

# Aligning Images

Deformable registration of CT and Pseudo-CT images using unsupervised Deep Learning based algorithms

---

Joel Honkamaa

# Aligning images

Deformable registration of CT and Pseudo-CT images using unsupervised Deep Learning based algorithms

**Joel Honkamaa**

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.  
Otaniemi, 22 Feb 2021

Supervisor: Ph.D. Riku Linna  
Advisor: Ph.D. Julius Koskela

**Aalto University**  
**School of Science**  
**Master's Programme in Life Science Technologies**

**Author**

Joel Honkamaa

**Title**

Aligning images: Deformable registration of CT and Pseudo-CT images using unsupervised Deep Learning based algorithms

**School** School of Science**Master's programme** Life Science Technologies**Major** Complex Systems**Code** SCI3060**Supervisor** Ph.D. Riku Linna**Advisor** Ph.D. Julius Koskela**Level** Master's thesis**Date** 22 Feb 2021**Pages** 67**Language** English**Abstract**

Medical image registration is the process of algorithmically aligning medical images anatomically. Deformable registration seeks to find a nonlinear mapping between anatomic locations of different images. In the past few years Deep Learning has been successfully applied to the problem. In this work unsupervised Deep Learning based deformable registration is considered. The developed methodologies are applied to registration of computer tomography (CT) and artificial CT images created from magnetic resonance images.

Regularization of the predicted mappings is an important aspect of deformable registration and diffeomorphisms, differentiable bijections with differentiable inverse, are often sought after. Very recently diffeomorphic registration frameworks have been applied to unsupervised Deep Learning registration and they have been shown to produce good results with very small run time.

A common limitation in Deep Learning is the ability of a model to fit into a GPU memory. As a result patch-wise approaches, where only sub-volumes of the whole data set are fed to the neural network at once, are often employed. The approach introduces several problems unique for registration, especially ones related to the regularization of transformations.

In this work an original framework for unsupervised Deep Learning registration based on image patches is introduced. The framework is shown to produce diffeomorphic transformations. Accuracy of the registrations is evaluated against a baseline and the method is shown to produce comparative results.

**Keywords** deformable registration, medical image registration, unsupervised Deep Learning, convolutional neural networks

**Tekijä**

Joel Honkamaa

**Työn nimi**

Kuvien sovittamisesta yhteen: CT ja pseudo-CT kuvien deformatiivinen rekisteröinti ohjaamattoman syväoppimisen menetelmin

**Korkeakoulu** Perustieteiden korkeakoulu**Maisteriohjelma** Life Science Technologies**Pääaine** Complex Systems**Koodi** SCI3060**Valvoja** lehtori F.T. Riku Linna**Ohjaaja** Tk.T. Julius Koskela**Työn laji** Diplomityö**Päiväys** 22.2.2021**Sivuja** 67**Kieli** englanti**Tiivistelmä**

Lääketieteellisten kuvien rekisteröinnissä sovitetaan lääketieteellisiä kuvia yhteen anatomiaan perustuen. Deformatiivisessa rekisteröinnissä pyritään löytämään epälineaarinen kuvaus anatomisesti vastaavien sijaintien välille kuvien koordinaattijärjestelmistä toiseen. Muutaman viime vuoden aikana syväoppimisen menetelmillä on saavutettu hyviä tuloksia deformatiivisessa rekisteröinnissä. Tässä työssä pyritään kehittämään menetelmä tietokonetomografiakuvien ja pseudo-tietokonetomografiakuvien, jotka on luotu magneettiresonanssikuvista, deformatiiviseen rekisteröintiin.

Ennustettujen kuvausten onnistunut regularisaatio on isossa osassa rekisteröinnissä. Usein pyritään löytämään kuvauksia, jotka ovat niin kutsuttuja diffeomorfismeja: derivoituvia kääntyviä kuvauksia, joiden käänteiskuvaus on myös derivoituva. Viime aikoina on onnistuttu hyödyntämään klassisesta rekisteröinnistä tuttuja menetelmiä diffeomorfisten kuvausten ennustamiseksi myös syväoppimisen avulla. Tulokset ovat olleet erittäin lupaavia.

Usein vastaan tuleva rajoite syväoppimisessa on, että mallin tulee mahtua näytönohjaimen muistiin. Tämän rajoituksen kiertämiseksi voidaan verkolle syöttää vain osia datajoukosta kerrallaan kokonaisen datajoukon sijaan. Rekisteröinnissä tämä lähestyminen kuitenkin aiheuttaa tiettyjä vain rekisteröinnille ominaisia ongelmia erityisesti kuvausten regularisaatioon liittyen.

Tässä työssä rakennetaan uusi rekisteröintialgoritmirunko osavoluumeihin perustuvaan ohjaamattomaan syväoppimiseen. Menetelmän osoitetaan tuottavan diffeomorfisia kuvauksia ja tuottavan vertailukelpoisia tuloksia suhteessa kirjallisuudesta otettuun verrokkiin.

**Avainsanat** deformatiivinen rekisteröinti, lääketieteellisten kuvien rekisteröinti, ohjaamaton syväoppiminen, konvoluutioneuroverkot

# Acknowledgements

I am very grateful for my advisor Julius Koskela for the very useful comments and the always encouraging spirit throughout the process. Thank you for my supervisor Riku Linna for good support, and especially for working hard to provide the very valuable feedback in time.

Furthermore, I would like to thank my lovely wife for the very important support and encouragement over the process.

Above all, I would like to thank the merciful God for giving me an opportunity to study such an interesting world.

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Tiivistelmä</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>iv</b>
<b>Contents</b>	<b>v</b>
<b>1. Introduction</b>	<b>1</b>
<b>2. Background and Theory</b>	<b>4</b>
2.1 Basics of Neural Networks . . . . .	6
2.1.1 Network Architectures . . . . .	8
2.2 Transformation Representations . . . . .	11
2.2.1 Displacement Field . . . . .	12
2.2.2 Velocity Field . . . . .	13
2.2.3 B-Splines . . . . .	23
2.3 Similarity Metrics . . . . .	24
2.4 Unsupervised Registration Using Deep Learning . . . . .	26
2.4.1 Network Training . . . . .	26
2.4.2 Transformation Representations and Inverse Con-	
sistency . . . . .	27
2.4.3 Patch-wise Registration . . . . .	29
<b>3. Methods</b>	<b>31</b>
3.1 Problem Characteristics . . . . .	31
3.2 Algorithm Overview . . . . .	32
3.3 Algorithm Description . . . . .	35
3.3.1 Practical Implementation . . . . .	36
<b>4. Experiments</b>	<b>43</b>

4.1	Evaluation Metrics . . . . .	44
4.2	Patch-wise Approach Evaluation . . . . .	45
4.3	Baseline Method . . . . .	45
4.4	Training and Prediction Details . . . . .	46
<b>5.</b>	<b>Results</b>	<b>47</b>
5.1	Training . . . . .	48
5.2	Prediction Runtime . . . . .	48
5.3	Metrics Results . . . . .	49
5.3.1	Transformation Regularity . . . . .	50
5.3.2	Generic Similarity Metrics . . . . .	50
5.3.3	Tissue Mask Metrics . . . . .	51
5.4	Patch-wise Approach Consistency . . . . .	52
5.5	Visual Evaluation . . . . .	53
<b>6.</b>	<b>Discussion and Conclusions</b>	<b>56</b>
<b>7.</b>	<b>Summary</b>	<b>58</b>

# 1. Introduction

Medical imaging is important part of modern clinical practice and often arising problem related to it is aligning different medical images with each other. This procedure is called registration. In general terms, registration of images is needed whenever one wants to combine information from two images in one task. For example, if a patient has been imaged using computational tomography (CT) and magnetic resonance imaging (MRI), these images contain different information about the patient. In order to use the both images together, for example in radiotherapy dose planning, the images need to be registered. When images to be registered are acquired from a single patient, the method is called intra-subject registration and when they are from different patients, the inter-subject registration is in question. Further, registration methods can be divided based on whether they are unimodal or multimodal, that is, whether the method is registering images of the same or different modality. In the context of registration, also different MRI constrasts are considered different modalities.

Registration is a task in which different image volumes are brought into single coordinate system under the assumption that the content of the images can be aligned spatially. In principle, one can register simultaneously any amount of image volumes but in the most common case two images are brought into single coordinate system. Registration methods can have different degrees of freedom. In simple rigid registration, volumes are aligned by rotation and translations. Affine transformations preserve straight lines. On the other hand, deformable registration refers to aligning images using nonlinear transformations. This work concentrates on unimodal deformable registration of rigidly preregistered volumes.

Different registration applications face different problems. Multimodal registration is especially challenging due to the difficulty of quantitatively defining similarity of registered images. Such a quantitative metric describ-

ing image similarity is called similarity metric and should be maximized or minimized when images are perfectly aligned. However, difficulties are not limited to multimodal registration. Depending on anatomy even intra-subject unimodal registration can be challenging because of large internal anatomic changes within a subject, for example due to breathing motion, peristaltic motion, or tumor growth. Nonetheless, in unimodal registration one can often define similarity of registered images quite straightforwardly which makes the problem significantly easier.

In general terms, the task is to find a mapping between coordinate systems of different volumes. Anatomical constraints set limitations for the set of possible mappings and thus constraining the transformation to some desired subset of transformations is an important aspect of a registration algorithm. Generally diffeomorphisms, differentiable bijections with differentiable inverse, are considered to be ideal especially when doing intra-subject registration which is the topic of this study.

Significant amount of research has been conducted on different registration methods during the past decades. First deformable medical image registration algorithms are already from the beginning of 80's by Broit, Bajcsy et al. [1–4]. Vastness of the research is hinted at the 1996 review article by Maintz et al. [5]. Different methods work well in different settings and comparison of registration methods can be extremely difficult due to lack of "correct registrations" or ground truths [6]. State-of-the-art classical registration methods include Large Diffeomorphic Distance Metric Mapping (LDDMM) [4], diffeomorphic Demons [7], DARTEL [8], and symmetric normalization (SyN) [9], all of which constrain the transformation to be diffeomorphic [10]. Despite the challenges, some systematic comparisons of different registration algorithms have been made [11, 12].

Recent advances in Deep Learning have opened way to new registration methods. Deep Learning based registration methods have shown promising results and are already outperforming classical registration methods in some aspects. Deep Learning can be applied to image registration in quite numerous ways. It can be used to deform an image directly or to learn some parametric representation of the deformation. Methods can be supervised or unsupervised. In the latter setting some similarity metric is usually used as a loss function. Attempts have been also made in learning similarity metrics using Deep Learning. Generative adversarial networks (GAN) are also showcasing promising results. One of the main possible benefits of using Deep Learning is faster registration as it enables an

algorithm to learn to predict a direct transformation instead of obtaining the transformation using an iterative process. [13]

In this work, registration of pseudo-CT images to real CT images is looked into. A pseudo-CT image refers to a CT-like image which has been produced artificially from a MRI image. Pseudo-CT images have also been produced using classical methods but Deep Learning is the state-of-the-art methodology on this area as well. Producing pseudo-CT images is not the topic of this work.

Being able to register CT and pseudo-CT images quickly and with good accuracy would open numerous possibilities, since they make it possible to convert multimodal registration problems into unimodal ones. Such a workflow has been used very recently by Fu et al. for registering CT and MR images [14, 15]. The main difference in this work is the use of diffeomorphic registration frameworks and patch-wise registration. We do not evaluate the multimodal registration but instead focus on the quality of the registration between CT and pseudo-CT images.

We will start by a quick survey into basics of Deep Learning, after which we will look into classical registration methods. Due to the vastness of the research, the survey is not comprehensive. Only methods with possible use in Deep Learning and this particular problem are looked into with the focus on how the registration fields are encoded. Exact nature of the iterative algorithms is not in our interest here as the purpose is not to develop such algorithms. The survey into classical methods is followed by a survey into recent advances in registration using Deep Learning. The focus will be on unsupervised methods. In Chapter 3 chosen methods are explained with rationale to selecting them. In Chapter 4 we look at the experiment setup and define the evaluation methods. This is followed by Chapter 5 where results are considered.

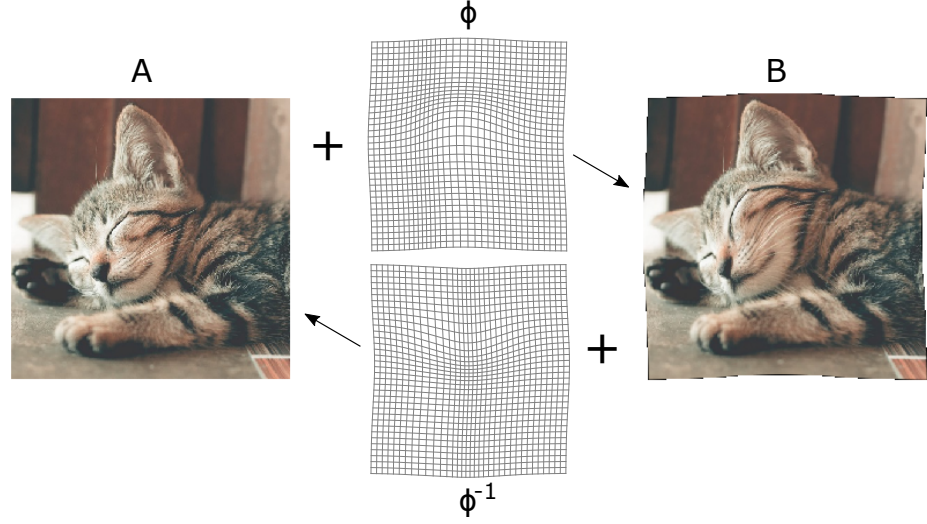
## 2. Background and Theory

In this work medical image registration is always considered between exactly two (and no more) images. An image will refer to a mapping  $I : \mathbb{R}^3 \rightarrow \mathbb{R}$  from image domain to intensity values. In practice one only has samples of the image domain, the so called voxels, and the rest is inferred by interpolation. Given two images  $I_1$  and  $I_2$ , registration of  $I_1$  to  $I_2$  is then the task of finding a mapping, also referred to as a transformation,  $\phi^{-1} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$  which maps anatomic locations of the image  $I_2$  to anatomic locations of the image  $I_1$ . This is demonstrated in Figure 2.1. Then  $I_1 \circ \phi^{-1} \approx I_2$  should hold where exact interpretation of  $\approx$  depends on the application (unimodal or multimodal).

An inverse mapping is sought after, since one wants to sample the transformed image in the target space. That is, given samples of  $\phi^{-1}$  at voxel locations we can interpolate  $I_1$  at the locations defined by the  $\phi^{-1}$  to obtain the transformed  $I_1$ .

There are multiple ways to classify image registration algorithms but the most common approach is to see the registration as an optimization problem over some space of transformations. In this approach registration algorithm consists of four components: the space of transformations, the similarity measure, the regularization of the transformations, and the optimizer. The task is then to minimize the loss function which is sum of the similarity measure and the regularization term using the optimizer over the space of transformations. [16]

If the output of the registration is simply a mapping from one domain to another, as described above, the similarity metric will be simply some function working on  $I_1 \circ \phi^{-1}$  and  $I_2$ . In the case of intra-subject unimodal registration this could be for example mean squared error  $\|I_1 \circ \phi^{-1} - I_2\|_{L^2}^2$ . However, if a registration method takes equally into account both images, this formulation of similarity metrics is not enough. In the case



**Figure 2.1.** Registration is a task of finding a coordinate transformation connecting two images. In the figure the transformations  $\phi$  and  $\phi^{-1}$  are the respective forward and backward transformations between the images A and B.

of symmetric image normalization (SyN)[9], the similarity metric will not be computed in the coordinate system of either of the images but instead in some intermediate coordinate system. The result is not only the mapping  $\phi^{-1}$  but also the intermediate steps between the coordinate systems. Because of that, for generic formulation, we will talk about generic transformation  $T$  belonging to generic set of transformations  $\mathcal{T}$  between the images with which the similarity metric is calculated. Let us denote the similarity metric by  $\mathcal{E}_s(I_1, I_2; T)$ .

The regularization term is the component of the loss function which depends only on the transformation. It usually imposes some constraints on the transformation such as smoothness. Correct transformation can not be inferred for every image position and the ambiguous areas are determined by the used regularization. Regularization terms are usually related to the second order derivatives of the transformation [16]. Typical examples include membrane energy, Laplacians, bending energy, and linear-elastic energy [17]. The loss function is then the trade-off between the similarity of the structures and the regularization of the transformation. [16] Let us denote the regularization term with  $\mathcal{E}_r(T)$ .

Given two images  $I_1$  and  $I_2$ , the registration task is then simply

$$\arg \min_{T \in \mathcal{T}} \mathcal{E}_s(I_1, I_2, T) + \mathcal{E}_r(T) \quad (2.1)$$

Deep Learning can be utilized in multiple ways for problems related to image registration. The most straightforward is prediction of the transformation. Others include using neural network as a similarity metric or for

registration quality evaluation. [13]

When using Deep Learning for transformation prediction, there are two basic approaches: supervised and unsupervised. In a supervised setting ground truth transformations are needed. The network is then trained to predict these transformations given the images to be registered. The unsupervised setting is usually very close to the problem described in Equation (2.1). The only difference to classical methods is that instead of optimizing the transformation iteratively, it is predicted directly. Basically, the problem is to find a function that predicts the transformation in one shot given the input images. However, the function that produces the transformation, i.e. the neural network, is optimized iteratively. [13] In this work we will focus on unsupervised Deep Learning registration methods.

For a Deep Learning method to learn the transformation directly we need to have some representation for it. Furthermore, some similarity metric and regularization terms are usually needed as a loss function of the Deep Learning method. Thus the classical registration literature plays an important role also in developing Deep Learning registration algorithms.

In this chapter we will first look at basics of neural networks. After that registration is viewed from the classical viewpoint, especially from two perspectives: transformation representations and the used similarity metrics. After the review of classical methods different existing unsupervised Deep Learning registration methods are considered.

## 2.1 Basics of Neural Networks

In this section, a short introduction to neural networks and Deep Learning is given. In addition to basics, some network architectures suitable for registration are looked at. The purpose of this section is to help the reader less familiar with Deep Learning to understand the thesis.

In essence, Deep Learning concerns fitting functions with very large number of parameters to data. In Deep Learning the functions are usually called neural networks due to inspiration from biological neural networks for how these functions are defined. The term network architecture is often used when referring to the structure of the chosen neural network.

The process of fitting a neural network to data is called training a network. For doing this, one needs a set of inputs and a way to define how desired the produced outputs are. The measure of how desired an output

is, is called a loss function which is some scalar function of the outputs. In the common setting, the target is to find parameters that minimize the loss function for given inputs. If some known desired outputs are available, loss function could be some difference measure between produced outputs and known outputs. The known outputs are often called ground truths. Updating weights based on a loss function is usually done using some sort of a gradient update scheme.

To be more explicit, let us define a neural network as  $f_w : \mathbb{R}^n \rightarrow \mathbb{R}^m$  parametrized by weights  $w \in \mathbb{R}^l$ . Then we have a set of training inputs  $X = (x_1, \dots, x_N)$  where  $x_i \in \mathbb{R}^n$ . Given the inputs we can calculate the set of outputs  $(f_w(x_1), \dots, f_w(x_N))$ . It then remains to define a loss function. Loss functions are usually independent for each output sample and also different for each sample, for example due to having specific ground truth output for each training input. We can thus define set of loss functions  $\mathcal{E}_i : \mathbb{R}^m \rightarrow \mathbb{R}$ , where  $i \in \{1, \dots, N\}$ . The target loss function is then usually defined as the sum of losses over the outputs  $\mathcal{E} : \mathbb{R}^{N \times m} \rightarrow \mathbb{R}$ ,

$$\mathcal{E}(y_1, \dots, y_N) := \sum_{i=1}^N \mathcal{E}_i(y_i)$$

Then in order to update the weights  $w$  based on the loss function it remains to calculate the total derivative

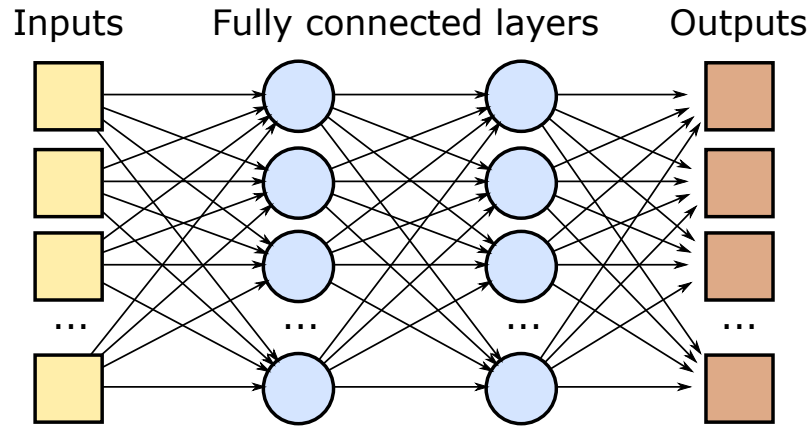
$$\frac{d}{dw} \mathcal{E}(f_w(x_1), \dots, f_w(x_N)).$$

Calculation of the derivative can be done using back-propagation algorithm proposed originally by Rumelhart et al. (1986) [18]. Back-propagation is basically applying the chain rule of differentiation repeatedly.

Typically, the number of samples  $N$  is so large that one cannot calculate the derivative for all of them at once and hence only a subset of samples are used at once for each update. For updating the weights based on the derivative some advanced optimization scheme is usually employed, for example Adam by Kingma et al. (2014)[19].

In order to avoid over-fitting, dropout strategy is often employed. In the simplest terms dropout refers to updating only part of the weights during each iteration and leaving some of them unchanged. [20]

Once the network weights have been trained, the function  $f_w$  can be used to predict an output for new samples which have not been used in the training. In this work we refer to it as a prediction phase.



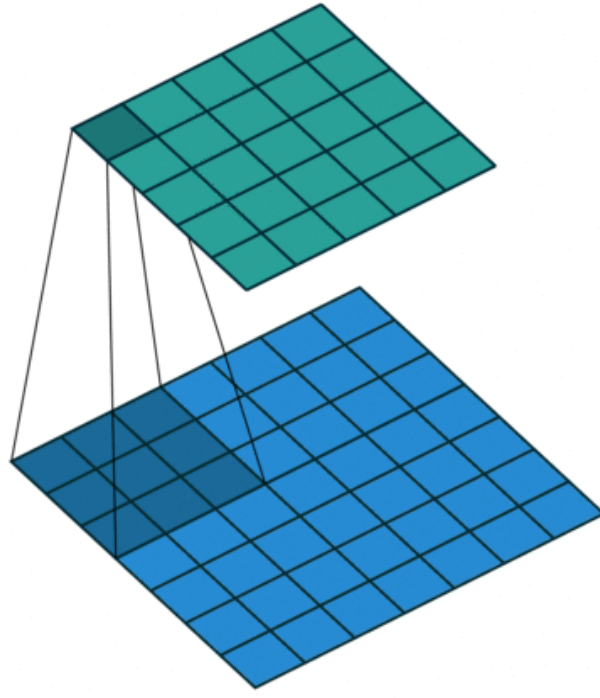
**Figure 2.2.** Basic architecture of a fully connected neural network. Yellow boxes represent an input with each rectangle being one scalar. The blue circles in the middle represent the computational nodes of the network. Output is represented by the rectangles. The number of inputs, outputs, and nodes in a fully connected layer can differ. Each node output is a linear combination of the inputs added with some constant and combined with a nonlinearity.

### 2.1.1 Network Architectures

Given the basic scheme, there remains the question of how to define a neural network function. A chosen architecture has significant impact on the results and different applications call for different architectures.

The most basic neural network design is the fully connected neural network. The core idea of the fully connected neural networks is shown in Figure 2.2. Fully connected neural networks consist of series of layers consisting of nodes connected to all of the nodes of a previous layer. Each node performs a basic linear calculation dependent on the weights followed by a unit performing nonlinear computation. Nonlinearity is important since otherwise the whole neural network could be represented by a single matrix. Commonly used nonlinearity is the rectified linear unit (ReLU) that gives zero output for input values less than zero and else performs identity mapping.

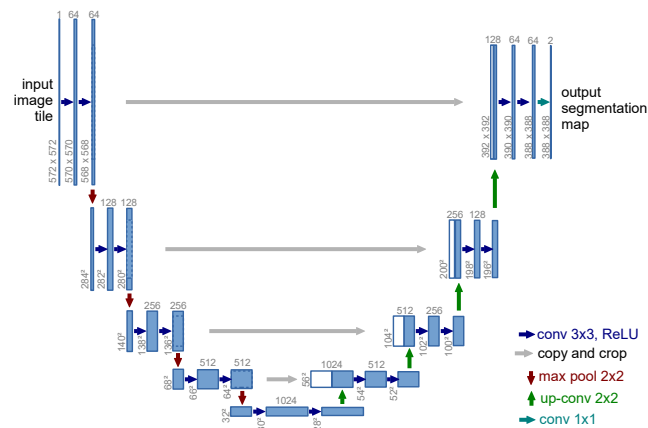
One of the most popular and successful branches of network architectures in the recent years has been convolutional neural networks [21]. The basic idea is that in convolutional layers different features are extracted from an input using different convolution filters. The filters are defined by kernels that are learned during the training and thus the network can learn which features are relevant for the task. The basic idea for one convolution filter is represented in Figure 2.3. One convolution layer would have multiple such filters acting on the input and that would result in different spatial feature representations of the data. In addition some nonlinearity such as ReLU is again applied to the output. Convolutions are spatially invariant,



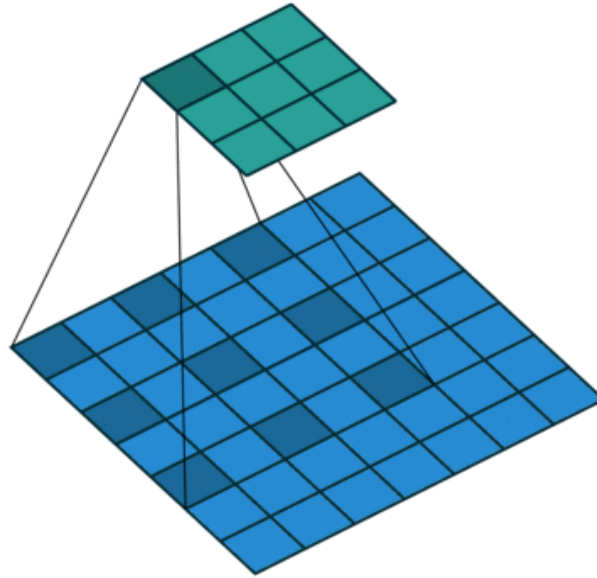
**Figure 2.3.** Basic idea of the convolutional filter in neural networks. In convolutional layers convolutional filters are applied to the input samples along the dimensions. In this example the data is two dimensional and the size of the filter size is  $3 \times 3$ . Each new value is a linear combination of 9 values of the original input. The image is from a paper by Dumoulin et al. (2016) [22].

which is often a desired property. For an introduction to basic building blocks and arithmetics of convolutional neural networks, see the paper by Dumoulin et al. (2016) [22].

Commonly multiple convolutional layers are cascaded with some down-sampling of the input in between the layers. Down-sampling can be done for example using max pooling operation which takes the largest value in each window. By consecutive convolutional layers and down-sampling one obtains a more accurate feature representation of the input while



**Figure 2.4.** Network architecture for U-Net from the original paper by Ronneberger et al. [23].



**Figure 2.5.** Dilated convolution filter with dilation rate of 1. The image is from a paper by Dumoulin et al. (2016) [22].

reducing spatial accuracy. However, for example in image segmentation or registration we need to have same spatial accuracy for inputs and outputs.

To get back to the original resolution, one can up-sample the data again from the feature representation. Up-sampling can be achieved using reverse convolutions. In reverse convolutions one input value affects many output values instead of the opposite for regular convolutions. There are multiple heuristics for implementing this basic idea, probably the simplest approach being to repeat each value a desired number of times and then apply a regular convolution to the up-sampled input.

Simple up-sampling from the spatially less accurate feature representation would still result in low spatial accuracy for the output. A very popular architecture that was originally successfully applied to medical image segmentation called U-Net developed by Ronneberger et al. (2015) solves this by so called residual connections [23]. In U-Net earlier feature representations from down-sampling stage are concatenated to the input during up-sampling stage. This way the final output can have both a good feature representation and a good spatial accuracy for the features. The original U-Net architecture can be seen in Figure 2.4. U-Net type architectures are also among the most popular architectures for registration [13].

Receptive field refers in neural networks to the radius of input values affecting a single output value. Receptive field of U-Net depends on the number of down-sampling steps and the size of the convolution kernels. In

order to increase receptive field, dilated convolutions can be used at low cost. In dilated convolutions, there are gaps between the values which the kernel reads. The basic idea is shown in Figure 2.5.

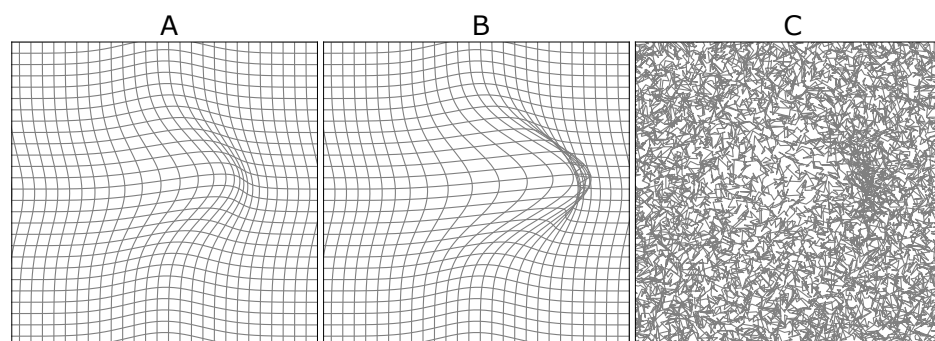
## 2.2 Transformation Representations

Figure 2.6 shows three different mappings from  $\mathbb{R}^2$  to  $\mathbb{R}^2$ . However, only the one on the left is anatomically feasible (assuming two dimensional tissue) since the other transformations include folding. Given the example, it is evident that the desired space of possible coordinate transformations is smaller than even the set of all spatially continuous mappings from an image domain to another image domain.

The question of transformation representation is essential as one needs to define the space of transformations over which to optimize the registration problem given in Equation (2.1). One way to obtain transformations with desired properties is by using a regularization term. However, even more desired outcome is if one can limit the space of transformations altogether to some more desirable subset of transformations.

A very commonly desired property is that the transformation should be diffeomorphic, that is, bijective with differentiable inverse. Diffeomorphisms are sought after especially in the case of intra-subject registration as there should be one-to-one correspondence between each anatomic location. In addition diffeomorphisms preserve connected and disjoint sets and smoothness of curves. Diffeomorphisms are thus a very natural choice for anatomical registration tasks. [4]

Another relevant concept is inverse consistency introduced by Chris-



**Figure 2.6.** The figure visualizes different types of transformations from  $\mathbb{R}^2$  to  $\mathbb{R}^2$  using a deformed grid. A: Smooth and invertible mapping. B: Smooth but not invertible mapping. C: Noncontinuous mapping. Only the mapping A has desired properties for anatomical registration as it is both continuous and invertible. Intuitively the desired mappings are such that they can be presented using a grid without the grid folding on top of itself. In the case of noncontinuous mapping the grid representation fails entirely.

tensen et al. (2001) [24]. It refers to the property that predicted transformation from image A to B and B to A should be inverses of each other.

Assuming that the transformation is everywhere differentiable, invertibility of it can be checked easily using Jacobian determinant. Given the mapping  $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^3$  the Jacobian determinant can be calculated as determinant of the Jacobian matrix:

$$\det \left[ \frac{\partial \phi}{\partial x} \right] = \det \begin{bmatrix} \frac{\partial \phi_1}{\partial x_1} & \frac{\partial \phi_1}{\partial x_2} & \frac{\partial \phi_1}{\partial x_3} \\ \frac{\partial \phi_2}{\partial x_1} & \frac{\partial \phi_2}{\partial x_2} & \frac{\partial \phi_2}{\partial x_3} \\ \frac{\partial \phi_3}{\partial x_1} & \frac{\partial \phi_3}{\partial x_2} & \frac{\partial \phi_3}{\partial x_3} \end{bmatrix}$$

Jacobian determinant assigns a scalar value to each point in the space representing the local "scaling" of the volume by the transformation. If it is less than one but larger than zero, the volume is being locally shrunk and if it is greater than one the volume is being locally expanded. However, a Jacobian determinant below value zero indicates folding and thus is undesired in registration context. If the determinant is everywhere positive, the transformation is invertible.

Significant number of different transformation representations have been tried in the past [6, 16, 25].

### 2.2.1 Displacement Field

In practice one has to discretize the transformation from one image domain to another somehow, and displacement fields can be seen as the most straightforward representation for that. For a continuous case, given a transformation  $\phi^{-1}$ , the displacement field is defined as  $u = \phi^{-1} - \text{Id}$ , where Id is an identity mapping. In discrete setting a three-dimensional displacement vector is usually assigned for each voxel. Many early deformable registration works optimize directly over displacement fields [5].

Usefulness of displacement fields results from the fact that one is usually interested in a grid of intensity values of the warped image at the voxel locations of the target image. Since the number of voxels usually stays the same over a registration task, a discrete displacement field gives transformation in the desired set of locations at the target space. To obtain the warped image one would have to interpolate the source image in the locations given by the displacement field.

To be explicit, let us have the voxel locations in the target space  $X = (x_1, \dots, x_n)$  where  $x_i \in \mathbb{R}^3$  for  $i \in 1, \dots, n$ . Now, given the mapping  $\phi^{-1}$  we

obtain a set of points  $Y = (y_1, \dots, y_n)$ , where  $y_i = \phi^{-1}(x_i)$ , representing the points in the source space corresponding to the voxel locations in the target space. To obtain the warped image we have to interpolate the source image in the locations  $Y$ . Now, if we have a discrete displacement field defined at the positions  $X$ , we obtain  $Y$  simply by  $y_i = u(x_i) + x_i$ .

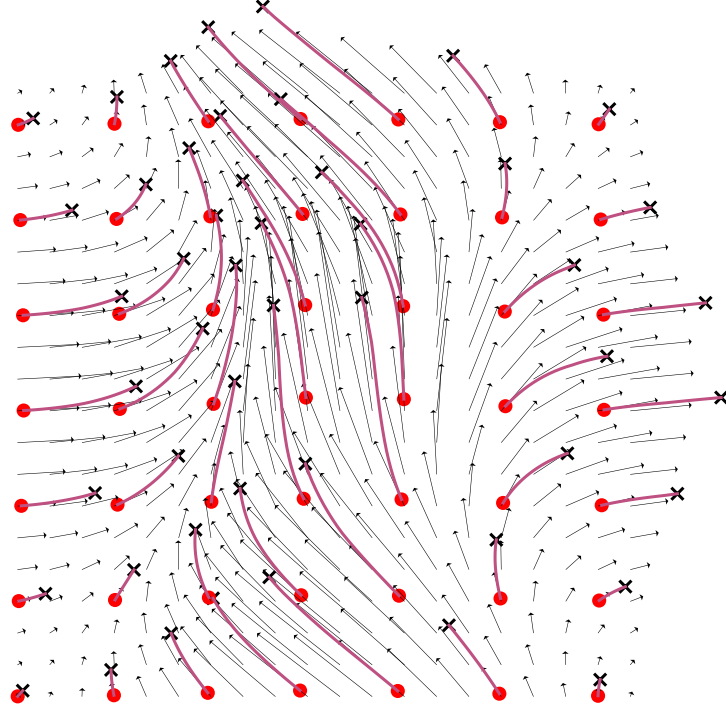
When using some other transformation representation, the representation basically has to be first converted to a displacement field for doing the actual warping of the image. From the point of view of unsupervised Deep Learning this has a clear significance. For unsupervised learning one has to be able to compute the warped image from the predicted transformation in such a way that the gradient of the loss function can be back-propagated through the computations. Many of the transformation representations require solving differential equations before obtaining the displacement field from the representation. Depending on the equation this can be extremely difficult to solve in such a manner that the gradient can be back-propagated and thus the representation would not be well suited for unsupervised Deep Learning.

### 2.2.2 Velocity Field

Displacement fields can describe any coordinate transformation between volumes but it is difficult to regularize the transformations in such a way that the resulting transformation can encode large and accurate displacements and at the same time have some nice properties such as invertibility. Many attempts to provide robust methods for generating diffeomorphic transformations have been developed in the recent, see for example [4, 7–9]. Many of these methods represent the transformation using velocity field with inspiration gotten from fluid dynamics.

The first works using velocity fields for generating diffeomorphic mappings in registration were done by Trounev et al. (1995) and Christensen et al. (1996) [26, 27]. The basis of the work lies in the notion of Lagrangian and Eulerian representations of the flow fields or Lagrangian and Eulerian coordinates. The Lagrangian representation follows some fluid parcel as it flows along the flow field parameterized by the initial position and time. On the other hand, in the Eulerian representation the properties of the fluid are described as it flows through a given static spatial position.

Let us now denote  $\psi(x, t)$  as position of a fluid parcel at time  $t$ , when given an initial position  $x$ . This gives us the Lagrangian representation of the flow. Let us then denote velocity of the parcels flowing through



**Figure 2.7.** The figure visualizes how a transformation is computed from a velocity field. The velocity field is represented by the black arrows and the red dots represent the original locations such as voxels. The locations move independently along the velocity field flow for a fixed time (usually an unit time) and end in the positions marked by the black crosses. The displacement field representing the transformation is the difference between the final positions and the original positions. For clear visualization, here the velocity field flow does not change over time.

position  $x$  at time  $t$  with  $v(x, t)$  which gives us velocities of the fluid parcels in Eulerian coordinates. These coordinate representations are related through the equation

$$\frac{\partial \psi}{\partial t}(x, t) = v(\psi(x, t), t). \quad (2.2)$$

The basic idea is that the transformation can be represented as a velocity field in Eulerian coordinates over unit time  $t \in [0, 1]$ . The transformation is then obtained as the Langrangian representation at time 1 with constraint  $\psi(x, 0) = \text{Id}$ , where  $\text{Id}$  is identity mapping. This idea is demonstrated in Figure 2.7.

If the Eulerian flow is from a moving image to a fixed image, one actually needs to solve the inverse flow instead of the forward flow, since one is interested in sampling the fixed image values in the target space. Here inverse flow refers to the inverse  $\psi^{-1}$  of the Langrangian representation  $\psi$  and gives the position where a fluid parcel located at position  $x$  at time  $t$  was at time 0. The transformation  $\phi^{-1}$  from the target domain to the original domain is then defined as  $\phi^{-1}(x) = \psi^{-1}(x, 1)$ .

The differential equation describing the inverse flow can be obtained

using material derivative. Material derivative is another equation connecting Eulerian and Lagrangian reference frames. In fluid dynamics the material derivative describes rate of change of some physical quantity for some fluid parcel at position given in Eulerian coordinates. Mathematically the quantity can be any differentiable time-varying scalar or vector field defined over the Eulerian coordinate system. In other words, the material derivative is the total time derivative of the given field along the trajectory defined by the flow. In the case of some vector field  $w$  over a Lagrangian flow representation  $\psi(y, t) = x$  we have using multivariate chain rule

$$\frac{dw}{dt}(x, t) = \frac{\partial w}{\partial t}(x, t) + [D_x w](x, t) v(x, t).$$

Here  $D_x w$  is the Jacobian of the  $w$  with respect to  $x$ . One must note that also the Jacobian is a function of  $x$  and  $t$ .

Now one can obtain the equation for the inverse flow. Let us first note that material derivative with respect to  $\psi^{-1}$  is  $\frac{d\psi^{-1}}{dt} = 0$  since the starting position of each fluid parcel must stay constant. As a result one obtains the inverse advection flow equation

$$0 = \frac{\partial \psi^{-1}}{\partial t} + (D\psi^{-1})v. \quad (2.3)$$

In a paper by Dupuis et al. (1998) [28] instead of using Equation (2.3) for solving the inverse at all time instants they simply integrate backwards the velocity field from time 1 to time 0. As done in the paper, this can be formulated for all time instants with the ordinary differential equation

$$\frac{\partial \eta}{\partial s}(x, t; s) = v(\eta(x, t; s), s), \quad (2.4)$$

given boundary condition  $\eta(x, t; t) = \text{Id}$  for all  $t$ . This definition results in identity  $\eta(x, t, 0) = \psi^{-1}(x, t)$ . Note that this formulation is practical for solving  $\psi^{-1}(x, t)$  only for one time instant, which oftentimes is enough.

### *Large Deformations Diffeomorphic Metric Mapping*

The basic concepts presented above give rise to one very popular registration framework with many variants named large deformations diffeomorphic metric mapping (LDDMM). The term was first introduced in a paper by Beg et al. (2005) [4]. An important role in the development was played by Trounev, Miller and Younes [26, 29–32]

The mathematical theory is built on top of concepts from Riemannian

geometry which studies differentiable manifolds with Riemannian metric, i.e. an inner product on a tangent space of a differentiable manifold. Riemannian metric makes it possible to define geometrical concepts such as length of a curve and angle on a manifold. Differentiable manifolds are essentially spaces in which one can do calculus, thus differentiable manifolds always have a tangent space. In this context the space of diffeomorphisms is the differentiable manifold with the space of velocity fields (in Eulerian frame) as the tangent space. Let us denote the tangent space by  $V$ .

Exact formulations of LDDMM differ but usually the tangent space  $V$  is chosen as a Hilbert space equipped with the inner product

$$\langle u, v \rangle_V := \langle Lu, Lv \rangle_{L^2}$$

inducing a Sobolev norm  $\|\cdot\|_V$  where  $L$  is self-adjoint ( $L = L^*$ ), linear and invertible differential operator. The inner product on the tangent space induces a right-invariant Riemannian metric on the manifold. The right-invariance follows here essentially directly from defining the metric in Eulerian frame.

Riemannian metric induces geometrical concepts in the space of diffeomorphisms. Let now  $v_t \in V$ ,  $t \in [0, 1]$  be a path in the space of velocity fields. As discussed earlier, it defines a path  $\psi_t := \psi(\cdot, t)$ ,  $t \in [0, 1]$  in the space of diffeomorphisms according to the differential equation (2.2) given some initial condition  $\psi_0$ . For example, the length of the path  $\psi_t$  is then

$$L(\psi_t) := \int_0^1 \|v_t\|_V dt.$$

Given the Riemannian metric, the notion of straight lines or geodesics also arises. Given the path above, one can define the energy of the path as

$$E(\psi_t) := \frac{1}{2} \int_0^1 \|v_t\|_V^2 dt.$$

Given some start point  $\psi_0$  and end point  $\psi_1$  in the space  $\mathcal{G}$ , there could be multiple possible continuous paths parameterized by  $v$ . The path minimizing the energy functional is the geodesic or straight path between the points. Such a path also minimizes the length functional but minimizing the energy functional provides us with constant speed parametrization. The result is relatively easy to obtain, since by Cauchy-Schwartz we have  $L(\psi_t)^2 \leq 2E(\psi_t)$  and the equality can hold only if  $\|v_t\|_V$  is constant.

Next, let us look at paths  $v_t : [0, 1] \rightarrow V$  in the space of velocity fields with

the initial condition  $v_0 = \text{Id}$  and  $L(v_t) < \infty$ . One of the main contributions of the papers by Trounev Trounev (1995) and Dupuis et al. (1998) [26, 28] was to show that given suitable differential operator, end points  $\psi_1$  defined by all such paths in the space of velocity fields induce a group of diffeomorphisms where geodesic exists between any two elements. Geodesic path length between two points of the space defines a metric on the space.

The induced metric in the space of diffeomorphisms gives rise to the definition of the variational LDDMM optimization problem originally defined by Dupuis et al. (1998) [28]

$$\arg \min_v \|I_1 \circ \psi^{-1}(x, 1) - I_2\|_{L^2}^2 + \int_0^1 \|v_t\|_V^2 dt. \quad (2.5)$$

The formula fits into the basic registration optimization framework as defined earlier Equation (2.1). The first part of the equation is the similarity term and the second part is the regularization term. Theoretical framework of the regularization term gives natural interpretation for the solutions of the optimization problem in terms of geodesics. The problem can be then seen as finding the shortest path in the space of diffeomorphisms connecting the two images according to the similarity term. The solution can be shown to satisfy the Euler-Lagrange equation, which is used by Begs et al.[4] for minimizing the functional.

The used differential operator defines the properties of the transformations. For example, given a suitable differential operator, the energy of the path is larger for less smooth transformations and thus smoothness of the transformation can be enforced.

Since the geodesics represent straight lines in the space of diffeomorphisms, an obvious question is whether one could parameterize the path simply by the initial velocity. This is indeed possible, as was done for example by Miller et al. (2006) [33] and Vialard et al. (2012) [34]. Calculating the transformation from the initial velocity field by following along the geodesic path is called Riemannian exponentiation of that velocity field. However, the equations are unfortunately difficult to work with and are thus not very suitable for Deep Learning purposes.

Another important modification of the LDDMM framework is the symmetric image normalization (SyN) method [9]. In the cases described before, registration requires arbitrary choice about which image is moving and which is not. SyN, however, considers both images equally. Instead of measuring similarity of the two images in the space of either one of the

images, SyN measures the similarity in an intermediate space between the two. This is accomplished by moving both of the images towards each other and the similarity is measured at time 0.5. An additional constraint is added so that the distance from both the images to the in between space is equal in the space of diffeomorphisms. By matching the images in the intermediate space they can also be registered to each other since the transformations provided by the LDDMM framework are invertible. The algorithm produces good results for unimodal registration using cross-correlation similarity measure.

### *Momentum Based Representations*

As defined in the previous section, in the context of LDDMM the regularization term is given as

$$\int_0^1 \|v_t\|_V^2 dt = \int_0^1 \langle Lv_t, Lv_t \rangle dt.$$

Since the differential operator  $L$  is self-adjoint, the regularization term can be also written as

$$\int_0^1 \langle v_t, L^*Lv_t \rangle dt = \int_0^1 \langle v_t, L^2v_t \rangle dt.$$

Let us denote the resulting differential operator as

$$K := L^2. \tag{2.6}$$

Given a velocity field  $v$  in Eulerian reference frame, we can obtain a new vector field

$$m := Kv$$

which is called Eulerian momentum field. The name is derived from its analogy in standard mechanical systems. As a result, the regularization term can now be defined as an inner product between the velocity field and the momentum field [33]

$$\int_0^1 \langle v_t, m_t \rangle dt. \tag{2.7}$$

The operator  $K$  is of our interest here as it allows for the formulation of the framework from another direction. In the original approach by Beg et al. (2005) [4] the operator  $L$  was chosen to be some differential operator, in their case of the form

$$L = \alpha\Delta + \gamma \text{Id}. \tag{2.8}$$

The chosen differential operator induces the operator  $K = L^2$  and since the operator is linear it can be represented as a kernel which we equate with the operator denoting it also as  $K$ . Now, instead of having differential operator which induces a kernel one can begin with the kernel as it is enough to define the regularization term according to Equation (2.7). Usually the kernel is defined by its inverse.

Typically isotropic kernels are used, meaning that the kernel is of the form  $K^{-1}(x, y) = K^{-1}(\|x - y\|_{L^2})$ , although exceptions can be found [36]. Also the operator in Equation (2.8) induces an isotropic kernel [37]. Isotropic kernels depend only on the distance between the arguments and are thus translation and rotation invariant and can be formulated as a convolution. Simple Gaussian kernels are often the choice [38]. Gaussian kernels can be shown to be induced by the limit case  $L = (\alpha \nabla^2 + \gamma \text{Id})^n$  when  $n$  approaches infinity [37].

### *Stationary Velocity Field*

So far we have looked at generating diffeomorphisms using time-varying velocity fields. A naturally arising question is whether a static velocity field representation would be able to encode all the relevant anatomical variation.

An important paper on this topic is by Arsigny et al. (2006) [39] where they provide several useful mathematical observations. The main focus of the paper is on developing metrics on the space of diffeomorphisms but the provided framework has proven to be fruitful in general. The basic idea in the paper is to look at so called one-parameter subgroups of diffeomorphisms. In general, given a group  $G$  with group operation  $\circ$ , the set  $(g(t))_{t \in \mathbb{R}}$  of  $G$  is a one-parameter subgroup of  $G$  if  $g(0)$  is an element of the group, and for all  $t, s \in \mathbb{R}$  it holds that  $g(t + s) = g(t) \circ g(s)$ . They note that for diffeomorphisms that form a group under composition, one-parameter subgroups are given exactly by integrating (smooth) stationary vector fields over time according to Equation (2.2), which simplifies to

$$\frac{\partial \psi}{\partial t}(x, t) = v(x(x, t)). \quad (2.9)$$

In group theory, Lie algebra  $\mathfrak{g}$  of a Lie group  $G$  is defined as the tangent space of the group at the identity. Exponentiation of an element  $X \in \mathfrak{g}$  of the Lie algebra is then defined as a map  $\exp : \mathfrak{g} \rightarrow G$ , so that  $\exp(X) = \gamma_X(1)$ , where  $\gamma_X : \mathbb{R} \rightarrow G$  is the unique one-parameter subgroup of  $G$ , for which the tangent vector at the identity is equal to  $X$ .

In the paper by Arsigny et al. [39] it is noted that this concept can be generalized to the infinite dimensional case of diffeomorphisms and the generalization proposed is the only possible way of doing this. Essentially, the Lie algebra is given by the stationary velocity fields and Lie group is the group of diffeomorphisms. As mentioned earlier, one-parameter subgroups of diffeomorphisms are obtained by integrating stationary velocity fields. Thus, given a stationary velocity field  $v : \mathbb{R}^3 \rightarrow \mathbb{R}^3$  one can define the exponentiation as

$$\exp(v) := \psi(x, 1), \quad (2.10)$$

where  $\psi$  is defined according to Equation (2.9). This exponential would be identical to the Riemannian exponential mentioned earlier if the Riemannian metric was bi-invariant. However, this is not the case in our application and thus the group exponential and the Riemannian exponential differ. However, it is interesting that both the transformations obtained using integration of stationary velocity fields and the transformations obtained as geodesic time-dependent velocity fields can be seen as results of exponentiation of a velocity field.

In the paper by Arsigny et al. the authors go on to define a logarithm in the space of diffeomorphisms based on the definition of the exponentiation. The logarithm can be used to create a metric in the space, which gives theoretical credence for the solution in general. More important for practical purposes is the authors' proposed method of integrating stationary vector fields or, in other words, computing the exponential  $\exp(v)$ . The concept is called scaling and squaring and is inspired by computing matrix exponentials. The basic idea is that the exponential is easy to approximate for values close to zero. It is also clear that given some  $t, s \in \mathbb{R}$ , it holds that

$$\exp((t + s)v) = \exp(tv) \circ \exp(sv).$$

To calculate the exponential one can then first calculate  $\exp(v/2^N)$  for some sufficiently large  $N \in \mathbb{N}$  (scaling step) followed by  $N$  squaring steps.

Squaring is defined simply as  $\exp(v)^2 := \exp(v) \circ \exp(v)$ . Thus we have

$$\begin{aligned}
 \exp(v/2^{N-1}) &= \exp(v/2^N)^2 \\
 \exp(v/2^{N-2}) &= \exp(v/2^{N-1})^2 \\
 &\dots \\
 \exp(v/2) &= \exp(v/2^2)^2 \\
 \exp(v) &= \exp(v/2)^2.
 \end{aligned} \tag{2.11}$$

Only  $N$  compositions are needed. In the case of diffeomorphisms, if  $N$  is large enough one simply has  $\exp(v/2^N)(x) \approx x + v(x)/2^N$ . Interpolation is required for calculating the composition in practical settings.

This concept was used and further discussed by Ashburner (2007) [8]. He introduces a new registration algorithm which he names DARTEL (Diffeomorphic Anatomical Registration using Exponentiated Lie algebra). The algorithm is based on the stationary velocity field representations which are exponentiated as defined above to produce the diffeomorphic transformation. He also notes that such a transformation has always a positive Jacobian in analogy to exponential of a real number being always a positive number. As a result the transformation can be guaranteed to be diffeomorphic. In practical setting this still depends on the number of time-steps used in the integration, how the velocity field is sampled, and how large changes are present in the sampled velocity field.

One of the main benefits of this framework, as noted by Ashburner, is the easy handling of inverse transformations. They can be obtained simply by exponentiating the negative velocity field. This also allows for straightforward testing of inverse consistency of the transformation. If the transformation is inverse-consistent, then  $\exp(v) \circ \exp(-v)$  should be identity transformation (or, in practice, close to it).

The paper by Ashburner discusses the possible limitations of the stationary velocity field model. The variable velocity field models presented above have the ability to maintain the anatomical correspondence between the velocity field and the warped image, since the velocity field evolves over time together with the image. On the other hand, in the stationary velocity field model no such correspondence between the velocity field and anatomical locations can be made as each anatomical point will be traveling along the stationary flow. As mentioned in the paper, in some cases, the LDDMM model above which aims to minimize the energy of the trajectory might be able to arrive in a trajectory with significantly smaller energy than what

is possible with stationary velocity fields. Also, some trajectories, which might be relatively easy to represent using time-varying velocity fields, are altogether impossible to represent using stationary velocity fields.

The paper leaves the quantitative comparison of time-varying and stationary models for future work and only evaluates the internal consistency of the framework. In the comparison done by Klein et al. (2009) DARTEL did quite well [11]. From the point of view of Deep Learning DARTEL is very interesting as static fields are significantly less complex than dynamic ones but the registration accuracy is still very good.

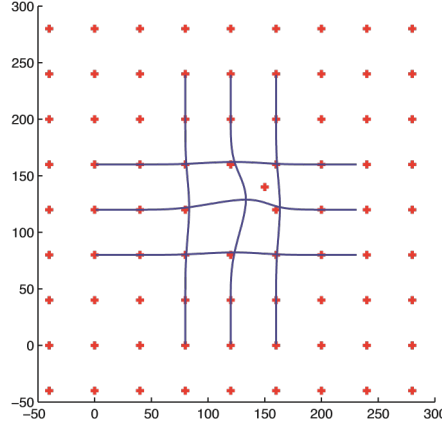
The DARTEL framework is in other aspects a relatively typical registration framework which some optimization strategy for optimizing a loss function. In the paper the loss function is derived using Bayesian framework. Regularization of the velocity field is viewed as a prior and the image similarity term is obtained from the likelihood term. The objective is then to find a maximum a posteriori (MAP) estimate of the transformation.

Another important practical application of the framework was Diffeomorphic Demons [7]. It uses classical Demons registration algorithm which considers registration as a diffusion process [40]. The original algorithm is not very suitable for Deep Learning purposes and is thus not reviewed in this work. Diffeomorphic Demons algorithm clarifies the theoretical basis of Demons algorithm as a two-stage iterative optimization problem and applies the diffeomorphic framework provided by Arsigny et al. [39] on top of that.

LDDMM framework can be considered for the stationary velocity fields as well; this was done by Hernandez et al. (2009) [41]. This formulation will not have all the beautiful properties of the non-stationary LDDMM approach but is considerably faster to compute. The basic idea is the same as in the normalization problem: minimization of a variational problem as defined in Equation (2.5). However, since the velocity field is now static, the optimization problem takes the form

$$\arg \min_v ||I_1 \circ \psi^{-1}(x, 1) - I_2||_{L^2}^2 + ||v_t||_V^2. \quad (2.12)$$

The resulting paths are now longer geodesics and thus part of the beautiful mathematical foundation is lost. However, the idea of minimizing the energy of the transformation connecting two images, that is, obtaining a path as close to geodesic path as possible, is still a reasonable goal, and computational cost is significantly reduced when using stationary velocity



**Figure 2.8.** An example of a 2D transformation formed using B-splines with lattice of control points. The figure is from [42].

fields.

### 2.2.3 B-Splines

Another commonly used method for representing transformations are splines which offer a very different representation compared to the velocity fields. Splines are special functions that can be represented using piecewise polynomials. The usefulness of splines comes from their representation using B-splines which are basis functions for splines. Naturally, B-splines themselves can also be represented piecewise by polynomials. Any spline can be represented as a linear combination of B-splines. By B-splines the transformation can be represented using a grid of control points as shown in Figure 2.8. For further details about the basics of using B-splines in registration we recommend the thesis by Schwarz, Loren [42].

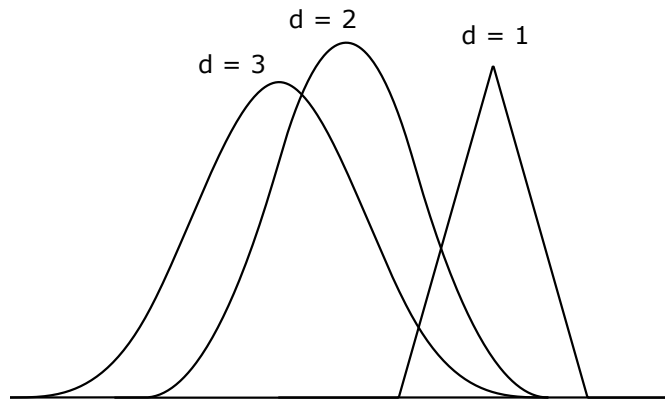
The resulting formulas for transformations are fairly simple. The commonly used representation for transformations  $T$  in three dimensions is of the form

$$T(x) = \sum_{l=0}^d \sum_{m=0}^d \sum_{n=0}^d B_l^d(x_1 - i) B_m^d(x_2 - j) B_n^d(x_3 - k) c_{l+i, m+j, n+k}, \quad (2.13)$$

where

$$\begin{cases} i = \lfloor x_1 \rfloor \\ j = \lfloor x_2 \rfloor \\ k = \lfloor x_3 \rfloor, \end{cases}$$

$B^d$  is B-spline of degree  $d$ , and vectors  $c_{l,m,n} \in \mathbb{R}^3$  define the three dimensional grid of control points. Depending on the desired density of the control points, linear scaling might have to be applied to the transforma-



**Figure 2.9.** Examples of B-splines with varying degrees ( $d$ ).

tion. Smoothness properties of the resulting transformation are defined by the degree of the used B-splines. In general,  $n$  degree B-splines result in  $n - 1$  continuous derivatives.

The support of the B-spline basis functions is compact. Thus any control point also has effect on the transformation only on limited distance which depends on the degree of the used B-splines. Examples of B-splines of varying degrees can be seen in Figure 2.9.

Regularization of transformations generated using B-splines is different from the dense displacement or velocity fields which we have looked at above. In a paper by Rueckert et al. [43] diffeomorphic framework for registration using cubic B-splines is constructed. In the paper they use a result by Choi and Lee [44] which basically states that if maximum displacement for the lattice control points is a constant (approximately 0.40) times the original lattice spacing, the resulting transformation is injective. That naturally gives maximum displacement depending on the density of the lattice. To overcome this rather strict constraint, the authors compose multiple such transformations on top of each other. Since each of the smaller transformations is diffeomorphic, also the resulting transformation is diffeomorphic.

### 2.3 Similarity Metrics

In this section we look at similarity metrics for registration. In general, similarity metrics are either intensity or feature based. Here we look only at intensity based metrics since we are not interested in using any highly application specific metrics for training the Deep Learning model.

Different similarity metrics are numerous [45–48], only the most common ones are referred at here.

Probably the most simple similarity metric is mean squared error (MSE). The underlining assumption with mean squared error is that corresponding anatomic locations should have same intensities. Thus small mean squared error indicates a good match.

Cross-Correlation is another common similarity metric. In registration context cross-correlation usually refers to functions of the form

$$CC(I_1, I_2) = \frac{\int_{\mathbb{R}^3} (I_1(x) - m_{I_1}(x))(I_2(x) - m_{I_2}(x)) \, dx}{\int_{\mathbb{R}^3} (I_1(x) - m_{I_1}(x))^2 dx \int_{\mathbb{R}^3} (I_2(x) - m_{I_2}(x))^2 \, dx}$$

where  $I_1$  and  $I_2$  refer to the compared images and  $m_{I_1}$  and  $m_{I_2}$  are mean functions of those images. By defining the mean functions as local averages over the given point has produced good results [9, 47]. Other versions of Cross-Correlation with different normalizations exist, such as Pearson Correlation Coefficient. Cross-Correlation is based on the assumption that image intensities are linearly correlated. As a result, the larger the Cross-Correlation the better the match should be.[45]

Mutual information and other information theory derived metrics have achieved excellent results in metrics comparison studies by providing very robust similarity metrics [47, 48]. Different information theoretic measures are numerous. Pluim et al. (2004) [49] provide good survey of the methods. A problem with information theoretic measures is however the need to see the whole image volumes for calculating the metric and thus they are not easily applicable for independent registration of sub-volumes.[45]

A relevant branch of similarity metrics is also gradient based metrics which try to match the image gradients instead of the intensity values. Mean squared error or Cross-Correlation can be modified to use gradient volumes instead of the intensity volumes. [46]

## 2.4 Unsupervised Registration Using Deep Learning

Using Deep Learning for registration is a very new field with the most relevant papers being from the last couple of years. During the year 2020 alone multiple different solutions have been proposed in the literature. In this section we will go through the relevant developments from the point of view of unsupervised registration. As a basis for our survey we use a review article by Fu et al. (2020) [13] which was written in December of 2019. However, we also try to take into account the papers published during 2020.

As mentioned in Section 2.2, in order to do unsupervised registration using Deep Learning we need the neural network to be able to obtain the deformed image produced by the network during the training phase. In other words, the chosen transformation representation has to be such that the deformed image can be computed from it using an algorithm for which the gradient from the output image can be back-propagated to the transformation representation. The most straightforward transformation representation is displacement field. Calculating the deformed image from displacement field requires basically interpolation. In 2015 Jaderberg et al. proposed a spatial transformer network [50] which essentially computes the transformed image using displacement field representation so that the gradients can be back-propagated. As a result, the possibility to do unsupervised registration using Deep Learning was noticed.

### 2.4.1 Network Training

Teaching a neural network in unsupervised registration has at least two approaches. The simpler one used by most of the publications is to simply use the loss function of the conventional registration optimization problem described in Equation (2.1) as a loss function for the neural network learning. The other approach using generative adversarial networks is to create a discriminator network which tries to discriminate unregistered and registered image pairs. Since no ground truths are available, the registered image pairs are generated by adding noise to duplicate images. Naturally this method works only in unimodal registration. The GAN approach was used for example by Elmahdy et al. (2019) in [51] and by Fan et al. (2019) in [52].

In principle, a benefit in using the generative adversarial network approach is that it simultaneously takes care of similarity measure and

regularization since the discriminator network will force the generator network to predict realistic looking deformed images. However, both of the papers mentioned above end up using an additional regularization term as the discriminator alone does not provide enough smoothness and the paper by Elmahdy et al. even ended up using additional similarity loss. Fan et al. note that the discriminator tends to learn too quickly making the network difficult to train.

GANs can also be used for additional regularization, something that was done by Fan et al. (2019) [52]: they used additional adversarial loss for ensuring that the deformed images seem realistic, in addition to classical similarity and regularization terms. In such a use case the discriminator is simply trained to discriminate between deformed and non-deformed images.

### 2.4.2 Transformation Representations and Inverse Consistency

Inverse-consistent and diffeomorphic registrations are often desired in registration as mentioned in the preceding chapters. In this section we look at different unsupervised Deep Learning registration methods for producing well behaving transformations.

Most of the unsupervised registration methods directly predict the displacement field using a fully convolutional neural network. However, there are at least three other options found in the literature: using B-splines [53, 54], using stationary velocity fields [10], or using momentum representation [55]. In these methods, the authors have developed a way to compute the displacement fields from the transformations predicted by the fully convolutional neural networks so that the gradient can be back-propagated.

Stationary velocity fields, as described in Section 2.2.2, were applied to Deep Learning by Dalca et al. (2019) [10]. Inspiration was taken mainly from DARTEL developed by Ashburner et al. (2007) [8]. For integrating the displacement field from the velocity field they use the scaling and squaring approach described in Equation 2.11) which is an approximate method for calculating the group exponential defined in Equation (2.10). The framework manages to produce extremely well behaving transformations, resulting in zero to a few folding voxels per volume with registration accuracy comparable to the best available methods. In the approach the loss function is formulated using Bayesian framework similarly to DARTEL. However, instead of simply maximizing the posterior probability they

employ variational auto-encoder approach. As a result the network learns to predict the uncertainty of the generated transformations. Unfortunately the authors do not analyze the usefulness of this estimate in the paper and thus it remains an open question.

Shet et al. (2019) [55, 56] use stationary LDDMM framework for registration. Instead of predicting the stationary velocity field directly they predict initial momentum. This way the network does not have to learn to predict smooth transformations. The velocity field is then calculated from the momentum field using a regularization kernel as defined in Equation (2.6). They use multi Gaussian kernel for regularization. Since the stationary LDDMM framework is used, the integration is done again with group exponential as defined in Equation (2.10) instead of Riemannian exponential. They leave the exact details for how the integration is done more open mentioning only that they use Runge-Kutta method for time integration. On average the resulting transformations have only few folding voxels per volume. The performance is similar or even better compared to the method by Dalca et al.

B-splines have been used in unsupervised registration by de Vos et al (2017, 2019) [53, 54]. We will focus on the latter paper which presents multi-step end-to-end registration framework. The basic component of the framework is a fully convolutional neural network receiving the moving and fixed images and predicting values for B-spline control points which define the transformation. As showcased in Equation (2.13), displacement field can be easily computed linearly given the B-spline control points and thus the gradient can be back-propagated. As the B-splines have local support using them makes sure that receptive field of each control point exceeds the range for which the point is affecting the transformation. In addition to B-splines, smoothness of the transformation is enforced by a physics inspired bending energy regularization term. Multi-step approach is used in order to avoid folding and to provide accurate transformations similarly to Rueckert et al. (2016) [43]. The first steps are given coarser resolution images and they predict less dense control point grids than the latter steps. Transformation parameters are composed in each step and are used to transform the original image for the next step. Total of three steps are used in the experiments with each step trained separately with weights of the other steps fixed. In addition, an affine registration network is added to the beginning. The framework manages to register images with relatively little folding voxels. However, significantly more folding is

present compared to the stationary velocity field approaches.

Methods for trying to achieve well behaving transformations with direct prediction of displacement fields also exist. In a typical solution transformation is regularized by some smoothness enforcing regularization term. If large displacements are present such standard regularization term is not enough to guarantee invertible transformations and additional regularization is needed.

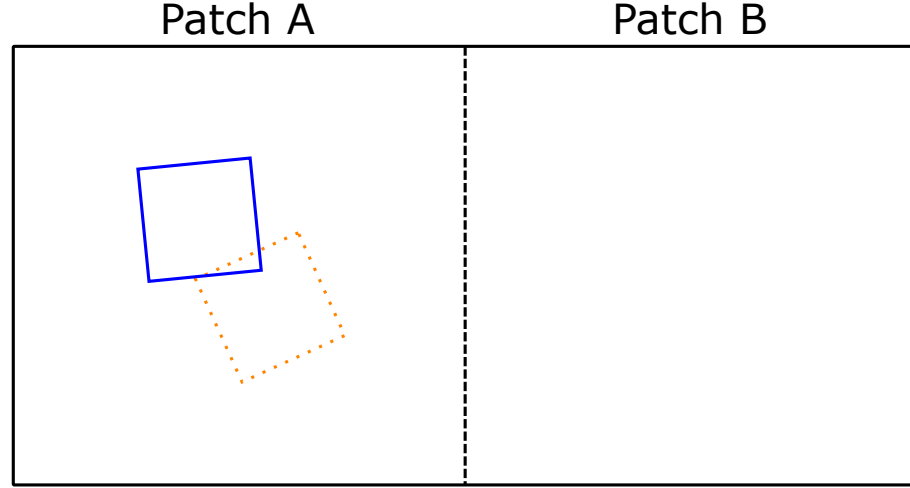
In work by Zhang et al. (2008) [57] inverse consistent registration is achieved using two constraints developed by the authors: inverse-consistent constraint and anti-folding constraint. The basic idea is that the transformation is predicted both ways, from image A to image B and image B to image A using two separate fully convolutional neural networks. The inverse consistency of these transformations is then enforced using the loss function term called inverse-consistent constraint. The additional constraint, anti-folding constraint, adds loss term relative to the absolute value of the Jacobian determinant for each voxel that has a negative Jacobian determinant thus penalizing folding in the transformations. Especially the anti-folding constraint proves effective in preventing folding in the transformations.

Somewhat similar solution to Zhang et al. can be found in a paper by Kim et al. (2019) [58] where cycle consistency loss is used to ensure diffeomorphic deformation. Again two separate fully convolutional networks are trained to predict the transformation in the opposite directions. Cycle consistency is obtained by applying the transformations to both of the images in succession (in different order) and then penalizing the dissimilarity between the original images and the twice transformed images in the loss function.

### 2.4.3 Patch-wise Registration

In many situations the 3D images are too large to fit into GPU memory in their full resolution. As a result, the training is often done with smaller sub-volumes called minipatches or just patches. During the prediction phase the whole transformation is calculated from the patches with sliding window approach, in other words the transformation is assembled from the predicted transformations for the sub-volumes. This kind of an approach has been used in multiple unsupervised Deep Learning frameworks [51, 52, 54, 59–64].

Major challenge in patch-wise registration, as noted for example by Yang



**Figure 2.10.** An example of a problematic situation for regularization with patch-wise approach to registration. In the figure the moving and target images are placed on top of each other represented respectively by the solid and dashed rectangles. White areas are ambiguous with no apparent features for matching the two images. Due to regularization, without further measures the situation might result in a transformation discontinuity at the patch border.

et al.(2017) [65], is that the transformation is ambiguous on the areas of equal image values and is defined only by regularization. Thus the transformation within an image patch is not defined only by the image values inside that patch but the neighboring patches might also affect the transformation. This is depicted in Figure 2.10.

The simplest way to try to solve this problem is to use only center part of the patches during the prediction. That way each voxel in the final predicted deformation has at least some context, depending on the size of the used padding.

Approach taken by many is to use multi-scale registration in which the images are first registered with lower resolution but larger field of view [59, 61, 64]. The patches are then taken from these preregistered images. That way the patches should only contain smaller movement. The final deformation field can be obtained by composing the patch-wise deformation fields with the coarser resolution deformation field(s).

The paper by Yang et al. [65] is about supervised Deep Learning registration but the solution they ended up is still worth mentioning. They use the LDDMM framework and train the network to predict initial momentum fields as described in Section 2.2.2. Initial momentum is calculated patch-wise but the patches are combined before applying the regularization kernel  $K$  defined in Equation (2.6). That way the regularization is transferred over the patch borders. The network is trained using ground truth initial momentum fields obtained using classical iterative LDDMM methods.

### 3. Methods

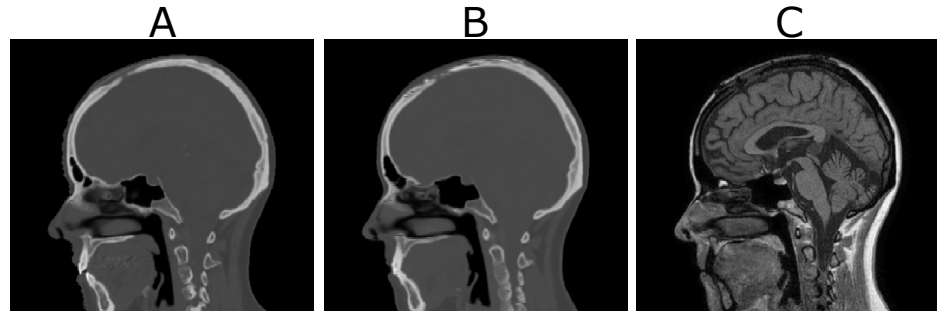
The purpose of this work is to develop well working unsupervised Deep Learning registration framework for registering pseudo-CT and CT images. Pseudo-CT images are synthetic CT images generated from magnetic resonance images. Accurate registration of CT and pseudo-CT images would open a way for multimodal registration between MRI and CT images. By registering a pseudo-CT image to a CT image one is very close to also registering the MRI image which was used for producing the pseudo-CT image to the CT image. In the simplest case this would mean applying the deformation field produced by the pseudo-CT - CT registration to the MRI image.

We will start the chapter by looking at the special characteristics of the registration problem at hand. After that we will continue by an overview and a rationale for the used methods followed by a more in-detail description.

#### 3.1 Problem Characteristics

Properties of the images to be registered are relevant for choosing the method for registering them. In this work both of the registered images are CT-like images. CT images are three-dimensional versions of so called x-ray images. Compared to MRI images, CT images show bones and air cavities more clearly whereas MRI images have significantly better soft tissue contrast.

Pseudo-CT images try to mimic CT images but never fully do so and have less anatomical details than real CT images. The used pseudo-CT images are also created using a Deep Learning algorithm but producing them is not part of this work and can be considered a relatively independent problem. The used pseudo-CT images are created for replacing CT images



**Figure 3.1.** Example case of the data to be registered with each image showing the same mid-sagittal slice of the same patient. A: CT-image. B: pseudo-CT image. C: MRI image used for creating the pseudo-CT image, only one contrast shown here. The medical images are Courtesy of Turku University Hospital / Department of Oncology & Radiotherapy.

in radiotherapy treatment planning and delivery. The main goal of the pseudo-CT images is to provide similar results to real CT images when computing radiation dose in tissue and positioning a patient during a treatment. The local average tissue density is the relevant factor for correct radiation dose calculation. As a consequence, the pseudo-CT images do not need to have anatomical details that would be present in real CT images.

The main reason for the lack of detail in pseudo-CT images is that the used MRI image contrasts simply might not hold enough information for accurate prediction of a CT image for every single anatomical location. This is especially true for peripheral locations which might have bad image quality. The used MRI sequences are primarily optimized for geometrical accuracy, and not for perfect tissue contrasts in terms of pseudo-CT generation, since good geometrical accuracy is very important for radiotherapy applications.

Another relevant feature of the used data is the image resolution. Images have axial resolution of  $400 \times 400$ . The number of axial slices vary between 230 and 270. The images are too large to fit into the used GPUs during the training. As a result the images have to be downsampled or a patch-wise approach must be taken.

### 3.2 Algorithm Overview

Due to the difficulties described above inherent to the used data set, good regularization of the predicted transformations is especially important. Deep Learning can be expected to be suitable for the task as it can be assumed to be more robust against undesired local minima than the classical registration methods [54].

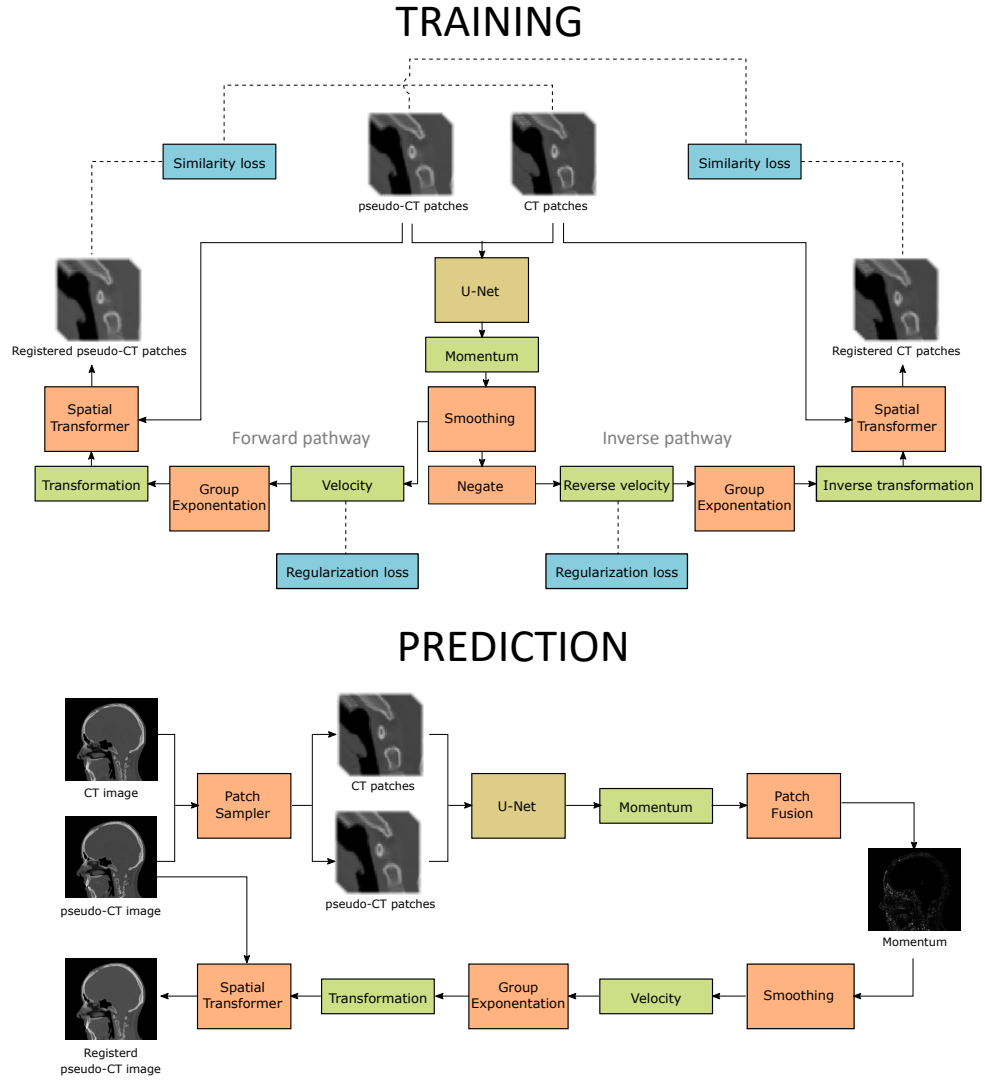
The most robust method for creating diffeomorphic transformations in

unsupervised Deep Learning registration is currently to employ the static velocity field approach used by Dalca et al. (2019) [10] and Shen et al. (2019) [55]. Hence, it is chosen as the transformation representation. For calculating the group exponential, scaling and squaring method introduced by Arsigny et al. [39] is used as it is the most efficient algorithm available for the task. In addition, it was shown by Dalca et al. to be usable also in unsupervised Deep Learning context.

Dalca et al. employ probabilistic variational approach. As the usefulness of the uncertainty estimates have not been studied thoroughly we do not employ the strategy here in order to limit the scope of the thesis. Also, computational cost is increased by the approach. Instead the stationary LDDMM initial momentum approach taken by Sheth et al. is used. LDDMM offers strong mathematical foundation for building diffeomorphic transformations. The strength of those results is weakened by the stationary approach but is still shown to work by Hernandez et al. [41]. In short, the question is about whether to use group or Riemannian exponential for calculating the displacement field from the initial velocity.

Patch-wise approach is employed to overcome the problem of the data not fitting to the GPU memory. Currently all the unsupervised patch-wise approaches use either dense displacement fields or B-splines. This is surprising as the velocity field approach offers very clean method for fusing the patches during the prediction phase. Since we are using actually momentum fields, not velocity fields, even the regularization can flow over the patch boundaries during the prediction. During the prediction phase, momentum is calculated for each patch separately after which the smoothing kernel is applied to the whole momentum volume. After that the velocity field is exponentiated using the group exponential to produce the displacement field. In that sense our approach is similar to the supervised approach taken by Yang et al. (2017) [65]

Often in patch-wise registration multi-scale registration is employed. We do not want to use that here for two reasons: firstly, to limit the scope of the thesis and, secondly, to have access on how well the initial momentum approach is able to work as a single step patch-wise registration method. If movement exceeding the chosen patch field of view is present, multi-step approach would obviously be needed. Here we preregister the images only rigidly (translation and rotation). Affine registration commonly employed as the first step is not used as it will introduce anatomically unfeasible shearing to the bone tissue.



**Figure 3.2.** Overview of the used architecture. The only component with trainable parameters is the U-Net. Only the forward pathway is shown for the prediction part. For CT to pseudo-CT registration the velocity field should be reversed and the final transformation be applied to the CT image instead of the pseudo-CT image. The medical images are Courtesy of Turku University Hospital / Department of Oncology & Radiotherapy.

As a network architecture we employ modified U-Net which used dilated convolutions following the very recent paper by Wang et al. (2020) [66]. The resulting network has fewer parameters but larger receptive field compared to the traditional U-Net.

As a loss function in addition to typical mean squared error we experiment with normalized mean squared error of the form  $x^2/(a^2 + x^2)$  where  $x$  is the error and  $a \in \mathbb{R}$  is some constant. A loss function of this form is not very commonly used but it provides us with more similar weight for all tissue boundaries compared to the simple mean squared error loss. Normalized cross correlation has similar benefits but this is significantly faster to compute.

Overview of the used framework can be seen in Figure 3.2.

### 3.3 Algorithm Description

We employ the stationary LDDMM approach for registration. The theoretical background for the methodology was described in Section 2.2.2. Instead of predicting the displacement field directly we predict the momentum field from which the displacement field can be calculated.

In our approach the neural network is a function which estimates the optimal transformation directly given the images to be registered. The task is to approximate function  $f$ , which given two images  $I_1$  and  $I_2$ , returns the optimal momentum  $m : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ . The optimal  $f$  is of the form

$$f(I_1, I_2) = \arg \min_m \langle v, m \rangle_{L^2} + \mathcal{E}_s(I_1, I_2, v)$$

$$v = K^{-1} * m,$$

where  $v = K^{-1} * m$  is the velocity field with  $*$  denoting convolution,  $K^{-1}$  is a convolution kernel, and  $\mathcal{E}_s$  is the similarity term. Coordinate transformation  $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^3$  from  $I_1$  to  $I_2$  can be obtained as  $\phi(x) = \psi(x, 1)$ , where  $\psi$  is defined by the differential equation

$$\frac{\partial \psi}{\partial t}(x, t) = v(x(x, t))$$

$$\psi(x, 0) = \text{Id}.$$

For the similarity term we use a symmetric approach

$$\mathcal{E}_s(I_1, I_2, v) = \frac{1}{2} \epsilon(I_1 \circ \phi^{-1}, I_2) + \frac{1}{2} \epsilon(I_2 \circ \phi, I_1),$$

where  $\epsilon$  is a functional returning a scalar value representing the similarity between the input images with larger values indicating less similar images. We do not employ symmetric image normalization approach where the similarity is calculated in the intermediate space at time  $t = 0.5$  between the images, since intuitively it is less robust towards situations where some anatomical feature is entirely missing from the other volume, which is something that might occur between CT and pseudo-CT images. The conclusion was also confirmed by initial experimentation. However, it remains an interesting option for other applications as it is very easy to apply in the stationary velocity field context.

Joining all the equations we can define the optimal momentum prediction

function  $f$  as

$$f(I_1, I_2) = \arg \min_m \langle v, m \rangle_{L^2} + \epsilon(I_1 \circ \phi^{-1}, I_2) + \epsilon(I_2 \circ \phi, I_1)$$

$$\text{with } \begin{cases} v = K^{-1} * m \\ \frac{\partial \psi}{\partial t}(x, t) = v(x(x, t)) \\ \psi(x, 0) = \text{Id}. \end{cases} \quad (3.1)$$

Let us denote the neural network predicting the momentum by  $\mathcal{N}_w$ , where  $w \in \mathbb{R}^n$  is the parameter vector of the neural network. The goal is then to find parameters  $w$  so that  $\mathcal{N}_w$  approximates  $f$ .

The approximation is done by gradient optimization using sample images, which is the common practice with neural networks. Let us write the optimization target function as

$$\mathcal{E}(m, I_1, I_2) := \langle v, m \rangle_{L^2} + \frac{1}{2}\epsilon(I_1 \circ \phi^{-1}, I_2) + \frac{1}{2}\epsilon(I_2 \circ \phi, I_1). \quad (3.2)$$

Basically, given two images, one needs to be able to calculate the gradient

$$\frac{d}{dw} \mathcal{E}(\mathcal{N}_w(I_1, I_2), I_1, I_2). \quad (3.3)$$

As mentioned, the gradient is calculated relative to image patches. As a result the  $\langle v, m \rangle_{L^2}$  is not exactly the same thing as it would be for whole volumes.

### 3.3.1 Practical Implementation

In practice we only have a grid of samples from the images and accordingly predict the momentum only for a grid of values. In this section we present the actual algorithm implementation in detail.

For calculating the gradient defined in Equation (3.3) we use Tensorflow library [67] and thus formulas for the gradient will not be presented in this work. However, we need to have formulas for calculating the loss function, Equation (3.2), so that the gradient can be back-propagated using Tensorflow.

#### *Calculating the Velocity Field*

Calculating the velocity field from the momentum field is defined using convolution. The width of  $K$  can be quite large and thus using Fast Fourier Transform (FFT) algorithm for the calculation is the most efficient way.

Gradient can also be back-propagated over FFT using Tensorflow. For the convolution kernel  $K^{-1}$  we choose the classical kernel used by Beg et al. (2005) [4] induced by the differential operator  $L = \alpha \nabla^2 + \gamma \text{Id}$  with  $K = L^2$ .

In order to calculate  $K^{-1} * m$  using FFT we need to have a frequency-domain representation for the  $K^{-1}$ . For discrete periodic function in one dimension the operator  $L : \mathbb{Z}/N \rightarrow \mathbb{R}$  can be defined as

$$\begin{cases} L(0) &= 2\alpha/\Delta x^2 + \gamma \\ L(1) &= L(-1) = -\alpha/\Delta x^2, \end{cases}$$

where  $\Delta x$  is the sampling resolution of the function. For the discrete Fourier-transform DFT of  $L$  we obtain

$$\begin{aligned} \text{DFT}[L](j) &= \sum_{k=0}^{N-1} L(k) e^{-2\pi i k \cdot j / N} \\ &= 2\alpha/\Delta x^2 + \gamma - \alpha/\Delta x^2 e^{-2\pi i j / N} - \alpha/\Delta x^2 e^{+2\pi i j / N} \\ &= 2\alpha \frac{1 - \cos(2\pi j / N)}{\Delta x^2} + \gamma, \end{aligned}$$

Since inverse is simply exponentiation with  $-1$  at frequency domain we obtain

$$\text{DFT}[K^{-1}](j) = \left[ 2\alpha \frac{1 - \cos(2\pi j / N)}{\Delta x^2} + \gamma \right]^{-2}.$$

The calculation can be done similarly for the three dimensional case. Given the kernel in frequency domain, we can calculate the convolution as a product in the frequency domain.

FFT calculates the convolution in cyclic domain which is not desired here. Thus we need to pad the momentum field with zeros before applying the convolution in frequency domain. Padding width needs to be chosen based on the used parameters  $\alpha$  and  $\gamma$  as they define the kernel width.

### *Calculating the Displacement Field*

Displacement field is calculated from the velocity field as in the paper by Dalca et al. (2019) [10] using the scaling and squaring approach presented in Equation (2.11). In discrete setting the only required special operator is the spatial transformer by Jaderberg et al. (2015) [50] which basically does interpolation.

Let  $u_d : \mathbb{Z}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ ,  $u_d(x, t)$  be the discrete displacement field at point  $x \in \mathbb{R}^3$  at time  $t \in \mathbb{R}$  of the integration. Let us denote interpolation operator of some discrete vector field  $w : \mathbb{Z}^3 \rightarrow \mathbb{R}^3$  at point  $x \in \mathbb{R}^3$  as  $\mathcal{I}[x, w]$ . We can now define the displacement field in  $\mathbb{R}^3$  as  $u : \mathbb{R}^3 \times \mathbb{R} \rightarrow \mathbb{R}^3$ ,  $u(x, t) =$

$$\mathcal{J}[x, u_d(\cdot, t)].$$

Now the velocity field can be integrated straightforwardly using the approach in Equation (2.11) to obtain the approximate displacement field at time  $t = 1$ . Using  $N$  scaling and squaring steps, we have:

$$\begin{aligned} u_d(x, 1/2^{N-1}) &= u_d(x, 1/2^N) + u(x + u_d(x, 1/2^N), 1/2^N) \\ u_d(x, 1/2^{N-2}) &= u_d(x, 1/2^{N-1}) + u(x + u_d(x, 1/2^{N-1}), 1/2^{N-1}) \\ &\dots \\ u_d(x, 1/2) &= u_d(x, 1/2^2) + u(x + u_d(x, 1/2^2), 1/2^2) \\ u_d(x, 1) &= u_d(x, 1/2) + u(x + u_d(x, 1/2), 1/2). \end{aligned} \tag{3.4}$$

If  $N$  is large enough  $u_d(x, 1/2^N) \approx v_d/2^N$  giving us the initial value. By reversing the initial value to  $-v_d/2^N$ , inverse transformation can be obtained.

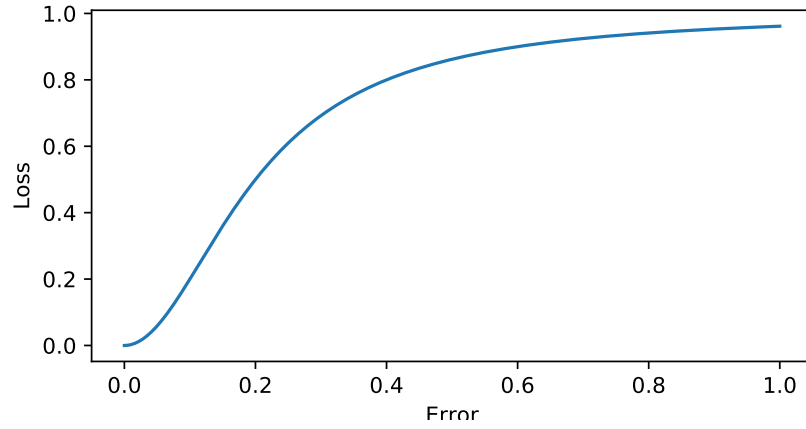
Note that although each updating step is in principle accurate, at each step the field is only updated at the discrete points and thus error is introduced at each step in addition to the original approximation error. Dalca et al. concluded that  $N = 7$  was the best value in their experiments. This is used in this work as well. Similar conclusion of either 6 or 7 steps being optimal was made also by Ashburner et al. (2007) [8] in a classical setting.

Using the discrete displacement field the transformed image can be obtained by using the spatial transformer module again by interpolating the image at locations defined by the displacement field.

### *Similarity Metric*

Choosing a good loss function is very important for registration performance. In unimodal registration simple mean squared error can work relatively well. However, during the experiments it was noticed that it failed to provide good registration for soft tissue and certain bones while it matched, for example, body outline very well.

Mean squared error gives relatively higher weights for large errors. However, registration can be in principle seen as a Boolean problem with tissue types either matching or not matching. This idea can be seen to be especially applicable to registration of CT-like images, since CT images have very discrete-like appearance with large areas of roughly equal intensity for basic tissue types (bone, fat, muscle etc.). On the other hand, giving relatively smaller weights for very small errors is still desired,



**Figure 3.3.** Plot of the loss function defined in Equation (3.5) (with constant  $a = 0.2$ ).

since in practice pseudo-CT images and CT do not have exactly equal values for the same anatomic locations. A good loss function for this task would hence allow for some tolerance while providing relatively similar weight for larger errors.

Normalizing the mean squared error provides us with the s-shaped curve of the form

$$\frac{x^2}{a^2 + x^2}. \quad (3.5)$$

Here  $a \in \mathbb{R}$  is some constant. The loss function is plotted in Figure 3.3. The value of the constant defines the half value of the loss function.

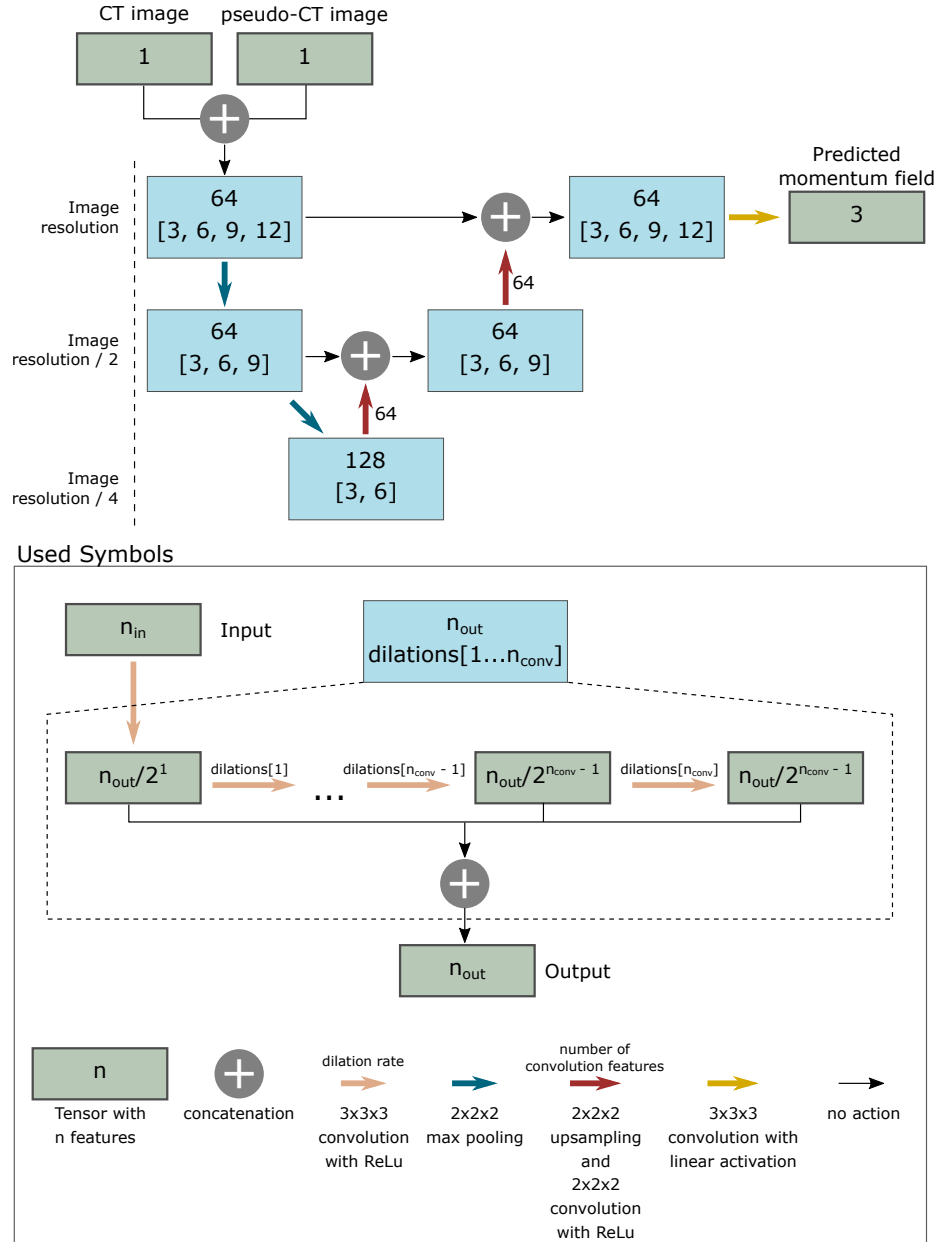
Other similarity metrics were also considered. Compared to normalized cross correlation the proposed metric is significantly faster to compute. Mutual information is also very interesting candidate but it is not suitable for registration of small patches. Instead, we will use mutual information for accessing the registration quality.

#### *Neural Network Architecture*

While the purpose of the present work was not to find an optimal network architecture, some work was put into choosing a decent one. U-Net type networks have been used in basically all the unsupervised registration related papers and thus it was a natural choice.

A large receptive field is important for a good registration algorithm, since it needs to take overall anatomy into account in order to find the mapping optimal in energy between the images. To provide easier access to larger receptive field, U-Net architecture taking advantage of dilated convolutions was adopted.

The chosen network architecture follows the architecture presented in the paper by Wang et al. (2020) [66]. Their architecture overperforms



**Figure 3.4.** The used neural network architecture is presented in the top half and the used symbols are explained in the bottom half. Dropout was added after the down-sampling steps and the first concatenation step.

the classical U-Net in a segmentation task while using significantly less parameters. The architecture is shown in Figure 3.3.

In our practical implementation, the second up-sampling step has  $3 \times 3 \times 3$  convolution filters instead of the more natural  $2 \times 2 \times 2$  due to a mistake. The mistake was noted late and thus the experiments were not rerun. It can be considered unlikely for it to have relevant impact on the results as in practice the network can still learn to predict  $2 \times 2 \times 2$  filters.

#### *Patch-wise Training and Prediction*

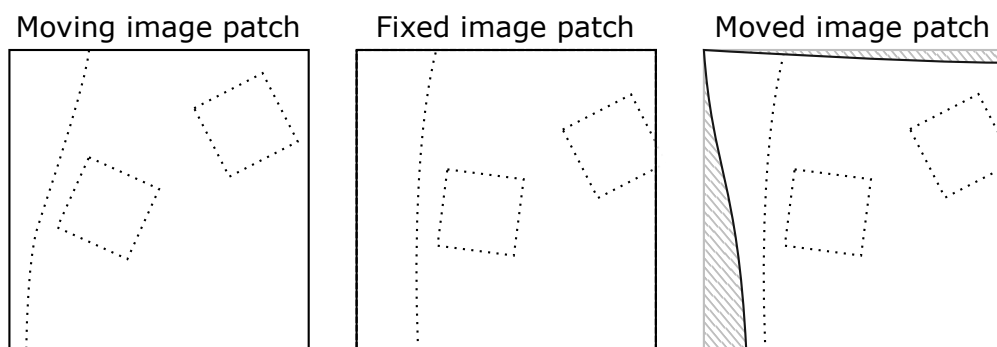
While the patch-wise approach allows one to fit the model to the GPU it presents us with a new set of problems. For registration the patch-

wise approach introduces some unique problems not found in other Deep Learning applications.

Often in registration the displacement field is assumed to be zero at the image borders but with the patch-wise this is not possible. Instead, the moving image moves over the patch borders. As a result, some values of the moved image are unknown. This is shown in Figure 3.5. During the training we want to back-propagate the gradient from the similarity metric between the moved image and the fixed image. In addition, we want to obtain metrics about progress during the training. To do this, we mask out the unknown area from the moving image for back-propagation and metrics calculation.

A general disadvantage in the patch-wise approach is that the network has less context, with the situation being worst on the patch borders. For registration the patch size should be, in principle, chosen so that the movement present in the data set fits inside a patch. However, if the movement is large in comparison to the patch size, only anatomical locations close to the patch borders can be seen in both the moving and the fixed image, and only if the movement for the location is away from the patch border. In that case the matched anatomic locations would be on the opposite sides of the patch in moving and fixed images. Intuitively, this would reduce registration quality significantly, since there would be less data by which to do the registration. However, the situation might not be that bad since the network could learn to register images based on secondary features other than directly matching anatomical locations. However, it is safe to assume that the patch size should be significantly larger than the present movement.

In the prediction phase, only the center parts of the patches are used. The width of this discarded region should be considered at least as important a



**Figure 3.5.** In patch-wise registration moving image can move over the patch borders. Image values on the area marked by the diagonal stripes are unknown. Weights are updated based on the known area during the training.

parameter as the patch size itself. If the movement present is less than the width of the discarded region, all the locations in the final prediction have had at least enough context to see their anatomical locations in the fixed image. Thus it is safe to assume that for accurate prediction, the margin should be chosen so that most of the movement fits inside it.

Another patch-related problem unique to registration is the regularization challenge presented in Section 2.4.3. To overcome the problem, we adopt the approach where in the prediction phase the momentum field is predicted separately for the patches which are then combined before calculating the predicted transformation. As a result, regularization flows over the patch borders. However, training the momentum prediction network remains a problem, since during the training the transformation should be calculated also for a single patch. As mentioned earlier, FFT is used for applying the smoothing kernel to a zero-padded momentum input. This introduces a bias towards the border areas of the patch as zeros can not be considered neutral values in this context. In practice, the network tends to learn higher values close to the edges of the image. Extrapolating predicted momentum values outside the patch could reduce the scaling problem. However, that would introduce other, possibly worse, biases since the only way to apply the smoothing properly would be to know the predicted momentum values outside the patch. Since there is no robust way for handling the patch borders properly, there remains the experimental question whether discarding the border regions in the prediction phase is enough to provide practical solution to the problem.

## 4. Experiments

We conduct two experiments that aim to evaluate the performance of the proposed registration methods: one with mean squared error as the similarity metric and other with the normalized mean squared error (MSE) described in Equation (3.5). The aim of the experiments is to evaluate performance and internal consistency of the proposed registration framework. The latter includes evaluation of how well the framework manages to produce invertible transformations and how robust is the proposed patch-wise registration approach.

Experiments are conducted on a data set of brain CT and MR images with a total of 119 different subjects from three different hospital sites. Pseudo-CT images were calculated from the MR images for each subject. The data set was divided into training, validation, and test sets. The number of subjects for training, validation, and testing were 66, 14, and 39, respectively. However, most of the subjects had two MR image versions, one with and one without contrast agent. As a result, the number of image pairs for training was 108, for validation 27, and for testing 66.

CT images were rigidly registered to the pseudo-CT images together with interpolating them to the same resolution and field of view. The voxel size of the pseudo-CT images is  $0.675 \times 0.675 \text{ mm}^2$  in axial plane and 1.0 mm in feet-head direction. In addition, their axial resolution is  $400 \times 400$  while the number of axial slices vary between 230 and 270. The originally used CT images have the same slice thickness as the pseudo-CT images but the axial voxel size is smaller. Thus it was a natural choice to interpolate the CT images to the pseudo-CT image resolution as this way we entirely avoid having to interpolate any of the pseudo-CT images.

In addition, the CT image intensity values were normalized to density values to eliminate effect of having data from different hospitals in the data set. Hospital site specific calibration curves were used for normalizing

the pseudo-CT images. Very large values caused by, for example, dental implants were cut to  $6.7140 \text{ kg dm}^{-3}$ .

Computations are done on a computer with NVIDIA Quadro RTX 5000 GPU, Intel Xeon W-2275 (3.30 GHz) CPU, and 128 GB of physical memory.

#### 4.1 Evaluation Metrics

For evaluating the registration accuracy we calculate a set of metrics between the registered and target image pairs. The most simple ones used are voxel-wise mean absolute error and mean squared error. As a more advanced metric we employ normalized mutual information. Additionally, we calculate Sørensen–Dice coefficient for tissue type masks using simple thresholding. To access the transformation regularity we calculate the number of voxels with a negative Jacobian.

For normalized mutual information we used definition by Studholme et al. [68], written as

$$\frac{H(X) + H(Y)}{H(X, Y)},$$

where  $X$  and  $Y$  are random variables and  $H(\cdot)$  is Shannon entropy of a random variable. In our use case the random variables  $X$  and  $Y$  refer to intensity values of a random voxel in the compared images. Shannon entropy can be defined for a continuous real valued random variable  $X$  as

$$H(X) = - \int_{\mathbb{R}} p(x) \ln(p(x)) dx$$

where  $p$  is the probability density of the random variable  $X$ . We approximate the probability density functions using a joint histogram of  $64 \times 64$  bins calculated from the intensity values of the two images. Normalized mutual information ranges between one and two with larger value indicating better match.

The threshold-based masks were produced for three generic tissue types: bone, water, and fat. In addition, mask of the whole body was used. Density values for different tissue types overlap significantly [69] and thus arbitrary choices were needed. For fat we used values between  $0.8 \text{ kg dm}^{-3}$  and  $0.97 \text{ kg dm}^{-3}$ , for bone anything above  $1.15 \text{ kg dm}^{-3}$ , and watery tissue was defined as anything between fat and bone, in other words, values between  $0.97 \text{ kg dm}^{-3}$  and  $1.15 \text{ kg dm}^{-3}$ .

The metric is then calculated by computing the masks for registered and unregistered volumes and calculating the dice score between the masks.

Sørensen–Dice coefficient or just Dice coefficient can be calculated as

$$\frac{2|A \cup B|}{|A| + |B|}$$

between two sets  $A$  and  $B$ .

To calculate the amount of folding voxels in the produced transformations, Jacobian is calculated for each voxel location. Negative Jacobian means that the transformation is folding on top of itself. Jacobian can be calculated as

$$\det \left[ \frac{\partial \phi}{\partial x} \right] = \det \begin{bmatrix} \frac{\partial \phi_1}{\partial x_1} & \frac{\partial \phi_1}{\partial x_2} & \frac{\partial \phi_1}{\partial x_3} \\ \frac{\partial \phi_2}{\partial x_1} & \frac{\partial \phi_2}{\partial x_2} & \frac{\partial \phi_2}{\partial x_3} \\ \frac{\partial \phi_3}{\partial x_1} & \frac{\partial \phi_3}{\partial x_2} & \frac{\partial \phi_3}{\partial x_3} \end{bmatrix}$$

for transformation  $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ . We approximate the partial derivatives by differences in the transformation between neighboring voxels.

## 4.2 Patch-wise Approach Evaluation

Due to the used patch-wise approach there will be anomalies on the patch borders as discussed in Section 3.3.1. Our method aims to discard a sufficient number of patch borders so that the anomalies do not affect the final deformation field except on the volume border regions. To evaluate if the approach can be considered valid, we calculate voxel-wise mean and standard deviation for momentum vector lengths over all the predicted patches. The motivation is to see how far the obvious border anomalies span.

## 4.3 Baseline Method

In order to provide a baseline against which to compare the method, NiftyReg [70] was used. The used algorithm behind NiftyReg is described in [71].

NiftyReg provides several alternative configurations for doing the registration. We use normalized mutual information as a loss function and stationary velocity field defined by cubic b-splines as transformation representation. NiftyReg uses hierarchical approach where registration is performed on multiple levels, beginning with the whole volume and then moving to smaller sub volumes. A hierarchy of 3 levels is used in our

experiments. The default value of 150 is used for the maximum number of iterations. For regularization, default values were also used, consisting of a bending energy term together with first order penalty.

#### 4.4 Training and Prediction Details

In both experiments we use the patch size of  $48 \times 48 \times 48$  voxels and the patches are fed to the GPU in batches of 12. The patches are extracted from 4 volumes at a time and fed to the network in random order.

For regularization, kernel parameters  $\alpha = 20$  and  $\gamma = 1$  are used. The weight of  $1e - 4$  is used for the regularization term in the neural network loss function for both models.

In calculating the velocity field from the momentum field we pad the volume with 12 voxels.

During the prediction 12 border-most voxels are discarded from each side. As a result only the 24 center-most voxels are used from each patch except for the patches at the volume borders. And decision to discard exactly 12 voxels was based on initial experimentation and its validity is analyzed in the results section.

For the normalized MSE loss function given in Equation (3.5) we use the constant parameter  $a = 0.2$ .

## 5. Results

In this chapter, we look at the results of the experiments described in the previous chapter. The aim is to evaluate the Deep Learning registration method we developed. Evaluation of registration results can be difficult due to the lack of ground truths [6]. There is always a lot of ambiguity in the estimated transformations. In practice, the best evaluation might be obtained by visual analysis but that too is tedious and observer-dependent. We evaluate the registration quality based on the set of metrics defined in the previous chapter. In addition, we provide qualitative remarks based on our visual observations.

We use NiftyReg[70] as a baseline method, as discussed in the previous chapter. The point is necessarily not to show that our method outperforms the baseline or vice versa. The main purpose is to see if the metrics calculated for our model are in the same scale as the ones calculated for the baseline model. Thus the baseline acts more as a sort of a sanity check for our method. The reason why comparing the metrics might not be meaningful is that the metrics are affected a lot by the used parameters. For example, relaxing regularization weight results in a better match between the images, resulting in better metrics. We also do not perform any sort of parameter optimization for NiftyReg but instead use it almost out of the box. The only difference to the default settings is that we use the velocity field representation for the transformation. This way NiftyReg provides invertible transformations, which is better in line with our method. Another thing to note is that NiftyReg performs the registration in multiple scales, whereas our method performs only patch-wise registration for rigidly registered images. Thus NiftyReg can be expected to handle large-scale transformations better. How much better, is the question of interest for us.

Another baseline we have are the original rigidly registered images.

Improvement of the metrics compared to the original rigid registration can provide some indication of the quality of the method.

We have got two different Deep Learning models to evaluate: one using mean squared error loss and another one using the normalized mean squared error, Equation (3.5). We refer to these models respectively by "DL MSE" and "DL Normalized MSE". To the initial registration we refer as "Rigid" and to the baseline method as "NiftyReg".

## 5.1 Training

The models were trained for 20 epochs but we used the epoch 12 in our experiments, because the validation loss had stabilized already by then. One epoch corresponded roughly to seeing the whole data set once. Training one epoch took a little less than three hours.

In order to get the MSE model to converge, one had to start with a lower value for regularization and increase it after one epoch. Normalized MSE model converged more easily.

## 5.2 Prediction Runtime

Runtime of the algorithms was not analyzed systematically but we can still provide qualitative results. Prediction of the transformation on GPU using the trained model takes between 100 and 150 seconds per case. The difference is due to the differing number of slices in the feet-head direction. The relatively long runtime for a Deep Learning algorithm results from the adopted patch-wise approach. The number of patches for which the prediction has to be made is quite high for the used patch size, around 2600 per image volume. This was largely due to only the center part of each patch being used, amounting to only 12.5% of each patch. As a result, the amount of data for which the prediction has to be made is ten times the size of the volume. Most of the run time goes to predicting the momentum field. Calculating the displacement field and the transformed image from the momentum field takes only a couple of seconds.

Even though the run time is large for a Deep Learning registration algorithm, it is still good when compared to most classical registration algorithms which take even hours to produce the result. However, for example the GPU version of NiftyReg can have a registration time of less

	DL Normalized MSE	DL MSE	NiftyReg
<b>CT to pseudo-CT</b>			
% of folding voxels	0.000005%	0%	0%
# of cases with folding voxels	1	0	0
Mean Jacobian	0.998392	0.999792	1.00062
<b>pseudo-CT to CT</b>			
% of folding voxels	0%	0.000003%	0%
# of cases with folding voxels	0	1	0
Mean Jacobian	1.0015	0.999838	0.999217

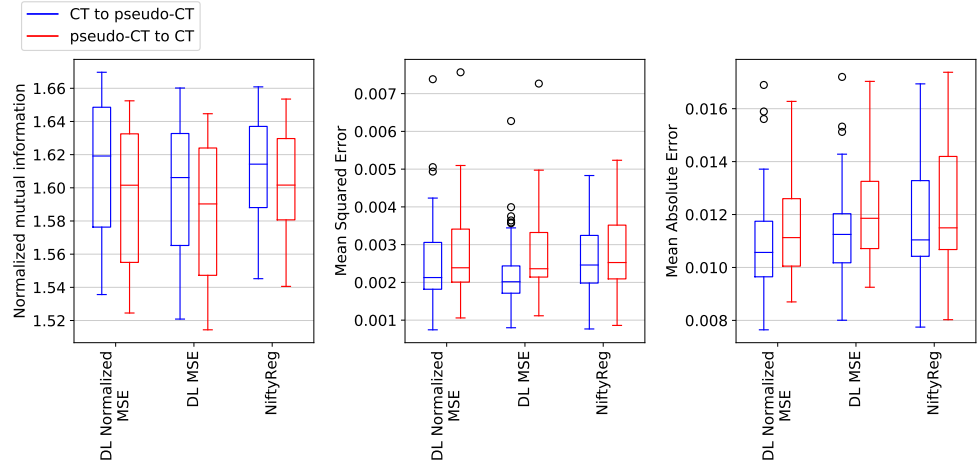
**Table 5.1.** The proportion of folding voxels (voxels with negative Jacobian determinant) and the mean of Jacobian determinant (the mean over means of deformations) calculated for the test set images registered using different registration methods. The image pair for which DL Normalized MSE model results in folding voxels has very large movement present and all the registration methods fail to register it even decently.

than 10 seconds on a modern GPU [71]. Using the static velocity field representation increases its run time significantly but it is still at least in the same scale as our method. The CPU version had a run time from 200 to 1100 seconds per image volume in our experiments. The difference results mostly from the different number of iterations needed until the algorithm converges. This is in contrast to our proposed Deep Learning method for which the run time is practically a linear function of the number of voxels in the image.

The run time of the algorithm is not optimized and could be improved. The most simple solution would be to use larger patch size as it would allow larger portion of the patch to be used. It is also noteworthy that the momentum field calculation is trivial to parallelize for multiple GPUs.

### 5.3 Metrics Results

In this section, we look at the evaluation metrics calculated for the registration results on the test set. Our method provides an invertible transformation between the input images. Thus each metric is calculated both ways, in the CT space and in the pseudo-CT space. These are referred to as "pseudo-CT to CT" registration and "CT to pseudo-CT" registration, respectively. The used NiftyReg algorithm also provides invertible transformations but the loss function is not symmetric. Thus we performed the registration separately both ways for a fair comparison.



**Figure 5.1.** Three generic similarity metrics calculated for the test set images registered using different registration methods.

### 5.3.1 Transformation Regularity

The results for transformation regularity can be seen in Table 5.1. The proposed methods provide diffeomorphic transformations with zero folding voxels for almost all of the test cases. Both Deep Learning models have folding voxels in exactly one image pair. The image pair for which the Normalized MSE model has folding voxels has very large movement and is the only image pair for which the registration fails completely for all the methods, including NiftyReg (also an only image pair for which any of the methods fails completely). Our proposed method thus manages to provide perfectly invertible transformations for practically all of the test set image pairs, despite predicting the transformation in separate patches.

In addition, the mean of the Jacobian determinant is close to one which indicates smooth deformations.

### 5.3.2 Generic Similarity Metrics

We used three generic similarity metrics for evaluating the results: normalized mutual information (NMI), mean squared error (MSE), and mean absolute error (MAE). The results can be seen in Figure 5.1 and Table 5.2. As can be seen from the results, all the methods improve significantly compared to the initial rigid registration.

The model using normalized MSE performs better than the model using MSE on all other metrics except for the pseudo-CT to CT MSE metric. It can be considered a surprise that the MSE model has even slightly worse mean result for the MSE metric in the CT to pseudo-CT direction. It indicates that the method has probably failed to converge.

NiftyReg can be considered to have the best results for the NMI metric as it has significantly lower variance than the other two methods. Since NiftyReg optimizes over normalized mutual information, the result is not surprising. However, the mean values for the DL Normalized MSE model and NiftyReg are about the same, with the DL Normalized MSE model having even slightly better mean NMI for the pseudo-CT to CT registration. We remind that the results are not directly comparable due to different regularization terms. None the less, the results being in the same scale is a good indicator.

The worst outliers visible for Deep Learning models are all the same image pairs for which all registration methods fail to find plausible transformation. However, NiftyReg manages to fit the body outline successfully, resulting in significantly better numbers especially for the MSE.

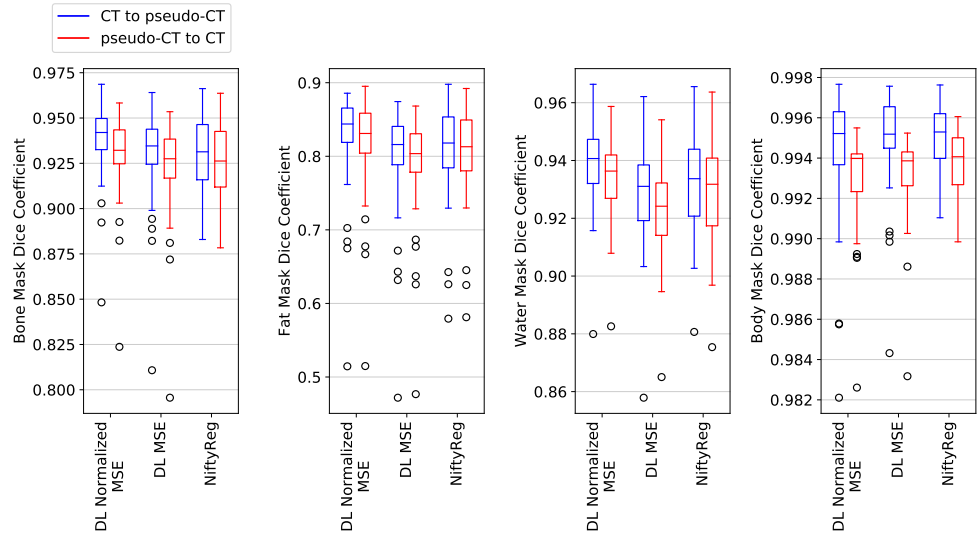
### 5.3.3 Tissue Mask Metrics

Thresholding-based tissue mask metrics results can be seen in Table 5.3 and in Figure 5.2. Again we can see that all the methods perform significantly better than the initial rigid registration.

In terms of the tissue mask metrics, the normalized MSE model performs better than the MSE model and the baseline. The mean values are better for all the mask types, except for the body mask for which the results are essentially identical. Again, direct comparison to NiftyReg is not meaningful due to different regularization but the results being better is a good indicator.

	Mean NMI	Mean MSE	Mean MAE
pseudo-CT to CT			
DL Normalized MSE	1.61219	0.00250076	0.0108585
DL MSE Loss	1.60019	0.002238	0.0113549
NiftyReg	1.61162	0.00267284	0.0116831
CT to pseudo-CT			
DL Normalized MSE	1.59666	0.00277285	0.0113437
DL MSE Loss	1.58382	0.00278784	0.0120448
NiftyReg	1.60301	0.00286777	0.0121914
Rigid	1.51141	0.0115741	0.0233327

**Table 5.2.** Means of the three generic similarity metrics calculated for the test set images registered using different registration methods.



**Figure 5.2.** Dice coefficients for the different tissue masks calculated for the test set images registered using different registration methods.

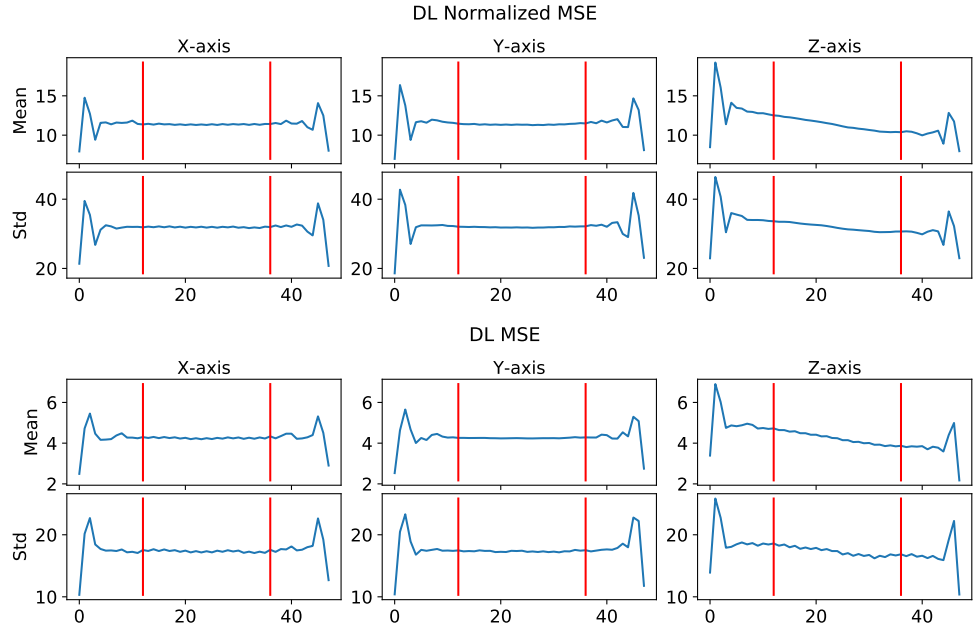
	Bone	Fat	Water	Body
Mean Dice coefficient pseudo-CT to CT				
DL Normalized MSE	0.930839	0.822708	0.936116	0.993538
DL MSE	0.925338	0.795413	0.925483	0.993706
NiftyReg	0.924734	0.810471	0.929381	0.994233
Mean Dice coefficient CT to pseudo-CT				
DL Normalized MSE	0.940477	0.833142	0.941176	0.995133
DL MSE	0.932754	0.805392	0.930842	0.995561
NiftyReg	0.92834	0.814012	0.931968	0.995454
Rigid	0.885745	0.722414	0.891636	0.977985

**Table 5.3.** Mean dice coefficients for the different tissue masks calculated for the test set images registered using different registration methods.

#### 5.4 Patch-wise Approach Consistency

To evaluate whether discarding 12 voxels from the patch borders is enough, we calculated voxel-wise mean and standard deviation of all predicted test set momentum patches, as discussed in Section 4.2. The results can be seen in Figure 5.3. As can be seen, border anomalies do not seem to reach over 12 voxels from the patch borders.

However, few other anomalies can be seen in the plots. The linear ramp along the z-axis is probably due to used image volumes, which result in empty voxels more often being in the top part of the patches. Additionally, there is slight oscillation present along the x and the z axes. Relative magnitude of the oscillation in the mean is from around 2% to 3% and is thus practically not seen in the final deformations after smoothing and velocity field integration. A possible reason for it might be the mistake in the up-sampling step, as described in Section 3.3.1.



**Figure 5.3.** Voxel-wise mean and standard deviation of vector lengths over all the predicted test set momentum patches plotted as marginals for the two used Deep Learning models. Only the  $24 \times 24$  center-most voxels along each axis are used for calculating the marginals since those are used in the prediction phase. The vertical red bars indicate the center-part along the plotted axis.

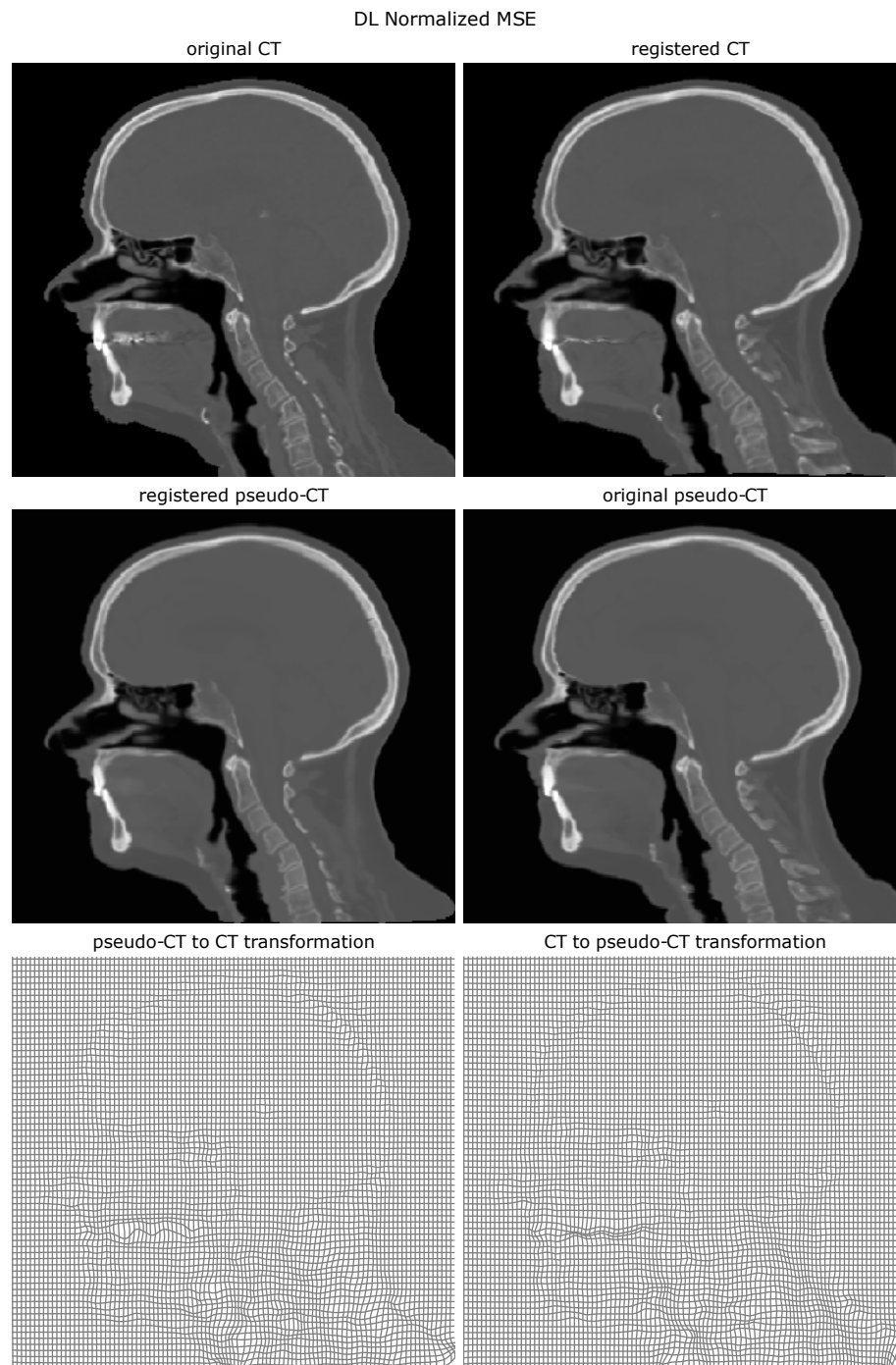
## 5.5 Visual Evaluation

In addition to the metrics evaluation, we performed visual analysis of the results. The upcoming remarks are qualitative in nature and thus can not be confirmed by the reader based on the results provided in this work, except for the few examples. However, we see them as providing valuable insight for anyone analyzing this work and they are hence included.

In general, visual analysis confirms well the result that using the normalized MSE loss improved the result compared to the ordinary MSE loss. This is especially evident for the bone and soft tissue accuracy. An obvious example can be seen in Figures 5.4 and 5.5.

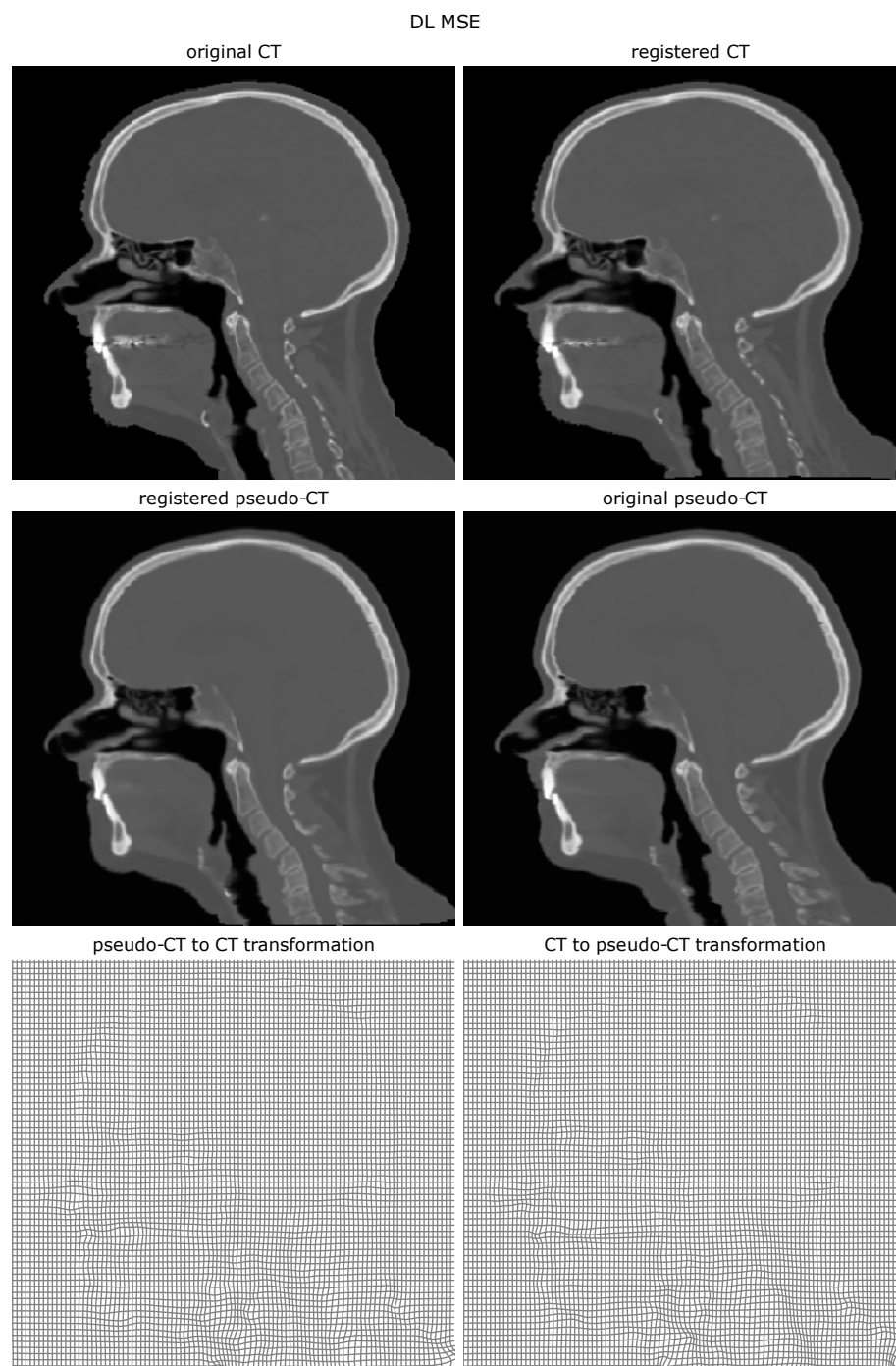
The proposed method does not always handle the volume border areas well. This might be the case since the patches on the border of the volumes include the border anomalies even after fusing the patches. In principle that should not be a problem, since in that case the patch border and volume border are the same and thus the network should have learned to predict exactly the desired momentum. However, convolutional neural networks are not naturally good for making location variant predictions even though matching volume border as a feature is indirectly possible. Based on Figure 5.3, the border anomalies do not seem to be very consistent.

The method does not always catch large scale movements well, either. The basic problem is that even though using the methods we are able



**Figure 5.4.** A test set image pair registered both ways using the Deep Learning model trained using the normalized MSE loss. The image pair is same as in Figure 5.5. Registration accuracy for the spinal cord is significantly better than for the ordinary MSE model. The medical images are Courtesy of Turku University Hospital / Department of Oncology & Radiotherapy.

to match specific features, large scale movements are not matched as smoothly as would be possible. This is not surprising, since only  $48 \times 48 \times 48$  voxel patches are seen at a time and combination of locally optimal transformations is not necessary the globally optimal one. On the other hand, on some occasions the algorithm is able to match large movements quite well.



**Figure 5.5.** A test set image pair registered both ways using the Deep Learning model trained using the ordinary MSE loss. The image pair is the same as in Figure 5.4. Registration accuracy for the spinal cord is significantly worse than for the normalized MSE model. (Courtesy of Turku University Hospital / Department of Oncology & Radiotherapy)

## 6. Discussion and Conclusions

In this work, we have provided unsupervised Deep Learning diffeomorphic registration framework which works on image patches. The work builds on the work by Dalca et al. [10] and Shen et al. [55], bringing patch-wise approach as a new feature allowing for significantly larger volumes to be registered.

The framework is shown to be consistent in that it provides diffeomorphic transformations between whole volumes. Using the momentum static velocity field approach for generating the transformations from the patches results in smooth and invertible deformations over the patch borders.

Accuracy of registration is evaluated by comparing similarity metrics against a baseline registration method, NiftyReg [70]. The method is shown to provide similar scale results as the baseline. Especially the model using the normalized MSE loss described in Equation (3.5) performs well in this aspect.

The main principal weakness of the approach is in our opinion the handling of the momentum field smoothing. Velocity field is calculated from the momentum field by applying a smoothing kernel to it. This introduces a problem for patch-wise training, since smoothing kernel does not have the values of the neighboring patches. The problem is solved by using only center-part of the patches during prediction and this solution is shown to be adequate for solving the problem in practice. However, a more advanced way of calculating the velocity field from some initial transformation representation would allow for a more robust algorithm and might also improve handling of the volume borders. In some sense an optimal approach would be one where the patch-wise prediction would find locations to match between the images and the fusion algorithm seeing the larger picture would find the optimal transformation connecting them.

One possible way to do it would be to teach a network to predict directly

displacement field points along with confidence measure for each. A separate step would then be to calculate a final displacement field prediction based on the initial prediction and the used regularization. Intermediate velocity field representation could also be used in the process to obtain diffeomorphic transformations. Calculating the final displacement field would be a separate task for which Deep Learning might be also suitable.

Another problem the approach has is inherent to the patch-wise approach: the lack of context. We used only rigidly registered images, whereas in patch-wise approaches usually multi-scale deformable registration approach is employed. We proved that in spite of that, our approach was able to create diffeomorphic transformations between the images. However, nothing prevents one from using a multi-scale approach together with the framework provided here. It is likely that it would help the algorithm to see larger scale movement in some instances and also provide more accurate registration as smaller movement would be left in the patch-wise registration phase. That would also solve at least partially the problem of patch fusing presented above as the magnitude of movement inside the patches would be smaller. A problem with multi-scale approach is that it tends to create unwanted stretching, for example, within bones which the lower-scale steps then have to fix. The strength of our approach is not having to do that.

The rationale of registering pseudo-CT and CT images is to provide multimodal registration framework between CT and MRI images. Using Deep Learning, one could train a network to predict transformations between CT and MRI images directly and only use the pseudo-CT images in the training phase for calculating the loss function. Initial experiments to evaluate the concept were done during the thesis work and it was shown to work on a conceptual level.

## 7. Summary

In this work we introduce a new patch-wise diffeomorphic approach for unsupervised unimodal Deep Learning registration. The used methods are closest to the works by Dalca et al. (2019) [10] and Shen et al. (2019) [55] which both provide diffeomorphic transformations using unsupervised Deep Learning. The new concept is the patch-wise prediction of the transformations allowing GPU acceleration for larger resolution image pairs.

The basic idea in applying Deep Learning to registration is to train a network which can predict deformations between image spaces in one shot, when given the set of images to be registered. This is in contrast to the classical registration approach where the deformation is updated iteratively for each new image pair.

In unsupervised registration the network is trained without using ground truth registrations. This is done by using the sum of a similarity term and a regularization term as a loss function for the network. In this aspect the approach is similar to classical registration methods which are typically formulated as optimization problems.

As a diffeomorphic framework we use static velocity field approach originally suggested by Arsigny et al. (2006) [39] and applied in practice by Ashburner (2007) [8]. We combined it with large deformation diffeomorphic metric mapping (LDDMM) framework, as was done for example by Hernandez et al. (2009) [41].

Instead of predicting a static velocity field, we predict a momentum field as was done by Shen et al. [55]. From a momentum field one can generate a velocity field by applying a smoothing convolution kernel. Using the momentum representation allows for regularization to flow over patch borders, the concepts discussed by Yang et al. (2017) [65] in the context of supervised Deep Learning.

We apply the method for registration of computer tomography (CT) and pseudo-CT images. Pseudo-CT images are CT-like images generated from magnetic resonance images. We prove that our diffeomorphic patch-wise approach manages to generate diffeomorphic transformations between the volumes while providing registration accuracy in terms of the used similarity metrics comparable to the used baseline method, NiftyReg [70]. Visual analysis shows good results in general while having some difficulties on volume borders and in matching large scale movement accurately. Thus the work opens up possibilities for fast and robust multimodal registration of CT and magnetic resonance images.

# Bibliography

- [1] C. Broit, “Optimal registration of deformed images”, 1981.
- [2] R. Bajcsy and C. Broit, “Matching of deformed images”, in *Sixth International Conference on Pattern Recognition (ICPR’82)*, 1982, pp. 351–353.
- [3] R. Bajcsy, R. Lieberman, and M. Reivich, “A computerized system for the elastic matching of deformed radiographic images to idealized atlas images.”, *Journal of computer assisted tomography*, vol. 7, no. 4, pp. 618–625, 1983.
- [4] M. F. Beg, M. I. Miller, A. Trounev, and L. Younes, “Computing large deformation metric mappings via geodesic flows of diffeomorphisms”, *International journal of computer vision*, vol. 61, no. 2, pp. 139–157, 2005.
- [5] J. A. Maintz and M. A. Viergever, “An overview of medical image registration methods”, in *Symposium of the Belgian hospital physicists association (SBPH/BVZF)*, Citeseer, vol. 12, 1996, pp. 1–22.
- [6] A. Sotiras, C. Davatzikos, and N. Paragios, “Deformable medical image registration: A survey”, *IEEE transactions on medical imaging*, vol. 32, no. 7, pp. 1153–1190, 2013.
- [7] T. Vercauteren, X. Pennec, A. Perchant, and N. Ayache, “Diffeomorphic demons: Efficient non-parametric image registration”, *NeuroImage*, vol. 45, no. 1, S61–S72, 2009.
- [8] J. Ashburner, “A fast diffeomorphic image registration algorithm”, *Neuroimage*, vol. 38, no. 1, pp. 95–113, 2007.
- [9] B. B. Avants, C. L. Epstein, M. Grossman, and J. C. Gee, “Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain”, *Medical image analysis*, vol. 12, no. 1, pp. 26–41, 2008.

- [10] 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, 2019.
- [11] A. Klein, J. Andersson, B. A. Ardekani, J. Ashburner, B. Avants, M.-C. Chiang, G. E. Christensen, D. L. Collins, J. Gee, P. Hellier, *et al.*, “Evaluation of 14 nonlinear deformation algorithms applied to human brain mri registration”, *Neuroimage*, vol. 46, no. 3, pp. 786–802, 2009.
- [12] Y. Ou, H. Akbari, M. Bilello, X. Da, and C. Davatzikos, “Comparative evaluation of registration algorithms in different brain databases with varying difficulty: Results and insights”, *IEEE transactions on medical imaging*, vol. 33, no. 10, pp. 2039–2065, 2014.
- [13] Y. Fu, Y. Lei, T. Wang, W. J. Curran, T. Liu, and X. Yang, “Deep learning in medical image registration: A review”, *Physics in Medicine & Biology*, 2020.
- [14] Y. Fu, Y. Lei, J. Zhou, T. Wang, S. Y. David, J. J. Beitler, W. J. Curran, T. Liu, and X. Yang, “Synthetic ct-aided mri-ct image registration for head and neck radiotherapy”, in *Medical Imaging 2020: Biomedical Applications in Molecular, Structural, and Functional Imaging*, International Society for Optics and Photonics, vol. 11317, 2020, p. 1 131 728.
- [15] Y. Fu, Y. Lei, J. Zhou, T. Wang, A. Jani, P. Patel, H. Mao, W. J. Curran, T. Liu, and X. Yang, “Non-rigid mri-ct image registration with unsupervised deep learning-based deformation prediction”, in *Medical Imaging 2020: Image Processing*, International Society for Optics and Photonics, vol. 11313, 2020, p. 1 131 329.
- [16] F. P. Oliveira and J. M. R. Tavares, “Medical image registration: A review”, *Computer methods in biomechanics and biomedical engineering*, vol. 17, no. 2, pp. 73–93, 2014.
- [17] J. Ashburner and K. J. Friston, “Nonlinear spatial normalization using basis functions”, *Human brain mapping*, vol. 7, no. 4, pp. 254–266, 1999.
- [18] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors”, *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

- [19] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization”, *arXiv preprint arXiv:1412.6980*, 2014.
- [20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting”, *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [21] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network”, in *2017 International Conference on Engineering and Technology (ICET)*, IEEE, 2017, pp. 1–6.
- [22] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning”, *ArXiv e-prints*, Mar. 2016. eprint: 1603.07285.
- [23] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation”, in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.
- [24] G. E. Christensen and H. J. Johnson, “Consistent image registration”, *IEEE transactions on medical imaging*, vol. 20, no. 7, pp. 568–582, 2001.
- [25] M. Holden, “A review of geometric transformations for nonrigid body registration”, 2007.
- [26] A. Trouné, “An infinite dimensional group approach for physics based models in pattern recognition”, *preprint*, 1995.
- [27] G. E. Christensen, R. D. Rabbitt, and M. I. Miller, “Deformable templates using large deformation kinematics”, *IEEE transactions on image processing*, vol. 5, no. 10, pp. 1435–1447, 1996.
- [28] P. Dupuis, U. Grenander, and M. I. Miller, “Variational problems on flows of diffeomorphisms for image matching”, *Quarterly of applied mathematics*, pp. 587–600, 1998.
- [29] A. Trouné, “Diffeomorphisms groups and pattern matching in image analysis”, *International journal of computer vision*, vol. 28, no. 3, pp. 213–221, 1998.
- [30] L. Younes, “Computable elastic distances between shapes”, *SIAM Journal on Applied Mathematics*, vol. 58, no. 2, pp. 565–586, 1998.
- [31] M. I. Miller and L. Younes, “Group actions, homeomorphisms, and matching: A general framework”, *International Journal of Computer Vision*, vol. 41, no. 1-2, pp. 61–84, 2001.

- [32] M. I. Miller, A. Trouvé, and L. Younes, “On the metrics and euler-lagrange equations of computational anatomy”, *Annual review of biomedical engineering*, vol. 4, no. 1, pp. 375–405, 2002.
- [33] —, “Geodesic shooting for computational anatomy”, *Journal of mathematical imaging and vision*, vol. 24, no. 2, pp. 209–228, 2006.
- [34] F.-X. Vialard, L. Risser, D. Rueckert, and C. J. Cotter, “Diffeomorphic 3d image registration via geodesic shooting using an efficient adjoint calculation”, *International Journal of Computer Vision*, vol. 97, no. 2, pp. 229–241, 2012.
- [35] N. Singh, J. Hinkle, S. Joshi, and P. T. Fletcher, “A vector momenta formulation of diffeomorphisms for improved geodesic regression and atlas construction”, in *2013 IEEE 10th International Symposium on Biomedical Imaging*, IEEE, 2013, pp. 1219–1222.
- [36] T. Schmah, L. Risser, and F.-X. Vialard, “Diffeomorphic image matching with left-invariant metrics”, in *Geometry, Mechanics, and Dynamics*, Springer, 2015, pp. 373–392.
- [37] R. I. McLachlan and S. Marsland, “N-particle dynamics of the euler equations for planar diffeomorphisms”, *Dynamical Systems*, vol. 22, no. 3, pp. 269–290, 2007.
- [38] L. Risser, F.-X. Vialard, R. Wolz, D. D. Holm, and D. Rueckert, “Simultaneous fine and coarse diffeomorphic registration: Application to atrophy measurement in alzheimer’s disease”, in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2010, pp. 610–617.
- [39] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, “Log-euclidean metrics for fast and simple calculus on diffusion tensors”, *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 56, no. 2, pp. 411–421, 2006.
- [40] J.-P. Thirion, “Image matching as a diffusion process: An analogy with maxwell’s demons”, 1998.
- [41] M. Hernandez, M. N. Bossa, and S. Olmos, “Registration of anatomical images using paths of diffeomorphisms parameterized with stationary vector field flows”, *International Journal of Computer Vision*, vol. 85, no. 3, pp. 291–306, 2009.

- [42] L. A. Schwarz, “Non-rigid registration using free-form deformations”, *Technische Universität München*, 2007.
- [43] D. Rueckert, P. Aljabar, R. A. Heckemann, J. V. Hajnal, and A. Hammers, “Diffeomorphic registration using b-splines”, in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2006, pp. 702–709.
- [44] Y. Choi and S. Lee, “Injectivity conditions of 2d and 3d uniform cubic b-spline functions”, *Graphical models*, vol. 62, no. 6, pp. 411–427, 2000.
- [45] F. P. Oliveira and J. M. R. Tavares, “Medical image registration: A review”, *Computer methods in biomechanics and biomedical engineering*, vol. 17, no. 2, pp. 73–93, 2014.
- [46] G. P. Penney, J. Weese, J. A. Little, P. Desmedt, D. L. Hill, *et al.*, “A comparison of similarity measures for use in 2-d-3-d medical image registration”, *IEEE transactions on medical imaging*, vol. 17, no. 4, pp. 586–595, 1998.
- [47] B. B. Avants, N. J. Tustison, G. Song, P. A. Cook, A. Klein, and J. C. Gee, “A reproducible evaluation of ants similarity metric performance in brain image registration”, *Neuroimage*, vol. 54, no. 3, pp. 2033–2044, 2011.
- [48] Q. R. Razlighi, N. Kehtarnavaz, and S. Yousefi, “Evaluating similarity measures for brain image registration”, *Journal of visual communication and image representation*, vol. 24, no. 7, pp. 977–987, 2013.
- [49] J. P. Pluim, J. A. Maintz, and M. A. Viergever, “F-information measures in medical image registration”, *IEEE transactions on medical imaging*, vol. 23, no. 12, pp. 1508–1516, 2004.
- [50] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, “Spatial transformer networks”, in *Advances in neural information processing systems*, 2015, pp. 2017–2025.
- [51] M. S. Elmahdy, J. M. Wolterink, H. Sokooti, I. Išgum, and M. Staring, “Adversarial optimization for joint registration and segmentation in prostate ct radiotherapy”, in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2019, pp. 366–374.

- [52] J. Fan, X. Cao, Q. Wang, P.-T. Yap, and D. Shen, “Adversarial learning for mono-or multi-modal registration”, *Medical image analysis*, vol. 58, p. 101545, 2019.
- [53] B. D. de Vos, F. F. Berendsen, M. A. Viergever, M. Staring, and I. Išgum, “End-to-end unsupervised deformable image registration with a convolutional neural network”, in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, Springer, 2017, pp. 204–212.
- [54] B. D. de Vos, F. F. Berendsen, M. A. Viergever, H. Sokooti, M. Staring, and I. Išgum, “A deep learning framework for unsupervised affine and deformable image registration”, *Medical image analysis*, vol. 52, pp. 128–143, 2019.
- [55] Z. Shen, X. Han, Z. Xu, and M. Niethammer, “Networks for joint affine and non-parametric image registration”, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4224–4233.
- [56] M. Niethammer, R. Kwitt, and F.-X. Vialard, “Metric learning for image registration”, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8463–8472.
- [57] J. Zhang, “Inverse-consistent deep networks for unsupervised deformable image registration”, *arXiv preprint arXiv:1809.03443*, 2018.
- [58] B. Kim, J. Kim, J.-G. Lee, D. H. Kim, S. H. Park, and J. C. Ye, “Unsupervised deformable image registration using cycle-consistent cnn”, in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2019, pp. 166–174.
- [59] Y. Fu, Y. Lei, T. Wang, K. Higgins, J. D. Bradley, W. J. Curran, T. Liu, and X. Yang, “Lungregnet: An unsupervised deformable image registration method for 4d-ct lung”, *Medical Physics*, vol. 47, no. 4, pp. 1763–1774, 2020.
- [60] V. Kearney, S. Haaf, A. Sudhyadhom, G. Valdes, and T. D. Solberg, “An unsupervised convolutional neural network-based algorithm for deformable image registration”, *Physics in Medicine & Biology*, vol. 63, no. 18, p. 185017, 2018.
- [61] Z. Jiang, F.-F. Yin, Y. Ge, and L. Ren, “A multi-scale framework with unsupervised joint training of convolutional neural networks for

- pulmonary deformable image registration”, *Physics in Medicine & Biology*, vol. 65, no. 1, p. 015 011, 2020.
- [62] Y. Lei, Y. Fu, J. Harms, T. Wang, W. J. Curran, T. Liu, K. Higgins, and X. Yang, “4d-ct deformable image registration using an unsupervised deep convolutional neural network”, in *Workshop on Artificial Intelligence in Radiation Therapy*, Springer, 2019, pp. 26–33.
  - [63] H. Yu, X. Zhou, H. Jiang, H. Kang, Z. Wang, T. Hara, and H. Fujita, “Learning 3d non-rigid deformation based on an unsupervised deep learning for pet/ct image registration”, in *Medical Imaging 2019: Biomedical Applications in Molecular, Structural, and Functional Imaging*, International Society for Optics and Photonics, vol. 10953, 2019, p. 109531X.
  - [64] T. Fechter and D. Baltas, “One shot learning for deformable medical image registration and periodic motion tracking”, *IEEE Transactions on Medical Imaging*, 2020.
  - [65] X. Yang, R. Kwitt, M. Styner, and M. Niethammer, “Quicksilver: Fast predictive image registration—a deep learning approach”, *NeuroImage*, vol. 158, pp. 378–396, 2017.
  - [66] S. Wang, S.-Y. Hu, E. Cheah, X. Wang, J. Wang, L. Chen, M. Baikpour, A. Ozturk, Q. Li, S.-H. Chou, *et al.*, “U-net using stacked dilated convolutions for medical image segmentation”, *arXiv preprint arXiv:2004.03466*, 2020.
  - [67] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, “Tensorflow: A system for large-scale machine learning”, in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.
  - [68] C. Studholme, D. L. Hill, and D. J. Hawkes, “An overlap invariant entropy measure of 3d medical image alignment”, *Pattern recognition*, vol. 32, no. 1, pp. 71–86, 1999.
  - [69] P. Hasgall, F. Di Gennaro, C. Baumgartner, E. Neufeld, M. Gosselin, D. Payne, A. Klingeböck, and N. Kuster, “It’s database for thermal and electromagnetic parameters of biological tissues”, *Version 4.0*, 2018.
  - [70] M. Modat, P. Daga, D. Cash, and S. Ourselin, *Niftyreg*, 2010.

- [71] M. Modat, G. R. Ridgway, Z. A. Taylor, M. Lehmann, J. Barnes, D. J. Hawkes, N. C. Fox, and S. Ourselin, “Fast free-form deformation using graphics processing units”, *Computer methods and programs in biomedicine*, vol. 98, no. 3, pp. 278–284, 2010.