor \frac{s_x - k_P}{u} \right\rfloor + 1\right) \cdot \left(\left\lfloor \frac{s_y - k_P}{u} \right\rfloor + 1\right). \tag{6.2}$$

Using each Query patch and non-expansive cross-correlation denoted by $\star$, an attention score matrix $\boldsymbol{S} \in \mathbb{R}^{s_x,s_y,F}$ is calculated from correlations with normalised patches by

$$\begin{aligned} \boldsymbol{S} &= [\boldsymbol{s}_1, \boldsymbol{s}_2, ...\boldsymbol{s}_F] \\ \boldsymbol{s}_f &= \boldsymbol{K} \star \frac{\boldsymbol{q}_f}{\|\boldsymbol{q}_f\|_2} \quad , \quad f \in \{0, 1, F-1\}. \end{aligned} \tag{6.3}$$

In practice, this operation is implemented as a non-trainable two-dimensional convolution layer, with a normalised and flipped tensor $\hat{\boldsymbol{Q}}$ inserted as the weight tensor.

A channel-wise Softmax-operation is used to transform the cross-correlation score at each location into a categorical distribution over the Query patches. At its core, the purpose of the CSA-layer is finding alignment between image regions, and since the overall structure of the input features is unchanged by the embedding via convolutional layers, this corresponds to the assumption of a roughly one-to-one alignment between Query and Key embeddings, which can efficiently be modelled by employing the Softmax on one of the components.

Afterwards, $\boldsymbol{S}$ is used as a weighting of local patches of the Value tensor. Again, a collection of local patches is extracted from the tensor, using the same extraction operator and sampling scheme as in Equation (6.1). A single patch around the pixel location r is extracted by

$$\boldsymbol{v}_r = R_{r,k_P}(\boldsymbol{V}) \tag{6.4}$$

and a tensor $\hat{\boldsymbol{V}} = [\boldsymbol{v}_1, \boldsymbol{v}_2, ...\boldsymbol{v}_F] \in \mathbb{R}^{F \times k_P \times k_P \times C_{att}}$ is formed from the patches. Permutation yields the filter kernel $\hat{\boldsymbol{V}}^{\intercal} \in \mathbb{R}^{att \times k_P \times k_P \times F}$. Again using a two-dimensional convolution layer in which $\hat{\boldsymbol{V}}^{\intercal}$ is placed as a weight matrix the output of the attention mechanism is calculated as a convolution between Softmax($\boldsymbol{S}$) and $\hat{\boldsymbol{V}}^{\intercal}$. In summary, the output $\boldsymbol{B}$ of size $s_x \times s_y \times C_{att}$ is computed via

$$\boldsymbol{B} = \mathrm{Softmax}\left(\boldsymbol{K} \star \mathrm{Norm}(\hat{\boldsymbol{Q}})\right) * \hat{\boldsymbol{V}}^{\intercal}, \tag{6.5}$$

with a channel-wise normalisation operation Norm($\cdot$). It should be noted that Equation (6.5) bears strong similarity to scaled dot-product attention.

Using a convolutional layer with a kernel size of 1 and a number of C_{in} filters, the attention output is adapted to the cell input. This way, the number of channels inside each attention cell can be kept constant, regardless of feature map size, while also scaling the output of the cell. Finally, the input to the layer is added to the output of the attention mechanism, forming the residual connection of the CSA-layer.

Due to its design, each CSA-layer incorporates $6 \cdot C_{in} \cdot C_{att} \cdot k^2 + 15$ trainable weights in its 3 learned embeddings[2], but also uses $2F \cdot k_P^2 \cdot C_{att}$ non-trainable values for the created patch filter maps. Therefore, a cell with $C_{in} = C_{att} = 16$, convolution kernel size $k = 5$, patch size and stride $k_P = 5$ and $u = 1$ and an input size of $s_x = s_y = 100$ has $\sim$ 40.000 trainable weights and $\sim$ 1.800.000 non-trainable weights. Since the non-trainable weights are calculated for each image in the mini-batch, care must be taken to monitor the resulting memory footprint. Furthermore, current framework limitations[3] negatively impact the network performance of the CSA layer.

6.4 Data

All deep learning models presented in this work were trained and evaluated on the *Automatic Cardiac Diagnosis Challenge* (ACDC) dataset. As the name implies, the dataset was originally used in a challenge setting to evaluate automatic diagnosis approaches for cardiac pathologies on real-world medical data. The dataset consists of tomographies of 150 patients, of which 120 exhibit various cardiac abnormalities

[2] Which are each composed of two CBR^2-layers, each with trainable weights in 2 two-dimensional convolutional layers and batch norm layers

[3] TensorFlow 2.0 currently does not support applying different convolution operations on each image in a mini-batch in parallel, which is essential to CSA

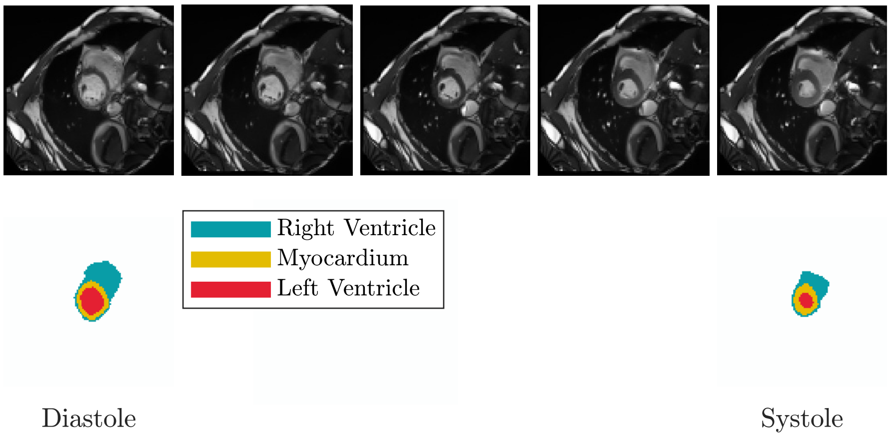


Figure 6.5: A sample slice of data contained in the ACDC dataset. Shown in the top row are 2D images taken from the 3D image volumes over the series, temporally resampled to five images. The far left image shows end diastolic volume, the far right one the end systolic volume, with the corresponding labels in the bottom row. These images are taken from a patient showing normal cardiac physiology.

such as myocardial infarctions or myopathies. The images of 100 patients are publicly available as a training set and form the dataset used in this work.

The data was acquired over a 6 year period at the University hospital of Dijon, France using two MRI scanner types (Siemens Area 1.5T and Siemens Trio Tim 3.0T, both from Siemens Medical Solutions, Germany). Breath hold gating was used to generate Cine MR images in short axis orientation using a steady-state free precession MRI sequence [100], yielding a high temporal resolution. Each data sample contains a sequence of temporally resolved volumetric scans of a real patients heart, capturing a full cardiac cycle in $28-40$ 3D images. The image volumes are positioned so that one of their dimensions aligns with the apex-base axis of the left ventricle. The images were of size 154×154 to size 428×512, resulting in spatial resolutions of around $1.4 - 1.7\frac{mm^2}{\text{px}}$.

Two discrete time steps (at systole and diastole) are equipped with semantic labels from medical experts, marking the left and right ventricle as well as the cardiac muscle tissue.

A sample from the dataset is shown in Figure 6.5. Here, a single slice of a measurement from a single patient is shown over time, and the number of time steps has been reduced to 5 by temporal downsampling. The expert labels of the two extremes (end systole and diastole) are shown as well under their respective MR images.

While the main use case of this dataset is, as its name suggests, the automated detection and classification of cardiac pathologies, the supplied high quality expert label maps serve as a good source for evaluating the quality of image registrations. As such, numerous data-driven registration algorithms [90], [101], [102] use the ACDC dataset as basis for training and evaluation of their models.

In this work, all images of the dataset were resized to a common size of 112×112 using bicubic spline interpolation and normalized to be in a range of $[0, 1]$.

6.4.1 Data Augmentation

As described in Section 3.2, the training data used to fit deep learning models has to provide a good representation of the distribution of the input space. If the dataset contains too few data points for the model to fit a good approximation to the underlying distribution, model performance will suffer. The ACDC dataset contains 743 unique image sequences, which might not be enough to train models without them overfitting to the available training data.

For a fixed model size and without the ability to add more data samples, an efficient way to circumvent data set size limitations is *data augmentation*. In data augmentation, one or several augmentation functions are used to create new training data samples from the existing training set, yielding a more robust or more general network fit to the underlying distribution. Typically, data augmentation is used solely for training data sets, while leaving the validation set as is.

With this being the case, very few limitations exist in choosing augmentation functions. However, the range of actually useful augmentations for image processing is quite slim, with most publicised computer vision approaches in deep learning using flipping, cropping and rotation augmentations.

For this specific use case, four augmentation functions are used to randomly augment the input data before each training epoch. Naturally, the same deformations are used for all images and labels in a given data point of the set. Where applicable, bi-cubic interpolation is used to form off-grid values in the images, and nearest-neighbour interpolation for the label maps.

The following augmentation functions are used for the experiments in this chapter:

- **Flipping:**

Randomly flip images and labels horizontally and vertically, enforcing mirror symmetries in the data distributions.

- **Cropping:**

Crop images and labels randomly between 80% and 100% of their original size around their centre.

- **Rotation:**

Randomly rotate images and labels for $0°, 90°, 180°$ or $270°$.

- **Random Deformation:**

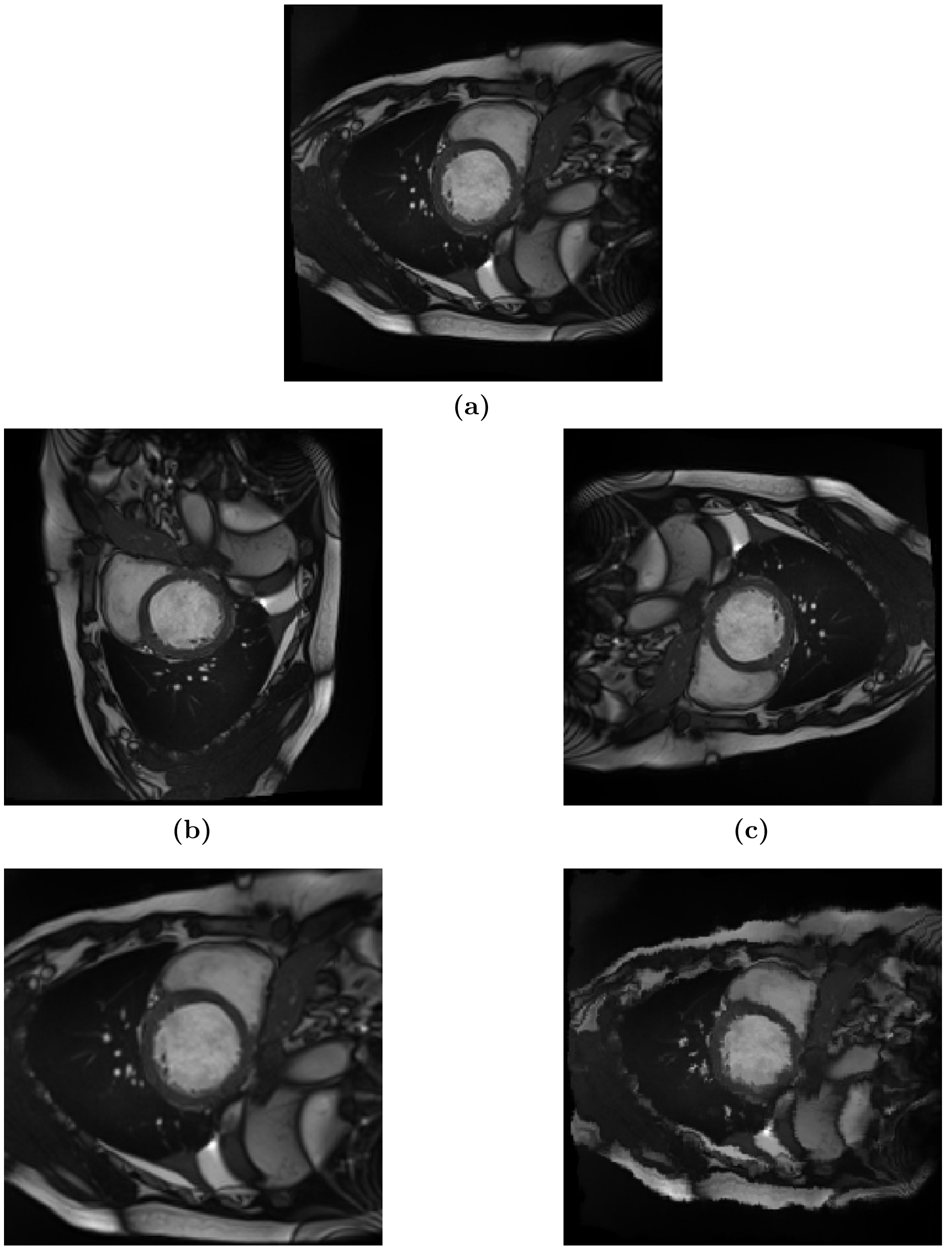
Apply a slight random deformation to images, which prevents convolutional layers from overfitting and provides network regularisation [103]. The deformation in all dimensions consists of strongly low-pass filtered random noise, so that an augmented image $\tilde{\boldsymbol{I}}$ can be produced from a base image $\boldsymbol{I}$ by

$$\tilde{\boldsymbol{I}} = \boldsymbol{I}(\boldsymbol{x} + \boldsymbol{g}(\boldsymbol{x})) \tag{6.6}$$

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix} f(\alpha_a \cdot \boldsymbol{a}) \\ f(\alpha_a \cdot \boldsymbol{b}) \end{bmatrix}, \tag{6.7}$$

where $\boldsymbol{a}, \boldsymbol{b}$ are random noise signals bounded in $(-1, 1)$. The low-pass filter operator f is implemented as a convolution with a $2D$ unit Gaussian function with a standard deviation of σ_a in both dimensions. The amplitude α_a and the standard deviation of the Gaussian bell σ_a control the strength and 'sharpness' of the deformation and have respectively been set to 10 and 2 in all experiments.

Figure 6.6 shows the 4 augmentations for a single example image from the dataset used in the experiments. Since the effect of the Gaussian deformation augmentation is not strong enough to be visible in graphics of this size, α_a was chosen as 30 for this demonstration.



(a) (b) (c) (d) (e)

Figure 6.6: The augmentations used for deep learning, applied to the single ACDC dataset image shown in (6.6a). Rotation (of 90°) is shown in (6.6b), flipping in (6.6c), cropping in (6.6d), and random deformation in (6.6e). The strength of the deformation shown here is 3 times stronger than in the actual implementation for visualisation purposes.

6.5 Loss and Quality Functions for Machine Learning

All machine learning approaches can, at their core, be categorized as optimisation problems. An integral part of the design and evaluation of such problems are suitable *loss functions* which map an approximation provided by the chosen algorithm to a real number, representing the distance of the output to a desired solution. Solving the optimisation problem then involves adapting the approximation produced by the network to minimise the chosen loss function.

As mentioned in Section 3.2, neural networks are trained to minimize convex, differentiable loss functions, which define the learned output distribution.

As this thesis is concerned with a subset of image registration tasks, the networks designed in this section will be tasked with aligning images. As such, the loss function used in these designs will have to be able to measure differences between images.

To accomplish this, this work generally uses a weighted sum of differentiable image quality metrics. Suitable image metrics are the ℓ_2-*loss*, which captures the pixel-wise error between images, and the SSIM, which aims to quantify the subjective difference a human observer might experience. Both errors are differentiable and were already defined in Section 5.4.

Since the goal is to minimize an objective function and have (in the best-case scenario) equalness between reference and registered image, the Structural *Dis*similarity Measure $\mathrm{DSSIM}(\mathbf{a}, \mathbf{b}) = 1 - \mathrm{SSIM}(\mathbf{a}, \mathbf{b})$ is used in the overall loss function.

It has been shown that training a neural network solely on the SSIM is not ideal, as noise or misalignment around image edges as well as uniform offsets in flat, detail-sparse image areas are not properly captured by the metric. While enhancements to the SSIM exist and alleviate some of its issues, pairing it with other loss functions empirically produces better results [104]. In this work, the base SSIM (or DSSIM) is used in conjunction with a more conventional ℓ_2-loss of the difference between the two images.

Another measure which aims to capture image quality as it appears to a human observer is the *Universal Image Quality Index (UIQI)* [105], which is a predecessor to the SSIM measure and similarly defined as

$$\mathrm{UIQI}\left(\mathbf{a}, \mathbf{b}\right) = \frac{2\sigma_{\mathbf{ab}}}{\sigma_{\mathbf{a}} + \sigma_{\mathbf{b}}} \cdot \frac{2\sigma_{\mathbf{a}}\sigma_{\mathbf{b}}}{\sigma_{\mathbf{a}}^2 + \sigma_{\mathbf{b}}^2} \cdot \frac{2\bar{a}\bar{b}}{\bar{a}^2 + \bar{b}^2}.$$

As in earlier equations, $\bar{a}$ denotes the mean value of $\boldsymbol{a}$, and $\sigma_{\mathbf{a}}^2$ its variance.

In contrast to the SSIM, the UIQI is regarded as unstable. This shortcoming was solved by the addition of stabilizing factors in the SSIM.

The ACDC dataset (as introduced in Section 6.4) features not only images, but expert labels for two time steps in the tomography on which the quality of the approximated registration can be evaluated as well. In the experiments in Section 6.6 and Section 6.7, the deformation approximated on the input images is evaluated on these labels by quantifying the overlap between two label maps - the reference map and the label map deformed by the approximated field. For this, the Sørensen–Dice coefficient (the *Dice score*) can be applied. For two sets of discrete data X and Y, the Dice score is defined as

$$\mathrm{Dice}(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|} \tag{6.8}$$

where $|\cdot|$ denotes the cardinality of each set. The score calculates the ratio between two times the data shared between the sets and the total number of entries in the sets. Hence, the coefficient equals 0 for disjoint sets and is 1 if $X = Y$.

The Dice coefficient (or one of its close relatives) are commonly employed to measure the performance of classification or segmentation algorithms. Similar to classification tasks, the Dice score is then used to compare the pixel-wise label- or one-hot encoded categorical maps. Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two such one-hot label encoded maps. Then the overlap is calculated via the Dice score by

$$\mathrm{Dice}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\mathrm{vec}(\boldsymbol{X})^\intercal \mathrm{vec}(\boldsymbol{Y})}{|\boldsymbol{X}|}, \tag{6.9}$$

where $\mathrm{vec}(\cdot)$ denotes vectorisation of the two maps. In the later chapters, the use of one-hot label maps is implicit, as the conversion is trivial.

6.6 Deep-Learning based Image Registration

This chapter will discuss the image registration approach published in the research paper titled "Convolutive Attention for Image Registration" [2] at the EUSIPCO 2022. The method presented in this section represents one of the major contributions of this work.

While the algorithm in Section 5.4 is able to robustly register affine motion, it is structurally unable to register elastic motion, which can only adequately be described by directly estimating the entire dense vector field. While elastic motion registration algorithms do exist (like the popular demons algorithm [106] or more recent developments estimating diffeomorphic deformations [107], [108]), they are often very sensitive to image noise, and might be unable to handle the image artifacts present in MRI scans.

In this chapter, an image registration algorithm is presented which aims to produce a more robust elastic registration for medical images using a deep learning architecture. This model incorporates the CSA-layer introduced in Section 6.3 to model long-distance relationships in the images.

This work was developed as sort of a stepping stone towards registration of images stemming from partial MR scans, and as a proof-of-concept for the convolutional attention layer in a registration setting. Thus, no MRI encoding is used in this approach and the network is trained and evaluated on artifact-free images.

Like most common image registration algorithms, the presented model accepts image pairs as input - a reference image $\boldsymbol{I}_{ref}$ and a deformed version $\boldsymbol{I}_{mov}$ - and returns an deformation field $\hat{\boldsymbol{y}}$ as a registration result, defining the registered image as $\boldsymbol{I}_{\boldsymbol{y}}$. In a best-case scenario, $\boldsymbol{I}_{ref} = \boldsymbol{I}_{mov}(\boldsymbol{x} + \hat{\boldsymbol{y}})$ holds, using an appropriate function to interpolate $\boldsymbol{I}_{mov}$ on off-grid positions. In the following sections, the short hand notation $\boldsymbol{I}_{\boldsymbol{y}} = \boldsymbol{I}_{mov}(\boldsymbol{x}+\hat{\boldsymbol{y}})$ is used to denote the moved image post registration.

6.6.1 Architecture

In this model, a U-Net architecture is chosen as the base structure. As seen in literature, the U-Net as an architecture is very capable of solving pixel-wise regression tasks. The network structure was originally developed and then extensively used for image segmentation and is also used in comparable tasks [90].

The architecture is schematically shown in Figure 6.7 and again makes extended use of the CBR^2-layer used in the CSA-cell and pictured in Figure 6.4, with the other building blocks being discussed in Section 3.5. This replaces the dual convolution layer structure of the U-Net as proposed by Ronneberger.

Following the design philosophy of U-Net, the proposed structure is split into an encoding part, which extracts feature maps at different resolutions, and a decoding part combining all extracted features. The input of each stage of the encoder is processed using a CBR^2-layer. The stages output is then subsampled

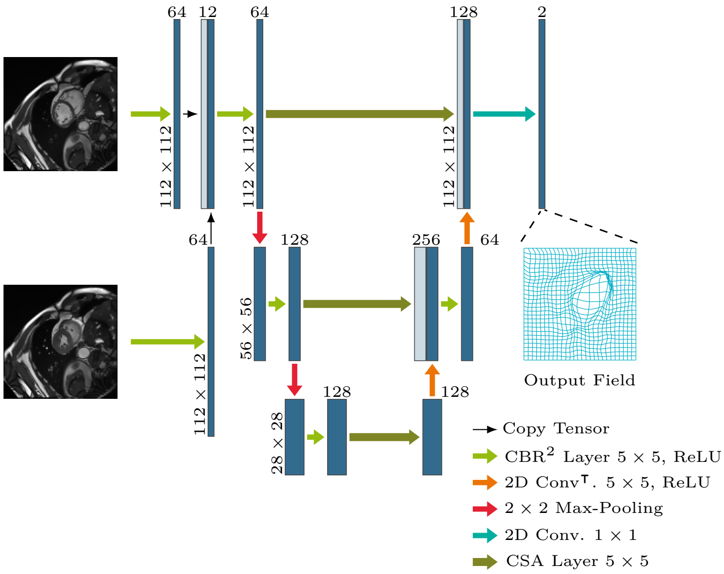


Figure 6.7: The Architecture used in registration experiments.

using 2×2 max-pooling, halving its spatial size. This subsampling then serves as the input to the next stage.

In the classical U-Net, the embeddings of each stage are then upsampled using a transposed convolution and concatenated with a *skip connection* - a shortcut to the corresponding stage in the encoder part before again being embedded using a convolutional layer.

In the proposed architecture, each skip connection is replaced with the CSA-layer detailed in Section 6.3. This allows the attention layers to process the image features in multiple resolutions. Each layer uses an internal number of attention channels $C_{att} = 16$, but, due to the makeup of the cell, the number of channels is identical in input and output of a CSA-layer.

The images which are processed in the network are first embedded using separate CBR^2-layers and concatenated before being processed in the multistage architecture. The output of the network is embedded using a $2D$ convolutional layer (without activation) to form the output - a $s_x \times s_y \times 2$ deformation field $\hat{\boldsymbol{y}}$ used to align the images and create $\boldsymbol{I}_y$ from $\boldsymbol{I}_{mov}$.

This architecture yields a network with approximately 3.000.000 trainable weights and is labelled as $\mathrm{U}_{\mathrm{CSA}}$ in the results.

6.6.2 Data Format

For this work, data from the ACDC dataset described in Section 6.4 was used to fit the network. While not being used directly to train the network, the semantic segmentation maps contained in the dataset were also used to evaluate registration quality metrics.

In the dataset, the expert label maps are only provided for the end systolic and end diastolic volumes. Hence, the volumes corresponding to these states are used to optimise the weights of the registration network. Each patients' volumetric image data is sliced along the apex-base axis into two-dimensional images, forming 743 image tuples $(\boldsymbol{I}_{ref}, \boldsymbol{I}_{mov})$. Each image is then resized using bicubic interpolation to a common size of 112×112. Afterwards, each tuple is rescaled so each pixel value is in the interval $[0, 1]$. The effect of additional whitening techniques was evaluated in preliminary experiments and found to be of little consequence for the networks' performance.

Similarly, the categorical label volumes of end systole and end diastole of each patient were sliced into two-dimensional images and resized to a size of 112×112, forming the label tuples $(\boldsymbol{L}_{ref}, \boldsymbol{L}_{mov})$ belonging to each image tuple. Nearest neighbour interpolation is used in the rescaling, as the integer value of each pixel in the maps corresponds to its local label.

6.6.3 Composite Loss Function and Performance Metrics

With a dataset $\mathcal{D}$ containing image tuples of reference and deformed images $(\boldsymbol{I}_{ref}, \boldsymbol{I}_{mov})$, the network is trained to minimise both a function comparing the reference image to the registered moving image, as well as a regularisation function $\mathcal{R}(\hat{\boldsymbol{y}})$ which penalises unfavourable properties of the output field. The loss function should capture both subjective as well as objective differences between the reference and the registered image. Therefore, a weighted mixture of an ℓ_2-loss and an SSIM-loss, as defined in Equation (5.14) and Equation (5.15), was chosen to evaluate the quality and accuracy of the deformation.

Medical tomography images (and with that the images from the in-vivo experiments and the data set) tend to have a region of 'activity' containing the structures of interest and large uniform patches of low intensity outside of it. Allowing the network to produce an arbitrary registration would not affect the objective registration result as measured by the above loss functions, but would be detrimental to the overall network output quality, as it would also allow the network to output basically arbitrary large deformations from little to no information. To avoid this, abrupt changes in the network output are penalised to force the vector field to be as smooth as possible. The field regulariser $R(\hat{\boldsymbol{y}})$ is then defined as the sum of squared spatial differences at every pixel using finite difference operators $\nabla_1,\ \nabla_2$ for both spatial dimensions:

$$R(\hat{\boldsymbol{y}}) = \frac{1}{2}\sum_{s=1}^{s_p} (\nabla_1 \hat{\boldsymbol{y}}_s)^2 + (\nabla_2 \hat{\boldsymbol{y}}_s)^2 \tag{6.10}$$

where again $s_p = s_x \cdot s_y$ is the total number of pixels in each image and $\hat{\boldsymbol{y}}_s$ is the deformation vector at position s.

These three functions make up the components of the component loss used for this research. For reference image $\boldsymbol{I}_{ref}$ and registered image $\boldsymbol{I}_{\boldsymbol{y}}$, the loss function has the form

$$\begin{aligned} \mathrm{E}(\boldsymbol{I}_{ref}, \boldsymbol{I}_{\boldsymbol{y}}, \hat{\boldsymbol{y}}) =& \lambda_S(1 - \mathrm{SSIM}(\boldsymbol{I}_{ref}, \boldsymbol{I}_{\boldsymbol{y}})) + \lambda_L \ell_2(\boldsymbol{I}_{ref}, \boldsymbol{I}_{\boldsymbol{y}})) \\ &+ \lambda_R R(\hat{\boldsymbol{y}}), \end{aligned} \tag{6.11}$$

where $\lambda_S,\ \lambda_L,\ \lambda_R$ are positive scalar weightings.

In addition to the loss function, several quality metrics are evaluated on the validation dataset as well. While these do not contribute to the training process, their values give some insight into the networks performance.

While the number of available image quality metrics is limited and the two most common metrics are already evaluated as part of the loss function, the Universal Image Quality Index (UIQI) [105] was additionally used as a metric, capturing both image similarity and quality of the registered image.

Even though they are not used in the networks loss function, the label maps contained in the ACDC dataset allow for the definition of an additional metric. Deforming the label map of the moved image $\boldsymbol{L}_{mov}$ with the deformation field approximated using $\boldsymbol{I}_{ref}$ and $\boldsymbol{I}_{mov}$ would - in a best-case scenario - yield the label map of the reference image $\boldsymbol{L}_{ref}$. Thus, the quality of the deformation generated from the images can be assessed by calculating the overlap between the reference and the (registered) moved label map. As outlined in Section 6.5, the Dice coefficient is a natural fit and allows to define the Dice metric as

$$\text{Dice}(\boldsymbol{L}_{mov}(\boldsymbol{x} + \hat{\boldsymbol{y}}),\ \boldsymbol{L}_{ref}). \tag{6.12}$$

As the presented architecture does not limit the registration fields to Lie groups, the influence of the regularisation on the field invertibility is analysed in the results. The hypothesis is that the regularisation sufficiently discourages outputting useless deformation in image areas of similar intensity, so that very few if any field folds occur. This registration approach does not explicitly depend on its fields being invertible, but invertibility is a good indicator for the feasibility of the learned mapping. Thus, a metric aiming to quantify the amount of folding that occurs during deformation, which is used as a proxy distance to exact invertibility, is evaluated alongside the image metrics.

To define the invertibility metric, the determinant of the Jacobian of the output deformation fields as presented in Equation (4.37) is leveraged. Evaluated for the i-th element, $\det(J(\hat{\boldsymbol{y}}_i))$ indicates its change in area when deforming with $\boldsymbol{y}$. Crucially, the measure is negative for elements which change orientation during deformation, causing occlusion of the affected element.

The use of Iverson bracket notation then allows the definition of a metric which counts occurrences of elements for which the determinant of the Jacobin turns negative. The resulting measure is defined for an output deformation field $\hat{\boldsymbol{y}}$ as

$$\delta_{J(\hat{\boldsymbol{y}})} = \frac{1}{s_p} \sum_{s=0}^{s_p-1} [\det\left(J(\hat{\boldsymbol{y}}_s)\right) \ < \ 0], \tag{6.13}$$

where spatial derivatives are approximated with a forward difference across each spatial dimension. Since the metric is weighted by the number of pixels of the affected image, its range is the interval $[0, 1]$, with 0 denoting no occurrence of field folding.

6.6.4 Baseline Systems

Furthermore, a re-implementation of the approach published by Hering et al. [90] for the same problem is evaluated. Their approach incorporates the region labels supplied by the ACDC dataset by minimising the ℓ_2 distance of $\boldsymbol{L}_{ref}(\boldsymbol{x})$ and $\boldsymbol{L}_{mov}(\boldsymbol{y})$ for an estimated dense registration field $\boldsymbol{y}$ during training. This approach is labelled $\mathrm{U_H}$ in the results.

The results of the two methods are not directly comparable, since labels are not used for training in the approach detailed in this chapter. As an alternative, a slight modification of the approach of Hering et al. was evaluated as well. Substituting their loss function with the one presented in Section 6.6.3 allows for a more direct comparison between their approach and the one introduced in Section 6.6.1 as it removes the dependency on the label maps during training. This modified architecture is labelled $\mathrm{U}_{\mathrm{mod}}$ in the results.

It should be noted that the images and labels under consideration exhibit rather strong self-similarity, with $\boldsymbol{I}_{mov}$ being close to $\boldsymbol{I}_{ref}$ even without registration. As such, the unregistered case where $\boldsymbol{y} = \boldsymbol{x}$ for each image is evaluated as well. While this is not a registration algorithm, it allows gauging the effect of the other approaches toward image alignment.

6.6.5 Training

Optimisation was carried out using MGD with a mini-batch size of 10, the Adam optimiser and a learning rate of 1e−5.

For the loss function, preliminary experiments found values of $\lambda_S = 1\text{e}6$, $\lambda_L = 5\text{e}2$, and $\lambda_R = 150$ to yield the overall best registration results[4]. The weighted error function is used to train U_{CSA} and U_{mod}, while for the baseline network U_H discussed in Section 6.6.4, their originally proposed loss function - including their proposed weighting - was used.

As a training strategy, k-fold cross validation with $k = 10$ was chosen for all experiments - the presented model and deep learning baselines. As mentioned in Section 6.4, the data contained in the ACDC set is very self-similar in each subset stemming from a specific patient. Therefore, assigning the data of a patient partially to the training and validation set would pollute the results by allowing the network access to validation data during training. Using a pseudo-random split, the data from each patient was assigned to the training or validation set for each step of the cross validation.

[4]While it is common practice to set the value of a single weighting to 1 and adapt both training rate and the other Lagrange weights, this was neglected during the experiments. Restating the loss function by rescaling the weights and learn rate so this convention is upheld is not permissible without repeating the experiments, as the moments-based optimiser used here is influenced by the actual unscaled error value

All networks were trained for 700 epochs. Convergence was typically reached much sooner, but as there was no sign of overfitting during training, further training is not seen as problematic. In each training epoch, each sample of the training set was augmented using all image augmentation methods presented in Section 6.4.1 and shown to the network once. Similarly, the label maps were augmented using the exact same augmentation, again using nearest-neighbour interpolation for the random deformation.

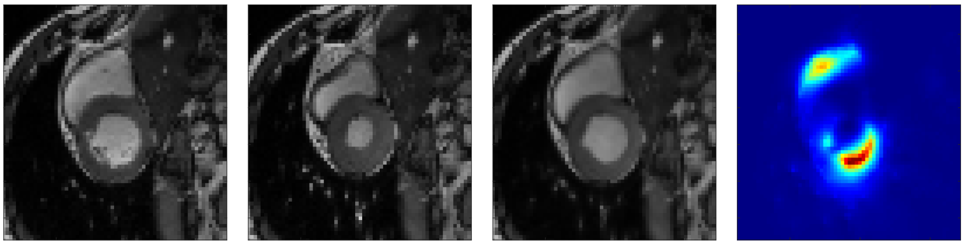
The network was implemented in TensorFlow 2.4.1 and trained on a single NVIDIA Geforce RTX 2080 GPU, on which inference of a single vector field from a pair of input images takes about 0.15 seconds.

6.6.6 Results

After training the deep learning models in each of the k cross validation runs, the presented registration metrics were evaluated for each model on its validation data points. Since the union of all validation sets encompasses all data points, combining these evaluations from all k models yields conclusive results for the whole dataset. These combined metrics for $\mathrm{U}_{\mathrm{CSA}}$ and its comparisons are shown in Figure 6.9. In the figure, the values of the four models and the unregistered case are presented as bar graphs, showing their median and spread.

As is visible in the results, $\mathrm{U}_{\mathrm{CSA}}$ offers a Dice coefficient and UIQI performance similar to the diffeomorphic demon algorithm and yields better results for SSIM than its competitors. Furthermore, the median ℓ_2-difference between images is greatly improved when compared to the other approaches. The network U_H used for comparison excels when comparing the Dice coefficient of the label maps post registration. These results are not unexpected - as laid out in the previous chapters, the comparison network U_H explicitly optimises for label difference minimization, naturally resulting in better Dice coefficients. The approach presented here, on the other hand, explicitly optimises for maximization of the SSIM.

However, when comparing metrics between $\mathrm{U}_{\mathrm{mod}}$ and $\mathrm{U}_{\mathrm{CSA}}$, it becomes apparent that the presented architecture achieves significantly better results for all evaluated metrics. This might indicate that the incorporated loss function is better suited for the presented network, especially as no further adaptation of the hyper-parameters of the loss function and training was carried out for $\mathrm{U}_{\mathrm{mod}}$. It also strongly suggests



a) $\boldsymbol{I}_{ref}$ b) $\boldsymbol{I}_{mov}$ c) $\boldsymbol{I}_{\boldsymbol{y}}$ d) $\|\hat{\boldsymbol{y}}\|^2$

Figure 6.8: A sample of the registration for an image tuple from the validation dataset. Here, a reference image (a) and the corresponding deformed image before (b) and after (c) deformation are shown. Also shown is the intensity of deformation in every pixel in the shown image region as the sum of squares of its components. The images are slightly cropped to only show the cardiac region.

Table 6.1: Mean occurrence of field foldings per pixel ($\delta_{J(\hat{\boldsymbol{y}})}$) and registration quality metrics reached with different regularisation strengths λ_R.

λ_R	$\delta_{J(\hat{\boldsymbol{y}})}$	Dice	UIQI	SSIM
0.15	3.3e−2 (3e−4)	0.745	0.897	0.986
1.5	1.0e−2 (5e−5)	0.796	0.899	0.988
15	1.9e−3 (4e−6)	0.830	0.892	0.985
150	3.2e−6 (8e−10)	0.803	0.866	0.972
1500	0 (0)	0.678	0.842	0.957

that the novel convolutional attention cell improves the results offered by deep image registration.

A single image pair from a validation set including the deformation field approximated by the network is shown in Figure 6.8. Anatomical structures are subjectively well registered in I_y, while the registration was not visibly perturbed by image noise.

An interesting detail concerns the bright spots surrounding the myocardium in $\boldsymbol{I}_{mov}$, which are absent after registration. Analysis of the deformation field in these regions shows that the network learned to 'spread out' these spots over a larger area, leading to better SSIM in the region. This indicates that the presented approach is better suited to handle images which break the equal brightness constraint crucial to optical flow algorithms.

As mentioned, the output of this network is not restricted to diffeomorphic fields. However, an assumption during the design of the loss function was that the spatial regulariser penalising the gradient of the deformation is sufficient to limit

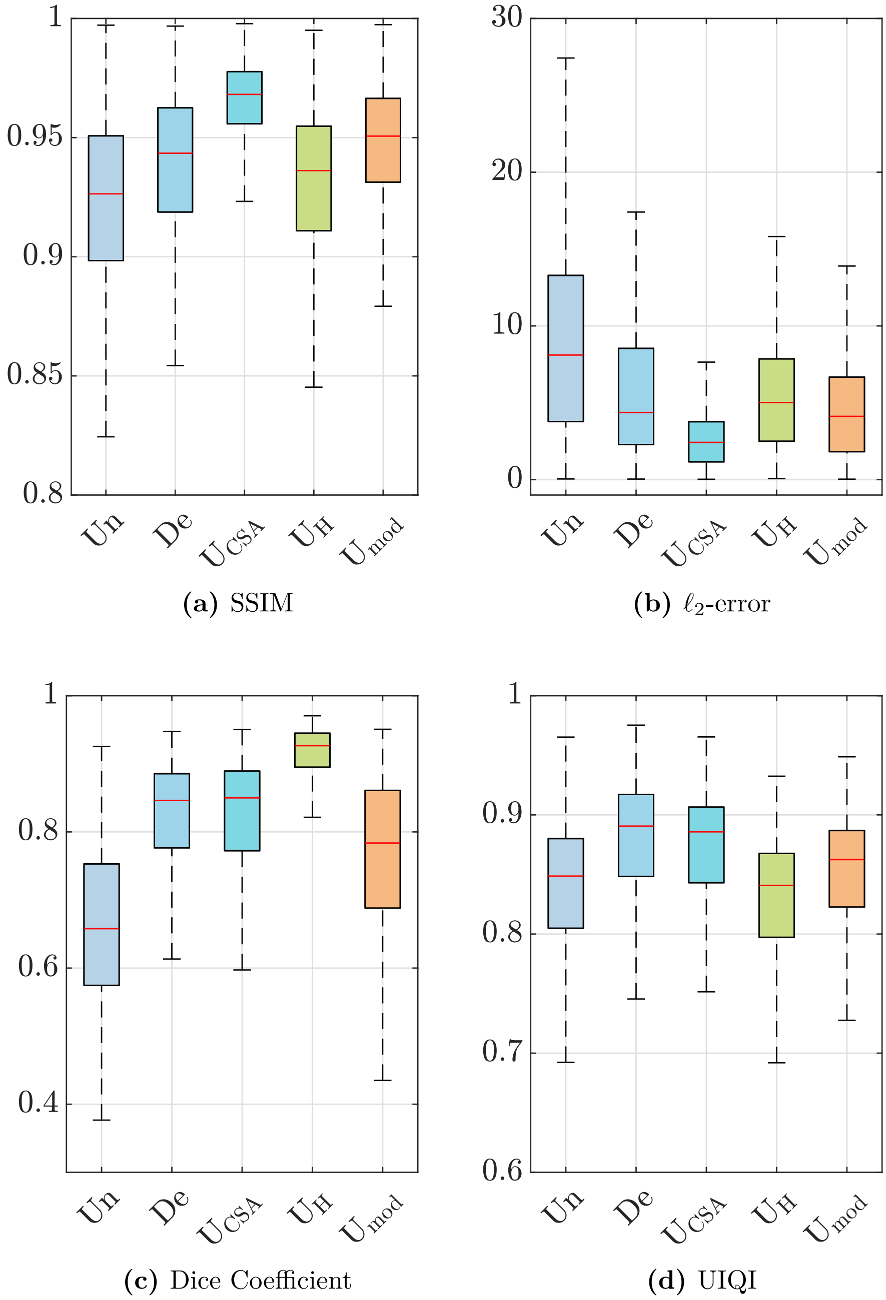


Figure 6.9: Results for evaluation metrics for the unregistered case (Un), Demons registration (De), the $\mathrm{U}_{\mathrm{CSA}}$ network and both U-Nets used for comparison. Median is shown as red line, the box marks 25th and 75th percentile, whiskers mark extremes of range. The metrics are evaluated on the union of the cross-validation validation sets, so that they represent the whole dataset.

occurrences of field folding. Varying the value of the weighting term λ_R allows to analyse the influence of the regularisation on field folding.

The results of this analysis are collected in Table 6.1. Here, the mean values of $\delta_{J(\hat{\boldsymbol{y}})}$ from all deformation fields approximated for the validation datasets is shown for 5 different strengths of λ_R. Furthermore, the registration metrics for these images are shown as well to allow comparison of alignment quality. All other weighting parameters were kept as stated in Section 6.6.3.

As expected, a decrease in λ_R allows the network to more freely capture complex motions, resulting in an increase in image quality (measured with SSIM and UIQI) as well as Dice coefficient. However, this also causes an increase in occurrences of vector field folding as a trade-off. With minimal regularisation strength, a decrease in Dice coefficient score can be observed, which is caused by heavy deformations as a response to image noise and areas of little activity or self-similarity. However, of note is that this did not cause a subjective degradation in image quality. Values of λ_R smaller than 0.15 were tested as well, but often resulted in divergence during network optimisation.

Even for small values of λ_R, the occurrence of folding all but disappears. With the chosen regularisation of $\lambda_R = 150$, 31 occurrences of folding across the whole data set are measured. While not being explicitly diffeomorphic, this is seen as acceptable.

Varying both other weightings of the loss function - namely λ_S and λ_D - by an order of magnitude did not significantly affect the registration outcome. As can be expected however, the convergence rate was affected by the scaling, though the network did not diverge in any setting.

6.7 Sequence-Based Image Reconstruction

As mentioned in the introduction to the previous chapter, the approach presented in Section 6.6 is designed to tackle a classical registration problem. As outlined in the rest of this thesis, MRI registration is more complicated, as the full image data is unavailable when approximating motion. Ideally, a deep network would still be able to infer dense vector fields for image alignment directly on the partial scan data, without the need to iteratively reconstruct an image. Since the perturbations caused by the measurement process are always the same for a given scan setting, the network should be able to discern the relevant structures in the underlying data. The actual reconstruction of the image can then be achieved using numeric methods using the networks approximation of the inherent movement. To the best of the authors knowledge, using neural networks to produce registration fields for registering partial MRI data is a novel approach to autofocusing.

This chapter contains the architecture and results for an unpublished extension to the approach presented in Section 6.6 to tackle this specific task. The method presented in this section represents one of the major contributions of this work.

The deep learning approach presented in this section aims to produce not a single dense registration, but a sequence of deformation fields which aim to align N MRI measurements to a common reference. The alignment problem posed is similar to the one in Section 6.6, but now involves multiple images and registrations, which are created from the partial measurements of a simulated MRI scan.

In contrast to other MRI specific machine learning approaches (for example, in [109]), the network is therefore not tasked with directly recovering a sharp image. Instead, the convex image recovery from Section 4.3 is used together with the approximated vector fields to reconstruct an image after the one-shot registration of all images in the sequence.

As described in Section 5.3, image sequences $\boldsymbol{I} = [\boldsymbol{I}^{(1)}, \boldsymbol{I}^{(2)}, \ldots, \boldsymbol{I}^{(N)}]$ are considered as the underlying data of an N-shot MRI measurement.

For the task of multi-image registration, a single image in the base image sequence is designated as a reference image. In the following setups, the reference image is chosen as the image in the center of the sequence, at the index $m = \lceil \frac{N}{2} \rceil$. Then, N registrations $\hat{\boldsymbol{y}}^{(n)}$ with $0 < n \leq N$ are inferred by the network based on the partial shot data, so that for each image $\boldsymbol{I}^{(n)}$ in the sequence, the best-case scenario is $\boldsymbol{I}^{(m)} = \boldsymbol{I}^{(n)}(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(n)})$. Again, linear interpolation accounts for off-grid positions. Since the target image is part of the input sequence, one of the images $\boldsymbol{I}^{(n)}$ is aligned with itself.

6.7.1 MRI and Data Model

While there are a few machine learning approaches which use complex input data, to the best of the authors knowledge, all established image processing networks use real data. While processing complex-valued image data poses an interesting problem on its own, here, an MRI simulation was used which produces real-valued partial images. This way, established neural network building blocks can be used without special accommodation for complex data.

As stated previously, the models presented in this section learn to align image sequences of N images, one for each partial MRI measurement. As in the experiments in the previous chapter, the base images used to generate the input sequences are taken from the ACDC dataset. In contrast to the earlier experiments however, the entire time-resolved data of each patient is used.

For every patient measurement available in the ACDC dataset, the labelled end diastolic and end systolic volumes are considered the start- and end-point of the sequence. The volumes are once again sliced along the apex-base axis and resized to a size of 112×112 pixels. The images along a single slice position are temporally resampled using a linear $1D$-interpolator to yield the required sequence of N two-dimensional images.

The extensive data augmentation routines described in Section 6.6.2 are used with the exact same augmentation applied to every image in a sequence.

After augmentation, each image of the sequence is transformed into frequency space, subsampled to simulate a shot of the PROPELLER trajectory and transformed back into image space, so that for the n-th image $\boldsymbol{I}^{(n)}$ of a sequence, a frequency subsampled image $\hat{\boldsymbol{I}}^{(n)}$ is formed by

$$\hat{\boldsymbol{I}}^{(n)} = \left(\boldsymbol{A}^{(n)}\right)^{\dagger} \boldsymbol{A}^{(n)} \boldsymbol{I}^{(n)}, \tag{6.14}$$

where the adjoint of the NFFT (denoted by $\dagger$) is used in place of the inverse of each system matrix $\boldsymbol{A}^{(n)}$.

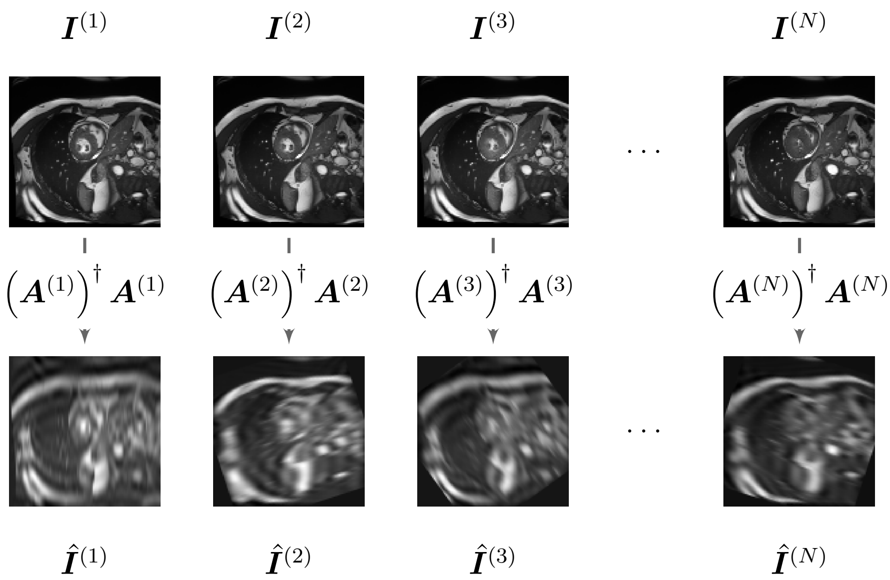


Figure 6.10: The encoding used to transform an image series of N images before they are input into the alignment network. Each image holds information from (partially overlapping) frequency subspaces after the transform. In the shown sequence, $N = 10$.

A sample encoding of an image sequence is shown in Figure 6.10. Both the encoding of the image into frequency space as well as the decoding of the subsampling into image space is implemented via NFFT. Note that this only pertains to the *network input* - the loss function is minimised using the untransformed images.

Using this procedure, the input to the networks is a sequence of images highly affected by the directional blurring and Gibbs ringing imposed by partial PROPELLER sampling.

To test the capability of the approach to handle image registration in the presence of these perturbations, multiple values of N are evaluated in the experiments. As the number of measurements directly influences the number of lines in a PROPELLER blade, the introduced perturbation per image is stronger for higher values of N.

As an additional baseline, sequences of $N = 5$ are evaluated without MRI encoding using the untransformed images.

The presented procedure allows the expert labels of the end systolic and end diastolic phases to be used as well. Slicing the relevant label volumes yields label maps belonging to the first and N-th image in each sequence which will be omitted during training and used only to verify the registration results.

Since an arbitrary image from the sequence is chosen as a reference, the nomenclature for label maps from Section 6.6.2 does no longer apply. In the context of this chapter, the label maps are named according to their position so that the end diastolic and end systolic label maps are denoted $\boldsymbol{L}^{(1)}$ and $\boldsymbol{L}^{(N)}$.

6.7.2 Architecture

Building off the results in Section 6.6, the base model architecture is chosen very similarly as a U-Net incorporating the CSA-layer in different configurations. Again, the architecture makes extensive use of the CBR^2-layer as its basic building block in the encoding and decoding paths.

All N input images are preprocessed using separate CBR^2-layers. The resulting feature maps are concatenated along their channel dimension and processed together. Again, the basic strategy of U-Net which incorporates 2×2 max-pooling in the encoding path and transposed convolution with stride size 2 in the decoding path is used.

After the decoding of the U-Net-like structure is post-processed by some further CBR^2-layers, it is finally embedded using a $2D$ convolutional layer without activation function. The output fields are inferred as a $s_x \times s_y \times 2 \cdot N$ tensor, which is interpreted as a concatenation of N 2-dimensional *velocity fields* corresponding to the deformation fields aligning the images in the series. As discussed in Section 4.5, matrix exponentiation is used to transform the output of the network into deformation fields. This is implemented using the scaling and squaring method. Since the linear interpolation used in the scaling and squaring implementation is differentiable, the matrix exponentiation does not obstruct the backpropagation algorithm used in the optimisation of the networks weights.

This change to the interpretation of the models output forces the overall architecture to learn and produce diffeomorphic registration fields, which are guaranteed to be invertible. After scaling and squaring, the network outputs a series of deformation fields $\hat{\boldsymbol{y}} = [\hat{\boldsymbol{y}}^{(1)}, \hat{\boldsymbol{y}}^{(2)}, \dots, \hat{\boldsymbol{y}}^{(N)}]$ for an input of a frequency subsampled image series $\hat{\boldsymbol{I}} = [\hat{\boldsymbol{I}}^{(1)}, \hat{\boldsymbol{I}}^{(2)}, \dots, \hat{\boldsymbol{I}}^{(N)}]$.

Since the CSA-layer is a novel building block of convolutional networks, an area of interest in this research is the optimal placement of the cell inside the network. An assumption in the experiments in Section 6.6 was that the CSA-cell is optimally placed inside the skip connection of the U-Net, but it is also possible that this is not yet ideal. Either applying the attention mechanism earlier, before the transfer

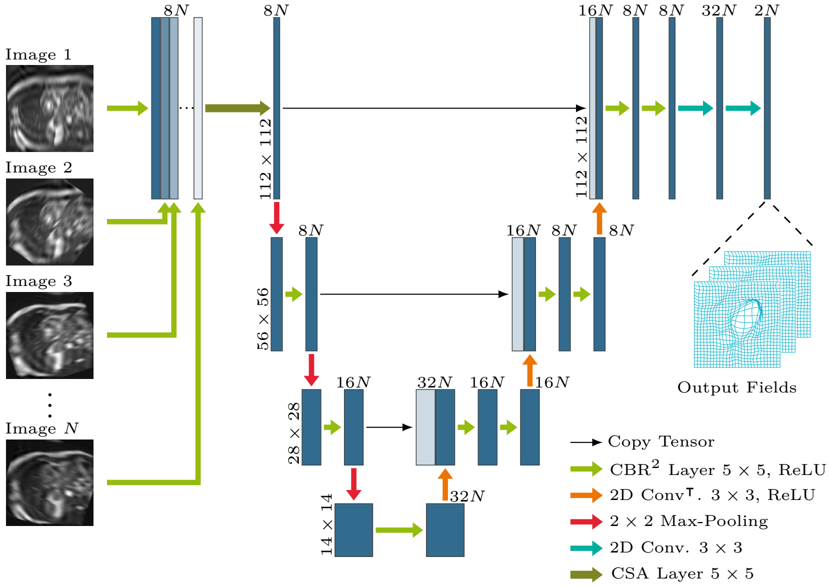


Figure 6.11: The U_B architecture used in registration experiments. Here, the attention cell is positioned directly after the concatenation of embedded input images and before the U-Net.

of the image data into flow fields, or later, after decoding the latent space of the U-Net, might yield better results for the task at hand.

To obtain insight into optimal placement of the convolutional attention, four different architectures are evaluated in the experiments, each with different placements of the attention cells. In the first experiment, the cells are placed in the preprocessing paths of the images, directly before the resulting feature maps are en- and decoded by the U-Net. Due to this placement, the network is named U_B. This architecture is shown in Figure 6.11.

The second architecture is shown in Figure 6.12. Here, the attention cells are placed in the skip connection, mimicking the architecture used in the registration experiments in Section 6.6. Since the attention cells are positioned in the middle of the U-Net, this architecture is dubbed U_M.

The third architecture places the attention cells in the post-processing path, after the latent space of the U-Net is decoded again. The U_A architecture is shown in Figure 6.13.

Finally, as comparison, a network structure without the attention cell is used which is called U_C in the results. The resulting architecture is pictured in Figure 6.14, and yields an alignment approach using a more conventional U-Net structure.

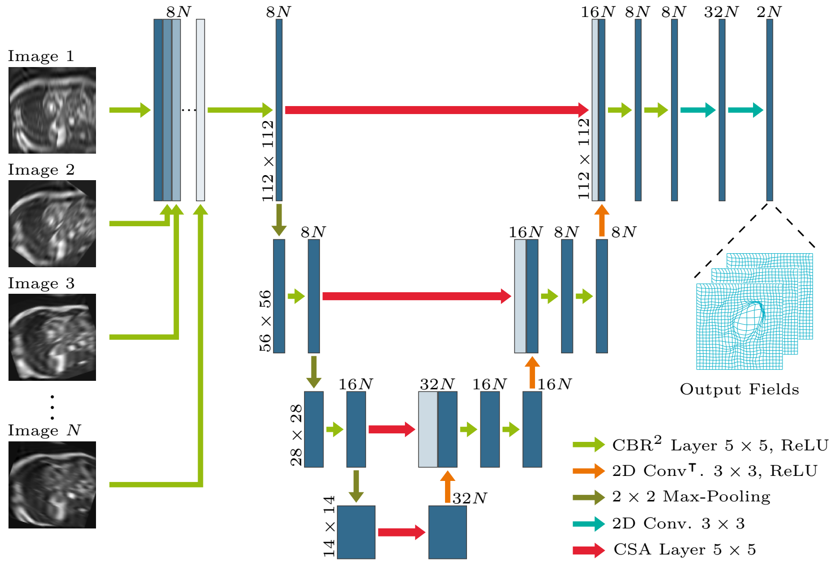


Figure 6.12: The $\mathrm{U_M}$ architecture used in registration experiments. As in the registration network developed for Section 6.6, CSA-layers are inserted in the middle of the U-Net into its skip-connections.

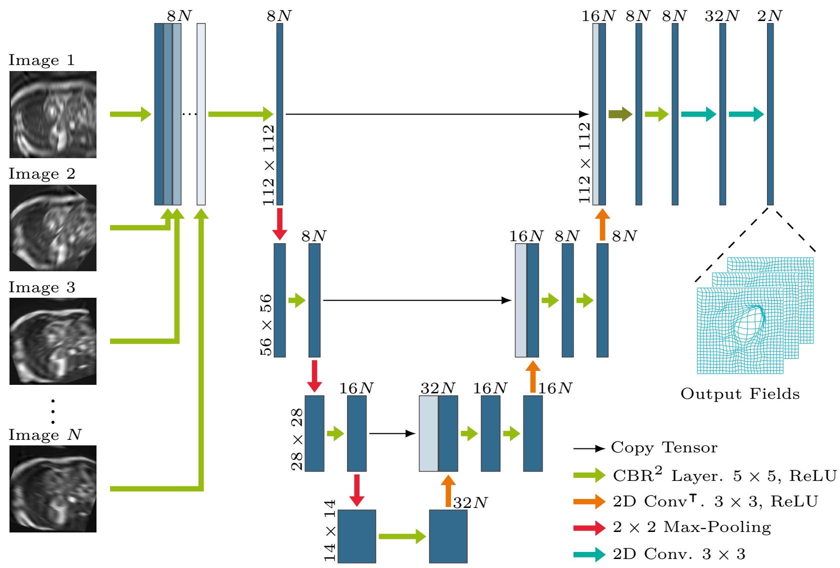


Figure 6.13: The $\mathrm{U_A}$ architecture used in registration experiments. Here, the CSA-cell is placed directly after the decoding pathway of the U-Net.

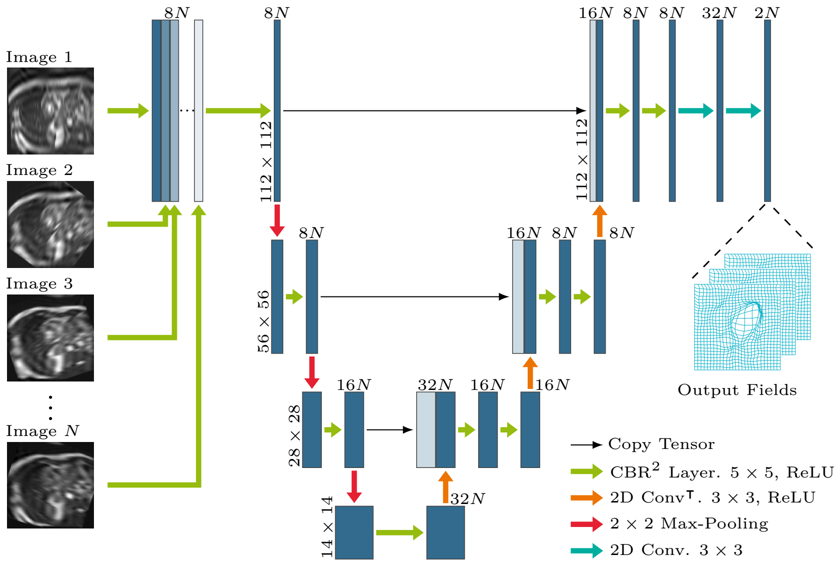


Figure 6.14: The U_C architecture used in the registration experiments. In this comparison architecture, the CSA-layer is omitted, resulting in a standard U-Net approach to deep registration.

All four architectures are built very similar, with the same number of channels in each convolutional layer. This number is chosen depending on the number of images in a sequence. In each of the three attention-supported architectures, the number of internal attention filters is set to $8N$ in each CSA-cell. Since the number of attention filters is set per layer, architecture U_M incorporates more attention filters than the rest.

6.7.3 Composite Loss Function and Performance Metrics

The loss function used to optimise the weights of the different models in these experiments is an extension of the one developed in Section 6.6.3 which now compares a number of images. The design philosophy employed here is the same: the loss function should penalise ℓ_2-differences between registered images and reference as well as subjective difference measured by SSIM. Furthermore, even though the designed fields are diffeomorphic, a *spatial* regularisation function is used which penalises the strength of the registration fields as to prevent the network to output arbitrarily strong fields for self-similar image regions. Additionally, a second, *temporal* regularisation term is added, which penalises the difference in the output fields for sequential images, as they are suspected to be at least somewhat similar.

The image quality functions and the spatial regularisation term from Section 6.6.3 are reused in this definition and extended to multiple registration fields simply by calculating their mean over all N registered images.

The network should, in an optimal state, be able to learn a sequence of deformation fields aligning the untransformed images $\boldsymbol{I}$ to the reference image $\boldsymbol{I}^{(m)}$ from the transformed network input $\hat{\boldsymbol{I}}$. Implicitly, the network should ignore the artifacts resulting from frequency subsampling. The network training loss is thus defined as

$$\begin{aligned} \mathrm{E}(\boldsymbol{I}, \boldsymbol{y}) = \sum_{n=1}^{N} \Big[& \lambda_S (1 - \mathrm{SSIM}(\boldsymbol{I}^{(m)}, \boldsymbol{I}^{(n)}(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(n)}))) + \\ & \lambda_L \ell_2 (\boldsymbol{I}^{(m)}, \boldsymbol{I}^{(n)}(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(n)})) + \lambda_R \mathcal{R}(\hat{\boldsymbol{y}}^{(n)}) \Big] \\ & + \sum_{n}^{N-1} \lambda_{Rt} \| \hat{\boldsymbol{y}}^{(n)} - \hat{\boldsymbol{y}}^{(n+1)} \|_2^2 \end{aligned} \tag{6.15}$$

where the definitions of $\ell_2(\cdot, \cdot)$, $\mathrm{SSIM}(\cdot, \cdot)$ and $\mathcal{R}(\cdot)$ correspond to the definitions used in Section 5.6 and Section 6.6.3. The second sum term weighted by the scalar λ_{Rt} is the spatial regulariser enforcing similarity between sequential fields by penalising their difference in an ℓ_2-norm. λ_S, λ_L, λ_R are again positive scalar weightings for the other components of the loss function.

Similar to the previous experiments, the Dice coefficient of deformed label maps is leveraged as a deformation performance metric. In contrast to these experiments, the network does no longer map images to either end systole or end diastole. In effect, both label maps belonging to these time points have to be interpolated with approximated deformations to compare their aligned overlap. Since the label maps for each sequence correspond to the first and N-th image, both maps can be aligned to a hypothetical label map at index m, so that in a best-case scenario

$$\boldsymbol{L}^{(1)} \left(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(1)} \right) = \boldsymbol{L}^{(m)} = \boldsymbol{L}^{(N)} \left(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(N)} \right), \tag{6.16}$$

with a nearest neighbour interpolation accounting for off-grid values.

Since the dataset does not contain the label map $\boldsymbol{L}^{(m)}$, the Dice coefficient metric used to evaluate network performance for the architectures presented in this chapter is instead defined as

$$\text{Dice}(\boldsymbol{L}^{(1)}(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(1)}),\ \boldsymbol{L}^{(N)}(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(N)})). \tag{6.17}$$

It is notable that the metric presented above exhibits a interesting false maximum, as deformation fields mapping all values either to some arbitrary pixel location or to points outside the image would evaluate to a Dice score of 1. This is not an issue with images, as the static image $\boldsymbol{I}^{(m)}$ does exist and is available in the dataset. Since the Dice coefficient is not used to optimise the network, this behaviour is unproblematic during training.

Lastly, the field folding metric $\delta_{J(\hat{\boldsymbol{y}})}$ is no longer evaluated, as the overall fields are diffeomorphic by design.

6.7.4 Training

As in the previous approach, the network was optimised using mini-batch gradient descent. Even though parts of the networks increased in complexity, a batch size of 10 still proved viable and did not lead to memory issues. The learning rate was set to 1e−5.

Preliminary evaluations of convergence behaviours of the architectures found weighting values for the loss function of $\lambda_S = 1\text{e}6$, $\lambda_L = 5\text{e}2$, $\lambda_{Rt} = 10$ and $\lambda_R = 150$ to yield stable results. The parameters were kept the same for all evaluated networks, without individual analysis of the hyper parameters of the loss function.

Since the networks were more expensive to train in comparison to the pure registration networks, a k-fold cross validation strategy with $k = 5$ was chosen, with each network being trained for 200 epochs. Again, care was taken not to cross-pollute the training and validation datasets by assigning image data from a single cardiac scan to both.

As mentioned previously, all available image augmentation methods were used in this approach. In each training epoch, each image sequence of the training set was augmented before being subsampled in frequency space and shown to the network once. Again, the label maps were augmented using the exact same augmentation using nearest-neighbour interpolation for the random deformation.

The network was implemented in TensorFlow 2.4.1 and trained on a single NVIDIA A100 Tensor Core GPU, on which inference of a vector field sequence takes about $0.2 \cdot N$ seconds.

6.7.5 Image Recovery

Using the approximation created by the network, the image recovery algorithm presented in Section 4.3 can be used for a motion compensated recovery of a sharp image from the sequence $\hat{\boldsymbol{I}}$.

For dense vector field approximates of motion present in the scan subject, no readily available equivalence exists in frequency space. Hence, the frequency encoding and motion operator required in the method can not be combined as in Section 5.5 and have to be evaluated separately.

As motion operator, an appropriate interpolator mapping image values to their registered off-grid positions as defined by the approximated deformation grids is used. Intuitively, the motion operator $\boldsymbol{T} : \mathbb{R}^{s_p} \rightarrow \mathbb{R}^{N \cdot s_p}$ as required in Equation (4.22) transforms a static image into N deformed variants. Since in this approach, the algorithm was tasked with deforming N images to a common target, $\boldsymbol{T}$ needs to incorporate the inverse to the approximated deformation. As the deformation is constructed to be diffeomorphic, the vector field inversion defined in Algorithm 1 can be used for inverting each deformation field.

The transposed motion operator $\boldsymbol{T}^{\intercal} : \mathbb{R}^{N \cdot s_p} \rightarrow \mathbb{R}^{s_p}$ is modelled as a deformation of each of the N images to the common target image. In this implementation, it therefore uses the estimated deformation fields to transform all N images in the sequence. The reduction of N to 1 images is implemented as the sum over all transformed images. Note that this formulation disregards the effect of the interpolation used to build the matrix.

The system matrix $\boldsymbol{A}$ and its Hermitian transpose are modelled as the NFFT-transform of the image for each k-subspace and the adjoint NFFT, respectively.

With these parameters, the ADMM can recover sharp images from a collection of partial scans. Empirically, values of $\rho = 2$, $\lambda_I = 0.001$ were found to yield good image reconstruction results. As in Section 5.4, the wavelet regulariser was chosen from the Biorthogonal 4.4 family, and 4 wavelet levels as well as a symmetric boundary condition were used.

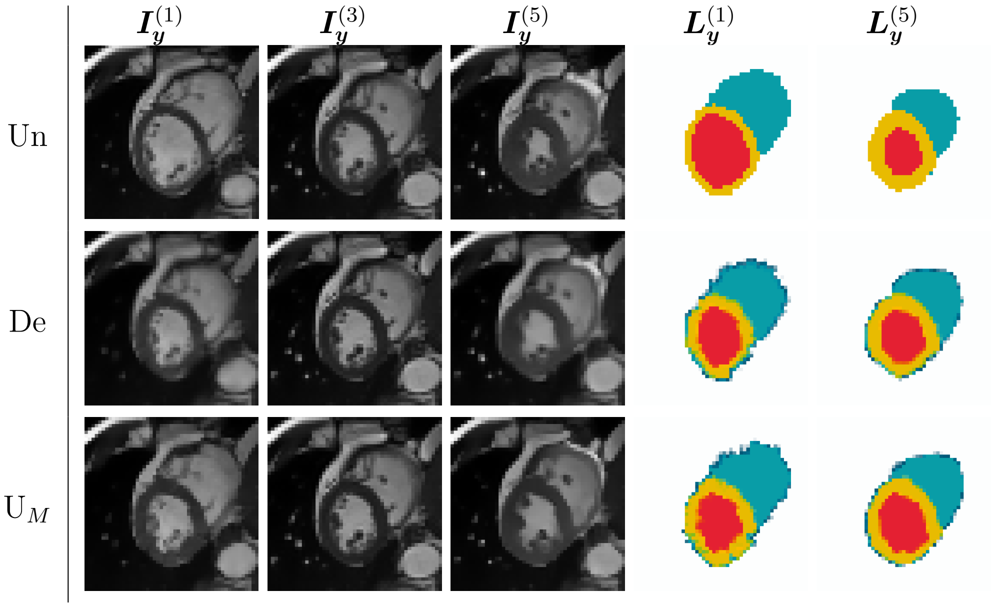


Figure 6.15: Example of deep sequence registration of sequences with $N = 5$ (without MRI simulation) from validation dataset. In the top row, the unregistered case is shown, followed by diffeomorphic demons registration (De) and the best evaluated network (U_M). Only minute differences can be observed between demons and deep registration approach.

6.7.6 Results

Due to the number of evaluated networks and comparisons, only a subset of the available data is shown in the results. Specifically, metrics are shown only for the first and last image in a sequence, as well as the target image. For each evaluated sequence length, a single example image from the validation dataset is shown, with registration outputs from the demons algorithm and the network $\mathrm{U_M}$. The first, middle and last image in the sequence as well as both registered label maps are shown. As in the results shown in Section 6.6.6, each image is cropped to the cardiac region. In the results, the shorthand notations $\boldsymbol{I}_{\boldsymbol{y}}^{(n)} = \boldsymbol{I}^{(n)}\left(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(n)}\right)$ and $\boldsymbol{L}_{\boldsymbol{y}}^{(n)} = \boldsymbol{L}^{(n)}\left(\boldsymbol{x} + \hat{\boldsymbol{y}}^{(n)}\right)$ are used. Note that the images shown are not afflicted by MRI artifacts, but their deformation is inferred from simulated partial MRI measurements of the data.

Figure 6.16 shows the results of metric and error evaluations for the comparison case, with an input sequence of 5 images and without simulating an MRI measurement on the data. As is visible from the graphs, $\boldsymbol{I}^{(3)}$ served as registration target.

As expected, the results from Section 6.6.6 are more or less replicated when comparing $\mathrm{U_M}$ to the demons algorithm and network $\mathrm{U_C}$, which does not incorporate attention. The $\mathrm{U_M}$ network, which closely mirrors the approach presented in

Section 6.6, slightly outperforms the other approaches and the diffeomorphic demons algorithm. The improvement offered by registration in comparison to the unregistered case is smaller than in the previous chapter, which stems from the smaller difference in images. In Section 6.6, the end points of each image series[5] were registered to each other, while here, both extremes are mapped to a middle image. Since the absolute values of all metrics reached by U_M is also slightly worse than in the previous registration experiments, the difference might also indicate an inability of the networks to correctly process multiple images simultaneously. The networks $\mathrm{U_A}$ and $\mathrm{U_B}$ perform worse than $\mathrm{U_M}$, with $\mathrm{U_B}$ especially underperforming on all available metrics.

In Figure 6.15, a registration sample from a validation sample is shown for the demons algorithm and the network $\mathrm{U_M}$. The unregistered case is shown as well to allow easier comparison. As expected, both demons algorithm and network produce subjectively good registration of the images in the sequence to the central target image.

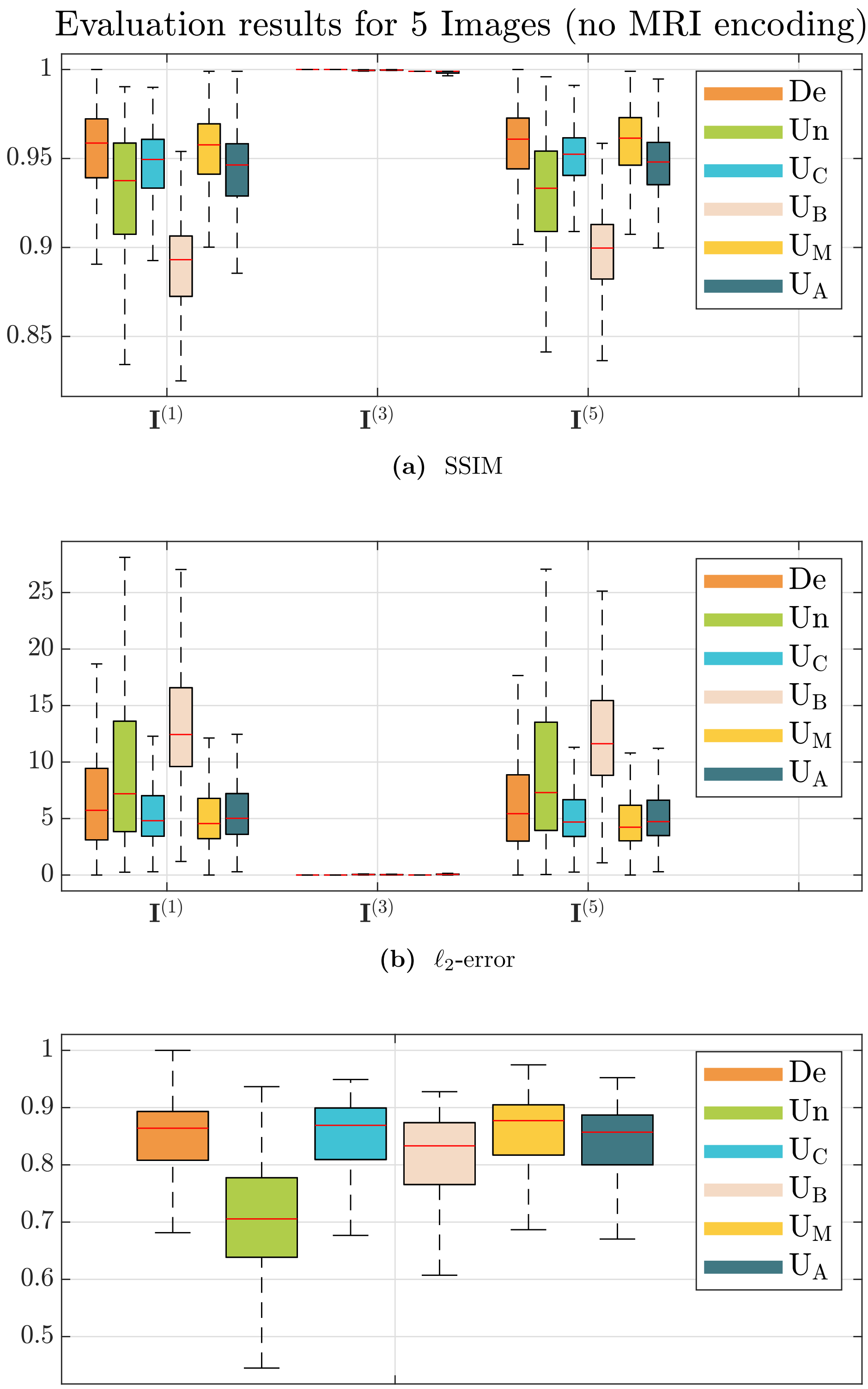
Interestingly, the deep registration approach apparently handles occlusion effects differently from the demons registration method. In the top right quadrant of the sample image, a bright pericardial structure becomes visible in later images in the sequence. The demons algorithm does not alter the new structure in its produced registration, as the method is not designed to handle occlusion effects or register objects of changing luminosity.

Since the deep registration algorithm, like the demons algorithm employed in these experiments, is restricted to producing diffeomorphic fields, it is unable to remove the bright structure. Instead, it learned to produce a deformation which compresses the structure during registration, resulting in better ℓ_2-difference between the image at the end of the sequence and the target. This behaviour continues for the longer sequences, where the demons algorithm delivers progressively worse performance due to the artifacts resulting from frequency subsampling.

In Figure 6.18, the results for the registration of sequences with 5 partial PROPELLER-measured images are shown, again with a registration target $\boldsymbol{I}^{(3)}$.

Here, it becomes readily apparent that the Demons algorithm is not suitable for registering partial MRI images, which is unsurprising as it lacks the capability

[5]The images of end systole and end diastole



(a) SSIM

(b) ℓ_2-error

(c) Dice-coefficient

Figure 6.16: Results of evaluation and error metrics for sequence registration experiments. Here, the comparison case with 5 images but no MRI simulation is shown. Evaluated algorithms are the unregistered case (Un), Demons registration (De), and the four tested network architectures $\mathrm{U_C}$, $\mathrm{U_B}$, $\mathrm{U_M}$, $\mathrm{U_A}$ are shown. Median is shown as red line, the box marks 25th and 75th percentile, whiskers mark extremes of range. As expected from previous experiments, U_M outperforms all comparison approaches.

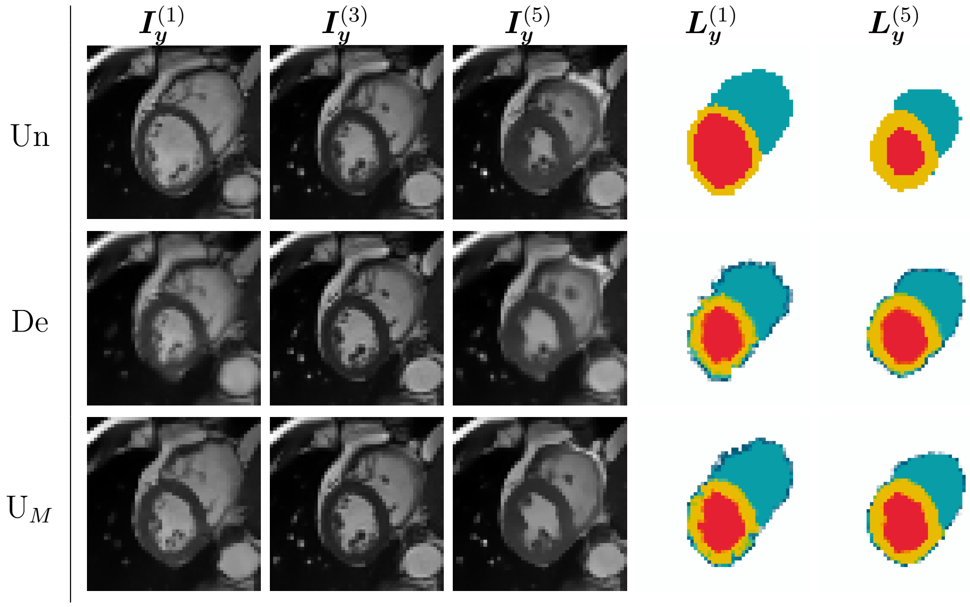


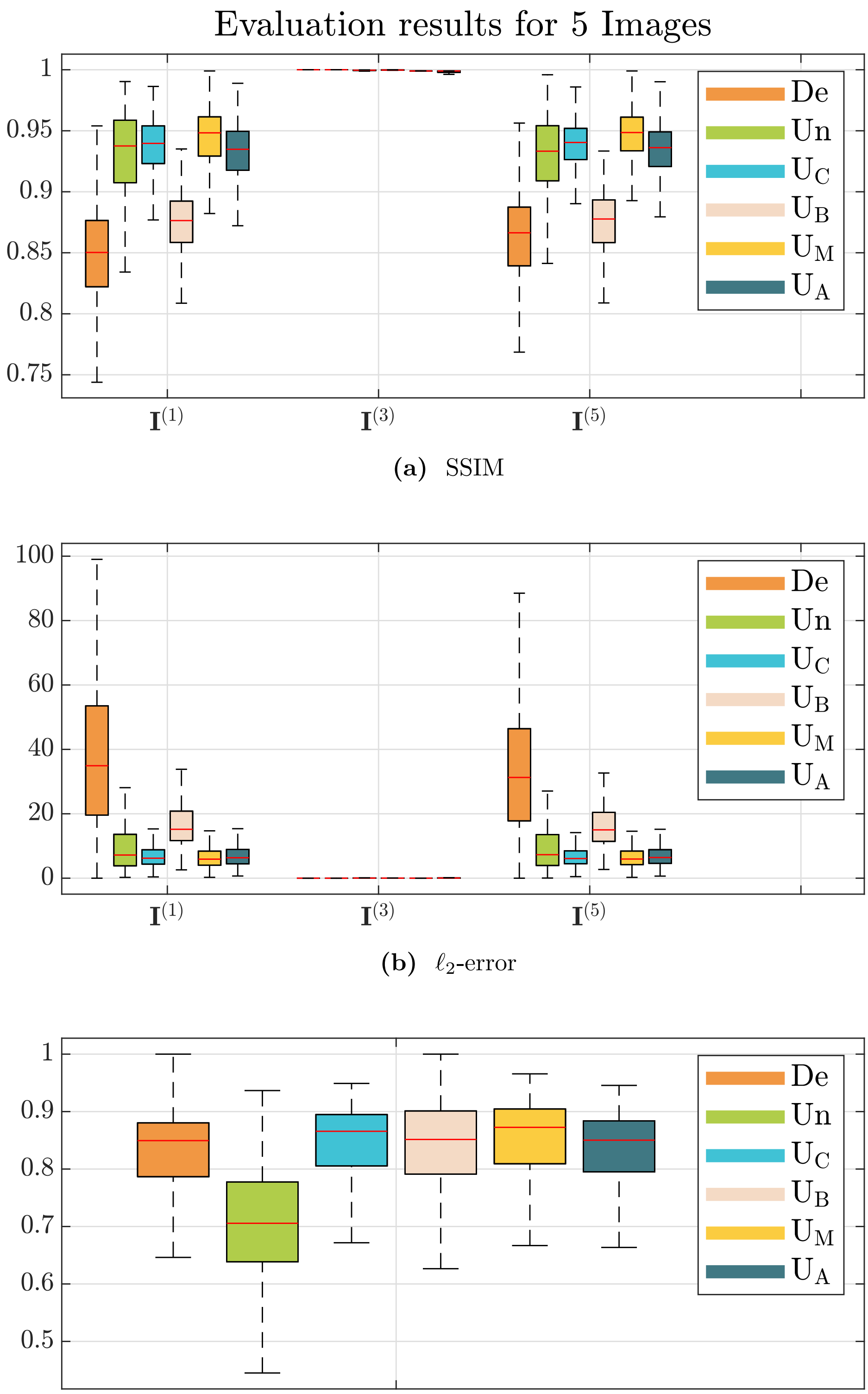
Figure 6.17: Example of deep sequence registration of sequences with $N = 5$ from validation dataset. In the top row, the unregistered case is shown, followed by diffeomorphic demons registration (De) and the best evaluated network (U_M). The demons algorithm struggles to align the finer details in the structures, while deep registration sill yields subjectively good results.

to discriminate between image content and ringing artifact caused by frequency subsampling. This is apparent by the strong deterioration in metrics and ℓ_2-error when comparing the results of the demons registration between the comparison case without MRI encoding and results when including the subsampling. Furthermore, most networks do not perform significantly better that not registering at all, with only U_M offering some improvement in ℓ_2 error and SSIM after registration. Interestingly, the Dice metric is less affected by the performance degradation, and the registration results still look subjectively acceptable.

These observation only continue in the experiments for longer image sequences. Figure 6.20 shows the results for experiments using sequences of 10 MRI encoded images as input. Here, the registration target is $\boldsymbol{I}^{(5)}$.

In these results, none of the evaluated networks offer any measurable improvement in ℓ_2-difference or SSIM when compared to the unregistered case. This indicates an inability of the network to learn useful generalisations from the corrupt input data and overfitting to the training data.

Notably, the Dice coefficient improve by registration for all cases. In the case of U_B, this is due to divergence in the networks training. In the U_B-architecture,



(a) SSIM

(b) ℓ_2-error

(c) Dice Coefficient

Figure 6.18: Results of evaluation and error metrics for sequence registration experiments. Here, the results for sequences with 5 MRI-encoded images are shown. Evaluated algorithms are the unregistered case (Un), Demons registration (De), and the four tested network architectures $\mathrm{U_C}$, $\mathrm{U_B}$, $\mathrm{U_M}$, $\mathrm{U_A}$ are shown. Median is shown as red line, the box marks 25th and 75th percentile, whiskers mark extremes of range. While still offering improvement, the performance of all approaches suffers from the perturbations introduced by MRI simulation.

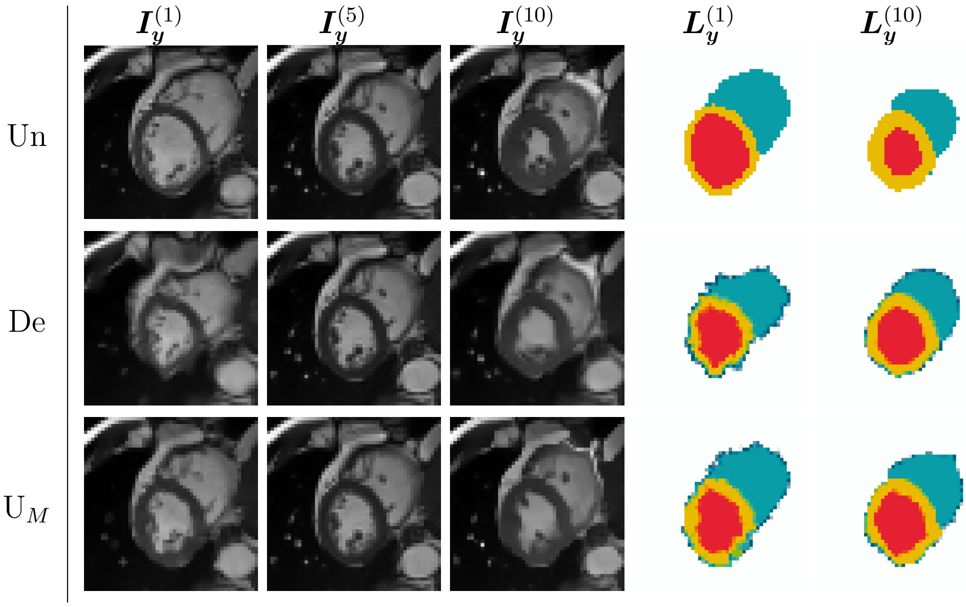


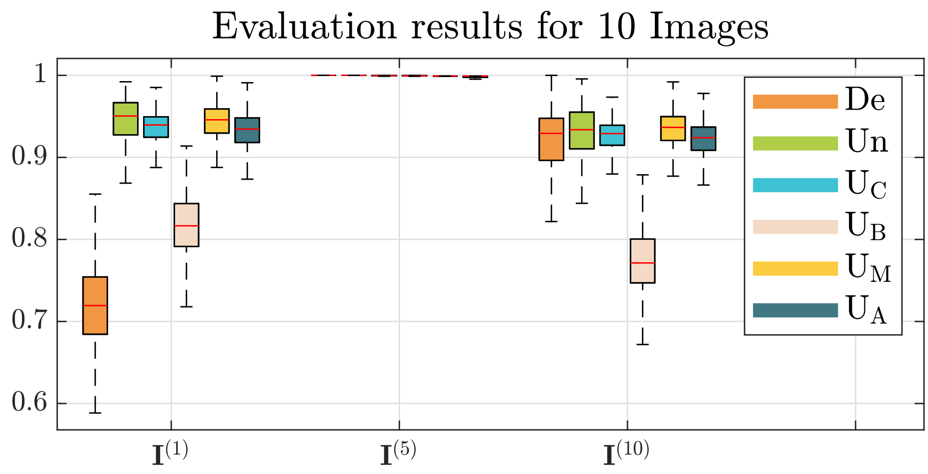
Figure 6.19: Example of deep sequence registration of sequences with $N = 10$ from validation dataset. In the top row, the unregistered case is shown, followed by diffeomorphic demons registration (De) and the best evaluated network (U_M). Note that the demons algorithm causes strong erroneous deformations around the cardiac region, which are absent in deep registration.

strong deformation moves parts of both label maps to the same arbitrary position in or outside the image. The unfeasible registration produced leads to improved Dice score and bad image quality scores.

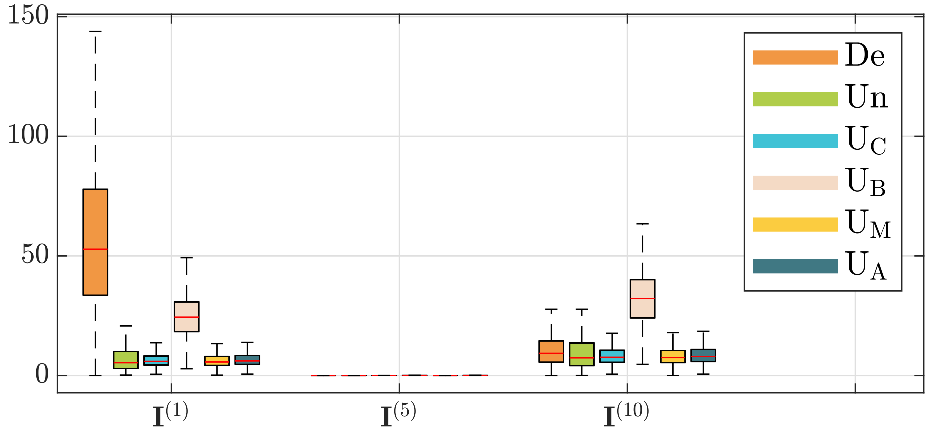
In spite of the values of ℓ_2 error and $SSIM$, the network U_M was still able to learn a subjectively correct deformation in the region of interest, shown in the excerpt of the registration results for this setting in Figure 6.19. While the demons algorithm is perturbed by the presence of ringing artifacts, as is visible in the erroneous deformation of the pericardium in the upper part of the images, deep registration correctly deforms only the cardiac region.

While the results here do not readily indicate a capability of the presented networks to tackle real-world MRI artifacts by one-shot registering, it is notable that the application of the CSA layer led to network improvements in all cases when compared to the attention-less network U_C. While being sufficient for the simpler registration task in Section 6.6, the chosen dataset might be too small to correctly let the network learn useful generalisations without overfitting to the training subset.

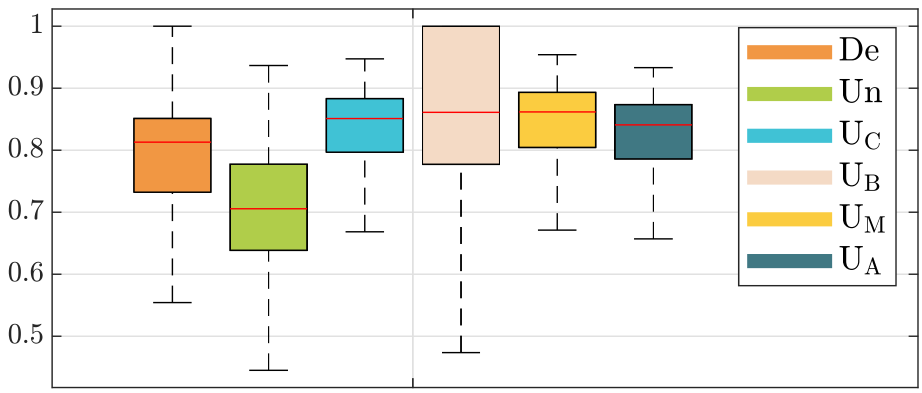
Reconstructing images from these imperfect registrations was challenging as well. As the inverse of the estimation was used in place of an estimation of the inverse,



(a) SSIM



(b) ℓ_2-error



(c) Dice Coefficient

Figure 6.20: Results of evaluation and error metrics for sequence registration experiments. Here, the results for sequences with 10 MRI-encoded images are shown. Evaluated algorithms are the unregistered case (Un), Demons registration (De), and the four tested network architectures $\mathrm{U_C}$, $\mathrm{U_B}$, $\mathrm{U_M}$, $\mathrm{U_A}$ are shown. Median is shown as red line, the box marks 25th and 75th percentile, whiskers mark extremes of range. Most networks and the demons algorithm worsen image quality measured by ℓ_2-error and SSIM by deformation.

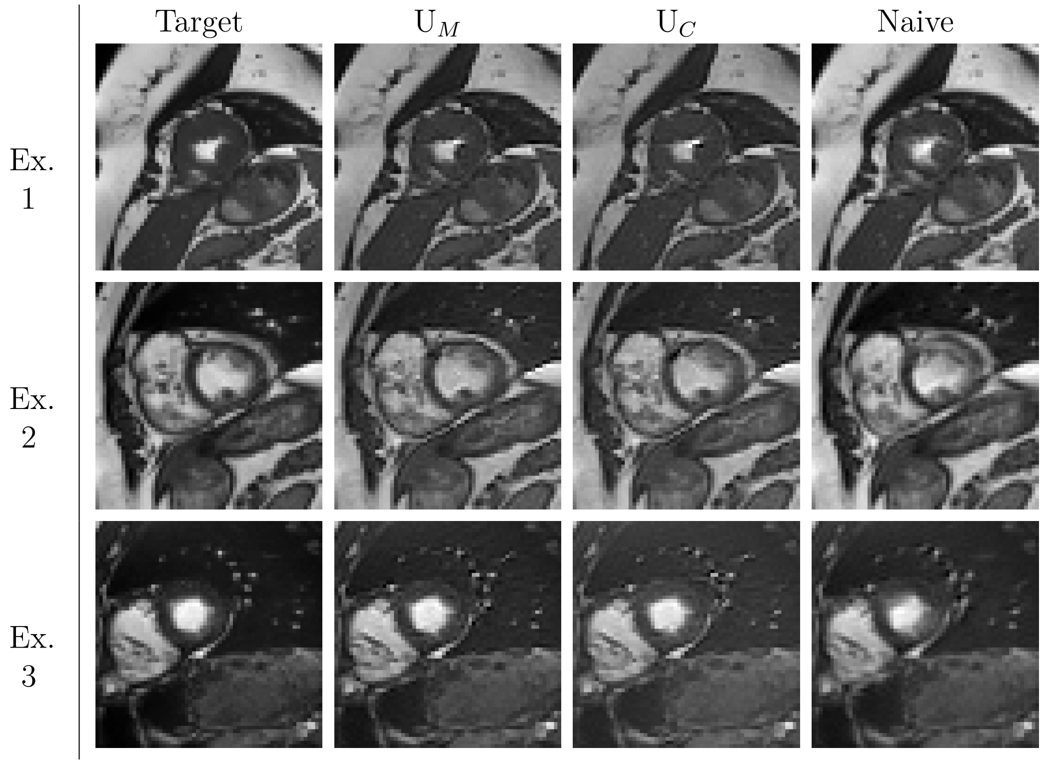


Figure 6.21: Three examples of motion compensated results using deep sequence registration of sequences with $N = 10$ from validation dataset. The motion compensated recovery results using the inferred deformation of U_M and U_C are shown, together with the ground truth target image and a naive recovery without using motion compensation.

minute errors in both accumulate and lead to deviations in the combined system used in the CGNR. In practice, this lead to divergence of the conjugate gradient method. The tendency of the optimisation method to diverge was circumvented by scaling the gradients of the CGNR with a factor of 0.2 and limiting its number of iterations inside the ADMM. Further research has to include either the inverse of the approximated deformation or the recovered image itself in the loss function of the network, or infer them directly. Another avenue would include stabilisation of the CGNR, for example by applying a preconditioning to the system of equations.

Sample recovered images of the networks are provided in Figure 6.21, which also includes a 'naive' recovery result without motion compensation. The images shown are again cropped to the cardiac region. Here, it becomes apparent that application of the motion recovery allows a significant sharpening of the moving organs, in this case the heart. While the recovery algorithm is powerful enough to reconstruct a good estimation of the image even without motion compensation, the regions which are in motion during the scan - for this dataset, the heart - are blurred. Motion compensation allows recovery of finer structures and sharp edges, while also removing some of the artifacts resulting from motion.

While this showcases the capabilities of the algorithm, it also exemplifies a sort of shortcoming in the data set: As the motion between images is subtle and restricted to a single contraction of the heart, motion artifacts appear in images produced by the chosen reconstruction method as a blurring of the moving tissue. While the presented network is able to reconstruct sharp images and tissue borders, the effect is similarly subtle. An analysis of its behaviour on larger-scale motions would therefore be an interesting next step for these architectures. While the images produced using deformations from the attention-supported network U_M are subjectively sharper than the ones using the output of its attention-less variant U_C, a more focused analysis of image quality post registration is a subject of future research.

7 Conclusion

7.1 Summary

The methods presented in this work approach the underlying task of registration in MRI from two very different perspectives. As such, methods explored in the thesis come from two different fields of research.

First, in Chapter 5, an efficient parameter search algorithm designed to find solutions to the non-convex problem of motion recovery was presented. While most common autofocusing algorithm leverage a convex optimisation scheme, they often suffer from a reduced motion range which can be approximated by the approach. Furthermore, gradient descent schemes rely on the application of a differentiable motion model.

In modern literature, parameter search approaches are often disregarded because of their inherent scaling issues, as the computational complexity scales exponentially with the number of free parameters in the used motion model. This disadvantage is overcome by the presented algorithm by embedding the parameter search in a multi-scale approach, yielding sub-pixel motion estimates in a feasible computation time. Furthermore, the calculations necessary for the algorithm are highly parallelizable, which further reduces runtime. This allows for time-efficient joint motion and image estimation by the presented algorithm. While experiments employing the presented method use simple rigid motion models, an extension to different, arbitrary motion models is possible, as the method does not place restrictions regarding their differentiability.

Pairing the motion estimation algorithm with a convex splitting scheme and taking advantage of the sparsity of natural images in wavelet domains, the presented algorithm is able to recover sharp images even from highly motion corrupted data. Its abilities were evaluated on a large number of simulations.

Second, in Chapter 6, a deep learning approach towards image registration was taken and convolutional neural networks were applied to the task. The hypothesis was that convolutional neural registration networks benefit from adding an attention mechanism to incorporate a model for long-range relationships in data feature maps. For this, a novel attention mechanism is introduced in Section 6.3. In Section 6.6, the resulting CSA-layer was then used as an additional layer inside the skip-connection path of a U-Net architecture. The resulting network was trained for elastic, model-free medical image registration from real-world MRI scans and it was shown that the usage of the novel mechanism enables the proposed network to outperform similar approaches. Furthermore, the results show that the employed regularisation function is suitable to deter the network from learning deformations which include occlusion effects.

In Section 6.7, the task of image registration is extended in a novel way to sequences of frequency subspaces sampled from sequences of measurements of real beating hearts as they might arise in one of the most common MRI scan procedures. Different architectures were compared, with different placements of the CSA-layer in their structures. Extensive evaluation of these architectures showed that the CSA-layer enables the networks to learn better registrations than attention-free architectures and numeric methods for this extended task. Unsurprisingly however, the overall quality of the registrations created from the subsampled k-spaces was still lacking in comparison to registration of fully reconstructed images.

The experimental results of this approach shown in Section 6.7.6 also contain exemplary image recoveries using the motion estimate of the network, where a wavelet regularised convex splitting algorithm incorporating the approximation was used to recover images. While the images that were reconstructed in a motion compensated scheme exhibit sharper edges around the moving tissue than non-compensated recoveries, the results mainly highlight the potential of the approach.

Experiments with larger motion amplitudes and therefore more intense motion artifacts should give further insights to the capabilities of the approach for blind fully elastic MRI motion compensation.

7.2 Outlook

Even though the approaches presented in this thesis were capable of improving their respective problems, several research hypotheses and ideas for further improvement remain open. Some of these proposals are presented in this section.

Different Motion Models for Parameter Search

While the results in Chapter 5 show that the method is able to recover images corrupted by strong motions, this method still has some caveats. The motion model chosen in the presented setup only estimates rigid motion, while the method itself is capable of finding parameter sequences for arbitrary motions. The capabilities of the method when paired with a different motion parametrisation is an interesting venue for further research, as is the research into suitable motion models itself. Since the computational complexity of the method rises exponentially with the number of estimated parameters, research into low-parametric motion models is particularly promising in this regard.

Another field of research connected to the above is a investigation into low-parametric learned models, which could be a bridge between more classical image recovery algorithms and deep learning methods. Since the motion expressed by a body part or organ is not random but defined by its function, it seems reasonable to find the space of associated motions to be a low-parametric embedded manifold. Deep learning, or even manifold embedding techniques, might then be able to recover a low-parametric estimation of that space.

CUDA implementation of Alternating Motion Model

The estimation of the error matrix used in the path search in Chapter 5 is highly parallelizable, as the values of different states are not dependent on each other. As the norm used in the equations can be replaced by a correlation, the use of widely available graphics hardware has the potential to massively speed up the process of setting up the matrices used in motion estimation. The same is true for the NUDFT-I transformation, which could in theory be implemented on GPU hardware.

Alternating Deep Motion and Image estimation

The deep learning architecture presented in Section 6.7 is currently posed as a one-shot autofocusing approach, estimating the motion field in a single inference on the whole image sequence. This implies that the model has an inherent idea of what the sharp image looks like. However, using an alternating approach to image recovery might be viable. Then, the deformation field produced in a single inference would be used to improve the image quality of the target image, and the estimation of registering each sequence image to the target repeated. This could also be formulated as a recurrent architecture, since all operations in use are differentiable.

Incorporation of Coil Sensitivity Profiles and Inhomogeneities

Every method in this work is formulated for a perfect, single coil MRI measurement with uniform sensitivity profile over the whole field of view. However, this is very seldom true in practice. Modern MRI uses parallelization techniques for its imaging process, receiving its signal on multiple coils with spatially dependent sensitivities that cover the region of interest. Incorporating these methods into the presented approaches will be another important step on the way of the approaches presented in this work towards clinical relevance. Furthermore, modern MRI is not perfect, as the gradient fields and RF pulses exhibit some inhomogeneities in their field profiles. The potential impact of these imperfections on the presented methods remains to be evaluated.

Clinical Testing of Methods

Currently, most methods presented in this thesis are evaluated only on - to some extend - simulated data. While the base image, its MRI measurement and motion are simulated in Chapter 5, the deep learning models presented in Chapter 6 train on real world data. However, the MRI measurement process used in Section 6.7 is again a simulation. Therefore, a necessary step in the further analysis of the presented approaches includes their evaluation on raw k-space data measured from real patients.

Evaluation of Registration Networks on different Datasets

While the results of the deep learning approaches towards registration of medical images are promising, the amount of available data likely limits its performance. In this thesis, strong data augmentation is in use to circumvent this problem. However, incorporating other datasets of natural images into its training scheme might improve the robustness of the approach. Furthermore, while the network was developed for images from cardiac scans, its performance on other motives remains to be evaluated.

Multihead attention for CSA-Layer

As stated in the Section 3.6, the attention mechanism in use in the transformer architecture is composed of multiple attention heads, allowing the layer to focus on different data structures at once. This is currently not in use for the convolutional attention layer developed for this work. As many current machine learning frameworks now include *grouped* convolutions, which stratify the learned weight kernel into multiple parts, the presented convolutional attention could be extended to include multiple attention heads by changing the grouping behaviour of its convolutional operations. This might allow the architecture to model biological structures and behaviours more closely and should be a viable way for its further development.

Abbreviations

ACDC . . . . . Automated Cardiac Diagnosis Challenge
ADAM . . . . Adaptive Moment Estimation Algorithm
ADMM . . . . Alternating Directions Method of Multipliers
AN . . . . Artificial Neuron
ANN . . . . . . Artificial Neural Network
CBR^2 . . . . . Dual $2D$-Convolution, Batch-Norm and ReLU-Layer
CG/CGNR . . Conjugate Gradient method (on the Normal Residual)
CSA . . . . . . Convolutive Self-Attention
DFT . . . . . . Discrete Fourier Transform
FC . . . . Fully Connected (Layer)
FFT . . . . . . Fast Fourier Transform
MGD . . . . . Mini-Batch Gradient Descent
ML . . . . Machine Learning
MR . . . . . . Magnetic Resonance
MRI . . . . . . Magnetic Resonance Imaging
NFFT . . . . . Non-Uniform Fast Fourier Transform
NUDFT-I . . . Non-Uniform Discrete Fourier Transform of Type I
PROPELLER Periodically Rotated Overlapping Parallel Lines with Enhanced Reconstruction
ReLU . . . . . Rectified Linear Unit
SGD . . . . . . Stochastic Gradient Descent
SSIM . . . . . Structural Similarity Index Measure
TV . . . . Total Variation
UIQI . . . . . . Universal Image Quality Index

List of Figures

References

[1] T. J. Parbs, A. Möller, and A. Mertins, "Retrospective Blind MR Image Recovery with Parametrized Motion Models," in *Bildverarb. Für Med. 2019*, H. Handels, T. M. Deserno, A. Maier, K. H. Maier-Hein, C. Palm, and T. Tolxdorff, Eds., ser. Informatik aktuell, Wiesbaden: Springer Fachmedien, 2019, pp. 140–145.

[2] T. J. Parbs, P. Koch, and A. Mertins, "Convolutive Attention for Image Registration," in *2022 30th Eur. Signal Process. Conf. EUSIPCO*, Aug. 2022, pp. 1348–1352.

[3] OECD, *Health at a Glance 2021.* 2021, 274 pp.

[4] Z.-P. Liang and P. C. Lauterbur, *Principles of Magnetic Resonance Imaging.* SPIE Optical Engineering Press Belllingham, WA, 2000.

[5] F. Bloch, "Nuclear Induction," *Phys. Rev.*, vol. 70, no. 7-8, pp. 460–474, Oct. 1, 1946.

[6] D. B. Twieg, "The k-trajectory formulation of the NMR imaging process with applications in analysis and synthesis of imaging methods," *Med. Phys.*, vol. 10, no. 5, pp. 610–621, 1983.

[7] S. Ljunggren, "A Simple Graphical Representation of Fourier-Based Imaging Methods," *Journal of Magnetic Resonance (1969)*, vol. 54, no. 2, pp. 338–343, Sep. 1, 1983.

[8] Y. Hohri, K. Itatani, A. Matsuo, *et al.*, "Estimating the Haemodynamic Streamline Vena Contracta as the Effective Orifice Area Measured from Reconstructed Multislice Phase-contrast MR Images for Patients with Moderately Accelerated Aortic Stenosis," *Magnetic Resonance in Medical Sciences*, vol. 21, Jul. 30, 2021.

[9] J. G. Pipe, "Motion Correction with PROPELLER MRI: Application to Head Motion and Free-Breathing Cardiac Imaging," *Magn. Reson. Med.*, vol. 42, no. 5, pp. 963–969, 1999.

[10] K. P. Forbes, J. G. Pipe, C. R. Bird, and J. E. Heiserman, "PROPELLER MRI: Clinical Testing of a Novel Technique for Quantification and Compensation of Head Motion," *J. Magn. Reson. Imaging*, vol. 14, no. 3, pp. 215–222, 2001.

[11] F.-N. Wang, T.-Y. Huang, F.-H. Lin, T.-C. Chuang, N.-K. Chen, H.-W. Chung, C.-Y. Chen, and K. K. Kwong, "PROPELLER EPI: An MRI Technique Suitable for Diffusion Tensor Imaging at High Field Strength With Reduced Geometric Distortions," *Magn Reson Med*, vol. 54, no. 5, pp. 1232–1240, Nov. 2005.

[12] K. P. Pruessmann, M. Weiger, M. B. Scheidegger, and P. Boesiger, "SENSE: Sensitivity Encoding for fast MRI," *Magn. Reson. Med.*, vol. 42, no. 5, pp. 952–962, 1999.

[13] Q. Wen, L. Feng, K. Zhou, and Y.-C. Wu, "Rapid Golden-Angle Diffusion-Weighted Propeller MRIfor Simultaneous Assessment of ADC and IVIM," *Neuroimage*, vol. 223, pp. 117–127, Dec. 2020.

[14] K. Arfanakis, A. A. Tamhane, J. G. Pipe, and M. A. Anastasio, "K-space Undersampling in PROPELLER Imaging," *Magn. Reson. Med.*, vol. 53, no. 3, pp. 675–683, 2005.

[15] A. Mertins, *Signaltheorie: Grundlagen der Signalbeschreibung, Filterbänke, Wavelets, Zeit-Frequenz-Analyse, Parameter- und Signal-schätzung.* Wiesbaden: Springer Fachmedien, 2020.

[16] L. A. Shepp and B. F. Logan, "The Fourier Reconstruction of a Head Section," *IEEE Trans. Nucl. Sci.*, vol. 21, no. 3, pp. 21–43, Jun. 1974.

[17] L. Sha, H. Guo, and A. W. Song, "An Improved Gridding Method for Spiral MRI Using Nonuniform Fast Fourier Transform," *Journal of Magnetic Resonance*, vol. 162, no. 2, pp. 250–258, Jun. 1, 2003.

[18] J. Fessler and B. Sutton, "Nonuniform Fast Fourier Transforms Using Min-Max Interpolation," *IEEE Trans. Signal Process.*, vol. 51, no. 2, pp. 560–574, Feb. 2003.

[19] J.-M. Lin, "Python Non-Uniform Fast Fourier Transform (PyNUFFT): An Accelerated Non-Cartesian MRI Package on a Heterogeneous Platform (CPU/GPU)," *J. Imaging*, vol. 4, no. 3, p. 51, 3 Mar. 2018.

[20] J. Keiner, S. Kunis, and D. Potts, "Using NFFT 3 - A Software Library for Various Nonequispaced Fast Fourier Transforms," *ACM Trans. Math. Softw.*, vol. 36, no. 4, 19:1–19:30, Aug. 1, 2009.

[21] C. Shannon, "Communication in the Presence of Noise," *Proc. IRE*, vol. 37, no. 1, pp. 10–21, Jan. 1949.

[22] H. Landau, "Sampling, Data Transmission, and the Nyquist Rate," *Proc. IEEE*, vol. 55, no. 10, pp. 1701–1706, Oct. 1967.

[23] H. J. Landau, "Necessary Density Conditions for Sampling and Interpolation of Certain Entire Functions," *Acta Math.*, vol. 117, pp. 37–52, Jan. 1967.

[24] D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory.* New York: Wiley, Jun. 15, 1949.

[25] W. S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1, 1943.

[26] F. Rosenblatt, "The perceptron - A perceiving and recognizing automaton," Cornell Aeronautical Laboratory, Ithaca, New York, 85-460-1, Jan. 1957.

[27] M. Minsky and S. Papert, *Perceptrons; an Introduction to Computational Geometry.* MIT Press, 1969, 272 pp.

[28] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks Are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, Jan. 1, 1989.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE Int. Conf. Comput. Vis. ICCV*, Dec. 2015, pp. 1026–1034.

[30] X. Glorot and Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks," in *Proc. Thirteen. Int. Conf. Artif. Intell. Stat.*, JMLR Workshop and Conference Proceedings, Mar. 31, 2010, pp. 249–256.

[31] D. R. Wilson and T. R. Martinez, "The General Inefficiency of Batch Training for Gradient Descent Learning," *Neural Networks*, vol. 16, no. 10, pp. 1429–1451, Dec. 1, 2003.

[32] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980*, 2014.

[33] E. M. Dogo, O. J. Afolabi, N. I. Nwulu, B. Twala, and C. O. Aigbavboa, "A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks," in *2018 Int. Conf. Comput. Tech. Electron. Mech. Syst. CTEMS*, Dec. 2018, pp. 92–99.

[34] S. Linnainmaa, "Taylor Expansion of the Accumulated Rounding Error," *BIT*, vol. 16, no. 2, pp. 146–160, Jun. 1, 1976.

[35] P. J. Werbos, "Applications of Advances in Nonlinear Sensitivity Analysis," in *Syst. Model. Optim.*, R. F. Drenick and F. Kozin, Eds., ser. Lecture Notes in Control and Information Sciences, Berlin, Heidelberg: Springer, 1982, pp. 762–770.

[36] S. Hochreiter, "Untersuchungen zu Dynamischen Neuronalen Netzen (Diploma Thesis), Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München," 1991.

[37] P. Wennker, "Machine Learning," in *Künstliche Intelligenz in der Praxis: Anwendung in Unternehmen und Branchen: KI wettbewerbs- und zukunftsorientiert einsetzen*, P. Wennker, Ed., Wiesbaden: Springer Fachmedien, 2020.

[38] K. Fukushima and S. Miyake, "Neocognitron: A New Algorithm for Pattern Recognition Tolerant of Deformations and Shifts in Position," *Pattern Recognition*, vol. 15, no. 6, pp. 455–469, Jan. 1, 1982.

[39] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, "Rectifier Nonlinearities Improve Neural Network Acoustic Models," in *Proc. Mach. Learn. Res.*, vol. 30, Atlanta, GA, 2013.

[40] T. Szandała, "Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks," *arXiv:2010.09458*, 2021.

[41] D. H. Hubel and T. N. Wiesel, "Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex," *J Physiol*, vol. 160, no. 1, pp. 106–154.2, Jan. 1962.

[42] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec. 1, 1989.

[43] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. - Vol. 2*, ser. NIPS'14, Cambridge, MA, USA: MIT Press, Dec. 8, 2014, pp. 3104–3112.

[44] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches," in *Proc. SSST-8 Eighth Workshop Syntax Semant. Struct. Stat. Transl.*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111.

[45] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd Int. Conf. Learn. Represent. ICLR 2015 San Diego CA USA May 7-9 2015 Conf. Track Proc.*, Y. Bengio and Y. LeCun, Eds., 2015.

[46] T. Brown, B. Mann, N. Ryder, *et al.*, "Language Models are Few-Shot Learners," in *Adv. Neural Inf. Process. Syst.*, vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901.

[47] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proc. 58th Annu. Meet. Assoc. Comput. Linguist.*, Online: Association for Computational Linguistics, 2020, pp. 7871–7880.

[48] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," *OpenAI*, 2018.

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Adv. Neural Inf. Process. Syst.*, vol. 30, Curran Associates, Inc., 2017.

[50] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional Sequence to Sequence Learning," in *Proc. 34th Int. Conf. Mach. Learn. - Vol. 70*, ser. ICML'17, Sydney, NSW, Australia: JMLR.org, Aug. 6, 2017, pp. 1243–1252.

[51] D. L. Donoho, "Unconditional Bases Are Optimal Bases for Data Compression and for Statistical Estimation," *Applied and Computational Harmonic Analysis*, vol. 1, no. 1, pp. 100–115, Dec. 1993.

[52] B. W. Silverman, J. C. Vassilicos, and D. J. Field, “Wavelets, Vision and the Statistics of Natural Scenes,” *Philos. Trans. R. Soc. Lond. Ser. Math. Phys. Eng. Sci.*, vol. 357, no. 1760, pp. 2527–2542, Sep. 1999.

[53] P. Roy, S. Ghosh, S. Bhattacharya, and U. Pal, “Effects of Degradations on Deep Neural Network Architectures,” *arXiv:1807.10108*, Jul. 26, 2018.

[54] O. Bernard, A. Lalande, C. Zotti, *et al.*, “Deep Learning Techniques for Automatic MRI Cardiac Multi-Structures Segmentation and Diagnosis: Is the Problem Solved?” *IEEE Trans. Med. Imaging*, vol. 37, no. 11, pp. 2514–2525, Nov. 2018.

[55] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing* (Applied and Numerical Harmonic Analysis). New York, NY: Springer New York, 2013.

[56] T. T. Nguyen, C. Soussen, J. Idier, and E.-H. Djermoune, “NP-Hardness of $\ell 0$ Minimization Problems: Revision and Extension to the Non-Negative Setting,” in *2019 13th Int. Conf. Sampl. Theory Appl. SampTA*, Jul. 2019, pp. 1–4.

[57] A. Loktyushin, H. Nickisch, R. Pohmann, and B. Schölkopf, “Blind Retrospective Motion Correction of MR Images,” *Magn Reson Med*, vol. 70, no. 6, pp. 1608–1618, Dec. 2013.

[58] R. Tibshirani, “Regression Shrinkage and Selection via the Lasso,” *J. R. Stat. Soc. Ser. B Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.

[59] J. Zeng, Z. Peng, and S. Lin, “Gaita: A Gauss–Seidel Iterative Thresholding Algorithm for ℓq Regularized Least Squares Regression,” *Journal of Computational and Applied Mathematics*, vol. 319, pp. 220–235, Aug. 1, 2017.

[60] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, “Least angle regression,” *Ann. Stat.*, vol. 32, no. 2, pp. 407–499, Apr. 2004.

[61] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed Optimization and Statistical Learning Via the Alternating Direction Method of Multipliers,” *Found. Trends® Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[62] M. R. Hestenes, E. Stiefel, *et al.*, “Methods of Conjugate Gradients for Solving Linear Systems,” *J. Res. Natl. Bur. Stand.*, vol. 49, no. 6, pp. 409–436, 1952.

[63] M. Chen, W. Lu, Q. Chen, K. Ruchala, and G. Olivera, “A Simple Fixed-Point Approach to Invert a Deformation Field,” *Medical physics*, vol. 35, pp. 81–8, Feb. 1, 2008.

[64] M. K. Chung, K. J. Worsley, T. Paus, C. Cherif, D. L. Collins, J. N. Giedd, J. L. Rapoport, and A. C. Evans, “A Unified Statistical Approach to Deformation-Based Morphometry,” *Neuroimage*, vol. 14, no. 3, pp. 595–606, Sep. 2001.

[65] C. Moler and C. Loan, “Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later,” *Society for Industrial and Applied Mathematics*, vol. 45, pp. 3–49, Mar. 1, 2003.

[66] X. Hu and S.-G. Kim, "Reduction of signal fluctuation in functional MRI using navigator echoes," *Magn. Reson. Med.*, vol. 31, no. 5, pp. 495–503, 1994.

[67] Z. W. Fu, Y. Wang, R. C. Grimm, P. J. Rossman, J. P. Felmlee, S. J. Riederer, and R. L. Ehman, "Orbital navigator echoes for motion measurements in magnetic resonance imaging," *Magn. Reson. Med.*, vol. 34, no. 5, pp. 746–753, 1995.

[68] X. Zhong, B. S. Spottiswoode, C. H. Meyer, C. M. Kramer, and F. H. Epstein, "Imaging three-dimensional myocardial mechanics using navigator-gated volumetric spiral cine DENSE MRI," *Magn. Reson. Med.*, vol. 64, no. 4, pp. 1089–1097, 2010.

[69] A. Loktyushin, H. Nickisch, R. Pohmann, and B. Schölkopf, "Blind multirigid retrospective motion correction of MR images," *Magn. Reson. Med.*, vol. 73, no. 4, pp. 1457–1468, 2015.

[70] A. Taymourtash, H. Kebiri, E. Schwartz, K.-H. Nenning, S. Tourbier, G. Kasprian, D. Prayer, M. Bach Cuadra, and G. Langs, "Spatio-Temporal Motion Correction and Iterative Reconstruction of In-Utero Fetal fMRI," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 603–612.

[71] E. Kuzmina, A. Razumov, O. Y. Rogov, E. Adalsteinsson, J. White, and D. V. Dylov, "Autofocusing+: Noise-Resilient Motion Correction in Magnetic Resonance Imaging," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 365–375.

[72] S. Wang, C. Qin, N. Savioli, C. Chen, D. P. O'Regan, S. Cook, Y. Guo, D. Rueckert, and W. Bai, "Joint Motion Correction and Super Resolution for Cardiac Segmentation via Latent Optimisation," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2021*, M. de Bruijne, P. C. Cattin, S. Cotin, N. Padoy, S. Speidel, Y. Zheng, and C. Essert, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2021, pp. 14–24.

[73] J. Pan, D. Rueckert, T. Küstner, and K. Hammernik, "Learning-Based and Unrolled Motion-Compensated Reconstruction for Cardiac MR CINE Imaging," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 686–696.

[74] J. Xu, D. Moyer, P. E. Grant, P. Golland, J. E. Iglesias, and E. Adalsteinsson, "SVoRT: Iterative Transformer for Slice-to-Volume Registration in Fetal Brain MRI," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 3–13.

[75] G. Forney, “The Viterbi Algorithm,” *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[76] 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, Aug. 1, 2006.

[77] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image Quality Assessment: From Error Visibility to Structural Similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[78] N. D. Cahill, J. A. Noble, and D. J. Hawkes, “A Demons Algorithm for Image Registration with Locally Adaptive Regularization,” in *Med. Image Comput. Comput.-Assist. Interv. – MICCAI 2009*, G.-Z. Yang, D. Hawkes, D. Rueckert, A. Noble, and C. Taylor, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2009, pp. 574–581.

[79] V. Popovic, K. Seyid, Ö. Cogal, A. Akin, and Y. Leblebici, “Real-Time Image Registration Via Optical Flow Calculation,” in *Design and Implementation of Real-Time Multi-Sensor Vision Systems*, V. Popovic, K. Seyid, Ö. Cogal, A. Akin, and Y. Leblebici, Eds., Cham: Springer International Publishing, 2017, pp. 199–224.

[80] M. Maus, “Lernen einer niedrigdimensionalen Repräsentation von Bewegungsvektorfeldern mit neuronalen Netzwerken,” M.S. thesis, Universität zu Lübeck, Lübeck, 2019.

[81] G. Balakrishnan, A. Zhao, M. Sabuncu, J. Guttag, and A. Dalca, “VoxelMorph: A Learning Framework for Deformable Medical Image Registration,” *IEEE Transactions on Medical Imaging*, Feb. 4, 2019.

[82] A. V. Dalca, G. Balakrishnan, J. Guttag, and M. R. Sabuncu, “Unsupervised Learning of Probabilistic Diffeomorphic Registration for Images and Surfaces,” *Medical Image Analysis*, vol. 57, pp. 226–236, Oct. 1, 2019.

[83] M. Hoffmann, B. Billot, D. Greve, J. Iglesias, B. Fischl, and A. Dalca, “SynthMorph: Learning Contrast-Invariant Registration Without Acquired Images,” *IEEE Transactions on Medical Imaging*, Sep. 29, 2021.

[84] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” version 2, *arXiv:2010.11929*, 2021.

[85] Y. Zhu and S. Lu, “Swin-VoxelMorph: A Symmetric Unsupervised Learning Model for Deformable Medical Image Registration Using Swin Transformer,” in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 78–87.

[86] Z. Wang and H. Delingette, “Attention for Image Registration (AiR): An Unsupervised Transformer Approach,” *arXiv:2105.02282*, 2023.

[87] J. Chen, D. Lu, Y. Zhang, D. Wei, M. Ning, X. Shi, Z. Xu, and Y. Zheng, "Deformer: Towards Displacement Field Learning for Unsupervised Medical Image Registration," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2022*, L. Wang, Q. Dou, P. T. Fletcher, S. Speidel, and S. Li, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2022, pp. 141–151.

[88] Y. Hu, M. Modat, E. Gibson, *et al.*, "Weakly-Supervised Convolutional Neural Networks for Multimodal Image Registration," *Med Image Anal*, vol. 49, pp. 1–13, Oct. 1, 2018.

[89] Z. Xu and M. Niethammer, "DeepAtlas: Joint Semi-supervised Learning of Image Registration and Segmentation," in *Med. Image Comput. Comput. Assist. Interv. – MICCAI 2019*, D. Shen, T. Liu, T. M. Peters, L. H. Staib, C. Essert, S. Zhou, P.-T. Yap, and A. Khan, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2019, pp. 420–429.

[90] A. Hering, S. Kuckertz, S. Heldmann, and M. P. Heinrich, "Enhancing Label-Driven Deep Deformable Image Registration with Local Distance Metrics for State-of-the-Art Cardiac Motion Tracking," in *Bildverarb. Für Med. 2019*, H. Handels, T. M. Deserno, A. Maier, K. H. Maier-Hein, C. Palm, and T. Tolxdorff, Eds., ser. Informatik aktuell, Wiesbaden: Springer Fachmedien, 2019, pp. 309–314.

[91] A. Hering, S. Häger, J. Moltz, N. Lessmann, S. Heldmann, and B. van Ginneken, "CNN-based lung CT registration with multiple anatomical constraints," *Med Image Anal*, vol. 72, p. 102 139, Aug. 2021.

[92] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Med. Image Comput. Comput.-Assist. Interv. – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2015, pp. 234–241.

[93] S. Apostolopoulos, S. De Zanet, C. Ciller, S. Wolf, and R. Sznitman, "Pathological OCT Retinal Layer Segmentation Using Branch Residual U-Shape Networks," in *Med. Image Comput. Comput. Assist. Interv. - MICCAI 2017*, M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, and S. Duchesne, Eds., ser. Lecture Notes in Computer Science, Cham: Springer International Publishing, 2017, pp. 294–301.

[94] Z. Gu, J. Cheng, H. Fu, K. Zhou, H. Hao, Y. Zhao, T. Zhang, S. Gao, and J. Liu, "CE-Net: Context Encoder Network for 2D Medical Image Segmentation," *IEEE Trans. Med. Imaging*, vol. 38, no. 10, pp. 2281–2292, Oct. 2019.

[95] H. Cao, Y. Wang, J. Chen, D. Jiang, X. Zhang, Q. Tian, and M. Wang, "Swin-Unet: Unet-Like Pure Transformer for Medical Image Segmentation," in *Computer Vision – ECCV 2022 Workshops*, L. Karlinsky, T. Michaeli, and K. Nishino, Eds., vol. 13803, Cham: Springer Nature Switzerland, 2023, pp. 205–218.

[96] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. Yuille, and Y. Zhou, "TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation," *arXiv:2102.04306*, Feb. 8, 2021.

[97] K. Wu, H. Peng, M. Chen, J. Fu, and H. Chao, "Rethinking and Improving Relative Position Encoding for Vision Transformer," in *2021 IEEECVF Int. Conf. Comput. Vis. ICCV*, Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2021, pp. 10 013–10 021.

[98] R. Xu, X. Wang, K. Chen, B. Zhou, and C. C. Loy, "Positional Encoding as Spatial Inductive Bias in GANs," in *2021 IEEECVF Conf. Comput. Vis. Pattern Recognit. CVPR*, Jun. 2021, pp. 13 564–13 573.

[99] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski, "An Intriguing Failing of Convolutional Neural Networks and the CoordConv Solution," in *Adv. Neural Inf. Process. Syst.*, vol. 31, Curran Associates, Inc., 2018.

[100] C. A. Miller, P. Jordan, A. Borg, R. Argyle, D. Clark, K. Pearce, and M. Schmitt, "Quantification of Left Ventricular Indices from SSFP Cine Imaging: Impact of Real-World Variability in Analysis Methodology and Utility of Geometric Modeling," *J Magn Reson Imaging*, vol. 37, no. 5, pp. 1213–1222, May 2013.

[101] J. Krebs, H. Delingette, B. Mailhé, N. Ayache, and T. Mansi, "Learning a Probabilistic Model for Diffeomorphic Registration," *IEEE Trans Med. Imaging*, vol. 38, no. 9, pp. 2165–2176, 2019.

[102] J. Lu, R. Jin, M. Wang, E. Song, and G. Ma, "A Bidirectional Registration Neural Network for Cardiac Motion Tracking Using Cine Mri Images," *Computers in Biology and Medicine*, vol. 160, p. 107 001, Jun. 1, 2023.

[103] P. Simard, D. Steinkraus, and J. Platt, "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis," in *Seventh Int. Conf. Doc. Anal. Recognit. 2003 Proc.*, Aug. 2003, pp. 958–963.

[104] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss Functions for Image Restoration With Neural Networks," *IEEE Trans. Comput. Imaging*, vol. 3, no. 1, pp. 47–57, Mar. 2017.

[105] Z. Wang and A. Bovik, "A Universal Image Quality Index," *IEEE Signal Process. Lett.*, vol. 9, no. 3, pp. 81–84, Mar. 2002.

[106] J. P. Thirion, "Image matching as a diffusion process: An analogy with Maxwell's demons," *Medical Image Analysis*, vol. 2, no. 3, pp. 243–260, Sep. 1, 1998.

[107] J. Ashburner, "A Fast Diffeomorphic Image Registration Algorithm," *Neuroimage*, vol. 38, no. 1, pp. 95–113, Oct. 15, 2007.

[108] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache, "Diffeomorphic Demons: Efficient Non-Parametric Image Registration," *NeuroImage*, Mathematics in Brain Imaging, vol. 45, S61–S72, 1, Supplement 1 Mar. 1, 2009.

[109] H.-M. Zhang and B. Dong, “A Review on Deep Learning in Medical Image Reconstruction,” *J. Oper. Res. Soc. China*, vol. 8, no. 2, pp. 311–340, Jun. 1, 2020.

Acknowledgements

First of all, I want to thank my supervisor, Prof. Alfred Mertins for his insightful guidance during my time at the Institute for Signal Processing. His ideas and support not only made this work possible, but my master's thesis as well, and I am grateful for the inspiring discussions shared with him.

Also, I'd like to thank my former colleagues at the Institute for Signal Processing. Their support and the inspiration has benefited my research greatly, and i am grateful for having them accompany me throughout my time at the institute.

Furthermore, I'd like to thank my entire family for their endless support, not only in the challenging phases of my academic career but during my entire life. Thank you for making me into the person I am.

I am especially thankful for my partner, for her immeasurable amount of support, both in my personal life and in managing the stresses of deadlines and writing this thesis. I am extraordinarily lucky to have you in my life.