Model-Driven Unsupervised Medical Image Registration

By

Xi Jia

A thesis submitted to
the University of Birmingham
for the degree of
DOCTOR OF PHILOSOPHY

School of Computer Science
College of Engineering and Physical Sciences
University of Birmingham
September 2023

ABSTRACT

In this thesis, we study pairwise medical image registration, a process that involves determining the optimal spatial deformation between two medical images. While traditional image registration methods relied on iterative optimization, recent years have witnessed a significant shift towards modern deep data-driven techniques. Within this context, this thesis focuses on unsupervised neural network-based registration approaches, with a primary emphasis on integrating prior knowledge derived from model-based strategies into network architectures to enhance their learning capabilities. More specifically, we address three critical research challenges currently facing the medical image registration community. The first challenge centers on whether deep data-driven registration methods can achieve higher registration accuracy. The second challenge concerns enhancing the data efficiency of deep registration methods. The third challenge involves accelerating the speed of deep registration methods, in terms of both training and inference.

To address the first challenge, we begin by offering a comprehensive guide to constructing a basic deformable registration method based on the U-Net architecture. Alongside this fundamental model, we then introduce three novel variants: Large Kernel U-Net (LKU-Net), which increases the network's effective receptive field through multiple parallel convolutional layers to capture finer spatial structures from the input data; Cascaded LKU-Net, which stacks multiple base LKU-Nets and thereby effectively manages large deformations; and LKU-Net-Affine, which incorporates multiple fully connected modules to learn transformation-specific parameters, enabling adaptability to parametric rigid and affine registration tasks. These variants empower the U-Net model to address both parametric and deformable registration and effectively capture complex deformations that may present in high-resolution medical image data, such as 3D expansion microscopy images and 3D lung CT images.

To address the second challenge, we introduce VR-Net, a variational registration network that unrolls the mathematical structure of iterative variational optimization through variable splitting and seamlessly integrates it into a deep neural network in a cascading fashion. VR-Net is designed to inherit prior knowledge drawn from model-based solvers, ensuring the preservation of their data efficiency and generalizability while maintaining the fast speeds of learning-based approaches. Extensive experiments conducted on 2D and 3D cardiac MRI datasets demonstrate that VR-Net outperforms both traditional model-based and deep data-driven approaches in terms of registration accuracy while retaining fast inference speed and data efficiency.

To address the third challenge, we present Fourier-Net, a novel approach that predicts a compact, low-dimensional representation of the deformation in the band-limited Fourier space. Within Fourier-Net, we propose a model-driven decoder to effectively reconstruct the full-resolution deformation field from the band-limited coefficients. We then introduce Fourier-Net+, an extension of Fourier-Net, which learns the band-limited deformation field from the band-limited representation of images. Fourier-Net+ further accelerates the registration speed by constraining both the input and output of the network to low-dimensional representations, reducing the need for repeated convolution operations. Fourier-Net and Fourier-Net+ are evaluated on cardiac and brain MRI registration tasks in both 2D and 3D scenarios, demonstrating significant reductions in multiply-addition operations, memory footprint, and CPU runtime.

DEDICATION

To my grandfather, who passed away in 2020.

To my grandmother, who passed away in 2022.

ACKNOWLEDGMENTS

I am grateful to Dr. Jinming Duan, my lead supervisor. It has been a great honor to have the privilege of being his first student. He has provided kind support to not only my studies but also my life over the past four years. This thesis would not have been possible without his insightful inspiration, invaluable discussions, and continuous feedback.

I am grateful to Prof. Iain Styles, my co-supervisor, and Prof. Peter Tino, my research monitoring member, for their guidance and encouragement during this research.

I am grateful to Prof. Mattias Heinrich and Dr. Alexander Krull, my thesis examiners, for their detailed comments and suggestions.

I am grateful to my collaboration researchers and fellow colleagues, especially Prof. Declan P. O'Regan and Dr. Antonio de Marvao for providing the 3D cardiac data, Prof. Daniel Rueckert and Huaqi Qiu for their expert technical support, Prof. Linlin Shen, Dr. Hyung Jin Chang, Dr. Wei Chen, Dr. Siyang Song, Alexander Thorley, Joseph Bartlett, Tianyang Zhang, XinXing Cheng, and Chenguang Xiao for their various contributions.

I am grateful to the University of Birmingham and the China Scholarship Council, for their financial support.

I am grateful to BEAR and Baskerville Tier 2 HPC services, operated by Advanced Research Computing at UoB, for providing the computational resources.

I am grateful to my parents, sister, and fiancée, for their unwavering love.

Contents

Page

List of Figures

List of Tables

Acronyms

CNN Convolutional Neural Network.

CT Computerized Tomography.

DFT Discrete Fourier Transformation.

iDFT Inverse Discrete Fourier Transformation.

MRI Magnetic resonance imaging.

Mult-adds Multiply-Addition Operations.

NCC Normalized Cross-Correlation.

NMI Normalized Mutual Information.

SSD the Sum of Squared Difference.

SVF Stationary Velocity Fields.

Chapter One

Introduction

1.1 Medical Image Registration

Medical images have been extensively used within diverse healthcare applications, serving multifarious purposes such as disease diagnosis, disease progression monitoring, treatment planning [33], and treatment guidance [65]. In most of these applications, multiple images are acquired from subjects or the patient cohorts at different times, it is, therefore, crucial to interpret valuable information conveyed in different images [113], explore the common characteristics, and investigate the differences between them. A fundamental step in this process is the registration of the images.

Image registration aims to find an optimal deformation field that establishes the spatial correspondences between two images such that the two images can be well aligned. Specifically, given a moving image $\boldsymbol{I_M}$ and a fixed image $\boldsymbol{I_F}$, the goal of image registration is to find the optimal spatial correspondences (deformation field $\boldsymbol{\phi}$) such that the transformed (warped) moving image $\boldsymbol{I_M} \circ \boldsymbol{\phi}$, is close to the fixed image $\boldsymbol{I_F}$. The whole procedure typically involves two steps: (1) estimating the deformation field ($\boldsymbol{\phi}$) between the moving and fixed images, and (2) deforming (warping) the moving image with the estimated deformation ($\boldsymbol{\phi}$). As figuratively illustrated in Fig. 1.1, given the moving image (which is a 'sad expression') and the fixed image (which is a 'happy expression'), a deformation field is desired such that the warped 'sad expression' becomes the 'happy expression', while the current deformation field is less satisfying as it only generates a 'neutral expression'.

1.1.1 Taxonomy

The field of medical image registration is of profound complexity and can be categorized based on various factors. According to whether the imaging techniques used for moving and fixed images are the same or not, medical image registration can be categorized as either mono-

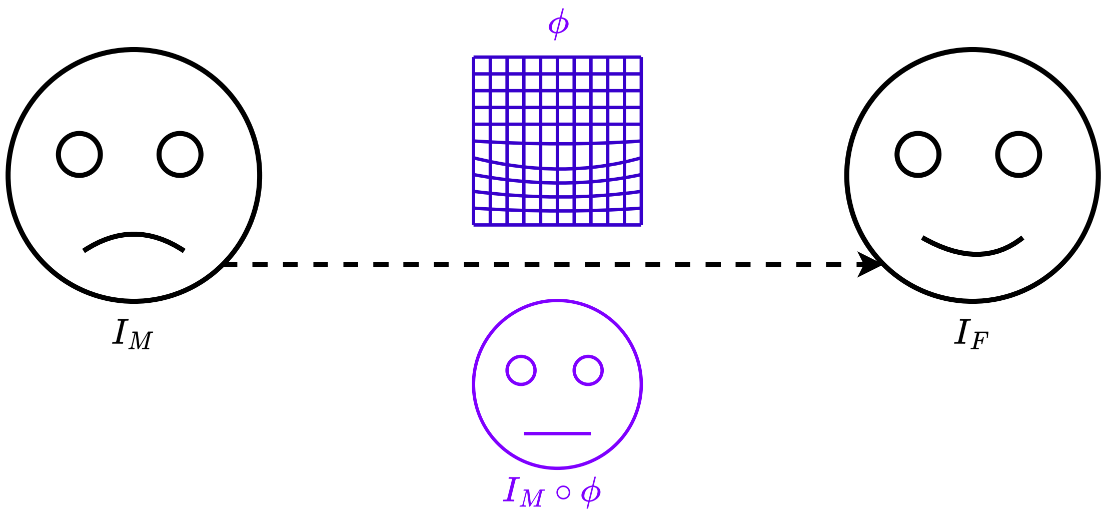


Figure 1.1: Schematic illustration of the image registration process. The goal is to find an optimal deformation ϕ such that the deformed moving image is close to the fixed image I_F, where '$I_M \circ \phi$' denotes the deformed image by ϕ, and '$\circ$' denotes the warping process. The current deformation field ϕ is less satisfying as it only generates a 'neutral expression'.

modality or multi-modality registration. According to the basis of registration, medical image registration can be divided into feature-based and intensity-based registration. Feature-based image registration first extracts the paired corresponding points, lines, surfaces, and other geometric landmarks and then fits a spatial transformation model to these correspondences. Intensity-based approaches, however, do not require additional explicit landmarks before registration, the correspondences are optimized by minimizing the distance based on voxel intensities between the deformed moving image and the fixed image.

According to the nature of transformation, medical image registration can be divided into rigid, affine, and deformable registration. Medical image registration can also be categorized according to the image dimensionality (e.g., 2D, 3D, and 3D temporal data), objects of interest (e.g., brain, lung, and heart), and optimization techniques. For a more comprehensive taxonomy of medical image registration, we suggest that the reader refers to the surveys presented in [99] and [145]. This thesis concentrates on intensity-based approaches for mono-modality deformable registration.

1.1.2 Chanllenges

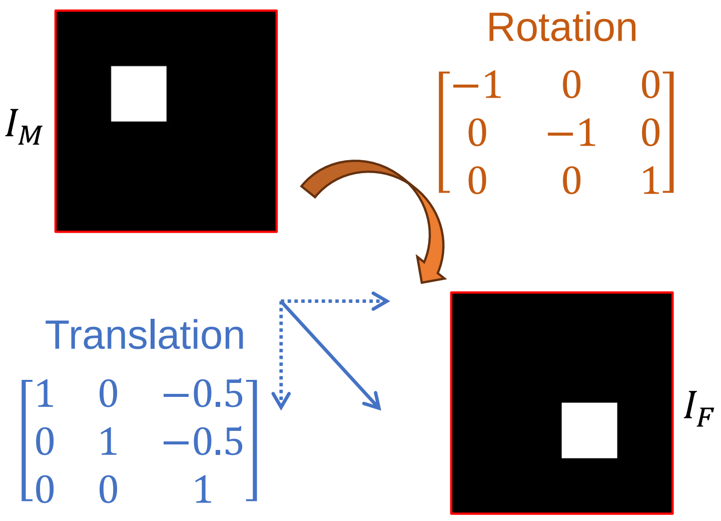
While its definition may appear straightforward, medical image registration remains a very challenging problem due to a number of factors.

Firstly, image registration is ill-posed [108]. In other words, the desired optimal deformation may not be unique, this ambiguity is often caused by the non-unique intensity presented in images [158]. For example, in Fig. 1.2, there are a moving image and a fixed image, the difference between these two images is the location of the 'white' square. The 'white' square is in the upper left corner of the moving image while being the lower right in the fixed one. Due to the specialty of these two images, it is clear that the moving image can be transformed into the fixed image with either a rotation transformation or a translation transformation. These two specific rotation and translation matrices are

$$A_{\text{rot}} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } A_{\text{trans}} = \begin{bmatrix} 1 & 0 & -0.5 \\ 0 & 1 & -0.5 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1.1}$$

respectively. It is evident that these two matrices will result in different deformation fields.

Secondly, there are numerous specific registration applications, and the optimal deformations in demands for different applications are different [100]. Though most registration algorithms are applicable to different organs and anatomical structures, some tasks require task-specific tailoring. For example, in cardiac motion analysis, the displacements should be ideally smooth and small between two consecutive frames, moreover, the displacements are expected to only appear in the heart region, while the background region should have



Rotation
$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
I_M
Translation
$\begin{bmatrix} 1 & 0 & -0.5 \\ 0 & 1 & -0.5 \\ 0 & 0 & 1 \end{bmatrix}$
I_F

Figure 1.2: The optimal deformation field may not be unique. In this specific example, I_M can be transformed into I_F by, but not limited to, a rotation or a translation transformation.

significantly less or zero displacements. However, for subject-to-subject registration on the 3D brain scans, larger displacements are often expected to better align the sophisticated anatomical structures.

Lastly, medical images are usually high-dimension and high-resolution data, registration between such images often takes a large amount of memory usage and computational costs, resulting in a slow inference time. For applications that require real-time registration, one needs to consider balancing the trade-off between computational costs and registration accuracy. It is noticeable that recent advancements in deep learning-based registration have largely accelerated the registration process during inference, while the training process of such deep registration models remains computational-intensive and time-consuming.

1.1.3 Applications

Image registration is an essential step in many medical image analysis applications, in this section, we briefly introduce three of the most common and representative scenarios where image registration is adopted.

Abnormality Detection

Abnormal regions are often caused by some diseases. The detection of abnormality with one single image is usually challenging, radiologists often compare the probe image with other available images from earlier dates for a reliable diagnosis [103]. It is predictable that the manual comparison of these images costs a lot of time, especially when the to-be-inspect images (or scans) are in high dimensions. To accelerate this process, computer-aided pipelines are developed to analyze the significance of differences between two images, mimicking the approach used by radiologists. However, the differences between two different temporal images are not necessarily caused by the actual abnormality, but by other artifacts such as patient movement and viewpoint changes. Therefore, it is necessary to first perform image registration to eliminate or minimize the effects caused by these normal differences [104], in other words, the success of abnormality detection relies on the pre-alignment of different scans achieved by the registration process [55].

Image Segmentation

Manually annotation of anatomical structures for a large cohort is time-consuming, especially when the image to be segmented has sophisticated structures, for example, there are about 40 anatomical structures of interest that need to be manually labeled in a 3D human brain [47]. It is labor-intensive for radiologists to manually annotate all scans in a cohort, therefore, one affordable approach is to first manually label the anatomical structure of a few images, and then adopt a registration algorithm to automatically propagate the labeled annotations to the rest images of the cohort. Such an approach is termed atlas-based segmentation in literature or few-shot (one-shot) segmentation in the context of recent deep registration-based segmentation approaches [151, 165]. By first learning a registration model from the large-scale unlabeled images, such approaches are then able to estimate the deformation field between the atlas and a target image with the registration model. Subsequently, the manually segmented mask of the atlas can be mapped to the target image by warping the mask with the estimated deformation field.

Motion Estimation

Image registration estimates the deformation field between two images, motion estimation, however, estimates the deformation fields between any two consecutive frames in a sequence [70]. By extending the registration from two static images into a temporal sequence, motion estimation can capture the movement of individual objects and track the object of interest efficiently. Motion estimation is usually involved in studies like estimating the cell motion in microscopy images or estimating the cardiac motion in an ultrasound or MRI circle. For example, myocardial motion is the key to differentiating normal and abnormal conditions in cardiovascular disease diagnosis [39]. However, for a single cardiac MRI study, it takes an average of 30 minutes for a cardiologist to manually contour the cardiac chambers in a cardiac cycle, more advanced analyses such as the calculation of ventricular strain and strain rates will require even more time. Therefore, the automatic and accurate estimation of cardiac motion lays a foundation for advanced analysis.

1.2 Research Questions

Medical image registration is a diverse topic [99, 145], while the thesis is limited to unsupervised, intensity-based, mono-modality image registration. Intensity-based registration is historically tackled by iterative optimization-based approaches [5–7, 13, 129, 130, 143]. By first converting the image registration to an optimization problem, such approaches then provide clear derivations that solve the established optimization problem. Iterative approaches are known for their strong and clear mathematical theory while producing the final deformation field can be computationally costly and slow, due to the multi-step and iterative optimization towards the final solution. Moreover, the optimization process has to be solved for each individual pair, which may involve pair-wise hyperparameter tuning, further limiting their applications to large-scale registration.

To overcome the slow registration process of optimization based approaches, deep data-driven models have been widely studied in recent years [23, 128]. Instead of designing a sophisticated optimization problem and solving it with clear derivations altogether, a deep registration model is desired to automatically learn the displacement field from a large amount of training data. Deep data-driven models include two significant stages: training and testing. During the training stage, the deep network takes a large amount of moving and fixed image pairs and estimates the deformation fields that minimize a training objective. Once the model is trained, the network can directly predict the displacement field for a pair of testing images. In other words, the estimation of a displacement field in deep data-driven models can only take one feedforward propagation, which is in stark contrast to the iterative optimization in model-based approaches.

On the other hand, though the inference stage of deep registration models can be orders of magnitude faster than their optimization-based counterparts, it is necessary to point out that the training process of deep registration models usually takes more time and computational resources. Moreover, it is noteworthy that the success of deep registration models relies on the training stage, to fully explore the capacity of a deep registration model, it usually requires a large amount of training data and many iterations of parameter updating.

1.2.1 Reduce the Data Dependency

Optimization-based registration approaches can yield satisfying results for each individual pair through proper hyperparameter tuning. These models leverage significant prior knowledge about the deformation field during their design process. In contrast, the deep registration model is randomly initialized, which does not explicitly include any prior knowledge about the deformation field. In other words, deep registration is purely data-driven and its performance largely depends on the amount of training data. To this end, **the first research question we ask is can we unify the two approaches into one framework such that the combined model can take advantage of both sides?**

We expect that by combining the data-driven deep network and iterative optimization approach, the new model-driven deep registration model will be able to produce comparable results with each approach while only requiring limited data for training, reducing the data dependencies of the deep data-driven approach. Moreover, the new model-driven deep registration model is expected to exhibit the same inference speed as the purely deep data-driven registration model, maintaining its competitivity on large-scale registration.

Note that we use the term data-driven [128] to denote the purely deep network-based methods that directly input original images and learn to estimate deformation fields solely from extensive training data. Conversely, a model-driven approach is a learning-based approach that incorporates prior knowledge from traditional registration models or iterative optimization solvers into the network architecture to facilitate the learning process.

1.2.2 Accelerate the Registration Process

On the other hand, it is highlighted that though a deep registration model can be very fast during inference, the training process, however, requires a considerable number of iterations to update the weights of the parameters within the network. The training process is the bottleneck when applying deep registration models to different scenarios. Therefore, **the second research question we ask is can we embed some model-based prior knowledge into the deep network to further accelerate the training as well as the inference process of deep registration models?**

We expect that by embedding prior knowledge into the deep network, we will be able to reduce the number of convolutional parameters, computational costs, and memory usage, promoting the usage of deep registration models on real-time scenarios or devices with limited memory and computational resources.

1.3 Methods and Contributions

1.3.1 Deep Affine and Deformable Registration

This thesis first studies one of the most commonly used architectures in deep registration, i.e., convolutional U-Net. With the plain U-Net architecture, we explore how different objective functions, optimizers, amounts of training data, and model sizes affect registration performance. We show that the plain U-Net can already achieve comparable registration accuracy with the traditional iterative optimization-based methods. By replacing the plain convolutional layer with a block comprising multiple parallel sub-layers using various kernel sizes, we further propose a Large Kernel U-Net (LKU-Net) to enhance the effective receptive field of U-Net. LKU-Net is compared with more recent registration models based on vision-transformer and proved to be competitive. Subsequently, we proposed LKU-Net-Affine for rigid and affine registration and Cascaded LKU-Net for estimating large and intricate displacements. We note that the proposed Cascaded LKU-Net was the winning solution of the 3D lung CT registration task on the MICCAI 2022 Learn2Reg challenge, and LKU-Net-Affine ranked second on the ISBI 2023 expansion microscopy registration challenge.

1.3.2 Model-Driven Variational Network

Deep data-driven approaches to image registration can be less accurate than conventional iterative approaches, especially when training data is limited. To address this issue and meanwhile retain the fast inference speed of deep learning, we propose VR-Net, a novel cascaded variational network for unsupervised deformable image registration. Using a variable splitting optimization scheme, we first convert the image registration problem, established in a generic variational framework, into two sub-problems, one with a point-wise, closed-form solution and the other one being a general denoising problem. We then propose two neural layers (i.e. warping layer and intensity consistency layer) to model the analytical solution and a residual U-Net (termed generalized denoising layer) to formulate the denoising problem. Finally, we cascade the three neural layers multiple times to form our VR-Net. Extensive experiments on three (two 2D and one 3D) cardiac MRI datasets show that VR-Net outperforms state-of-the-art deep learning methods on registration accuracy, whilst maintaining the fast inference speed of deep learning and the data efficiency of variational models.

1.3.3 Registration with Band-Limited Representation

U-Net style networks are commonly utilized in unsupervised image registration to predict dense displacement fields in the full-resolution spatial domain. For high-resolution volumetric image data, this process is however resource-intensive and time-consuming. To tackle this challenge, we first propose Fourier-Net, which replaces the U-Net style network's expansive path with a parameter-free model-driven decoder. This results in fewer network parameters, memory usage, and multiply-addition operations (mult-adds). Specifically, instead of directly predicting a full-resolution displacement field in the spatial domain, our Fourier-Net learns a low-dimensional representation of the displacement field in the band-limited Fourier domain. This representation is then decoded by our model-driven decoder to obtain the dense, full-resolution displacement field in the spatial domain. Expanding upon Fourier-Net, we then introduce Fourier-Net+, which takes the band-limited spatial representation of the images as input, instead of their original full-resolution counterparts. This leads to a reduction in the number of convolutional layers in the U-Net style network's contracting path, resulting in a further decrease in network parameters, memory usage, and mult-adds. Finally, to enhance the registration performance, we propose a cascaded version of Fourier-Net+. We evaluate our proposed methods on three datasets, including one cardiac and two brain MRI datasets, comparing them against various state-of-the-art approaches. Our proposed Fourier-Net and its variants achieve comparable results with these approaches while exhibiting faster inference speeds with lower memory footprints and fewer mult-adds.

1.4 Publications

This thesis has led to the following peer-reviewed publications:

- Jia, Xi, et al. "Learning a model-driven variational network for deformable image registration." IEEE Transactions on Medical Imaging Vol. 41. No. 1 Pages: 199-212. 2021.
- Jia, Xi, et al. "U-net vs transformer: Is u-net outdated in medical image registration?." International Workshop on Machine Learning in Medical Imaging. Cham: Springer Nature Switzerland, 2022.

- Jia, Xi, et al. "Fourier-Net: Fast Image Registration with Band-Limited Deformation." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 37. No. 1. 2023.

These three publications correspond to Chapters 4, 5, and 6 of this thesis. The first publication, presented in Chapter 4, compares two popular deep network backbones on two publicly available datasets and shows that the classical U-Net architecture is still competitive with modern transformer-based architectures on deformable image registration.

The second publication which is presented in Chapter 5, describes the deep model-driven VR-Net, which is a combination of a variational model and deep data-driven U-Net. The proposed model-driven variational network outperforms the data-driven U-Net-based methods and exhibits less dependence on the training data.

The third publication proposes the Fourier-Net, described in Chapter 6, which estimates a band-limited representation of the full-resolution deformation field. By embedding Fourier transform (DFT) and inverse Fourier transform (iDFT) as prior knowledge into the network, Fourier-Net is able to discard part of the expansive path in U-Net and therefore be faster than U-Net while maintaining its registration accuracy.

1.5 Thesis Structure

The remainder of this thesis is organized as follows:

Chapter 2 provides the preliminary knowledge commonly associated with deep image registration, including the warping operation, evaluation metrics, regularization terms, optimization techniques, and a brief introduction to convolutional networks.

Chapter 3 starts with a review of the deep registration approaches, which is followed by extensive experiments investigating the influence of different similarity metrics, regularization terms, optimization techniques, and model sizes on registration performance.

Chapter 4 introduces the LKU-Net and compares it with vision transformer-based approaches on deformable registration. Cascaded LKU-Net and LKU-Net-Affine are also introduced in this chapter.

Chapter 5 presents the model-driven variational network (VR-Net) with applications to 2D and 3D cardiac motion estimation.

Chapter 6 presents the model-driven Fourier-Net and Fourier-Net+ that show the computational efficiency over full-resolution U-Net by learning band-limited deformation.

Chapter 7 summarises the key findings of the research, discusses their implications, and provides recommendations for further exploration in this field.

Chapter Two

Deep Registration: Background

2.1 Transformation Model

According to the nature of transformations, a registration algorithm can be categorized into rigid, affine, and deformable registration. Different registration tasks favor different transformation models. For example, in the application of radiological image registration with pathology, assuming a new lesion has developed in the interval of a temporal series of intra-subject scans, one would be able to locate the lesion with rigid transformation if the task is merely lesion detection, while the more complex deformable registration is favored if the task is surgical resection of the lesion and the lesion has displaced surrounding the normal tissues. Therefore the selection criterion for the transformation model is task-specific and depends on the characteristics of the task [30].

2.1.1 Rigid Transformation

Both the 2D rigid and affine transformations from $\mathbb{R}^2$ to $\mathbb{R}^2$ can be formulated as the matrix $\boldsymbol{A}$ in the Homogeneous coordinates,

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.1}$$

where there are six different elements (i.e., a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, and a_{23}), which represent different physical transformations in rigid and affine matrices. By this means, rigid and affine registration is also referred to as parametric registration.

Rigid transformation contains only rotation and translation transformations, therefore $\boldsymbol{A}$

is essentially the composition of the rotation and translation matrices, i.e.,

$$A_{\text{rot}} = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } A_{\text{trans}} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.2}$$

as such the rigid transformation matrix can be rewritten as

$$\boldsymbol{A} = \begin{bmatrix} cos(\theta) & -sin(\theta) & \Delta x \\ sin(\theta) & cos(\theta) & \Delta y \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

in which the four elements (i.e., a_{11}, a_{12}, a_{21}, a_{22}) in Eq. (2.1) are parameterized by the rotation angle θ, while a_{13} and a_{23} denote the translation displacements in the horizontal (x) and vertical (y) directions, parameterized by Δx and Δy. From Eq. (2.3), it is evident that a rigid transformation can be parameterized by 3 parameters. In other words, the degree of freedom for 2D rigid transformation is 3.

We note that for a given point $\boldsymbol{p} = (x,\ y)$, its transformed location $\boldsymbol{p'} = (x',\ y')$ with the aforementioned rigid transformation in the Cartesian coordinates becomes

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} x\ cos(\theta) - y\ sin(\theta) + \Delta x \\ x\ sin(\theta) + y\ cos(\theta) + \Delta y \end{bmatrix}, \tag{2.4}$$

where the transformation involves one matrix-vector multiplication and one vector addition, these two operations can be efficiently performed with one single matrix-vector multiplication with the matrix A in Eq. (2.3) in Homogeneous coordinates, i.e.,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & \Delta x \\ \sin(\theta) & \cos(\theta) & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x\ cos(\theta) - y\ sin(\theta) + \Delta x \\ x\ sin(\theta) + y\ cos(\theta) + \Delta y \\ 1 \end{bmatrix}. \tag{2.5}$$

It is easy to infer that if the rotation angle θ and the translation displacements in both directions are zeros, the rigid transformation matrix in Eq. (2.3) becomes an identity matrix, the transformation becomes an identity mapping and the transformed point $\boldsymbol{p'}$ will be exactly the same as $\boldsymbol{p} = (x,\ y)$.

2.1.2 Affine Transformation

Affine transformation is slightly more complex than rigid transformation as it introduces two additional transformations, i.e., scaling and shearing. For affine transformation, the 2D matrix in Eq. (2.1) can be decomposed into four different matrices including the translation and rotation matrices in Eq. (2.2), a scaling matrix

$$A_{\text{scale}} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.6}$$

and a shearing matrix

$$A_{\text{shear}} = \begin{bmatrix} 1 & h_x & 0 \\ h_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.7}$$

where S_x and S_y are the scaling factors in the x and y directions. h_x and h_y are the shearing factors in the x and y directions.

The shearing matrix in Eq. (2.7) can also be represented with

$$A_{\text{shear}} = \begin{bmatrix} 1 & cot(\theta_{h_x}) & 0 \\ cot(\theta_{h_y}) & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.8}$$

where $cot(\cdot)$ denotes the cotangent of an angle, the $\theta_{h_x} = arccot(h_x)$ and $\theta_{h_y} = arccot(h_y)$ are the shearing angles from the horizontal and vertical axes. For example, when $h_x = h_y = 1$, the shearing angles $\theta_{h_x} = \theta_{h_y} = \frac{\pi}{4}$.

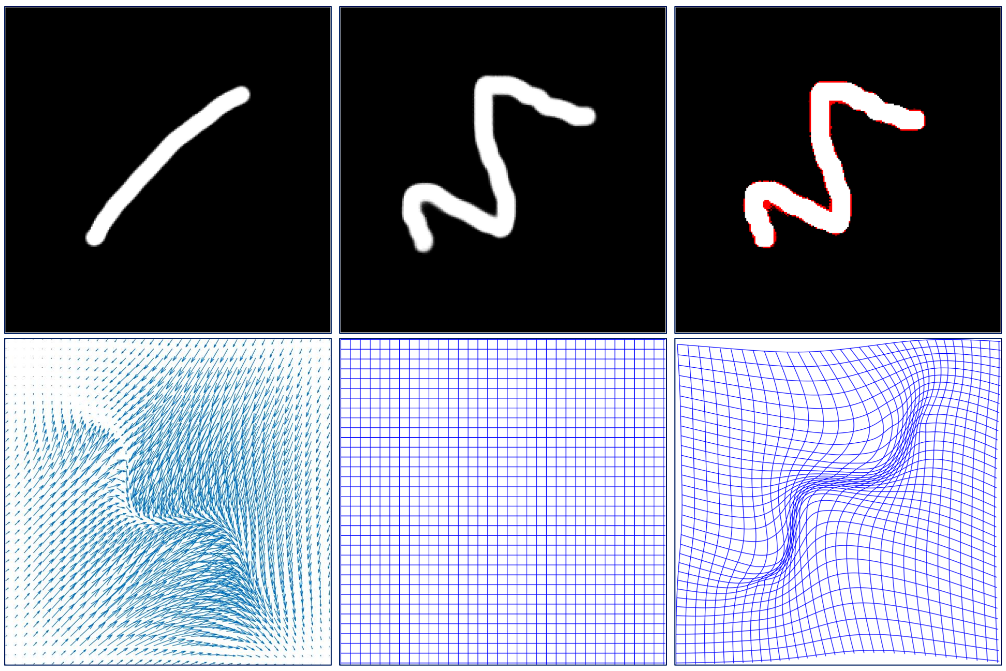


Figure 2.1: Relationship between the displacement and deformation field. Top row: moving image, fixed image, and warped moving image (where the pixels marked with red denote the difference between the warped and fixed images). Second row: displacement field $\boldsymbol{u}(x)$ (where each element is a vector containing the displacements in different directions), identity field $\boldsymbol{Id}$, and the final deformation field $\boldsymbol{\phi}$.

To this end, we have described the 2D rigid and affine transformation matrix, while the 3D rigid and affine transformations from $\mathbb{R}^3$ to $\mathbb{R}^3$ comes with a more complex form, which will be introduced in Section 4.4.

2.1.3 Deformable Transformation

Different from the rigid and affine registration that only estimates a few specific parameters as in Eq. (2.1), deformable registration is non-parametric, as it estimates a dense non-linear full-resolution deformation field $\boldsymbol{\phi}$, which is composed of a full-resolution vectorized displacement field $\boldsymbol{u}(x)$ and an identity map, i.e.,

$$\phi = \boldsymbol{u}(x) + Id, \tag{2.9}$$

where each element in $\boldsymbol{u}(x)$ is a vector that denotes the displacements in horizontal, vertical, and depth (for 3D volumetric input) directions, as illustrated in Fig. 2.1.

2.2 Warping

Once the displacement (deformation field) is produced, a warping operation is adopted to deform the moving image. Since the warped coordinate does not always correspond to the true pixel coordinate, the warping can be implemented in two different ways: one is forward warping while the other is backward warping. A figurative illustration of the two warpings is given in Fig. 2.2. Forward warping pushes the pixels or voxels in a moving image to an intermediate image and then requires interpolation to generate the final warped image. We note that forward warping may not be able to assign an intensity value for every pixel in the new image and may assign multiple values for some pixels. On the other hand, backward warping pulls pixels/voxels from the moving image, and the interpolation is first operated on the coordinate system of the moving image [132, 136]. As such, for every pixel of the warped image, a corresponding coordinate is first computed from the moving image.

As illustrated in Fig. 2.2, the interpolation is inevitable to produce the warped moving image, no matter what direction of warping is used. 'Nearest' and 'bilinear' are two of the most commonly used interpolation techniques for image warping. The former is very straightforward, the intensity of each new point directly copies the intensity of its nearest point in the moving image according to the deformation field, while 'bilinear' interpolation is more complex as it requires integrating the intensities from its nearest four points (2D,

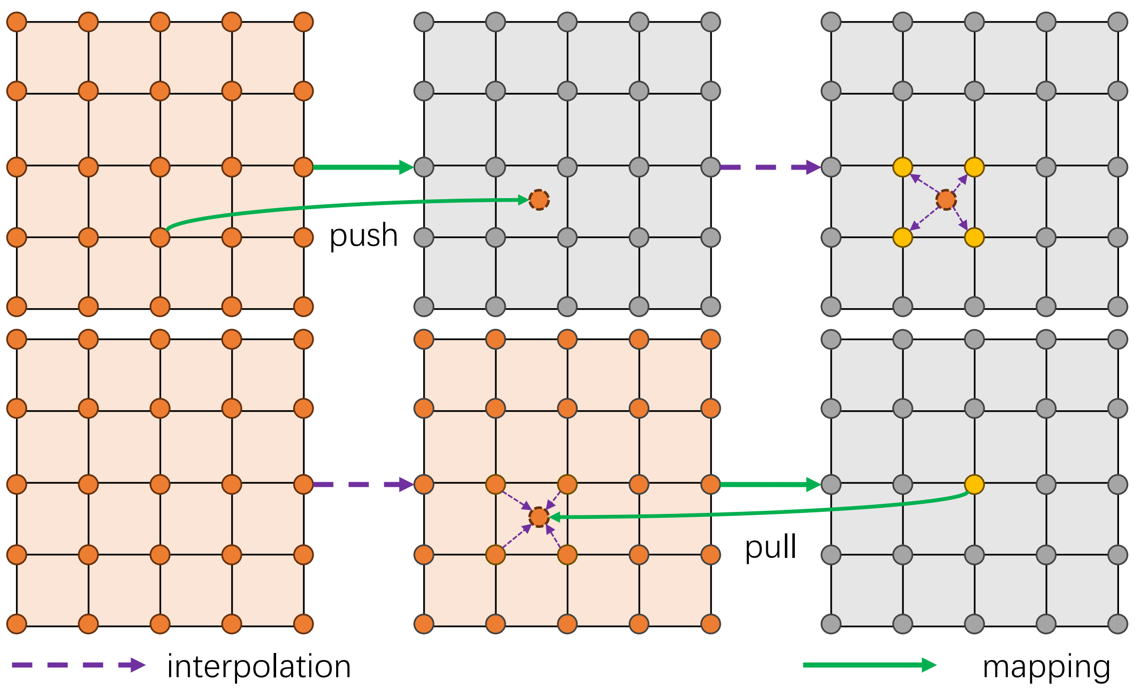


Figure 2.2: Illustration of forward and backward warping. Forward warping pushes the pixels from a moving image to an intermediate image and then adopts interpolation to produce the final warped image. Backward warping pulls the pixels from a moving image in which interpolation is first imposed.

$2^2 = 4$). Mathematically speaking, the bilinear function can be formulated as

$$\begin{aligned} V(x', y') = &\frac{(x_2 - x')(y_2 - y')}{(x_2 - x_1)(y_2 - y_1)} V(x_1, y_1) + \frac{(x_2 - x')(y' - y_1)}{(x_2 - x_1)(y_2 - y_1)} V(x_1, y_2) \\ &+ \frac{(x' - x_1)(y_2 - y')}{(x_2 - x_1)(y_2 - y_1)} V(x_2, y_1) + \frac{(x' - x_1)(y' - y_1)}{(x_2 - x_1)(y_2 - y_1)} V(x_2, y_2), \end{aligned} \tag{2.10}$$

where $V(x, y)$ denotes the intensity value of a given point (x, y). When warping 3D images, the 'bilinear' interpolation, of course, will be extended to 'trilinear' interpolation to integrate the intensities from the adjacent eight points ($2^3 = 8$) for a given point.

For the rigid and affine transformation, instead of enumerating each point using the transformation matrix (similar to what we described in Eqs. (2.4) and (2.5)), the transformation matrix can also be first converted into a dense **linear** displacement field. Then the warping

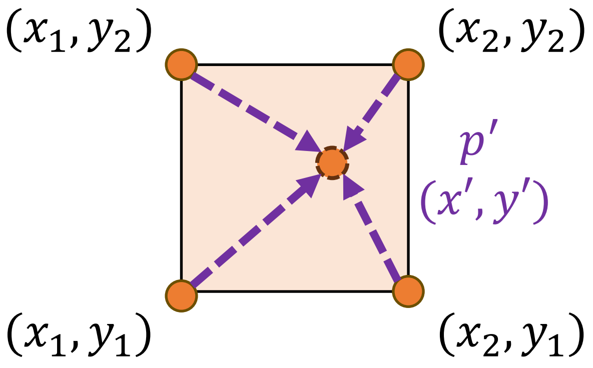


Figure 2.3: Bilinear Interpolation. The intensity of $p' = (x', y')$ is computed from the intensities of four different points.

operation can be imposed with the dense displacement field as described above.

2.3 Objective Function

Unsupervised intensity-based registration includes three core components: 1) a deformation model that seeks the spatial transformation, 2) an objective function that constrains the similarity between the warped moving image and fixed image and regularizes the deformation field, and 3) an optimization function that minimizes the objective function. In the context of deep registration, the deformation model is automatically learned by training the network, the optimization method is usually gradient descent, while the training objective function requires hand-crafted design. A general form of the objective function $\mathcal{L}(I_M, I_F; \phi)$ is

$$\mathcal{L}(I_M, I_F; \phi) = \mathcal{L}_{Data}(I_M(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})), I_F) + \lambda\ \mathcal{L}_{Reg}(\boldsymbol{u}(\boldsymbol{x})), \tag{2.11}$$

where $\mathcal{L}_{Data}(\cdot)$ is the data term, or sometimes termed similarity term in the literature, while $\mathcal{L}_{Reg}(\cdot)$ is the regularization term. The data term measures the distance between the warped moving image $I_M \circ \phi = I_M(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))$ and the fixed image I_F, while the regularization term imposes prior constraints, such as smoothness, over the displacement field $\boldsymbol{u}(\boldsymbol{x})$. The hyperparameter λ is the weight imposed on the regularization term, balancing the trade-off between the two terms.

2.3.1 Data Term

Commonly used data terms in unsupervised deep registration include the sum of absolute differences (SAD), the sum of square differences (SSD), normalized cross-correlation (NCC), and normalized mutual information (NMI). There are many other data terms available, such as Normalized Gradient Fields (NGF) [54] and modality independent neighborhood descriptor (MIND) [60], while this work only experiments with the former four terms.

Based on Euclidean distance, **SAD** and **SSD** directly quantify the pixel-level intensity residual differences. The reader might find that some registration models used mean absolute error (MAE) and mean squared error (MSE) as the data terms, we note that SAD and SSD are essentially the same as MAE and MSE, the only difference is using the 'mean' or 'sum' of all pixels when producing the final loss value. The equations of these two losses are given in Eqs. (2.12) and (2.13),

$$\mathrm{SAD} = \sum_{i=1}^{n} |[I_M \circ \phi](x_i) - I_F(x_i)|, \tag{2.12}$$

$$\mathrm{SSD} = \sum_{i=1}^{n} ([I_M \circ \phi](x_i) - I_F(x_i))^2, \tag{2.13}$$

where n is the number of pixels in the image, $[I_M \circ \phi]$ is the warped moving image. SAD and SSD (or MAE and MSE) losses are commonly used when the moving and fixed image shares the same image intensity distributions and local contrast [10].

Normalized cross-correlation, on the other hand, is more applicable when the two images have locally large intensity variances [6]. NCC is also called the Pearson correlation coefficient, which is defined in Eq. (2.14),

$$\mathrm{NCC} = \frac{\sum_{i=1}^{n}(I_F(x_i) - \bar{I_F})([I_M \circ \phi](x_i) - [\bar{I_M \circ \phi}])}{\sqrt{\sum_{i=1}^{n}(I_F(x_i) - \bar{I_F})^2}\sqrt{\sum_{i=1}^{n}([I_M \circ \phi](x_i) - [\bar{I_M \circ \phi}])^2}}, \tag{2.14}$$

where $\bar{I_F}$ and $\bar{I_M \circ \phi}$ denote the average intensity of the fixed image and warped moving image, for example, $\bar{I_F} = \frac{1}{n}\ \sum_{i=1}^{n} I_F(x_i)$. The value of NCC ranges from -1 to 1, in which the value 1 indicates the perfect alignment of the two images.

In the registration of large-resolution 3D scans, local normalized cross-correlation (LNCC) instead of NCC is usually adopted, which computes the NCC in a local patch instead of the whole image. The formula of LNCC is given in Eq. (2.15),

$$\mathrm{LNCC} = \frac{\sum_{i=1}^{k}(I_F(x_i) - \bar{I_F})([I_M \circ \phi](x_i) - [\bar{I_M \circ \phi}])}{\sqrt{\sum_{i=1}^{k}(I_F(x_i) - \bar{I_F})^2}\sqrt{\sum_{i=1}^{k}([I_M \circ \phi](x_i) - [\bar{I_M \circ \phi}])^2}}, \tag{2.15}$$

where k is the number of pixels (voxels) in the local 2D or 3D patch. Subsequently, $\bar{I_F}$ and $\bar{I_M \circ \phi}$ accordingly denote the average intensity of the fixed patch and warped moving patch, i.e., $\bar{I_F} = \frac{1}{k}\ \sum_{i=1}^{n} I_F(x_i)$.

The loss function becomes $\mathcal{L} = -\mathrm{NCC}(I_F, I_M \circ \phi)$ or $\mathcal{L} = -\mathrm{LNCC}(I_F, I_M \circ \phi)$ when training the deep registration models, as a smaller $\mathcal{L}$ denotes a higher LNCC, which in return indicates a better alignment of images [10].

Based on the information theory, **mutual information** quantifies the amount of information that one image contains about a second image [130]. The formula of Normalized Mutual Information (NMI) is given as

$$\mathrm{NMI} = \frac{H(I_F) + H(I_M \circ \phi)}{H(I_F, I_M \circ \phi)}, \tag{2.16}$$

where $H(I_F), H(I_M \circ \phi)$, and $H(I_F, I_M \circ \phi)$ denote the marginal entropy of I_F, the marginal entropy of $I_M \circ \phi$, and their joint entropy, respectively. Theoretically, the entropy is defined by $H(X) = -\sum_{x \in X} p(x)\log(x)$ for a discrete variable X, where x is all possible values of X.

For an image, the computation of entropy is performed on its histogram, the probabilities are computed according to the intervals or **bins** of the histogram and the number of pixels that fall into different bins. It is evident that the computation based on the histogram is non-differentiable, in order to use the NMI in the training of deep registration models, different approximate techniques are adopted to construct differentiable NMI, such as the Parzen window function [122].

2.3.2 Regularization Term

Due to the ill-posed nature of the image registration problem, optimizing the data term alone may generate unrealistic and implausible deformation fields in order to bring the two images as close as possible. As such, constraint about the deformation field, i.e., the regularization term in Eq. (2.11), is necessary.

Diffusion regularizer, given in Eq. (2.17), is one of the most commonly used smoothness regularization terms.

$$\mathcal{R}(\boldsymbol{u}(\boldsymbol{x})) = \|\nabla \boldsymbol{u}(\boldsymbol{x})\|_2^2, \tag{2.17}$$

where $\boldsymbol{u}(\boldsymbol{x})$ denotes the displacement field and ∇ is the gradient operator. Diffusion regularizer constrains the squared l_2 norm of the first-order gradient of the displacement to be small, which explicitly encourages the learned displacement to be smooth.

Total variation, on the other hand, constrains the l_1 norm of the first-order derivatives of the displacement, which comes with the form

$$\mathcal{R}(\boldsymbol{u}(\boldsymbol{x})) = \|\nabla \boldsymbol{u}(\boldsymbol{x})\|_1. \tag{2.18}$$

Bending energy [77, 130] constrains the second-order gradient of the displacement, with its 2D and 3D formulas shown in Eqs. (2.19) and (2.20),

$$\mathcal{R}(\boldsymbol{u}(\boldsymbol{x})) = \|\nabla^2 \boldsymbol{u}(\boldsymbol{x})\|^2 = \sum_{\boldsymbol{x}} \left[\left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial^2 x} \right)^2 + \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial^2 y} \right)^2 + 2 \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial x \partial y} \right)^2 \right], \tag{2.19}$$

$$\begin{aligned} \mathcal{R}(\boldsymbol{u}(\boldsymbol{x})) = \sum_{\boldsymbol{x}} \Bigg[& \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial^2 x} \right)^2 + \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial^2 y} \right)^2 + \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial^2 z} \right)^2 \\ & + 2 \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial x \partial y} \right)^2 + 2 \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial x \partial z} \right)^2 + 2 \left(\frac{\partial^2(\boldsymbol{u}(\boldsymbol{x}))}{\partial y \partial z} \right)^2 \Bigg]. \end{aligned} \tag{2.20}$$

The regularization term itself is also a hyperparameter in construing the registration objective function, and the selection of regularization terms should also be based on the specific registration tasks.

2.3.3 Consistency

With the objective function defined in Eq. (2.11), giving a moving and fixed image pair ($\boldsymbol{I_M}$ and $\boldsymbol{I_F}$), an image registration model $\boldsymbol{f}$ is ideally able to produce the optimal deformation $\boldsymbol{\phi} = \boldsymbol{f}(\boldsymbol{I_M}, \boldsymbol{I_F})$ such that the warped moving image is close to the fixed image, i.e., $\boldsymbol{I_M} \circ \boldsymbol{\phi} = \boldsymbol{I_F}$. Accordingly, if we swap the order of moving and fixed images when feeding the image pair into $\boldsymbol{f}$, the estimated backward deformation field $\boldsymbol{\phi}'$ will be able to warp the fixed image towards the moving image, i.e., $\boldsymbol{I_F} \circ \boldsymbol{\phi}' = \boldsymbol{I_M}$.

The estimated backward deformation field $\boldsymbol{\phi}'$ is expected to be the inverse of the forward deformation field $\boldsymbol{\phi}$ (i.e., $\boldsymbol{\phi}^{-1} = \boldsymbol{\phi}'$). The inverse consistency between the forward and backward deformation fields suggests the registration model indeed learns unique correspondence between the two input images. The majority of the image registration models, however, can not guarantee the estimated backward deformation field is the inverse of the forward deformation field (i.e., $\boldsymbol{\phi}^{-1} \neq \boldsymbol{\phi}'$), which means further analysis based on these inconsistent image registration models can be affected by the order of inputs.

The inverse inconsistency is usually caused by the fact that the objective function $\mathcal{L}$ defined in Eq. (2.11) has many local minima, resulting in many possible ambiguous displacements locally. This inconsistency is more likely to occur with the increasing complexity and dimensionality of the input images [26]. To overcome the issues brought by the asymmetric registration, more advanced constraints of the deformation fields are explicitly imposed when designing the objective functions of registration models, including symmetry, inverse consistency [26, 161], topology preservation [27], and diffeomorphism [5].

Symmetry: Extending the asymmetric objective function defined in Eq. (2.11) to symmetric is straightforward, all we need to do is enable the estimation of the backward deformation field in the objective function. By explicitly defining and optimizing the backward deformation $\phi' = \boldsymbol{v}(\boldsymbol{x}) + Id$, a symmetric objective function can be defined as

$$\begin{aligned} \mathcal{L}_{sym}(I_F, I_M; \phi, \phi') &= \mathcal{L}(I_M, I_F; \phi) + \mathcal{L}(I_F, I_M; \phi'), \\ \mathcal{L}(I_M, I_F; \phi) &= \mathcal{L}_{Data}(I_M(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})), I_F) + \lambda\ \mathcal{L}_{Reg}(\boldsymbol{u}(\boldsymbol{x})), \\ \mathcal{L}(I_F, I_M; \phi') &= \mathcal{L}_{Data}(I_F(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})), I_M) + \lambda\ \mathcal{L}_{Reg}(\boldsymbol{v}(\boldsymbol{x})). \end{aligned} \tag{2.21}$$

It is noteworthy that though Eq. (2.21) provides the symmetric objective function that jointly optimizes the forward and backward deformation fields, it does not directly constrain the two estimated deformation fields are inverse consistent, as ϕ and ϕ' are independently optimized in Eq. (2.21).

Inverse Consistency: The inverse consistency imposes a stronger constraint over the forward and backward deformation field than the symmetry. Inverse consistency encourages that ϕ and ϕ' are the inverse transformations of each other by minimizing

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{sym} + \lambda_{inv}\ \mathcal{L}_{inv}, \\ \mathcal{L}_{inv} &= \|\boldsymbol{u}(\boldsymbol{x}) - \tilde{\boldsymbol{v}(\boldsymbol{x})}\|^2 + \|\boldsymbol{v}(\boldsymbol{x}) - \tilde{\boldsymbol{u}(\boldsymbol{x})}\|^2, \end{aligned} \tag{2.22}$$

where the hyperparameter λ_{inv} denotes the weight of inverse consistency constraint $\mathcal{L}_{inv}$, $\tilde{\boldsymbol{u}(\boldsymbol{x})}$ and $\tilde{\boldsymbol{v}(\boldsymbol{x})}$ are the inverse displacement field of ϕ and ϕ', such that $\phi^{-1} = \tilde{\boldsymbol{u}(\boldsymbol{x})} + Id$

and $(\phi')^{-1} = \tilde{\boldsymbol{v}(\boldsymbol{x})} + Id$. Though Eq. (2.22) explicitly encourages inverse consistency of the two estimated deformation fields, we note that true consistency is difficult to achieve for real-world applications.

Topology Preservation and Diffeomorphism: Inverse consistency encourages unique correspondence between the input images but does not necessarily preserve the topology of the objects of interest. Because the deformation field estimated from the inverse consistency constraints in Eq. (2.22) may exist local folding, i.e., local negative Jacobian determinants, resulting in the distortion of registered images [26].

Topology persevering registration models, on the other hand, are designed to maintain the local relationships between structures, i.e., adjacent structures remain adjacent after registration [27]. In such models, the Jacobian determinant of the deformation field is greater than zero [136]. Topology preserving transformation is homeomorphic, providing a continuous, onto, locally one-to-one mapping and has a continuous inverse.

Diffeomorphic transformation can be regarded as a stronger transformation than homeomorphic transformation, as it is differentiable and has differentiable inverse [5, 136]. Diffeomorphic registration provides inverse consistency as well as preserves topology, therefore, is a recommended regularization when no prior knowledge is known about the deformations [26]. Computing a diffeomorphic deformation can be treated as modeling a dynamical system [13]: $\partial\phi/\partial t = \mathbf{v}_t(\phi_t)$, where $\phi_0 = Id$ is the identity transformation and $\mathbf{v}_t$ indicates the velocity field at time t ($t \in [0, 1]$). Alternatively, a diffeomorphic deformation can also be modeled with stationary velocity fields (SVFs): $\partial\phi/\partial t = \mathbf{v}(\phi_t)$, where the velocity fields are assumed to remain constant over time [5]. In the context of deep diffeomorphic registration, SVF parameterized deformation is usually estimated by a scaling and squaring approach [31]

$$\phi = \mathbf{Exp}(\mathbf{v}). \tag{2.23}$$

Algorithm 1 Mini-batch stochastic gradient descent (Mini-batch SGD)

```
1:  Inputs : training data $\mathcal{I} = \{(I_{F^{(i)}}, I_{M^{(i)}})\}_{i=1}^{N}$, batch size $b$, and the learning rate $\alpha$.
2:  Initialize : $\phi^0$.
3:  for $n_{iter} = 1 : N_{iter}$ do                     ▷ # $N_{iter}$ is the maximum iterations.
4:      Shuffle the pairs in $\mathcal{I}$.
5:      for $n = 1 : \frac{N}{b}$ do                      ▷ # Divide the training data into different batches.
6:          $\phi = \phi^{n-1} - \alpha \, \frac{1}{b} \, \sum_{i=1}^{b} \frac{\partial \mathcal{L}(I_{F^{(i)}}, I_{M^{(i)}}; \phi^{n-1})}{\partial \phi}$      ▷ # $b = 1$ denotes the original SGD.
7:                                                        ▷ $\mathcal{L}$ is the training objective.
8:      end for
9:      return $\boldsymbol{\phi}^* = \boldsymbol{\phi}$                  ▷ # return the final deformation field.
10: end for
```

2.4 Optimization

To this end, we have defined the basic objective of deep unsupervised deformable registration. To estimate the final displacements, optimization techniques are required. Gradient Descent (GD) is one of the most effective optimization techniques, which optimizes the parameter according to the direction of its negative gradient,

$$\phi^{t+1} = \phi^t - \alpha \frac{\partial \mathcal{L}_{\mathcal{I}}}{\partial \phi} = \phi^t - \alpha \, \frac{1}{N} \sum_{i=1}^{N} \frac{\partial \mathcal{L}(I_{F^{(i)}}, I_{M^{(i)}}; \phi)}{\partial \phi}, \tag{2.24}$$

where $\mathcal{L}$ is the training objective and can be defined as in Eq. (2.11), ϕ^t is the optimized deformation field at the t^{th} iteration, $\mathcal{I} = \{(I_{F^{(i)}}, I_{M^{(i)}})\}_{i=1}^{N}$ is the whole training set, and N is the number of training moving and fixed image pairs, α is the step of optimization and is also called the ***learning rate*** in the context of deep learning.

The gradient descent method presented in Eq. (2.24) is called batch gradient descent, as the parameter is optimized once after enumerating the whole training dataset $\mathcal{I}$, and the final optimization direction is determined by the gradient of all training samples. Apparently, when the dataset has a large number of training samples, batch gradient descent will require lots of computational costs and memory usage.

To overcome the aforementioned shortcomings, stochastic gradient descent (SGD) proposes to update the parameter according to the gradient of one single sample in each iteration. SGD is much faster than batch gradient descent and is widely used for large-scale optimization tasks, however, as it only considers one single sample in each iteration, it may introduce noise during the training process and lead to unstable training. Mini-batch SGD, serving as the trade-off between batch gradient descent and stochastic gradient descent, is proposed to update the parameter with a limited number of samples in each iteration. Mini-batch SGD combines the advantage of both approaches and is expected to ease the computational burden of batch gradient descent and stabilize the training process of stochastic gradient descent. The parameter updating schemes of both SGD and mini-batch SGD are provided in Algorithm 1.

2.5 Evaluation

It is extremely hard to manually annotate the ground truth deformations for large-scale high-dimensional medical image data. Therefore, in order to compare the performance of different registration models, many surrogate measurements have been proposed. Commonly used evaluation metrics include image intensity-based measurements and anatomical structure-based measurements. However, a similar intensity between the warped image and fixed image does not necessarily mean a better registration [124], therefore the Jacobian of displacement field is also reported to provide a comprehensive comparison of the deformation itself.

Intensity-based measurements compute the similarity between the warped moving and fixed image based on the pixel-/voxel-level intensity, such measurements, including SAD,

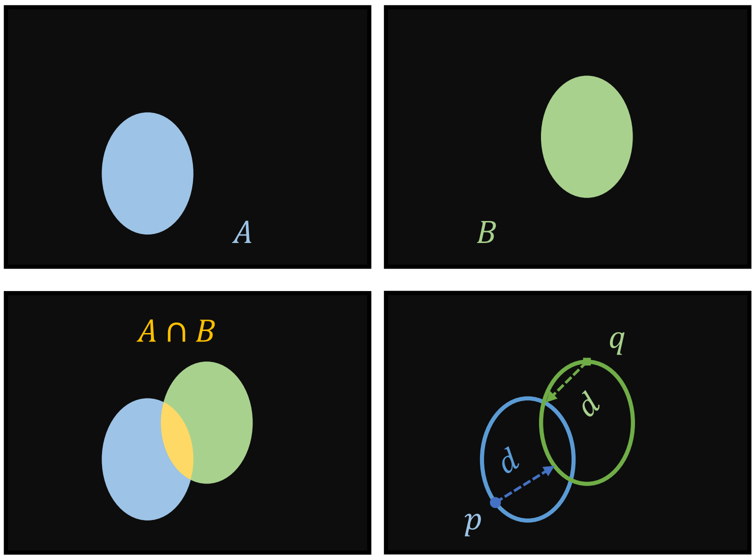


Figure 2.4: Computing Dice and Hausdorff Distance (HDist).

SSD, NCC, and NMI, have been discussed in Section 2.3. This section hereby introduces the anatomical structure-based measurements and the Jacobian of displacement field.

2.5.1 Anatomical Structure Based Measurements

Dice coefficient and Hausdorff Distance (HDist) are two commonly used anatomical structure-based measures. **Dice** score is defined as the intersection ratio between the overlap area and the average area of two masks. If the moving image is perfectly aligned with the fixed image by the estimated deformation, it is assumable that the segmentation mask of the moving image will also be well-aligned with that of the fixed image. In other words, a better registration indicates a higher Dice score. The formula for computing Dice is

$$\text{Dice} = \frac{2\,|A \cap B|}{|A| + |B|}, \tag{2.25}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ denote the mask of the fixed image and the warped mask of the moving image, respectively.

Hausdorff Distance (HDist) [8] evaluates the maximum distance between the contour ∂A of mask A and the contour ∂B of mask B, as listed in Eq. (2.26). Apparently, a better registration indicates a lower HDist distance.

$$\text{HDist} = \max\left(\max_{p \in \partial A} d(p, \partial B), \max_{q \in \partial B} d(q, \partial A)\right), \tag{2.26}$$

where p is a point belongs to the contour ∂A and $d(p, \partial B)$ denotes its euclidean distance to contour ∂B, likewise, q is a point belong to the contour ∂B and $d(q, \partial A)$ denotes its distance to contour ∂A.

It is notable that the computations in both Eqs. (2.25) and (2.26) imply there is only one anatomical structure available in each mask, as illustrated in Fig. 2.4. For more general cases with multiple anatomical structures, the Dice and HDist for each individual structure are first computed and then the averaged scores on all labeled structures are reported.

2.5.2 Jacobian Determinant

Intensity-based and anatomical structure-based measurements cannot distinguish realistic and plausible deformations from unrealistic ones. For example, the 'Completely Useless Registration Tool' (CURT) proposed in [124] can generate indistinguishable warped images, but the displacement fields are randomly sorted. Therefore, metrics that evaluate the displacement itself, such as the number of negative Jacobian determinants, are needed.

Jacobian matrix encodes the local transformation of displacement through the partial derivatives at a point (x_1, x_2, x_3). The Jacobian matrix of 2D deforamtion field is

$$J = \begin{pmatrix} \frac{\partial \phi_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial \phi_1(\boldsymbol{x})}{\partial x_2} \\ \frac{\partial \phi_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial \phi_2(\boldsymbol{x})}{\partial x_2} \end{pmatrix}, \tag{2.27}$$

and the corresponding matrix of the 3D deformation field is

$$J = \begin{pmatrix} \frac{\partial \phi_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial \phi_1(\boldsymbol{x})}{\partial x_2} & \frac{\partial \phi_1(\boldsymbol{x})}{\partial x_3} \\ \frac{\partial \phi_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial \phi_2(\boldsymbol{x})}{\partial x_2} & \frac{\partial \phi_2(\boldsymbol{x})}{\partial x_3} \\ \frac{\partial \phi_3(\boldsymbol{x})}{\partial x_1} & \frac{\partial \phi_3(\boldsymbol{x})}{\partial x_2} & \frac{\partial \phi_3(\boldsymbol{x})}{\partial x_3} \end{pmatrix}, \tag{2.28}$$

where ϕ_1, ϕ_2, ϕ_3 denotes the deformation field along the horizontal, vertical, and depth directions, respectively. The determinant of the Jacobian matrix can then be computed to represent the changes in this region, for example, determinant $\det(J)$ equals 1 represents the local space around this point remains the same. If the absolute value of $\det(J)$ is between zero to 1, i.e., $|\det(J)| \in (0, 1)$, the space around this point is contracting finitely. If $\det(J) = 0$, the space around this point is contracting infinitely. If $|\det(J)| > 1$, the space around this point is expanding finitely. A negative determinant denotes one-to-one mapping is lost, therefore, the number of negative Jacobian determinants in a deformation field is usually adopted as a measurement of the diffeomorphism of a deformation.

2.6 Convolutional Networks

From the prestigious LeNet-5 [87, 88] for handwritten zip code recognition to the deep AlexNet [86] for large-scale natural image classification on ImageNet, convolutional networks have historically been used to extract latent feature maps through the weight-sharing convolutional layer and summarize the global information conveyed in the feature maps through fully connected (FC) layers. The success of AlexNet in large-scale image recognition inspired the renaissance of deep learning approaches in image segmentation [95, 125] and optical flow estimation. The advancements in these fields further promote the emergence of new techniques in deep medical image registration [11, 21, 110, 119, 148].

This section first walks through some representative deep convolutional networks such as LeNet-5 [87, 88], AlexNet [86], and U-Net [125], because their extensions have served as the backbones of the deep registration model. Followed by the introduction of the spatial transformer network [72], which proposed the core of deep registration models, i.e., differentiable spatial transformer module.

2.6.1 LeNet-5 and AlexNet

LeNet-5 [87, 88], AlexNet [86], as well as the well-known VGG [134] and ResNet [58], share essentially the same design: each contains some convolutional layers to learn low-level image features and several subsequent FC layers to summarize global information from the learned feature maps.

Each convolutional layer contains multiple convolutional kernels. Each convolutional kernel is expected to learn a specific feature representation from the input features produced by the previous layer. For the first convolutional layer, the input is the original image. By stacking multiple convolutional layers, the network is expected to learn both low-level and high-level feature maps. FC layer directly maps the feature maps to a global feature vector and is adopted at the deeper layers of a network. Each neuron in the FC layer is densely connected to every neuron in the previous layer, integrating features and learning information globally. There are two main advantages of using the convolutional operation instead of FC layers at the shallow layers: 1) firstly, each neuron in layer l is sparsely connected to a local patch (through the convolutional kernel) on the previous layer $l-1$; and 2) the weights in each convolutional kernel are shared across the whole feature maps from layer $l-1$, which greatly reduced the number of parameters and computations compared to FC layers.

Both the convolutional and FC layers are linear functions and can only learn linear mappings, in order to improve the capacity of a neural network, it is crucial to include non-linear activation layers in a convolutional neural network. The original LeNet-5 architecture adopts

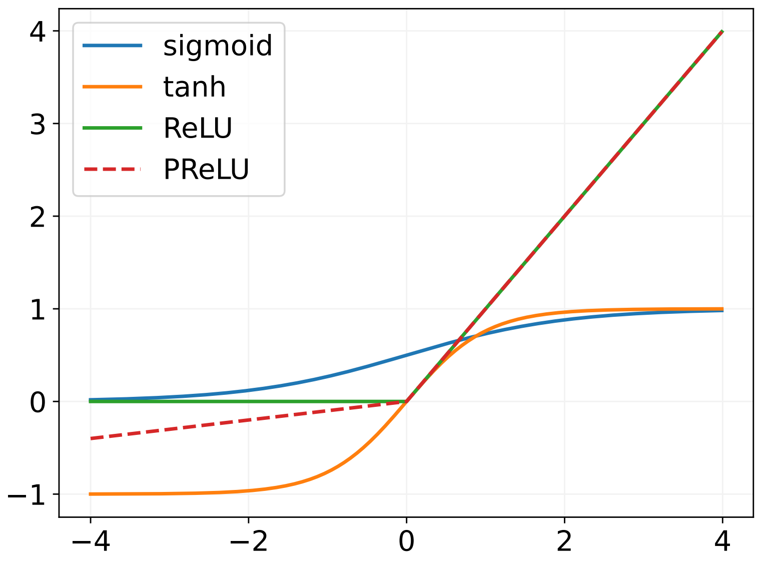


Figure 2.5: Graphs of the sigmoid, tanh, ReLU, and PReLU functions, where the slope α of PReLU is manually set to be 0.1.

the sigmoid squashing function as the activation function to perform non-linearity on the neurons. The sigmoid squashing function normally comes in two forms, one is the logistic function while the other one is the hyperbolic tangent function, as listed in Eqs. (2.29) and (2.30). AlexNet, on the other hand, is more favorable to the Rectified Linear Unit (ReLU) function, which is then extended to Parametric Rectified Linear Unit (PReLU) by introducing an extra parameter for generalization [59], the formula of the two functions are given in Eqs. (2.31) and (2.32).

$$\text{sigmoid}(x) = \frac{1}{1 + \exp(-x)}, \tag{2.29}$$

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}, \tag{2.30}$$

$$\text{ReLU}(x) = \max(0, x), \tag{2.31}$$

$$\text{PReLU}(x) = \max(0, x) + \alpha \min(0, x), \tag{2.32}$$

where α is a parameter to be learned that activates the negative features. In addition, the graphs of these activation functions are given in Fig. 2.5.

Deep affine registration models basically follow the same design as the LeNet-5 and AlexNet, the input image pair (I_M and I_F) is first fed through a number of convolutional layers to extract the spatial information. Then several FC layers are adopted to predict the transformation parameters from the convolutional feature maps. We will describe the rigid and affine registration models in Section 4.4.

2.6.2 U-Net

By adopting the FC layers, LeNet-5 and AlexNet essentially learn a global and compact feature that summarizes the contextual information about the input image, and thus can be used for deep rigid and affine registration. In order to estimate the full-resolution dense displacement field for deformable registration, a fully convolutional network is needed.

Long et al. [95] proposed a fully convolutional network (FCN) for image segmentation, by discarding the fully connected layers and employing several up-sampling layers, FCN is able to predict the full-resolution mask for an input image. Employing the design principle of the FCN [95], many prestigious segmentation frameworks have been established, and one of the most effective frameworks is U-Net [125]. Originally proposed for medical image segmentation, U-Net has drawn a lot of attention due to its clarity and effectiveness. Compared to FCN, the feature maps from down-sampling (contracting) layers are symmetric and skip-connected (concatenated) to the up-sampling (expansive) layers in U-Net, assembling a more precise spatial output.

Deformable registration tasks, similar to image segmentation, take the full-resolution images as input and estimate the full-resolution displacements. U-Net architecture, therefore, can naturally be transferred from segmentation into registration. To adopt U-Net as the registration backbone, the first problem raised is how to deal with two different images as

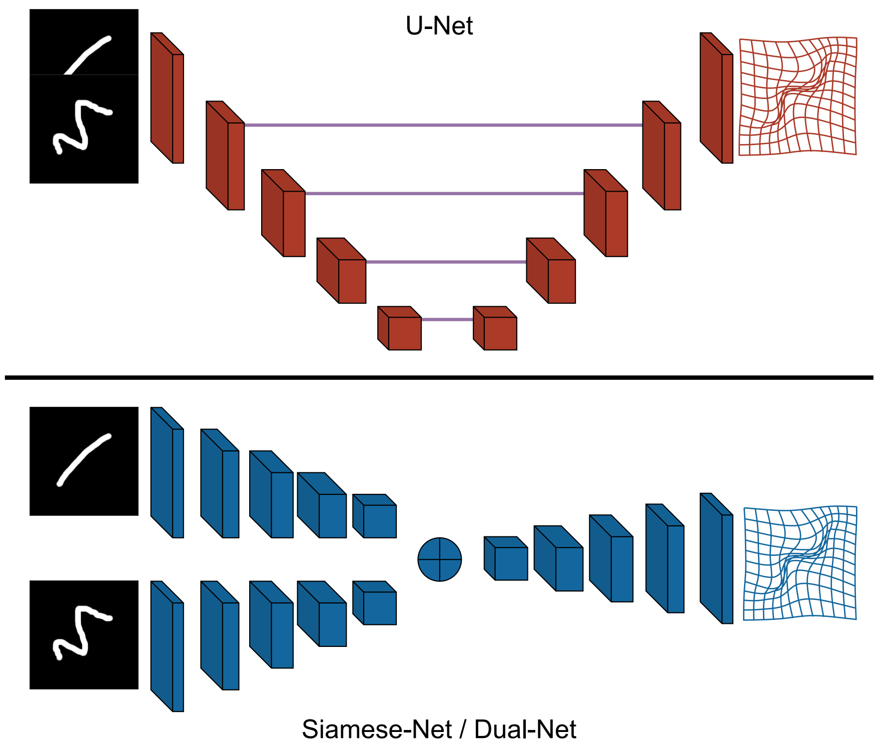


Figure 2.6: U-Net and Dual Network Architectures for Image Registration. U-Net(-like) approaches concatenate the moving and fixed images into a multi-channel image as input. Siamese Network or Dual network inputs the moving and fixed images into two separate branches, and the features are fused in later stages.

input. Firstly, it is straightforward that the moving and fixed images can be directly stacked together as a 2-channel image. This design only needs to perform the slightest modification on the first convolutional layer. On the other hand, the moving and fixed images can also be separately fed to separate encoder branches. The Siamese network [119, 121] or Dual network [69] has two identical branches and can naturally take two images as input. Features from the two branches are then fused and fed into a series of up-sampling layers to generate dense deformations. Except for the two separate branches, the rest of the Siamese network is very similar to the U-Net [125]. A schematic illustration of the U-Net and Siamese / Dual framework for image registration is given in Fig. 2.6.

2.6.3 Spatial Transformer Network

Jaderberg et al. [72] proposed the Spatial Transformer Network (STN) for image classification. STN is essentially a combination of a convolutional neural network and a learnable differentiable *spatial transformer* module, which allows explicit transforming of the input data. The spatial transformer module contains three components: a localization net, a grid generator, and a sampler.

The localization net takes the input feature map and outputs a spatial transformation (which can be rigid or affine). The grid generator and sampler together warp the input feature map according to the spatial transformation from the localization net. It is noticeable that though we have discussed the nearest and bilinear interpolation (Eq. (2.10)) in Section 2.2, they are not differentiable. We hereby introduce the differentiable nearest and bilinear interpolation proposed by [72] in Eqs. (2.33) and (2.34), respectively.

$$V_i = \sum_{n}^{H}\sum_{m}^{W} U_{nm}\delta\left(\lfloor x_i + 0.5\rfloor - m\right)\delta\left(\lfloor y_i + 0.5\rfloor - n\right), \tag{2.33}$$

$$V_i = \sum_{n}^{H}\sum_{m}^{W} U_{nm}\max\left(0, 1 - |x_i - m|\right)\max\left(0, 1 - |y_i - n|\right), \tag{2.34}$$

where $U \in \mathbb{R}^{H\times W}$ is the input feature to the sampler, H and W denote the height and width of the feature. U_{nm} is the intensity of feature U at location (n, m). V_i is the intensity of warped feature at pixel i, which corresponds to location (x_i, y_i) of U. $\lfloor x_i + 0.5\rfloor$ serves as the *round* function that returns the nearest integer of x_i. The $\delta(\cdot)$ function in Eq. (2.33) denotes the Kronecker delta function, its value is equal to 1 if and only if the input is zero; for non-zero inputs, its value is equal to 0.

Almost all deep unsupervised registration networks can be regarded as extensions of the basic spatial transformer module, it is noticeable that there are similarities as well as differences between them: 1) while the localization net in a spatial transformer takes one input image and outputs the transformation parameters, which aims to learn and augment multi-view features of the input image for the goal of supervised classification, a deep registration network takes two images as input and estimates the deformation that can establish spatial correspondence between them; 2) while the localization net is only capable of parametric transformation, a deep registration network takes two images as input and estimates the rigid and affine transformation as well as dense deformable displacement field; 3) the transformation from the localization net and the input feature are then fed into the grid generator and sampler to produce a transformed feature, this step is essentially the same in image registration, which inputs the moving image to the grid generator and sampler and outputs the warped moving image; 4) at last, while the spatial transformer have finished its job after producing a transformed feature, a registration model further inputs the warped image to a objective function and computes the loss with the fixed image, and the loss is used to update the parameters of the network with backpropagation.

2.7 Summary

This chapter walks through the preliminary and basic concepts of deep registration. In Sections 2.1 and 2.2, we introduced the rigid, affine, and deformable transformation models and the warping operation. From these two sections, one can warp an image using either a transformation matrix or a dense deformation field. In Sections 2.3 and 2.4, we introduced the general objective function, data terms, regularization terms, and optimization techniques. From these two sections, one can estimate the deformation field giving a moving and fixed image pair. In Section 2.5, we introduced the evaluation metrics of image registration, from which one can compare the performance of different registration models. At last, in Section 2.6, we introduced the architectures of some prestigious convolutional networks and their applications in deep registration, which lays the foundation for subsequent content.

Chapter Three

Deep Registration: Basics

3.1 Introduction

A registration algorithm has three key components: the registration model, objective function, and optimization function, as illustrated in Fig. 3.1. For traditional approaches like FFD [130], LDDMM [13], ANTs [7], SyN [6], Elastix [83], NiftyReg [107], DARTEL [5], and Demons [143], their registration models are carefully designed based on the prior knowledge about the images and deformations, these models have strong theoretical guarantee and clear mathematical derivations but are usually sophisticated. In the context of deep registration, the registration model is however replaced by a deep network. The deep network is expected to learn the knowledge about registration itself under a large amount of training data. By doing so, the requirement of prior knowledge in the registration model is diminished.

The basic components that construct deep unsupervised registration algorithms have been introduced in Chapter 2. This chapter, however, focuses on experimentally evaluating how different components affect the registration performance within a deep registration network. The chapter starts with a literature review on deep registration methods including both supervised and unsupervised works, then walks through the whole procedure of building a deep unsupervised U-Net registration model. At last, different combinations of the objective functions, optimization functions, and network architectures are evaluated.

3.2 Related Works

In this section, we review two types of deep registration algorithms: supervised and unsupervised. These algorithms differ in the construction and computation of the loss function, as shown in Fig. 3.2. Specifically, the deep supervised registration method involves multiple steps. Firstly, it obtains the ground-truth deformation fields for the entire training dataset.

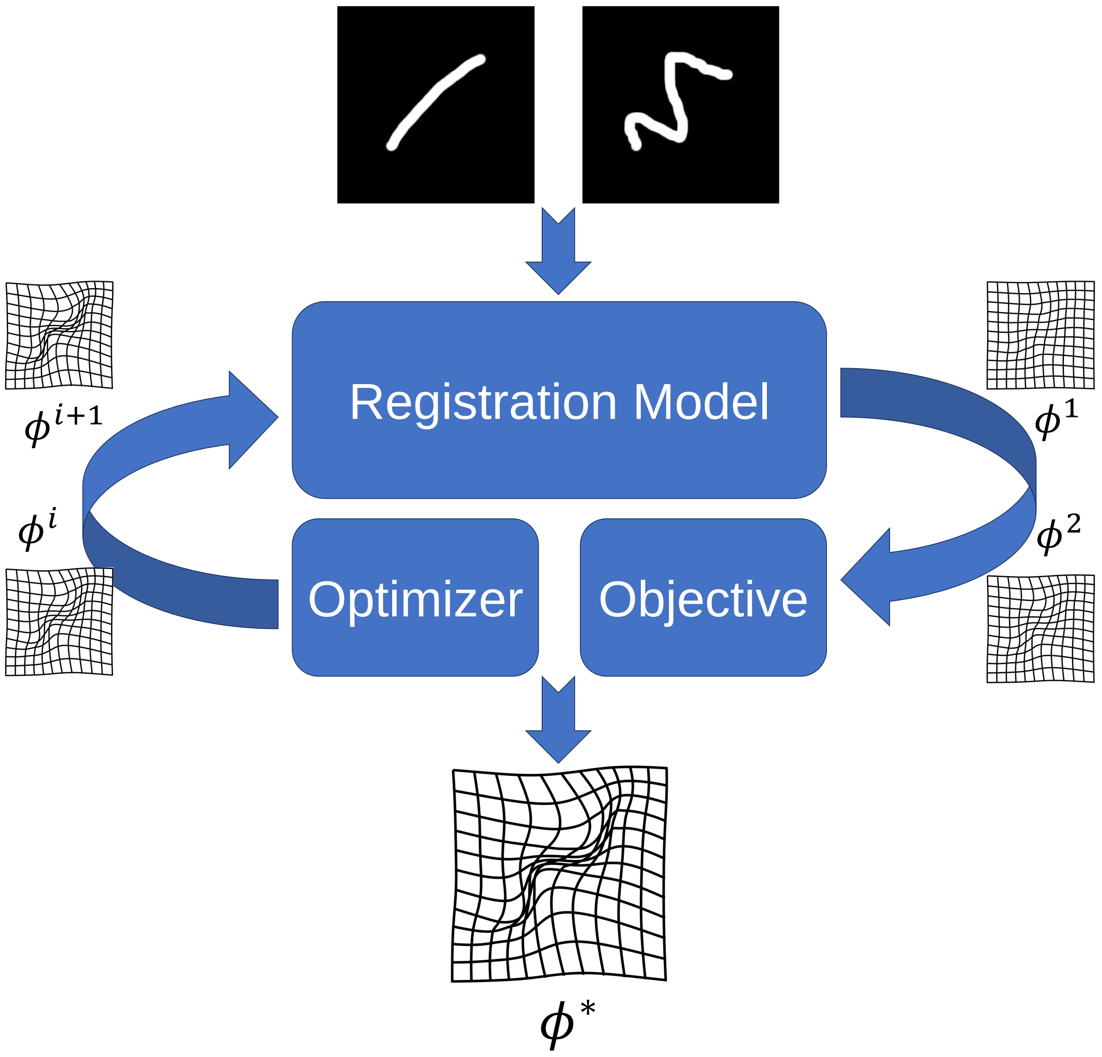


Figure 3.1: General pipeline of registration. For model-based registration algorithms, the final deformation is iteratively optimized for each input pair. For deep registration algorithms, the so-called model is replaced by a deep network, and the weights of the network are iteratively optimized on a training dataset, while only one forward propagation is required in inference.

Then, it trains the network by updating its weights to minimize the distance between the estimated and ground-truth deformation fields. The deep unsupervised registration, however, is usually end-to-end. It directly takes the moving and fixed image pair as input and

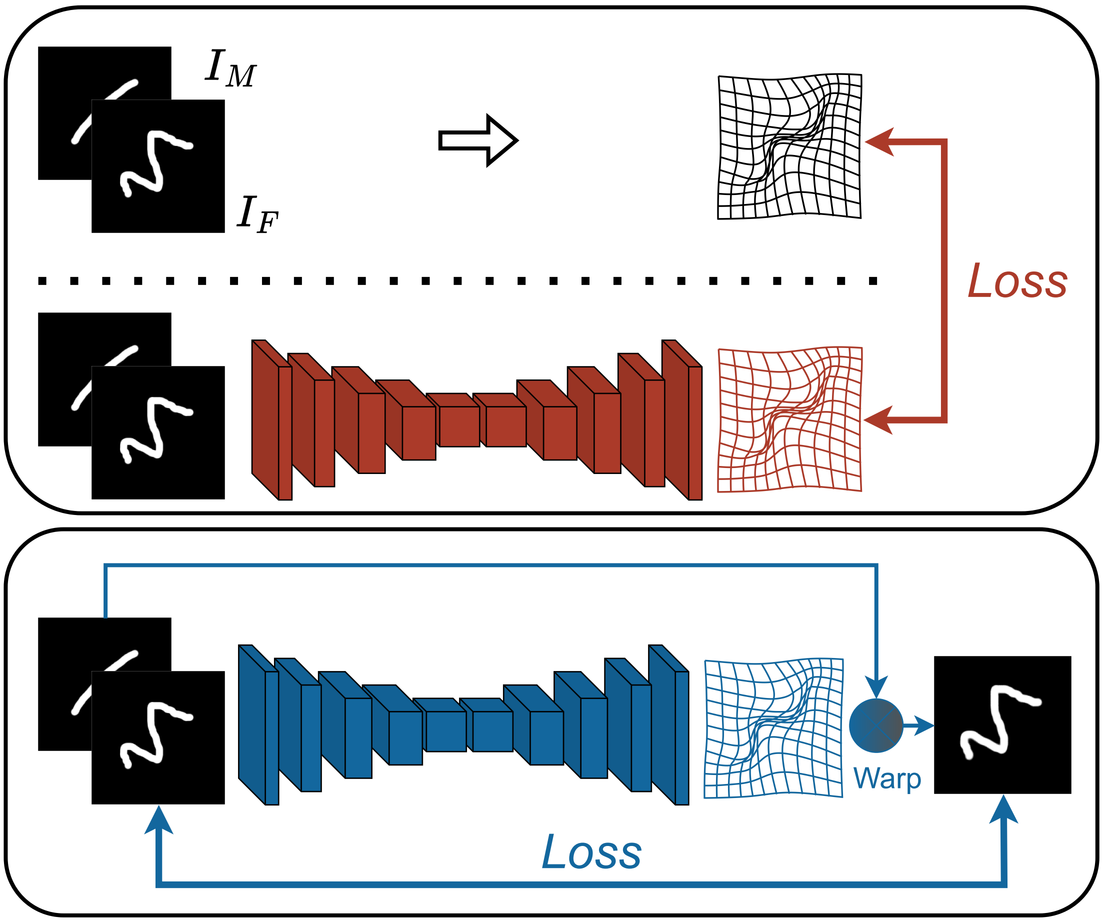


Figure 3.2: Illustrations of supervised and unsupervised registration algorithms. The loss function is directly imposed on the displacement fields in supervised methods, aiming to minimize the distance between the estimated and the ground-truth deformation (displacement) field, whereas the loss function is imposed on the warped moving image and fixed image in unsupervised approaches, implying the optimal displacement field is the one that produces the least intensity distance between warped image and fixed image. The term unsupervised denotes that no ground-truth deformation is adopted in training.

estimates the deformation field by minimizing an unsupervised loss function such as $\mathcal{L}$ in Eq. (2.11), implicitly optimizing the estimated deformation field.

3.2.1 Deep Supervised Methods

For large-scale high-dimensional medical data, it is nearly impossible to manually annotate the voxel-wise displacement fields. Therefore, in supervised approaches, the ground-truth deformation is generally obtained in two different ways: simulated displacements or predicted displacements using an existing method.

Simulated Displacements

Given a training dataset $\mathcal{I} = \bigcup_{i=1}^{N}(I_{M^{(i)}}, I_{F^{(i)}})$ that contains N moving and fixed image pairs. Deep methods based on simulated displacements do not necessarily produce the displacement between the moving and fixed image. For instance, in studies by Eppenhof and Pluim [41] and Sokooti et al. [135], a supervised method first randomly produces a number of displacements (denoted as K) for a moving image and then generates K warped moving images with these displacements, by repeating this process for all the moving images, it ends up with $N \times K$ pairs of moving and warped image as well as $N \times K$ ground-truth displacements. During training, the deep network takes a moving and warped image pair as input and estimates a displacement. The network's weights are updated by minimizing the distance between the estimated displacement and the ground truth.

Generating simulated displacements for rigid and affine transformation is straightforward, as full-resolution displacements essentially rely on the limited parameters of the transformation matrices. Therefore, adjusting different parameters within these matrices will yield various displacement fields.

Generating dense displacements, however, might need some tweaking. For example, the pipeline proposed by Sokooti et al. [135] contains three basic steps: 1) initialize a displacement field with zeros, 2) randomly assign $\mathcal{P}$ displacement vectors in the range of $[-m, m]$ for $\mathcal{P}$ points, and 3) smooth the displacement field with a Gaussian kernel with a standard deviation of σ. With these randomly generated artificial displacements, Sokooti et al. [135] first obtained the warped moving images and then regressed the ground-truth displacements from the moving and warped image pairs. Trained on 10 pairs and evaluated on 7 pairs of 3D chest CT images, their method [135] is proven can achieve comparable results with traditional B-Spline registration, suggesting that simulated displacements can be successfully applied to non-rigid registration with real-world data. However, it is notable that they extracted 2100 patches from each image of the 10 training pairs and generated multiple displacements for each patch, resulting in a total of 252,000 training examples.

Though the deep supervised methods based on simulated displacements can achieve comparable results with traditional iterative methods, one of the non-negligible drawbacks is that the randomly simulated displacements might be unrealistic. Therefore, it is necessary to generate a large number of displacements for each moving image to simulate the displacements that might occur in real-world scenarios. However, a larger K will increase the computational burdens, thereby limiting its applicability in large-scale registration tasks.

Predicted Displacements

Alternatively, there are other supervised deep approaches [18, 121, 150, 158] that use the predicted displacements from an existing method such as the traditional FFD [130], SyN [7], Demons [143], and LDDMM [13] to supervise the network training. The classical iterative method has strong theoretical guarantees while can be less competitive in terms of computational costs and inference speed. While the deep supervised approach may yield results inferior to those of reference classical models, it possesses the potential to substantially diminish the intensive computational costs associated with iterative optimization approaches and accelerate the registration process. Deep registration based on predicted displacements, therefore, has been an active research topic in deep registration.

Suppose $\boldsymbol{f}_{\mathcal{R}}$ is the referenced classical registration model, and $\mathcal{I} = \bigcup_{i=1}^{N}(\boldsymbol{I}_{M^{(i)}}, \boldsymbol{I}_{F^{(i)}})$ is the training dataset. In order to train the deep supervised model $\boldsymbol{f}$, the reference model $\boldsymbol{f}_{\mathcal{R}}$ is firstly used to predict the displacement for every training pair in the training set $\mathcal{I}$, constructing the ground-truth displacement set $\mathcal{D}_{\mathcal{R}} = \bigcup_{i=1}^{N} \boldsymbol{f}_{\mathcal{R}}(I_{M^{(i)}}, I_{F^{(i)}})$. $\boldsymbol{f}$ can then be trained with the $\mathcal{I}$ and $\mathcal{D}_{\mathcal{R}}$. Alternatively, since $\mathcal{D}_{\mathcal{R}}$ are not true displacements of $\mathcal{I}$, therefore, another strategy of training $\boldsymbol{f}$ is to first warp the moving images with $\mathcal{D}_{\mathcal{R}}$ and then regress $\mathcal{D}_{\mathcal{R}}$ with the warped images. Qiu et al. [121] compared these two strategies (either using original $\mathcal{I}$ or using the newly warped image pairs) on cardiac motion estimation. Their experimental results suggested that models trained with these two strategies achieved similar performance, both of which were inferior to the performance achieved by the reference model $\boldsymbol{f}_{\mathcal{R}}$, which was the B-Spline FFD [127]. Despite the inferior performance of the trained supervised models, they offer the advantage of faster inference speed compared to the reference model.

By first converting the relaxation formula of LDDMM [13] into a shooting formula, one can parameterize the deformation with the initial momentum field. Quicksilver [158] estimates the initial momentum of LDDMM in a supervised network, the overall runtime is, therefore, an order of magnitude faster than direct LDDMM optimization. Specifically, Quicksilver trains a deep network with the supervision of the ground-truth momentum computed by the numerical LDDMM optimization approach. Additionally, due to the large number of computations involved, Quicksilver adopted a sliding-window strategy to predict the patch-based momentums. Notably, the performance of Quicksilver is also inferior to the numerical LDDMM optimization approach.

Wang et al. [150] proposed DeepFlash, a deep supervised method aimed at accelerating the registration process of the traditional algorithm Flash [163] by regressing its velocity field. Flash is abbreviated for Fourier-approximated Lie algebras for shooting, as the reference model, Flash first estimates the velocity fields in the band-limited domain by numerical optimization for the training image pairs. Then the original moving and fixed image pairs are used to train the DeepFlash by regressing the ground-truth velocity fields in the band-limited domain. Once DeepFlash is trained, it can be used to predict the band-limited velocity field for each testing image pair but the final deformation field is determined by integrating the band-limited velocity field in an Euler-Poincare differential equation (EPDiff). In their experiments, DeepFlash achieves significantly faster inference speed compared to Flash, albeit with inferior performance.

Compared to the simulated displacement-based approaches, though predicting displacements from existing methods seems more logical and elegant, it is notable that the registration accuracy of the trained network may be limited by the underlying models used to generate the training displacements.

3.2.2 Deep Unsupervised Approaches

The supervised method is trained by minimizing the distance between the estimated and ground-truth deformation field, it needs to first construct a large amount of training data, either by simulating numerous artificial displacements or by producing the displacements from a well-established model. To overcome the shortcomings in the supervised approach, the end-to-end unsupervised approach is proposed. Instead of the complex and inefficient construction of ground-truth displacements, the unsupervised approach is more flexible, as it estimates the optimal displacements by minimizing the distance between the warped and fixed image pairs. When compared to the supervised approaches, unsupervised approaches are more efficient and can achieve competitive accuracy.

B-Spline Based Registration

DIRNet [149] is one of the earliest works for deep unsupervised registration, which consists of a regressor, spatial transformer, and a resampler. While DIRNet estimates a grid of control points in the regressor, the rest two modules are similar to the grid generator and sampler in STN [72], the spatial transformer of DIRNet generates the full-resolution displacements by cubic B-Spline function and the sampler warps moving image with linear interpolation. DIRNet has subsequently been extended in [148], where the authors proposed a multi-stage, coarse-to-fine network (termed DLIR) for cardiac MRI and chest CT image registration. There are two CNNs in this framework, the first one is a dual-path CNN which estimates 12 parameters for 3D affine transformation, and the second one takes concatenated pre-aligned images from the affine CNN as input and generates a set of B-spline points. By stacking multiple CNNs to account for global and local deformation, DLIR was able to perform more accurate registration. The model was trained with the mini-batch SGD to minimize the NCC loss with a bending energy penalty.

Qiu et al. [122] proposed a diffeomorphic B-splines model for modality-invariant registration using mutual information, termed MIDIR. MIDIR employed a U-Net style network to output the B-spline parameterized velocities, these velocities in the form of regular spacing control points can then be interpolated into the dense diffeomorphic fields using scaling and squaring [5]. The smoothness based on first-order gradients of the velocity is adopted as the regularizer and the model is optimized using Adam.

U-Net (Dual-Net and Encoder-Decoder) Style Registration

Different from the B-Spline registration which only estimates a set of control points with the convolutional network, there are earlier attempts that estimate the dense displacement field from the network for 3D deformable registration. For instance, Li and Fan [90] proposed an unsupervised FCN to estimate the displacement field. The FCN model is optimized with Adam using NCC as the data term and total variation as the regularizer. Using U-Net [125] architecture as the backbone, Balakrishnan et al. [10, 11] proposed the VoxelMorph framework for unsupervised deformable image registration, where the authors experimented with the MSE and LNCC losses to constrain image similarity and the diffusion regularizer to penalize the spatial variations in deformation. Additionally, a Dice loss based on anatomical segmentation masks is employed during training to improve registration accuracy in [10].

Qin et al. [119] proposed a framework for joint learning of motion and segmentation for cardiac sequences. They adopted a Siamese-style recurrent multi-scale spatial transformer network as the registration network and optimized the network by 1) minimizing the MSE between the warped frames and the target frame and 2) penalizing both the spatial and temporal smoothness of the deformation. Hu et al. [69] proposed a dual-stream pyramid network (Dual-PRNet) for image registration. While VoxelMorph is a single-stream U-Net style network that directly estimates the full-resolution displacement at the last convolutional layer, the Dual-PRNet progressively regresses the displacements in a coarse-to-fine manner, and the final displacement is integrated (through up-sampling and warping) by the intermediate displacements. This Dual-PRNet is subsequently extended to Dual-PRNet^{++} [79] by adding a correlation layer to further aggregate the computed pyramid features and a residual block to enhance the network capacity. The loss function adopts the NCC for similarity and first-order diffusion smoothness for regularization.

Zhao et al. [166] proposed an end-to-end deep recursive cascade architecture to successively warp moving images, termed RC-Net, enabling large displacements estimation. Each cascade in RC-Net is a U-Net-like CNN and estimates the deformation for the current input image pair, which contains the warped moving image from the last cascade and the fixed image. The final displacement is composed of all the small displacements predicted by each cascade. By cascading multiple CNNs, e.g. 10 cascades in the paper, RC-Net achieved significant gains over VoxelMorph [11] on both liver and brain registration tasks.

Although these algorithms [10, 11, 148, 166] are capable of fast and unsupervised registration, they do not guarantee or even encourage inverse-consistency and topology-preserving for registration. To address these shortcomings, Zhang [161] proposed an inverse-consistent deep network (IC-Net) for unsupervised registration. By imposing inverse consistency on the forward and backward displacements, IC-Net encouraged that a pair of images are symmetrically deformed toward each other. To further reduce foldings in the displacements, IC-Net additionally developed an anti-folding constraint and adopted a local smoothness term. In [31] and its subsequent work [32], Dalca et al. proposed a probabilistic diffeomorphic registration framework to encourage the diffeomorphism of the learned displacement. First, they proposed to learn the stationary velocity field distributions by a 3D variational U-Net. Secondly, the stationary velocity field was passed through the diffeomorphic integration module (i.e., multiple scaling and squaring layers [5]) and transferred into a diffeomorphic deformation field. Built on IC-Net and probabilistic diffeomorphic registration [31, 161], a number of works are proposed for more accurate diffeomorphic registration, including SYM-Net [110], LapIRN [111], CycleMorph [81], just to name a few.

Very recently, vision transformer [36, 94] based architectures have drawn a lot of attention in the registration community due to their capacity to model long-range information [21, 22]. However, we note that they are essentially replacing the basic convolutional block with more advanced transformer backbones, the overall pipeline still stems from the U-Net style or encoder-decoder style convolutional network.

Generative Model Based Registration

Generative models such as generative adversarial network (GAN) [50, 106] and denoising diffusion network [66] are also used in medical image registration. Fan et al. [42] proposed an adversarial similarity network and evaluated this network on four brain MRI datasets. The registration performance is comparable in both accuracy and efficiency compared with the state-of-the-art methods. Their network contains a U-Net style generator that predicts the deformations, a deformable transformation layer that warps the input with the predicted deformations, and a discriminator that judges whether images are well aligned. Besides the adversarial loss, they only employed a smooth regularization on the deformations, which suggests that adversarial loss is sufficient for measuring similarity.

Yan et al. [157] proposed an adversarial image registration network (AIR-Net) for MRI and transrectal ultrasound (TRUS) image fusion. AIR-Net contains a generator that takes MRI and TRUS images and outputs an estimated transformation, an image resampler that warps the moving image with the estimated transformation or ground truth transformation, and a discriminator that judges whether the fixed and warped moving image pair is from the estimated transformation or ground truth transformation. Differently from Fan et al. [42], the AIR-Net used ground truth transformation annotated by clinical experts.

Li et al. [91] proposed an adversarial learning framework for deformable image registration (AL-DIR) and applied it to 3D liver ultrasound image fusion. Different from other GAN-based approaches [42, 157], the proposed network 1) employed the vessel masks as extra input information, and 2) used an autoencoder for measuring the anatomical shape difference before and after registration.

The aforementioned three GAN-based approaches all employed the one-pass structure, however, Mahapatra et al. [98] proposed an adversarial registration framework based on the CycleGAN [168]. The cyclic structure allows the forward transformation from the fixed image to the moving image and the backward transformation from the moving image to the fixed image. Besides the similarity measures (such as perceptual [78] and SSIM loss) constrained on the image intensity level, they additionally employed the inverse consistency loss [26] to ensure the reversibility of the deformation field.

Very recently, Kim et al. [80] proposed DiffuseMorph, which first employed the denoising diffusion model conditioned on the moving and fixed pairs to learn the latent features that contain the pair-wise spatial correspondence. Then a deformation network was proposed to convert the learned latent feature to a deformation field and perform image registration. Therefore, once the denoising diffusion model and deformation network are jointly trained in an end-to-end manner, DiffuseMorph can not only provide continuous deformation fields through the deformation network but also produce continuous image generation through the inverse diffusion process. However, the DiffuseMorph was limited to 2D images.

3.3 Methodology

Deep dense unsupervised registration model aims to learn a network-based function $\boldsymbol{f(\Theta)}$, parameterized by $\boldsymbol{\Theta}$, to predict the dense displacement. Once the network is well-trained, the weights (parameters) $\boldsymbol{\Theta}$ are fixed, and the network is believed to have learned the knowledge of image registration. Then the trained network is able to perform fast registration for a new given test pair with one single forward pass, achieving faster inference than the costly instance-level optimization in traditional iterative algorithms. Although the learning process of the network's weights $\boldsymbol{\Theta}$ is fully automated, the design of network architecture, objective function, and the selection of optimizer still require manual interaction.

3.3.1 Model Architecture

U-Net style networks are the common choices for image registration networks to learn the dense deformations. We use a U-Net style network [161] in this chapter, as shown in Fig. 3.3, where there are five convolutional blocks in the contracting path and five convolutional blocks in the expansive path. Specifically, taking the 3D moving and fixed volumes with resolution (D, H, W) as input, the resolution of feature maps flows as $(C, D, H, W) \rightarrow (2C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2}) \rightarrow (4C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4}) \rightarrow (8C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8}) \rightarrow (16C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$ in the contracting path. Reversely, the resolutions in the expansive path will be progressively upsampled to $(3, D, H, W)$ as the final estimation of the displacements, where $\boldsymbol{C}$ indicates the number of channels (kernels) in the first convolutional block, and is set to be 8 in default.

Each contracting convolutional block contains two convolutional layers. The first layer encodes the input feature maps, while the second layer downsamples the resolution of feature maps using a stride of 2 and doubles the number of channels. Conversely, each expansive convolutional block contains three convolutional layers. The first layer upsamples the input feature maps using a fractional convolution with a stride of 2, while the subsequent two layers halve the number of channels. Skip connections are established between blocks containing feature maps of the same resolution. All convolutional layers use $3 \times 3 \times 3$ kernels and are followed by a ReLU activation function, except for the last layer, which is followed by a Softsign function $y = \frac{x}{1+|x|}$ to output the displacement within the range [-1,1]. By simply changing all convolutional kernels from 3D to 2D and using 2 output channels instead of 3 at the last layer, this architecture is applicable for 2D image registration.

3.3.2 Objective Function

The U-Net estimates a dense displacement field for the input image pair. The estimated displacement can then be used to warp the moving image through a warping layer. We adopt the spatial warping layers based on linear interpolation in [10, 72]. Then, an unsupervised loss is computed from the moving image $\boldsymbol{I_M}$, the fixed image $\boldsymbol{I_F}$, and the predicted displacement field $\boldsymbol{\phi}$ or velocity field $\boldsymbol{v}$ in the case of diffeomorphic registration. The loss function $\mathcal{L}(\boldsymbol{\Theta})$ of our U-Net is

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{Sim}(I_{M_i} \circ (\boldsymbol{\phi}_i(\boldsymbol{\Theta})), I_{F_i}) + \frac{\lambda}{N}\sum_{i=1}^{N}\mathcal{L}_{Reg}(\boldsymbol{\phi}_i(\boldsymbol{\Theta})), \tag{3.1}$$

where N is the number of training pairs, $\boldsymbol{\Theta}$ is the network parameters to be learned, $\circ$ is the warping operator.

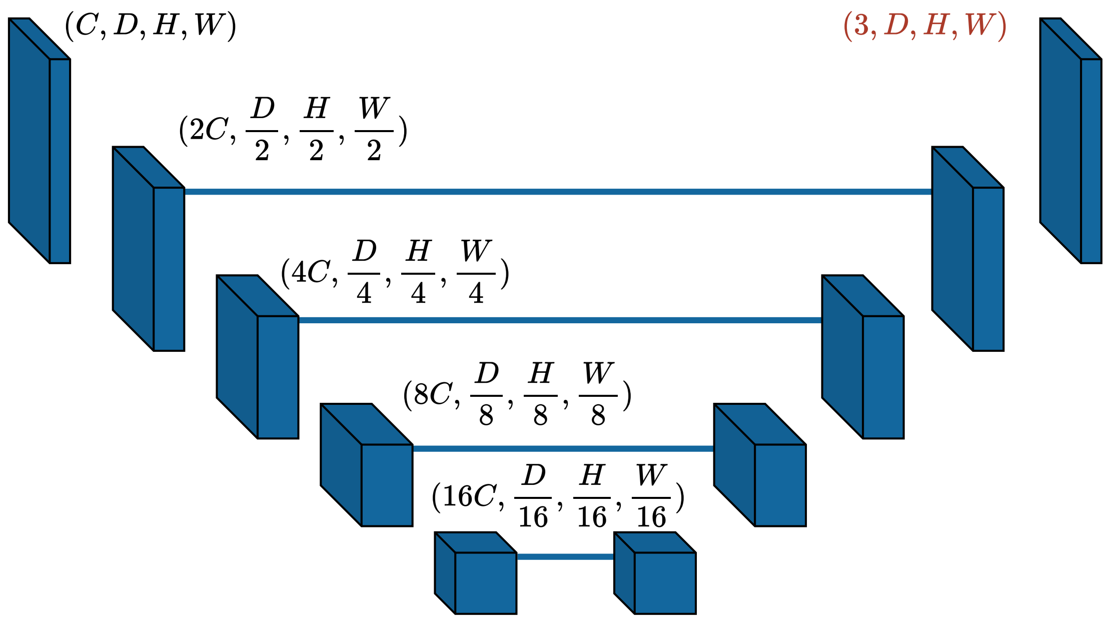

(C, D, H, W)
$(3, D, H, W)$
$(2C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2})$
$(4C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(8C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$
$(16C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$

Figure 3.3: Architecture of U-Net. There are five convolutional blocks in the contracting path and five convolutional blocks in the expansive path, with blue lines indicating the skip connections. C is the number of channels in the first convolutional blocks, which is set to 8 by default.

Instead of directly estimating the displacement field, we can also estimate the stationary velocity field $\boldsymbol{v}$ and then adopt scaling and squaring layers [5, 31] to encourage diffeomorphic deformation field, i.e., $\boldsymbol{\phi} = Exp(v)$. In experiments, seven scaling and squaring layers are used. Overall, the objective function for diffeomorphic U-Net is defined as

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}_{Sim}(I_{M_i} \circ Exp(\boldsymbol{v}_i(\boldsymbol{\Theta})), I_{F_i}) + \frac{\lambda}{N}\sum_{i=1}^{N}\mathcal{L}_{Reg}(\boldsymbol{v}_i(\boldsymbol{\Theta})). \tag{3.2}$$

In both two equations, the first term $\mathcal{L}_{Sim}$ defines the similarity between warped moving images and fixed images, and the second term $\mathcal{L}_{Reg}$ defines the smoothness of displacement fields. λ is a hyper-parameter balancing the two terms. $\mathcal{L}_{Sim}$ can be either SSD, NCC, or NMI, and $\mathcal{L}_{Reg}$ can be either the diffusion regularizer or the bending energy, which we clarify in experiments.

3.3.3 Optimization Function

SGD and Adam are two of the most popular optimizers used in training deep networks. In Section 2.4, we have briefly introduced the naive SGD and Mini-batch SGD. In training large convolution neural networks, however, the SGD is often jointly used with momentum or Nesterov momentum [137] to accelerate the training process, the pseudo-code of the momentum accelerated SGD is given in Algorithm 2.

SGD used the same updating rate during the whole training process. Adam [82], on the other hand, adaptively adjusts the updating rates for different parameters by estimating two consecutive moments of the gradients, the whole updating rule of the gradient in Adam is given as

$$\begin{aligned} g_t &\leftarrow \partial f(\theta_{t-1})/\partial \theta_t, \\ m_t &\leftarrow \beta_1\ m_{t-1} + (1-\beta_1)\ g_t, \\ v_t &\leftarrow \beta_2\ v_{t-1} + (1-\beta_2)\ g_t^2, \\ \tilde{m}_t &\leftarrow m_t/(1-\beta_1^t), \\ \tilde{v}_t &\leftarrow v_t/(1-\beta_2^t), \\ \theta_t &\leftarrow \theta_{t-1} - \alpha\ \tilde{m}_t/(\sqrt{\tilde{v}_t}+\epsilon). \end{aligned} \tag{3.3}$$

where α is the learning rate, β_1 and $\beta_2 \in [0,1)$ are the decay rates in moment estimates. ϵ avoids zero-division error.

3.3.4 Datasets and Evaluation

Two different datasets, i.e., the public OASIS-1 dataset and a local 3DCMR dataset, were employed to evaluate the registration performance.

OASIS-1 dataset [102] consists of a cross-sectional collection of T1-weighted brain MRI

Algorithm 2 Momentum accelerated SGD

```
1: Inputs : training data $\mathcal{I} = \left\{\left(I_{F^{(i)}}, I_{M^{(i)}}\right)\right\}_{i=1}^{N}$, batch size $b$, the learning rate $\alpha$, the
   momentum $\mu$, the bool flag of nesterov.
2: Initialize : $\Phi_0$.
3: Shuffle the pairs in $\mathcal{I}$.
4: for $t = 1 : N_{iter}$ do                          ▷ # $N_{iter}$ is the maximum iterations.
5:     for $n = 1 : \frac{N}{b}$ do                     ▷ # $b$ denotes the batch size.
6:         $g_t = \frac{1}{b} \sum_{i=1}^{b} \frac{\partial \mathcal{L}(I_{F^{(i)}}, I_{M^{(i)}}; \Phi_{(t-1)})}{\partial \Phi_{(t-1)}}$
7:         if t==1 then
8:             $v_t = g_t$
9:         else
10:            $v_t = \mu v_{(t-1)} + g_t$              ▷ # Adding momentum.
11:        end if
12:        if nesterov==True then
13:            $g_t = g_t + \mu v_t$                    ▷ # Adding Nesterov momentum.
14:        else
15:            $g_t = v_t$
16:        end if
17:        $\Phi_t = \Phi_{(t-1)} - \alpha\ g_t$
18:    end for
19:    return $\boldsymbol{\Phi}^* = \boldsymbol{\Phi_t}$     ▷ # return the final deformation field.
20: end for
```

scans from 416 subjects. In experiments, we use the pre-processed OASIS data [67] provided by the Learn2Reg challenge [63] to perform subject-to-subject (inter-subject) brain registration. This dataset has 414 2D 160×192 slices and masks extracted from their corresponding 3D $160 \times 192 \times 224$ volumes. We randomly split this 2D dataset into 201, 12, and 201 images for training, validation, and testing. After pairing, we end up with 400 ([201-1]×2), 22 ([12-1]×2), and 400 ([201-1]×2) image pairs for training, validation, and testing, respectively. Each 2D segmentation mask contains 24 automated labels from FreeSurfer.

3DCMR dataset [37, 140] consists of 220 pairs of 3D high-resolution cardiac MRI scans, in which each scan is captured during only one single breath-hold and includes the end-diastolic (ED) to end-systolic (ES) frames of the cardiac cycle. In our experiments, we re-sampled all scans from original resolution 1.2×1.2×2.0mm^3 to 1.2×1.2×1.2mm^3 and center-cropped 128×128×96 sized volumes. We randomly split the data into 100, 20, and 100 corresponding to training, validation, and testing sets. Both ES and ED segmentation masks on three anatomical structures, i.e., the left ventricle cavity (LV), myocardium (Myo), and right ventricle cavity (RV), are provided for performance evaluation.

Evaluation matrices: The percentage of negative Jacobian determinants presented in the displacement fields, denoted as $|J|_{<0}\%$, are reported to assess whether the displacement is realistic and plausible for both two datasets. A larger percentage indicates there are more foldings in the displacement field, while a smaller percentage indicates fewer foldings. For diffeomorphic algorithms, a value of zero is desired, as diffeomorphic deformation is a continuous one-to-one mapping [26], therefore has no folding.

For the 3DCMR dataset, we evaluate the registration performance using segmentation masks from ES and ED frames. Specifically, we first estimate the deformation field from the ES to ED scan and apply this deformation to warp the segmentation mask from the ES scan. We then calculate the Dice and HDist between the warped ES mask and the ground truth mask of ED. These metrics are then averaged across the LV, Myo, and RV structures. For the 2D OASIS-1 inter-subject brain registration, similarly, we estimate the deformation field between the image pairs and evaluate the registration performance on the segmentation masks. The Dice score averaged on 24 different anatomical structures is reported.

Table 3.1: Comparisons between different data terms on OASIS-1 and 3DCMR. Dice and $|J|_{<0}\%$ are reported for both datasets. HDist (in *mm*) is additionally reported for the 3DCMR dataset. The first row indicates the initial results without any registration.

Data Term	OASIS-1		3DCMR		
	Dice↑	$\lvert J\rvert_{<0}\%$	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$
–	0.540±0.088	–	0.493±0.043	10.08±1.07	–
SSD	0.752±0.042	0.788±0.413	0.769±0.028	6.16±0.75	0.312±0.202
NCC	0.749±0.045	1.775±0.497	0.711±0.032	6.82±1.01	0.049±0.065
NMI	0.753±0.042	1.667±0.572	0.767±0.028	5.88±0.75	0.069±0.112

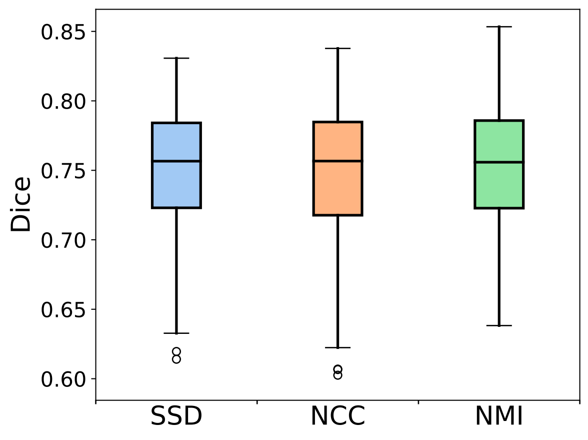
3.4 Experiments

3.4.1 Objective Function

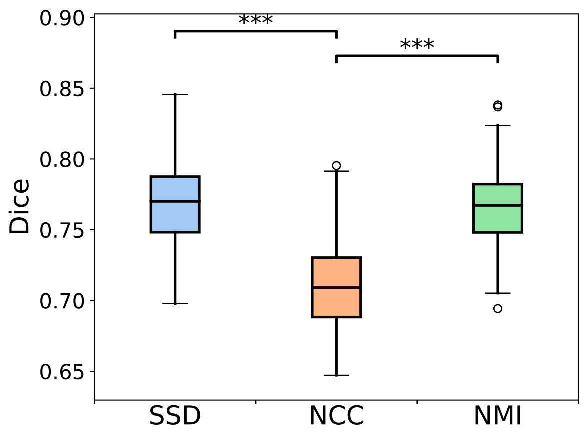
Data Term

SSD, NCC, and NMI, as aforementioned in Section 2.3, are three commonly used distance (similarity) measurements based on image intensity. We hereby experiment with the three data terms and investigate the influence of different data terms on registration performance.

For the OASIS-1 dataset, as listed in Table 3.1, we notice that the U-Net models trained with three different metrics achieve similar performance in terms of Dice score, i.e., 0.752, 0.749, and 0.753 for SSD, NCC, and NMI, respectively. The p-values between every two of them are 0.3, 0.7, and 0.2, indicating no significant differences are presented. Boxplots together with significance bars are provided in Fig. 3.4 for a more comprehensive comparison. However, we note there are clear differences on $|J|_{<0}\%$: only 0.788% pixels have negative determinants in estimated displacements from SSD, while the percentage is increased to



(a) OASIS-1



(b) 3DCMR

Figure 3.4: Boxplots of Dice scores achieved by the three data terms. No significant differences are observed on the OASIS-1 dataset. SSD and NMI achieve similar Dice scores (with a 0.5 p-value, $P > 0.05$, not significant) on 3DCMR while NCC obtains the lowest Dice score ($P \leq 0.001$, marked with three asterisks $***$).

1.667% and 1.775% for NMI and NCC, respectively. A larger $|J|_{<0}\%$ denotes more foldings are presented in the displacement field, as shown in Fig. 3.5. For this specific example, the Dice between the moving and fixed pair is increased from the initial 0.590 to 0.788, 0.789, and 0.779 when warping with deformations estimated by SSD, NCC, and NMI, while producing 0.716, 1.286, and 1.507 $|J|_{<0}\%$, respectively.

However, for the 3DCMR dataset, the model trained with SSD produces relatively much more foldings than NCC and NMI, reaching a 0.312 $|J|_{<0}\%$. Though the model trained with NCC produces less folding, its performance in terms of both Dice (0.711) and HDist (6.82 mm) is inferior to that of SSD and NMI. The model trained with NMI achieves a competitive Dice score (0.767) and the lowest HDist (5.88 mm). The visual comparison of registration using the three data terms is shown in Fig. 3.6, for this specific image, the deformation produced by NMI has 0.037 $|J|_{<0}\%$, while the other two losses obtain zero foldings.

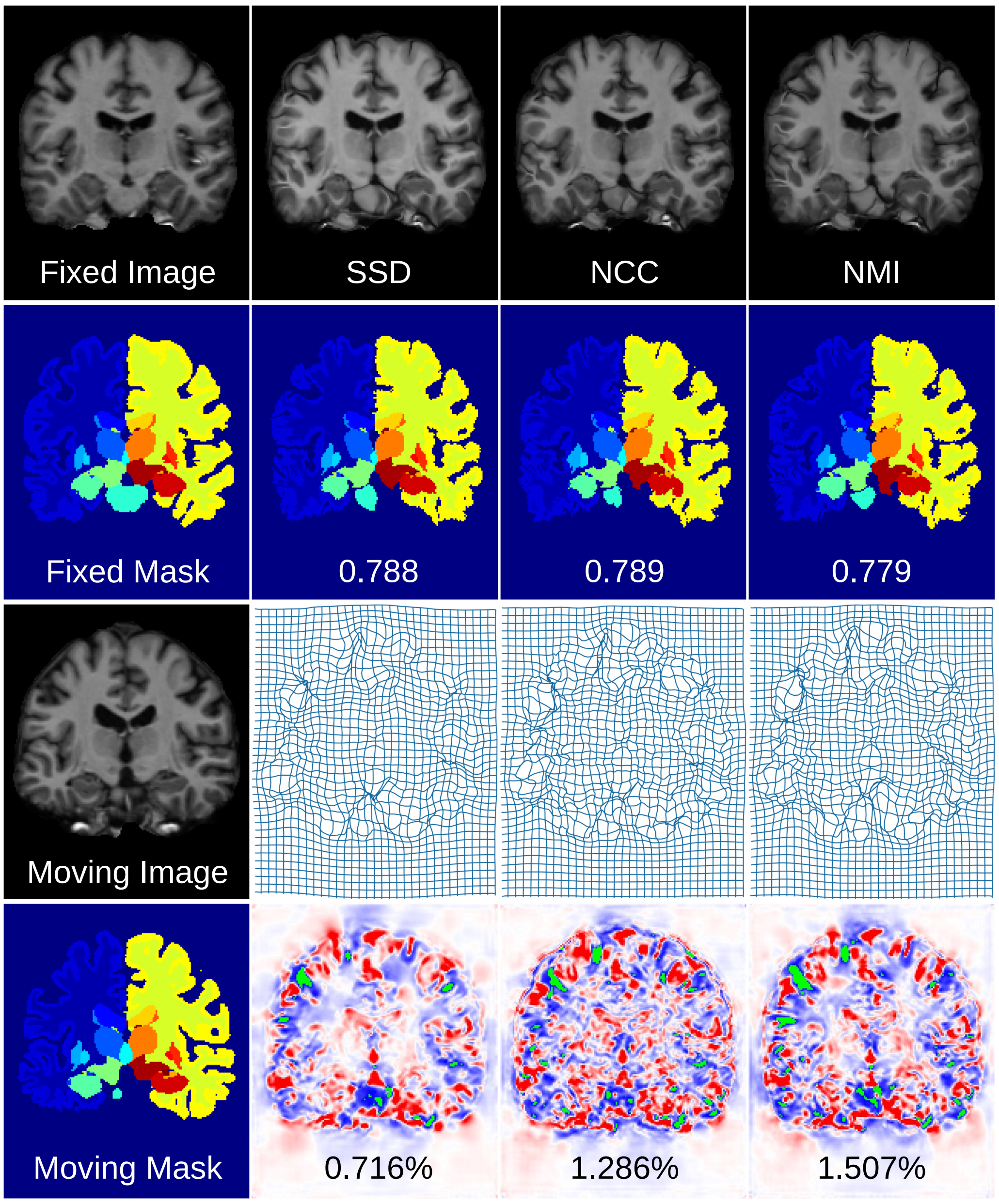


Figure 3.5: Visual comparison between different data terms on OASIS-1. Top 2 rows: fixed image and mask, warped images and masks (with Dice) from SSD, NCC, and NMI. Bottom 2 rows: moving image and mask, grids, and Jacobian determinant maps (with $|J|_{<0}\%$) from SSD, NCC, and NMI. The redder color in the Jacobian determinant map denotes a larger Jacobian determinant while the bluer color denotes a smaller Jacobian determinant. Negative Jacobians are masked to green for better illustration.

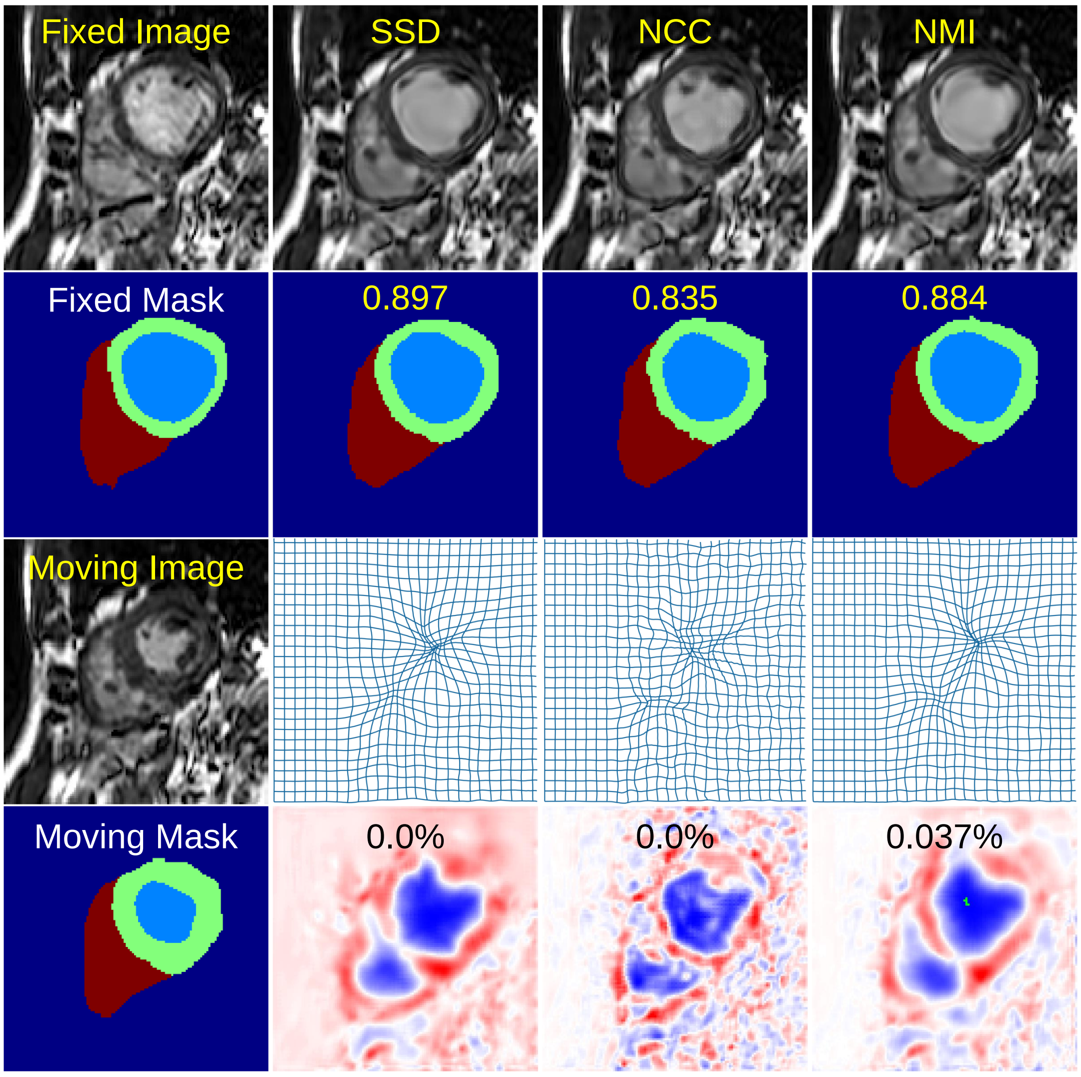


Figure 3.6: Visual comparison between different data terms on 3DCMR. Top 2 rows: fixed image and mask, warped images and masks from SSD, NCC, and NMI. Bottom 2 rows: moving image and mask, grids, and Jacobian determinant maps from SSD, NCC, and NMI. In this specific example, only the Jacobian map produced by NMI has foldings (0.037%, marked with green color). SSD, NCC, and NMI achieved 0.897, 0.835, and 0.884 Dice scores on this specific example, respectively.

Table 3.2: Comparisons between different regularization terms on OASIS-1 and 3DCMR.

Regularization	OASIS-1		3DCMR		
	Dice↑	$\lvert J\rvert_{<0}\%$	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$
–	0.540±0.088	–	0.493±0.043	10.08±1.07	–
Diffusion	0.752±0.042	0.788±0.413	0.728±0.031	6.49±0.98	0.340±0.143
Bending Energy	0.734±0.049	2.627±0.833	0.767±0.028	5.88±0.75	0.069±0.112

From these results in Table 3.1, we highlight that different data terms may yield different results, and different registration tasks may favor different data measurements, therefore the selection of suitable data measurement is important. We note that we use SSD as the data term for 2D OASIS-1 and NMI as the data term for 3DCMR in the rest of the chapter.

Regularization Term

We experiment with two commonly used regularization terms, i.e., first-order diffusion regularizer and second-order bending energy. For the OASIS-1 dataset, as listed in Table 3.2, the model trained with diffusion regularizer outperforms the model trained with bending energy in terms of Dice, achieving a 1.8% improvement from 0.734 to 0.752. In the meantime, the model trained with diffusion regularizer produces significantly less folding on the displacements than the model trained with bending energy, obtaining smaller $|J|_{<0}\%$: 0.788% versus 2.627%.

Differently, for the 3DCMR dataset, the model trained with bending energy achieves a much higher Dice than the model trained with diffusion regularizer (0.767 vs 0.728). The superiority of using bending energy can also be observed from the lower HDist, where the model trained with bending energy can achieve a 5.88mm HDist, which is lower than that of the diffusion regularizer (6.49mm). Additionally, the $|J|_{<0}\%$ produced by bending energy is also significantly lower than that of displacements produced by diffusion regularizer.

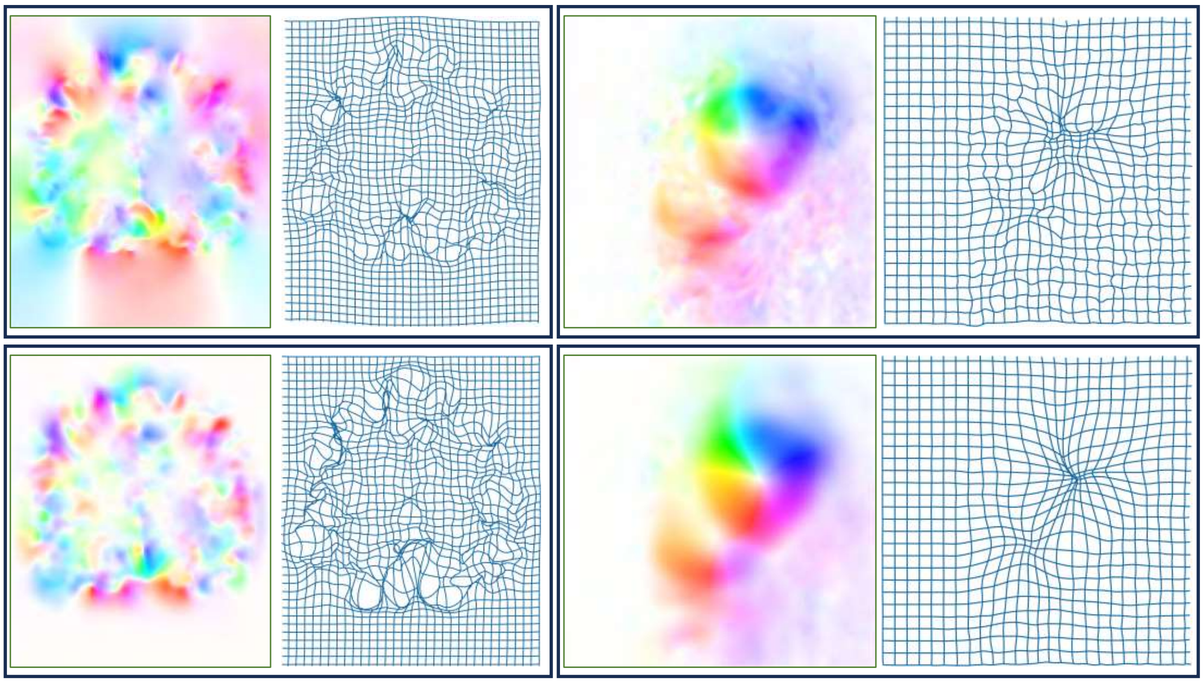


Figure 3.7: Visual comparison between different regularization terms. From left to right: estimated displacement field (in the HSV color mode) and deformation grid on OASIS-1 and 3DCMR. From top to bottom: diffusion regularizer and bending energy.

We reckon the reason why these two datasets favor different regularization terms is that the anatomical structures from different subjects might have large variations and sharp displacements are therefore needed for inter-subject brain registration. Moreover, since 2D slices are adopted from the 3D brain scan, this might further enlarge the variations. Cardiac registration is intra-subject and requires smoother displacements, which may be more favorable for bending energy as it is designed to penalize sharply curved deformations [21]. This assumption can also be observed from the illustration in Fig. 3.7, where bending energy indeed produces smoother displacements as can be seen from the clearer background. To register the sophisticated structures between different brain slices, the model trained by bending energy however produces more foldings in order to map large displacements.

Hyper-parameter Tuning

Table 3.3: Comparisons between different λ on OASIS-1.

Diffusion			Bending Energy		
λ	Dice↑	$\lvert J\rvert_{<0}\%$	λ	Dice↑	$\lvert J\rvert_{<0}\%$
–	0.540±0.088	–	–	0.540±0.088	–
0	0.712±0.052	12.655±1.341	0.001	0.725±0.053	5.944±1.223
0.001	0.742±0.049	3.586±0.883	0.005	0.729±0.050	4.120±1.023
0.005	0.750±0.044	1.552±0.604	0.01	0.734±0.048	3.058±0.898
0.01	0.752±0.042	0.788±0.413	0.02	0.734±0.049	2.627±0.833
0.02	0.738±0.043	0.283±0.237	0.05	0.732±0.047	1.704±0.716
0.1	0.693±0.052	0.005±0.028	0.1	0.728±0.045	0.988±0.541

One of the key hyper-parameters in Eq. (3.1) is the λ, which balances the regularization term and data similarity term. It is assumed and predictable that a very large λ will lead to an extremely smooth displacement but less accurate registration, a very small λ helps the optimization of the data term, but it may produce unrealistic displacement. To assess our assumption, we use a straightforward grid-search strategy to select the optimal λ and provide detailed comparisons of registration performance achieved with different λ.

In Table 3.3, we compare the performance of six different λ for both diffusion and bending energy on the 2D OASIS-1 dataset. Firstly, when the λ is equal to zero, meaning no regularization is applied, the objective function is optimizing the data term alone, and its performance (0.712) still significantly outperforms the initial 0.540 in terms of Dice. In the case of using diffusion regularization, the registration performance first gradually increases with a larger λ, and the highest Dice is achieved with $\lambda = 0.01$. However, the performance rapidly decreased when continuously increasing the λ. When the λ is increased to 0.1, the achieved Dice (0.693) is even lower than the 0.712 achieved by zero λ. On the other hand, we note the percentage of negative Jacobian determinants ($\lvert J\rvert_{<0}\%$) consistently decreases as the

λ increases. In the case of using bending energy as the regularization, similar trends: 1) the Dice is first increased then decreased with continuous increasing λ and 2) the $|J|_{<0}\%$ consistently decreases as the λ increases. However, the model trained with bending energy exhibits significantly more foldings compared to the one trained with diffusion on this dataset, due to the sophisticated anatomical structures of the brain, as we have discussed before.

In Fig. 3.8, we have presented the deformation field, Jacobian determinants map, warped image, and difference map between warped and fixed image. It is clear that the warped image is closer to the fixed image with a smaller λ (indicated by the MAE values), while zero λ produces the most unrealistic deformation (11.191 $|J|_{<0}\%$). From this figure, we highlight that **a smaller data distance does not necessarily mean a better registration**.

In Table 3.4 and Fig. 3.9, we have provided the quantitative and qualitative results with different λ on the 3DCMR dataset, similar trends with Table 3.3 can also be observed. From these two Tables, we highlight that different λ configurations largely affect the final performance, thus, the selection of an appropriate λ holds significance. For simplicity and the purpose of proving the concepts, we use a very basic grid-search strategy to select the optimal λ, one may refer to [67, 109] for automatic tuning to ease the tedious tuning process of λ when training deep registration models.

Diffeomorphism

Table 3.5 lists the quantitative comparisons between non-diffeomorphic and diffeomorphic registration on the 2D OASIS-1 and 3DCMR datasets. On both two datasets, the values of $|J|_{<0}\%$ are largely reduced with diffeomorphic registration, achieving (or at least approaching) zeros. As illustrated in Fig. 3.10, for this specific example brain image, the deformation field estimated by diffeomorphism constraint has zero foldings.

On the other hand, the registration performance on cardiac images is also largely improved

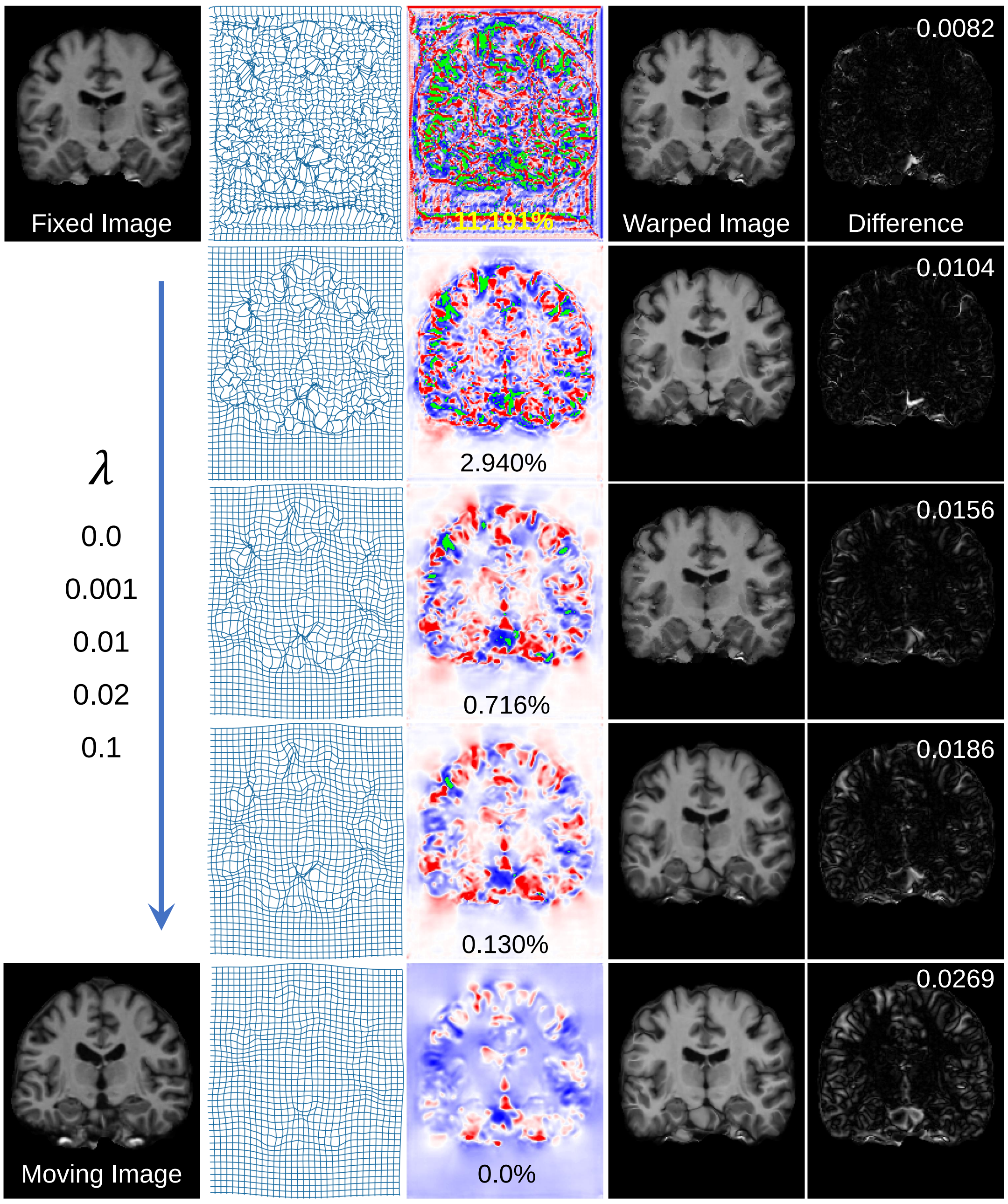


Figure 3.8: Impacts of using different λ on OASIS-1. From left to right (excluding the first column): deformation, Jacobian map (with $|J|_{<0}\%$), warped image, and difference map between warped and fixed image (with MAE on the upper right). From top to bottom, increasing the value of λ from 0.0 to 0.1. A smaller λ produces more similar warped images to the fixed one (clearly indicated by MAE), while unrealistic deformation (indicated by $|J|_{<0}\%$). A larger λ produces plausible deformation but a less similar warped image.

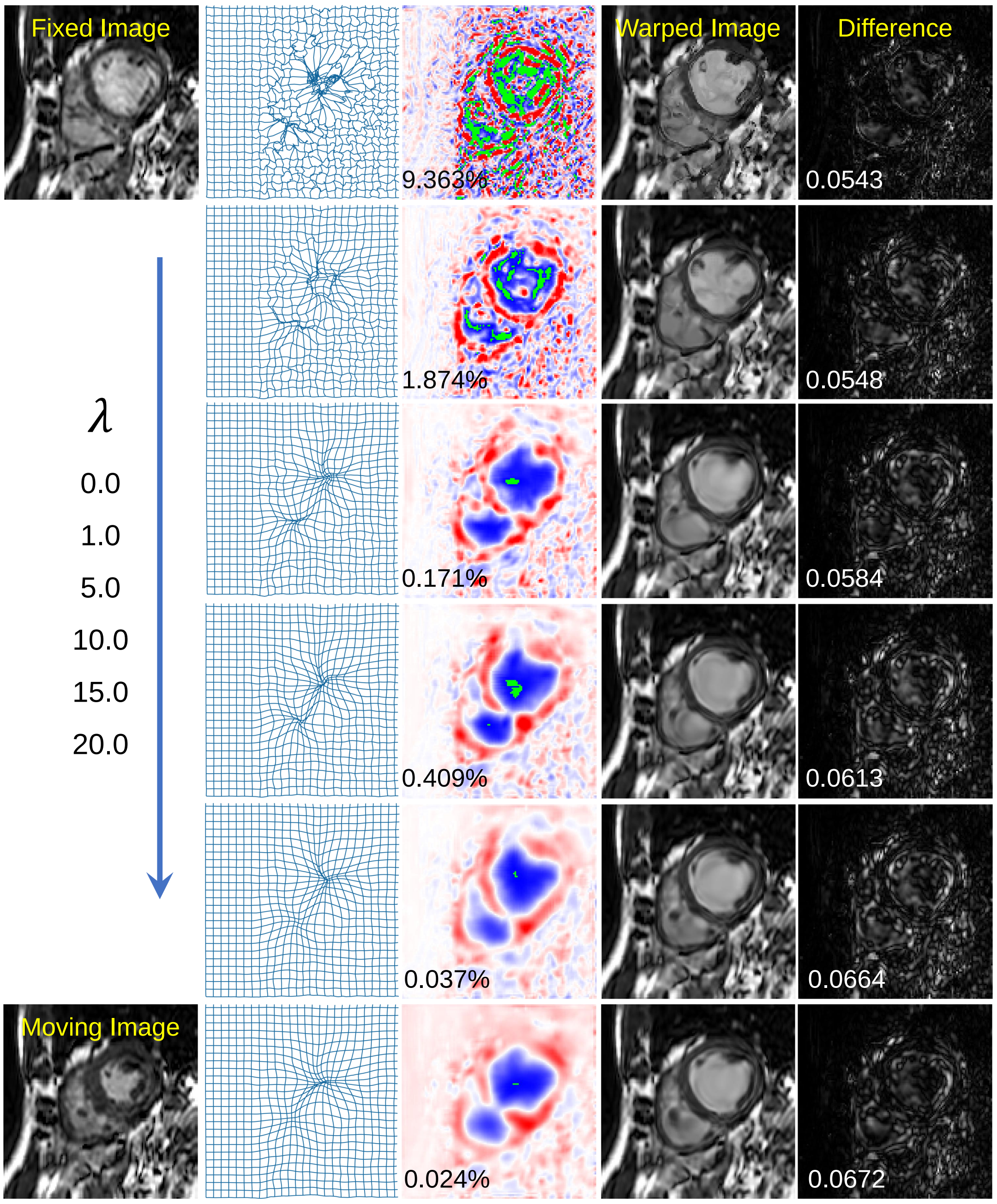


Figure 3.9: Impacts of using different λ on 3DCMR. The λ is increased from 0.0 to 20.0 from top to bottom. $|J|_{<0}\%$ is given in the lower left of the Jocabian map, while the MAE is labeled on the lower left of the difference map. Note that both the $|J|_{<0}\%$ and MAE fluctuated with λ=10.0.

Table 3.4: Comparisons between different λ on 3DCMR.

Diffusion				Bending Energy			
λ	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$	λ	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$
	0.493±0.043	10.08±1.07	–		0.493±0.043	10.08±1.07	–
0.0	0.661±0.037	9.13±0.94	7.132±1.703	1.0	0.730±0.030	6.46±0.93	1.642±0.503
1.0	0.695±0.032	7.70±1.04	2.745±0.664	5.0	0.759±0.029	6.07±0.87	0.491±0.280
5.0	0.728±0.031	6.49±0.98	0.340±0.143	10.0	0.767±0.027	5.91±0.80	0.230±0.200
10.0	0.705±0.035	6.81±1.06	0.007±0.007	15.0	0.767±0.028	5.88±0.75	0.069±0.112
15.0	0.675±0.039	7.43±1.05	<2E-6	20.0	0.767±0.028	5.90±0.78	0.053±0.081

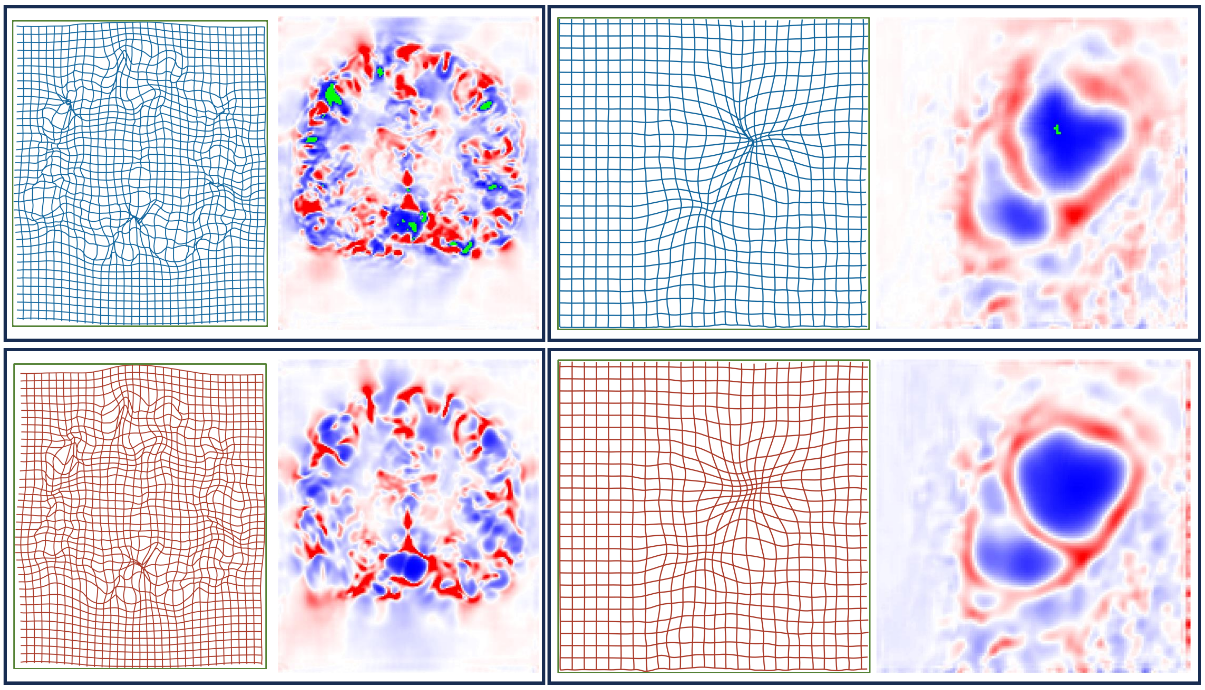


Figure 3.10: Visual comparisons between non-diffeomorphic (top row) and diffeomorphic (bottom row) registration on OASIS-1 and 3DCMR. No folding (marked with green) is presented in the bottom row.

with the diffeomorphism constraint in terms of both the Dice and HDist. The performance on the 2D OASIS-1 dataset, however, merely changed. These results are in line with our

Table 3.5: Comparisons between non-diffeomorphic and diffeomorphic registration on OASIS-1 and 3DCMR.

diffeomorphism	OASIS-1		3DCMR		
	Dice↑	$\lvert J\rvert_{<0}\%$	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$
–	0.540±0.088	–	0.493±0.043	10.08±1.07	–
✗	0.752±0.042	0.788±0.413	0.767±0.028	5.88±0.75	0.069±0.112
✓	0.751±0.040	<0.00008	0.788±0.030	5.77±0.74	0.0±0.0

expectations as cardiac motion is desired to be a smooth and one-to-one mapping, while sophisticated displacements are required for inter-subject brain registration, and thus diffeomorphism can not ensure a better Dice score.

3.4.2 Optimization

In Section 3.4.1, we have studied the effects of different configurations of the objective function. We hereby study the effects of the optimization process on the final registration performance. First and foremost, we investigate the effects of training data, will less amount of training data result in an inferior registration? Secondly, we investigate the influence of the optimization algorithm itself, will different optimization algorithms yield distinct outcomes?

Training Data

In Table 3.6, different amounts of training data are used to train the proposed U-Net model. For example, we have in total 400 and 100 training pairs for OASIS and 3DCMR, 50% denotes we randomly selected half of the training pairs to train the registration model, corresponding to 200 and 50 pairs, respectively. It is clear that the amount of training data is crucial to the training process.

Table 3.6: Comparisons between different amounts of training data on OASIS-1 and 3DCMR. The first row indicates the initial results without any registration.

Amount	OASIS-1		3DCMR		
	Dice↑	$\lvert J\rvert_{<0}\%$	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$
–	0.540±0.088	–	0.493±0.043	10.08±1.07	–
10%	0.691±0.052	1.218±0.534	0.726±0.030	6.15±0.97	0.149±0.144
20%	0.725±0.048	0.967±0.484	0.759±0.032	6.38±0.89	0.095±0.135
50%	0.738±0.044	0.860±0.433	0.756±0.029	6.19±0.80	0.116±0.141
100%	0.752±0.042	0.788±0.413	0.767±0.028	5.88±0.75	0.069±0.112

On the 2D OASIS-1 dataset, 10% training data can only achieve a 0.691 Dice, which is 6.1% lower than that of the model trained with all the training pairs. This gap can be gradually reduced when more training data is involved. For example, a 0.738 Dice score can be achieved with 50% training data, it is still 1.4% lower than the original result. On the other hand, we notice the values of $|J|_{<0}\%$ are also decreased with more training data, implying **the model is more robust and able to produce more realistic displacements when training with more data.** Similar trends can be observed in the 3DCMR dataset, more training data improves the registration performance and produces displacement fields with fewer foldings.

Optimizer

We compare two different optimizers, i.e., SGD and Adam in Table 3.7, together with different batch sizes to investigate the impact of batch size on optimization algorithms. From Table 3.7, we find that SGD demonstrates improved performance with larger batch sizes, whereas Adam tends to exhibit diminished performance with larger batch sizes.

With a batch size of 5 and 10, SGD can achieve 0.751±0.044 and 0.752±0.044 average Dice

Table 3.7: Comparisons between SGD and Adam optimizers on OASIS-1. Three different batch sizes, i.e., 1, 5, and 10 are utilized for each optimizer.

Optimizer	Batch Size	Dice↑	$\lvert J\rvert_{<0}$%
SGD	1	0.744±0.045	0.761±0.397
	5	0.751±0.044	0.733±0.367
	10	0.752±0.044	0.761±0.404
Adam	1	0.752±0.042	0.788±0.413
	5	0.741±0.044	0.819±0.472
	10	0.742±0.044	0.770±0.397

scores, which are basically the same as the 0.752±0.042 achieved by Adam with a batch size of 1, the p-values between every two of the three models are 0.6, 0.6, and 0.9, indicating no significant evidence of difference. However, it is noticeable that a larger batch size results in more computational cost and thus a longer training time, therefore, we use Adam with a batch size of 1 as the default optimizer.

3.4.3 Model

We have evaluated the influence of objective and optimization functions. In this section, we investigate the impacts of deep network architectures on registration performance.

Activation Function

We first study the impact of different activation functions on registration performance. We have been using ReLU as the activation function in the U-Net architecture with $C = 8$, we hereby replace the ReLU with Tanh and PReLU. As reported in Table 3.8, the performance of Tanh is clearly inferior to that of ReLU and PReLU, while ReLU and PReLU can achieve

Table 3.8: Comparisons between different activation functions on OASIS-1 and 3DCMR.

Activation	OASIS-1		3DCMR		
	Dice↑	$\lvert J\rvert_{<0}$%	Dice↑	HDist↓	$\lvert J\rvert_{<0}$%
Tanh	0.744±0.040	0.675±0.367	0.759±0.030	6.30±0.76	0.235±0.142
ReLU	0.752±0.042	0.788±0.413	0.767±0.028	5.88±0.75	0.069±0.112
PReLU	0.750±0.042	0.818±0.417	0.768±0.032	6.06±0.78	0.099±0.127

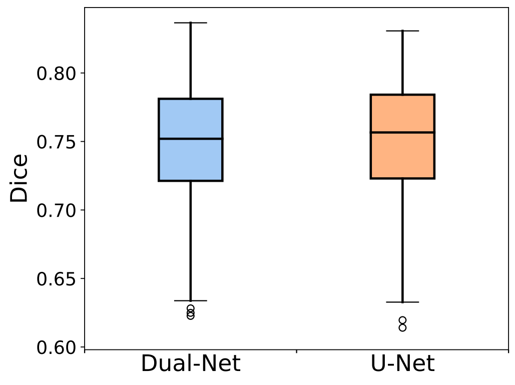
similar performance on both datasets.

Architecture

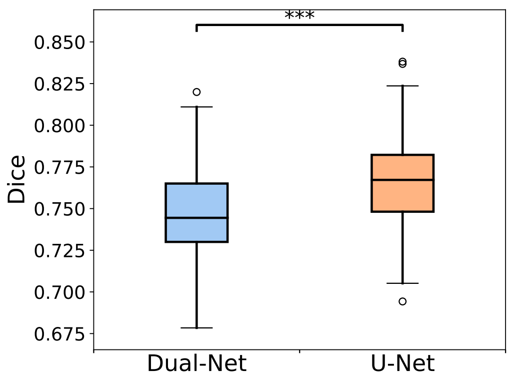
In Tables 3.9 and 3.10, we compare the registration performance of U-Net and Dual network. For a fair comparison, the Dual-Net is constructed by directly duplicating the contracting layers in the U-Net. The weights of two contracting branches are separately learned. Experimental results in Tables 3.9 and 3.10 suggest the two networks can achieve similar results. Specifically, while the mean Dice score achieved by Dual-Net is 0.002 lower than U-Net on 2D OASIS, it however has a smaller variation (0.040 versus 0.042). Moreover, their p-value is 0.449, suggesting no evidence of difference. Similarly, Dual-Net achieves a better HDist on the 3DCMR dataset, while U-Net achieves a better Dice score. A more direct comparison is given by the boxplots with statistical p-values in Fig. 3.11, where no significant differences are observed on the 2D OASIS-1 dataset ($P > 0.05$, not significant). On the 3DCMR dataset, the Dual-Net produces lower Dice while better HDist, with $P \leq 0.001$ and $P \leq 0.01$, marked with three asterisks $***$ and two asterisks $**$, respectively. It is therefore difficult to distinguish a better architecture.

Model Size

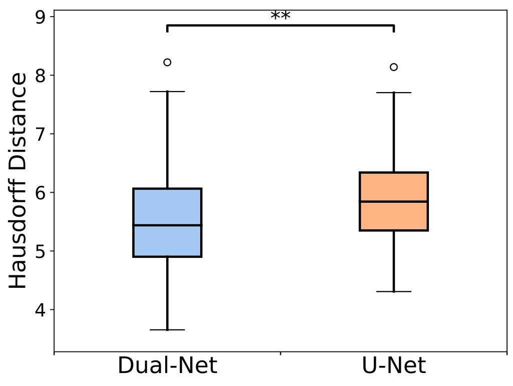
To this end, we wonder if the performance of U-Net can be further improved by utilizing



(a) OASIS-1 Dice



(b) 3DCMR Dice



(c) 3DCMR HDist

Figure 3.11: Boxplots of Dice and HDist achieved by Dual-Net and U-Net.

Table 3.9: Comparisions between Dual-Net and U-Net on OASIS-1.

Network	Dice↑	$\lvert J\rvert_{<0}\%$	Params	Mult-adds	Memory
Dual	0.750±0.040	0.894±0.421	600,176	670.54	31.64
U-Net	0.752±0.042	0.788±0.413	379,136	555.30	24.43

heavier models, i.e., increasing the number of parameters. A straightforward approach is directly changing the number of channels in each convolutional layer, in Tables 3.11 and 3.12, we experiment with three more different settings, i.e., 4, 16, and 32 instead of the original $C = 8$. First of all, it is clear that registration performance in terms of Dice is all

Table 3.10: Comparisions between Dual-Net and U-Net on 3DCMR.

Network	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$	Params	Mult-adds	Memory
Dual	0.747±0.027	5.56±0.88	0.167±0.159	1,776,000	57.44	1237.88
U-Net	0.767±0.028	5.88±0.75	0.069±0.112	1,113,744	50.98	982.31

Table 3.11: Comparisons between different numbers of channels on the 2D OASIS-1 dataset.

C	Dice↑	$\lvert J\rvert_{<0}\%$	Params	Mult-adds(G)	Memory
4	0.732±0.044	0.829±0.441	95,088	0.14	12.45
8	0.752±0.042	0.788±0.413	379,136	0.54	24.43
16	0.759±0.041	0.799±0.384	1,514,112	2.19	48.40
32	0.764±0.040	0.734±0.359	6,051,584	8.73	96.33

Table 3.12: Comparisons between different numbers of channels on the 3DCMR dataset.

C	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$	Params	Mult-adds(G)	Memory
4	0.744±0.027	5.52±0.84	0.099±0.138	279,064	13.42	509.16
8	0.767±0.028	5.88±0.75	0.069±0.112	1,113,744	50.98	982.31
16	0.770±0.028	5.91±0.78	0.094±0.121	4,449,952	198.49	1928.62
32	0.772±0.028	5.95±0.77	0.168±0.140	17,789,760	783.08	3821.25

improved with more channels, for example, the Dice score is improved by 3.2% and 2.8% from C=4 to C=32 on OASIS-1 and 3DCMR, respectively. However, a non-trivial fact is that the computational costs in terms of mult-adds and memory usage are also rapidly increased with the increment of channels. For example, on the 2D U-Net, the mult-adds are increased more than 62 and 16 times by changing C=4 and C=8 to C=32, **it is therefore questioned whether the achieved improvement by sacrificing the computational efficiency is meaningful.**

Table 3.13: Comparsions with the Flash on 2D OASIS-1.

Methods	Dice↑	$\|J\|_{<0}\%$	CPU Runtime
Flash (16×16)	0.702±0.051	0.033±0.126	13.699
Flash (20×24)	0.727±0.046	0.205±0.279	22.575
Flash (40×48)	0.734±0.045	0.049±0.080	85.773
U-Net	0.752±0.042	0.788±0.413	0.010
Diff-U-Net	0.751±0.040	<0.00008	0.013

3.4.4 Discussion

In Table 3.13, we compare the registration performance of U-Net, diffeomorphic U-Net, and Flash [163], which is a state-of-the-art iterative diffeomorphic registration method. For Flash, since it estimates a low-dimensional band-limited patch to accelerate the registration process, we experimented with three different patches, i.e., 16×16, 20×24, and 40×48, where 16×16 is the recommended default resolution. We can observe that a larger patch will increase the runtime but improve the registration performance of Flash. It is also clear that the two deep models outperform the Flash in terms of both Dice and CPU runtime. We note that the CPU runtimes (in seconds) are averaged on the whole testing set and tested on the same machine with 128G RAM and 16 3.80GHz Intel(R) Core(TM) i7-9800X CPUs.

3.5 Summary

In Section 3.2, we reviewed the current state-of-the-art deep registration approaches. We summarized in Section 3.3 that a deep registration algorithm is composed of three core components: a deep network architecture (U-Net, Dual-Net), an objective function (SSD, NCC, NMI, diffusion regularization, bending energy), and an optimizer (SGD, Adam). We, therefore, performed extensive parameter studies on each of the components on two different datasets in Section 3.4. From the extensive experiments, we highlight that a deep unsupervised registration model is able to achieve competitive registration performance with the traditional iterative optimization-based approach while being faster and more flexible. However, the performance of the deep registration model relies on the amount of training data, the selection of objective functions, optimizers, and network architectures. Different configurations often yield diverse results.

Chapter Four

Deep Registration: Advances

This chapter is based on the following publication:

U-net vs transformer: Is u-net outdated in medical image registration? [74] which was published in the International Workshop on Machine Learning in Medical Imaging.

4.1 Introduction

This chapter introduces three extensions for the U-Net model described in Chapter 3. The first extension is the Large Kernel U-Net (LKU-Net), which is proposed to boost the long-range dependency modeling ability of U-Net. Inspired by RepVGG and RepLK-ResNet [34, 35] that use multiple parallel convolutional layers with various kernel sizes to enlarge the effective receptive field, LKU-Net uses four parallel convolutional layers to replace the 3×3×3 convolutional layer in U-Net. In experiments, by comparing the proposed LKU-Net with the plain U-Net and more recent transformer-based architectures such as TransMorph [21] on two public brain MRI datasets, we prove that the classical U-Net is still competitive with recent transformer-based approaches in terms of both registration accuracy and speed.

Built on LKU-Net, we further proposed a cascaded version, named Cascaded LKU-Net. Each cascade in Cascaded LKU-Net progressively takes the warped moving image produced by the previous cascade (the original moving image for the first cascade) and the original fixed image as input and estimates the displacement between them. By composing the displacements estimated in every cascade altogether, Cascaded LKU-Net can learn large complicated displacements. Experimental results on the NLST 3D lung CT dataset prove the superiority of Cascaded LKU-Net over LKU-Net as well as other state-of-the-art methods.

The third extension is LKU-Net-Affine, which supports the 3D rigid and affine parametric registration. Since the learning of rigid and affine matrices only involves limited parameters, we are able to discard the whole expansive path of LKU-Net and replace it with multiple fully connected modules to estimate the rigid and affine parameters. The proposed LKU-Net-Affine is evaluated on three public high-resolution expansion microscopy datasets from the ISBI 2023 RnR-ExM challenge and proved to be efficient.

4.2 Large Kernel U-Net (LKU-Net)

Due to their extremely long-range modeling capability, vision transformer-based networks have become increasingly popular in deformable image registration. We believe, however, that the receptive field of a 5-layer convolutional U-Net is sufficient to capture accurate deformations without needing long-range dependencies. The first purpose of this section is therefore to investigate whether U-Net-based methods are outdated compared to modern transformer-based approaches when applied to medical image registration.

4.2.1 Motivation

Many iterative optimization approaches [7, 13, 130, 143] have been proposed to tackle intensity-based deformable registration, and shown great registration accuracy. However, such methods suffer from instance-level optimization and slow inference speeds. Though some works, such as Nesterov accelerated ADMM [140] and ConvexAdam [133], propose certain techniques to accelerate the computation, their speed still can not compare to approaches purely based on deep networks. Deep data-driven methods [10, 11, 81, 110, 122, 161, 166] have become a powerful benchmark for large-scale medical image registration, due to their fast inference speed and comparable accuracy with iterative methods.

Deep registration methods [10, 110, 122, 161] directly take moving and fixed image pairs as input, and estimate deformations in one forward pass. Most deep methods use a U-Net style architecture [125] as their backbone and only vary preprocessing steps and loss functions. Such an architecture includes a contraction path (encoder) to encode the spatial information from the input image pair, and an expansion path (decoder) to decode the spatial information to compute a deformation field (or stationary velocity field in the case of diffeomorphic registration). Inspired by the success of the transformer architecture [36,

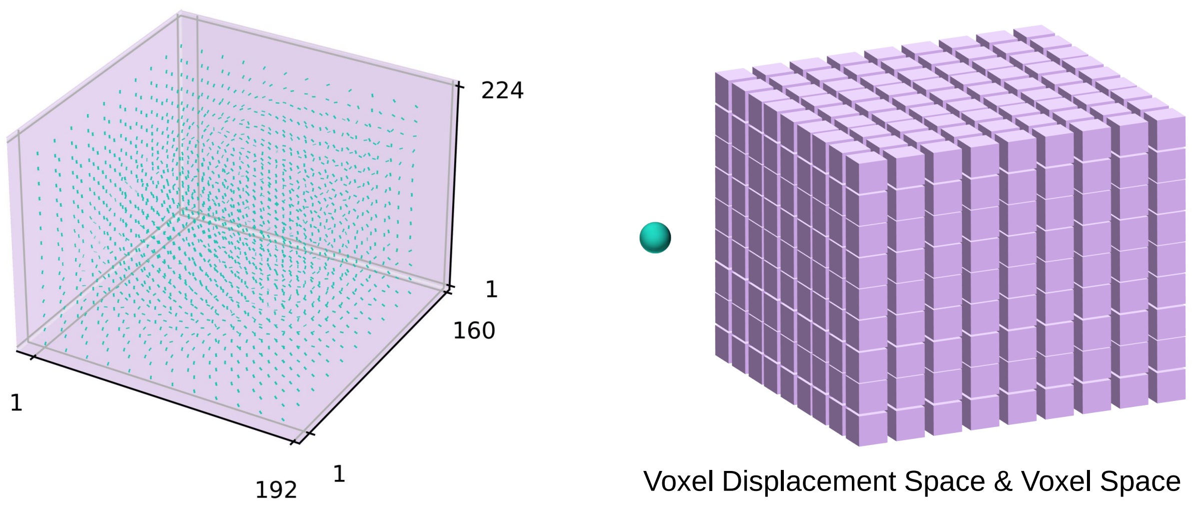


Figure 4.1: Displacement fields computed from the IXI brain dataset. The left figure plots the displacement vectors in voxel averaged over the whole training data (average lengths of these vectors along x-axis, y-axis, and z-axis are 2.1 voxels, 2.3 voxels, and 1.4 voxels, respectively). The right figure is an illustration of the left figure where vectors are represented by a sphere, the size of which is much smaller than the cubic which represents the true size of the volumetric image.

142], several recent registration works [21, 22, 164] have used it as the backbone to replace the standard convolutional U-Net. In this section, we investigate whether convolutional U-Net is outdated compared to modern vision transformer architectures, such as the new state-of-the-art TransMorph [21], for image registration.

The motivation for this work can be found in Fig. 4.1, where we plotted the average displacement vector fields of the IXI brain dataset estimated by TransMorph. We notice that the average length of displacements along the x-axis, y-axis, and z-axis are 2.1 voxels, 2.3 voxels, and 1.4 voxels, respectively. Moreover, the maximum displacements along the x, y, and z axes averaged on the whole testing set are 13, 19, and 15 voxels, respectively. These displacements are very small compared to the actual volumetric size (160×192×224) of the image. Therefore, we argue that it may not be necessary to adopt a transformer to model long-range dependencies for deformable image registration. We believe the theoretical receptive field of the plain U-Net is enough for modeling displacement of such length. However, according to [34, 35], the effective receptive field of a network might be smaller in practice. We, therefore, propose a large kernel U-Net (LKU-Net) by increasing the effective receptive field of a vanilla U-Net with large kernel blocks and show that our LKU-Net outperforms TransMorph by using only 1.12% of its parameters and 10.8% of its mult-adds.

4.2.2 Related Works

U-Net-based registration: VoxelMorph [10, 11], one of the pioneering works for medical image registration, used a 5-layer U-Net style network to estimate the full-resolution displacement. VoxelMorph has achieved comparable accuracy to state-of-the-art traditional methods (such as ANTs [7]) while operating orders of magnitude faster. Inspired by its success, many subsequent registration pipelines [75, 81, 110, 161, 166] used the U-Net style architecture as their registration network backbone. Among them, some work [75, 166] cascaded multiple U-Nets to estimate deformations. An initial coarse deformation was first predicted and the resulting coarse deformation was then refined by subsequent networks. This coarse-to-fine method usually improves the final performance, but the number of cascades is restricted by the GPU memory available, so training them on large-scale datasets may not be feasible with small GPUs. In experiments, we show that, without any cascading, a single U-Net style architecture can already outperform transformer-based networks.

Transformer-based registration: Transformer [142], originally proposed for machine translation tasks, is built upon the attention mechanism. Recently, this architecture has been rapidly explored in computer vision tasks [36, 94], because it successfully alleviates the inductive biases of convolutions and is capable of capturing extreme long-range dependencies.

Some recent registration methods [21, 22, 164] have embedded the transformer as a block in the U-Net architecture to predict dense deformations. Building on a 5-layer U-Net, Zhang et al. [164] proposed a dual transformer network (DTN) for diffeomorphic image registration, but such a dual setting requires lots of GPU memory and greatly increases the computational complexity. Chen et al. [22] proposed ViT-V-Net by adopting the vision transformer (ViT) [36] block in a V-Net style convolutional network [105]. To reduce the computational cost, they input encoded image features to ViT-V-Net instead of image pairs. Their results were comparable with VoxelMorph. The authors of ViT-V-Net [22] later improved upon ViT-V-Net and proposed TransMorph [21] by adopting a more advanced transformer architecture (Swin-Transformer [94]) as its backbone. They conducted thorough experiments and showed the superiority of their TransMorph [21] over several state-of-the-art methods.

4.2.3 Methodology

Large kernel (LK) block: According to [3], it is easy to compute that the receptive field of a vanilla 5-layer $3 \times 3 \times 3$ U-Net is large enough to cover the area which could impact the deformation field around a given voxel. However, as per RepVGG [34] and RepLK-ResNet [35], in practice the effective receptive field of a convolutional network is much smaller than the theoretical one we compute. We therefore adopt a LK block to increase the effective receptive field, as illustrated in Fig. 4.2. Specifically, there are four parallel sub-layers in each LK block, including a large kernel convolutional layer ($k \times k \times k, k \geq 3$), a 3×3×3 layer, a 1×1×1 layer, and an identity shortcut. The subsequent outputs of these sub-layers are then added element-wise to produce the final output, which is followed by a PReLU activation layer to introduce non-linearity. The parallel paths in each LK block not only handle distant spatial information but also capture and fuse spatial information at a finer scale.

Directly enlarging the kernel size of a convolutional layer leads to the number of parameters

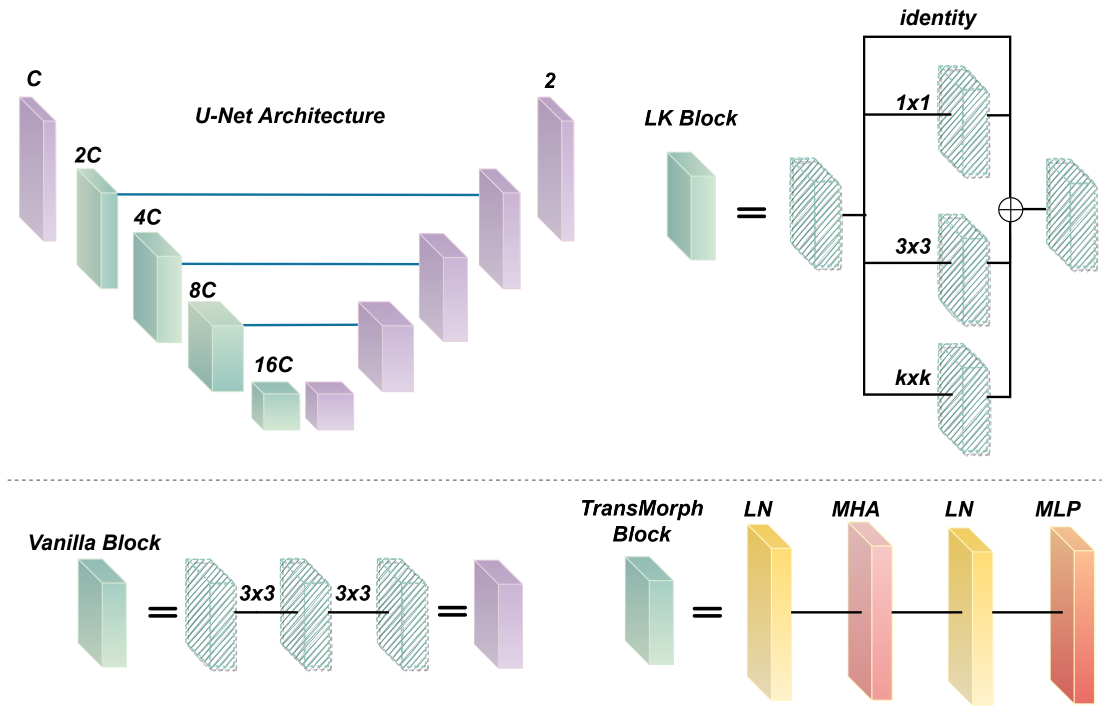


Figure 4.2: Blocks used in U-Net, TransMorph, and LKU-Net. The vanilla U-Net uses the same blocks in both the encoder and decoder, each consisting of multiple sequential 3×3 convolutional layers. TransMorph replaces four convolutional blocks in the encoder with transformer-based blocks, each of which is built on a combination of layer norm (LN), multi-head self-attention (MHA), and multi-layer perceptron (MLP). For a fair comparison with TransMorph, we use four LK blocks in LKU-Net. Each LK block contains one identity shortcut and three parallel convolutional layers (that have kernel sizes of 1×1, 3×3, and $k \times k$, respectively). The outputs of all sub-layers are combined through element-wise addition, with PReLU used as the activation function to introduce non-linearity to the LK block.

growing exponentially. For example, the number of parameters increases by 463% and 1270% when enlarging a 3×3×3 kernel to 5×5×5 and 7×7×7, respectively. The resulting network is then cumbersome and prone to collapsing or over-fitting during training. The benefits of using the proposed LK block are that both the identity shortcut and the 1×1×1 convolutional layer help the training. These numerical results are listed in our ablation studies (Table 4.1).

LKU-Net: We then propose LKU-Net for registration by integrating the LK blocks into the vanilla U-Net as in Fig. 4.2. In order to perform a fair comparison with TransMorph, which uses four transformer blocks in the contracting path of its architecture, we only use four LK blocks in the contracting path of the proposed LKU-Net. Note that, the proposed LK block is a plug-in block that can be integrated into any convolutional network.

Parameterization: The resulting LKU-Net takes a stacked image pair as input and outputs the estimated deformation. LKU-Net itself has two sets of architecture-specific hyperparameters: the number of kernels and the size of the kernels in each convolutional block. For simplicity, we set the number of kernels in the first layer as C; then the number of kernels is doubled after each down-sampling layer in the contraction path and halved after each up-sampling layer in the expansion path; the number of kernels in the last layer is set to 3 for 3D displacements (2 for 2D displacements). On the other hand, though multiple LK blocks are used within our LKU-Net, we use the same kernel size $k \times k \times k$ for all LK blocks.

Diffeomorphism: Besides directly estimating displacements from LKU-Net, we also proposed a diffeomorphic variant, termed Diff-LKU-Net, in which the final output of the LKU-Net is a stationary velocity field $\boldsymbol{v}$. We then use seven scaling and squaring layers to induce diffeomorphisms, i.e., the final deformation $\boldsymbol{\phi} = Exp(\boldsymbol{v})$ as in [5, 31].

Network loss: We adopt an unsupervised loss composed of both a data term (NCC) and a diffusion regularization term (applied to either the displacement or velocity field), which are balanced by a hyperparameter λ. The overall loss $\mathcal{L}(\boldsymbol{\Theta})$ is defined as

$$\mathcal{L}(\boldsymbol{\Theta}) = -\frac{1}{N}\sum_{i=1}^{N} \mathrm{NCC}(\boldsymbol{I}_{M^{(i)}} \circ \boldsymbol{\phi}_i(\boldsymbol{\Theta}) - \boldsymbol{I}_{F^{(i)}}) + \frac{\lambda}{N}\sum_{i=1}^{N} \|\nabla \boldsymbol{\phi}_i(\boldsymbol{\Theta})\|_2^2 \tag{4.1}$$

or

$$\mathcal{L}(\boldsymbol{\Theta}) = -\frac{1}{N}\sum_{i=1}^{N} \mathrm{NCC}(\boldsymbol{I}_{M^{(i)}} \circ \boldsymbol{\phi}_i(\boldsymbol{\Theta}) - \boldsymbol{I}_{F^{(i)}}) + \frac{\lambda}{N}\sum_{i=1}^{N} \|\nabla \boldsymbol{v}_i(\boldsymbol{\Theta})\|_2^2, \tag{4.2}$$

where N is the number of training pairs, $\boldsymbol{I}_F$ denotes the fixed image, $\boldsymbol{I}_M$ represents the moving image, $\boldsymbol{\Theta}$ are the network parameters to be learned, $\circ$ is the warping operator, and ∇ is the first order gradient operator.

4.2.4 Experiments

Datasets: We used two datasets in our experiments. First, **OASIS-1** dataset [102] consists of 416 cross-sectional T1-weighted MRI scans. We used the pre-processed version (including 414 3D scans and 414 2D images) [67] provided by the Learn2Reg 2021 challenge [63] for inter-subject brain registration. Each scan has been skull stripped, aligned, normalized, and has a resolution of 160×192×224. Label masks of 35 anatomical structures were used to evaluate registration performance. In this dataset, there are 394 scans (unpaired) for training, and 19 image pairs (20 scans) for validation and public leaderboard ranking. We report our 3D results on their validation set in Table 4.3. For fast evaluation of different methods and parameters, in Table 4.1, we used 414 2D images with size 160×192, each being one slice of its respective 3D volume. Each 2D segmentation mask contains 24 automated labels. We randomly split the data into 201 images for training, 12 images for validation, and 201 images for testing. After pairing, we used 40200 ([201-1]$\times$200), 22 ([12- 1]$\times$2), and 400 ([201-1]$\times$2) image pairs for training, validation, and testing, respectively.

Second, **IXI** dataset[1] contains nearly 600 3D MRI scans from healthy subjects, collected at three different hospitals. We used the pre-processed IXI data and the exact training/testing/validation provided by [21]. Specifically, we used 576 T1-weighted brain MRI images to perform atlas-to-subject brain registration, in which 403, 58, and 115 images were used for training, validation, and testing, respectively. The atlas is generated by TransMorph [21] using CycleMorph [81]. All volumes were cropped to the size of 160×192×224. Label maps of 30 anatomical structures were used to evaluate registration performance.

[1]IXI data is available in https://brain-development.org/ixi-dataset/

Table 4.1: Ablation and Parameter Studies of LKU-Net on 2D OASIS-1.

Method	Model	k	C	Identity	1x1	Dice↑	Parameters	Mult-Adds (M)
A1	U-Net	-	8	-	-	76.16(4.08)	379,136	555
A2	U-Net	5	8	-	-	76.25(4.04)	596,224	666
A3	U-Net	7	8	-	-	76.41(4.13)	921,856	831
A4	U-Net	9	8	-	-	76.33(3.98)	1,356,032	1050
A5	U-Net	11	8	-	-	75.80(4.03)	1,898,752	1330
B1	LKU-Net	3	8	✓	✗	76.26(4.18)	390,768	604
B2	LKU-Net	3	8	✗	✓	76.40(4.06)	400,416	611
B3	LKU-Net	3	8	✓	✓	76.47(3.98)	400,416	611
B4	LKU-Net	5	8	✓	✗	76.36(4.08)	542,320	707
B5	LKU-Net	5	8	✗	✓	76.30(4.06)	551,968	713
B6	LKU-Net	5	8	✓	✓	76.51(4.10)	551,968	713
C1	LKU-Net	7	8	✓	✓	76.55(4.06)	779,296	867
C2	LKU-Net	9	8	✓	✓	76.45(4.03)	1,082,400	1070
C3	LKU-Net	11	8	✓	✓	76.31(4.05)	1,461,280	1330
D1	LKU-Net	5	16	✓	✓	77.19(3.86)	2,204,864	2830
D2	LKU-Net	5	32	✓	✓	77.38(3.89)	8,813,440	11250
D3	LKU-Net	7	32	✓	✓	77.52(3.90)	12,450,688	13710

Implementation details: The vanilla 5-layer U-Net architecture used in this chapter is the same one from Chapter 3, which was first proposed in [161], the only change we made was setting all kernels to 3×3×3. 2D U-Net shares the same architecture except that all kernels are changed from 3D convolutions to 2D 3×3 convolutions. In all experiments, we used the Adam optimizer, with batch size being set to 1, and the learning rate being kept fixed at 1×10^{-4} throughout training. Note that for the 3D OASIS registration (Table 4.3), following [21], we additionally adopt an extra Dice Loss (see Eq. (4.5)) with weight being 1.

Ablation and parameter studies: In Table 4.1, we compare the registration performance of our LKU-Net with the vanilla U-Net using Dice on 2D OASIS-1 data. Methods A1-A5 in Table 4.1 indicate that using different kernels in the vanilla U-Net affects the network's performance. Specifically, replacing all 3×3 kernels with 5×5 and 7×7 ones improves Dice by 0.09 and 0.25, respectively. However, when using 9×9 and 11×11 kernels, the performance begins to decline. Comparing the results of Methods A1 & B3, A2 & B6, A3 & C1, A4 & C2, and A5 & C3, it is easy to see that our LKU-Net outperforms the U-Net when we use the same kernels size k, and that LKU-Net is consistently better than the vanilla U-Net (A1). Meanwhile, the results from B1-B6 suggest that using either the identity shortcut or the 1×1 layer improves the registration performance, and that combining both leads to the best performance. Comparing B3, B6, and C1, we see that the performance of LKU-Net improves when we increase the kernel size. Lastly, comparing B6, D1, and D2, we find that using larger models also improves the performance, i.e., when we increase C from 8 to 16 and to 32 in LKU-Net$_{8,5}$, Dice improves from 76.51 to 77.19 and to 77.38, respectively.

Atlas-to-subject registration on IXI: To guarantee a fair comparison with TransMorph [21] on this dataset, we used the exact same data pre-processing, training/validation split, and testing protocol. We report our results in Table 4.2, where C and k in U-Net$_C$ and LKU-Net$_{C,k}$ denote the number of kernels in the first layer and the kernel size in the LK blocks, respectively.

In the case of non-diffeomorphic registration, U-Net$_4$ achieves 0.733 Dice score which is comparable with VoxelMorph-1 (0.729), VoxelMorph-2 (0.732), ViT-V-Net (0.734), CoTr (0.735), and PVT (0.727), but lower than those of nnFormer (0.747), TransMorph (0.754), and TransMorph-Bayes (0.753). However, LKU-Net$_{4,5}$ outperforms all other competing methods with a Dice score of 0.758. Note that the parameters and mult-adds of LKU-Net$_{4,5}$ are only 1.12% and 10.8% of those of TransMorph, respectively. Furthermore, by increasing the number of channels from $C = 4$ (LKU-Net$_{4,5}$) to $C = 8$ (LKU-Net$_{8,5}$), a better Dice score of 0.765 can be achieved. The visual comparison of estimated deformations is given in Fig. 4.3. For diffeomorphic registration, Diff-U-Net$_4$, Diff-LKU-Net$_{4,5}$, and Diff-LKU-Net$_{8,5}$ achieve Dice scores of 0.751, 0.752, and 0.760, respectively. They all outperform

Table 4.2: Performance comparison between different methods on IXI. Note that the listed results (except the last six rows) are directly taken from TransMorph [21].

Model	Dice	% of $\lvert J\rvert <= 0$	Parameters	Mult-Adds (G)
Affine	0.386±0.195	-	-	-
SyN	0.645±0.152	<0.0001	-	-
NiftyReg	0.645±0.167	0.020±0.046	-	-
LDDMM	0.680±0.135	<0.0001	-	-
deedsBCV	0.733±0.126	0.147±0.050	-	-
VoxelMorph-1	0.729±0.129	1.590±0.339	274,387	304.05
VoxelMorph-2	0.732±0.123	1.522±0.336	301,411	398.81
Diff-VoxelMorph	0.580±0.165	<0.0001	307,878	89.67
CycleMorph	0.737±0.123	1.719±0.382	361,299	49.42
MIDIR	0.742±0.128	<0.0001	266,387	47.05
ViT-V-Net	0.734±0.124	1.609±0.319	9,815,431	10.60
PVT	0.727±0.128	1.858±0.314	58,749,007	193.61
CoTr	0.735±0.135	1.292±0.342	38,684,995	1461.61
nnFormer	0.747±0.135	1.595±0.358	34,415,851	686.77
TransMorph	0.754±0.124	1.579±0.328	46,771,251	657.64
TransMorph-Bayes	0.753±0.123	1.560±0.333	21,205,491	657.69
B-Spline-TransMorph	0.761±0.122	<0.0001	46,806,307	425.95
Diff-TransMorph	0.594±0.163	<0.0001	46,557,414	252.61
U-Net$_4$	0.733±0.125	1.524±0.353	279,086	58.73
Diff-U-Net$_4$	0.751±0.123	<0.0001	279,086	58.73
LKU-Net$_{4,5}$	0.758±0.130	0.023±0.018	522,302	71.00
Diff-LKU-Net$_{4,5}$	0.752±0.132	<0.0001	522,302	71.00
LKU-Net$_{8,5}$	0.765±0.129	0.117±0.058	2,086,342	272.09
Diff-LKU-Net$_{8,5}$	0.760±0.132	<0.0001	2,086,342	272.09

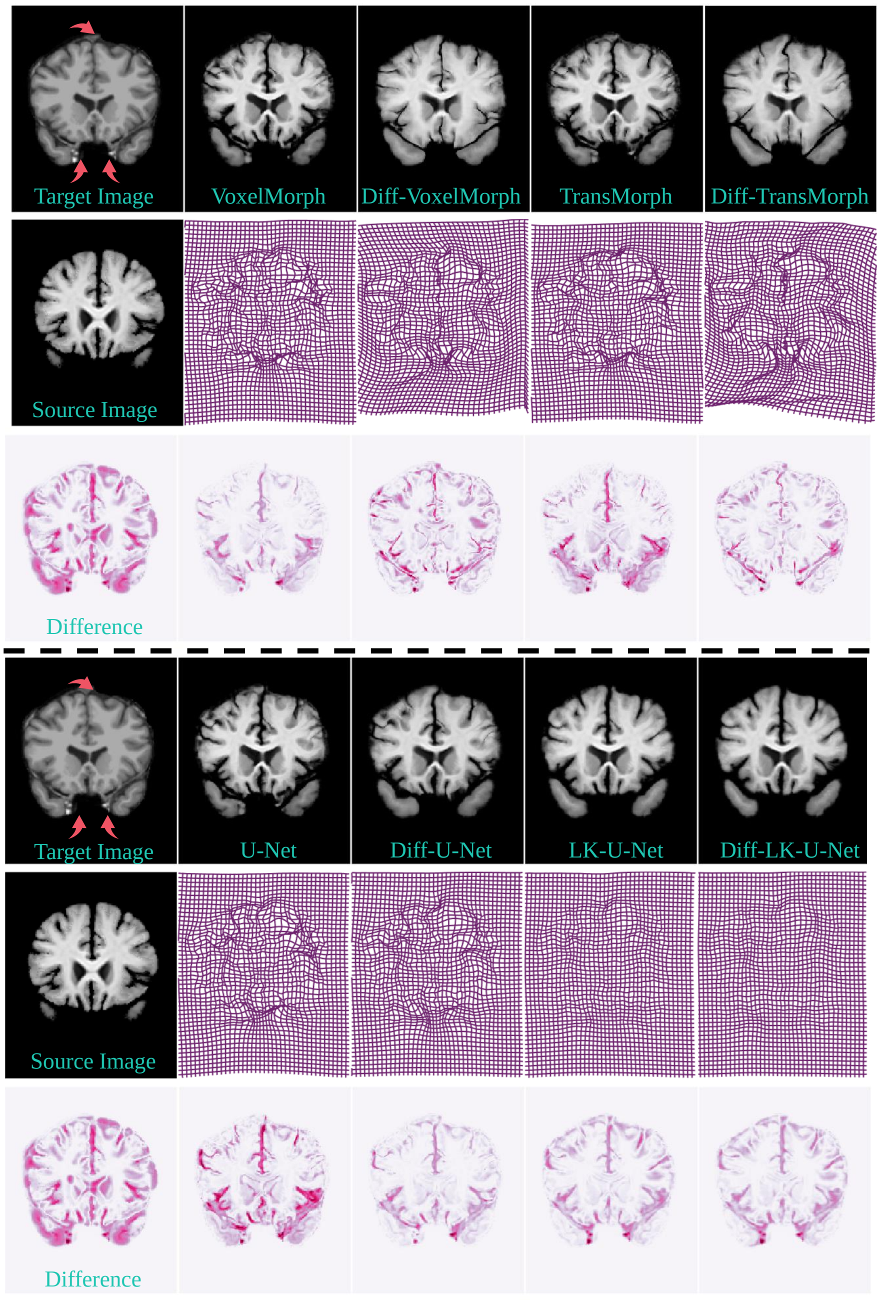


Figure 4.3: Visual comparison between different methods. The first column displays the fixed image, the moving image, and the difference between them. Excluding the first column, from top to bottom we show the warped moving images, the deformations, and the difference maps (between warped moving images and fixed image) of different methods. LKU-Net is able to produce smooth deformations and hence the most realistic warped moving images (see the regions marked with pink arrows).

Table 4.3: Performance comparison of different methods on the validation set of Learn2Reg 2021 challenge Task 3. HDist95 is 95% percentile of Hausdorff distance of segmentations. The standard deviation of the logarithmic Jacobian determinant (SDlogJ) is a measurement of the smoothness, a lower SDlogJ indicates better smoothness.

Methods	Dice	HDist95	SDlogJ
ConvexAdam[133]	0.846±0.016	1.500	0.067
LapIRN[111]	0.861±0.015	1.514	0.072
TransMorph[21]	0.858±0.014	1.494	0.118
TransMorph-Large[21]	0.862±0.014	1.431	0.128
U-Net	0.866±0.015	1.374	0.096
LKU-Net	**0.886±0.015**	**1.262**	0.517

Diff-TransMorph (0.594) by a large margin. The Dice of Diff-LKU-Net$_{8,5}$ (0.760) is comparable to that of B-Spline-TransMorph (0.761) but uses fewer parameters and mult-adds.

Subject-to-subject registration on OASIS: We further evaluated the proposed LKU-Net on the validation set of MICCAI Learn2Reg 2021 challenge (Task 3), and the results are shown in Table 4.3. The plain U-Net ($C = 32$) has already outperformed ConvexAdam [133], LapIRN [111], TransMorph, and TransMorph-Large (which is a larger version of TransMorph) [21] in terms of both Dice and HDist95. The proposed LKU-Net ($C = 32, k = 5$) also outperforms all the compared methods with large margins over Dice and HDist95. However, the displacement fields produced by LKU-Net exhibit the least smoothing compared to other methods, as indicated by the highest SDlogJ.

With the experimental results on the two public brain datasets, we highlight that **the plain U-Net can achieve comparable registration performance with transformer-based methods for both inter-subject and atlas-to-subject registration tasks**. The proposed LKU-Net can outperform modern transformer-based methods, these results prove that U-Net is still competitive if devised properly.

4.3 Cascaded LKU-Net

Either U-Net or LKU-Net estimates the displacement with one forward propagation, this approach is efficient but can be less accurate when dealing with large displacement existing on complicated structures, as suggested by RC-Net [166]. Therefore, following RC-Net, we propose Cascaded LKU-Net to support large complicated displacement estimation.

4.3.1 Methodology

Cascaded LKU-Net: Fig. 4.4 presents the overall pipeline for a Cascaded LKU-Net with k cascades. The process in the first cascade is exactly the same as in the single LKU-Net, which takes the original image pair (I_M and I_F) as input and estimates an initial deformation field $\delta\phi^{(1)}$, with the estimated $\delta\phi^{(1)}$, we are able to generate a warped image $I_M^{w(1)}$. The input images of the second cascade however will be accordingly changed to $I_M^{w(1)}$ and I_F, the second cascade estimates the second deformation field $\delta\phi^{(2)}$ and generates the second warped image $I_M^{w(2)}$ by warping $I_M^{w(1)}$. Successively, the k^{th} cascade takes the warped moving image $I_M^{w(k-1)}$ generated from $(k-1)^{\text{th}}$ cascade as input and estimate $\delta\phi^{(k)}$, by warping $I_M^{w(k-1)}$ with $\delta\phi^{(k)}$, the final warped image $I_M^{w(k)}$ can be obtained. The final deformation field ϕ^* can be composed by the estimated deformation fields from all the intermediate cascades,

$$\phi^* = \delta\phi^{(1)} \circ \delta\phi^{(2)} \circ \cdots \circ \delta\phi^{(k-1)} \circ \delta\phi^{(k)}. \tag{4.3}$$

Similarly, the final warped image I_M^k can also be obtained by directly warping I_M with ϕ^*,

$$I_M^{w(k)} = I_M \circ \phi^* = ((((I_M \circ \delta\phi^{(1)}) \circ \delta\phi^{(2)}) \circ \cdots) \circ \delta\phi^{(k-1)}) \circ \delta\phi^{(k)}. \tag{4.4}$$

Objective Function: It is noteworthy that though many cascades exist, the training objective function is solely imposed at the top of the cascaded network to encourage that the final composed deformation can accurately capture the correspondence between the moving and fixed image. The loss function of unsupervised Cascaded LKU-Net can be the same as Eq. (3.1). Moreover, to alleviate the segmentation masks some tasks may provide, we can employ the Dice-based segmentation loss [10]

$$\mathcal{L}_{seg}(\boldsymbol{\Theta}) = \frac{1}{N_A} \sum_{n_a=1}^{N_A} -Dice(S_M^{(n_a)} \circ \boldsymbol{\phi}(\boldsymbol{\Theta}), S_F^{(n_a)}), \tag{4.5}$$

where S_M and S_F are respectively the segmentation masks of I_M and I_F. N_A is the number of labeled anatomical structures. The Dice loss is averaged on all anatomical structures.

4.3.2 Experiments

Dataset: The National Lung Screening Trial (**NLST**) Lung CT dataset [138, 139] from the MICCAI 2022 Learn2Reg challenge is used to evaluate the proposed Cascaded LKU-Net. This task is to perform intra-patient registration of follow-up longitudinal lung CT scans. The released NLST dataset from the Learn2Reg organizers contains 220 training cases and 30 test cases (from which there are 10 pairs used for validation as well), additionally, there are automatically generated lung masks and key points on the training images for deep learning supervision. The lung masks and image data are in the resolution of 224×192×224 voxels.

Implmentation Details: The C in Cascaded LKU-Net is set to be 32. The data term and regularization term are respectively set to NCC and diffusion regularizer, similar to Eq. (4.1). To utilize the given auxiliary lung mask and key points, we first map the key points to an additional mask with the resolution of 224×192×224 and impose the Dice loss (see Eq. (4.5)). Adam optimizer with a batch size of 1 is employed. The maximum number of iterations is set to be 20,000, denoting 100 training epochs.

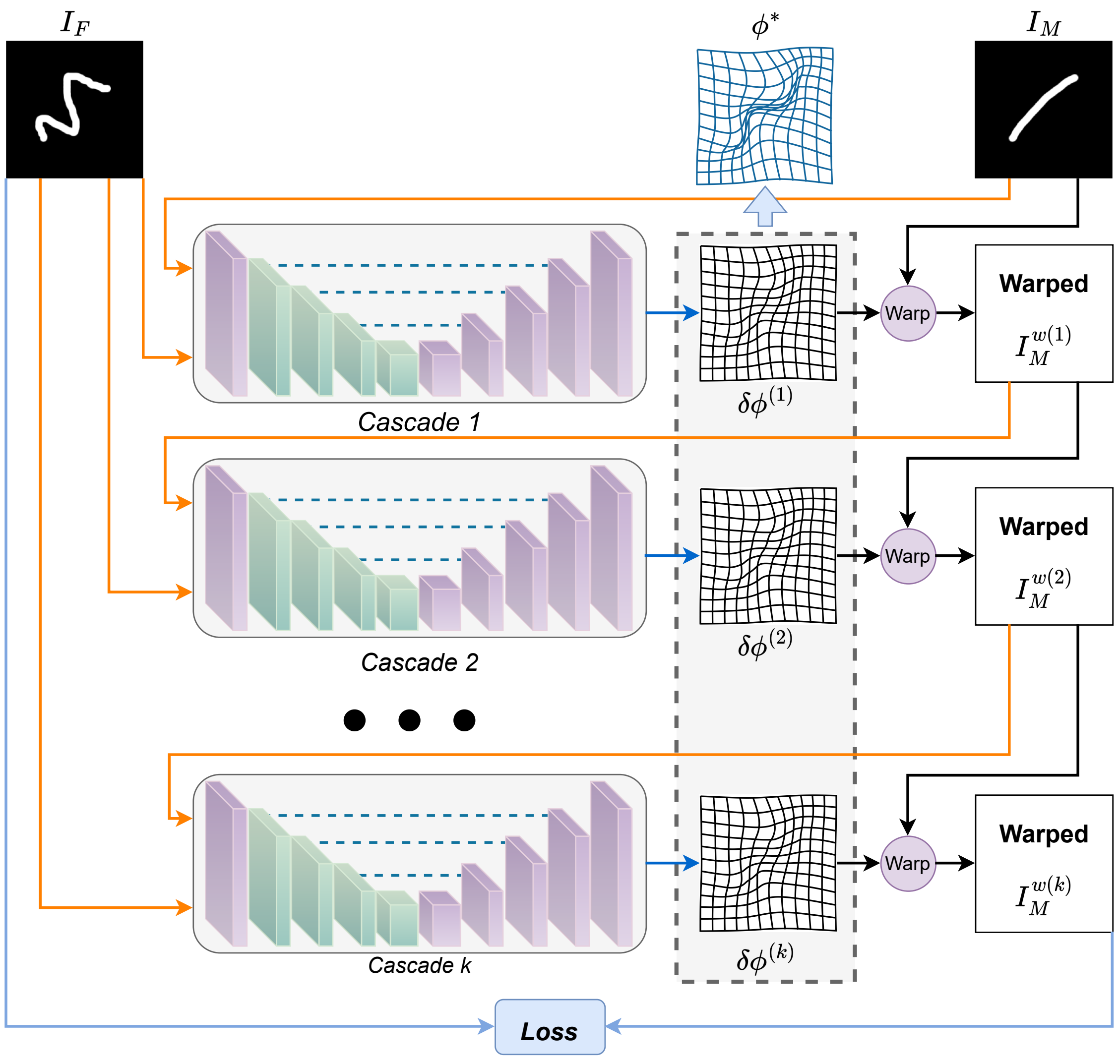


Figure 4.4: Architecture of the proposed Cascaded LKU-Net. The final deformation field ϕ^* is composed by the deformations from each cascade. The loss function is used only once at the top of the network.

Paramter Study: In Table 4.4, we first compare the performance of LKU-Net and Cascaded LKU-Net on the public validation set. We note that while using 2 and 3 cascades respectively doubles and triples the computation cost of LKU-Net, **the experimental results clearly suggest that more cascades will improve the registration performance** in terms of target registration error on the key points, i.e., TRE(KP). For example, the TRE(KP)

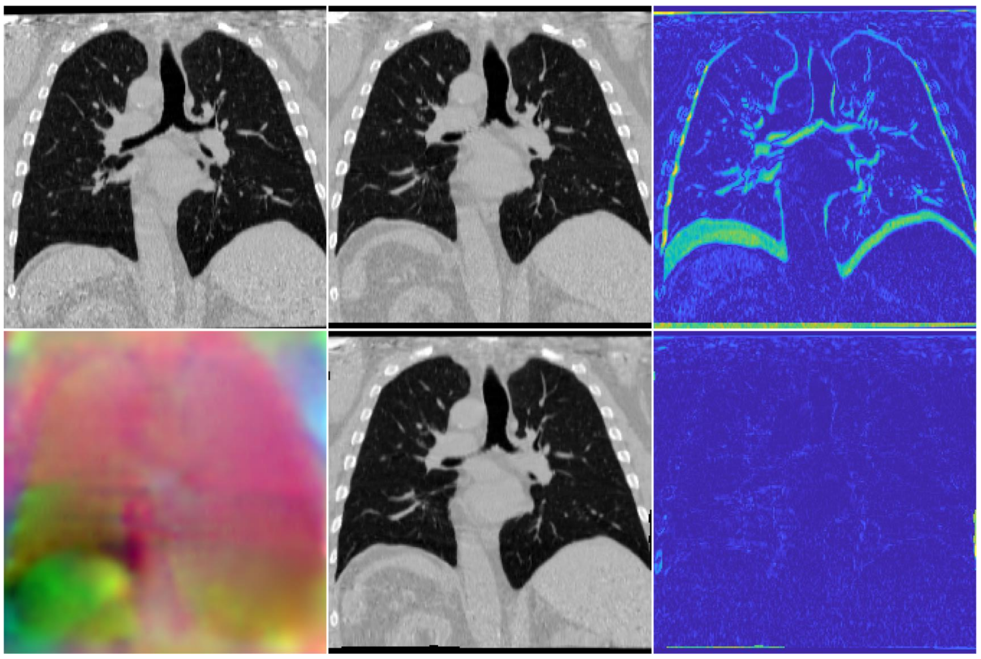


Figure 4.5: Visual results on the NLST dataset. Top: moving image, fixed image, and difference map between them. Bottom: displacement field, warped moving image, and difference map between warped and fixed image.

is reduced from 1.358 to 0.883 when using 2 cascades, it can be further reduced to 0.845 when using 3 cascades. On the other hand, increasing the number of cascades has a negative influence on the smoothness of the displacements (indicated by SDlogJ), which is predictable as more cascaded is desired to map large and irregular displacements by composing smaller and smoother displacements. A visual example is given in Fig. 4.5.

Testing Results In Table 4.5, we compare the performance of 4 different methods on the private testing set of NLST, including MEVIS [56], ConvexAdam [133], VORC [40, 155], and LapIRN [111]. The proposed Cascaded LKU-Net achieves the lowest TRE on key points (TRE(KP)) and the second lowest TRE on landmarks (TRE(LM)) while retaining a competitive inference speed (8.9 seconds per image pair). We note that this table is taken from the organizers of the MICCAI Learn2Reg 2022 challenge.

Table 4.4: Comparision between LKU-Net and Cascaded LKU-Net on the NLST validation set. A smaller TRE(KP) suggests a better registration.

Method	#Cascade	TRE(KP) ↓	SDlogJ
LKU-Net	–	1.358±0.319	0.058±0.008
Cascaded LKU-Net	2	0.883±0.144	0.062±0.006
Cascaded LKU-Net	3	0.845±0.137	0.064±0.007

Table 4.5: Comparsions on the private testing set of NLST.

Methods	TRE(KP)	TRE(LM)	SDlogJ	Time
Initial	9.76	10.21	–	–
MEVIS [56]	1.07	2.04	0.02	13.8
ConvexAdam [133]	1.05	2.23	0.04	–
VORC [40, 155]	0.84	1.76	0.04	24.7
LapIRN [111]	0.81	1.67	0.04	–
Cascaded LKU-Net	0.79	1.70	0.05	8.9

4.4 LKU-Net-Affine

Deep deformable registration is formulated as the minimization problem including a data term that measures the distance between the fixed and warped moving image pair and a regularization term that penalizes non-smooth deformations. Differently, deep rigid and affine registration methods eliminate the need for a regularization term and solely optimize the data term, as rigid and affine transformations inherently provide regularization.

4.4.1 Revisting Rigid and Affine Registration

In Section 2.1, 2D rigid and affine transformation and their matrices notation represented in homogeneous coordinates are discussed. 3D rigid and affine transformation, however, involves more parameters to cater to the transformations in the depth direction. Both 3D rigid and affine transformations can be represented by a 4×4 matrix $\boldsymbol{A}$ in homogeneous coordinates, utilizing 12 parameters.

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.6}$$

Affine registration is essentially the estimation of the 12 parameters in the affine matrix $\boldsymbol{A}$ in Eq. (4.6). However, according to a recent study by Chen et al. [24], it is challenging to jointly learn the 12 parameters from a given image pair. To address this issue, the authors adopted a strategy of learning transformation-specific parameters, which decomposes the original matrix $\boldsymbol{A}$ into six different transformation matrices, i.e., the three rotation matrices in Eqs. (4.7), (4.8), and (4.9), the translation matrix in Eq. (4.10), the scaling matrix in Eq. (4.11), and the sharing matrix in Eq. (4.12), resulting in predicting a total of **15** transformation-specific parameters. We note that this strategy of separately learning transformation-specific parameters is also used in [21].

To begin with, the rigid transformation comprises only rotation and translation transformation, thus the matrix $\boldsymbol{A}$ of rigid transformation can be represented by three different rotation matrices

$$\boldsymbol{A}_{rot-x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\theta_x) & -sin(\theta_x) & 0 \\ 0 & sin(\theta_x) & cos(\theta_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.7}$$

$$\boldsymbol{A}_{rot-y} = \begin{bmatrix} cos(\theta_y) & 0 & sin(\theta_y) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\theta_y) & 0 & cos(\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.8}$$

$$\boldsymbol{A}_{rot-z} = \begin{bmatrix} cos(\theta_z) & -sin(\theta_z) & 0 & 0 \\ sin(\theta_z) & cos(\theta_z) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.9}$$

and one translation matrix

$$\boldsymbol{A}_{trans} = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 1 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.10}$$

accounting for the rotation about the x-axis ($\boldsymbol{A}_{rot-x}$), rotation about the y-axis ($\boldsymbol{A}_{rot-y}$), rotation about the z-axis ($\boldsymbol{A}_{rot-z}$), and translation ($\boldsymbol{A}_{trans}$), in which θ_x, θ_y, and θ_z are the rotation angles and Δx, Δy, and Δz are the translation displacements, respectively.

Affine transformation, on the other hand, includes not only rotation and translation but also scaling transformation

$$\boldsymbol{A}_{scale} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

and shearing transformation

$$\boldsymbol{A}_{shear} = \begin{bmatrix} 1 & h_{xy} & h_{xz} & 0 \\ h_{yx} & 1 & h_{yz} & 0 \\ h_{zx} & h_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.12}$$

where S_x, S_y, and S_z denote the scaling factors on the x, y, and z directions respectively. h_{xy} and h_{xz} denote the shearing factors in x direction, h_{yx} and h_{yz} denote the shearing factors in y direction, and h_{zx} and h_{zy} denote the shearing factors in z direction.

4.4.2 Methodology

As aforementioned in Section 2.6, the classical CNN architectures like LeNet-5 [87] and AlexNet [86] can be used as the backbone to perform parametric registration, in which the convolutional layers are used to extract spatial information from images while the FC layers are used to integrate global transformation parameters.

We built LKU-Net-Affine on the basis of LKU-Net, specifically, we discarded the whole expansive path and plugged multiple additional fully connected modules on top of the contracting path to estimate the 15 affine parameters. As given in Fig. 4.6, each module contains two FC layers. The initial FC layer receives features from the contracting path and generates a high-dimensional compact representation. Therefore, we refer to this layer as the Global Feature Integration Layer. Subsequently, the second FC layer converts this representation into transformation-specific parameters. Therefore, we term the second FC layer the Transformation Estimation Layer.

Following [21], we set the number of neurons in the first Global Feature Integration Layer to 100. In the *translation* module, we have 3 neurons in the transformation estimation layer

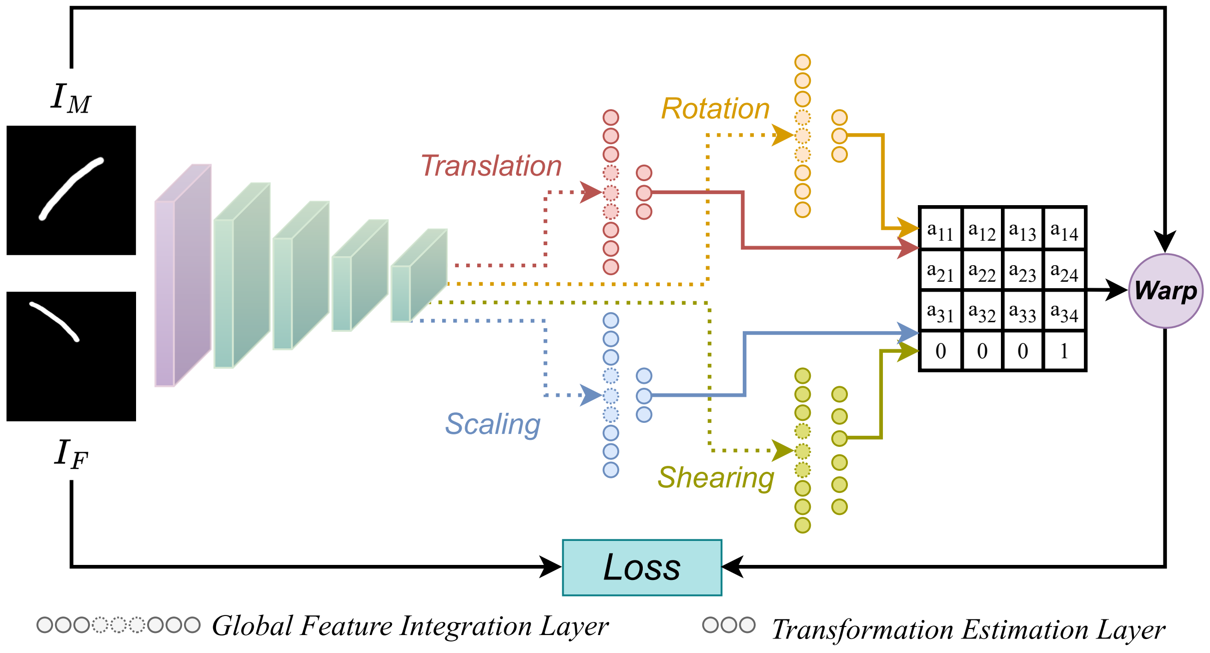


Figure 4.6: LKU-Net-Affine has four transformation-specific modules. Each module contains a *Global Feature Integration Layer* that fuses spatial information learned from convolutional layers and a *Transformation Estimation Layer* that estimates three transformation-specific parameters (six for shearing).

to estimate three translation displacements: Δx, Δy, and Δz. Similarly, there are 3 neurons to estimate the rotation angles θ_x, θ_y, and θ_z in *rotation* module; there are 3 neurons account for the scaling factors S_x, S_y, and S_z in *scaling* module; and there are 6 neurons account for the six shearing factors h_{xy}, h_{xz}, h_{yx}, h_{yz}, h_{zx}, and h_{zy} in *shearing* module.

4.4.3 Experiments

Datasets

Three expansion microscopy datasets from the ISBI 2023 RnR-ExM challenge are used, which are captured from three commonly imaged species including the mouse (cortex), ze-

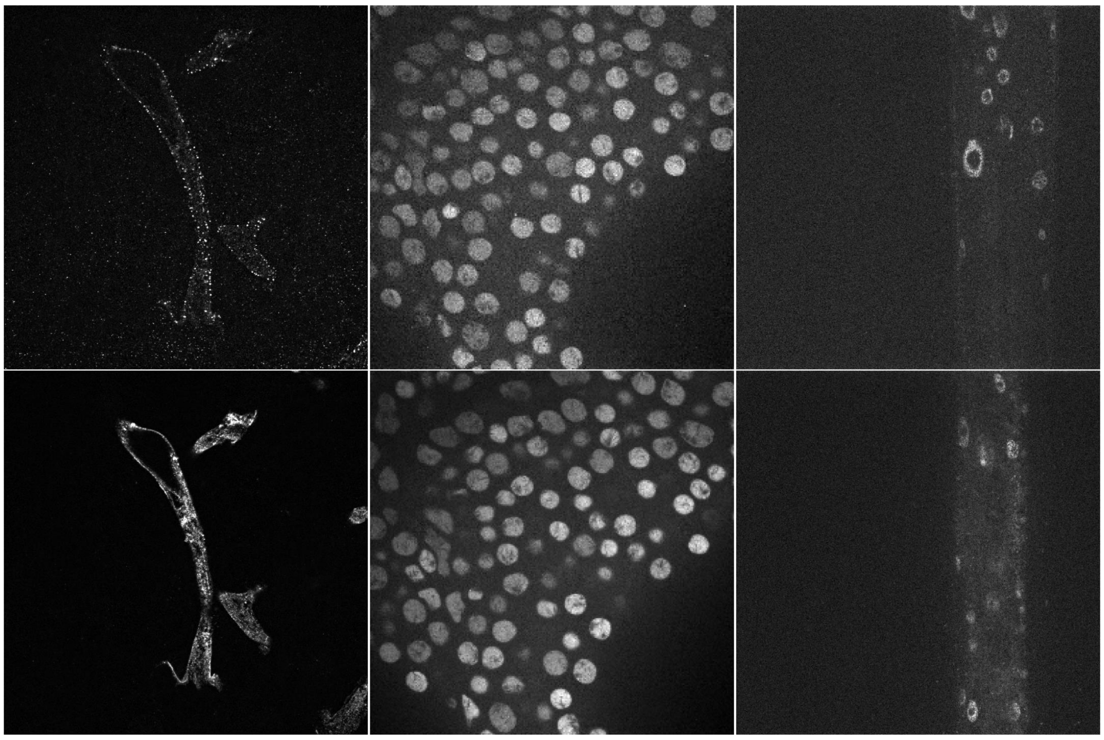


Figure 4.7: Example slices of moving (top) and fixed (bottom) volumes from the Mouse (left), Zebrafish (middle), and C. elegans (right).

.

brafish (brain), and Caenorhabditis elegans (C. elegans, whole), respectively. Each dataset includes four training volume pairs, one validation pair, and three testing pairs. Each volume has a large resolution of z×2048×2048, the voxel spacing is [0.1625 um, 0.1625um, 0.4 um] for both the fixed and moving volumes.

The limited number of training pairs poses the first challenge for training deep networks. Additionally, there are cases with extreme intensity differences in the Mouse dataset. The volumes in the Zebrafish dataset have a strong z-offset (133-195) and repetitive small-scale textures. The volumes in the C. elegans have also strong z-offsets (300-500+) and large volume sizes (559×2048×2048). Example slices of moving and fixed volumes from the three datasets are given in Fig. 4.7, note that they are down-sampled for illustration.

Data Pre-processing

For the Mouse dataset, all training and validation volume pairs are down-sampled into $81 \times 128 \times 128$ volumes. Subsequently, we applied a min-max intensity normalization on the down-sampled data, which ensured that the input data ranged between 0 and 1. For Zebrafish, we down-sampled all moving and fixed volumes into $195 \times 256 \times 256$ and $133 \times 256 \times 256$ resolutions, respectively. The min-max intensity normalization is adopted to reduce the influence of intensity variances. For C. elegans, we down-sampled all pairs into $z \times 256 \times 256$ volumes, and the z dimension was maintained the same as the original volume. Min-max intensity normalization is applied as well.

Training and Fine-Tuning

Given the considerable domain gaps that existed between the training and testing volumes such as the differences in imaging intensity and image resolutions. LKU-Net-Affine therefore has two significant steps for registration: the initial training step, followed by an instance-level optimization (fine-tuning) step.

Training: To mitigate the issue of limited training data, we incorporated an online 3D data augmentation approach to train our proposed LKU-Net-Affine model. This online augmentation involved randomly adding noise, cropping, affine, and deformable transformation. The augmented training data are then fed into the LKU-Net-Affine and used to estimate the affine matrix. The estimated affine matrix was subsequently converted into a dense displacement to warp the moving image. We employed a multi-scale locally normalized cross-correlation (LNCC) as the similarity metric for all three datasets. During the training process, we utilized the Adam optimizer with a fixed learning rate of 0.0001 for mouse data and a 0.001 learning rate for both the C. elegans and zebrafish datasets. We used a batch size of 1 and set the maximum number of training iterations to 4000.

Note that the online data augmentation and loss function are both employed on the normalized downsampled volumes when training the LKU-Net-Affine model.

Fine-tuning: Since significant domain gaps exist between training and testing image pairs. Once the LKU-Net-Affine model is trained, we perform instance-level optimization (fine-tuning) for each testing pair. The fine-tuning learning rates are set to be 0.0001, 0.001, and 0.001 for the Mouse, Zebrafish, and C. elegans datasets, respectively.

Comparison with State-of-the-Art

As listed in Table 4.6, the proposed LKU-Net-Affine ranks first place on the C. elegans, second on the Mouse, and third on the Zebrafish, proving its competitivity on the affine registration tasks as well. We note that Dice30 denotes the 30th percentile of the Dice values over all segmentation labels, which is used to assess algorithm robustness. The results are from the organizers of ISBI 2023 Robust Non-Rigid Registration Challenge for Expansion Microscopy (RnR-ExM), where the compared methods include BigStream [2], LapIRN [111], and the official baseline [3].

Although the proposed solution is straightforward, we observed a few limitations of our method during the challenge. Firstly, the scarcity of training samples (only 4 pairs) made it difficult to train the model, necessitating the use of heavy online data augmentation. Secondly, due to the limited computational resources available, we were only able to experiment with a maximum image resolution of $z \times 256 \times 256$, we believe that by increasing the input resolution, the performance of our LKU-Net-Affine model can be further enhanced.

[2]https://github.com/JaneliaSciComp/bigstream

[3]https://rnr-exm.grand-challenge.org/

Table 4.6: Top 3 performing methods on each testing set. Our LKU-Net-Affine ranks first place on the C. elegans, second on the Mouse, and third on the Zebrafish.

Datasets	Methods	Dice	Dice30	HDist95	SDlogJ
C. elegans	LKU-Net-Affine	0.873	0.874	3.478	0.0007
	LapIRN	0.891	0.882	3.168	0.1099
	BigStream	0.883	0.874	3.150	0.1638
Mouse	BigStream	0.892	0.838	10.742	<0.0001
	LKU-Net-Affine	0.883	0.858	13.151	0.0001
	Baseline	0.861	0.840	14.683	0.0006
Zebrafish	BigStream	0.972	0.972	2.206	<0.0001
	LapIRN	0.973	0.971	2.191	0.0003
	LKU-Net-Affine	0.971	0.970	2.303	0.0001

4.5 Summary

With two public 3D brain datasets, in Section 4.2, we have shown that the plain **U-Net is still a competitive backbone** for deep unsupervised deformable registration. Moreover, the proposed LKU-Net, built on the U-Net architecture, can outperform modern transformer-based methods for both inter-subject and atlas-to-subject registration tasks.

Based on LKU-Net, Cascaded LKU-Net is proposed in Section 4.3 to learn large displacements for more complex registration tasks. Evaluated on the NLST 3D CT lung registration task, the Cascaded LKU-Net has shown superiority to LKU-Net. In Section 4.4, LKU-Net-Affine is proposed to tackle rigid and affine parametric registration, it shows competitive accuracy on three challenging high-resolution expansion microscopy registration tasks from the ISBI 2023 RnR-ExM challenge.

Chapter Five

Model-Driven Variational Registration Network

This chapter is based on the following publications:

Learning a model-driven variational network for deformable image registration [75] which is published in IEEE Transactions on Medical Imaging.

5.1 Introduction

In mono-modal intensity-based registration, a variational framework is often used in which the problem is framed as an optimization of the form

$$\arg\min_{\boldsymbol{u}} \frac{1}{2} \int_{\Omega} |I_1(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) - I_0(\boldsymbol{x})|^2 \mathrm{d}\boldsymbol{x} + \lambda \mathcal{R}\left(\boldsymbol{u}(\boldsymbol{x})\right), \tag{5.1}$$

where I_0 and I_1: $\left(\Omega \subseteq \mathbb{R}^d\right) \rightarrow \mathbb{R}$ represent the fixed image and the moving image, respectively. $\boldsymbol{u}(\boldsymbol{x}) : \Omega \rightarrow \mathbb{R}^d$ denotes the displacement vector field. In this chapter, we study $d = 2, \boldsymbol{u}(\boldsymbol{x}) = (u_x(\boldsymbol{x}), u_y(\boldsymbol{x}))^{\mathrm{T}}$ and $d = 3, \boldsymbol{u}(\boldsymbol{x}) = (u_x(\boldsymbol{x}), u_y(\boldsymbol{x}), u_z(\boldsymbol{x}))^{\mathrm{T}}$ which correspond to 2D and 3D cases. The first term (i.e., data term) is the sum of squared differences. Minimization of the data term alone is typically an ill-posed problem with many possible (ambiguous) solutions. Hence, the second regularization term is needed, which is normally chosen to control the smoothness of the deformation.

The variational model is among the most successful and accurate approaches to calculating a deformation between two images [159]. Given a specific regularization term, such a model has a clear mathematical structure and it is also well understood which mathematical space the solution lies in (Hilbert space [13, 20, 46], bounded variation [48, 146], etc.). However, the variational model has limitations: (1) For each image pair, the hyper-parameter λ may need to be tuned carefully to deliver a precise deformation. While too small a λ leads to an irregular, non-smooth deformation, setting it too high reduces the deformation magnitude and therefore loses the ability to model large deformations. (2) The second regularization term is typically manually designed based on assumptions about the deformation. However, existing assumptions may be too simple to capture complex changes in image content associated with biological tissues. (3) The variational model is nonlinear and therefore needs to be optimized iteratively, which is computation-intensive and time-consuming, especially for high-dimensional data inputs.

Many deep learning approaches have been proposed for unsupervised deformable medical image registration [11, 62, 85, 119, 161, 166]. In order to learn a deformation, almost all of these learning-based approaches follow the formulation of $\boldsymbol{u} = f\left(I_0, I_1|\mathbf{W}\right)$, where f is a convolutional neural network (CNN) and $\mathbf{W}$ denotes the weights of the CNN. These approaches are purely data-driven and differ from iterative variational approaches in two main aspects. (1) Data-driven approaches take images as input and directly output the estimated deformations under a loss criterion, while iterative approaches take an initial deformation as input, and output a final refined deformation which is built upon the previous deformations in the iterative optimization. Whilst data-driven approaches often require substantial quantities of training data to reach an adequate level of performance, the iterative methods can work well in low data regimes. Additionally, the heavy data dependence of deep learning can result in a network that overfits the training data and therefore lacks generalization abilities. (2) Classical iterative methods explicitly use prior and domain knowledge to construct a mathematical formulation. In contrast, data-driven methods implicitly learn domain knowledge through the optimization of respective loss functions rather than explicitly building this knowledge into the network architecture itself.

In order to take advantage of both methods, in this chapter, we unify a data-driven and an iterative approach into one framework and propose a model-driven variational registration network, which we term VR-Net (Fig. 5.1). By unifying these two approaches, the VR-Net can leverage information from the entire training set to help registration in every single case, thereby having the potential to outperform iterative approaches.

Specifically, with the help of a *variable splitting* scheme for optimization, we decompose the original iterative variational problem into two sub-problems. One has a point-wise, closed-form solution and the other can be formulated as a general denoising problem. Next, we formulate the point-wise, closed-form solution with a warping layer and an intensity consistency layer. We then propose a residual U-Net for the denoising sub-problem which

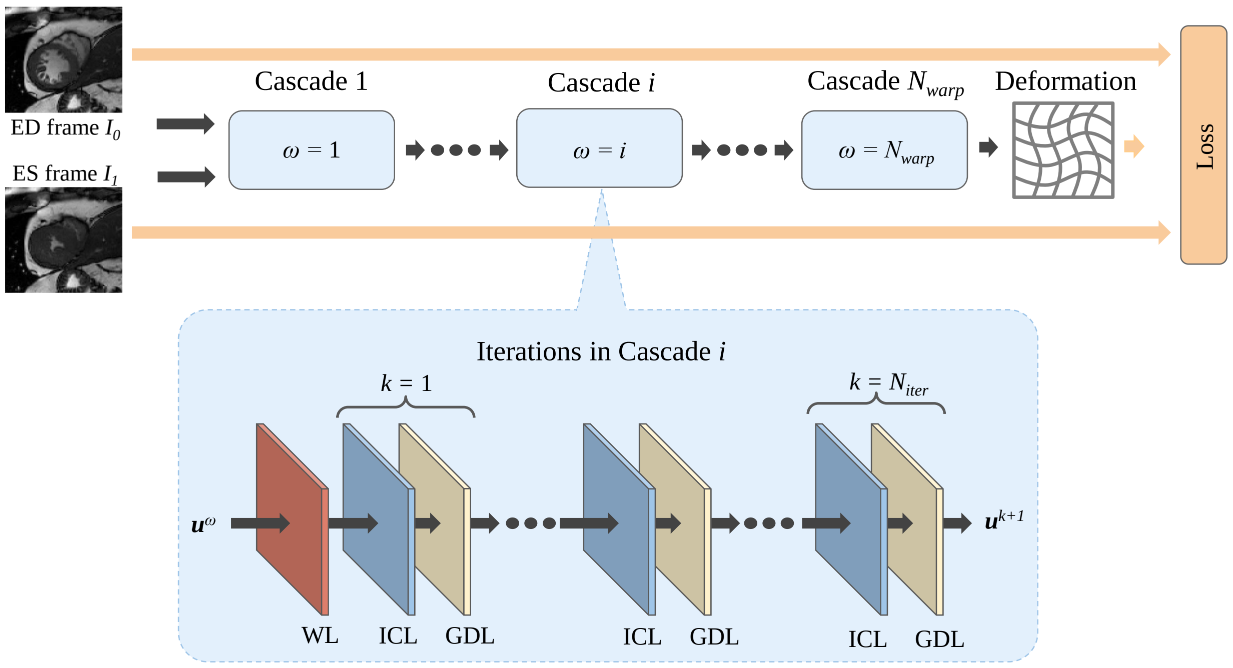


Figure 5.1: The architecture of VR-Net. WL, ICL, and GDL denote the warping layer, intensity consistency layer, and generalized denoising layer, respectively. These layers (detailed in Fig. 5.2) are designed as per the minimization of a generic variational model for image registration. The cascade number is controlled by $N_{warp}\times N_{iter}$, which mimics the iterative process of minimization.

can be regarded as a learnable regularization term embedded in the VR-Net, replacing the hand-crafted hyper-parameter seen in iterative variational methods. Finally, within the VR-Net we cascade the warping layer, the intensity consistency layer, and the generalized denoising layer to mimic the iterative process of solving a variational model.

To evaluate the proposed VR-Net, we use two 2D publicly available cardiac MRI datasets, i.e., the UK Biobank dataset [117], Automatic Cardiac Diagnosis Challenge (ACDC) dataset [14], and the local 3DCMR dataset used in Section 3.3.4. Extensive experiments on the three datasets show that our VR-Net outperforms model-based approaches in terms of registration accuracy while retaining the fast inference speed of deep data-driven approaches. However, we note that our VR-Net is based on an intensity consistency assumption which may not work for medical images with large contrast variances or from different modalities. Collectively, our main contributions are:

- VR-Net embeds the mathematical structure from the minimization of a generic variational model into a neural network. The network mapping function $\boldsymbol{f}$ therefore inherits prior knowledge from the variational model, whilst maintaining the data efficiency of iterative methods and retaining the fast inference speed of deep data-driven methods. As such, it has the advantages of both communities and is shown to exceed state-of-the-art registration methods in terms of Dice and HDist on both 2D and 3D datasets.
- For iterative variational approaches to perform well for individual image pairs, one often needs to define a suitable regularization and then tune the corresponding regularization parameter λ in Eq. (5.1). Instead, our model-driven VR-Net is trained with a global regularization with parameter α in Eq. (5.8) for the whole dataset, while making the original regularization term and λ in Eq. (5.1) a learnable denoising layer within the network. This enables the model to effectively learn the regularization term and the value of λ from data, which avoids the tuning on individual image pairs and thus results in a more generalizable model compared with its iterative counterparts.

5.2 Related Works

Iterative Approaches: There is a wide range of classical variational methods to account for local deformations such as diffusion models [46], total variation models [48, 146], fluid models [13, 20], elastic models [16, 92, 144], biharmonic (linear curvature) models [45, 108], mean curvature models [28, 29], optical flow models [17, 153, 159], fractional-order variation models [156, 160], non-local graph models [115, 123, 154].

Data-Driven Approaches: Learning-based methods are normally trained with a large amount of paired images. U-Net and Dual style networks [10, 11, 119] adopt one single network to directly estimate the displacement from input images. Recent works have adopted the coarse-to-fine scheme to progressively estimate the final deformation using multiple cascades. De Vos et al. [148] proposed a multi-stage, coarse-to-fine network, which has two types of CNNs that account for global and local transformations, respectively. Guo et al. [52] also proposed a coarse-to-fine, multi-stage registration framework. However, their method estimates only rigid transformations while our method predicts dense displacements for nonrigid registration. Zhao et al. [166] proposed RC-Net, a recursive cascade architecture, for dense registration. By adopting multiple cascades, RC-Net achieved significant performance gains over VoxelMorph [11] on both liver and brain registration tasks. Similar to RC-Net, the proposed VR-Net also uses a cascaded, end-to-end trainable network architecture. Within each cascade of VR-Net, however, we solve a point-wise, closed-form optimization problem induced by minimizing a generic variational model, which is a major difference from other recursive [166] or multi-stage [52, 148] networks.

Model-Driven Approaches: The authors in [25, 57, 84] studied trainable variational networks to address supervised, linear image restoration and reconstruction problems, while we tackle an unsupervised, nonlinear image registration problem. Their methods are based on proximal gradient descent, and they learn the regularization term based on the Field of Experts (FoE) [126]. The nonlinearity (a derivative of the potential function in the regularization) in their method is imposed by the radial basis kernels. The optimization of their method is done through the specialized inertial incremental proximal method (IIPG), which is not implemented in a standard deep learning framework (e.g. PyTorch) and therefore may be difficult to generalize to other problems. In contrast, our network is based on a linearized variable splitting method, one advantage of which is that we can impose an exact data term (due to its closed-form solution) in each cascade which cannot be done by gradient-based methods. Our regularization is formulated as CNNs, where the nonlinearity is imposed by the activation functions (such as ReLU), and the parameters are optimized by Adam. There also exist works [1, 38, 44, 131, 162] that have explored variational formulations in the deep learning framework. However, instead of image registration, they were used either for image restoration and reconstruction or for video understanding. Recently, Blendowski et al. [15] proposed a supervised iterative descent algorithm (SUITS) for multi-modal image registration, which has similar ingredients to our method. SUITS uses a CNN to extract image features, which are then plugged into the Horn and Schunck (HS) model [68] to compute displacements. In other words, they need to solve an iterative model within the network each time when new displacements are required. This method can be expensive because (1) the HS model needs to have many data terms (12 in their paper) in order to align all extracted features; (2) solving the HS model itself is costly and requires iterations, and (3) they need to solve the HS model many times within the network. In contrast, we do not need to iteratively solve any optimization model within our network. Instead, we use the iterative process for optimization only to guide the design of network architecture. Moreover, unlike [15] which uses an algebraic multigrid solver (AMG) to solve the linear system of equations, all subproblems (network layers) in our method have closed-form, point-wise solutions. ConvexAdam [133] is another model-driven approach for deformable registration, it contains a feature extractor that maps the input images into a feature space specified by MIND [60] or nnU-Net, a dense correlation layer followed by a coupled convex optimization which provides an initial displacement, and finally, an Adam optimizer that performs instance-level refinement of the initial displacement. ConvexAdam has achieved competitive registration accuracy on multiple mono- and multi-modality tasks, but it requires sophisticated hyperparameter tuning.

5.3 Generic Variational Method

In this section, we study a more general variational model [68, 159] for image registration, which is given by

$$\arg\min_{\boldsymbol{u}} \frac{1}{s} \int_{\Omega} |I_1(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x})) - I_0(\boldsymbol{x})|^s \mathrm{d}\boldsymbol{x} + \lambda \mathcal{R}\left(\boldsymbol{u}(\boldsymbol{x})\right), \tag{5.2}$$

where the variables in this formulation have the same meaning as in Eq. (5.1). The objective is to find the optimal deformation $\boldsymbol{u}^*(\boldsymbol{x}) : (\Omega \subseteq \mathbb{R}^d) \to \mathbb{R}^d$, that minimizes the formulation. Within the data term, $s = 1$ corresponds to L_1 estimation that is robust to outliers, while $s = 2$ gives the estimation based on the sum of squared differences. The second term is a generic regularization term, which imposes a smoothness constraint on the deformation. The hyper-parameter λ controls the smoothness of the solution. However, it is non-trivial to select both the regularization term and λ optimally.

In the data term, we notice that the non-linearity in the function $I_1(\boldsymbol{x}+\boldsymbol{u})$ with respect to $\boldsymbol{u}$ poses a challenge to optimize Eq. (5.2). To benefit from closed-form solutions, we use the Gauss–Newton algorithm [9, 159] to handle Eq. (5.2). By employing the first-order Taylor expansion at $\boldsymbol{u}^\omega$, we end up solving the following alternative problem,

$$I_1\left(\boldsymbol{x}+\boldsymbol{u}\right) \approx I_1\left(\boldsymbol{x}+\boldsymbol{u}^\omega\right) + \langle \nabla I_1(\boldsymbol{x}+\boldsymbol{u}^\omega), \boldsymbol{u}-\boldsymbol{u}^\omega \rangle \tag{5.3a}$$

$$\boldsymbol{u}^{\omega+1} = \arg\min_{\boldsymbol{u}} \frac{1}{s} \int_{\Omega} |\rho(\boldsymbol{u})|^s d\boldsymbol{x} + \lambda \mathcal{R}\left(\boldsymbol{u}\right), \tag{5.3b}$$

where

$$\rho(\boldsymbol{u}) = I_1\left(\boldsymbol{x}+\boldsymbol{u}^\omega\right) + \langle \nabla I_1(\boldsymbol{x}+\boldsymbol{u}^\omega), \boldsymbol{u}-\boldsymbol{u}^\omega \rangle - I_0(\boldsymbol{x}). \tag{5.4}$$

In Eq. (5.3a), ∇ is the gradient operator, ∇I_1 represents partial derivatives of I_1, $\langle \cdot, \cdot \rangle$ denotes the inner product and ω denotes the ω^{th} iteration. The linearized version of Eq. (5.2), seen in Eq. (5.3b), must be solved iteratively. As the data term in Eq. (5.3b) is in a linear, convex form, one can derive a closed-form solution. Of note, to solve Eq. (5.2) approximately, one needs to iterate between Eq. (5.3a) and Eq. (5.3b), meaning that there exist two loops in the resulting numerical implementation.

The regularization $\mathcal{R}(\boldsymbol{u})$ has many choices depending on what the final deformation $\boldsymbol{u}^*$ looks like, such as piecewise smooth and piecewise constant. A widely used choice is the Total Variation (TV) [97, 146, 159], which is a powerful regularization that allows discontinuities in the resulting deformation. However, a major issue for hand-crafted regularization is that it may not be optimal for more complex, task-specific applications. To circumvent these, we propose an end-to-end trainable VR-Net detailed in Section 5.4.1.

5.3.1 Variable Splitting

To design an appropriate VR-Net, we first adopt a variable splitting scheme [38, 97, 159] to minimize the linearized variational model Eq. (5.3b). Specifically, we introduce an auxiliary splitting variable $\boldsymbol{v} : (\Omega \subseteq \mathbb{R}^d) \to \mathbb{R}^d$, converting Eq. (5.3b) into the equivalent constrained minimization problem

$$\arg\min_{\boldsymbol{u},\boldsymbol{v}} \frac{1}{s} \int_{\Omega} |\rho(\boldsymbol{u})|^s d\boldsymbol{x} + \lambda \mathcal{R}(\boldsymbol{v}) \quad s.t. \quad \boldsymbol{u} = \boldsymbol{v}.$$

The introduction of the constraint $\boldsymbol{u} = \boldsymbol{v}$ above decouples $\boldsymbol{u}$ in the regularization term from the data term, therefore a multi-channel denoising problem can be explicitly constructed and a closed-form, point-wise solution can be derived. Using the penalty function method, we then add the constraint back into the model and minimize the single problem

$$\arg\min_{\boldsymbol{u},\boldsymbol{v}} \frac{1}{s} \int_{\Omega} |\rho(\boldsymbol{u})|^s d\boldsymbol{x} + \lambda \mathcal{R}(\boldsymbol{v}) + \frac{\theta}{2} \int_{\Omega} |\boldsymbol{v} - \boldsymbol{u}|^2 d\boldsymbol{x},$$

where θ is the introduced penalty weight. To solve the multi-variable minimization problem, one needs to minimize it with respect to $\boldsymbol{u}$ and $\boldsymbol{v}$ separately.

1) ***u*-subproblem** is a linear problem and handled by considering the following minimiza-

tion problem

$$\boldsymbol{u}^{k+1} = \operatorname*{argmin}_{\boldsymbol{u}} \frac{1}{s} \int_\Omega |\rho(\boldsymbol{u})|^s d\boldsymbol{x} + \frac{\theta}{2} \int_\Omega |\boldsymbol{v}^k - \boldsymbol{u}|^2 d\boldsymbol{x},$$

the solution of which depends on the order of s. In the case of $s = 1$, the solution is given by the following thresholding equation

$$\boldsymbol{u}^{k+1} = \boldsymbol{v}^k - \frac{\hat{z}}{\max\left(|\hat{z}|, 1\right)} \frac{\nabla I_1}{\theta}, \tag{5.5}$$

where $\hat{z} = \theta\rho(\boldsymbol{v}^k)/(|\nabla I_1|^2 + \epsilon)$ and ϵ is a small positive value added to avoid division by zero to prevent vanishing gradients in the image. In Appendix 1 we develop a novel primal-dual method to derive this solution (5.5). Our new derivation allows the proposed method to easily adapt to vector images which usually appear in data terms that use image patch or (higher-order) gradient information [114, 147].

In the case of $s = 2$, the respective problem is differentiable and we can derive the following Sherman–Morrison formula [12, 143] by differentiating this subproblem with respect to $\boldsymbol{u}$

$$(\mathbf{J}\mathbf{J}^{\mathrm{T}} + \theta\mathbb{1})(\boldsymbol{u} - \boldsymbol{u}^{\omega}) = \theta(\boldsymbol{v}^k - \boldsymbol{u}^{\omega}) - \mathbf{J}(I_1 - I_0), \tag{5.6}$$

where $\mathbf{J}\mathbf{J}^{\mathrm{T}}$ (where $\mathbf{J} = \nabla I_1$) is the rank-1 outer product and $\mathbb{1}$ is an identity matrix. Due to the identity matrix, the Sherman–Morrison formula will lead to a close-form, point-wise solution to $\boldsymbol{u}^{k+1}$. In Appendix 2, we present detailed derivations in both 2D and 3D.

2) ***v*-subproblem** is handled by considering the following minimization problem

$$\boldsymbol{v}^{k+1} = \operatorname*{argmin}_{\boldsymbol{v}} \lambda\mathcal{R}(\boldsymbol{v}) + \frac{\theta}{2} \int_\Omega |\boldsymbol{v} - \boldsymbol{u}^{k+1}|^2 d\boldsymbol{x}. \tag{5.7}$$

Given a known $\boldsymbol{u}^{k+1}$, this problem essentially is a denoising problem with the generic regularization $\mathcal{R}(\boldsymbol{v})$. Note that we assume the noise here is additive and follows a Gaussian distribution. On the other hand, if the regularization $\mathcal{R}(\boldsymbol{v})$ is TV, then it is a TV denoising problem, as in Zach's paper [159].

Putting these derivations together, we have Algorithm 3 to minimize Eq. (5.2) using variable splitting. Since Taylor expansion is used to linearize the non-linear function, Eq. (5.3a)

Algorithm 3 VS for generic variational registration model

```
1: Inputs : $I_0$, $I_1$ and $(\theta, \lambda, N_{warp}, N_{iter})$.
2: Initialize : $\boldsymbol{u}^1$ and $\boldsymbol{v}^1$.
3: for $\omega = 1 : N_{warp}$ do                                ▹ # Taylor expansions
4:     $I_1^{\omega} = warping(I_1, \boldsymbol{u}^{\omega})$
5:     while $k < N_{iter}$ do                                     ▹ # iterations
6:         update $\boldsymbol{u}^{k+1}$ via (5.5), (A.8) or (A.9) with $I_1 = I_1^{\omega}$
7:         $\boldsymbol{v}^{k+1} = denoiser(\boldsymbol{u}^{k+1})$
8:     end while
9:     $\boldsymbol{u}^{\omega} = \boldsymbol{u}^{k+1}$
10: end for
11: return $\boldsymbol{u}^* = \boldsymbol{u}^{\omega}$                          ▹ # return final solution
```

holds only if the resulting deformation $\boldsymbol{u}^*$ is small. As such, we adopt an extra *warping* operation (via $\boldsymbol{u}^{\omega}$) in Algorithm 3, i.e., $I_1^{\omega} = I_1(\boldsymbol{x} + \boldsymbol{u}^{\omega})$. With *warping*, we can break down a large deformation into N_{warp} small ones, each of which can be solved iteratively and optimally. The total iterations for the algorithm is $N_{warp} \times N_{iter}$.

5.4 Learning a Variational Registration Network

So far, we have shown how the variable splitting scheme can be derived to tackle the generic variational registration model. We first handle the original problem Eq. (5.2) with the Gauss-Newton method. For the resulting linearized minimization problem Eq. (5.3b) we have two sub-problems, one with a closed-form, point-wise solution for either choice of the s and the other one being a denoising problem with $\mathcal{R}(\boldsymbol{u})$. As of yet, we have not defined the exact form of *denoiser* in Algorithm 3. In this section, we detail the full VR-Net architecture and show how a residual CNN is used as our *denoiser* to solve the second denoising sub-problem.

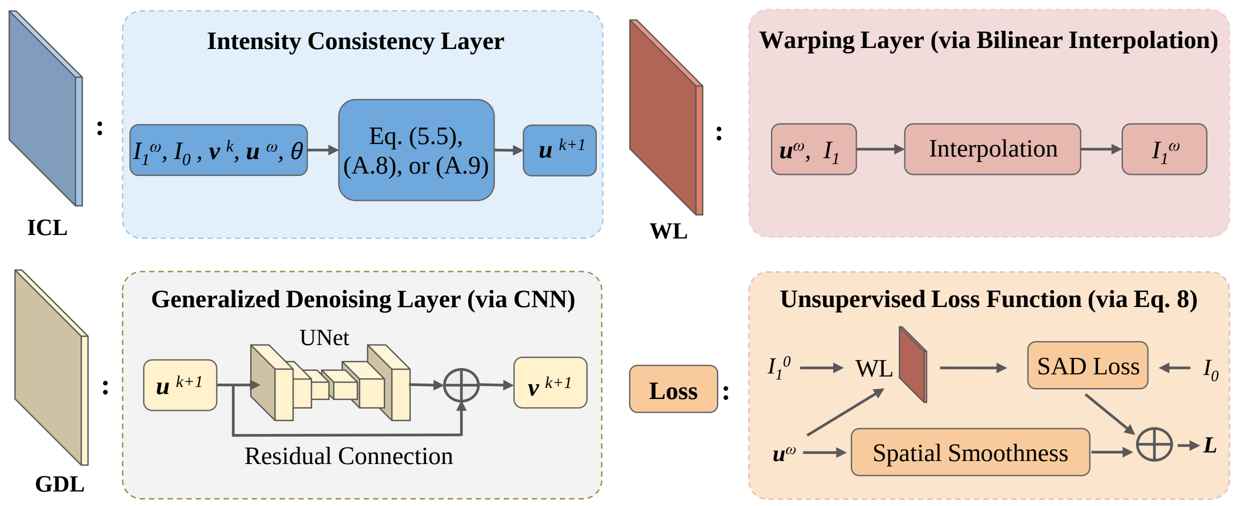

Intensity Consistency Layer
$I_1^\omega, I_0, v^k, u^\omega, \theta$
Eq. (5.5), (A.8), or (A.9)
u^{k+1}
ICL
Warping Layer (via Bilinear Interpolation)
u^ω, I_1
Interpolation
I_1^ω
WL
Generalized Denoising Layer (via CNN)
UNet
u^{k+1}
v^{k+1}
Residual Connection
GDL
Loss
Unsupervised Loss Function (via Eq. 8)
I_1^0
WL
SAD Loss
I_0
u^ω
Spatial Smoothness
L

Figure 5.2: Computational graph of each layer in VR-Net. ICL and GDL are designed based on solutions of sub-problems resulting from applying variable splitting to the original image registration model (5.2).

5.4.1 Network Architecture

We construct the proposed VR-Net by unrolling the iterative procedure in Algorithm 3. Fig. 5.1 depicts the resulting network architecture. There are two types of cascade in the architecture to learn a large displacement: (1) *cascade-iter* indicated by $k \in \{1, ..., N_{iter}\}$, stands for the inner loop in Algorithm 3; (2) *cascade-warp* indicated by $\omega \in \{1, ..., N_{warp}\}$, corresponds to the outer loop in Algorithm 3. Note that *cascade-warp* contains multiple nested *cascade-iter*s. In Fig. 5.2, we show the computational layers contained in the network, such as the *warping layer* (WL), the *intensity consistency layer* (ICL), and the *generalized denoising layer* (GDL). They respectively correspond to Steps 4, 6, and 7 in Algorithm 3.

Warping layer is achieved by using a bilinear interpolation for 2D images, following the spatial transformer networks [72]. Recall that the ***warping*** operation is defined in Algorithm 3 by $I_1^\omega = I_1(\boldsymbol{x} + \boldsymbol{u}^\omega)$, where $\boldsymbol{u}^\omega$ is the estimated displacement. The bilinear interpolation is continuous and piecewise smooth, and the partial gradients with respect to $\boldsymbol{u}^\omega$ can be derived as in [72]. The 2D warping layer can be easily extended to transform 3D volumes, as in [10]. In Fig. 5.2, we show the computational graph of this layer, which takes $\boldsymbol{u}^{\omega}$ and I_1 as the inputs and outputs the warped image $\boldsymbol{I}_1^{\omega}$.

Intensity consistency layer is crucial as it effectively imposes intensity consistency between the warped image ($\boldsymbol{I}_1^{w}$) and the target image ($\boldsymbol{I}_0$) such that the data term in Eq. (5.2) can be minimized. Fig. 5.2 presents the computational graph of this layer. Specifically, the input $\boldsymbol{I}_1^{w}$ from the upstream warping layer, concurrently with I_0, $\boldsymbol{v}^{k}$, $\boldsymbol{u}^{\omega}$ and θ, are passed through Eq. (5.5), (A.8) or (A.9) to produce $\boldsymbol{u}^{k+1}$, which then feeds the downstream generalized denoising layer. Note that the calculations in this layer are both computationally efficient and numerically accurate thanks to the existence of point-wise, analytical solutions from Eq. (5.5), (A.8) or (A.9). The penalty weight θ is often manually selected in iterative methods, however, in VR-Net we instead make it a learnable parameter.

Generalized denoising layer is a residual U-Net that explicitly defines *denoiser* in Algorithm 3. As illustrated in Fig. 5.2, we intend to denoise a two-channel displacement $\boldsymbol{u}^{k+1}$ with the residual U-Net and produce its denoised version $\boldsymbol{v}^{k+1}$ for ICL in next iteration. Since the input and output of ICL and GDL are both deformations, it is natural that we can adopt a residual connection between two adjacent cascades. As the generalized denoising layer represents the denoising subproblem Eq. (5.7), it implicitly absorbs the hyper-parameters λ and θ and thus there is no need to tune them manually. Note that while we use a residual U-Net as the backbone here, our setup is generic and therefore allows for the incorporation of more advanced denoising CNN architectures.

The function in Eq. (5.5) needs special attention when implemented as a neural layer. Although it is a continuous and piecewise smooth function, it is non-differentiable. As such, the concept of sub-gradients must be used during network back-propagation. As a result, this gives us a sub-differentiable mechanism with respect to network parameters, which allows loss gradients to flow back not only to the GDL and WL but also to the ICL.

5.4.2 Network Loss and Parameterizations

Network loss: While the design of VR-Net architecture follows the philosophy of conventional optimization for iterative methods, training the network parameters is another optimization process, for which a loss function must be explicitly formulated. Due to the absence of ground truth transformations in medical imaging, we adopt an unsupervised loss function, using the moving image I_1, the fixed image I_0, and the predicted deformation $\boldsymbol{u}$. The loss is

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N}\|I_1^i(\boldsymbol{x}+\boldsymbol{u}_i(\boldsymbol{\Theta})) - I_0^i(\boldsymbol{x})\|_1 + \frac{\alpha}{N}\sum_{i=1}^{N}\|\nabla\boldsymbol{u}_i(\boldsymbol{\Theta})\|_2^2, \tag{5.8}$$

where N is the number of training image pairs, $\boldsymbol{\Theta}$ are the network parameters to be learned and α is a hyper-parameter balancing the two losses. Note that the first term defines the sum of absolute differences (SAD) between the warped images and the fixed images and the second term defines the smoothness of the resulting displacements. The graph representation of the two loss functions is detailed in Fig. 5.2. We note that the SAD loss and diffusion regularizer are used for 2D experiments, while for the 3DCMR dataset, the data term and regularizer are selected as NMI and bending energy following Chapter 3.

Despite the model-driven components of our VR-Net, the method is essentially a deep learning approach so it also requires a smoothness parameter α that regularizes the learned displacements for the whole dataset. In contrast to the manual tuning of θ in Eqs. (5.5) and (5.6) which may be required for each test pair image in traditional iterative methods, this smoothness parameter α is only tuned for the whole training set and used for inference without further optimization. As shown in Fig. 5.1, we evaluate the loss function Eq. (5.8) at the very end of the VR-Net.

Parameterizations: The network learnable parameters $\mathbf{\Theta}$ include both the residual U-Net parameters $\mathbf{W}$ in GDL layers and the penalty weights θ in the ICLs. Recall that in VR-Net (see Fig. 5.1) we have *cascade-iter* and *cascade-warp*, and therefore each GDL and ICL layer has a set of parameters $\mathbf{W}$ and θ, respectively. We experimented with two parameterization settings: $\mathbf{\Theta^1} = \{\mathbf{W}, \theta\}$ and $\mathbf{\Theta^2} = \{\{\mathbf{W}_{k,\omega}, \theta_{k,\omega}\}_{k=1}^{N_{iter}}\}_{\omega=1}^{N_{warp}}$. For $\mathbf{\Theta^1}$, we let the parameters $\mathbf{W}$ and θ respectively be shared by the DLs and the ICLs across *cascade-warp* and *cascade-iter*. In $\mathbf{\Theta^2}$, the parameters are not shared in either *cascade-iter* or *cascade-warp*, meaning that each layer (GDL or ICL) has its own learnable parameter. For both parameterizations we experimented with, backpropagation is employed to minimize the loss with respect to the network parameters $\mathbf{\Theta}$ in an end-to-end fashion.

5.4.3 Initialization

While data-driven methods take image pairs as input and directly output the estimated deformations, we need an initial displacement as input of VR-Net as stated in Step 2 of Algorithm 3. The initial displacement is then refined by the iterative process. We proposed 3 different initialization strategies. The first strategy is to initialize the $\boldsymbol{u}^1$ and $\boldsymbol{v}^1$ with zeros, which is used in the original TV-L$_1$ paper [159]. The second strategy is using the Gaussian noise as initialization. However, initializing $\boldsymbol{u}^1$ and $\boldsymbol{v}^1$ with zeros or noise is not necessarily the optimal choice. Inspired by [44], we propose to learn the initialization from the data by concatenating a U-Net prior to the first WL. Note that the additional concatenated U-Net is not pre-trained. It is a part of the VR-Net and its weights are updated along with the whole VR-Net during the training process.

We evaluate the three different initialization strategies in Section 5.5.4 and show that the registration performance benefits from making the initialization learnable.

5.5 Experimental results

In this section, we introduce the datasets and quantitative metrics used for experiments. Then we describe the implementation details of the proposed method as well as parameter studies using different configurations. Finally, we compare the proposed VR-Net with state-of-the-art methods, including both iterative methods and data-driven approaches.

5.5.1 Datasets

2D Datasets: We evaluate the proposed VR-Net on the **UK Biobank** dataset [117] and the **ACDC** dataset [14]. The UK Biobank [117] is a large-scale cardiac MRI image dataset designed for cohort studies on 100,000 subjects. MRI scans in this dataset were acquired from healthy volunteers using the same equipment and protocols, and the in-plane and through-plane resolutions are 1.8mm and 10mm, respectively. We randomly selected 220 subjects and split them into 100, 20, and 100 for training, validation, and testing, respectively. The ACDC dataset [14] was created from real clinical exams. Acquisitions were obtained over a 6-year period with two MRI scanners of different magnetic strengths. The dataset is composed of 150 patients evenly divided into 5 types of pathology. We select the 100 subjects that have ground truth segmentation masks for experiments. We split these subjects into 40, 10, and 50 for training, validation, and testing, respectively. Since the in-plane resolution varies from 1.34 to 1.68mm, we resample all the images to 1.8mm before experiments. For both datasets, we perform experiments on only basal, mid-ventricular, and apical image slices.

3D Dataset: The 3DCMR dataset [37] used in Section 3.3.4 is employed. To train comparative deep learning methods and tune hyperparameters in different methods, the dataset is split into 100/20/100 corresponding to training, validation, and test sets. We report final quantitative results on the test set only.

For the three datasets, similar to Section 3.3.4, we evaluate the registration performance by comparing the warped ES segmentation with the ground truth ED segmentation. We compute the Dice and HDist for the LV, Myo, and RV structures.

5.5.2 Implementation Details

We implement the proposed 2D VR-Net with U-Net [125] as the backbone for all generalized denoising layers. We used the original U-Net architecture in [125] and no further optimization of the architecture was performed. As the input and output of such layers are displacements, we also apply a residual connection to the U-Net. To numerically discretize the partial derivatives $\nabla \boldsymbol{I}_1$ in Eq. (5.5) and Eq. (5.6) and $\nabla \boldsymbol{u}$ in the loss Eq. (5.8), the central finite difference method is adopted. To train the 2D VR-Net, the batch size is set to 10 pairs of images. $\boldsymbol{\alpha}$ in Eq. (5.8) is selected using the grid-search strategy on the validation set and is set to 0.1 for UK Biobank and 0.05 for ACDC. For training, we use the basal, mid-ventricular, and apical image slices in all frames from all subjects in the training set. During inference, we evaluate the 2D VR-Net and other comparative approaches using the three slices at the ED and ES phases from all subjects in the test set.

Extending the intensity consistency layer of 2D VR-Net to 3D is straightforward. The major difference between the 2D and 3D VR-Net is the generalized denoising layer, we adopt the U-Net used in VoxelMorph [11] as the backbone. The training batch size of 3D VR-Net is set to 1.

Both 2D and 3D VR-Nets are implemented with Pytorch [116]. An Adam optimizer [82] with two beta values of 0.9 and 0.999 is used and the initial learning rate is set to be 0.0001. Note that we train our VR-Net using each dataset separately. For UK Biobank, the maximum iterations are 50,000 and the learning rate is gradually reduced after 25,000 iterations. For

Table 5.1: Comparison of registration performance on UKBB and ACDC using different configurations for the proposed VR-Net. Dice and HDist are computed by averaging that of LV, Myo and RV at the basal, mid-ventricular, and apical image slices from all subjects in the test set. Mean and standard deviation (in parenthesis) are reported.

Methods	UK Biobank		ACDC		Methods	UK Biobank		ACDC	
	Dice↑	HDist↓	Dice↑	HDist↓		Dice↑	HDist↓	Dice↑	HDist↓
R-L_2-1×1	.785(.047)	10.69(3.05)	.860(.058)	6.74(2.40)	U-L_2-1×1	.784(.046)	10.65(3.03)	.860(.062)	6.72(2.53)
R-L_2-2×1	.795(.046)	10.69(3.15)	.850(.063)	6.67(2.19)	U-L_2-2×1	.791(.045)	10.52(3.06)	.861(.058)	6.57(2.43)
R-L_2-2×2	.798(.046)	10.49(3.12)	.867(.054)	6.60(2.38)	U-L_2-2×2	.793(.044)	10.48(3.09)	**.866(.056)**	6.58(2.53)
R-L_2-3×2	.799(.044)	10.63(3.18)	**.872(.052)**	**6.44(2.38)**	U-L_2-3×2	.802(.043)	**10.32(3.09)**	.866(.060)	**6.52(2.43)**
R-L_2-6×1	**.804(.043)**	**10.26(3.07)**	.869(.054)	6.65(2.50)	U-L_2-6×1	**.803(.043)**	10.47(3.14)	.856(.062)	6.76(2.52)
R-L_1-1×1	.779(.048)	10.77(3.03)	.853(.061)	6.75(2.53)	U-L_1-1×1	.781(.047)	10.77(3.07)	.854(.063)	6.88(2.58)
R-L_1-2×1	.789(.047)	10.62(3.13)	.865(.058)	6.51(2.42)	U-L_1-2×1	.783(.048)	10.74(3.04)	.858(.062)	6.77(2.54)
R-L_1-2×2	.794(.045)	10.58(3.00)	.865(.060)	6.48(2.38)	U-L_1-2×2	.793(.046)	10.56(3.01)	.867(.058)	6.55(2.48)
R-L_1-3×2	.796(.046)	10.54(3.16)	**.873(.050)**	**6.33(2.13)**	U-L_1-3×2	**.797(.045)**	**10.53(3.14)**	**.872(.052)**	**6.39(2.50)**
R-L_1-6×1	**.800(.045)**	**10.35(3.10)**	.850(.063)	6.67(2.19)	U-L_1-6×1	.790(.098)	10.61(3.10)	.845(.068)	7.03(2.62)

ACDC, the maximum iterations are 20,000 and the learning rate is gradually reduced after 10,000 iterations. For the 3DCMR dataset, the maximum iterations are 100,000 and the learning rate is kept fixed during training.

5.5.3 Parameter Studies

In this section, we explore the impact of using different data terms, denoising networks, parameterizations, and varying numbers of cascades. For simplicity, we use shorthand notations to represent different configurations. For example, R-L$_1$-3×2 indicates that we use the U-Net with residual connection, Eq. (5.5) (L_1 data term), $N_{warp} = 3$ and $N_{iter} = 2$ in VR-Net. U-L$_2$-6×1 indicates that we use the U-Net without residual connection, Eq. (5.6) (L_2 data term), $N_{warp} = 6$ and $N_{iter} = 1$.

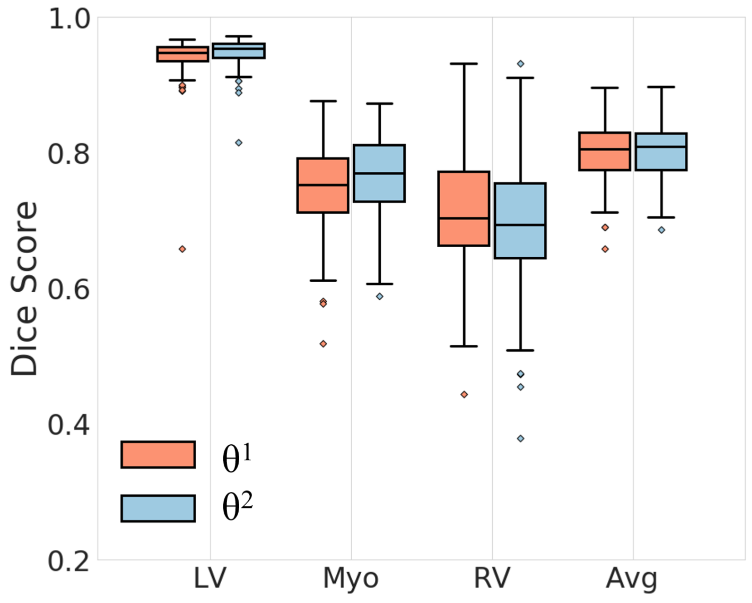


Figure 5.3: The Dice score of VR-Net on UK Biobank with two different parameterizations: $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$. Boxplots on the LV, Myo, RV three structures, and their average are provided.

We first compare the results obtained by using different cascades in VR-Net. From Table 5.1, we observe that the best results almost all come from using 6 cascades (either 3×2 or 6×1), indicating that increasing cascade number improves the performance. On the UK Biobank, the best result is achieved by R-L$_2$-6×1 (0.804 Dice and 10.26mm HDist), while on ACDC the best result is achieved by R-L$_1$-3×2 (0.873 Dice and 6.33mm HDist). When comparing the best performance among different data terms, L_1 performs worse than L_2 on UK Biobank, while on ACDC L_1 is better. This suggests that the proposed VR-Net is robust to different data terms. Next, we compare the results obtained by using different denoising networks, and we notice a tiny improvement when a residual connection is applied.

In Fig. 5.3, we show the performance of VR-Net on UK Biobank with two different parameterizations: $\mathbf{\Theta}^1$ and $\mathbf{\Theta}^2$ in Sec. 5.4.2. From these boxplots, we see that using $\mathbf{\Theta}^2$ performs better on RV and Myo anatomical structures, while on RV using $\mathbf{\Theta}^1$ is better. The averaged results (last two columns) on the three anatomical regions indicate a similar performance

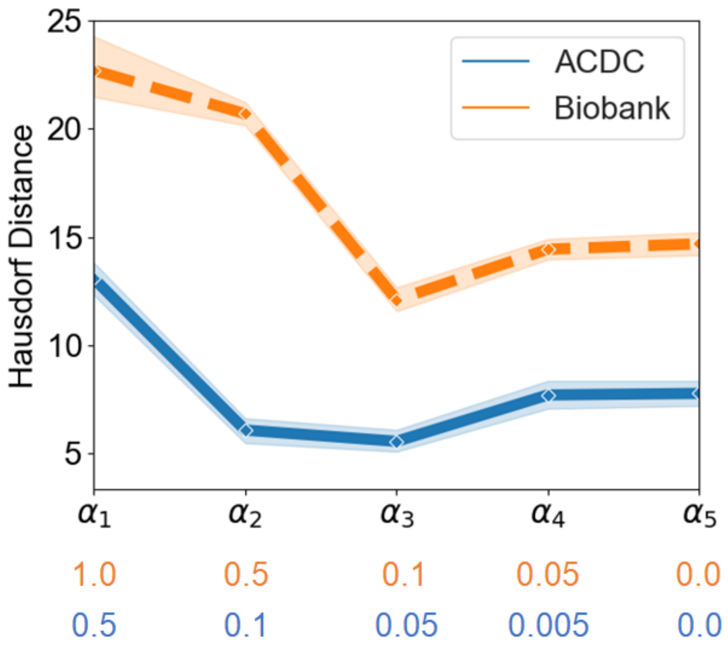


Figure 5.4: Impact of using different α in terms of HDist on the UKBB and ACDC datasets. The specific values of α are labeled with colors for both datasets.

between the two parameterizations. Note that the number of trainable network parameters in $\mathbf{\Theta}^1$ is much fewer than in $\mathbf{\Theta}^2$ due to the parameter sharing scheme.

While the original regularization weight λ is absorbed in the $\boldsymbol{\nu}$-subproblem to avoid manual choice, by using the training loss in Eq. (5.8) we do introduce another parameter α. However, tuning α is based on the whole dataset and we tune it only during training. We presented a curve plot that illustrates how different α affect the registration accuracy (as shown in Fig. 5.4). Specifically, we used five different values of α to train the proposed VR-Net five times on UKBB and ACDC datasets, i.e., $\alpha_{UKBB} = \{1, 0.5, 0.1, 0.05, 0\}$ and $\alpha_{ACDC} = \{0.5, 0.1, 0.05, 0.005, 0\}$. We then plot their registration accuracy (in terms of HDist) on each dataset as α varies. As suggested by the curve plot, the optimal values of α for UK Biobank and ACDC are 0.1 and 0.05, respectively.

Table 5.2: Performance of VR-Net on UK Biobank using different initialization strategies. Note the U-Net is not pre-trained, it is also a learnable layer in the whole VR-Net.

Initialization	Dice$\uparrow$	HDist$\downarrow$
U-Net	.804(.043)	10.26(3.07)
Noise	.740(.048)	11.68(3.05)
Zeros	.744(.051)	11.85(3.18)

5.5.4 Initialization Strategies

In Table 5.2, we explore the performance of VR-Net using different initialization strategies on the UK Biobank dataset. As is evident in this table, with zeros or noises as the initial displacements, the Dice scores of VR-Net dropped by 6.0% and 6.4%, and the HDist values increased by 1.59mm and 1.42mm, respectively. These results suggest that making the initialization learnable is crucial as (1) registration is nonconvex and its solution depends on initialization, and (2) our network builds on iterative optimization methods and thus also relies on initialization. Furthermore, our VR-Net is derived using the Taylor linearization and as such computes only a small displacement in each iteration. When we initialize the input displacement with noise or zeros, six iterations are not sufficient to perform a good registration.

5.5.5 Comparison with State-of-the-Art

In this section, we compare our VR-Net with iterative methods (i.e. FFD [130] and TV-L_1 [159]) and deep data-driven methods (i.e., VoxelMorph [10, 11], Siamese network [119, 121], and RC-Net [166]) on the UK Biobank, ACDC and 3DCMR dataset.

2D Methods: For FFD, we use the implementation in MIRTK [130], where we chose the

SSD similarity with bending energy regularisation. We use a 3-level multi-resolution scheme and set the spacing of B-spline control points on the highest resolution to $8mm$. For TV-L$_1$, which uses the L_1 data term and the total variation regularization, we implement its ADMM solver, in which we use a similar three-level multi-scale strategy for the minimization. We implement TV-L$_1$ using the same variable splitting and therefore its overall iterative structure is very similar to our VR-Net. However, because TV-L$_1$ is cheap to iterate, we can set sufficient numbers of inner iterations (associated with variable splitting) and outer iterations (associated with Taylor expansions) to compute the final deformation. In other words, we tune TV-L$_1$ to its maximum capability to compete with our method. The regularization weights in the two methods are tuned to maximize the accuracy performance on validation sets. For the data-driven methods, we first compare our VR-Net with VoxelMorph [10] which we re-implement for 2D registration. We also compare VR-Net with the Siamese network regularized by the approximated Huber loss [119, 121]. Lastly, we re-implement a 2D version of RC-Net [166]. Overall, the backbone of both VoxelMorph and RC-Net is a U-Net and the loss functions (without segmentation loss) are similar to ours.

Note that all the compared data-driven methods (including Siamese, VoxelMorph, and RC-Net) are only trained with the training data and no test-time (instance-level) optimization is adopted. The hyper-parameters of all data-driven methods are tuned individually according to the validation set for a fair comparison.

3D Methods: We again use a three-level pyramid scheme with SSD similarity and bending energy regularisation for FFD, tuning control point spacing on the validation set. Next, we compare our VR-Net with the diffeomorphic Demons [143] implemented in SimpleITK [96]. For Demons, we use a three-level pyramid scheme and optimize the hyper-parameters (such as the number of iterations, smoothing parameters, etc.) on the validation set. Finally, we compare with the official ANTs SyN implementation [7] with SSD similarity and a four-level pyramid scheme. Hyper-parameters in ANTs SyN such as similarity, number of

Table 5.3: Comparison of registration performance using different methods on UK Biobank. 'Avg' means that Dice (HDist) is computed by averaging that of LV, Myo, and RV of all subjects in the test set. #×RC-Net stands for the number of cascades used in RC-Net.

Methods	Dice↑				HDist↓				$\lvert J\rvert_{<0}$%
	LV	Myo	RV	Avg	LV	Myo	RV	Avg	
Initial	.634(.072)	.344(.086)	.551(.080)	.510(.055)	11.99(1.64)	10.08(2.91)	24.52(6.24)	15.53(2.40)	–
FFD	.934(.025)	.711(.081)	.672(.110)	.772(.051)	4.87(2.15)	7.86(4.03)	21.18(7.69)	11.30(3.26)	0.23(0.29)
TV-L_1	.937(.036)	.717(.076)	.701(.105)	.785(.047)	4.75(1.67)	7.12(3.16)	19.73(7.21)	10.53(2.86)	0.65(0.30)
Siamese	.932(.022)	.706(.069)	.695(.099)	.778(.046)	4.75(1.65)	6.52(3.23)	20.69(7.02)	10.65(3.01)	0.42(0.21)
VoxelMorph	.931(.029)	.717(.072)	.685(.102)	.778(.047)	4.57(1.47)	6.71(3.43)	21.78(7.27)	10.69(3.06)	0.07(0.10)
2×RC-Net	.942(.022)	.737(.066)	.703(.099)	.794(.044)	4.43(1.57)	6.65(3.35)	20.29(7.15)	10.46(3.01)	0.22(0.17)
3×RC-Net	.944(.036)	.736(.068)	**.705(.105)**	.795(.048)	4.28(1.81)	7.39(3.32)	**19.96(6.53)**	10.55(2.80)	0.70(0.36)
4×RC-Net	.945(.022)	.736(.065)	.701(.120)	.794(.045)	4.36(1.54)	7.23(3.39)	20.54(7.32)	10.71(2.98)	0.49(0.24)
5×RC-Net	.944(.033)	.723(.065)	.703(.109)	.790(.045)	4.24(1.60)	8.09(3.51)	20.16(6.83)	10.83(2.77)	1.18(0.53)
6×RC-Net	.941(.024)	.714(.068)	.696(.114)	.784(.048)	4.60(1.53)	7.66(3.23)	20.51(7.17)	10.92(2.98)	1.29(0.55)
7×RC-Net	.943(.025)	.721(.066)	.695(.113)	.786(.047)	4.52(1.56)	7.66(3.20)	20.42(7.06)	10.87(2.89)	1.00(0.44)
R-L_2-6×1	**.948(.021)**	**.764(.060)**	.700(.105)	**.804(.043)**	**3.90(1.41)**	**6.49(3.79)**	20.38(7.21)	**10.26(3.07)**	0.38(0.18)

pyramid levels, and number of iterations in each level are tuned on the whole validation set.

In Tables 5.3 and 5.4, we show the quantitative results obtained by using different methods on UK Biobank and ACDC. In the tables, one can see that VR-Net outperforms iterative methods and data-driven methods on both datasets for almost all anatomical structures.

On UK Biobank, RC-Net achieves the best results on RV in terms of both Dice and HDist, which are 0.005 and 0.42mm better than those obtained by our best configuration (R-L_2-6×1). However, in terms of Dice, VR-Net achieves 0.948 on LV and 0.764 on Myo, outperforming 3×RC-Net by 0.004 and by 0.028, respectively. In terms of HDist for LV and Myo, our VR-Net improves 3×RC-Net from 4.28mm to 3.90mm and from 7.39mm to 6.49mm, respectively. On average, the proposed VR-Net achieves a better Dice and HDist than 3×RC-Net, making our VR-Net the best method on this dataset.

Table 5.4: Comparison of registration performance using different methods on ACDC.

Methods	Dice↑				HDist↓				$\lvert J\rvert_{<0}\%$
	LV	Myo	RV	Avg	LV	Myo	RV	Avg	
Initial	.666(.178)	.540(.143)	.672(.145)	.626(.108)	12.21(4.34)	7.65(2.67)	12.74(4.56)	10.87(3.13)	–
FFD	.920(.063)	.792(.067)	.803(.126)	.838(.059)	5.16(2.14)	5.87(2.18)	9.60(4.56)	6.88(2.40)	0.32(0.42)
TV-L_1	.902(.106)	.793(.086)	.835(.117)	.843(.075)	5.97(3.25)	6.11(2.85)	9.51(4.49)	7.20(2.81)	0.52(0.37)
Siamese	.872(.106)	.723(.113)	.778(.132)	.791(.081)	7.34(3.49)	6.05(1.84)	10.55(4.30)	7.98(2.68)	0.15(0.17)
VoxelMorph	.924(.066)	.789(.096)	.837(.104)	.850(.062)	5.51(2.81)	5.83(2.26)	9.33(4.02)	6.89(2.50)	0.38(0.35)
2×RC-Net	.931(.053)	.798(.083)	.864(.082)	.864(.050)	5.46(2.49)	6.22(2.56)	8.63(3.97)	6.77(2.46)	0.54(0.37)
3×RC-Net	.931(.048)	.794(.076)	.852(.095)	.859(.049)	6.08(2.61)	7.02(2.71)	9.15(3.95)	7.42(2.40)	1.02(0.57)
4×RC-Net	.926(.056)	.789(.075)	.849(.086)	.855(.050)	6.01(2.78)	6.62(2.70)	9.32(3.76)	7.32(2.53)	0.95(0.55)
5×RC-Net	.919(.072)	.780(.075)	.849(.091)	.849(.055)	6.38(2.80)	7.29(2.96)	9.37(3.78)	7.68(2.40)	1.24(0.68)
6×RC-Net	.926(.050)	.779(.088)	.853(.090)	.853(.051)	6.67(2.73)	8.02(3.69)	9.19(3.89)	7.96(2.63)	1.94(0.91)
7×RC-Net	.927(.048)	.769(.085)	.842(.104)	.846(.053)	6.72(2.57)	8.44(3.35)	9.44(3.86)	8.20(2.50)	2.17(1.00)
R-L_1-3×2	**.934(.052)**	**.815(.078)**	**.869(.082)**	**.873(.050)**	**5.09(2.20)**	**5.48(2.19)**	**8.43(3.72)**	**6.33(2.13)**	0.32(0.25)

On ACDC, the proposed VR-Net with the configuration of R-L_1-3×2 outperforms all other methods across all anatomical structures. While 2×RC-Net also obtains comparable results, one can notice that its performance drops rapidly with more cascades. To visualize this, we plotted the average Dice scores of both RC-Net and VR-Net versus the number of cascades in Fig. 5.5 on this dataset. As is evident from this figure, there is a sharp decrease in the performance of RC-Net, which is due to RC-Net overfitting the small training set of 40 subjects. In contrast, **VR-Net performs constantly well using an increasing number of cascades, demonstrating its data efficiency.** This is attributable to the integration of the iterative variational model (prior knowledge) into the VR-Net.

On the 3DCMR dataset, as listed in Table 5.5, the proposed VR-Net achieves both the highest Dice (0.790, R-L_1-3×2) and lowest HDist (5.69mm, U-L_1-1×1) among the compared methods. Specifically, VoxelMorph already outperforms the compared iterative methods (ANTs SyN [7], Demons [143], and FFD [130]) in terms of both Dice and HDist, while RC-Net and our VR-Net can further improve the performance of VoxelMorph. An overview of the Dice and HDist of different methods can be found in the boxplots in Fig. 5.6.

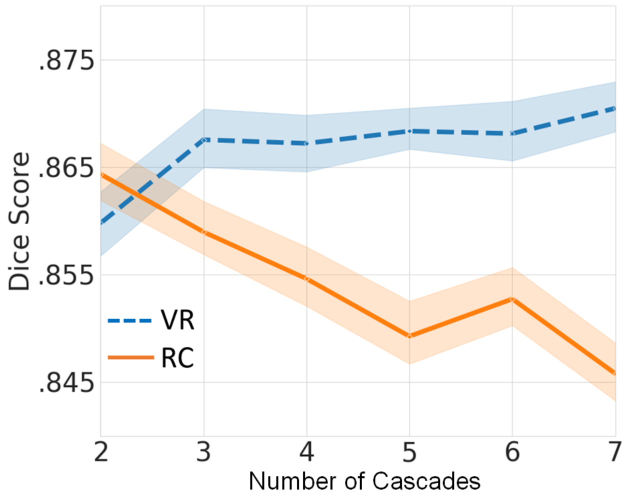
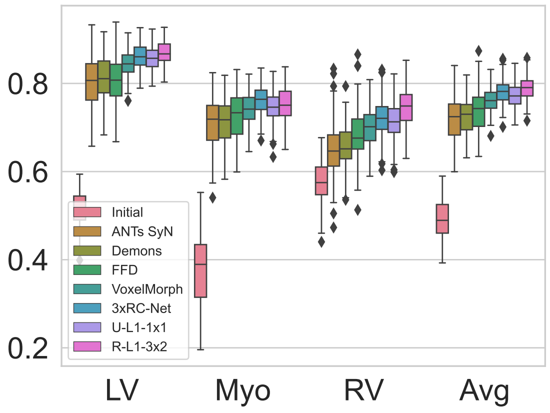
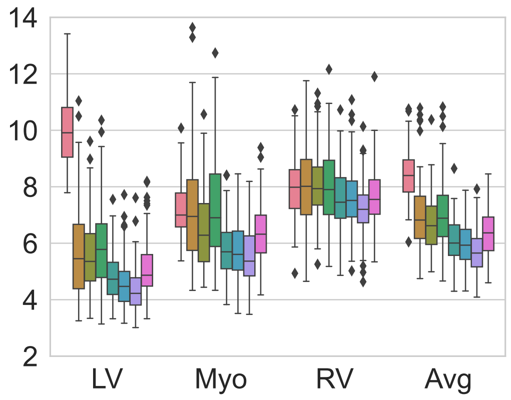


Figure 5.5: Comparing VR-Net and RC-Net using a different number of cascades on ACDC.



(a) 3DCMR Dice



(b) 3DCMR HDist

Figure 5.6: Boxplots of Dice and HDist of different methods on the 3DCMR dataset.

We also listed the values of $|J|_{<0}\%$ for all compared methods. From Tables 5.3 and 5.4, we can see that VR-Net produces fewer foldings than RC-Net with the same number of cascades, i.e., 0.38% of R-L$_2$-6×1 and 1.00% of 7×RC-Net on UK Biobank, and 0.32% of R-L$_1$-3×2 and 2.17% of 7×RC-Net on ACDC. On the 3DCMR dataset, as shown in Table 5.5, VR-Net however produces more folding than the RC-Net.

Table 5.5: Comparison of registration performance using different methods on 3DCMR.

Methods	Dice				HD				$\|J\|_{<0}\%$
	LV	Myo	RV	Avg	LV	Myo	RV	Avg	
Initial	.516(.039)	.384(.084)	.579(.044)	.493(.043)	11.99(1.37)	8.66(1.19)	9.60(1.32)	10.08(1.07)	–
Demons	.812(.049)	.710(.052)	.659(.047)	.727(.040)	5.64(1.28)	6.46(1.28)	8.12(1.21)	6.74(1.00)	0.00(0.00)
FFD	.808(.055)	.726(.059)	.684(.061)	.739(.047)	5.89(1.52)	7.26(1.78)	7.98(1.38)	7.04(1.20)	0.77(0.30)
SyN	.803(.060)	.710(.061)	.649(.064)	.721(.051)	5.70(1.62)	7.26(1.84)	7.98(1.38)	6.98(1.23)	0.01(0.01)
VoxelMorph	.846(.030)	.740(.037)	.701(.047)	.762(.029)	4.85(0.87)	5.79(0.99)	7.59(1.11)	6.08(0.79)	0.11(0.13)
2×RC-Net	.859(.030)	.757(.037)	.720(.050)	.779(.031)	4.81(0.88)	5.88(0.97)	7.61(1.14)	6.10(0.77)	0.01(0.03)
3×RC-Net	.862(.030)	**.760(.035)**	.723(.047)	.781(.030)	4.63(0.92)	5.77(1.00)	7.61(1.15)	6.00(0.78)	<0.01
4×RC-Net	.856(.035)	.759(.035)	.720(.049)	.778(.031)	4.64(0.94)	5.73(0.99)	7.51(1.15)	5.96(0.78)	<0.01
5×RC-Net	.861(.031)	.760(.036)	.720(.047)	.780(.030)	4.70(0.98)	5.75(1.00)	7.53(1.11)	5.99(0.80)	<0.001
6×RC-Net	.862(.032)	.759(.035)	.723(.050)	.781(.031)	4.76(0.93)	5.84(1.03)	7.54(1.11)	6.05(0.77)	0.01(0.02)
R-L_1-1×1	.856(.031)	.748(.039)	.726(.047)	.776(.029)	4.67(0.90)	5.72(0.97)	7.55(1.07)	5.98(0.74)	0.10(0.13)
R-L_2-1×1	.853(.033)	.746(.039)	.725(.048)	.775(.030)	4.71(0.92)	5.77(0.97)	7.54(1.06)	6.01(0.75)	0.05(0.09)
U-L_1-1×1	.856(.029)	.746(.037)	.716(.046)	.772(.029)	**4.35(0.87)**	**5.52(0.97)**	**7.21(0.98)**	**5.69(0.75)**	0.09(0.14)
U-L_2-1×1	.854(.032)	.750(.037)	.723(.048)	.775(.029)	4.56(0.84)	5.68(0.97)	7.39(1.05)	5.88(0.73)	0.09(0.13)
R-L_1-3×2	**.869(.030)**	.753(.039)	**.748(.046)**	**.790(.028)**	5.12(1.02)	6.38(1.06)	7.66(1.06)	6.39(0.80)	0.37(0.23)
R-L_2-3×2	.856(.030)	.755(.038)	.727(.045)	.779(.027)	4.69(1.00)	5.85(1.00)	7.55(1.03)	6.03(0.76)	0.14(0.16)
U-L_1-3×2	.862(.029)	.760(.037)	.734(.046)	.785(.028)	4.88(1.04)	5.86(1.00)	7.74(1.10)	6.16(0.82)	0.05(0.09)
U-L_2-3×2	.860(.031)	.759(.034)	.735(.048)	.785(.027)	4.83(0.93)	5.96(1.00)	7.63(1.10)	6.14(0.74)	0.09(0.12)

In Table 5.6, we list the runtime of different methods. Although we adopt the mathematical structure of a variational model, our VR-Net is very close to the purely data-driven deep learning methods as the solutions are point-wise closed-form, and it is much faster than traditional iterative methods. The runtime is measured and averaged over 100 test subjects.

Lastly, in Figs. 5.7 and 5.8, we compare the visual results of different methods by showing two image registration examples from the ACDC and 3DCMR datasets. On the ACDC, as can be seen, FFD (2nd column), which used L_2 regularization, over-smooths the displacement, and the warped ES images are also over-smoothed around the Myo/LV area resulting in the high absolute differences. In contrast, TV-L_1 (3rd column), which used L_1 regularization, preserves edges in the resulting displacements. However, the shape of RV warped by TV-L_1 is not very smooth. This side effect also can be seen in the Siamese network result,

Table 5.6: Runtimes of different methods.

Methods	2D		3D	
	CPU	GPU	CPU	GPU
TV-L_1	10.01	–	–	–
Demons	–	–	13.01	–
SyN	–	–	77.39	–
FFD	5.15	–	141.38	–
Siamese	0.07	0.01	–	–
VoxelMorph	0.07	0.01	0.46	0.14
2×RC-Net	0.13	0.01	1.05	0.16
3×RC-Net	0.19	0.02	1.51	0.22
7×RC-Net	0.44	0.03	–	–
R,U-$L_{1,2}$-1×1	0.13	0.01	1.01	0.17
R,U-$L_{1,2}$-2×1	0.19	0.02	1.39	0.23
R,U-$L_{1,2}$-6×1	0.42	0.03	–	–

and the Siamese network also produces a very high difference map. The displacement results of VoxelMorph, RC-Net, and VR-Net are smooth and look more natural. Additionally, the Jacobian map of RC-Net has more foldings than VR-Net(highlighted in green). The warped Myo of VoxelMorph from ACDC has an unsmooth shape. The unsmooth shape can also be found in the warped masks of RC-Net. On the 3DCMR, though the warped ES image of VR-Net (R-L_1-3×2) is closer to the target ED image, there are many foldings exist in the displacement.

5.5.6 Discussion

Relationship with VoxelMorph and RC-Net: In the proposed VR-Net, we use an additional U-Net to learn initial displacements. We emphasize here that this U-Net is not pre-trained and instead it is part of the VR-Net, which is trained end-to-end. In this case, without any DL, WL, or ICL layers this initial U-Net alone is essentially VoxelMorph, the

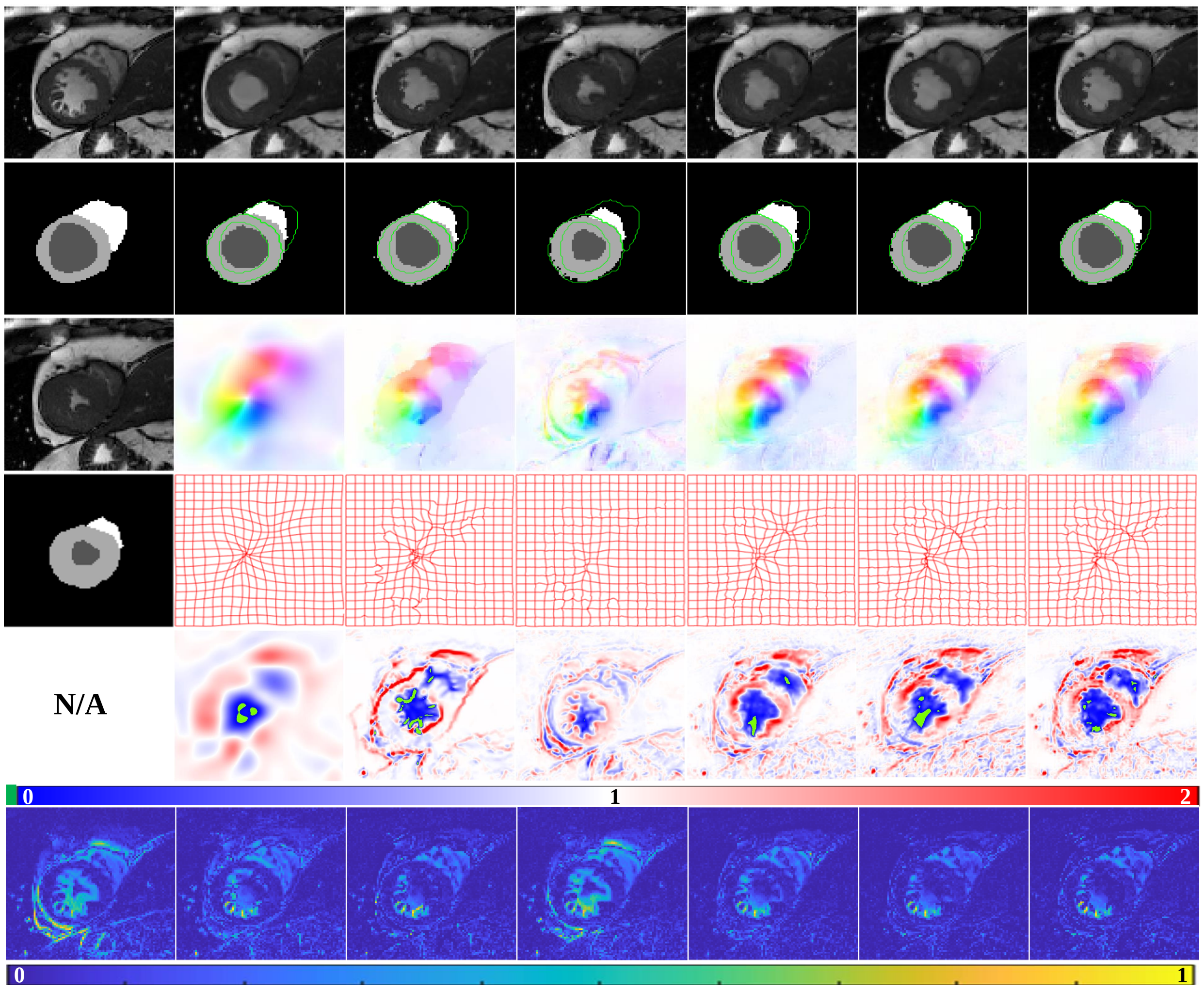


Figure 5.7: Visual results obtained by different methods on the ACDC. The 1st column includes the ED image, ED mask, ES image, ES mask, and the absolute difference between the ES image and the ED image. Excluding the 1st column, from left to right: FFD, TV-L_1, Siamese Net, VoxelMorph, RC-Net, and VR-Net results, respectively. From top to bottom: warped ES images, warped ES masks (with ground truth mask shown in green contours), estimated deformations (shown in HSV mode and grid), the Jacobian map, and absolute differences between warped ES images and the ground truth ED image, respectively.

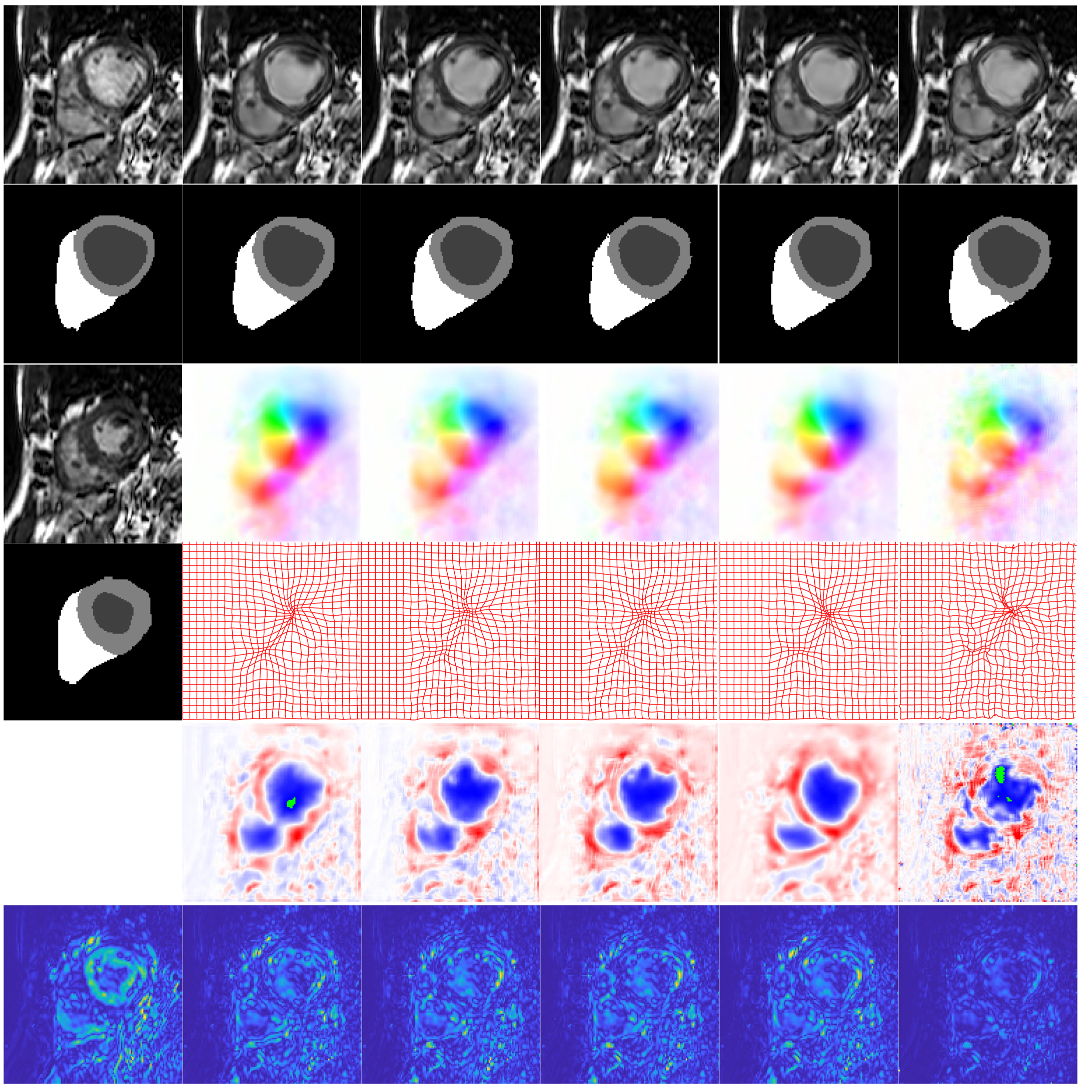


Figure 5.8: Visual results obtained by different registration methods on the 3DCMR. The 1st column includes the ED image, ED mask, ES image, ES mask, and the absolute difference between ES and ED. Excluding the 1st column, from left to right: VoxelMorph, 2×RC-Net, 3×RC-Net, U-L_1-1×1, and R-L_1-3×2 results, respectively. From top to bottom: warped ES images, warped ES masks, estimated deformations (shown in HSV mode and grid), Jacobian maps, and difference maps between warped ES images and the ED image, respectively.

performance of which is inferior to our VR-Net by a large margin as shown in Tables 5.3, 5.4 and 5.5. If we recursively use the U-Net multiple times without using other subsequent layers (such as ICL/GDL), then the model is equivalent to the RC-Net, the performance of which is also inferior to our VR-Net as shown in Tables 5.3, 5.4 and 5.5.

Generalized Denoising Layers: To understand how the GDL layer is functioning within the network, in Fig. 5.9 we illustrate the output of this layer after each cascade of the VR-Net using two different setups. Specifically, we use R-L_2-6×1 using both the U-Net and random noise initialization from Table 5.2. For the U-Net initialization, we add Gaussian noise to the input displacement to demonstrate whether this layer can produce any smoothing effect. As shown in the top two rows of Fig. 5.9, the deformation becomes gradually smooth as cascades proceed. The deformation also gets increasingly smooth for the random noise initialization. This visualization suggests that our GDL has the denoising effect. However, the capability of GDL is beyond denoising alone. As shown in the last two rows in Fig. 5.9, this layer can turn a pure random noise into deformation, indicating its capability of inducing smoothness whilst going beyond denoising and contributing to the deformation itself.

Identifying the Optimal Structure: In Table 5.1, we list 24 configurations of VR-Net, along with proposed two parameterizations of $\mathbf{\Theta^1}$ and $\mathbf{\Theta^2}$. Empirically searching for the best structure can be computationally expensive. What is the strategy to efficiently determine the combination? We observe that the best results almost all come from VR-Net with 6 cascades in 2D datasets, indicating that increasing the cascade number improves performance. We therefore suggest using more cascades if one can afford them. As for the two parameterizations (Θ^1 and Θ^2), we notice a slight improvement using Θ^2 and therefore use this parameterization for all our comparative experiments. Comparing different data terms, we find that L_1 and L_2 are on par with each other, which may be due to their closed-form solutions. We therefore use both L_1 and L_2 data terms for comparative experiments. Next, by comparing the results obtained by using different denoising networks in Tables 5.1 and 5.5,

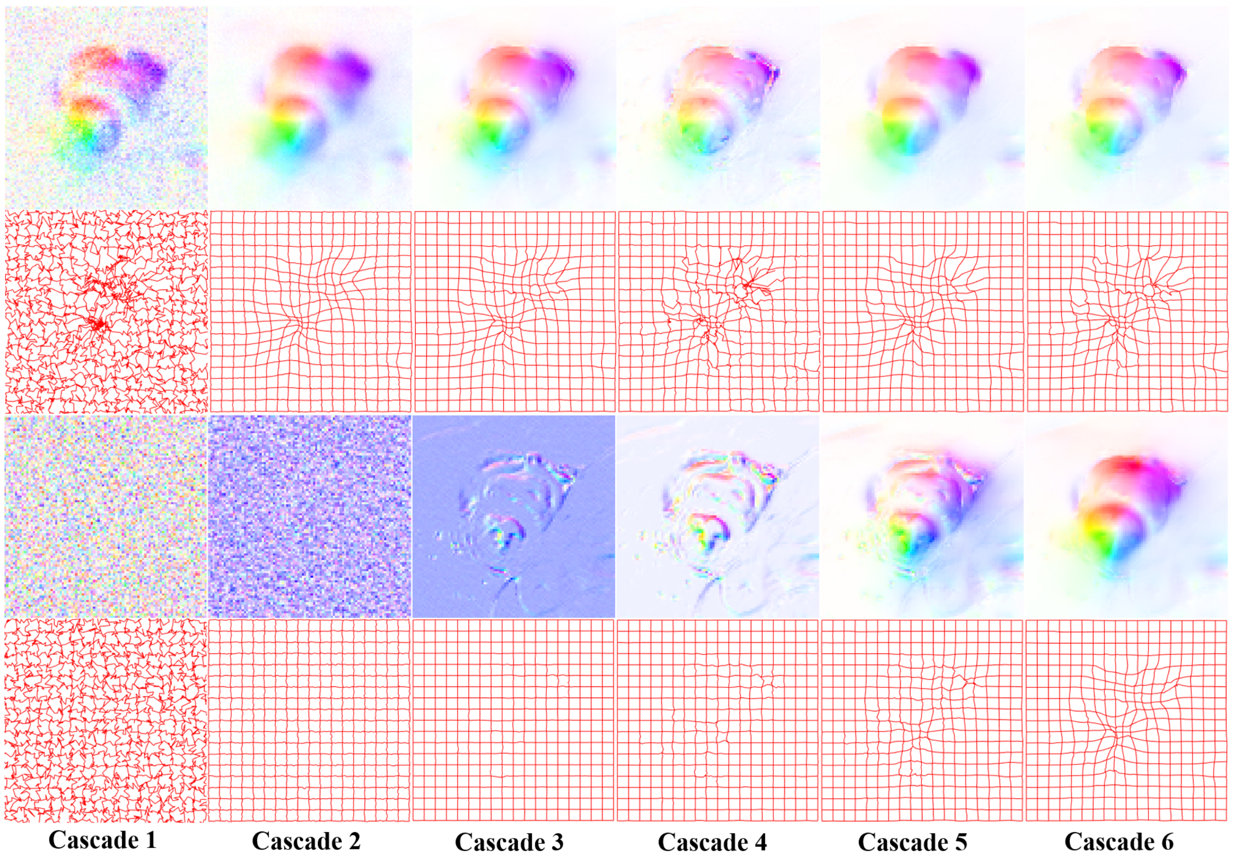


Figure 5.9: Visualizing deformation in each cascade (after GDL) using noise corrupted deformation (top) and random noise (bottom) as initialization. The top two rows show a noise-corrupted deformation is denoised by GDL as cascades proceed. The bottom two rows show if we input random noise, VR-Net is still capable of producing a smooth deformation.

we find residual U-Net performs better and therefore use it for the comparative experiments.

Brightness constancy assumption: The brightness constancy assumption in Eq. (5.2) is often not suited for medical images with contrast variances and therefore our method will not work well for those images. However, we would like to point out that the proposed framework has the flexibility to accommodate various similarity/dissimilarity metrics, such as NCC (invariant to multiplicative illumination changes), MI (suitable for multi-modality image registration), and others. The key is to leverage the power of modern auto-differentiation engines to create the forward computation graph automatically, as implemented in [120]. On the other hand, it is also possible to consider other L_1 or L_2 based data terms, including contrast invariant descriptors based on image gradients [114, 147] or modality independent image descriptors such as nonlocal MIND [60].

5.6 Summary

In this chapter, we propose a model-driven VR-Net for deformable image registration, which combines the iterative variational method with modern data-driven deep learning methods. By taking advantage of both approaches, our VR-Net outperforms deep data-driven methods as well as classical iterative methods on three cardiac MRI datasets. Extensive experimental results show our VR-Net is fast, accurate, and data-efficient.

Chapter Six

Model-Driven Fourier-Net

This chapter is based on the following publications:

Fourier-net: Fast image registration with band-limited deformation [73] which is published in the Proceedings of the AAAI Conference on Artificial Intelligence (2023).

Fourier-Net+: Leveraging band-limited representation for efficient 3D medical image registration [76] which is preprinted on Arxiv (2023).

6.1 Introduction

Deformable image registration plays a critical role in many medical image analysis applications, such as population modeling, longitudinal studies, and statistical atlases [136]. In general, deformable image registration for medical imaging can be categorized as iterative optimization-based registration approaches, their deep learning-based counterparts, and a combination of both approaches.

Before the emergence of deep learning, deformable image registration relied heavily on iterative optimization techniques. Popular approaches from this paradigm include FFD [130], LDDMM [13], DARTEL [5], Demons [143], etc. Though widely applied and theoretically robust, these approaches require instance-level optimization and are computationally intensive, thus limiting their applications in real-time and large-scale volumetric image registration.

Recently, there has been a surge in the use of deep learning-based approaches for medical image registration [23]. The most effective methods, such as VoxelMorph [10, 11], require only one forward pass during inference, making them orders of magnitude faster than traditional iterative methods. Following VoxelMorph, numerous deep neural networks have been proposed for various registration tasks [21, 62, 74, 81, 110, 161, 166, 167]. These networks generally enhance the registration performance through two strategies: cascading U-Net style networks [62, 166, 167] and replacing basic convolutional blocks with more sophisticated alternatives, such as attention-based transformers [21] and parallel convolutional blocks [74]. However, these changes increase the number of network parameters and mult-adds, negatively impacting training and inference efficiency.

An alternative approach for image registration is to combine deep data-driven models with iterative methods, as proposed by [15, 75, 120]. These methods embed the mathematical structure of minimizing a generic objective model into a neural network. By doing

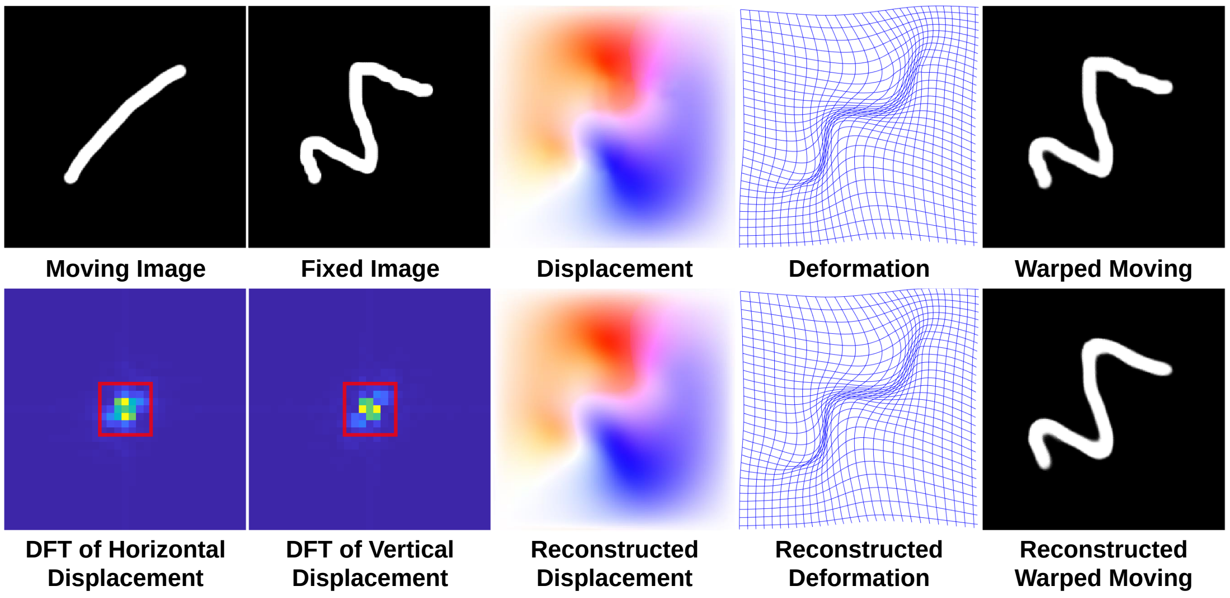

Moving Image
Fixed Image
Displacement
Deformation
Warped Moving
DFT of Horizontal Displacement
DFT of Vertical Displacement
Reconstructed Displacement
Reconstructed Deformation
Reconstructed Warped Moving

Figure 6.1: An example showing connections between deformations in the spatial domain and band-limited Fourier domain: a small number of coefficients (red square) in the band-limited Fourier domain is sufficient to reconstruct a full-resolution deformation accurately.

so, the network mapping process inherits prior knowledge from the reference model while maintaining the data efficiency of iterative methods. As a result, these model-driven networks have the advantages of both communities and have been shown to outperform purely data-driven registration methods. However, despite being faster than traditional iterative optimization-based methods, emulating the iterative optimization process often requires the use of multiple U-Nets and other sophisticated neural layers in these networks (e.g., intensity consistency layer [75]), which often result in a slower registration speed when compared to purely network-based methods.

A common characteristic among learning-based registration approaches is the utilization of U-Net style networks. In this chapter, we argue that for such styles of registration networks, it may not be necessary to include the entire expansive path of the decoder. Additionally, we suggest that training and inference efficiency can be improved by learning a low-dimensional representation of the displacement field in the band-limited Fourier domain. Our observations are based on the results shown in Fig. 6.1, which demonstrate that a small number of coefficients in the band-limited Fourier domain are sufficient to reconstruct a full-resolution deformation accurately. Inspired by this insight, we propose Fourier-Net, an end-to-end model-driven unsupervised registration model that is able to learn such a low-dimensional, band-limited representation of the displacement field. Specifically, by removing several layers in the expansive path of a U-Net style architecture, Fourier-Net outputs only a small patch containing low-frequency coefficients of the displacement field in the Fourier domain. A model-driven decoder then recovers the full-resolution spatial displacement field from these coefficients, using a zero-padding layer that broadcasts complex-valued low-frequency signals into a full-resolution complex-valued map, and an iDFT layer that recovers the spatial displacement field from the map. Both zero-padding and iDFT layers are parameter-free, making our Fourier-Net very fast. We also propose a diffeomorphic variant, termed Diff-Fourier-Net, which learns the band-limited representation of the velocity field and uses the scaling and squaring layers to encourage the output deformation to be diffeomorphic [31].

Building on the results of Fourier-Net, we hypothesize that it may be feasible to learn the band-limited displacement field or band-limited velocity field directly from a band-limited representation of the input image pairs, rather than from the original full-resolution image pairs. This has the potential to reduce the number of convolutional layers in the contracting path in Fourier-Net and further accelerate its registration speed. To this end, we propose Fourier-Net+, which utilizes the same decoder as Fourier-Net but features an improved encoder that aims to reduce computational overhead. As per standard U-Net style architectures, Fourier-Net is designed to take the original full-resolution image pairs as input, with an encoder involving multiple 3D convolutional layers which are computationally expensive operations, particularly in the earlier layers of the model. In contrast, the encoder of Fourier-Net+ includes a model-driven encoding of the original full-resolution image pairs

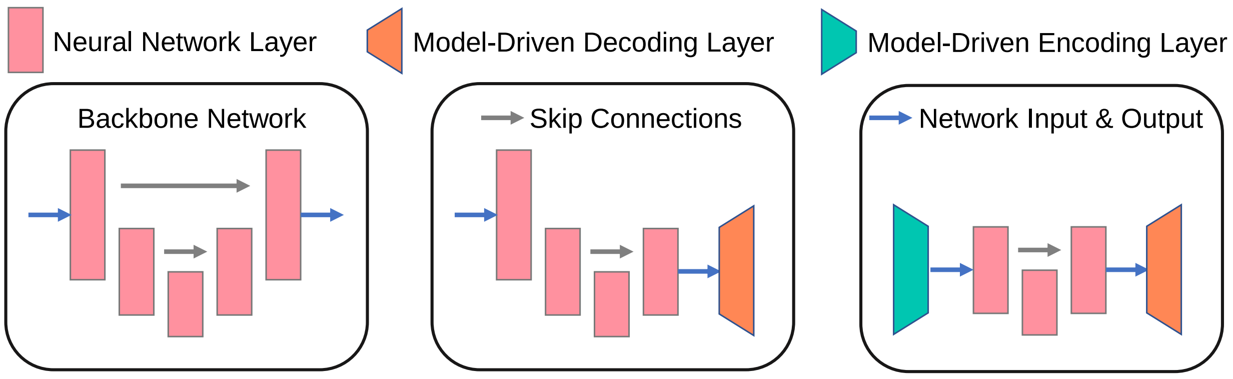


Figure 6.2: Schematic overview of deep registration models. Left: U-Net style architecture that most methods adopted to predict the dense deformation field. Middle: Approaches like B-Spline [122], DeepFlash [150], and our Fourier-Net that discard part of the expansive path and use model-driven decoding layer instead. Right: Fourier-Net+, in which both the contracting and expansive path are model-driven, greatly reducing the parameters and computational costs of the network.

into a low-dimensional band-limited representation of such image pairs, followed by 3D convolutions. This encoder enables 3D convolutions to operate on smaller resolutions, making Fourier-Net+ a much lighter network compared to Fourier-Net and U-Net style backbones. Fig. 6.2 provides a visual comparison of the architectural differences between U-Net, Fourier-Net, and Fourier-Net+. To enhance the registration performance of Fourier-Net+, we then propose a cascaded version of this network, namely Cascaded Fourier-Net+. Despite the potential increase in computational cost from using multiple cascades, the efficient design of Fourier-Net+ allows the cascaded version of this network to still have fewer mult-adds than Fourier-Net in our experiments.

6.2 Related Works

6.2.1 Iterative Optimization-Based Approaches

Iterative methods based on instance-level optimization are prohibitively slow, especially when the images to be registered are of a high-dimensional form. Over the past decades, many works have been proposed to accelerate such methods. For example, rather than directly estimating dense displacement fields, FFD models [130] were proposed in which a deformation grid over a few control points is interpolated to a dense field using B-splines. Computational challenges in diffeomorphic registration are even more pronounced. For this task, Ashburner [5] introduced DARTEL to accelerate the computation of LDDMM, which integrates diffeomorphic deformations through non-stationary velocities over time using the Lagrange transport equation. DARTEL used a stationary velocity field (SVF) representation [4, 89] and computed the resulting deformation through scaling and squaring of the SVF. Zhang and Fletcher [163] developed Flash, where they proposed to speed up the solution of the EPDiff equation used to compute diffeomorphic deformations from velocity fields in the band-limited Fourier domain. Hernandez [64] reformulated the Stokes-LDDMM variational problem used in [101] in the domain of band-limited non-stationary vector fields and utilized GPUs to parallelize their methods. Another fast approach for deformable registration is Demons [143], which imposed smoothness on displacement fields by incorporating inexpensive Gaussian convolutions into its iterative process. The diffeomorphisms in Demons were achieved by the greedy composition of the speed vector fields [143]. Recently, Thorley et al. [140] proposed a convex optimization model that used an arbitrary order regularisation term. By combining Nesterov's accelerated gradient descent and the alternating direction method of multipliers (ADMM), this model was shown to register pairs of 3D cardiac MR images ($128 \times 128 \times 96$) within 2s on GPU.

6.2.2 Deep Registration Approaches

Unsupervised methods: Deep unsupervised data-driven methods have recently been used to accelerate the speed of registration [11, 69, 119]. U-Net style networks, in particular, have been shown to be effective in learning deformations between pairwise images [10, 75, 81, 110, 161]. VoxelMorph [10] demonstrated that a simple U-Net can achieve comparable registration performance to iterative methods, with inference times orders of magnitude faster. Building on the success of VoxelMorph, various extensions have been proposed to improve registration accuracy and address specific challenges in medical image registration.

One such extension used multiple recurrent or cascaded U-Net architectures, where the deformation was iteratively composed by incorporating the output from each cascade [62, 166, 167]. These methods were shown to be particularly effective at handling large deformations in input image pairs by allowing the network to capture both global and local features of the images. Another approach to improving U-Net-based registration models was to augment the U-Net backbone with more representative layers to better capture correspondences between input images. For example, transformer-based methods, such as ViT-V-Net [22] and TransMorph [21], replaced some of the regular convolutional layers in the U-Net style network with transformer layers, allowing for the modeling of long-range dependencies between voxels, and LKU-Net [74] employed parallel large-kernel convolutional layers to handle different scales in input images. Although combining U-Nets with more representative layers has been shown to achieve promising registration performance, these methods come with a significantly higher computational cost, leading to slower inference speed. Another alternative to U-Net style architectures proposed the use of Siamese or dual-stream networks as the backbone [119, 121]. These models utilized separate encoder branches to extract features from the moving and fixed images, which were then fused and passed through the decoder to generate the deformation. Similar to U-Net style networks, Siamese and dual networks also used a contracting path for image encoding and an expansive path for decoding the deformations from the encoded image features.

There are several other extensions which have been shown to improve registration performance, including the use of different loss functions and regularization techniques, the incorporation of prior knowledge, and the estimation of deformation using B-spline control points. For instance, the segmentation Dice loss [10, 53] measures the overlap between segmented regions of fixed and moving images and has been used to improve registration performance between different anatomical structures. Regularization techniques, such as inverse consistency [161] and scaling and squaring [31], are utilized to encourage diffeomorphic deformations. In addition, combining deep learning with conventional model-based approaches [75, 120] and incorporating prior knowledge [43] have also demonstrated improvements in registration performance. Another line of work is to estimate a grid of B-spline control points with regular spacing [122, 148], which is then interpolated based on cubic B-spline basis functions [129, 130]. These networks predict deformations more efficiently by estimating only a few control points, although currently they are less accurate.

Supervised methods: Instead of minimizing a similarity metric and regularization term as per unsupervised methods, supervised approaches learn from ground truth deformation fields. However, they have several pitfalls: 1) it is generally hard to provide human-annotated ground truth deformations for supervision; and 2) if trained using numerical solutions of other iterative methods, the performance of these supervised registration methods may be limited by iterative methods. Yang et al. proposed Quicksilver [158] which is a supervised encoder-decoder network and trained using the initial momentum of LDDMM as the supervision signal. Wang et al. extended Flash [163] to DeepFlash [150] in a learning framework in lieu of iterative optimization. Compared to Flash, DeepFlash accelerated the computation of the initial velocity fields but still needed to solve an EPDiff equation in the Fourier domain to recover the full-resolution deformation in the spatial domain, which was shown to be slow.

The fact that DeepFlash required the numerical solutions of Flash [163] as training data attributes to lower registration performance than Flash.

Although DeepFlash also learns a low-dimensional band-limited representation, it differs from our Fourier-Net in four aspects, which represent our novel contributions to this area. First, DeepFlash is a supervised method that requires ground truth velocity fields calculated from Flash (30 minutes per 3D image pair in CPU) prior to training, whilst Fourier-Net is a simple and effective unsupervised method thanks to our proposed model-driven decoder. Second, DeepFlash is a multi-step method whose network's output requires an additional PDE algorithm [163] to compute final full-resolution spatial deformations, whilst Fourier-Net is a holistic model that can be trained and used in an end-to-end manner. Third, DeepFlash requires two individual convolutional networks to estimate real and imaginary signals in the band-limited Fourier domain, whilst Fourier-Net uses only one single network directly mapping image pairs to a reduced-resolution displacement field without the need of complex-valued operations. Lastly, DeepFlash is essentially an extension of Flash and it is difficult for the method to benefit from vast amounts of data, whilst Fourier-Net is flexible and can easily learn from large-scale datasets.

6.3 Methodology

In this section, we first introduce Fourier-Net, and its extended versions: Fourier-Net+ and Cascaded Fourier-Net+. We then detail their network architectures and loss functions.

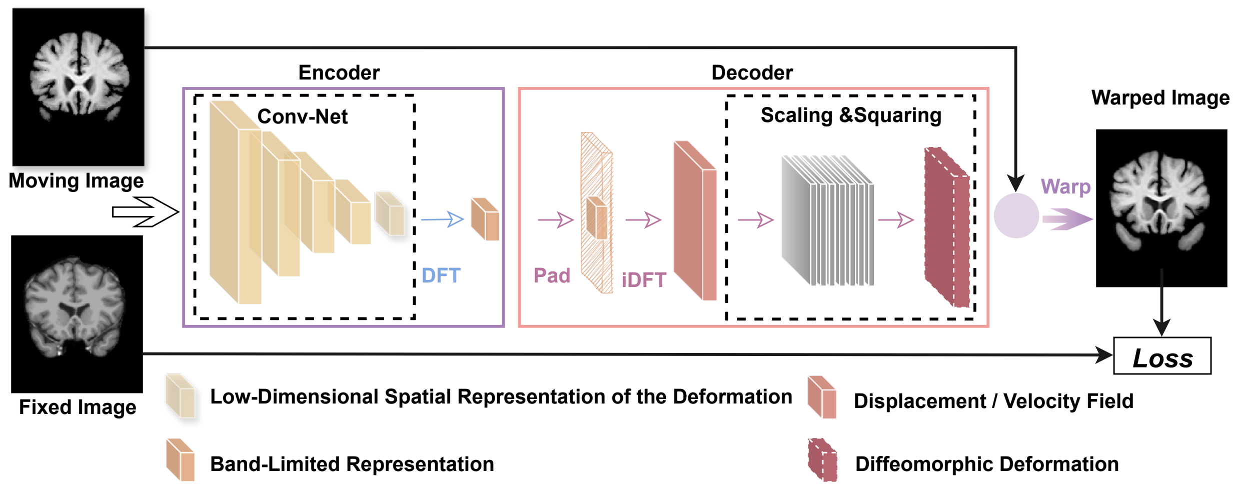

Moving Image
Fixed Image
Encoder
Conv-Net
DFT
Decoder
Pad
iDFT
Scaling &Squaring
Warp
Warped Image
Loss
Low-Dimensional Spatial Representation of the Deformation
Band-Limited Representation
Displacement / Velocity Field
Diffeomorphic Deformation

Figure 6.3: Architecture of Fourier-Net. It contains 1) a convolutional encoder that first produces a low-dimensional spatial representation of a displacement or velocity field, followed by an embedded DFT layer to map this low-dimensional representation into the band-limited Fourier domain; 2) a parameter-free model-driven decoder that adopts a zero-padding layer, an iDFT layer, and seven optional scaling and squaring layers to reconstruct the displacement field or deformation into the full-resolution spatial domain from its band-limited Fourier domain; 3) a warping layer to deform the moving image; and 4) a learning objective function.

6.3.1 Fourier-Net

We illustrate Fourier-Net in Fig. 6.3, where its encoder takes a pair of spatial images as input and encodes them to a low-dimensional representation of the displacement field (or velocity field if diffeomorphisms are imposed) in the band-limited Fourier domain. The decoder then brings the displacement field (or velocity field) from the band-limited Fourier domain to the spatial domain via zero-padding and iDFT. The decoder ensures that the input and output have the same spatial size. Next, scaling and squaring layers are optionally used to encourage a diffeomorphism in the final deformation. Finally, by minimizing the loss function, the warping layer deforms the moving image to be similar to the fixed image.

Encoder of Fourier-Net: the encoder aims to learn a displacement or velocity field in the band-limited Fourier domain, which requires the encoder to handle complex-valued numbers. One may directly use complex convolutional networks [141], which were designed for the case where both input and output are complex values. Note that complex convolutional operations sacrifice computational efficiency. Instead, DeepFlash [150] tackles this problem by first converting input image pairs to the Fourier domain and then using two individual real-valued convolutional networks to learn the real and imaginary signals separately. However, such an approach makes training and inference costly.

To bridge the domain gap between real-valued spatial images and complex-valued band-limited displacement fields without increasing complexity, we propose to embed a DFT layer at the end of the encoder. This is a simple and effective way to produce complex-valued band-limited displacement fields without the network handling complex values itself. Let us denote the moving image as $\boldsymbol{I_M}$, the fixed image as $\boldsymbol{I_F}$, the convolutional network as $\boldsymbol{CNN}$ with the parameters Θ^1, the DFT layer as $\mathcal{F}$, the full-resolution spatial displacement field as $\boldsymbol{\phi}$, and the complex band-limited displacement field as $\mathbb{B}_{\boldsymbol{\phi}}$. We therefore define our encoder as $\mathbb{B}_{\boldsymbol{\phi}} = \mathcal{F}(\boldsymbol{CNN}(\boldsymbol{I_M}, \boldsymbol{I_F}; \Theta^1))$, resulting in a compact, efficient implementation in contrast to DeepFlash [150].

We also attempted to regress $\mathbb{B}_{\boldsymbol{\phi}}$ directly from $\boldsymbol{I_M}$ and $\boldsymbol{I_F}$ using only convolutional layers, i.e., $\mathbb{B}_{\boldsymbol{\phi}} = \boldsymbol{CNN}(\boldsymbol{I_M}, \boldsymbol{I_F}; \Theta^1)$. However, our experimental results suggested that this is very difficult for the network to learn, resulting in lower performance as detailed in Table 6.1. If the CNN is required to directly learn a band-limited displacement field, it must go through two domains in total: first mapping the spatial images to the spatial displacement field and then mapping this displacement field into its band-limited Fourier domain. In this case, we believe the domain gap is too big for a CNN to learn such a mapping. Our network however only needs to go through one domain and then the DFT layer ($\mathcal{F}$) handles the second domain. Experimentally, we found this approach to be more effective.

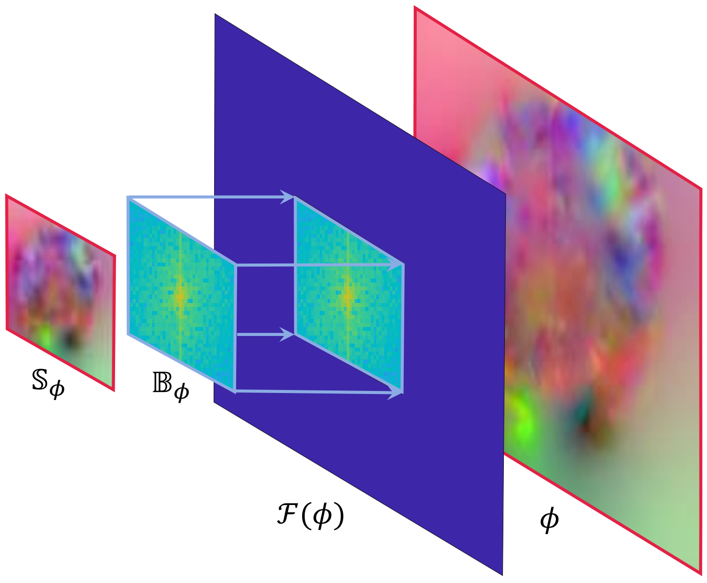


Figure 6.4: Connection between low-dimensional spatial displacement field $\mathbb{S}_{\boldsymbol{\phi}}$, band-limited Fourier coefficients $\mathbb{B}_{\boldsymbol{\phi}}$, full-resolution Fourier coefficients $\mathcal{F}(\boldsymbol{\phi})$ by zero-padding $\mathbb{B}_{\boldsymbol{\phi}}$, and full-resolution displacement field $\boldsymbol{\phi}$ by taking iDFT of $\mathcal{F}(\boldsymbol{\phi})$.

So far, we have provided an intuitive explanation of the learning process of the encoder in our network. We now discuss the mathematical relationship between the low-dimensional spatial displacement field $\mathbb{S}_{\boldsymbol{\phi}} = \boldsymbol{CNN}(I_0, I_1; \Theta^1)$, its band-limited representation $\mathbb{B}_{\boldsymbol{\phi}} = \mathcal{F}(\mathbb{S}_{\boldsymbol{\phi}})$, as well as the displacement field $\boldsymbol{\phi}$ in the full-resolution spatial domain (see Fig. 6.4 for details). For simplicity, we use a 2D displacement field as an example and the formulations below can be easily extended to 3D cases. A general DFT used on $\boldsymbol{\phi}$ can be given as

$$[\mathcal{F}(\boldsymbol{\phi})]_{k,l} = \sum_{i=0}^{H-1}\sum_{j=0}^{W-1} \boldsymbol{\phi}_{i,j} e^{-\sqrt{-1}\left(\frac{2\pi k}{H}i + \frac{2\pi l}{W}j\right)}, \tag{6.1}$$

where $\boldsymbol{\phi}$ is of size $H \times W$, $i \in [0, H-1]$ and $j \in [0, W-1]$ are the discrete indices in the spatial domain, and $k \in [0, H-1]$ and $l \in [0, W-1]$ are the discrete indices in the frequency domain. In our Fourier-Net, $\boldsymbol{\phi}$ is actually a low-pass filtered displacement field. If we define a $H \times W$ sized sampling mask $\mathcal{D}$ whose entries are zeros if they are on the positions of high-frequency signals in $\boldsymbol{\phi}$ and ones if they are on the low-frequency positions. With $\mathcal{D}$, we recover the displacement field $\boldsymbol{\phi}$ from Eq. (6.1),

$$\boldsymbol{\phi}_{i,j} = \frac{1}{HW} \sum_{k=0}^{H-1} \sum_{l=0}^{W-1} \mathcal{D}_{k,l} [\mathcal{F}(\boldsymbol{\phi})]_{k,l} e^{\sqrt{-1}\left(\frac{2\pi i}{H}k + \frac{2\pi j}{W}l\right)}. \tag{6.2}$$

If we shift all low-frequency signals of $\boldsymbol{\phi}$ to a center patch of size $\frac{H}{a} \times \frac{W}{b}$ ($\frac{H}{a}, \frac{W}{b}, a = 2Z_a, b = 2Z_b, Z_a, Z_b \in \mathbb{Z}^+$), center-crop the patch (denoted by $\mathbb{B}_{\boldsymbol{\phi}}$), and then perform the iDFT on this patch, we obtain $\mathbb{S}_{\boldsymbol{\phi}}$ in Eq. (6.3),

$$[\mathbb{S}_{\boldsymbol{\phi}}]_{\widehat{i},\widehat{j}} = \frac{ab}{HW} \sum_{\widehat{k}=0}^{\frac{H}{a}-1} \sum_{\widehat{l}=0}^{\frac{W}{b}-1} [\mathbb{B}_{\boldsymbol{\phi}}]_{\widehat{k},\widehat{l}} e^{\sqrt{-1}\left(\frac{2\pi a\widehat{i}}{H}\widehat{k} + \frac{2\pi b\widehat{j}}{W}\widehat{l}\right)}, \tag{6.3}$$

where $\widehat{i} \in [0, \frac{H}{a} - 1]$ and $\widehat{j} \in [0, \frac{W}{b} - 1]$ are the indices in the spatial domain, and $\widehat{k} \in [0, \frac{H}{a} - 1]$ and $\widehat{l} \in [0, \frac{W}{b} - 1]$ are the indices in the frequency domain. Note that $\mathbb{S}_{\boldsymbol{\phi}}$ is a low-dimensional spatial representation of $\boldsymbol{\phi}$ and we are interested in their mathematical connection. Another note is that $\mathbb{S}_{\boldsymbol{\phi}}$ actually contains all the information of its band-limited Fourier coefficients in $\mathbb{B}_{\boldsymbol{\phi}}$. As such, we do not need the network to learn the coefficients in $\mathbb{B}_{\boldsymbol{\phi}}$ and instead only to learn its real-valued coefficients in $\mathbb{S}_{\boldsymbol{\phi}}$.

Since most of entries ($\frac{a\times b-1}{a\times b}$) in $\mathcal{F}(\boldsymbol{\phi})$ are zeros, and the values of remaining entries are exactly the same as in $\mathbb{B}_{\boldsymbol{\phi}}$, we can conclude that $\mathbb{S}_{\boldsymbol{\phi}}$ contains all the information $\boldsymbol{\phi}$ can provide, and their mathematical connection is

$$[\mathbb{S}_{\boldsymbol{\phi}}]_{\widehat{i},\widehat{j}} = ab \times \boldsymbol{\phi}_{a\widehat{i},b\widehat{j}}, \tag{6.4}$$

with this derivation, we show that we can actually recover a low-dimensional spatial representation $\mathbb{S}_{\boldsymbol{\phi}}$ from its full-resolution spatial displacement field $\boldsymbol{\phi}$, as long as they have the same low-frequency coefficients $\mathbb{B}_{\boldsymbol{\phi}}$. This shows that there exists a unique mapping function between $\mathbb{S}_{\boldsymbol{\phi}}$ and $\boldsymbol{\phi}$ and that it is reasonable to use a network to learn $\mathbb{S}_{\boldsymbol{\phi}}$ directly.

Model-driven decoder: The decoder contains no learnable parameters. We instead replace the expansive path with a zero-padding layer, an iDFT layer, and an optional scaling and squaring layer. The output from the encoder is a band-limited representation $\mathbb{B}_{\boldsymbol{\phi}}$. To recover the full-resolution displacement field $\boldsymbol{\phi}$ in the spatial domain, we first pad the patch $\mathbb{B}_{\boldsymbol{\phi}}$ containing mostly low-frequency signals to the original image resolution with zeros (i.e., $\mathcal{F}(\boldsymbol{\phi})$). We then feed the zero-padded complex-valued coefficients $\mathcal{F}(\boldsymbol{\phi})$ to an iDFT layer consisting of two steps: shifting the Fourier coefficients from centers to corners and then applying the standard iDFT to convert them into the spatial domain. The output from Fourier-Net is thus a full-resolution spatial displacement field. An illustration of this process is given in Fig. 6.4. Both padding and iDFT layers are differentiable and therefore Fourier-Net can be optimized via standard back-propagation.

We also propose a diffeomorphic variant of Fourier-Net which we term Diff-Fourier-Net. In Diff-Fourier-Net, the output of the iDFT layer can be regarded as an SVF denoted by $\boldsymbol{v}$. In group theory, $\boldsymbol{v}$ is a member of Lie algebra, and we can exponentiate this SVF to obtain a diffeomorphic deformation (i.e., $Exp(\boldsymbol{v})$). In this chapter, we use seven scaling and squaring layers [5, 31] to impose such a diffeomorphism.

6.3.2 Fourier-Net+

In Fourier-Net, we proposed a model-driven decoder to replace some of the expansive convolutional layers in a U-Net style backbone. By doing so, Fourier-Net removed the need of progressively decoding the displacement/velocity field from the latent features learned from the encoder. Such a decoder is thus capable of reducing the computational cost and improving inference speed. However, Fourier-Net still takes the original full-resolution image pairs as input. For 3D images, the encoder of Fourier-Net involves multiple 3D convolutional layers, which are computationally expensive operations, particularly in the shallow layers. To further reduce the computational cost and memory usage, we propose Fourier-Net+ by embedding a model-driven encoding layer before the contracting convolutional layers.

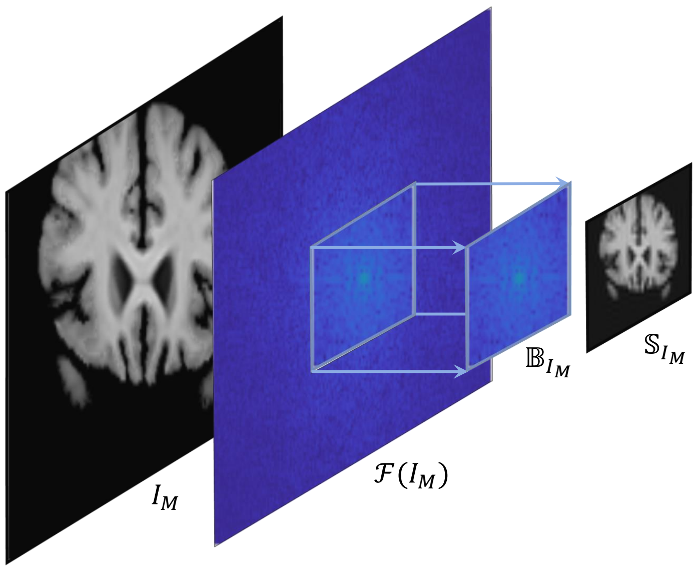


Figure 6.5: Instead of using the original full-resolution images I_M and I_F as input to our network, we use the band-limited representation of the images $\mathbb{S}_{I_M}$ and $\mathbb{S}_{I_F}$.

Model-driven encoder: It is more efficient for learning if we feed our encoder a band-limited representation of input images, which we term band-limited images in line with band-limited displacements. With this consequently much lighter convolutional network, we are able to estimate their band-limited displacements from these band-limited images and further reduce computational costs. In Fourier-Net+, the input images I_M and I_F are first mapped into the frequency domain using a DFT layer, forming $\mathcal{F}(I_M)$ and $\mathcal{F}(I_F)$. We then perform center-cropping on the low-frequency regions, forming $\mathbb{B}_{I_M}$ and $\mathbb{B}_{I_F}$, which are transformed to the spatial domain using iDFT. The real numbers are then taken, and the resulting spatial patches $\mathbb{S}_{I_M}$ and $\mathbb{S}_{I_F}$ are the band-limited images, as illustrated in Figures 6.5. Once we have the band-limited images, we only need a small $\boldsymbol{CNN}$ parameterized by Θ^2 to estimate their band-limited displacements, i.e., $\mathbb{S}_\phi = \boldsymbol{CNN}(\mathbb{S}_{I_M}, \mathbb{S}_{I_F}; \Theta^2)$. This encoder removes several convolutional layers in the contracting path of the Fourier-Net encoder, which we expect to further accelerate the registration process and reduce the memory footprint of Fourier-Net.

The architecture of Fourier-Net+ is shown in Fig. 6.6. Except for the incorporation of a

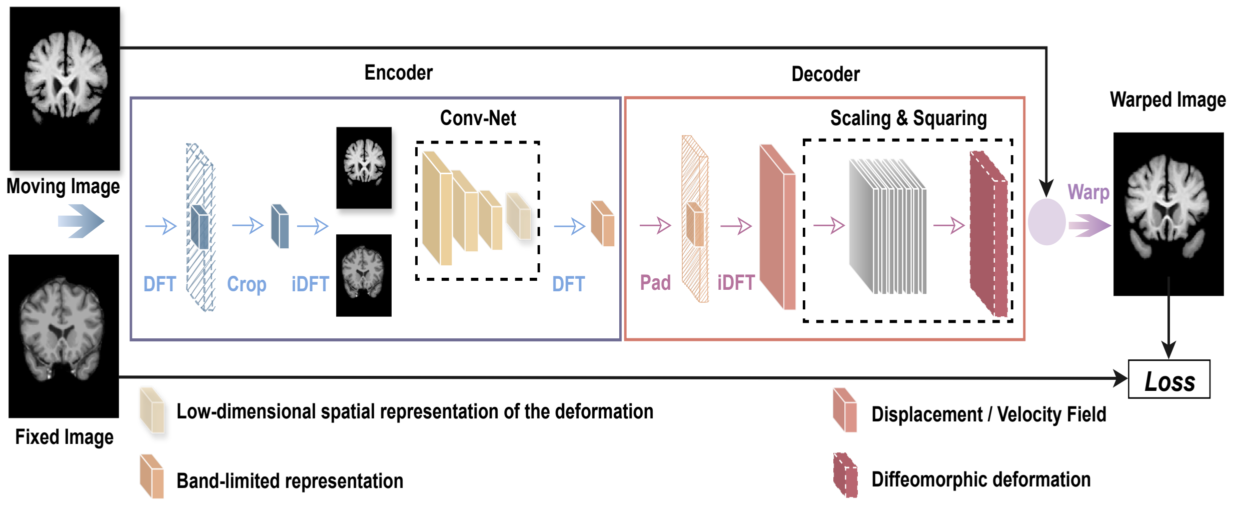

Encoder
Decoder
Conv-Net
Scaling & Squaring
Warped Image
Moving Image
Warp
DFT
Crop
iDFT
DFT
Pad
iDFT
Fixed Image
Loss
Low-dimensional spatial representation of the deformation
Displacement / Velocity Field
Band-limited representation
Diffeomorphic deformation

Figure 6.6: Architecture of Fourier-Net+. The shallow layers of the contracting path in Fourier-Net are replaced with the embedded DFT, Crop, and iDFT layers. This model-driven encoder directly reduces the resolution of the input images.

model-driven encoding layer, the remaining parts of Fourier-Net+ are exactly the same as Fourier-Net. By adding the scaling and squaring layers at the end of Fourier-Net+, we get its diffeomorphic version, which we term Diff-Fourier-Net+. An important note here is that the warping layer in Fourier-Net+ or Diff-Fourier-Net+ warps the originally sized images instead of the band-limited images in order to compute the final loss.

6.3.3 Cascaded Fourier-Net+

Due to the band-limited representation of both images and deformations, Fourier-Net+ is lighter than Fourier-Net in terms of the number of parameters and computations. However, such a light network may face limitations in accurately capturing complex deformations (e.g., in brain images). To tackle this problem, we propose a cascaded version of Fourier-Net+, which is illustrated in Fig. 6.7. For the diffeomorphic version of this network, we impose the scaling and squaring layers after the last cascade. The loss function is also used after the

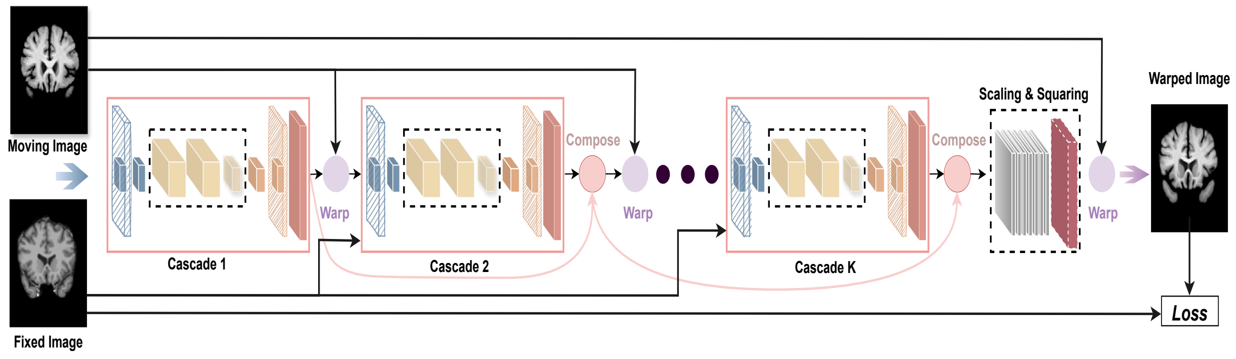


Figure 6.7: Architecture of Cascaded Fourier-Net+ with K cascades ($K\times$Fourier-Net+). Each cascade contains a single Fourier-Net+, where the final deformation is composed of the deformations from each individual cascade. The network is trained end-to-end using only a single loss function following the final cascade or scaling and squaring layer if present.

last cascade. For terminology, $K\times$Fourier-Net+ means that we use a Fourier-Net+ with K cascades, and Diff-$K\times$Fourier-Net+ is the diffeomorphic version of $K\times$Fourier-Net+. This Cascaded Fourier-Net+ is parameterized by Θ^3, and its network parameters are not shared across different cascades.

Technically, in the first cascade of $K\times$Fourier-Net+, the input is the moving image I_M and the fixed image I_F. From the second cascade onward, the input is $I_M^{w(k)}$ (a warped version of I_M, $k \in [1, K]$) and I_F. The process iterates until Cascade k, where the input is $I_M^{w(k-1)}$ and I_F and the output $\delta\phi^{(k)}$. Specifically, as in [166], $I_M^{w(k)}$ is defined as

$$I_M^{w(k)} = ((((I_M \circ \delta\phi^{(1)}) \circ \delta\phi^{(2)}) \circ \cdots) \circ \delta\phi^{(k-1)}) \circ \delta\phi^{(k)}, \tag{6.5}$$

or equivalently

$$I_M^{w(k)} = I_M \circ \boldsymbol{\phi}^{(k)}, \tag{6.6}$$

where $\boldsymbol{\phi}^{(k)}$ is composed by the outputs from Cascade 1 to k,

$$\boldsymbol{\phi}^{(k)} = \delta\phi^{(1)} \circ \delta\phi^{(2)} \circ \cdots \circ \delta\phi^{(k-1)} \circ \delta\phi^{(k)}. \tag{6.7}$$

In the case of $k = K$, $\boldsymbol{\phi}^{(K)}$ denotes the final output deformation used to warp the original moving image I_M in the computation of the loss.

6.3.4 Network Architectures and Loss Functions

Network architectures: For all networks using a U-Net backbone in our experiments, we employed the architecture defined in Fig. 6.8, which is similar to that used in [74, 110, 161]. In this network, there are five convolutional blocks in the contracting path and five convolutional blocks in the expansive path. Specifically, given a pair of moving and fixed images in 3D, each with a size of $D \times H \times W$, the resolution of features in the network flows as $(C, D, H, W) \rightarrow (2C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2}) \rightarrow (4C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4}) \rightarrow (8C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8}) \rightarrow (16C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$ in the contracting path. The features in the expansive path were progressively upsampled to $(3, D, H, W)$, which is the size of the final displacement/velocity field.

The network of Fourier-Net given in Fig. 6.8 was modified from the U-Net backbone by discarding several blocks in the expansive path. In the two variants of Fourier-Net, we used fewer blocks in the expansive path, which leads to a low-resolution output and avoids the convolutional computations in higher-dimensional space. A smaller size of the output rapidly decreases the model parameters and speeds up training and inference time, but may lead to lower performance. A larger size of the output will retain registration accuracy but eliminate the efficiency advantage of our methods. We noticed that different datasets favor different sizes and investigated in our experiments two different sizes for the output layer, i.e., $(3, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$ and $(3, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$. In Fourier-Net+, we instead used small-sized band-limited images as input, allowing us to remove some convolutional blocks in the contracting path to further save the computational cost. In our experiments we studied two different sizes of such band-limited images, i.e., $(\frac{D}{2}, \frac{H}{2}, \frac{W}{2})$ and $(\frac{D}{4}, \frac{H}{4}, \frac{W}{4})$, which in combination with the two sizes of band-limited displacements results in a total of four Fourier-Net+ variants.

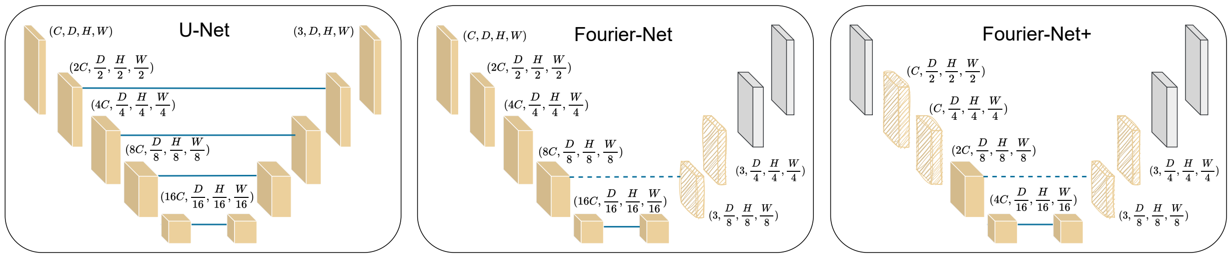

U-Net
(C, D, H, W)
$(2C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2})$
$(4C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(8C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$
$(16C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$
$(3, D, H, W)$
Fourier-Net
(C, D, H, W)
$(2C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2})$
$(4C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(8C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$
$(16C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$
$(3, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(3, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$
Fourier-Net+
$(C, \frac{D}{2}, \frac{H}{2}, \frac{W}{2})$
$(C, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(2C, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$
$(4C, \frac{D}{16}, \frac{H}{16}, \frac{W}{16})$
$(3, \frac{D}{4}, \frac{H}{4}, \frac{W}{4})$
$(3, \frac{D}{8}, \frac{H}{8}, \frac{W}{8})$

Figure 6.8: Comparison between U-Net, Fourier-Net, and Fourier-Net+ used in our experiments. Here, (C, D, H, W) represents the channels, depth, height, and width of a block's output. Fourier-Net is based on a U-Net, where grey blocks in the decoder are removed. The shaded blocks are optionally used to control the resolution of the band-limited deformation. Fourier-Net+ additionally forgoes the first block of the U-Net, with the following two optional blocks corresponding to the choice of band-limited image resolution. Optimal choices for band-limited images and displacements were determined through our experimentation.

Loss functions: For Fourier-Net, Fourier-Net+, and $K\times$Fourier-Net+, the final output is the full-resolution displacement field $\boldsymbol{\phi}$ (in $K\times$Fourier-Net+ we let $\boldsymbol{\phi} = \boldsymbol{\phi}^K$). We define an unsupervised training loss

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(I_{M_i} \circ (\boldsymbol{\phi}_i(\boldsymbol{\Theta}) + \mathrm{Id}) - I_{F_i}) + \frac{\lambda}{N}\sum_{i=1}^{N} \|\nabla \boldsymbol{\phi}_i(\boldsymbol{\Theta})\|_2^2. \tag{6.8}$$

For all diffeomorphic versions of Fourier-Net, Fourier-Net+, and $K\times$Fourier-Net+, the training loss $\mathcal{L}(\boldsymbol{\Theta})$ is defined as

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(I_{M_i} \circ Exp(\boldsymbol{v}_i(\boldsymbol{\Theta})) - I_{F_i}) + \frac{\lambda}{N}\sum_{i=1}^{N} \|\nabla \boldsymbol{v}_i(\boldsymbol{\Theta})\|_2^2, \tag{6.9}$$

where N is the number of training pairs, Id is the identity grid, $\circ$ is the warping operator, and ∇ is the first order gradient. The first term can be either SSD, NCC, or NMI, which we clarify in our experiments. The second term defines the smoothness of displacement fields, and λ is a hyperparameter balancing the two terms. Depending the network architecture, $\boldsymbol{\Theta}$ is either Θ^1, Θ^2, or Θ^3 (see Sec. 6.3.1, 6.3.2, and 6.3.3 for details).

6.4 Experiments

In this section, we provide the used datasets and experimental details. We then perform ablation and parameter studies to demonstrate the utility of our methods. We finally compare our method with a range of state-of-the-art methods across different registration tasks.

6.4.1 Datasets

OASIS-1 dataset [102] is employed to perform inter-subject registration. Similar to Section 4.2.4, 414 2D 160×192 slices and masks (containing 24 automated labels) extracted from their corresponding 3D $160 \times 192 \times 224$ volumes were paired up to 40200, 22, and 400 image pairs for training, validation, and testing, respectively.

IXI dataset[1] [21], similar to Section 4.2.4, is employed to perform atlas-based brain registration. The performance of this task was evaluated with 30 labeled anatomical structures.

3DCMR dataset [37, 75, 140] is also employed as in Section 3.3.4. The Dice and HDist between the warped ES mask and the ED mask were measured.

6.4.2 Implementation Details

The U-Net backbone used in experiments is given in Fig. 6.8. There were 5 blocks in both the contracting and expansive paths. In the encoder, the first block directly encoded the input image pair to C feature maps, each with a size of $D \times H \times W$. For the remaining 4 blocks, each contained 2 sequential convolutional layers, where the first layer maintained the same spatial resolution as its input, and the second layer performed a down-sampling with a stride of 2 and

[1]https://brain-development.org/ixi-dataset/

doubled the number of feature channels. In the decoder, each of the first 4 blocks contained a fractionally-strided convolution layer followed by 2 sequential convolutional layers, where the fractionally-strided convolution layer performed an up-sampling with a stride of 2, and the convolutional layers halved the number of feature channels. The output from the last block was the final displacement field. The kernel size in all blocks was $3 \times 3 \times 3$, and each convolution was followed by a PReLU activation except the last output layer, which did not have any activation function.

We implemented all our proposed networks (see Fig. 6.8 middle and right) using PyTorch, where training is optimized using Adam with a batch size of 1 and a learning rate of 0.0001. To adapt to 2D images, 3D kernels were changed to 2D, each with a size of 3×3. For training in both 2D and 3D, we tuned built-in hyper-parameters on a held-out validation set. In terms of loss functions, we used MSE to train all Fourier-Net variants on OASIS-1 for 10 epochs, and the optimal performance was achieved with $\lambda = 0.01$ for all our networks. On IXI, we trained Fourier-Net, Diff-Fourier-Net, Fourier-Net+, and K×Fourier-Net+ with the NCC loss for 1000 epochs with $\lambda = 5$. In contrast, K×Fourier-Net+ and Diff-K×Fourier-Net+ are trained optimally with $\lambda = 2$. On 3DCMR, we used MSE to train all Fourier-Net variants for 1000 epochs with $\lambda = 0.001$.

6.4.3 Ablation Studies

In this section, we investigated whether the proposed modules in Fourier-Net and its variants were effective, all experiments undertaken were conducted on OASIS-1.

Impact of embedding a DFT layer: In Table 6.1, we show the necessity of embedding a DFT layer at the end of the encoder for Fourier-Net (see Fig. 6.3). Without this layer, such an encoder would be purely a CNN that has to learn complex-valued Fourier coefficients

Table 6.1: Comparisons between Fourier-Net and Fourier-Net without an embedded DFT in the encoder. 'Output' denotes the output resolution of the ***CNN*** encoder. DFT indicates embedding the DFT layer in the encoder. Multiple metrics such as Dice, $|J|_{<0}\%$, the number of mult-adds (in millions) required for inference, and the memory footprint in Megabyte (MB) required for one forward/backward pass are reported.

DFT	Output	Dice↑	$\|J\|_{<0}\%$	Mult-adds	Memory
✗	20×24	0.664±0.040	0.159±0.207	890.55	33.53
✓	20×24	0.732±0.042	0.434±0.355	679.19	31.18
✗	40×48	0.675±0.038	0.279±0.257	1310.0	42.95
✓	40×48	0.756±0.039	0.753±0.408	888.25	35.89

from the spatial image pairs. This setup is similar to DeepFlash [150], where two encoders were used to learn the real and imaginary parts of these complex coefficients, respectively. As reported in Table 6.1, with this DFT layer the registration Dice score was shown to improve from 0.664 to 0.732 (6.8%↑) and from 0.675 to 0.756 (8.1%↑) for the output sizes of 20×24 and 40×48, respectively. On the other hand, due to the dual encoders, the network required more mult-adds and memory footprint to learn the band-limited displacement field. The improvements to both computational efficiency and performance indicate the necessity of using such a DFT layer in our Fourier-Net.

Impact of using a band-limited representation: The necessity of learning a band-limited displacement field might be questioned when one could simply estimate a low-resolution displacement field and then directly up-sample to a full-resolution one using linear interpolation. In Table 6.2, we performed such an experiment by replacing the DFT layer and the decoder in Fourier-Net with a simple bilinear interpolation (termed as Bilinear-Net). We observed that in terms of Dice, Bilinear-Net was 1.3% and 1% lower than our Fourier-Net for the output size of 20×24 and 40×48, respectively. This experiment showed that compared

Table 6.2: Comparisons between band-limited representations of image and displacements versus up- and down-sampling via bilinear-interpolation. 'Input' denotes the input resolution of the $\boldsymbol{CNN}$. $\mathbb{B}_{I_M,I_F}$ indicates the use of the band-limited image or bilinear-downsampled image. $\mathbb{B}_{\boldsymbol{\phi}}$ indicates the use of band-limited displacement or bilinear-upsampled deformation.

Methods	$\mathbb{B}_{I_M,I_F}$	$\mathbb{B}_{\boldsymbol{\phi}}$	Input	Output	Dice ↑
Bilinear-Net	–	✗	160×192	20×24	0.719±0.045
Fourier-Net	–	✓	160×192	20×24	0.732±0.042
Bilinear-Net	–	✗	160×192	40×48	0.746±0.041
Fourier-Net	–	✓	160×192	40×48	0.756±0.039
Bilinear-Net+	✗	✗	40×48	40×48	0.696±0.040
Bilinear-Net+	✗	✓	40×48	40×48	0.695±0.040
Fourier-Net+	✓	✓	40×48	40×48	0.717±0.042
Bilinear-Net+	✗	✗	80×96	40×48	0.725±0.040
Bilinear-Net+	✗	✓	80×96	40×48	0.726±0.041
Fourier-Net+	✓	✓	80×96	40×48	0.738±0.041

to the low-resolution directly down-sampled displacement field, it was more effective to learn the band-limited representation of the displacement field. As an additional experiment to demonstrate the utility of band-limited images, we evaluated the performance of Fourier-Net+ (see Fig. 6.6) against two variants of Bilinear-Net+: one used bilinear down-sampled images as input and estimated a down-sampled displacement field, and the other one used bilinear down-sampled images as input and estimated a band-limited displacement field. We found that in terms of Dice the two Bilinear-Net+ variants achieved similar results but both are inferior to Fourier-Net+ with a clear performance gap, i.e., 2.2%↓ on resolution 20×24 and 1.2%↓ on resolution 40×48.

Diffeomorphisms: In Table 6.3, we compared the performance of Fourier-Net and

Table 6.3: Comparisions between variants of Fourier-Net and Fourier-Net+, alongside their diffeomorphic counterparts which utilize the scaling and squaring (SS) layer.

Methods	SS	Input	Output	Dice ↑	$\|J\|_{<0}\%$
Fourier-Net	✗	160×192	20×24	0.732±0.042	0.434±0.355
Diff-Fourier-Net	✓	160×192	20×24	0.735±0.037	0.0±0.0
Fourier-Net	✗	160×192	40×48	0.756±0.039	0.753±0.408
Diff-Fourier-Net	✓	160×192	40×48	0.756±0.037	<0.0001
Fourier-Net+	✗	40×48	40×48	0.717±0.042	0.400±0.302
Diff-Fourier-Net+	✓	40×48	40×48	0.722±0.038	0.0±0.0
Fourier-Net+	✗	80×96	40×48	0.738±0.041	0.674±0.377
Diff-Fourier-Net+	✓	80×96	40×48	0.740±0.039	0.0±0.0

Fourier-Net+ and their diffeomorphic counterparts. The scaling and squaring (SS) layers encouraged diffeomorphisms for the estimated deformation, resulting in a lower $|J|_{<0}\%$. Besides the influence on negative Jacobians, it was notable that the incorporation of such layers slightly fluctuated the Dice score of different models.

6.4.4 Parameter Studies

In this section, we investigate how different parameter combinations affect the registration performance of our proposed networks using both the 2D OASIS-1 and 3D IXI datasets.

Resolution of band-limited displacement fields: For the 2D OASIS-1 experiments in Table 6.4, we studied Fourier-Net with two resolutions (i.e., 20×24 and 40×48) of the predicted band-limited displacement field, which were $\frac{1}{8} \times \frac{1}{8}$ and $\frac{1}{4} \times \frac{1}{4}$ of the original image resolution (160×192), respectively. It can be seen that resolution 40×48 improved the Dice score by 2.4% compared to resolution 20×24 (0.732 vs 0.756), with an increase in mult-adds

Table 6.4: Comparisons between U-Net backbone and variants of Fourier-Net and Fourier-Net+ on OASIS-1 and IXI. 'Input' and 'Output' correspond to the input and output resolution of the $\boldsymbol{CNN}$, where the value represents the down-sampling rate on each of the dimensions. For example, 1/2 will change the resolution to $\frac{H}{2} \times \frac{W}{2} \times \frac{D}{2}$. K denotes the number of cascades in a Cascaded Fourier-Net+. Multiple metrics such as the Dice, $|J|_{<0}\%$, mult-adds in millions (M) or billions (G), and the memory footprint (MB) are reported.

Methods	Input	Output	K	OASIS				IXI			
				Dice ↑	$\lvert J\rvert_{<0}\%$	Mult-adds(M)	Memory	Dice ↑	$\lvert J\rvert_{<0}\%$	Mult-adds (G)	Memory
Fourier-Net	1	1/8	–	0.732±0.042	0.434±0.355	679.19	31.18	0.754±0.135	<0.0001	135.61	4538.28
Diff-Fourier-Net	1	1/8	–	0.735±0.037	0.0±0.0	679.19	31.18	0.755±0.131	0.0±0.0	135.61	4538.28
Fourier-Net	1	1/4	–	0.756±0.039	0.753±0.408	888.25	35.89	0.763±0.129	0.024±0.019	169.07	4802.93
Diff-Fourier-Net	1	1/4	–	0.756±0.037	<0.0001	888.25	35.89	0.761±0.131	0.0±0.0	169.07	4802.93
U-Net	1	1	–	0.766±0.039	0.702±0.343	2190	96.33	0.768±0.127	0.134±0.065	868.37	16717.97
Diff-U-Net	1	1	–	0.762±0.039	<0.0001	2190	96.33	0.765±0.130	0.0±0.0	868.37	16717.97
Fourier-Net+	1/4	1/4	1	0.717±0.042	0.400±0.302	35.84	3.25	0.736±0.138	<0.0001	3.76	142.73
Diff-Fourier-Net+	1/4	1/4	1	0.722±0.038	0.0±0.0	35.84	3.25	0.739±0.134	0.0±0.0	3.76	142.73
Fourier-Net+	1/4	1/4	2	0.742±0.038	0.370±0.274	71.68	6.50	0.753±0.135	0.0±0.0	7.52	285.46
Diff-Fourier-Net+	1/4	1/4	2	0.737±0.038	0.0±0.0	71.68	6.50	0.756±0.130	0.0±0.0	7.52	285.46
Fourier-Net+	1/4	1/4	3	0.743±0.038	0.282±0.244	107.52	9.75	0.756±0.133	∼ 0.0007	11.28	428.19
Diff-Fourier-Net+	1/4	1/4	3	0.737±0.037	<0.0001	107.52	9.75	0.758±0.130	0.0±0.0	11.28	428.19
Fourier-Net+	1/4	1/4	4	0.744±0.038	0.224±0.207	143.36	13.00	0.760±0.128	0.067±0.042	15.04	570.92
Diff-Fourier-Net+	1/4	1/4	4	0.738±0.038	0.0±0.0	143.36	13.00	0.761±0.127	0.0±0.0	15.04	570.92
Fourier-Net+	1/2	1/4	1	0.738±0.041	0.674±0.377	142.66	9.52	0.748±0.131	0.066±0.051	19.30	670.20
Diff-Fourier-Net+	1/2	1/4	1	0.740±0.039	0.0±0.0	142.66	9.52	0.750±0.130	0.0±0.0	19.30	670.20
Fourier-Net+	1/2	1/4	2	0.755±0.038	0.508±0.274	285.32	19.04	0.762±0.133	0.009±0.008	38.60	1340.40
Diff-Fourier-Net+	1/2	1/4	2	0.749±0.039	<0.0001	285.32	19.04	0.762±0.130	0.0±0.0	38.60	1340.40
Fourier-Net+	1/2	1/4	3	0.759±0.039	0.354±0.278	427.98	28.56	0.766±0.131	0.015±0.011	57.90	2010.60
Diff-Fourier-Net+	1/2	1/4	3	0.753±0.039	<0.0001	427.98	28.56	0.765±0.128	0.0±0.0	57.90	2010.60
Fourier-Net+	1/2	1/4	4	0.761±0.039	0.278±0.232	570.64	38.08	0.765±0.128	0.236±0.115	77.204	2680.80
Diff-Fourier-Net+	1/2	1/4	4	0.755±0.039	<0.0001	570.64	38.08	0.765±0.128	0.0±0.0	77.204	2680.80

(from 679.19M to 888.25M) and memory footprint (from 31.18MB to 35.89MB). Using resolution 40×48, the Dice scores of our Fourier-Net and Diff-Fourier-Net were 1% and 0.6% lower than those of the full-resolution U-Net and Diff-U-Net. However, in terms of mult-adds and memory footprint, such two U-Nets were **2.5** and **2.7** times more expensive than our Fourier-Net and Diff-Fourier-Net, respectively. For the 3D IXI experiments (with resolutions of $160 \times 192 \times 224$) in Table 6.4, we also observed that learning a displacement field with a smaller resolution (20×24×28) was less accurate than using a larger one (40×48×56) in terms of their Dice (0.754 vs 0.763). It also can be seen that learning resolution 40×48×56 performed marginally worse than the full-resolution U-Net and Diff-U-Net, with Dice scores only 0.5% and 0.4% lower, respectively. However, in terms of mult-adds and memory footprint, such two U-Nets were **5.1** and **3.5** times more expensive than our Fourier-Net and Diff-Fourier-Net, respectively.

Resolution of band-limited images: In Fourier-Net, we selected the optimal resolutions of the band-limited displacements for OASIS (40×48) and IXI ($40 \times 48 \times 56$). In Fourier-Net+, we also considered the resolution of the band-limited images. In Table 6.4, we experimented on Fourier-Net+ with two resolutions (i.e., 80×96 and 40×48) on 2D OASIS-1. We observed that a larger resolution (80×96) was superior to a smaller one (40×48). Specifically, with 40×48 band-limited images, Fourier-Net+ achieved a Dice score of 0.717, which was 2% lower than the Fourier-Net+ variant with 80×96. On 3D IXI, Fourier-Net+ achieved a Dice score of 0.748 using resolution 80×96×112, which was 1.2% higher than that of Fourier-Net+ using resolution 40×48×56.

Impact of cascade number: As can be seen from Table 6.4, although Fourier-Net+ significantly reduced the computational cost and memory footprint, its performance was inferior to Fourier-Net. To overcome this accuracy issue, we proposed a Cascaded Fourier-Net+ by stacking multiple Fourier-Net+. Given significant computational savings in Fourier-Net+, Cascaded Fourier-Net+ still had an efficiency advantage compared to Fourier-Net and U-Net. From Table 6.4, on OASIS-1, we observe that using more cascades indeed improves performance. For both Fourier-Net+ and Diff-Fourier-Net+ (with resolution 80×96 and 40×48 for input and output respectively), increasing the number of cascades k from 1 to 4 continuously improved the registration performance, i.e., from 0.738 to 0.761 and from 0.740 to 0.755, respectively. Note that, even with 4 cascades, Fourier-Net+ (570.64M) had 35.76% less mult-adds than Fourier-Net (888.25M), whilst showing performance gains of

0.5% in terms of Dice (0.761 vs 0.756). Our 4×Fourier-Net+ and Diff-4×Fourier-Net+ were on par with the full-resolution U-Net and Diff-U-Net, with Dice scores only 0.5% and 0.7% lower, respectively. However, in terms of mult-adds and memory footprint, such two U-Nets were **3.8** and **2.5** times more expensive than our 4×Fourier-Net+ and Diff-4×Fourier-Net+, respectively. We notice a similar improvement of performance with the addition of cascades in 3D IXI: specifically, 3×Fourier-Net+ achieved a 0.766 Dice score which is only 0.2% lower than that of U-Net. Meanwhile, the Dice score of Diff-3×Fourier-Net+ was the same as Diff-U-Net (0.765), but U-Net and Diff-U-Net had **15** times mult-adds and **8.3** times memory footprint than 3×Fourier-Net+ and Diff-3×Fourier-Net+.

6.4.5 Comparison with the State-of-the-Art

We have so far shown that Fourier-Net can learn a band-limited displacement field to represent the full-resolution deformation with minimal performance loss compared to full-resolution U-Net architectures. We then showed, with Cascaded Fourier-Net+, that learning such a band-limited displacement field from band-limited images can achieve similar performance with Fourier-Net as well as full-resolution U-Net architectures. In this section, we compare our Fourier-Net and its variants with a few state-of-the-art methods on the three datasets. We note that all reported CPU and GPU runtimes were tested on a machine with 128G RAM, 16 3.80GHz Intel(R) Core(TM) i7-9800X CPUs, and 1 NVIDIA Geforce RTX 2080Ti GPU. The computational time includes the cost of loading models and images and was averaged on the whole testing set with batch size 1.

Comparison on Inter-Subject Brain Registration

In Table 6.5, we compared the performance of our Fourier-Net and Fourier-Net+ with Flash [163], DeepFlash [150], and Diff-B-Spline [122] on the challenging task of 2D inter-subject brain registration (OASIS-1). We compiled and ran Flash[2] on CPU, but encountered *segmentation fault* errors with the official GPU version. As such we have not compared GPU inference times of Flash. We reported the performance of Flash on three band-limited resolutions (i.e., 16×16, 20×24, and $40{\times}48$), and we grid-searched its built-in hyper-parameters over 252 different combinations on the whole validation set for each resolution. We also attempted to run the official DeepFlash[3] with supervision from Flash's results. We trained DeepFlash on all 40200 training pairs for up to 1000 epochs with more than 40 different combinations of hyper-parameters. Diff-B-Spline [122] was trained by using its official implementation[4] on all image pairs in the training set. The hyper-parameters were tuned on the held-out validation set, and the highest performing model used MSE data similarity and λ =0.01 smoothness regularisation.

All our variants of Fourier-Net outperformed competing methods in terms of Dice. Compared to Flash using a 40×48 resolution, Diff-Fourier-Net improved Dice by 2.2% and was **5718** times faster on CPU. Although DeepFlash was much faster than Flash, we found it extremely difficult to successfully train a model and would expect its true potential to fall close to that of Flash, in line with their published work. We note that DeepFlash is not an end-to-end method, because its output (band-limited velocity field) requires an additional PDE algorithm to compute the final deformation. As such, the method is slower than other deep learning methods such as our methods or Diff-B-Spline (0.012s per image pair on CPU). Fourier-Net+ and Diff-Fourier-Net, with similar mult-adds and memory footprint as Diff-B-Spline, achieved comparable results to Diff-B-Spline in terms of Dice but were 2 and 1.5 times faster than Diff-B-Spline, respectively. Our highest performing 4×Fourier-Net+ achieved a Dice score of 0.761, which was able to bridge the gap between Fourier-Net+ and Fourier-Net, whilst still achieving very fast runtimes with fewer mult-adds and less memory

[2]https://bitbucket.org/FlashC/flashc/src/master/
[3]https://github.com/jw4hv/deepflash
[4]https://github.com/qiuhuaqi/midir

Table 6.5: Performance comparison between different methods on 2D OASIS-1. We include the patch size of the band-limited displacement/velocity field used by each method, apart from Diff-B-Spline which instead corresponds to the lower dimensional deformation used for cubic B-Spline interpolation.

Methods	Resolution	Dice↑	$\lvert J\rvert_{<0}\%$	Parameters	Mult-adds (M)	Memory (MB)	CPU (s)	GPU (s)
Initial	-	0.544±0.089	-	-	-	-	-	
Flash [163]	16×16	0.702±0.051	0.033±.126	-	-	-	13.699	-
Flash [163]	20×24	0.727±0.046	0.205±.279	-	-	-	22.575	-
Flash [163]	40×48	0.734±0.045	0.049±.080	-	-	-	85.773	-
DeepFlash [150]	16×16	0.615±0.055	0.0±0.0	-	-	-	0.487	-
DeepFlash [150]	20×24	0.597±0.066	0.0±0.0	-	-	-	0.617	-
Diff-B-Spline [122]	20×24	0.710±0.041	0.014±0.072	70,226	86.16	6.58	0.012	0.015
Diff-B-Spline [122]	40×48	0.737±0.038	0.015±0.069	88,690	139.40	7.49	0.012	0.015
Fourier-Net	40×48	0.756±0.039	0.753±0.408	1,427,376	888.25	35.89	0.011	0.015
Diff-Fourier-Net	40×48	0.756±0.037	<0.0001	1,427,376	888.25	35.89	0.015	0.015
Fourier-Net+	40×48	0.738±0.041	0.674±0.377	300,477	142.66	9.52	0.006	0.014
Diff-Fourier-Net+	40×48	0.740±0.039	0.0±0.0	300,477	142.66	9.52	0.009	0.014
4×Fourier-Net+	40×48	0.761±0.039	0.278±0.232	1,201,908	570.64	38.08	0.021	0.017
Diff-4×Fourier-Net+	40×48	0.755±0.039	<0.0001	1,201,908	570.64	38.08	0.025	0.017

footprint. We also listed the values of $|J|_{<0}\%$ for all compared methods in Table 6.5. Though both Flash and Diff-B-Spline are diffeomorphic approaches, neither of them produced perfect diffeomorphic deformations on this dataset. Diff-Fourier-Net and Diff-Fourier-Net+ however generated nearly zero negative Jacobian determinants.

Comparison on Atlas-Based Brain Registration

In Table 6.6, we compared our Fourier-Net and its variants with iterative methods such as ANTs SyN [7] and Flash [163], as well as deep learning methods such as TransMorph [21] and LKU-Net [74]. To guarantee a fair comparison between different methods, we used the IXI dataset with the exact same pre-processing steps and testing protocol as [21, 74]. Because of this, we directly took relevant results from the original papers [21] and [74], and

Table 6.6: Performance comparison between different methods on the 3D IXI dataset.

Methods	Dice↑	$\lvert J \rvert_{<0}\%$	Parameters	Mult-adds (G)	Memory	CPU (s)	GPU (s)
Affine*	0.386±0.195	-	-	-	-	-	
SyN* [7]	0.645±0.152	<0.0001	-	-	-	-	
NiftyReg* [107]	0.645±0.167	<0.0001	-	-	-	-	
LDDMM* [13]	0.680±0.135	<0.0001	-	-	-	-	
Flash [163]	0.692±0.140	0.0±0.0	-	-	-	1760	-
deedsBCV* [61]	0.733±0.126	0.147±0.050	-	-	-	-	
VoxelMorph-1* [10]	0.728±0.129	1.590±0.339	274,387	304.05	2999.88	2.075	0.398
VoxelMorph-2* [10]	0.732±0.123	1.522±0.336	301,411	398.81	3892.38	2.321	0.408
Diff-VoxelMorph* [10]	0.580±0.165	<0.0001	307,878	89.67	1464.26	1.422	0.398
Diff-B-Spline* [122]	0.742±0.128	<0.0001	266,387	47.05	1233.23	5.649	0.378
TransMorph* [21]	0.754±0.124	1.579±0.328	46,771,251	657.64	4090.31	4.094	0.516
Diff-TransMorph* [21]	0.594±0.163	<0.0001	46,557,414	252.61	1033.18	2.797	0.419
B-Spline-TransMorph* [21]	0.761±0.122	<0.0001	46,806,307	425.95	1563.41	7.582	0.417
LKU-Net* [74]	0.765±0.129	0.109±0.054	2,086,342	272.09	8713.36	2.304	0.398
Diff-LKU-Net* [74]	0.760±0.132	0.0±0.0	2,086,342	272.09	8713.36	5.914	0.390
Fourier-Net	0.763±0.129	0.024±0.019	4,198,352	169.07	4802.93	1.029	0.384
Diff-Fourier-Net	0.761±0.131	0.0±0.0	4,198,352	169.07	4802.93	4.668	0.384
Fourier-Net+	0.748±0.131	0.066±0.051	880,109	19.30	670.20	0.455	0.381
Diff-Fourier-Net+	0.750±0.130	0.0±0.0	880,109	19.30	670.20	4.035	0.378
3×Fourier-Net+	0.766±0.131	0.015±0.011	2,640,327	57.90	2010.60	2.331	0.387
Diff-3×Fourier-Net+	0.765±0.128	0.0±0.0	2,640,327	57.90	2010.60	4.981	0.396

such results are labeled with $*$ in Table 6.6. Note that the runtimes of all compared methods were computed on our end using the same machine. For Flash [163], we grid-searched 200 combinations of hyper-parameters using only 5 randomly selected pairs from the validation set, because the registration process of Flash takes more than 30 minutes on the CPU for each input image pair.

Our Fourier-Net achieved a 0.763 Dice score which is competitive with top-performing learning-based methods including Transmorph and LKU-Net, whilst reducing CPU inference time to close to a second (1.029s). Fourier-Net+ traded a small performance drop for subsecond CPU runtimes and had the lowest memory footprint and number of mult-adds across all methods. Specifically, Fourier-Net+ achieved a 0.748 Dice score with only 19.30G mult-adds, 670.2MB memory footprint, and 0.455s per pair speed. Compared to VoxelMorph-1 and VoxelMorph-2, Fourier-Net+ improved Dice by 2% and 1.6% and was **4.6** and **4.9** times faster, whilst using only 6.35% and 4.84% their mult-adds, and 22.34% and 17.22% their memory footprint. Our cascaded 3×Fourier-Net+ and Diff-3×Fourier-Net+ equaled the performance of top state-of-the-art methods whilst retaining a similar inference time compared to those diffeomorphic and non-diffeomorphic learning-based methods. Amongst iterative diffeomorphic methods, Flash achieved the highest Dice score of 0.692, but it needed 1,760s to register an image pair on average. Our Diff-3×Fourier-Net+ achieved a Dice score of 0.765, with 57.90G mult-adds and 4.981s runtimes, outperforming Diff-B-Spline, B-Spline-TransMorph, and Diff-LKU-Net in terms of Dice, mult-adds, and CPU runtimes. We note that those diffeomorphic methods based on dense SVF were around 3.6 seconds slower than their non-diffeomorphic versions (see Diff-LKU-Net vs LKU-Net and Diff-Fourier-Net vs Fourier-Net), with the extra 3.6 seconds accounting for the computation of 7 scaling and squaring layers. Such dense SVF-based diffeomorphic methods were however faster than B-Spline methods such as Diff-B-Spline and B-Spline-TransMorph because these methods require additional transposed convolutional layers in order to first recover a full-resolution SVF, which costs extra time.

Fig. 6.9 shows that whilst Flash's deformation grids have no foldings, it over-smooths the displacement field resulting in a less accurate warping. Fig. 6.9 (last row) shows that only Flash and Fourier-Net produce strictly band-limited Fourier coefficients. The deformation of Diff-Fourier-Net, 3×Fourier-Net+, and Diff-3×Fourier-Net+ are no longer band-limited due to the use of the scaling and squaring layers and cascades. We additionally plot the performance of different methods on 7 representative brain structures with the respective boxplot shown in Figure 6.10, including the brain stem, left/right cerebellum white matter,

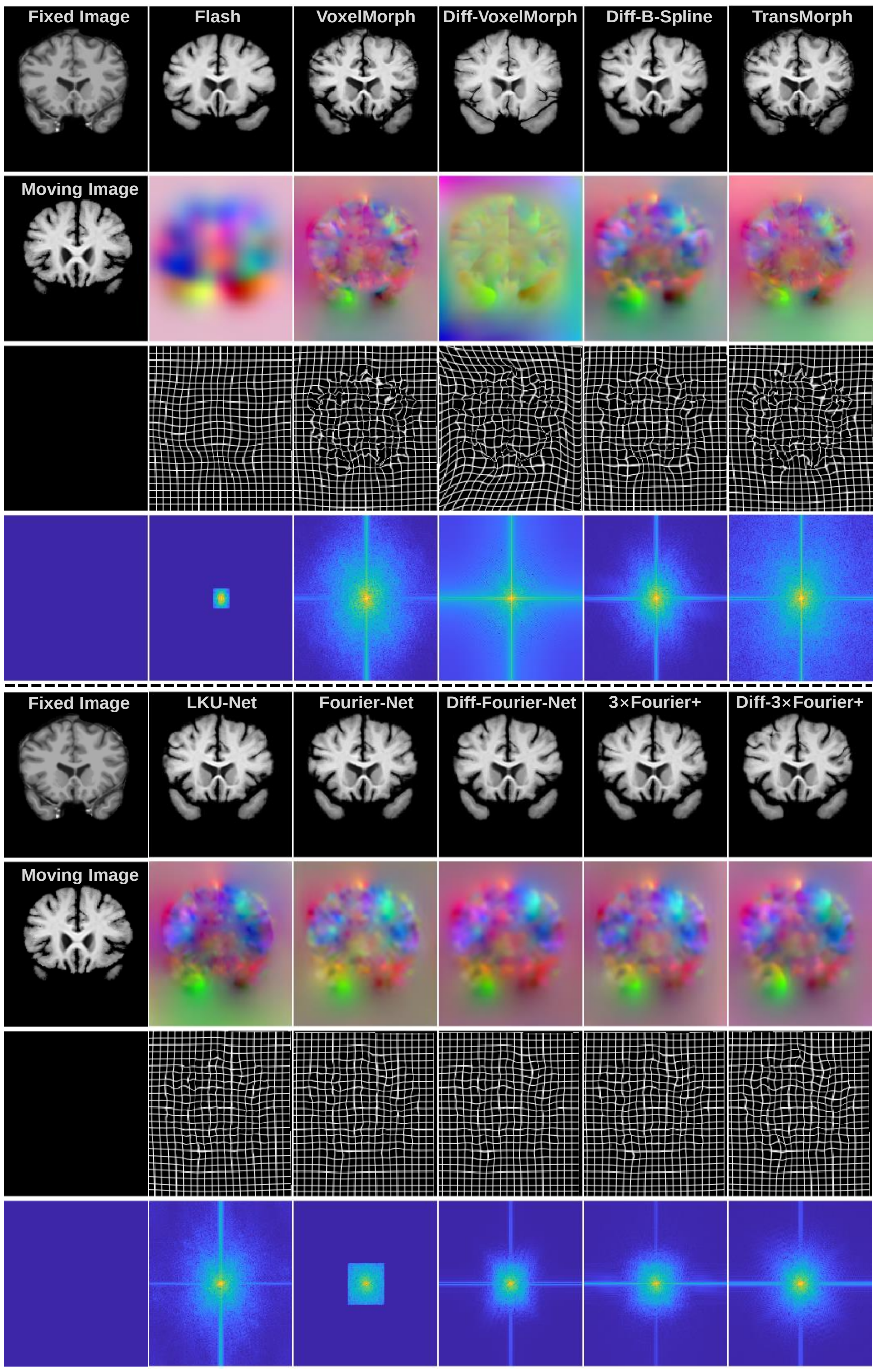


Figure 6.9: Visual comparison between different methods on the 3D IXI dataset. The 1st column displays a fixed image, a moving image, and two placeholders. From top to bottom, the rows excluding the 1st column are as follows: warped moving images, displacement fields as RGB images, deformation grids, and displacement fields after DFT.

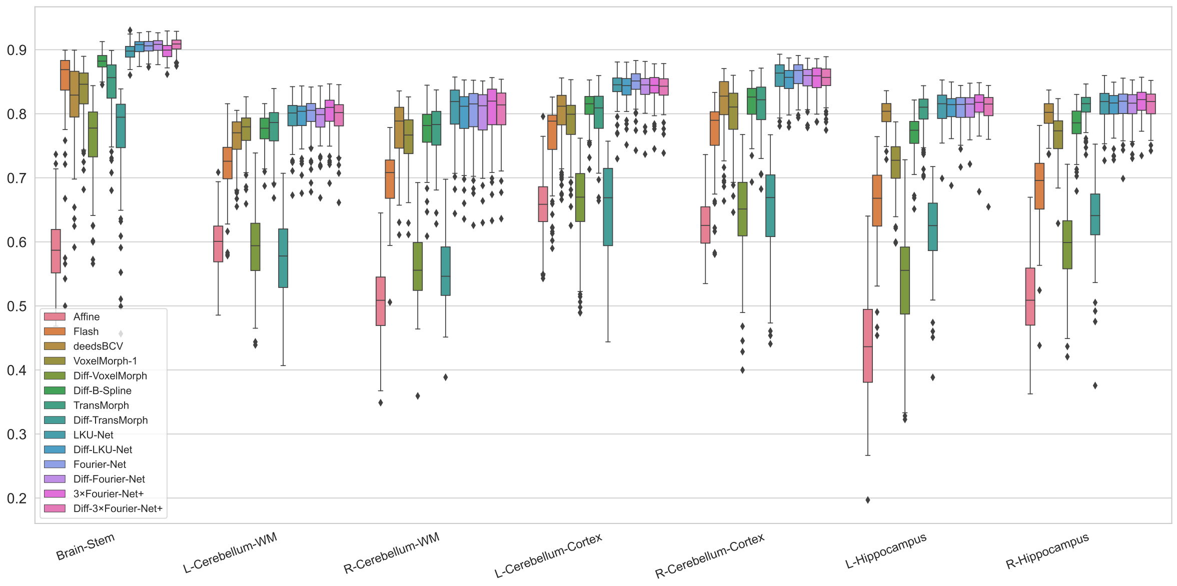


Figure 6.10: Dice distributions across the various state-of-the-art methods on different brain structures seen in the 3D IXI dataset, including the brain stem, left/right cerebellum white matter, left/right cerebellum-cortex, and left/right hippocampus.

left/right cerebellum-cortex, and left/right hippocampus. As can be seen, our Fourier-Net variants consistently perform well over all classes.

Comparison on 3D Cardiac Motion Estimation

We conducted a final experiment on the 3DCMR dataset to ensure we have validated the performance of our models beyond only the task of brain registration. In Table 6.7, we compared the registration performance between different methods, including FFD [130], Demons [143], ANTs SyN [7], RC-Net [166], VR-Net [75], TransMorph [21], and LKU-Net [74]. Since we used the same pre-processed data as in Section 5.5.5, the results of these iterative methods such as Demons, FFD, and ANTs SyN are directly adopted.

The detailed parameter settings used for each deep method was described as follows. We note that, if not specified, the model was trained with the NMI data term with the bending energy regularization, where λ was set to 15.0 in default.

- VoxelMorph: We trained VoxelMorph-1 [11] and VoxelMorph-2 [10] . The only difference between them is that VoxelMorph-1 has 8 channels at the first and last convolutional layers, while VoxelMorph-2 has 16 channels at these layers.
- RC-Net [166]: Two variants, namely 3×RC-Net and 4×RC-Net, are trained using 3 and 4 cascades of VoxelMorph-1, respectively.
- SYM-Net [110]: The initial number of kernels was set to 8.
- Diff-B-Spline [122]: The spacing of the B-spline control point grid is set to 8. The model was trained with SSD data term and diffusion regularization with λ being 0.01.
- VR-Net [75]: U-L$_2$-1×1 from Section 5.5.5 is used, in which the data term was set to 'L_1', the number of warping cascades is set to 1, VoxelMorph-1 is used as the backbone, and the number of intensity consistency layers in each cascade is set to 1.
- TransMorph [21]: The official TransMorph code[5] is adopted.
- LKU-Net [74]: The number of channels in the initial layer was set to 8 and the large kernel size was 5×5×5 for both LKU-Net and its diffeomorphic version Diff-LKU-Net.

In Table 6.7, we observe that all methods estimating full-resolution deformations including non-diffeomorphic displacement fields and diffeomorphic velocities were outperformed (in terms of Dice) by the methods estimating a low-dimensional representation of the displacement or velocity field, such as Diff-B-Spline and our Fourier-Net variants. We note that the deformations produced by such methods are inherently very smooth. In contrast to the intricate and detailed deformations required in brain registration tasks, the deformation in the left and right ventricles between ED and ES is also smooth. As Dice and HDist of

[5]https://github.com/junyuchen245/TransMorph_Transformer_for_Medical_Image_Registration

Table 6.7: Performance comparison between different methods on the 3DCMR dataset.

Methods	Dice↑	HDist↓	$\lvert J\rvert_{<0}\%$	Parameters	Mult-adds (G)	Memory	CPU (s)	GPU (s)
Initial	0.493±0.043	8.40±0.89	-	-	-	-	-	
Demons	0.727±0.040	6.74±1.00	0.0±0.0	-	-	-	-	
FFD	0.739±0.047	7.04±1.20	0.667±0.358	-	-	-	-	
ANTs SyN	0.721±0.051	6.98±1.23	1.388±0.467	-	-	-	-	
VoxelMorph-1 [10]	0.762±0.029	6.08±0.79	0.108±0.132	274,387	69.50	685.69	0.459	0.138
VoxelMorph-2 [10]	0.762±0.029	5.98±0.74	0.090±0.119	301,411	91.16	889.69	0.503	0.140
3×RC-Net [166]	0.781±0.030	6.00±0.78	0.001±0.014	823,161	208.5	2057.07	1.513	0.222
4×RC-Net [166]	0.778±0.031	5.96±0.78	0.001±0.014	1,097,548	278.0	2742.76	2.011	0.286
SYM-Net [110]	0.782±0.028	5.73±0.78	0.0±0.0	1,124,448	67.82	1018.31	1.362	0.162
VR-Net [75]	0.772±0.029	5.69±0.75	0.094±0.137	549,854	140.4	1371.38	1.007	0.171
Diff-B-Spline [122]	0.809±0.033	6.04±0.73	0.0±0.0	211,059	5.33	249.71	1.457	0.137
TransMorph [21]	0.770±0.026	6.51±0.99	0.811±0.231	46,771,251	150.39	935.16	1.005	0.169
LKU-Net [74]	0.775±0.030	5.89±0.76	0.071±0.099	2,086,342	62.19	1991.62	0.491	0.141
Diff-LKU-Net [74]	0.785±0.031	5.82±0.77	0.0±0.0	2,086,342	62.19	1991.62	1.279	0.143
Fourier-Net	0.814±0.024	6.00±0.77	2.137±0.658	3,891,533	31.00	1037.32	0.225	0.140
Diff-Fourier-Net	0.827±0.027	6.08±0.85	<0.0003	3,891,533	31.00	1037.32	0.985	0.141
Fourier-Net+	0.803±0.031	6.06±0.72	2.746±1.033	211,081	0.52	19.38	0.111	0.134
Diff-Fourier-Net+	0.828±0.030	5.89±0.81	<0.0002	211,081	0.52	19.38	0.813	0.136
2×Fourier-Net+	0.816±0.027	6.02±0.75	1.455±0.784	422,162	1.04	38.76	0.245	0.135
Diff-2×Fourier-Net+	0.821±0.029	5.76±0.70	0.0±0.0	422,162	1.04	38.76	1.002	0.137

the LV, RV, and Myo are used as the surrogate for registration accuracy in this task, we believe the inherent smoothness of Diff-B-Spline and Fourier-Nets is an advantage in this task. Although Diff-B-Spline was very competitive on this dataset, the highest Dice and the lowest HDist were all achieved by our methods, with our highest performing method (Diff-Fourier-Net+) outperforming Diff-B-Spline by 1.9% in Dice and 0.15mm in HDist. On the other hand, our Fourier-Net+ outperformed TransMorph and LKU-Net, with improvements of 3.3% and 2.8% in terms of Dice. Additionally, our Fourier-Net+ was **9.05** and **4.42** times faster while utilizing only 0.35% and 0.84% of their mult-adds, and 2.07% and 0.97% of their memory usage.

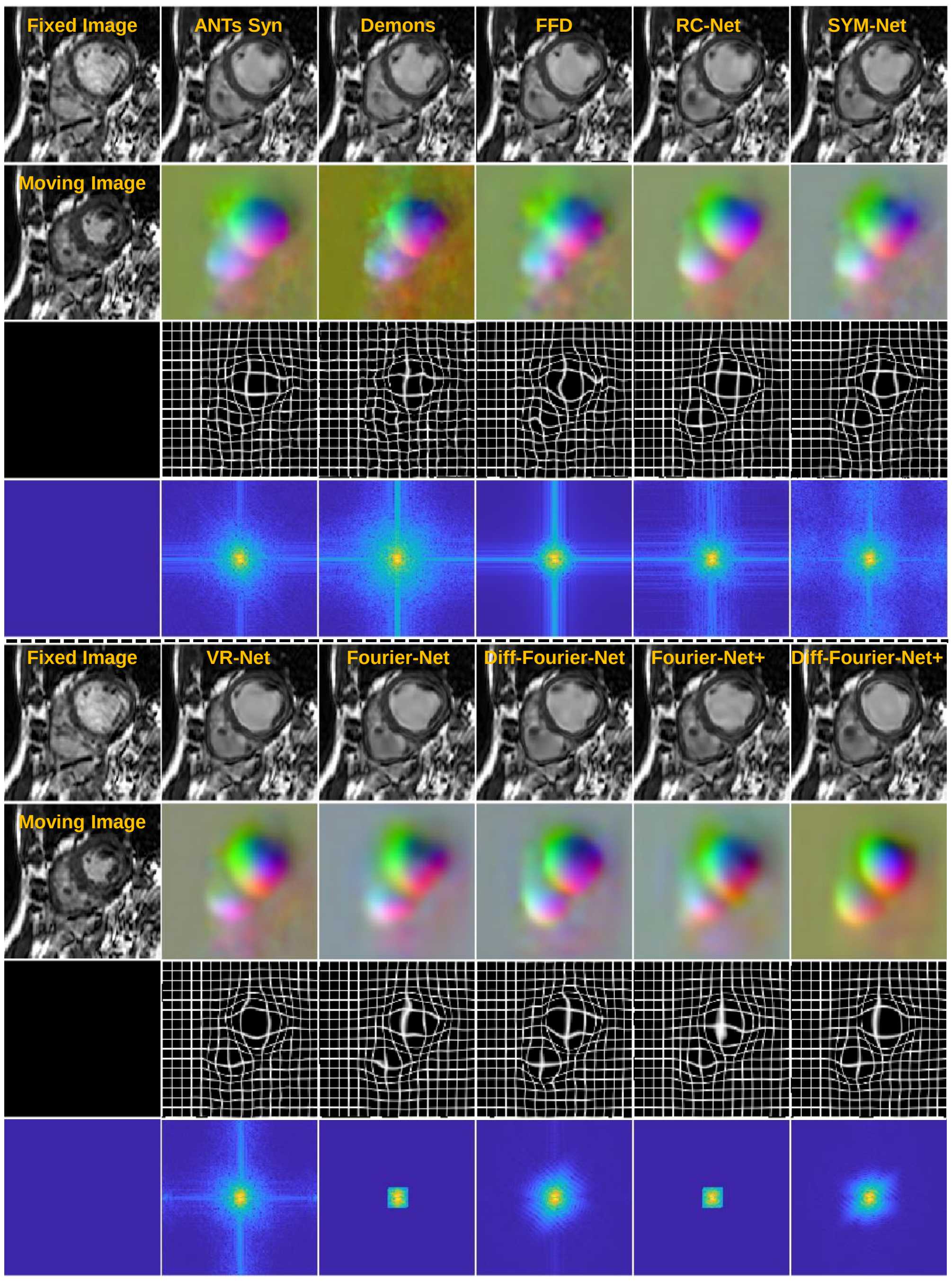


Figure 6.11: Visual comparison between different methods on the 3DCMR dataset. The 1st column displays a fixed image, a moving image, and two placeholders. From top to bottom, the rows excluding the 1st column correspond to the following: warped moving images, displacement fields as RGB images, deformation grids, and displacement fields after DFT.

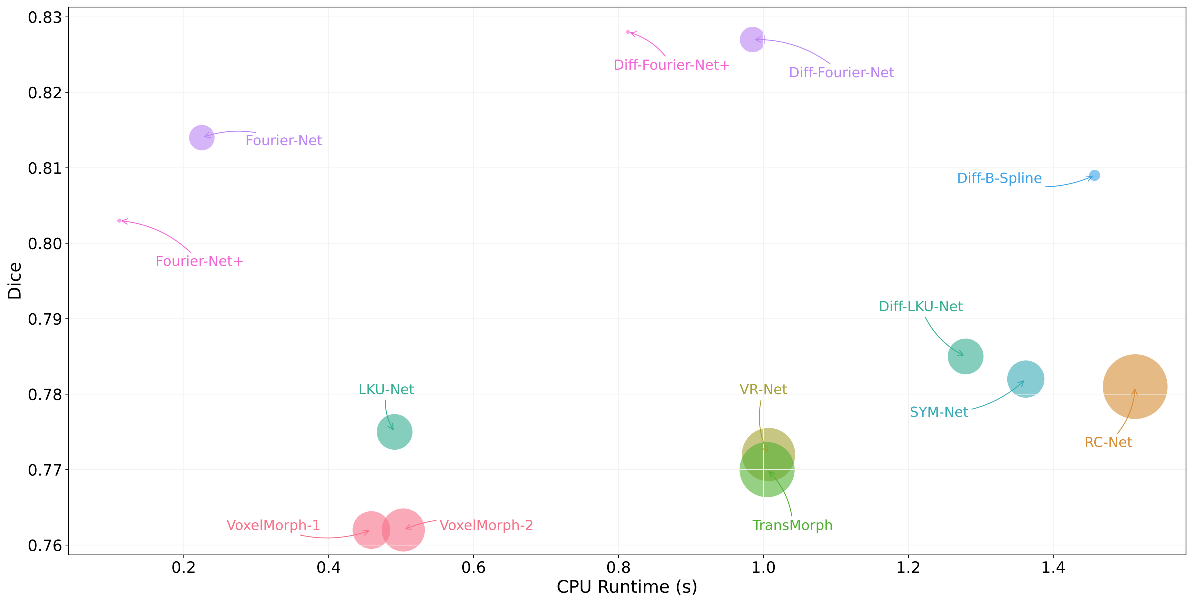


Figure 6.12: Comparisons of the computational cost of different methods: The x-axis and the y-axis denote the runtime in CPU (s) and Dice score. The number of mult-adds is expressed by the area of the circle.

In Fig. 6.11, we plot estimated displacement fields, deformation grids, and warped moving images from each method. We can clearly see that the estimated deformations from our Fourier-Nets are smoother than the competing methods, and Fourier-Net and Fourier-Net+ produce strictly band-limited deformations. In Fig. 6.12, we compared the computational cost of different methods, where the x-axis and y-axis denote the runtime in CPU (s) and Dice score, respectively. The number of mult-adds is expressed by the area of a circle. As can be seen, Fourier-Net, Fourier-Net+, and Cascaded Fourier-Net+ achieved a higher Dice with faster inference speed and fewer computational costs, while their diffeomorphic versions take a while longer and produce slightly better performance.

6.5 Summary

To reduce the computational cost and memory footprint of U-Net style registration networks, we first proposed Fourier-Net to learn the band-limited displacement/velocity field in the Fourier domain. Building upon Fourier-Net and to further boost the registration efficiency, we then proposed Fourier-Net+ and Cascaded Fourier-Net+, aiming to learn the band-limited displacement/velocity field directly from band-limited images. As band-limited images and displacement/velocity fields are of low-resolution representation, our experiments on three datasets showed that Fourier-Net, Fourier-Net+, and Cascaded Fourier-Net+ were significantly more efficient than other state-of-the-art U-Net-based approaches, whilst retaining a comparative performance in terms of registration accuracy.

Specifically, on the 2D OASIS-1 and 3D IXI brain datasets, we showed that Fourier-Net can learn effectively a band-limited displacement field to represent the full-resolution deformation with minimal performance loss as compared to full-resolution U-Net architectures. We then showed with Cascaded Fourier-Net+, that learning such a band-limited displacement field directly from band-limited images was able to achieve similar performance to Fourier-Net and full-resolution U-Net architectures. On the 3DCMR dataset where cardiac motion is intrinsically smooth and relatively simple, our Fourier-Net+ alone performed already very well, with the registration accuracy on par with Fourier-Net, Cascaded Fourier-Net+, and other state-of-the-art methods, but with significantly less computational cost and memory footprint. We also noticed that diffeomorphic Fourier-Net variants were often more accurate than their non-diffeomorphic counterparts in terms of Dice, but were slightly slower in terms of inference speed due to the use of scaling and squaring layers. However, our diffeomorphic methods were still the most efficient approaches when compared to other competing diffeomorphic methods.

Though our proposed Fourier-Net, Fourier-Net+, and Cascaded Fourier-Net+ achieved comparable performance with other state-of-the-art methods, it is notable that Fourier-Net assumes that the displacement or velocity field lacks high-frequency signals. This assumption is valid for most smooth and diffeomorphic deformations, and our experimental results on all three datasets also support this assumption. However, we would obviously expect a performance drop in tasks where this assumption does not hold. In Fourier-Net+, intuitively, one might assume the removal of high-frequency image information within the encoder to be prohibitive in tasks such as brain registration which contain complex structures within images. We showed however that the efficient design of Fourier-Net+ allows the cascaded version of this network to have fewer mult-adds than even Fourier-Net, whilst retaining similar performance despite the removal of high-frequency image information in our encoder.

Chapter Seven

Conclusion

7.1 Findings

From this thesis, we are able to provide these findings below:

- In Section 3.4.1, we highlight that a smaller difference between a warped and fixed pair in intensity does not necessarily mean a better registration.
- In Sections 3.4.1, 4.2.4, and 6.4.5, we highlight that diffeomorphic registration does not necessarily improve the registration accuracy, but it can obtain more plausible deformation fields with fewer foldings.
- In Section 3.4.2, we highlight that the registration performance of a deep registration model depends on the amount of training data. A deep model can achieve higher accuracy when trained with more data. When training data is limited, using an excessively large model may hinder registration accuracy, as indicated in Table 4.1 and Fig. 5.5.
- In Sections 3.4.3 and 4.2.4, we highlight that a larger deep model can improve registration accuracy, albeit with more computational costs.
- In Section 4.2, we highlight that the classical U-Net-based registration model is still competitive to modern transformer-based approaches in terms of registration accuracy.
- In Sections 4.3 and 6.4.4, we highlight that using more cascades can improve the registration performance, but at a significantly higher computational cost.
- In Section 4.4, we highlight that the classical CNN architecture, comprised of plain convolutional and FC layers, is still competitive on parametric registration tasks.
- In Section 5.5.5, we highlight that, by embedding the variational structure into deep networks, the model-driven VR-Net can outperform both iterative optimization-based methods and deep learning-based methods in terms of registration accuracy.

- In Section 5.5.5, we highlight that, in comparison with pure data-driven methods, VR-Net can retain the data efficiency of the iterative variational model (Fig. 5.5) and maintain the fast inference speed (Table 5.6).
- In Section 6.4.5, we highlight that, by embedding (inverse) Fourier transform as prior knowledge within the decoder, Fourier-Net can accurately reconstruct the dense deformation field from the predicted band-limited deformation field.
- In Section 6.4.5, we highlight that, using the model-driven decoder, Fourier-Net can discard the convolutional layers for learning dense full-resolution deformation, therefore, be faster than U-Net style networks in terms of inference speed.
- In Section 6.4.5, we highlight that, by embedding (inverse) Fourier transform in the encoder, Fourier-Net+ can predict band-limited deformation field from band-limited images, further accelerating the registration speed.

7.2 Discussion

Though deep learning-based registration methods provide a promising solution, many challenges stay unsolved in the field of medical image registration, such as defining the optimal objective function for unsupervised deformable registration, eliminating the ambiguity of registration, and measuring the quality of registration.

Objective and how to learn it?

We have proved that the registration performance can be significantly influenced by different configurations of data and regularization terms in Chapter 3, even with the same backbone network, training data, and optimizer. Therefore, we wonder if there can be an automatic algorithm that chooses the optimal objective function given a specific dataset/task.

Automatic learning of regularization terms [2, 112, 152] and the regularization weight λ [67, 109] have been actively studied, while less attention has been paid to the automatic learning of data terms [51]. One of the most straightforward approaches is to combine different data terms (such as SSD, NCC, NMI) and regularization terms (such as diffusion, total variation, and bending energy) all together, forming a multi-level multi-objective optimization problem [93]. However, this approach is eventually an ad hoc combination of hand-crafted data and regularization terms, which may not fully explore data-/task-specific information.

Uncertainty and how to use it?

Due to the ill-posed nature of registration, unique correspondence is often difficult to find. Many deep learning models [21, 31] have introduced uncertainty learning [49, 71] (including both *model uncertainty* and *data uncertainty*) in registration to reduce registration error and boost accuracy.

We believe both model and data uncertainty essentially reflect the ambiguity of registration, therefore, learning uncertainty can provide an additional way to interpret the registration process. Moreover, the learned uncertainty map can also be used as an attention map to better guide registration. However, of note, a dilemma here is that a prediction with lower uncertainty is not necessarily correct, and vice versa.

Quality and how to find it?

The aforementioned two issues of objective function and ambiguity (uncertainty) are essentially caused by the lack of true deformations, but the truth is hard to seek [118]. Therefore, the deep models have to be trained on the surrogate data terms based on intensities, and evaluated on surrogates of anatomical structures, key points, and landmarks, which again may not necessarily gauge the quality of registration [30, 124].

“So we beat on. . . "

Appendix One

Appendix of VR-Net

A.1 Appendix 1

In this section, we propose to derive the solution of $\boldsymbol{u}-$subproblem $(s = 1)$ in Section 5.3.1 using a primal-dual method, originally proposed in [19] for Total Variation denoising [127]. Here we use all notations in 3D only. First, we rewrite the subproblem into its discrete form

$$\arg\min_{\boldsymbol{u}} \|\rho(\boldsymbol{u})\|^1 + \frac{\theta}{2}\|\boldsymbol{v}^k - \boldsymbol{u}\|^2, \tag{A.1}$$

where $\rho(\boldsymbol{u}) = \langle \nabla I_1, \boldsymbol{u} - \boldsymbol{u}^\omega \rangle + I_1 - I_0$. This minimization problem (A.1) can be converted equivalently to a saddle-point problem by writing the first term as a maximization, i.e.,

$$\|\rho(\boldsymbol{u})\|^1 = \max_{\|z\|_\infty \le 1} \langle \rho(\boldsymbol{u}), z \rangle,$$

over the dual variable $z \in \mathbb{R}^{MNH}$ where MNH is the the image size, and $\|z\|_\infty = \max_{i,j,l} \left|z_{i,j,l}\right|$ and $\langle \rho(\boldsymbol{u}), z \rangle = \sum_{i,j,l} (\rho(\boldsymbol{u}))_{i,j,l} z_{i,j,l}$ where i, j, l denote image indices.

The minimization problem (A.1) is equivalent to the primal-dual (min-max) problem

$$\arg\min_{\boldsymbol{u}} \max_{z, \|z\|_\infty \le 1} \langle \rho(\boldsymbol{u}), z \rangle + \frac{\theta}{2}\|\boldsymbol{v}^k - \boldsymbol{u}\|^2 \tag{A.2}$$

over the primal variable $\boldsymbol{u} \in \mathbb{R}^{MNH}$ and the dual variable z.

First, we differentiate (A.2) with respect to $\boldsymbol{u}$ and derive its first-order optimality condition, resulting in the following closed-form solution

$$\boldsymbol{u} = \boldsymbol{v}^k - z\frac{\nabla I_1}{\theta}. \tag{A.3}$$

We then plug the solution (A.3) into (A.2), converting the primal-dual problem into the following dual problem

$$\max_{z, \|z\|_\infty \le 1} \langle \rho(\boldsymbol{v}^k - z\frac{\nabla I_1}{\theta}), z \rangle + \frac{1}{2\theta}\|z \nabla I_1\|_2^2. \tag{A.4}$$

If we differentiate (A.4) with respect to z and derive its first-order optimality condition, we have the following formulation

$$\hat{z} = \frac{\theta \rho(\boldsymbol{v}^k)}{|\nabla I_1|^2},$$

which needs to be projected to the convex set $Z = \left\{ z \in \mathbb{R}^{MNH} : \|z\|_\infty \leq 1 \right\}$ to satisfy the constraint $\|z\|_\infty \leq 1$. This results in

$$z = \frac{\hat{z}_{i,j,l}}{\max\left(\left|\hat{z}_{i,j,l}\right|, 1\right)}. \tag{A.5}$$

Note that, although the KKT condition is not considered when we handle the inequality constraint $\|z\|_\infty \leq 1$, the derivation of z above still makes sense as it is equivalent to a one-step proximal gradient descent with the optimal step size.

Finally, we plug (A.5) into (A.3) which leads to the solution for $\boldsymbol{u}$ without involving the dual variable z, i.e.,

$$\boldsymbol{u} = \boldsymbol{v}^k - \frac{\hat{z}_{i,j,l}}{\max\left(\left|\hat{z}_{i,j,l}\right|, 1\right)} \frac{\nabla I_1}{\theta}, \tag{A.6}$$

which is a point-wise, closed-form solution, the same as (5.5) of the $\boldsymbol{u}-$subproblem in Section 5.3.1. We highlight that our derivation presented here can be easily applied to vector images, which usually appear in data terms that use image patches or gradient information.

A.2 Appendix 2

In this section, we derive the solution of the Sherman Morrison formula (5.6) in 2D and 3D. For both cases, we need to invert the left-hand side matrix in Eq. (5.6). As per [12], we have

$$(\mathbf{J}\mathbf{J}^{\mathrm{T}} + \theta \mathbb{1})^{-1} = \theta^{-1}\mathbb{1} - \frac{\mathbf{J}\mathbf{J}^{\mathrm{T}}}{\theta^2 + \theta \mathbf{J}^{\mathrm{T}}\mathbf{J}}.$$

In 2D, this matrix is a 2×2 symmetric matrix for which each entry is of the 2D image size (MN). In 3D, it becomes a 3×3 symmetric matrix for which each entry is of the 3D image size (MNH). The solution $\boldsymbol{u}^{k+1}$ is therefore given by

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^\omega + \left[\mathbb{1} - \frac{\mathbf{J}\mathbf{J}^{\mathrm{T}}}{\theta + \mathbf{J}^{\mathrm{T}}\mathbf{J}}\right] \left[\boldsymbol{v}^k - \boldsymbol{u}^\omega - \theta^{-1}\mathbf{J}(I_1 - I_0)\right], \tag{A.7}$$

which is the form in terms of matrix and vector multiplication. With Eq. (A.7), it is now trivial to derive the final point-wise, closed-form solutions in both 2D and 3D.

First, in 2D where $I_1 \in \mathbb{R}^{MN}$, we have

$$\mathbf{J}\mathbf{J}^{\mathrm{T}} = \begin{bmatrix} I_1^x I_1^x & I_1^x I_1^y \\ I_1^y I_1^x & I_1^y I_1^y \end{bmatrix} \in \left(\mathbb{R}^{MN}\right)^4$$

and $\mathbf{J}^{\mathrm{T}}\mathbf{J} = |\nabla I_1|^2 = I_1^x I_1^x + I_1^y I_1^y$, where $I_1^x \in \mathbb{R}^{MN}$ and $I_1^y \in \mathbb{R}^{MN}$ are the horizontal and vertical derivatives of the source image I_1, respectively. With $\boldsymbol{u} = (u_1, u_2)^{\mathrm{T}} \in \left(\mathbb{R}^{MN}\right)^2$ and $\boldsymbol{v} = (v_1, v_2)^{\mathrm{T}} \in \left(\mathbb{R}^{MN}\right)^2$, we can rewrite Eq. (A.7) into the following forms in terms of both components of $\boldsymbol{u}$

$$\begin{cases} u_x^{k+1} = u_x^{\omega} + \dfrac{\begin{matrix}(I_1^y I_1^y + \theta)(v_x^k - u_x^{\omega}) - I_1^x I_1^y (v_y^k - u_y^{\omega}) \\ -I_1^x (I_1 - I_0)\end{matrix}}{I_1^x I_1^x + I_1^y I_1^y + \theta} \\ u_y^{k+1} = u_y^{\omega} + \dfrac{\begin{matrix}(I_1^x I_1^x + \theta)(v_y^k - u_y^{\omega}) - I_1^y I_1^x (v_x^k - u_x^{\omega}) \\ -I_1^y (I_1 - I_0)\end{matrix}}{I_1^x I_1^x + I_1^y I_1^y + \theta} \end{cases}. \tag{A.8}$$

Then, in 3D where $I_1 \in \mathbb{R}^{MNH}$, we have

$$\mathbf{J}\mathbf{J}^{\mathrm{T}} = \begin{bmatrix} I_1^x I_1^x & I_1^x I_1^y & I_1^x I_1^z \\ I_1^y I_1^x & I_1^y I_1^y & I_1^y I_1^z \\ I_1^z I_1^x & I_1^z I_1^y & I_1^z I_1^z \end{bmatrix} \in \left(\mathbb{R}^{MNH}\right)^9$$

and $\mathbf{J}^{\mathrm{T}}\mathbf{J} = |\nabla I_1|^2 = I_1^x I_1^x + I_1^y I_1^y + I_1^z I_1^z$, where $I_1^x \in \mathbb{R}^{MNH}$, $I_1^y \in \mathbb{R}^{MNH}$ and $I_1^z \in \mathbb{R}^{MNH}$ are the derivatives of I_1 along x, y and z directions, respectively. With $\boldsymbol{u} = (u_1, u_2, u_3)^{\mathrm{T}} \in \left(\mathbb{R}^{MNH}\right)^3$ and $\boldsymbol{v} = (v_1, v_2, v_3)^{\mathrm{T}} \in \left(\mathbb{R}^{MNH}\right)^3$, we can rewrite Eq. (A.7) into the following forms in terms of each component of $\boldsymbol{u}$, i.e.,

$$\begin{cases} u_x^{k+1} = u_x^{\omega} + \dfrac{\begin{array}{c}(I_1^y I_1^y + I_1^z I_1^z + \theta)(v_x^k - u_x^{\omega}) - I_1^x I_1^y (v_y^k - u_y^{\omega}) \\ -I_1^x I_1^z (v_z^k - u_z^{\omega}) - I_1^x (I_1 - I_0)\end{array}}{I_1^x I_1^x + I_1^y I_1^y + I_1^z I_1^z + \theta} \\ u_y^{k+1} = u_y^{\omega} + \dfrac{\begin{array}{c}(I_1^x I_1^x + I_1^z I_1^z + \theta)(v_y^k - u_y^{\omega}) - I_1^y I_1^x (v_x^k - u_x^{\omega}) \\ -I_1^y I_1^z (v_z^k - u_z^{\omega}) - I_1^y (I_1 - I_0)\end{array}}{I_1^x I_1^x + I_1^y I_1^y + I_1^z I_1^z + \theta} \\ u_z^{k+1} = u_z^{\omega} + \dfrac{\begin{array}{c}(I_1^x I_1^x + I_1^y I_1^y + \theta)(v_y^k - u_y^{\omega}) - I_1^z I_1^x (v_x^k - u_x^{\omega}) \\ -I_1^z I_1^y (v_z^k - u_z^{\omega}) - I_1^z (I_1 - I_0)\end{array}}{I_1^x I_1^x + I_1^y I_1^y + I_1^z I_1^z + \theta} \end{cases} . \tag{A.9}$$

We note that both 2D and 3D solutions, i.e., Eqs. (A.8) and (A.9), are closed-form and point-wise and therefore can be computed very efficiently.

References

[1] Hemant K Aggarwal, Merry P Mani, and Mathews Jacob. “MoDL: Model-based deep learning architecture for inverse problems”. In: *IEEE Transactions on Medical Imaging* 38.2 (2018), pp. 394–405.

[2] Ebrahim Al Safadi and Xubo Song. “Learning-based image registration with meta-regularization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10928–10937.

[3] André Araujo, Wade Norris, and Jack Sim. “Computing receptive fields of convolutional neural networks”. In: *Distill* 4.11 (2019), e21.

[4] Vincent Arsigny, Olivier Commowick, Xavier Pennec, and Nicholas Ayache. “A Log-Euclidean Framework for Statistics on Diffeomorphisms”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 924–931.

[5] John Ashburner. “A fast diffeomorphic image registration algorithm”. In: *Neuroimage* 38.1 (2007), pp. 95–113.

[6] Brian B Avants, Charles L Epstein, Murray Grossman, and James C Gee. “Symmetric diffeomorphic image registration with cross-correlation: evaluating automated labeling of elderly and neurodegenerative brain”. In: *Medical Image Analysis* 12.1 (2008), pp. 26–41.

[7] Brian B Avants, Nicholas J Tustison, Gang Song, Philip A Cook, Arno Klein, and James C Gee. "A reproducible evaluation of ANTs similarity metric performance in brain image registration". In: *Neuroimage* 54.3 (2011), pp. 2033–2044.

[8] Wenjia Bai, Matthew Sinclair, Giacomo Tarroni, Ozan Oktay, Martin Rajchl, Ghislain Vaillant, Aaron M Lee, Nay Aung, Elena Lukaschuk, Mihir M Sanghvi, et al. "Automated cardiovascular magnetic resonance image analysis with fully convolutional networks". In: *Journal of Cardiovascular Magnetic Resonance* 20.1 (2018), p. 65.

[9] Simon Baker and Iain Matthews. "Lucas-kanade 20 years on: A unifying framework". In: *International Journal of Computer Vision* 56.3 (2004), pp. 221–255.

[10] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca. "VoxelMorph: A Learning Framework for Deformable Medical Image Registration". In: *IEEE Transactions on Medical Imaging* 38.8 (Aug. 2019), pp. 1788–1800. ISSN: 1558-254X.

[11] Guha Balakrishnan, Amy Zhao, Mert R Sabuncu, John Guttag, and Adrian V Dalca. "An unsupervised learning model for deformable medical image registration". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 9252–9260.

[12] Maurice S Bartlett. "An inverse matrix adjustment arising in discriminant analysis". In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 107–111.

[13] M Faisal Beg, Michael I Miller, Alain Trouvé, and Laurent Younes. "Computing large deformation metric mappings via geodesic flows of diffeomorphisms". In: *International Journal of Computer Vision* 61.2 (2005), pp. 139–157.

[14] Olivier Bernard, Alain Lalande, Clement Zotti, Frederick Cervenansky, Xin Yang, Pheng-Ann Heng, Irem Cetin, Karim Lekadir, Oscar Camara, Miguel Angel Gonzalez Ballester, et al. "Deep learning techniques for automatic MRI cardiac multi-structures

segmentation and diagnosis: is the problem solved?" In: *IEEE Transactions on Medical Imaging* 37.11 (2018), pp. 2514–2525.

[15] Max Blendowski, Lasse Hansen, and Mattias P. Heinrich. "Weakly-supervised learning of multi-modal features for regularised iterative descent in 3D image registration". In: *Medical Image Analysis* 67 (2021), p. 101822.

[16] Chaim Broit. "Optimal registration of deformed images". In: (1981).

[17] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. "High accuracy optical flow estimation based on a theory for warping". In: *European Conference on Computer Vision (ECCV)*. Springer. 2004, pp. 25–36.

[18] Xiaohuan Cao, Jianhua Yang, Jun Zhang, Qian Wang, Pew-Thian Yap, and Dinggang Shen. "Deformable image registration using a cue-aware deep regression network". In: *IEEE Transactions on Biomedical Engineering* 65.9 (2018), pp. 1900–1911.

[19] Antonin Chambolle. "An algorithm for total variation minimization and applications". In: *Journal of Mathematical Imaging and Vision* 20.1 (2004), pp. 89–97.

[20] Chong Chen, Barbara Gris, and Ozan Oktem. "A new variational model for joint image reconstruction and motion estimation in spatiotemporal imaging". In: *SIAM Journal on Imaging Sciences* 12.4 (2019), pp. 1686–1719.

[21] Junyu Chen, Eric C Frey, Yufan He, William P Segars, Ye Li, and Yong Du. "Transmorph: Transformer for unsupervised medical image registration". In: *Medical Image Analysis* 82 (2022), p. 102615.

[22] Junyu Chen, Yufan He, Eric C Frey, Ye Li, and Yong Du. "ViT-V-Net: Vision Transformer for Unsupervised Volumetric Medical Image Registration". In: *arXiv preprint arXiv:2104.06468* (2021).

[23] Junyu Chen, Yihao Liu, Shuwen Wei, Zhangxing Bian, Shalini Subramanian, Aaron Carass, Jerry L Prince, and Yong Du. "A survey on deep learning in medical image registration: New technologies, uncertainty, evaluation metrics, and beyond". In: *arXiv preprint arXiv:2307.15615* (2023).

[24] Xu Chen, Yanda Meng, Yitian Zhao, Rachel Williams, Srinivasa R. Vallabhaneni, and Yalin Zheng. "Learning Unsupervised Parameter-Specific Affine Transformation for Medical Images Registration". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Cham: Springer International Publishing, 2021, pp. 24–34.

[25] Y. Chen and T. Pock. "Trainable Nonlinear Reaction Diffusion: A Flexible Framework for Fast and Effective Image Restoration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1256–1272.

[26] Gary E Christensen and Hans J Johnson. "Consistent image registration". In: *IEEE Transactions on Medical Imaging* 20.7 (2001), pp. 568–582.

[27] Gary E Christensen, Richard D Rabbitt, and Michael I Miller. "Topological properties of smooth anatomic maps". In: (1995).

[28] Noppadol Chumchob and Ke Chen. "Improved variational image registration model and a fast algorithm for its numerical approximation". In: *Numerical Methods for Partial Differential Equations* 28.6 (2012), pp. 1966–1995.

[29] Noppadol Chumchob, Ke Chen, and Carlos Brito-Loeza. "A fourth-order variational image registration model and its fast multigrid algorithm". In: *Multiscale Modeling & Simulation* 9.1 (2011), pp. 89–128.

[30] William R Crum, Lewis D Griffin, Derek LG Hill, and David John Hawkes. "Zen and the art of medical image registration: correspondence, homology, and quality". In: *NeuroImage* 20.3 (2003), pp. 1425–1437.

[31] Adrian V Dalca, Guha Balakrishnan, John Guttag, and Mert R Sabuncu. "Unsupervised learning for fast probabilistic diffeomorphic registration". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 729–738.

[32] Adrian V Dalca, Guha Balakrishnan, John Guttag, and Mert R Sabuncu. "Unsupervised learning of probabilistic diffeomorphic registration for images and surfaces". In: *Medical Image Analysis* 57 (2019), pp. 226–236.

[33] CJ Dean, JR Sykes, RA Cooper, P Hatfield, B Carey, S Swift, SE Bacon, D Thwaites, D Sebag-Montefiore, and AM Morgan. "An evaluation of four CT–MRI co-registration techniques for radiotherapy treatment planning of prone rectal cancer patients". In: *The British Journal of Radiology* 85.1009 (2012), pp. 61–68.

[34] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. "RepVGG: Making VGG-Style ConvNets Great Again". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 13733–13742.

[35] Xiaohan Ding, Xiangyu Zhang, Yizhuang Zhou, Jungong Han, Guiguang Ding, and Jian Sun. "Scaling up your kernels to 31x31: Revisiting large kernel design in cnns". In: *arXiv preprint arXiv:2203.06717* (2022).

[36] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. "An image is worth 16x16 words: Transformers for image recognition at scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[37] Jinming Duan, Ghalib Bello, Jo Schlemper, Wenjia Bai, Timothy JW Dawes, Carlo Biffi, Antonio de Marvao, Georgia Doumoud, Declan P O'Regan, and Daniel Rueckert. "Automatic 3D bi-ventricular segmentation of cardiac images by a shape-refined

multi-task deep learning approach". In: *IEEE Transactions on Medical Imaging* 38.9 (2019), pp. 2151–2164.

[38] Jinming Duan, Jo Schlemper, Chen Qin, Cheng Ouyang, Wenjia Bai, Carlo Biffi, Ghalib Bello, Ben Statton, Declan P O'Regan, and Daniel Rueckert. "VS-Net: Variable splitting network for accelerated parallel MRI reconstruction". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2019, pp. 713–722.

[39] Nicolas Duchateau, Andrew P. King, and Mathieu De Craene. "Machine Learning Approaches for Myocardial Motion and Deformation Analysis". In: *Frontiers in Cardiovascular Medicine* 6 (2020), p. 190.

[40] Jan Ehrhardt, Alexander Schmidt-Richberg, René Werner, and Heinz Handels. "Variational registration: A flexible open-source itk toolbox for nonrigid image registration". In: *Bildverarbeitung für die Medizin 2015: Algorithmen-Systeme-Anwendungen. Proceedings des Workshops vom 15. bis 17. März 2015 in Lübeck*. Springer. 2015, pp. 209–214.

[41] Koen AJ Eppenhof and Josien PW Pluim. "Supervised local error estimation for nonlinear image registration using convolutional neural networks". In: *Medical Imaging 2017: Image Processing*. Vol. 10133. SPIE. 2017, pp. 526–531.

[42] Jingfan Fan, Xiaohuan Cao, Zhong Xue, Pew-Thian Yap, and Dinggang Shen. "Adversarial similarity network for evaluating image alignment in deep learning based registration". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 739–746.

[43] Jingfan Fan, Xiaohuan Cao, Pew-Thian Yap, and Dinggang Shen. "BIRNet: Brain image registration using dual-supervised fully convolutional networks". In: *Medical Image Analysis* 54 (2019), pp. 193–206.

[44] Lijie Fan, Wenbing Huang, Chuang Gan, Stefano Ermon, Boqing Gong, and Junzhou Huang. “End-to-end learning of motion representation for video understanding”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2018, pp. 6016–6025.

[45] Bernd Fischer and Jan Modersitzki. “Curvature based image registration”. In: *Journal of Mathematical Imaging and Vision* 18.1 (2003), pp. 81–85.

[46] Bernd Fischer and Jan Modersitzki. “Fast diffusion registration”. In: *Contemporary Mathematics* 313 (2002), pp. 117–128.

[47] Bruce Fischl, David H Salat, Evelina Busa, Marilyn Albert, Megan Dieterich, Christian Haselgrove, Andre Van Der Kouwe, Ron Killiany, David Kennedy, Shuna Klaveness, et al. “Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain”. In: *Neuron* 33.3 (2002), pp. 341–355.

[48] Claudia Frohn-Schauf, Stefan Henn, and Kristian Witsch. “Multigrid based total variation image registration”. In: *Computing and Visualization in Science* 11.2 (2008), pp. 101–113.

[49] Yarin Gal and Zoubin Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *International Conference on Machine Learning.* PMLR. 2016, pp. 1050–1059.

[50] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative adversarial nets”. In: *Advances in Neural Information Processing Systems.* 2014, pp. 2672–2680.

[51] Daniel Grzech, Mohammad Farid Azampour, Ben Glocker, Julia Schnabel, Nassir Navab, Bernhard Kainz, and Loıc Le Folgoc. “A variational Bayesian method for similarity learning in non-rigid image registration”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2022, pp. 119–128.

[52] Hengtao Guo, Melanie Kruger, Sheng Xu, Bradford J. Wood, and Pingkun Yan. "Deep adaptive registration of multi-modal prostate images". In: *Computerized Medical Imaging and Graphics* 84 (2020), p. 101769.

[53] In Young Ha, Matthias Wilms, and Mattias Heinrich. "Semantically guided large deformation estimation with deep networks". In: *Sensors* 20.5 (2020), p. 1392.

[54] Eldad Haber and Jan Modersitzki. "Intensity Gradient Based Registration and Fusion of Multi-modal Images". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 726–733.

[55] Mohamed Hachama, Agnès Desolneux, and Frédéric Richard. "Combining Registration and Abnormality Detection in Mammography". In: *Biomedical Image Registration*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 178–185.

[56] Stephanie Häger, Stefan Heldmann, Alessa Hering, Sven Kuckertz, and Annkristin Lange. "Variable fraunhofer MEVIS RegLib comprehensively applied to Learn2Reg challenge". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2020, pp. 74–79.

[57] Kerstin Hammernik, Teresa Klatzer, Erich Kobler, Michael P. Recht, Daniel K. Sodickson, Thomas Pock, and Florian Knoll. "Learning a variational network for reconstruction of accelerated MRI data". In: *Magnetic Resonance in Medicine* 79.6 (2018), pp. 3055–3071.

[58] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778.

[59] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[60] Mattias P Heinrich, Mark Jenkinson, Manav Bhushan, Tahreema Matin, Fergus V Gleeson, Michael Brady, and Julia A Schnabel. “MIND: Modality independent neighbourhood descriptor for multi-modal deformable registration”. In: *Med Image Anal* 16.7 (2012), pp. 1423–1435.

[61] Mattias P Heinrich, Oskar Maier, and Heinz Handels. “Multi-modal Multi-Atlas Segmentation using Discrete Optimisation and Self-Similarities.” In: *VISCERAL Challenge@ ISBI* 1390 (2015), p. 27.

[62] Alessa Hering, Bram van Ginneken, and Stefan Heldmann. “mlVIRNET: Multilevel Variational Image Registration Network”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*. Cham: Springer International Publishing, 2019, pp. 257–265.

[63] Alessa Hering, Lasse Hansen, Tony CW Mok, Albert Chung, Hanna Siebert, Stephanie Häger, Annkristin Lange, Sven Kuckertz, Stefan Heldmann, Wei Shao, et al. “Learn2Reg: comprehensive multi-task medical image registration challenge, dataset and evaluation in the era of deep learning”. In: *arXiv preprint arXiv:2112.04489* (2021).

[64] Monica Hernandez. “Band-limited stokes large deformation diffeomorphic metric mapping”. In: *IEEE Journal of Biomedical and Health Informatics* 23.1 (2018), pp. 362–373.

[65] Derek LG Hill, Philipp G Batchelor, Mark Holden, and David J Hawkes. “Medical image registration”. In: *Physics in medicine & biology* 46.3 (2001), R1.

[66] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[67] Andrew Hoopes, Malte Hoffmann, Bruce Fischl, John Guttag, and Adrian V Dalca. “Hypermorph: Amortized hyperparameter learning for image registration”. In: *International Conference on Information Processing in Medical Imaging*. Springer. 2021, pp. 3–17.

[68] Berthold KP Horn and Brian G Schunck. “Determining optical flow”. In: *Techniques and Applications of Image Understanding*. Vol. 281. International Society for Optics and Photonics. 1981, pp. 319–331.

[69] Xiaojun Hu, Miao Kang, Weilin Huang, Matthew R. Scott, Roland Wiest, and Mauricio Reyes. “Dual-Stream Pyramid Registration Network”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*. Cham: Springer International Publishing, 2019, pp. 382–390.

[70] Thomas S Huang and RY Tsai. *Image sequence analysis: Motion estimation*. Springer, 1981.

[71] Po-Yu Huang, Wan-Ting Hsu, Chun-Yueh Chiu, Ting-Fan Wu, and Min Sun. “Efficient uncertainty estimation for semantic segmentation in videos”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 520–535.

[72] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. “Spatial transformer networks”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 2017–2025.

[73] Xi Jia, Joseph Bartlett, Wei Chen, Siyang Song, Tianyang Zhang, Xinxing Cheng, Wenqi Lu, Zhaowen Qiu, and Jinming Duan. “Fourier-net: Fast image registration with band-limited deformation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 1. 2023, pp. 1015–1023.

[74] Xi Jia, Joseph Bartlett, Tianyang Zhang, Wenqi Lu, Zhaowen Qiu, and Jinming Duan. “U-net vs transformer: Is u-net outdated in medical image registration?” In: *International Workshop on Machine Learning in Medical Imaging*. Springer. 2022, pp. 151–160.

[75] Xi Jia, Alexander Thorley, Wei Chen, Huaqi Qiu, Linlin Shen, Iain B Styles, Hyung Jin Chang, Ales Leonardis, Antonio De Marvao, Declan P O’Regan, et al. “Learning a Model-Driven Variational Network for Deformable Image Registration”. In: *IEEE Transactions on Medical Imaging* 41.1 (2021), pp. 199–212.

[76] Xi Jia, Alexander Thorley, Alberto Gomez, Wenqi Lu, Dipak Kotecha, and Jinming Duan. “Fourier-Net+: Leveraging Band-Limited Representation for Efficient 3D Medical Image Registration”. In: *arXiv preprint arXiv:2307.02997* (2023).

[77] Hans J Johnson and Gary E Christensen. “Consistent landmark and intensity-based image registration”. In: *IEEE Transactions on Medical Imaging* 21.5 (2002), pp. 450–461.

[78] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. “Perceptual losses for real-time style transfer and super-resolution”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 694–711.

[79] Miao Kang, Xiaojun Hu, Weilin Huang, Matthew R. Scott, and Mauricio Reyes. “Dual-stream pyramid registration network”. In: *Medical Image Analysis* 78 (2022), p. 102379.

[80] Boah Kim, Inhwa Han, and Jong Chul Ye. “DiffuseMorph: unsupervised deformable image registration using diffusion model”. In: *European Conference on Computer Vision*. Springer. 2022, pp. 347–364.

[81] Boah Kim, Dong Hwan Kim, Seong Ho Park, Jieun Kim, June-Goo Lee, and Jong Chul Ye. "CycleMorph: cycle consistent unsupervised deformable image registration". In: *Medical Image Analysis* 71 (2021), p. 102036.

[82] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[83] Stefan Klein, Marius Staring, Keelin Murphy, Max A Viergever, and Josien PW Pluim. "Elastix: a toolbox for intensity-based medical image registration". In: *IEEE Transactions on Medical Imaging* 29.1 (2009), pp. 196–205.

[84] Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. "Variational Networks: Connecting Variational Methods and Deep Learning". In: *Pattern Recognition.* Lecture Notes in Computer Science. Springer, 2017, pp. 281–293.

[85] Julian Krebs, Hervé Delingette, Boris Mailhé, Nicholas Ayache, and Tommaso Mansi. "Learning a probabilistic model for diffeomorphic registration". In: *IEEE Transactions on Medical Imaging* 38.9 (2019), pp. 2165–2176.

[86] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems.* 2012, pp. 1097–1105.

[87] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1.4 (1989), pp. 541–551.

[88] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[89] Antoine Legouhy, Olivier Commowick, François Rousseau, and Christian Barillot. "Unbiased longitudinal brain atlas creation using robust linear registration and log-Euclidean framework for diffeomorphisms". In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE. 2019, pp. 1038–1041.

[90] Hongming Li and Yong Fan. "Non-rigid image registration using self-supervised fully convolutional networks without training data". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE. 2018, pp. 1075–1078.

[91] Zisheng Li and Masahiro Ogino. "Adversarial Learning for Deformable Image Registration: Application to 3D Ultrasound Image Fusion". In: *Smart Ultrasound Imaging and Perinatal, Preterm and Paediatric Image Analysis*. Springer, 2019, pp. 56–64.

[92] Tungyou Lin, Carole Le Guyader, Ivo Dinov, Paul Thompson, Arthur Toga, and Luminita Vese. "Gene expression data to mouse atlas registration using a nonlinear elasticity smoother and landmark points constraints". In: *Journal of Scientific Computing* 50.3 (2012), pp. 586–609.

[93] Risheng Liu, Zi Li, Xin Fan, Chenying Zhao, Hao Huang, and Zhongxuan Luo. "Learning deformable image registration from optimization: perspective, modules, bilevel training and beyond". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.11 (2021), pp. 7688–7704.

[94] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10012–10022.

[95] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[96] Bradley Christopher Lowekamp, David T Chen, Luis Ibáñez, and Daniel Blezek. "The design of SimpleITK". In: *Frontiers in Neuroinformatics* 7 (2013), p. 45.

[97] Wenqi Lu, Jinming Duan, Zhaowen Qiu, Zhenkuan Pan, Ryan Wen Liu, and Li Bai. "Implementation of high-order variational models made easy for image processing". In: *Mathematical Methods in the Applied Sciences* 39.14 (2016), pp. 4208–4233.

[98] Dwarikanath Mahapatra, Bhavna Antony, Suman Sedai, and Rahil Garnavi. "Deformable medical image registration using generative adversarial networks". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE. 2018, pp. 1449–1453.

[99] JB Antoine Maintz and Max A Viergever. "A survey of medical image registration". In: *Medical Image Analysis* 2.1 (1998), pp. 1–36.

[100] Timo Makela, Patrick Clarysse, Outi Sipila, Nicoleta Pauna, Quoc Cuong Pham, Toivo Katila, and Isabelle E Magnin. "A review of cardiac image registration methods". In: *IEEE Transactions on Medical Imaging* 21.9 (2002), pp. 1011–1021.

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

[102] Daniel S Marcus, Tracy H Wang, Jamie Parker, John G Csernansky, John C Morris, and Randy L Buckner. "Open Access Series of Imaging Studies (OASIS): cross-sectional MRI data in young, middle aged, nondemented, and demented older adults". In: *Journal of Cognitive Neuroscience* 19.9 (2007), pp. 1498–1507.

[103] K Marias, JM Brady, RP Highnam, S Parbhoo, AM Seifalian, and M Wirth. "Registration and matching of temporal mammograms for detecting abnormalities". In: *Medical Imaging Understanding and Analysis* (1999).

[104] R Marti, R Zwiggelaar, and C Rubin. "Automatic mammographic registration: towards the detection of abnormalities". In: *S T1U Conference on Medical Image Understanding and Analysis*. 2001, pp. 149–152.

[105] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. "V-net: Fully convolutional neural networks for volumetric medical image segmentation". In: *2016 fourth International Conference on 3D Vision (3DV)*. IEEE. 2016, pp. 565–571.

[106] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[107] Marc Modat, Gerard R Ridgway, Zeike A Taylor, Manja Lehmann, Josephine Barnes, David J Hawkes, Nick C Fox, and Sébastien Ourselin. "Fast free-form deformation using graphics processing units". In: *Computer Methods and Programs in Biomedicine* 98.3 (2010), pp. 278–284.

[108] Jan Modersitzki. *Numerical methods for image registration*. Oxford University Press on Demand, 2004.

[109] Tony C. W. Mok and Albert C. S. Chung. "Conditional Deformable Image Registration with Convolutional Neural Network". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Cham: Springer International Publishing, 2021, pp. 35–45.

[110] Tony C.W. Mok and Albert C.S. Chung. "Fast Symmetric Diffeomorphic Image Registration with Convolutional Neural Networks". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[111] Tony CW Mok and Albert Chung. "Large deformation diffeomorphic image registration with laplacian pyramid networks". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2020, pp. 211–221.

[112] Marc Niethammer, Roland Kwitt, and Francois-Xavier Vialard. “Metric learning for image registration”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2019, pp. 8463–8472.

[113] Francisco PM Oliveira and Joao Manuel RS Tavares. “Medical image registration: a review”. In: *Computer Methods in Biomechanics and Biomedical Engineering* 17.2 (2014), pp. 73–93.

[114] Nils Papenberg, Andrés Bruhn, Thomas Brox, Stephan Didas, and Joachim Weickert. “Highly accurate optic flow computation with theoretically justified warping”. In: *International Journal of Computer Vision* 67.2 (2006), pp. 141–158.

[115] Bartłomiej W Papież, Adam Szmul, Vicente Grau, J Michael Brady, and Julia A Schnabel. “Non-local graph-based regularization for deformable image registration”. In: *Medical Computer Vision and Bayesian and Graphical Models for Biomedical Imaging.* Springer, 2016, pp. 199–207.

[116] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. “PyTorch: An imperative style, high-performance deep learning library”. In: *Advances in Neural Information Processing Systems.* 2019, pp. 8024–8035.

[117] Steffen E Petersen, Paul M Matthews, Fabian Bamberg, David A Bluemke, Jane M Francis, Matthias G Friedrich, Paul Leeson, Eike Nagel, Sven Plein, Frank E Rademakers, et al. “Imaging in population science: cardiovascular magnetic resonance in 100,000 participants of UK Biobank-rationale, challenges and approaches”. In: *Journal of Cardiovascular Magnetic Resonance* 15.1 (2013), p. 46.

[118] Josien PW Pluim, Sascha EA Muenzing, Koen AJ Eppenhof, and Keelin Murphy. “The truth is hard to make: Validation of medical image registration”. In: *2016 23rd International Conference on Pattern Recognition (ICPR).* IEEE. 2016, pp. 2294–2300.

[119] Chen Qin, Wenjia Bai, Jo Schlemper, Steffen E Petersen, Stefan K Piechnik, Stefan Neubauer, and Daniel Rueckert. “Joint learning of motion estimation and segmentation for cardiac MR image sequences”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 472–480.

[120] Huaqi Qiu, Kerstin Hammernik, Chen Qin, Chen Chen, and Daniel Rueckert. “Embedding Gradient-Based Optimization in Image Registration Networks”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2022, pp. 56–65.

[121] Huaqi Qiu, Chen Qin, Loic Le Folgoc, Benjamin Hou, Jo Schlemper, and Daniel Rueckert. “Deep learning for cardiac motion estimation: supervised vs. unsupervised training”. In: *International Workshop on Statistical Atlases and Computational Models of the Heart*. Springer. 2019, pp. 186–194.

[122] Huaqi Qiu, Chen Qin, Andreas Schuh, Kerstin Hammernik, and Daniel Rueckert. “Learning Diffeomorphic and Modality-invariant Registration using B-splines”. In: *Medical Imaging with Deep Learning*. 2021.

[123] René Ranftl, Kristian Bredies, and Thomas Pock. “Non-local total generalized variation for optical flow estimation”. In: *European Conference on Computer Vision (ECCV)*. Springer. 2014, pp. 439–454.

[124] Torsten Rohlfing. “Image similarity and tissue overlaps as surrogates for image registration accuracy: widely used but unreliable”. In: *IEEE Transactions on Medical Imaging* 31.2 (2011), pp. 153–163.

[125] Olaf Ronneberger, Philipp Fischer, and Thomas 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.

[126] S. Roth and M.J. Black. “Fields of Experts: a framework for learning image priors”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 2. 2005, 860–867 vol. 2.

[127] Leonid I Rudin, Stanley Osher, and Emad Fatemi. “Nonlinear total variation based noise removal algorithms”. In: *Physica D: Nonlinear Phenomena* 60.1-4 (1992), pp. 259–268.

[128] D. Rueckert and J. A. Schnabel. “Model-Based and Data-Driven Strategies in Medical Image Computing”. In: *Proceedings of the IEEE* 108.1 (Jan. 2020), pp. 110–124.

[129] Daniel Rueckert, Paul Aljabar, Rolf A. Heckemann, Joseph V. Hajnal, and Alexander Hammers. “Diffeomorphic Registration Using B-Splines”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 702–709.

[130] Daniel Rueckert, Luke I Sonoda, Carmel Hayes, Derek LG Hill, Martin O Leach, and David J Hawkes. “Nonrigid registration using free-form deformations: application to breast MR images”. In: *IEEE Transactions on Medical Imaging* 18.8 (1999), pp. 712–721.

[131] Jo Schlemper, Jose Caballero, Joseph V Hajnal, Anthony N Price, and Daniel Rueckert. “A deep cascade of convolutional neural networks for dynamic MR image reconstruction”. In: *IEEE Transactions on Medical Imaging* 37.2 (2017), pp. 491–503.

[132] Loren Arthur Schwarz. “Non-rigid registration using free-form deformations”. In: *Technische Universität München* 6.8 (2007).

[133] Hanna Siebert, Lasse Hansen, and Mattias P Heinrich. “Fast 3D registration with accurate optimisation and little learning for Learn2Reg 2021”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2021, pp. 174–179.

[134] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[135] Hessam Sokooti, Bob de Vos, Floris Berendsen, Boudewijn P. F. Lelieveldt, Ivana Išgum, and Marius Staring. "Nonrigid Image Registration Using Multi-scale 3D Convolutional Neural Networks". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2017*. Cham: Springer International Publishing, 2017, pp. 232–239.

[136] Aristeidis Sotiras, Christos Davatzikos, and Nikos Paragios. "Deformable medical image registration: A survey". In: *IEEE Transactions on Medical Imaging* 32.7 (2013), pp. 1153–1190.

[137] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. "On the importance of initialization and momentum in deep learning". In: *International Conference on Machine Learning*. PMLR. 2013, pp. 1139–1147.

[138] National Lung Screening Trial Research Team. "Reduced lung-cancer mortality with low-dose computed tomographic screening". In: *New England Journal of Medicine* 365.5 (2011), pp. 395–409.

[139] National Lung Screening Trial Research Team. "The national lung screening trial: overview and study design". In: *Radiology* 258.1 (2011), pp. 243–253.

[140] Alexander Thorley, Xi Jia, Hyung Jin Chang, Boyang Liu, Karina Bunting, Victoria Stoll, Antonio de Marvao, Declan P O'Regan, Georgios Gkoutos, Dipak Kotecha, et al. "Nesterov Accelerated ADMM for Fast Diffeomorphic Image Registration". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2021, pp. 150–160.

[141] C Trabelsi, O Bilaniuk, Y Zhang, D Serdyuk, S Subramanian, JF Santos, S Mehri, N Rostamzadeh, Y Bengio, and CJ Pal. “Deep complex networks. arXiv preprint arXiv: 170509792”. In: (2017).

[142] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. “Attention is all you need”. In: *Advances in Neural Information Processing Systems* 30 (2017).

[143] Tom Vercauteren, Xavier Pennec, Aymeric Perchant, and Nicholas Ayache. “Diffeomorphic demons: Efficient non-parametric image registration”. In: *NeuroImage* 45.1 (2009), S61–S72.

[144] Luminita A Vese and Carole Le Guyader. *Variational methods in image processing.* CRC Press, 2016.

[145] Max A Viergever, JB Antoine Maintz, Stefan Klein, Keelin Murphy, Marius Staring, and Josien PW Pluim. *A survey of medical image registration–under review.* 2016.

[146] Valery Vishnevskiy, Tobias Gass, Gabor Szekely, Christine Tanner, and Orcun Goksel. “Isotropic total variation regularization of displacements in parametric image registration”. In: *IEEE Transactions on Medical Imaging* 36.2 (2016), pp. 385–395.

[147] Christoph Vogel, Stefan Roth, and Konrad Schindler. “An evaluation of data costs for optical flow”. In: *German Conference on Pattern Recognition.* Springer. 2013, pp. 343–353.

[148] Bob D. de Vos, Floris F Berendsen, Max A Viergever, Hessam Sokooti, Marius Staring, and Ivana Išgum. “A deep learning framework for unsupervised affine and deformable image registration”. In: *Medical Image Analysis* 52 (2019), pp. 128–143.

[149] Bob D. de Vos, Floris F. Berendsen, Max A. Viergever, Marius Staring, and Ivana Išgum. “End-to-End Unsupervised Deformable Image Registration with a Convolutional Neural Network”. In: *Deep Learning in Medical Image Analysis and Multi-*

modal Learning for Clinical Decision Support. Cham: Springer International Publishing, 2017, pp. 204–212.

[150] Jian Wang and Miaomiao Zhang. "DeepFlash: An efficient network for learning-based medical image registration". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4444–4452.

[151] Shuxin Wang, Shilei Cao, Dong Wei, Renzhen Wang, Kai Ma, Liansheng Wang, Deyu Meng, and Yefeng Zheng. "LT-Net: Label transfer by learning reversible voxel-wise correspondence for one-shot medical image segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9162–9171.

[152] Yinsong Wang, Huaqi Qiu, and Chen Qin. "Conditional Deformable Image Registration with Spatially-Variant and Adaptive Regularization". In: *arXiv preprint arXiv:2303.10700* (2023).

[153] Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. "An improved algorithm for tv-l 1 optical flow". In: *Statistical and Geometrical Approaches to Visual Motion Analysis*. Springer, 2009, pp. 23–45.

[154] Manuel Werlberger, Thomas Pock, and Horst Bischof. "Motion estimation with non-local total variation regularization". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 2464–2471.

[155] René Werner, Alexander Schmidt-Richberg, Heinz Handels, and Jan Ehrhardt. "Estimation of lung motion fields in 4D CT data by variational non-linear intensity-based registration: A comparison and evaluation study". In: *Physics in Medicine & Biology* 59.15 (2014), p. 4247.

[156] Cheng Xu, Ying Wen, and Bing He. "A Novel Fractional Order Derivate Based Log-demons with Driving Force for High Accurate Image Registration". In: *ICASSP*

2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE. 2019, pp. 1997–2001.

[157] Pingkun Yan, Sheng Xu, Ardeshir R Rastinehad, and Brad J Wood. “Adversarial image registration with application for MR and TRUS image fusion”. In: *International Workshop on Machine Learning in Medical Imaging*. Springer. 2018, pp. 197–204.

[158] Xiao Yang, Roland Kwitt, Martin Styner, and Marc Niethammer. “Quicksilver: Fast predictive image registration–a deep learning approach”. In: *NeuroImage* 158 (2017), pp. 378–396.

[159] Christopher Zach, Thomas Pock, and Horst Bischof. “A duality based approach for realtime TV-L 1 optical flow”. In: *Joint Pattern Recognition Symposium*. Springer. 2007, pp. 214–223.

[160] Jianping Zhang and Ke Chen. “Variational image registration by a total fractional-order variation model”. In: *Journal of Computational Physics* 293 (2015), pp. 442–461.

[161] Jun Zhang. “Inverse-consistent deep networks for unsupervised deformable image registration”. In: *arXiv preprint arXiv:1809.03443* (2018).

[162] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. “Learning deep CNN denoiser prior for image restoration”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3929–3938.

[163] Miaomiao Zhang and P Thomas Fletcher. “Fast diffeomorphic image registration via fourier-approximated lie algebras”. In: *International Journal of Computer Vision* 127.1 (2019), pp. 61–73.

[164] Yungeng Zhang, Yuru Pei, and Hongbin Zha. “Learning dual transformer network for diffeomorphic registration”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2021, pp. 129–138.

[165] Amy Zhao, Guha Balakrishnan, Fredo Durand, John V Guttag, and Adrian V Dalca. "Data augmentation using learned transformations for one-shot medical image segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 8543–8553.

[166] Shengyu Zhao, Yue Dong, Eric I-Chao Chang, and Yan Xu. "Recursive Cascaded Networks for Unsupervised Medical Image Registration". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[167] Shengyu Zhao, Tingfung Lau, Ji Luo, I Eric, Chao Chang, and Yan Xu. "Unsupervised 3D end-to-end medical image registration with volume tweening network". In: *IEEE Journal of Biomedical and Health Informatics* 24.5 (2019), pp. 1394–1404.

[168] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2223–2232.