Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

Kristoffer Røise

Deep Learning Based Ultrasound Volume Registration for Interventional Applications

Master's thesis in Electronics Systems Design and Innovation
Supervisor: Gabriel Hanssen Kiss and Ilangko Balasingham
June 2020

NTNU
Norwegian University of Science and Technology

Kristoffer Røise

Deep Learning Based Ultrasound Volume Registration for Interventional Applications

Master's thesis in Electronics Systems Design and Innovation
Supervisor: Gabriel Hanssen Kiss and Ilangko Balasingham
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems

NTNU
Norwegian University of
Science and Technology

Summary

With the recent development of new technology, minimally invasive interventions can be performed with equivalent outcomes compared with conventional sternotomy. The growing acceptance of percutaneous catheter-based interventions have seen new echocardiographic systems be developed to help guide and monitor the intervention. Many interventional procedures rely on real time three-dimensional transesophageal echocardiography (TEE) for monitoring instrument manipulation during the intervention. When using fused perioperative computed tomography (CT) and ultrasound during surgery, it is important to detect and correct for probe movement to keep the preoperative information in sync with the current ultrasound images. Auto-registration of ultrasound to ultrasound is therefore highly desirable to avoid manual realignment during surgery.

This thesis aims to contribute towards fully automated monomodal ultrasound image registration of perioperative echocardiographic recordings by investigating the feasibility of fast, automatic image registration in TEE images using unsupervised deep learning methods. A registration pipeline is proposed, composed of a deep neural network to do local registration on patches, and a Procrustes method that takes the patch predictions and transforms them to a global alignment and introduces a rigidity constraint that is applied to the full volume. The predictions are estimated using convolutional and linear layers that are combined to form a spatial transformer network, inspired by recent advancements in image registration. The network is trained in an unsupervised fashion, and thus avoiding the need for ground truth annotations. To evaluate the full potential of the registration method, different preprocessing algorithms were evaluated (bilateral and non-local mean (NLM) filtering), together with a comparison of registration on end-diastolic and end-systolic frame. Mixed precision training was evaluated to check the feasibility of doing full volume training and prediction.

The model was trained and evaluated on samples from 26 patients (23 for training and validation, 3 for testing). Due to the small amount of datasets available, five-fold cross-validation was performed to check the networks ability to generalize to previously unseen data, with good results. The most successful registration achieves a 7.3% increase in normalized cross correlation (NCC) compared to the baseline NCC prior to registration. Worse registration results were observed in samples with poor quality or large transformations between the volumes. End-systolic registration showed promising results, but the results were inconclusive. The results on preprocessing showed that the advanced NLM algorithm only achieved slightly better results compared to the simpler bilateral filter. Mixed precision training achieved almost the same results as full precision training, at a 45% reduction in memory consumption. With the low memory requirement of mixed precision, training and prediction on larger volumes is feasible using this method. In conclusion, ultrasound volume registration using this method is feasible if efforts are made to gather more data, reduce the inference time and improve robustness.

This page intentionally left blank.

Sammendrag

Med den nylige utviklingen av ny teknologi kan minimalt invasive intervensjoner utføres med like gode utfall sammenlignet med konvensjonell åpen hjertekirurgi. Den økende aksepten for perkutane kateterbaserte intervensjoner har sett nye ekkokardiografiske systemer bli utviklet for å hjelpe og overvåke intervensjonen. Mange intervensjonelle prosedyrer er avhengige av tredimensjonal transøsofagal ekkokardiografi (TØE) i sanntid for observasjon av instrument-manipulering under intervensjonen. Når man bruker fusjonert perioperativ computertomografi (CT) og ultralyd under operasjonen, er det viktig å oppdage og korrigere for probebevegelse for å holde den preoperative informasjonen synkronisert med nåværende ultralydbilde. Automatisk registrering av ultralyd til ultralyd er derfor svært ønskelig for å unngå manuell justering under operasjonen.

Denne oppgaven har som mål å bidra til helautomatisert monomodal ultralydbildegistrering av perioperative ekkokardiografiske opptak ved å undersøke muligheten for rask, automatisk bilderegistrering i TØE-bilder ved bruk av ikke-styrte dype læringsmetoder. Det blir foreslått en todelt registreringsmetode, sammensatt av et dypt nevralt nettverk for å gjøre lokal registrering på delvolumer, og en Procrustes-metode som tar prediksjoner fra delvolumene og transformerer dem til en global justering og introduserer en stivhetsbegrensning som blir brukt på hele volumet. Prediksjonene er estimert ved hjelp av konvolusjonelle og lineære lag som er kombinert for å danne et romlig transformasjonsnettverk, inspirert av nylige fremskritt innen bilderegistrering. Nettverket er trent på en ikke-styrt måte, og unngår dermed behovet for sanne justeringer. For å evaluere hele potensialet i registreringsmetoden ble forskjellige preprosesseringsalgoritmer evaluert (bilateral og ikke-lokal middelverdi (ILM) filtrering), sammen med en sammenligning av registrering på sluttdiastolisk og sluttsystolisk bilde. Blandet presisjonstrening (16-bit med 32-bit mastervekter) ble evaluert for å undersøke muligheten for å gjøre trening og prediksjon på volumer av full størrelse.

Modellen ble trent og evaluert på data fra 26 pasienter (23 for trening og validering, 3 for testing). På grunn av den lille mengden datasett som var tilgjengelig, ble fem-fold kryssvalidering utført for å sjekke nettverkets evne til å generalisere til tidligere usette data, med gode resultater. Den mest vellykkede registreringen oppnår en 7,3% økning i normalisert kryss korrelasjon (NKK) sammenlignet med grunnverdi NKK før registrering. Verre registreringsresultater ble observert i volumer med dårlig kvalitet eller store transformasjoner mellom volumene. Sluttsystolisk registrering viste lovende resultater, men resultatene var uavklarte. Resultatene fra preprosessering viste at den avanserte ILM-algoritmen oppnådde minimalt bedre resultater sammenlignet med det enklere bilaterale filteret. Blandet presisjonstrening oppnådde nesten de samme resultatene som full presisjonstrening, med en reduksjon på 45% i minneforbruket. Med det lave minnekravet når blandet presisjon brukes, er trening og prediksjon på større volumer gjennomførbart ved bruk av denne metoden. Som konklusjon er ultralydvolumregistrering ved bruk av denne metoden mulig hvis det arbeides for å samle inn mer data, redusere inferensstiden og forbedre robustheten.

This page intentionally left blank.

Preface

This thesis represents the end of my master's degree in Electronics Systems Design and Innovation at the Norwegian University of Science and Technology (NTNU) in Trondheim. These five years have gone by so fast, and have left me with lots of knowledge, lots of new friends and lots of new experiences.

I chose this project because it required a wide range of skills in mathematics, programming and computer science, some of which I possessed at the start of the project and some of which I had to learn. It has been a challenging project which leaves me with insights in fields that I had no prior knowledge of. I have also had the privilege of working together with experts in other fields than my own, which has been a great experience.

The year 2020 will be a year that I will remember forever. Not only does it mark the year where I completed my master's degree, it was also the year where the COVID-19 pandemic put the entire society on its head. Fortunately, I have worked with people that have been able to adapt to the situation, and it is with relief that I now hand in my final thesis.

Acknowledgements

I would like to thank Erik Andreas Rye Berg, Bjørnar Grenne, Håvard Dalen and Espen Holte at the Department of Circulation and Medical Imaging (ISB) for acquiring and sharing the datasets used in this thesis. Without them, this project would not have been possible to complete. Postdoc Erik Smistad at the same department, provided access to a high-performance GPU which made implementation and execution of the project much easier.

Above all, I would like to express my gratitude to my supervisor, Dr. Gabriel Hanssen Kiss. Throughout the duration of this project, he has always made time, provided helpful comments and suggestions for improvement. He also provided me with implementations of supporting methods that were needed to produce the results. Emails and messages have been answered at all times of the day, always with a thoughtful answer and for that, I am very grateful.

Lastly I would like to thank my girlfriend and fellow student Kristin Schive Hjelde for continuously listening to my ideas, problems and frustrations, and for help with proofreading and corrections. For moral and practical support, I would like to thank my parents, Anne Reidun Røise and Dag Røise.

Kristoffer Røise
Trondheim, June 13, 2020

This page intentionally left blank.

Table of Contents

List of Tables

This page intentionally left blank.

List of Figures

This page intentionally left blank.

1 | Introduction

1.1 Background

Open-heart surgery has been successfully utilized over the last decades to treat heart diseases and fix valvular defects. With the development of new technology, minimally invasive or catheter-based percutaneous interventions can be performed with equivalent or superior outcomes compared with conventional sternotomy [1–3]. Studies have also shown that these procedures are associated with faster recovery, shorter hospital stay and less pain for the patient [4–6]. With the growing acceptance and usage of percutaneous catheter-based interventions, new echocardiographic systems have been developed to help guide and monitor the intervention [7].

Echocardiography refers to ultrasound imaging of the heart. Three-dimensional echocardiography has emerged as an important tool in preprocedural planning and many interventional procedures rely on 3D ultrasound imaging for intraprocedural guidance, such as percutaneous mitral valve procedures, transcatheter aortic valve implantation and left atrial appendage closure [8–10]. In particular, real-time 3D transesophageal echocardiography (TEE) is used in cardiac interventions. Due to the absence of interference from lungs or ribs and the close proximity to the heart, TEE can provide higher quality images than conventional transthoracic echocardiography (TTE) [10].

Because of the increasing complexity of interventional procedures, TEE imaging is combined with other imaging modalities for precise preprocedural planning and intraprocedural image guidance [11, 12]. This technique of combining different imaging modalities is referred to as *image fusion.* During the procedure, computed tomography (CT) fluoroscopy is fused with 3D TEE to help guide the intervention. TEE probe movement under intervention could lead to mis-registration with respect to the preoperative plan and needs to be corrected. This generally requires manual realignment which is time-consuming as constant realignment is required. Auto-registration of ultrasound to ultrasound is highly desirable to avoid manual re-labelling and to correct probe movement such that the perioperative CT and ultrasound image can be fused without loss of information.

Image registration is the process of spatially aligning a reference (fixed) and a source (moving) image and is generally achieved through either non-rigid (deformable) or rigid (affine) transformations. Deformable transformations can account for local deformations through e.g. B-splines [13] and is frequently used in multi-modal image fusion or when the object that is imaged is assumed to be a non-rigid body [14, 15]. A rigid transformation is a linear mapping that preserves points, straight lines and planes and is less complex than non-rigid ones. Affine transformations include scaling, rotation, skewing and translation. When choosing the type of spatial transformation, assumptions on heart rigidity are generally made. The heart is indeed a non-rigid body but can be assumed to be rigid with periodic heart motion throughout the

imaging process [16]. Together with the fact that ultrasound to ultrasound registration is a monomodal image fusion process, affine transformation is assumed to be sufficient to correctly auto-register two ultrasound volumes.

Efforts have been made to provide fully automatic registration of ultrasound volumes. SimpleElastix is an open-source command line program extension of SimpleITK [17] that can be used both for deformable and affine registration of several imaging modalities. By iteratively aligning the volumes until convergence of a similarity metric, the volumes are aligned. Methods aimed at monomodal ultrasound registration have also been proposed. Danudibroto et al. proposed a spatiotemporal method based on a multiscale iterative Farnebäck optic flow and piecewise 1-D cubic B-spline interpolation [18]. Other methods have also been proposed [19–21], but together with the aforementioned, they are not suitable for interventional setups as they are not fast enough. Recently, Pham et al. [22] proposed a method similar to Danudibroto et al., which implements the Farnebäck decompostion on the graphical processing unit (GPU) to allow for close to real-time registration on an ultrasound scanner. Their results show promise but fails to accurately register volumes in some cases.

Deep learning methods based on convolutional neural networks (CNNs) have revolutionized several fields of research in medical imaging, such as tumor detection [23] and pulmonary lobe segmentation [24]. Such deep learning techniques are well suited for image registration because they learn to aggregate the information of various complexities in images, leaving only evaluation of the regression model at runtime. Moreover, CNNs are highly parallelizable which makes implementation and execution on GPUs extremely fast.

To avoid the need for costly ground truth annotations, many recent image registration methods exploits the use of unsupervised learning in model training. Convolutional stacked autoencoders have been frequently used to perform both monomodal [25–28] and multimodal [29–31] deformable image registration. However, since in the case of intra-patient alignment, a rigid deformation between the volumes is desirable. Although not explicitly introduced as a method for image registration, the spatial transformer network (STN) presented by Jaderberg et al. [32] was one of the first methods that utilized deep learning for aligning images. The STN is designed to be integrated in any neural network, with its task being to spatially transform input images to simplify the classification task. Based on the STN, Chee and Wu [33] proposed an unsupervised learning method for affine image registration on 3D magnetic resonance (MR) images. Their method uses a convolutional encoder to extract features from the MR scans and produce the transformation parameters to align the volumes. End-to-end unsupervised 3D image registration frameworks have been proposed by both de Vos et al. [13] and Zhao et al. [34], both achieving close to real-time registration. Both methods perform an initial affine registration step of an image pair based on the STN, before a convolutional autoencoder is used to perform deformable registration.

A 3D convolutional layer have a high computational complexity and a high memory consumption. Due to the memory constraints, the proposed networks are trained with either 3D patches [35, 36] or aggressively downsampled volumes [26, 37]. Other methods uses low resolution inputs [38] or CT and MR images which typically have a limited amount of axial slices [13, 33], avoiding the memory consumption issue. Recently, Lee et al. [24] presented the PLS-Net for pulmonary lobe segmentation in high resolution volumetric CT images, achieving state of the art performance by using 3D depthwise separable convolutions, dilated convolutions

and input reinforcement while significantly reducing memory usage by utilization of mixed precision training. They also proposed a dilated residual dense block (DRDB) to efficiently enlarge the receptive field of a network to capture wide-ranging, multi-scale context features.

1.2 Aim and Method

Multimodal image fusion is frequently used to help preprocedural planning and intraprocedural guidance of percutaneous interventions. Manual registration of preprocedural TEE echocardiography with intraprocedural TEE is a time-consuming process. Therefore, once a manual alignment is achieved between CT and ultrasound, it is highly desirable that subsequent ultrasound misalignments are automatically compensated for. This thesis aims to contribute towards fully automated monomodal ultrasound image registration of perioperative echocardiographic recordings through investigating the feasibility of fast, automatic image registration in TEE images using unsupervised deep learning methods.

Inspired by the success of the STN [32] in other affine image registration methods [13, 33, 34], the STN is adapted to the ultrasound volume registration problem to estimate the transformation parameters between an image pair. Encouraged by the state-of-the-art performance of the PLS-Net [24], a modified version of their DRDB is used in an encoder to extract multi-scale context features in the TEE recordings.

Because of the limited amount of training data available and GPU memory restrictions, both training and prediction is done in a patch-based manner. To check the feasibility of training and prediction on entire high-resolution recordings with respect to memory consumption, mixed precision training is evaluated. To examine the full potential of the image registration method, different preprocessing algorithms are evaluated, together with a comparison of registration on end-diastolic and end-systolic frames.

This master's thesis is a continuation of a project thesis that started in the fall of 2019. The theory chapter is an extended version of the one in the project. In addition, we present theory on the human heart and ultrasound imaging. Residual learning and dense connectivity is added to the section on deep learning and a thorough explanation of the Procrustes analysis is provided at the end of the theory chapter.

Differently from the project, we present a completely new network with separate pipelines and an improved bending penalty is added to the training. We also evaluate the method with respect to several preprocessing algorithms, and at different time-points along the heart cycle (end-diastolic and end-systolic) and investigate mixed precision training.

1.3 Outline of Thesis

In this first section, the motivation for automatic ultrasound volume registration was covered along with a summary of previous efforts to do medical image registration. The theoretical background needed to follow the rest of this thesis is presented in Chapter 2, covering the

basics of the human heart, ultrasound imaging and deep learning. In Chapter 3, the proposed method for the problem is specified along with description of the data, preprocessing steps and model architecture. Chapter 4 presents the results of the automatic image registration. These results are discussed in Chapter 5 together with suggestions for future work, before a concluding summary is presented in Chapter 6.

The preprocessing algorithms that are used in this thesis are covered in the appendices. Appendix A covers bilateral filtering and Appendix B covers non-local means filtering.

2 | Theory

2.1 The Human Heart

The human heart is a muscular organ that is located within the thoracic cavity, medially between the lungs in the mediastinum, which is responsible for the distribution of blood inside the body [39]. As shown in Figure 2.1, the heart is made up of a right and left side that works together as a parallel pump. The left and the right side are separated by the septum, and they each have one *atrium* and one *ventricle*. Each of the upper chambers, the right and the left atrium, acts as receiving chambers and contracts to push blood to the lower chambers, the right and the left ventricle [39].

Deoxygenated blood returning from the body flows into the right atrium through the superior and inferior vena cava. From the right atrium, blood passes into the right ventricle through the tricuspid valve. The deoxygenated blood is then pumped into the lungs through the pulmonary arteries, where it receives oxygen. From the lungs, highly oxygenated blood flows through the pulmonary veins to fill the left atrium. The left atrium pumps blood through the mitral valve and into the left ventricle, which in turn pumps oxygenated blood into the aorta and out to the body [39, 40].

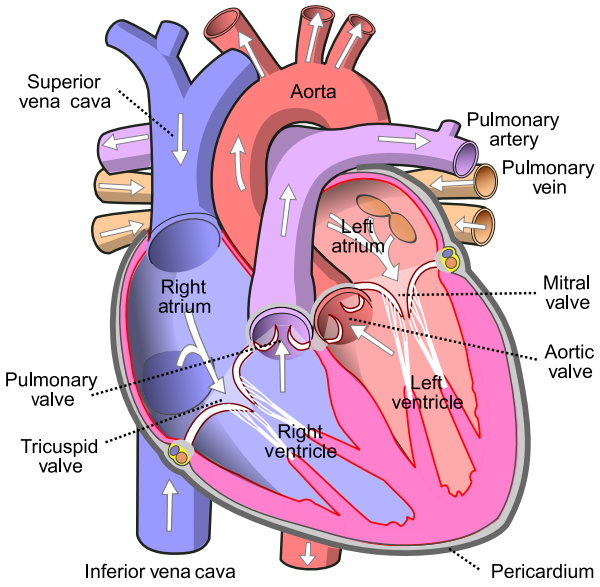


Figure 2.1: Illustration of the cardiac structure. White arrows show the normal direction of blood flow. Illustration: "Diagram of the human heart" by Wapcaplet[1], licensed under CC BY-SA 3.0 (`https://creativecommons.org/licenses/by-sa/3.0/`)

[1] `https://commons.wikimedia.org/wiki/File:Diagram_of_the_human_heart_(cropped).svg`

The cardiac cycle includes two phases which are referred to as *diastole* and *systole*, and they correspond roughly to relaxation and contraction of the heart, respectively [39]. The systolic phase begins when the ventricular pressure exceeds the atrial pressure and the tricuspid and mitral valves closes. This point in the cardiac cycle is referred to as end-diastole (ED), and is where the ventricular volume is greatest. With the atrioventricular valves closed, the ventricles rapidly contracts and the ventricular pressure increases. When the blood pressure within the ventricle is higher than within the arteries, the aortic and pulmonary valves opens and blood flows through the arteries. When the ventricular pressure is lower than the pressure in the arteries, the semilunar valves close. Closing of the aortic valve marks the end of the systolic phase and beginning of the diastolic phase, referred to as end-systole (ES). In the cardiac cycle, this is where the ventricular volume is lowest. Now the atria contracts while the ventricles relax. The atrial pressure exceeds that of the ventricles and the atrioventricular valves open again to allow for filling of the ventricles before the systolic phase begins again [41, 42]. Figure 2.2 shows a Wiggers diagram of the cardiac cycle events occurring in the left ventricle. Aortic, ventricular and atrial pressure is shown, together with the evolvement of the ventricular volume.

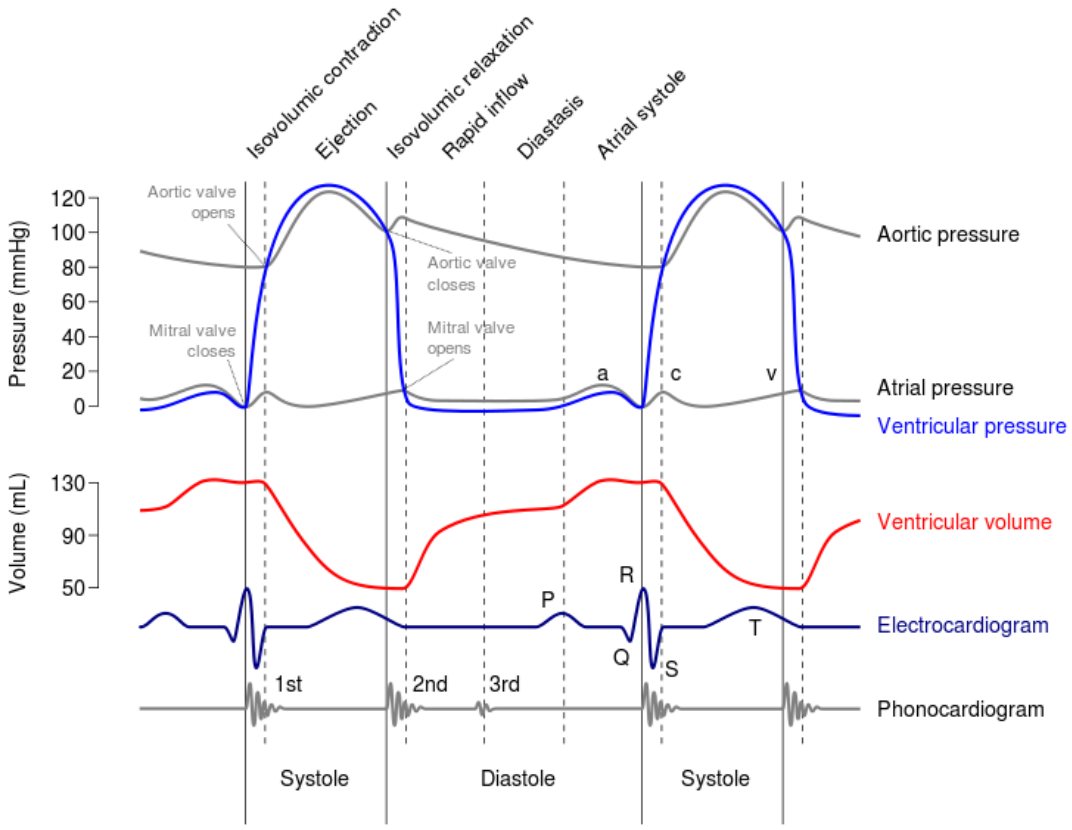


Figure 2.2: Wiggers diagram, showing the relation between blood pressure, ventricular volume and other measurements in the cardiac cycle. Illustration: "Wiggers Diagram" by DanielChangMD revised original work of DestinyQx; redrawn as SVG by xavax[2], licensed under CC BY-SA 2.5 (`https://creativecommons.org/licenses/by-sa/2.5/`)

[2] `https://commons.wikimedia.org/wiki/File:Wiggers_Diagram.svg`

2.2 Ultrasound Imaging

Ultrasound imaging is a widely used diagnostic tool, used in medical disciplines such as cardiology, obstetrics, gynecology, surgery, pediatrics, radiology, and neurology [43]. It is based on echoes produced by reflection of ultrasound waves at tissue boundaries and scattering from small irregularities within tissues [44]. Different ultrasound modes can be produced both in two and three dimensions in addition to time. In this thesis, *B-mode* images, where B is for brightness, are covered.

To form a 2D B-mode image, an ultrasound transducer transmits short pulses of ultrasonic waves into the patient. These pulses are directed along narrow beam-shaped paths called *scan-lines*. As the waves travel into the tissues of the body, they are reflected and scattered, generating echoes, some of which are received and detected by the transducer. Using the speed of sound in human tissue c and the depth d to the object that produced the echo at time t, the go and return time can be calculated as $t = 2d/c$. Rearranging, the depth d can be calculated as $d = ct/2$. This technique is known as echo ranging and is illustrated in Figure 2.3. At each scan line, the intensity of the received echo is plotted as a function of the distance to the probe, forming a *B-mode scan-line*. The final 2D B-mode image is formed from a large number of such B-mode scan-lines [44], and an example of such an image is shown in Figure 2.4.

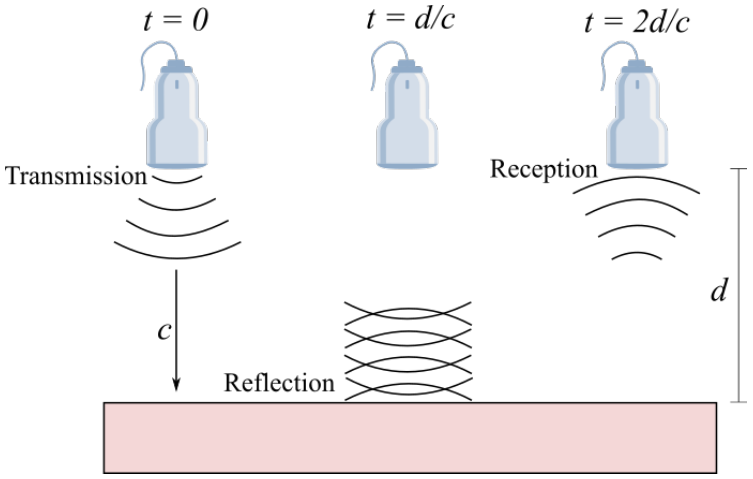


Figure 2.3: Echo ranging. The distance to an object is determined by the arrival time of the returning echo and the speed of sound.

In 2D imaging, only the thin slice of the patient can be viewed at any time. To form an impression of the 3D anatomy, the technician must mentally integrate many 2D images which is an ineffective and subjective process [43]. Three-dimensional ultrasound images can be generated to allow for arbitrary orientation of the image viewing plane within the volume. Generally, there are two types of systems that can generate 3D ultrasound images. Conventional transducers steer the beam within a 2D plane, and collection of 3D data is achieved by movement of the transducer across different orientations. In 2D-array transducers, the ultrasound beam is electronically swept through a 3D volume while the transducer is held still [44].

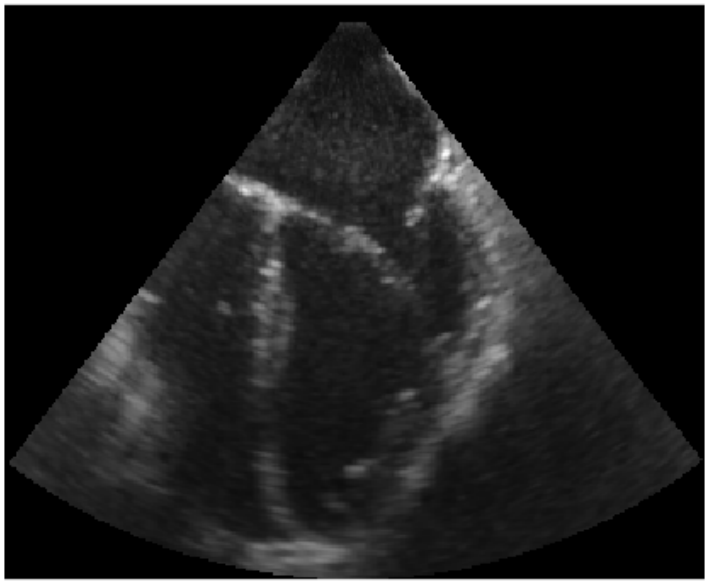


Figure 2.4: B-mode ultrasound image showing the mid-esophageal four-chamber view of the heart.

Medical ultrasound has several advantages compared to other popular imaging modalities such as CT and MR imaging. The use of non-ionizing radiation makes it a safe option as opposed to CT, it is less expensive and more portable than both CT and MR and it has the ability to produce images in real-time [45]. One of the main limitations of the technology is that the view is obstructed by bones and air due to the high reflection from such components.

2.2.1 Echocardiography

Echocardiography is an essential ultrasound imaging modality to assess cardiac function and is done in one of two ways. They differ in the invasiveness of the procedure and the placement of the probe, and is illustrated in Figure 2.5. Transthoracic echocardiography (TTE) is performed by placing the ultrasound probe on the patient's chest, making it a non-invasive, easy to set up procedure. However, TTE suffers from noise from the lungs and ribs and must be held still by an examiner during imaging and as such is not well suited for surgical procedures.

Transesophageal echocardiography (TEE) is an invasive procedure, where a specialized ultrasound probe is passed into the patient's esophagus. In the human body, the esophagus passes immediately posteriorly to the left atrium [40], which enables ultrasound imaging of the heart without acoustic obstructions from the lungs or ribs. In addition, the shorter distance from the probe to the heart facilitates the use of higher frequencies, yielding a higher spatial and temporal resolution in the image. However, insertion of the probe through the esophagus is very uncomfortable for the patient and usually requires general anesthesia or conscious sedation, and is therefore mostly used when TTE imaging is not sufficient.

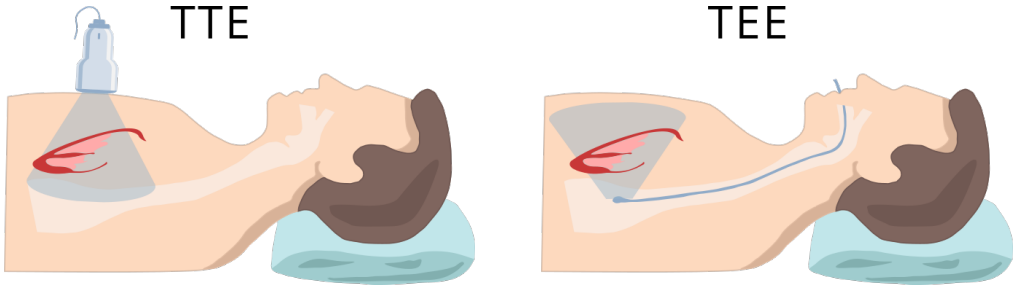
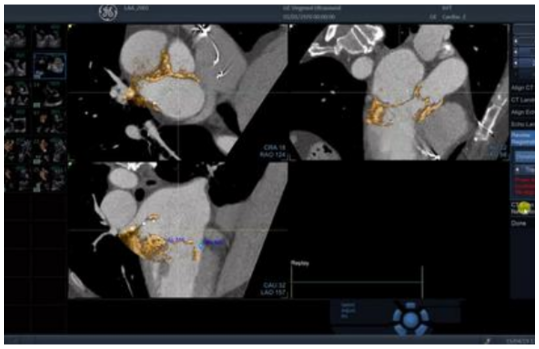


Figure 2.5: Probe placement in transthoracic echocardiography (TTE) and transesophageal echocardiography (TEE). Illustration: Redrawn from the original[3] as SVG by the author.

TEE has many uses in clinical practice, which is generally divided into diagnostic and intraprocedural TEE. Diagnostic TEE is performed in situations where the results of TTE are non-diagnostic or expected to be non-diagnostic and in critically ill patients [46]. Intraprocedural TEE is used for both preprocedural planning and for monitoring instrument manipulation during interventional procedures in the catheterization laboratory [8, 10, 46]. Real-time 3D TEE has become an invaluable monitoring adjunct for operative and percutaneous procedures, particularly in the field of transcatheter based interventions, as it provides crucial real-time information to help guide and assess procedural results [47]. Fusion of such real-time 3D TEE images with preoperative CT can be used not only to improve navigation and ultrasound-based guidance for interventional procedures, but also to improve diagnostic value [48]. When using fused perioperative CT and ultrasound during surgery, it is important to detect and correct for probe movement to keep the preoperative information in sync with the current ultrasound images. Figure 2.6a shows CT visualization of the heart for three different views and Figure 2.6b shows how fused CT and TEE ultrasound is visualized on the ultrasound scanner.

(a) CT images for the three different views of the heart represented in the top right corner for each view.



(b) Fused CT and TEE ultrasound as it is represented on the ultrasound scanner.

Figure 2.6: Image fusion of CT and TEE ultrasound. In the fused image, the yellow parts surrounding the grey areas represents the ultrasound recording. Screenshots are courtesy of GE Vingmed Ultrasound.

[3] `https://ww2.bangkokhospital.com/hearthospital/uploads/image/tte%20and%20tee.png`

2.3 Deep Learning

Machine Learning (ML) technology powers many aspects of the modern society: from web searches to targeted commercials on websites to identifying objects in images. Conventional ML methods require careful feature extraction from raw data to transform the data into a suitable representation from which the learning system can learn. ML methods are usually divided into three sub-categories: supervised-, unsupervised- and reinforcement learning.

In supervised learning, training data is fed to the learning algorithm together with the ground truth of the data, called *labels*. During training, the model makes a prediction from the training data it is fed. The prediction is compared with the label, resulting in some score which is used to update the parameters of the model to make it predict closer to the ground truth.

Unsupervised learning is a method where the training data is unlabeled. When the model makes a prediction, the score is only based on the training data, and a similarity metric between the training data and the prediction is used to update the parameters.

Reinforcement learning is a different approach than the aforementioned methods. The learning system, which in the context is called an *agent*, observes an *environment* and performs an action. In return it gets reward (positive or negative) based on the action. It is not explicitly told how to perform the task; it must learn the best *policy* itself. A policy defines what an agent should do when in a given situation [49].

Deep learning (DL) is a sub-field of machine learning, where simple non-linear modules are composed to transform the representation at one level into a representation at a higher, more abstract level [50]. Combining many such transformations can facilitate the learning of very complex functions.

2.3.1 Deep Feed-forward Neural Network

An Artificial Neural Network (ANN) is formed from layers of non-linear computing units, called *neurons*. ANNs that have no feedback connections, in which outputs of the model are fed back to itself are called feedforward neural networks. It is possible to design networks with such feedback connections, called Recurrent Neural Networks, but they are not within the scope of this thesis and will not be covered any further. Every ANN has an input layer where the inputs are known, an output layer where the outputs can be observed and one or more *hidden* layers. When an ANN has two or more hidden layers, it is called a Deep Neural Network (DNN) [49, 51].

Each neuron in a feedforward network computes a sum of products before the sum is activated at the output. For a given input x with a weight w, the output h of the i-th neuron in a layer is given as

$$h_i = f(\boldsymbol{w_i}^T \boldsymbol{x} + b_i) \tag{2.1}$$

where b is an added bias and f is the non-linear activation function. Commonly, sigmoid, tanh or rectified linear units (ReLUs) are used as activation functions, where experimental results suggest that the latter outperforms the other two in DNNs [51]. The computation is associated with a directed acyclic graph [52], which is illustrated in Figure 2.7.

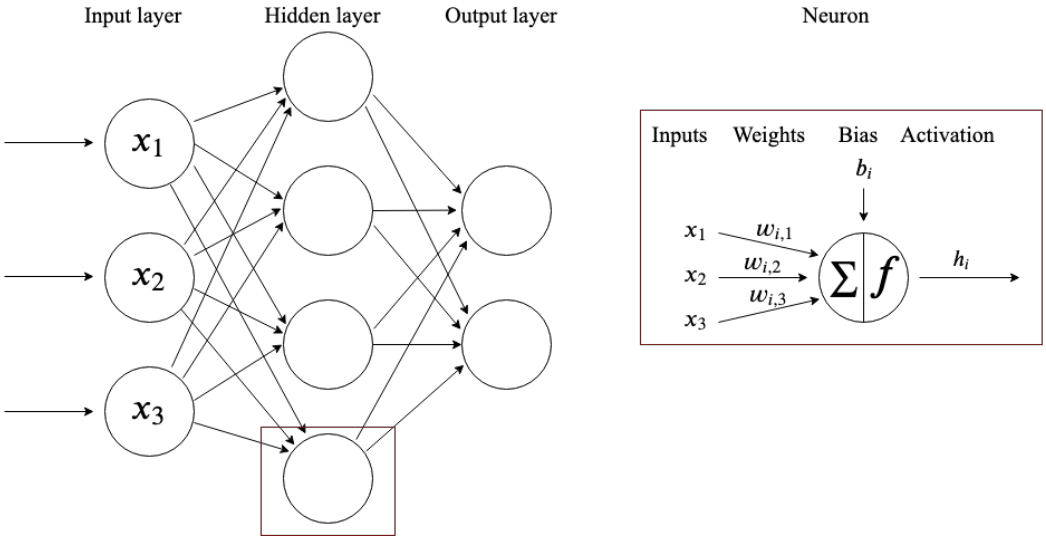


Figure 2.7: A feedforward neural network with three three neurons in the input layer, four hidden neurons and two output neurons. One of the neurons in the hidden layer is highlighted to show how computations are made within each neuron.

When every neuron in a layer is connected to all neurons in the next layer, such as in Figure 2.7, they are referred to as *fully connected* (FC) layers. FC networks are the simplest form of neural networks, and are used for tasks such as regression and classification. Although Figure 2.7 shows a FC network, feedforward networks are not necessarily fully connected, and such a network is covered next.

2.3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) emerged from study of the brain's visual cortex system, and the networks have been used for image recognition since the 1980s [49]. Although proven successful in many applications such as document recognition [53], they were largely forsaken by the computer-vision community until the ImageNet competition in 2012 [50]. After deep convolutional networks were applied to the dataset and almost halved the error rates from the competitors [54], CNNs became the dominant approach for almost all recognition and detection tasks.

Convolutional neural networks, named after the convolution operation, are a different kind of ANN than fully connected networks that we discussed in the previous section. The main difference is that a layer l in a CNN is only connected to a subset of pixels in the previous layer l_{-1}. By having several layers, this allows for the first hidden layers to adapt to low-level features and then assemble them into higher-level features in the following layers [49].

In digital image processing, 3D convolution is used to perform spatial filtering of a 3D image by moving the center of a 3D filter over the image, computing the sum of products at each location. Such filters are referred to as convolutional *kernels* [51]. During training, a CNN finds the most suitable filters for the task at hand and learns to combine them to obtain more complex patterns [49]. The output of the spatial filtering is called a *feature map*.

For an input feature map I and a three-dimensional kernel K, the convolution value at any point (x, y, z) in the filtered image H is given by [51]

$$H_{x,y,z} = \sum_{l} \sum_{m} \sum_{n} K_{l,m,n} \cdot I_{x+l,y+m,z+n} + b, \tag{2.2}$$

where (k, l, m) span the dimensions of the kernel, (x, y, z) span the dimensions of the input and b is the bias. Note that the operation presented in (2.2) is actually the cross-correlation, which is how convolution is implemented in most deep learning libraries. This is equivalent to the convolution operation if the kernel is flipped [52].

When repeated for all locations in the input image, the process described in (2.2) results in a 3D set of values that are stored in the next feature map [51]. This can be viewed as sliding the kernel over the input image, and is illustrated in Figure 2.8. Several feature maps stacked together are collectively known as a *convolutional layer*. For an input feature map of size $L \times M \times N$ and a convolutional kernel of size $K \times K \times K$, the filtered feature map will be of size $L - (K - 1) \times M - (K - 1) \times N - (K - 1)$. The input feature map may be zero-padded to keep volume dimensions.

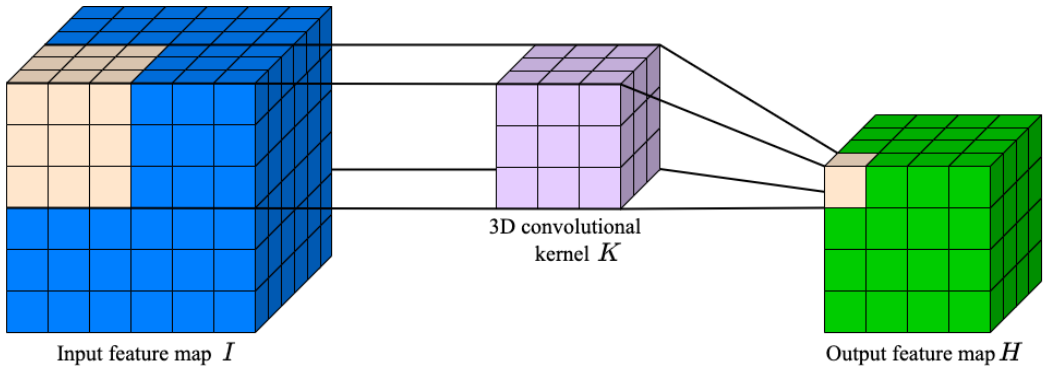


Figure 2.8: Filtering of 3D input feature map I with a 3×3×3 convolutional kernel K to produce the filtered 3D output feature map H.

When several convolutional layers are stacked together, the *receptive fields* of the resulting pixels increases. The result is that each successive layer is composed of more abstract features, with the first layers usually ending up as edge and corner detectors. Later layers combine the first layers into higher level features.

2.3.3 Spatial Transformers

Although CNNs define a powerful set of models, they are limited by the lack spatial invariance to the input data. Jaderberg et al. [32] introduced the *Spatial Transformer Network* (STN), where a Spatial Transformer (ST) is a learnable module which explicitly allows for spatial manipulation of data within the network. When the ST is dropped into a network, the network is called a Spatial Transformer Network.

The STN is split into three parts, and illustrated in Figure 2.9. The localization network takes an input feature map $U \in \mathbb{R}^{HxWxDxC}$ with height H, width W, depth D and C channels and outputs the parameters θ. These are the parameters of the transformation $\mathcal{T}_\theta$, which is to be applied to the feature map: $\theta = f_{loc}(U)$.

The localization network is not limited to being a fully-connected network or a convolutional network, but it should include a final regression layer to produce the transformation parameters. For affine transformation of a 3D input feature map, the final regression layer should output 12 transformation parameters.

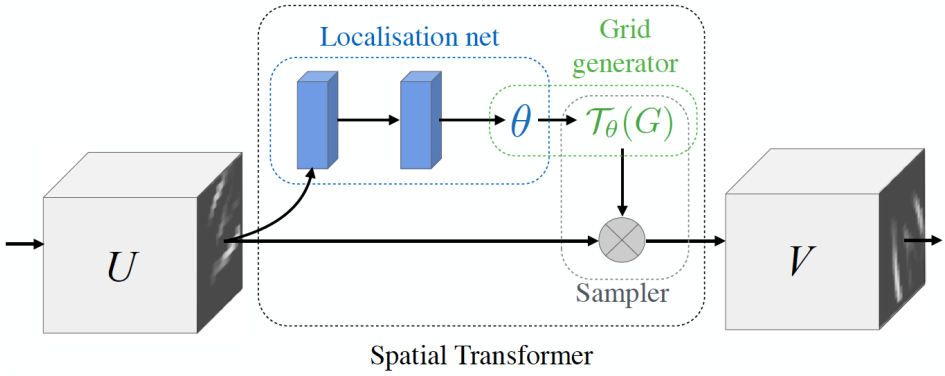


Figure 2.9: Spatial Transformer Network. [32]

To perform a warping of the input feature map U, we need to define a sampling grid. In general, the output pixels are defined to be on a regular grid $G = \{G_i\}$ of pixels $G_i = \left(x_i^t, y_i^t, z_i^t\right)$, which forms the output feature map $V \in \mathbb{R}^{H'xW'xD'xC'}$, where H', W' and D' are the height, width and depth of the sampling grid and C is the number of channels, equal to the number of channels in the input.

If we assume that the transformation $\mathcal{T}_\theta$ is a 3D affine transformation, the pointwise transformation is

$$\begin{pmatrix} x_i^s \\ y_i^s \\ z_i^s \end{pmatrix} = \mathcal{T}_\theta(G_i) = A_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ z_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} \\ \theta_{31} & \theta_{32} & \theta_{33} & \theta_{34} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ z_i^t \\ 1 \end{pmatrix} \tag{2.3}$$

where (x_i^s, y_i^s, z_i^s) are the source coordinates of the input feature map that define the sampling

points, (x_i^t, y_i^t, z_i^t) are the target coordinates in output feature map and A_θ is the affine transformation matrix. The affine transformation matrix has 12 degrees of freedom (DOF) which allows for spatial cropping, translation, rotation, scaling and skew.

To perform a spatial transformation of the input feature map, the sampler takes the set of sampling points $\mathcal{T}_\theta(G)$ from the grid generator and the input feature map U, and produce the sampled output feature map V. If a bilinear sampling kernel is used, the value at a particular pixel in the output V can be written as

$$V_i^c = \sum_n^H \sum_m^W \sum_l^D U_{nml}^c \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|) \max(0, 1 - |z_i^s - l|) \qquad (2.4)$$

where U_{nml}^c is the value in channel c at location (n, m, l) in the input feature map and V_i^c is the value of pixel i in channel c of the output feature map.

2.3.4 Training Neural Networks

Training a neural network is an optimization problem that is solved with gradient-based methods. A *cost function* $J(\theta)$ is minimized to obtain the optimal parameters θ for the network. The cost function is defined as the average of the *loss functions* in a training set, where the loss function computes the error for a single training example. The loss function reflects the goal of the training, and for unsupervised learning, it is usually a similarity metric that is maximized during optimization. Learning algorithms uses gradient vectors to make a step in the direction that decreases the cost function and updates the parameters stepwise. This technique is known as gradient descent. Gradients are calculated first for the last layer with respect to the second to last layer's output. Then, the second to last layer's gradients are calculated with respect to the previous layer's output and so on, forming the basis of the backpropagation algorithm. This process continues all the way to the input layer before the parameters are updated. When a network has updated the parameters for all samples once, is defined as one *epoch.*

When training a neural network in an unsupervised fashion, the available data is split into a training set and a validation set, with the training set usually being 80% of the available data set. The validation set is kept unseen from the network during training in order to evaluate the model. Throughout the training process, loss on the training set and validation set is monitored to diagnose two issues that can arise in any DL model: underfitting and overfitting. For the model to perform well on new data, training and validation set loss should be as low as possible and the distance between the two should be small. If the model is stuck at a sub-optimal loss value for both training and validation set loss, it is said to be underfitting. Usually when a model underfits, it is too simple for the optimization task. Overfitting occurs when the model specializes too much on the training data, causing it to generalize poorly on the validation set and the validation set loss starts to increase after a number of training steps. This happens when the model is to complex or if the available data is limited. Different regularization techniques such as *dropout* or *data augmentation* are commonly used to reduce

the risk of overfitting. Figure 2.10 shows learning curves for a model that is underfitting overfitting and optimally fitting from left to right.

Modern neural networks have millions of parameters, depending on the dataset they are applied to [55]. This makes training a time-consuming task, infeasible for regular computers. Instead, deep neural networks are usually trained on powerful GPUs, which provides speedups of 2-24 times that of regular central processing units (CPUs) [56]. Due to memory limitations of the GPUs and the large datasets required for training a model, the training set is divided randomly into non-overlapping batches. If the mini-batch size is chosen to be a power of two, further performance gains can be made because of the computer architecture of the GPUs.

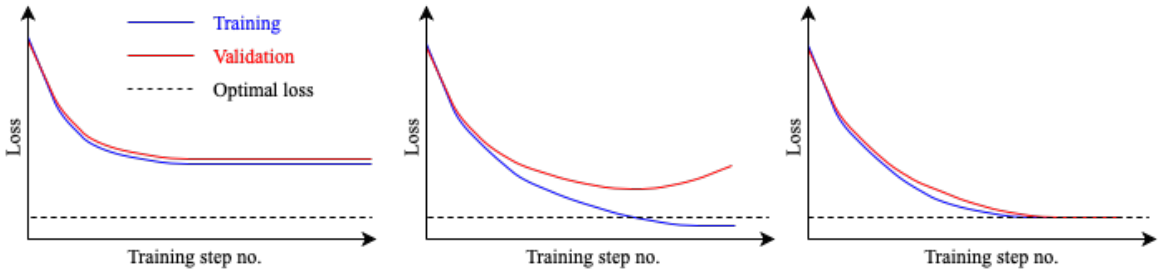
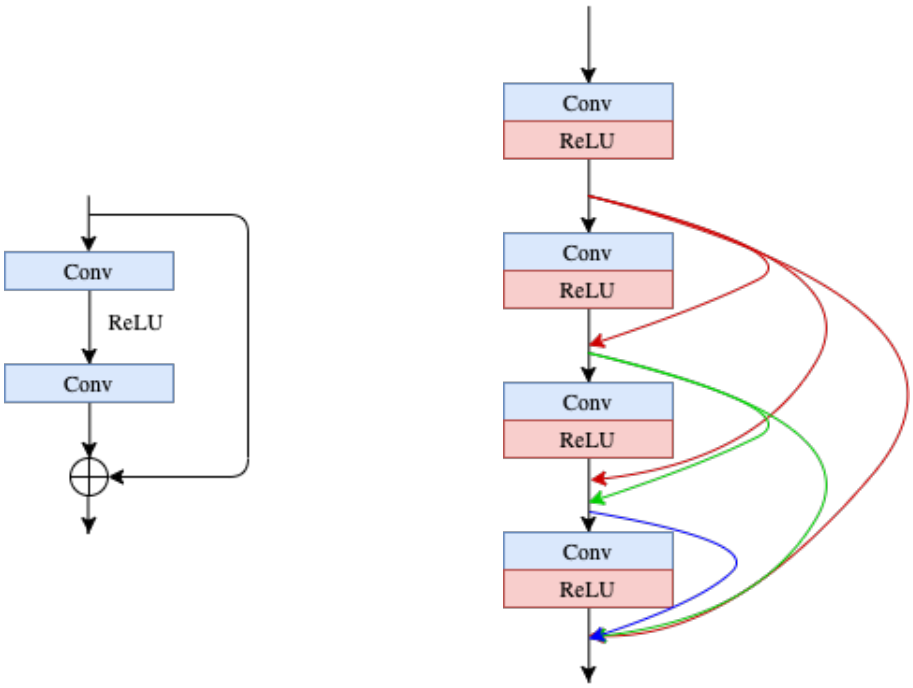


Figure 2.10: Loss curves showing an underfitted model, an overfitted model and an optimally fitted model from left to right.

2.3.5 Residual Learning and Dense Connectivity

As CNNs become increasingly deep, the input information and gradients pass through many layers. If the network is deep enough, the gradients can vanish during backpropagation, also known as the vanishing gradient problem. In the worst case, this may completely stop the DNN for further training.

To overcome the vanishing gradient problem, He et al. [57] introduced the concept of residual learning in their ResNet. Residual learning uses identity shortcut connections, which promotes gradient propagation by connecting the output of a previous layer to the output of a future layer, while skipping one or more layers in between. The connection is done through elementwise addition of the outputs and is shown in Figure 2.11a. Another way of overcoming the problem was introduced in DenseNet by Huang et al. [58]. By connecting each layer to every other layer in a feed-forward fashion, they obtained significant improvements over the state-of-the-art on object recognition benchmark tasks. In the densely connected CNN, for each layer, the feature maps of all preceding layers are used as inputs, and its own feature maps is used as inputs to all subsequent layers. Unlike ResNet, DenseNet uses channel-wise concatenation of feature maps, as shown in Figure 2.11b. Densely connected CNNs benefit from a strong gradient flow due to the dense connections, alleviating the vanishing gradient problem. They also strengthen feature propagation, encourage feature reuse and reduce the number of parameters substantially [58].



(a) Residual learning **(b)** Dense connectivity

Figure 2.11: Residual learning and dense connectivity. For residual learning, the connection is done through elementwise addition whereas dense connectivity uses channel-wise concatenation.

2.4 Procrustes Analysis

A common need in the field of computer vision is to compute the 3D rigid body transformation that aligns two sets of features which have a known one-to-one correspondence [59]. Procrustes analysis is a form of statistical shape analysis used to analyze the distribution of two such sets of features. In practice, point features are most commonly used, and a set of point features is referred to as a point cloud. Closed form solutions are robust and efficient and are popular solutions for aligning point clouds.

If we assume that there exist two correspondent point clouds $\{m_i\}$ and $\{d_i\}$, $i = 1..N$, such that they are related by

$$d_i = \mathbf{R}m_i + \mathbf{T} + \mathbf{V}_i \tag{2.5}$$

where $\mathbf{R}$ is a 3×3 rotation matrix, $\mathbf{T}$ is a three-dimensional translation vector and $\mathbf{V}_i$ is a noise vector, the optimal transformation $[\hat{\mathbf{R}}, \hat{\mathbf{T}}]$ that maps $\{m_i\}$ onto $\{d_i\}$ can be solved by minimizing a least square error criterion given by

$$\Sigma^2 = \sum_{i=1}^{N} ||d_i - \hat{\mathbf{R}} m_i - \hat{\mathbf{T}}||^2. \tag{2.6}$$

As a consequence of the solution to (2.6), the point clouds $\{m_i\}$ and $\{d_i\}$ should have the same centroid. Using the new constraint, and defining

$$\begin{aligned} \bar{d} &= \frac{1}{N}\sum_{i=1}^{N} d_i \qquad & d_{c_i} &= d_i - \bar{d} \\ \bar{m} &= \frac{1}{N}\sum_{i=1}^{N} m_i \qquad & m_{c_i} &= m_i - \bar{m} \end{aligned} \tag{2.7}$$

the equation in (2.6) can be rewritten and reduced to

$$\begin{aligned} \Sigma^2 &= \sum_{i=1}^{N} ||d_{c_i} - \hat{\mathbf{R}} m_{c_i}||^2 \\ &= \sum_{i=1}^{N} (d_{c_i}^{\mathrm{T}} d_{c_i} + m_{c_i}^{\mathrm{T}} m_{c_i} - 2 d_{c_i}^{\mathrm{T}} \hat{\mathbf{R}} m_{c_i}). \end{aligned} \tag{2.8}$$

This equation can be minimized by maximizing the last term, which is equivalent to maximizing Trace($\hat{\mathbf{R}}\mathbf{H}$), where $\mathbf{H}$ is a correlation matrix defined by

$$\mathbf{H} = \sum_{i=1}^{N} m_{c_i} d_{c_i}^{\mathrm{T}}. \tag{2.9}$$

If the singular value decomposition (SVD) of $\mathbf{H}$ is given by $\mathbf{H} = \mathbf{U}\Lambda\mathbf{V}^{\mathrm{T}}$, the optimal rotation matrix $\hat{\mathbf{R}}$ that maximizes the trace is

$$\hat{\mathbf{R}} = \mathbf{V}\mathbf{U}^{\mathrm{T}}. \tag{2.10}$$

The optimal translation vector aligns the centroid of $\{d_i\}$ with the rotated centroid of $\{m_i\}$, and can be expressed as

$$\hat{\mathbf{T}} = \bar{d} - \hat{\mathbf{R}}\bar{m}, \tag{2.11}$$

where R is guaranteed to be orthonormal. The Procrustes method can also deal well with measurement noise.

3 | Materials and Method

The objective of the image registration task is to find a spatial transformation that aligns an image pair, and can be formulated as an optimization problem, where the goal is to maximize the similarity between the image pair.

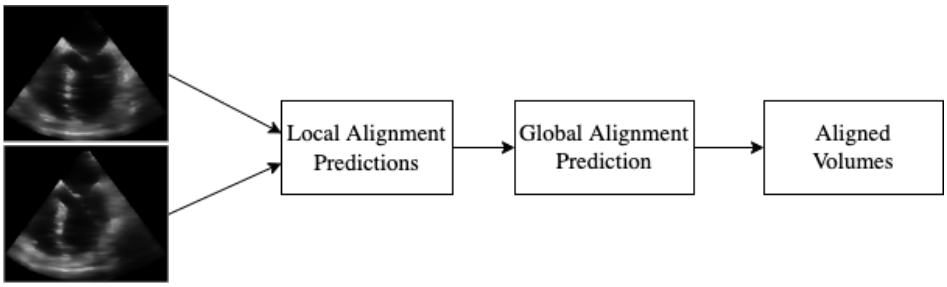


Figure 3.1: Proposed pipeline for ultrasound volume registration of TEE images.

In Figure 3.1, the proposed pipeline for ultrasound volume registration is proposed. A DNN was used to estimate a transformation field between local patches of the full volume. Using the local predictions from the set of patches, a closed form statistical shape analysis was used to estimate a global transformation field that finally aligned the volumes.

3.1 Dataset and Preprocessing

3.1.1 Patient Data

To evaluate the method, 3D TEE B-mode images were obtained from 28 patients by trained cardiologists, using GE Vivid E95 and E9 systems with a 6VT-D probe (GE Vingmed Ultrasound, Horten, Norway). All the patients were examined in the clinic for diagnostic purposes. For each patient, a minimum of two recordings with at least one cardiac cycle were captured. The pixel intensity was recorded in the range [0, 255]. The resolution of the images was in the range of $171 \times 171 \times 171$ to $313 \times 257 \times 313$ and the frame rate in the range of 5 to 24 frames per second. No selection of patients was made, and all patient data was anonymized before analysis. To facilitate processing of the data, the volumes were converted from the proprietary DICOM format to 3D volumes by applying a polar-Cartesian transform on raw B-mode lines. All datasets were resampled to isotropic volumes with a voxel size of $0.7 \times 0.7 \times 0.7$ mm. Data from two patients was omitted due to insufficient quality and different resolution between the volumes.

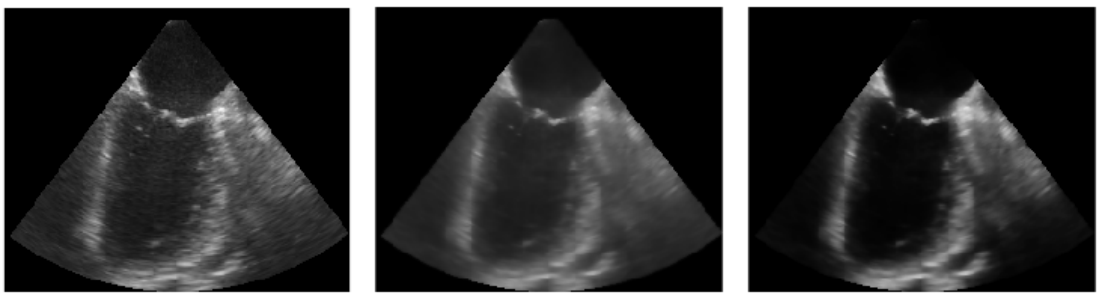
During acquisition, the volumes were rotated by the operators to provide a desired field of view. To assess the performance of the volume registration method properly, the volumes were manually realigned such that the 4-chamber view occurred at 0 degrees.

3.1.2 Data Preprocessing

For the ultrasound volumes to be suitable for registration, they should be low in noise and key areas in the cardiac structure should be enhanced. Motivated by these properties, two preprocessing steps were applied to the volumes. First, the volumes were filtered with a smoothing filter to reduce the amount of speckle in the volumes. Second, a custom 1D transfer function was applied in order to attenuate the gray values inside the heart cavity.

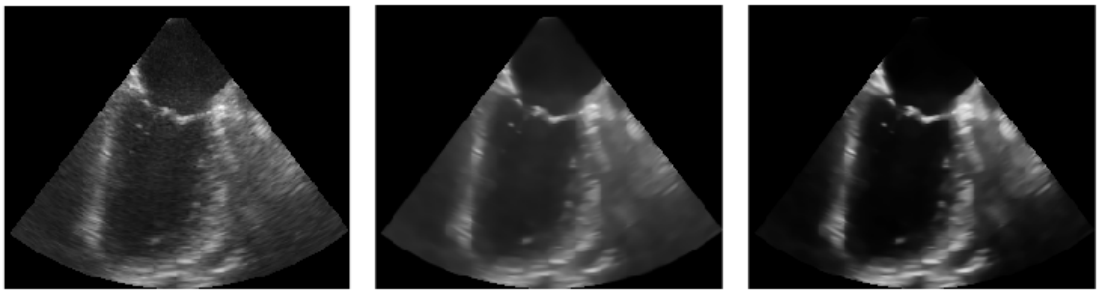
To evaluate the registration method, two different smoothing filters were applied to the volumes: (i) a bilateral filter and (ii) a non-local means filter (NLMF). Bilateral filtering [60] smooths images while still preserving sharp edges in the filtered image by using a non-linear combination of nearby image values. It uses a weighted average of intensity values from nearby pixels to replace pixels in the original image. The preprocessing pipeline with bilateral filtering is shown in Figure 3.2 and details on bilateral filtering can be found in Appendix A. The range parameter σ_r and spatial parameter σ_d were empirically set to 40 and 5 respectively. Unlike local mean filters such as the bilateral filter, the NLMF [61] takes a mean of all pixels in the image, weighted by how similar they are to the target pixel. This results in greater clarity and less loss of details in the filtered image. Due to the complexity of the NLMF algorithm, the search window is generally restricted to a smaller block centered around the pixel instead of the whole image. Figure 3.3 shows the preprocessing pipeline using the NLMF, and details on the algorithm can be found in Appendix B. The search window block size was empirically set to 45 to give a good trade-off between computation time and noise reduction.

Smoothing of a volume with dimensions $230 \times 200 \times 230$ took 78s with bilateral filtering and 808s with the NLMF on an Intel(R) Core i5(R) CPU @ 2.40GHz Dual-Core with 8GB RAM. However, as both implementations are CPU based, processing times can be significantly reduced by using a GPU.



(a) Original frame **(b)** Bilateral filtered frame **(c)** Custom 1D transfer function processed frame

Figure 3.2: Image preprocessing pipeline with bilateral filter as the smoothing filter.



(a) Original frame **(b)** NLMF filtered frame **(c)** Custom 1D transfer function processed frame

Figure 3.3: Image preprocessing pipeline with NLMF as the smoothing filter.

3.2 Method

3.2.1 Volume Preprocessing

The performance of deep learning methods is highly dependent on the amount of data available. In this project, only 28 TEE recordings were available, where two of the recordings had to be discarded. In order to generate more samples, the volumes were divided into smaller subvolumes which will be referred to as *patches*. Patching of the volumes have the additional advantage that it enables training with deeper networks without running out of GPU memory, which in previous projects have been an issue. The volumes are divided into isotropic patches with a given *stride*, where a stride equal to zero meaning full overlap and a stride equal to the patch size meaning no overlap.

3.2.2 Local Prediction Network

To perform the automatic volume registration, the approach for affine registration used in [13, 34], where they used STN [32] to affinely register volumes, was adapted to this application. Figure 3.4 shows a visualization of the training procedure of the registration method. A fixed volume F and a moving volume M forms an image pair that is passed into the registration network in separate pipelines as patches. The registration network produces deformation grids that are interpolated using bilinear interpolation and used to warp the moving patches. Then the warped patches are compared to the fixed patches using a loss function that combines normalized cross-correlation with a bending penalty. This allows the network to be trained in a fully unsupervised fashion. Note that the moving volume is captured at the subsequent frame with respect to the reference volume, meaning that if the reference volume is captured at the end-diastolic frame, the moving volume is captured at the subsequent end-diastolic frame.

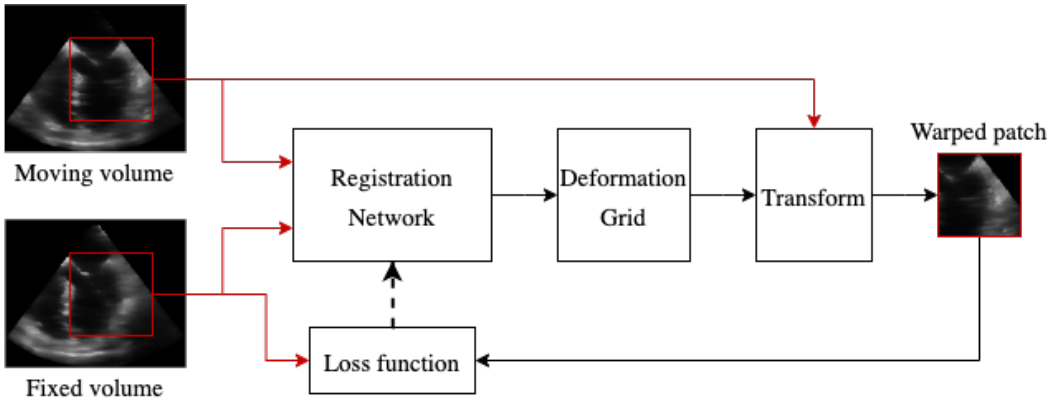


Figure 3.4: Training procedure for the image registration network. An image pair is passed into the registration network as patches. The registration network produces deformation grids that are interpolated and used to warp the moving patches. Normalized cross-correlation is used together with a bending penalty to form the loss function, which gradients are used to update the parameters in the registration network.

The registration network is built up by (i) an encoder used to localize features in the volumes, and (ii) an affine regressor that is used to produce the output transformation parameters. Based on the success of the PLS-Net [24] and their dilated residual dense block (DRDB), a modified DRDB is adapted to the ultrasound volume registration problem and used as the basic building block in the encoder. The DRDB block uses progressively increased dilation to enlarge the receptive field, to enable segmentation of large volumes. However, regularly dilated convolutions are sufficient when working on smaller patches of a volume, thus increased dilation is not used in this method and we refer to the modified building block as just residual dense block (RDB). The structure of the RDB is shown in Figure 3.5. It consists of four $3 \times 3 \times 3$ convolutional layers followed by a $1 \times 1 \times 1$ convolutional layer and residual learning.

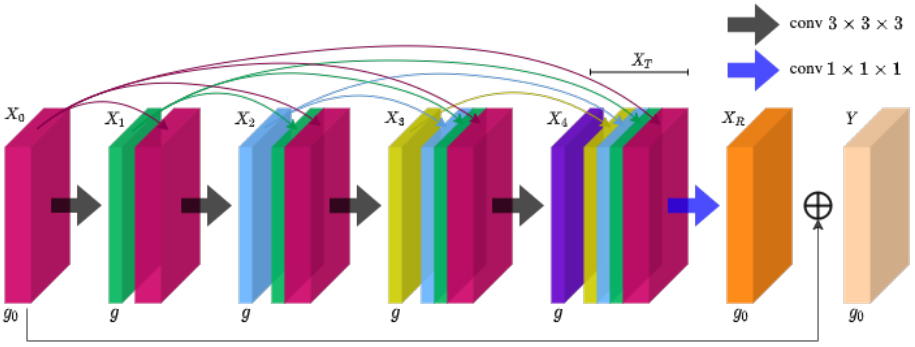


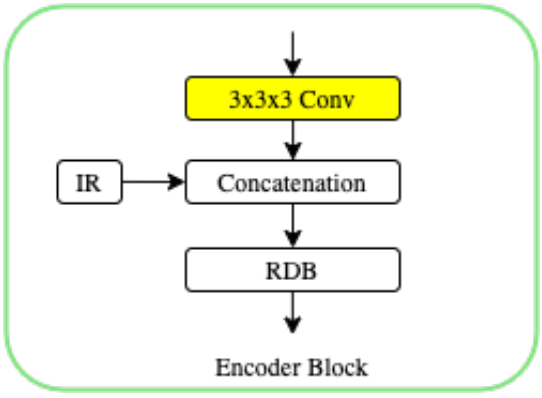
Figure 3.5: Residual Dense Block (RDB) architecture. Each block represents a 4D feature map tensor, with the fourth dimension being the number of channels. The growth rate g decides how many layers that are added to the information from the previous layers.

To capture multi-scale features, dense connectivity is introduced. The layers are connected such that the l-th layer of the RDB can be computed as $X_l = H([X_0, X_1, ..., X_{l-1}])$, where H is the function applied to the concatenated feature maps. If X_0 has g_0 feature maps, and

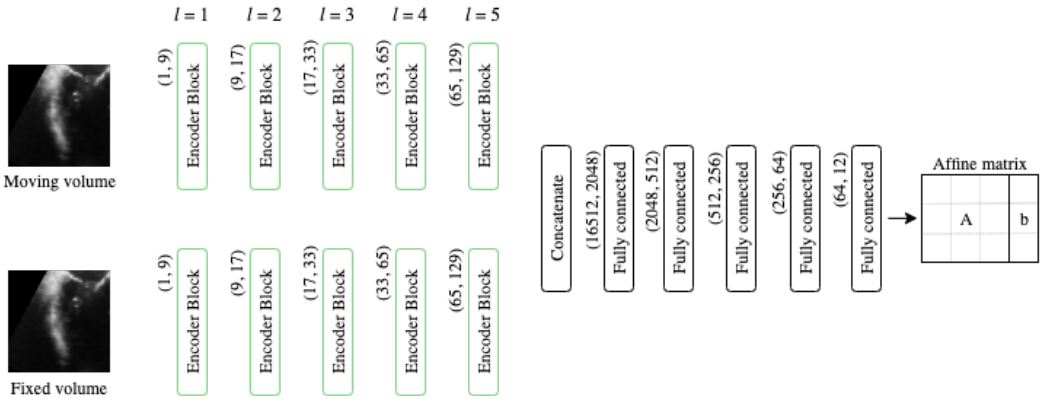
each successive layer generates g feature maps, the output X_T has $g_0 + 4g$ feature maps where g is referred to as the growth rate. A $1 \times 1 \times 1$ convolutional kernel is applied to the output to improve computational efficiency. Thus, the output of the RDB can be written as $Y = X_R + X_0$.

A straightforward implementation of the RDB requires a significant amount of GPU memory during training. Intermediate activations produced in the forward pass are usually stored in memory for backpropagation. These activations are responsible for much of the memory usage, but they are cheap to compute. By discarding the activations in the forward pass and recomputing them in the backward pass, a reduction in memory consumption from quadratic to linear is traded for a small increase in training time.

The encoder is built up by stacked encoder blocks, and the architecture of the encoder block is illustrated in Figure 3.6a. Concatenation follows the encoder before fully connected layers produce the affine transformation matrix, and the full network architecture is illustrated in Figure 3.6b.



(a) Encoder block



(b) Full network architecture

Figure 3.6: The architecture of the proposed registration network. The numbers on top of each encoder block denotes the number of input and output channels from each block, and the number on top of the fully connected layers denotes input and output features. The yellow box represents strided convolutions that perform downsampling and l indicates resolution level.

Each resolution level l in the encoder includes a $3 \times 3 \times 3$ strided convolution to downsample the volume and produce feature maps, input reinforcement (IR) to help retain spatial information throughout the encoding process and a RDB to capture multi-scale features. To mitigate spatial information loss, Lee et al. [24] introduced a IR scheme where a downsampled version of the input image is concatenated with the output of the strided convolution. The original image was downsampled with a factor of 2^l via trilinear interpolation, and the growth rate g of the RDB was empirically set to 8. At resolution level $l = 5$, the spatial resolution of the feature maps is reduced to $\frac{1}{32}$ of the input resolution. The output of each encoder pipeline is concatenated and passed through five fully connected linear layers, where the output of the last linear layer is the affine transformation matrix $\theta = Ax + b$, where A is the 3×3 rotation matrix and b is the three-dimensional translation vector.

All convolutional and linear layers in the network are followed by batch normalization (BN) and ReLU activations except for the last linear layer. Zero-padding was used in all convolutional layers to keep feature map dimensions. Readers interested in the source code for this thesis are referred to the Github repository[4].

3.2.3 Loss Function

Intuitively, a deep neural network is trained by minimizing the loss function. For the image registration task, we aim to maximize the image similarity, that is minimizing the image dissimilarity. In this experiment, we use the normalized cross-correlation (NCC) as the image similarity measurement, which is the cross-correlation confined to [0, 1]. It is a popular metric as it is robust to intensity variations in intra-modality images [26, 36]. Mathematically, the image similarity loss L_{sim} can be written as

$$L_{\text{sim}} = 1 - \text{NCC} = 1 - \frac{\sum_{x,y,z}(\boldsymbol{F} - \overline{F}_{x,y,z}) \cdot (\boldsymbol{M} - \overline{M}_{x,y,z})}{\sqrt{\sum_{x,y,z}(\boldsymbol{F} - \overline{F}_{x,y,z})^2 \cdot \sum_{x,y,z}(\boldsymbol{M} - \overline{M}_{x,y,z})^2}} \tag{3.1}$$

where F and M represents the fixed and moving image respectively and (x, y, z) spans the image dimensions. For intra-patient affine image registration problems, it is usually the case that the input images only need a small transformation before they are affinely aligned. The regularization loss L_{reg} introduced in [37] penalizes deviations of the composed affine transform from the identity matrix, and is written as

$$L_{\text{reg}} = \lambda_r(||A - I||_F^2 + ||b||_2^2) \tag{3.2}$$

where $||.||_F$ denotes the Frobenius norm and $\lambda_r \geq 0$ is an epoch dependent weight factor designed to be large at the beginning of training to constrain initial large transformations, before it gradually decays to zero. The weight factor is defined as

[4] `https://github.com/krisroi/us_volume_registration`

$$\lambda_r = \frac{C_r K_r}{K_r + \exp\left(n/K_r\right)} \tag{3.3}$$

where C_r is a constant, K_r controls the decay rate and n is the epoch count. The complete loss function L can then be written as

$$L = L_{\text{sim}} + L_{\text{reg}} \tag{3.4}$$

3.2.4 Global Volume Alignment

During prediction, the output of the registration network is an affine transformation matrix for each patch. To compute a global 3D rigid body transformation of these local transformations, the Procrustes method introduced in section 2.4 was used.

The affine transformation matrix and the corresponding position of each patch in the volume is used to create a point cloud m_i. Using the algorithm presented, the point cloud is mapped onto d_i, and the final output of the Procrustes method is an affine transformation matrix that optimally aligns the original point cloud with the transformed one.

3.2.5 Implementation

All volumes were divided into isotropic patches with size $128 \times 128 \times 128$ and a stride of 25. Patching of the volumes was done by utilizing the code provided in [62]. To avoid training on patches that covers the outside of the ultrasound sector and therefore contain no useful data, only patches containing more than 70% non-zero data were selected. Before generation of patches, the pixel intensity was normalized to be in the range of [0, 1]. Patching of the volumes generated around 1100 samples in total, depending on the filter type and whether end-diastolic or end-systolic frame was selected. During training, 20% of the patches were used for validation and the rest for training.

All convolutional and linear weights were initialized using He initialization [63], except for the final linear layer which was initialized to regress the identity matrix (zero weights, identity transform bias). The batch size used was 16 pairs of patches, and the network was optimized using the Adam optimizer [64]. Initial learning rate was set to 0.001 and divided by 10 every 25 epochs. For a fair comparison between results, training was ended after 60 epochs. Each training session with full precision required about 7.6 GB of GPU memory. The regularization parameters were set to $\{C_r = 6, K_r = 2\}$, empirically chosen from the results on the validation set.

The code was implemented on a remote Ubuntu 18.04.2 server with Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz, 64 GB RAM and a Nvidia Quadro P5000 GPU with 16 GB of RAM and 2560 CUDA cores, using the deep learning library PyTorch[5] v.1.4.0, CUDA v.10.0.130, cuDNN v.7.6.3_0 and Python 3. Network training was done on the GPU.

[5] `https://pytorch.org`

3.2.6 Validation Study

Four experiments were conducted for this thesis: (i) five-fold cross-validation to evaluate the proposed volume registration method, (ii) evaluation of the method with respect to which frame of ES and ED that is used for registration, (iii) evaluation of different filter types for preprocessing, and (iv) evaluation of mixed precision training to evaluate the feasibility of doing volume registration on larger volumes. Table 3.1 summarizes the details of each experiment.

Table 3.1: Detailed overview of the frame, filter type and precision used in each experiment.

Experiment	Frame	Filter type	Precision	tr/val samples
(i)	End-diastolic	Bilateral	Full	875/219
(ii)	End-systolic	Bilateral	Full	878/222
(iii)	End-diastolic	NLMF	Full	873/218
(iv)	End-diastolic	Bilateral	Mixed	875/219

Due to the limited amount of training data, five-fold cross-validation was used to ensure that the model is able to generalize well to previously unseen data. The patches were split into five non-overlapping folds, each of which consisted of 20% randomly selected patches. Each fold was then iteratively used once for validation while the remaining four folds were used for training.

Mixed precision training was done using the PyTorch extension Apex[6] with optimization level O2. Due to numerical instabilities with the default values in the Adam optimizer with mixed precision, the epsilon value was increased to $1e$-7.

[6] `https://github.com/nvidia/apex`

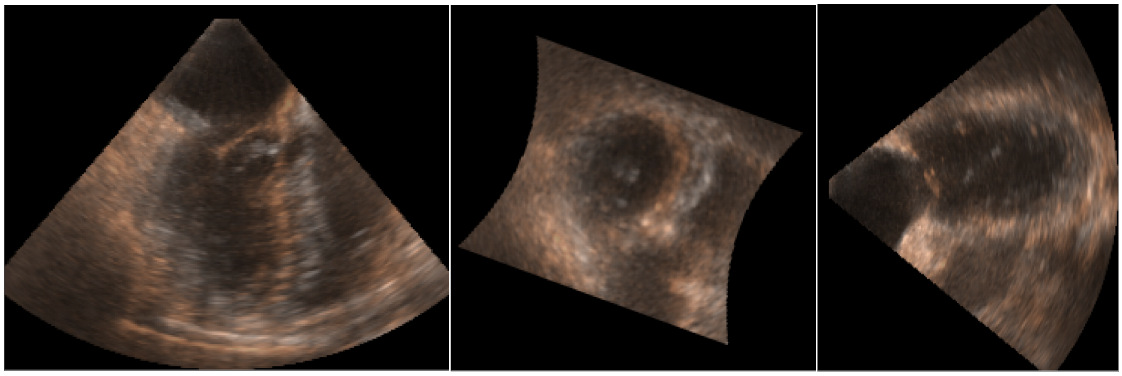
4 | Results

To evaluate the registration method and the additional experiments, pre-alignment values and visualization of the prediction sets are provided. These values and visualizations are used for comparison purposes and are therefore shown initially to avoid repetition. To be able to compare the two preprocessing steps without introducing any bias, prediction itself is done on preprocessed images while the global alignment is applied to the raw image. As such the NCC values that are used to evaluate the registration accuracy can be compared directly between experiments.

Table 4.1 summarizes end-diastolic and end-systolic NCC prior to alignment for the prediction sets. Figure 4.1 and 4.2 show pre-alignment end-diastolic views of prediction set 2 and 3 respectively, and Figure 4.3 and 4.4 show pre-alignment end-systolic views of prediction set 2 and 3 respectively. These sets were chosen for visualization as they represent the worst and best case after registration.

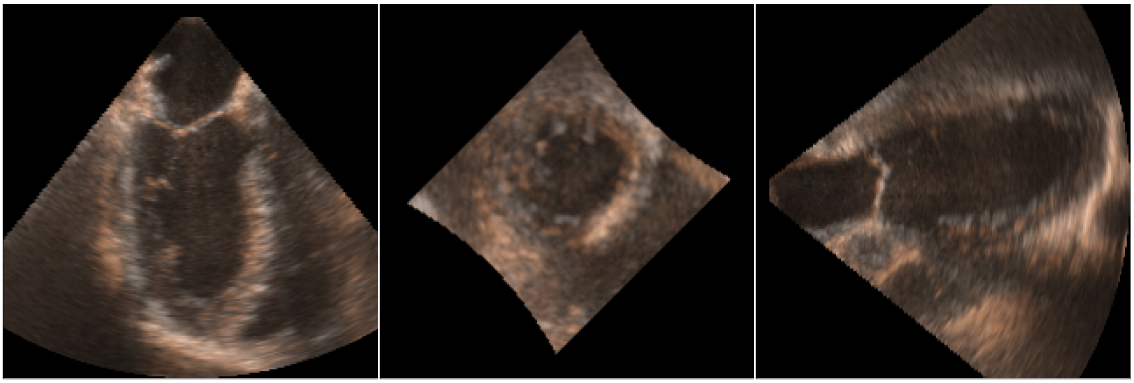
Table 4.1: Pre-alignment NCC values for the three prediction sets.

	NCC	
Frame	End-diastolic	End-systolic
Prediction set 1	0.8363	0.8599
Prediction set 2	0.8784	0.9028
Prediction set 3	0.8702	0.8716



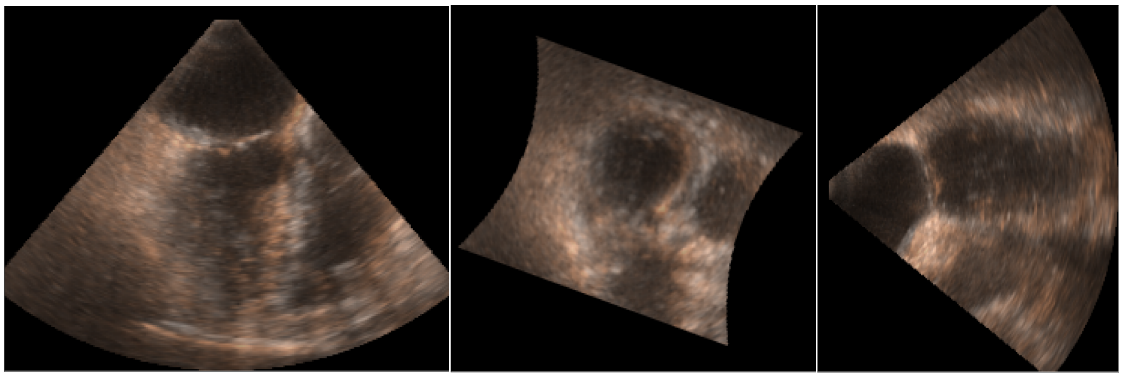
(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.1: Pre-alignment end-diastolic views of prediction set 2.



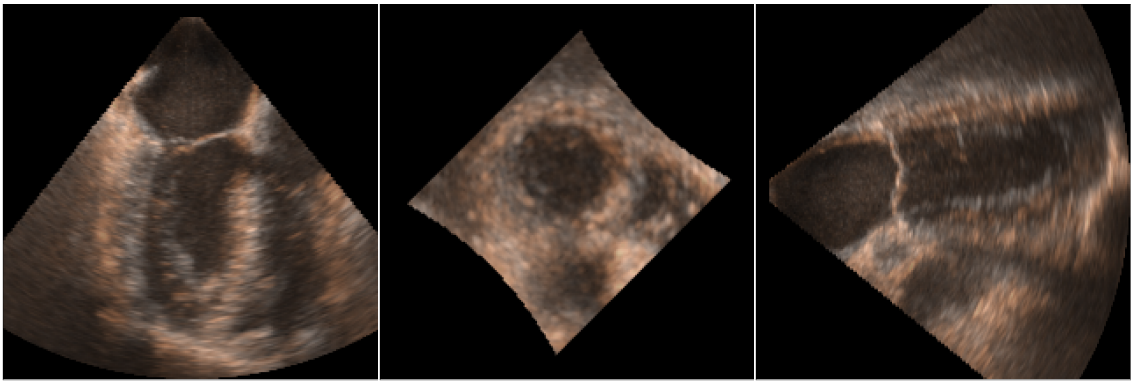
(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.2: Pre-alignment end-diastolic views of prediction set 3.



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.3: Pre-alignment end-systolic views of prediction set 2.



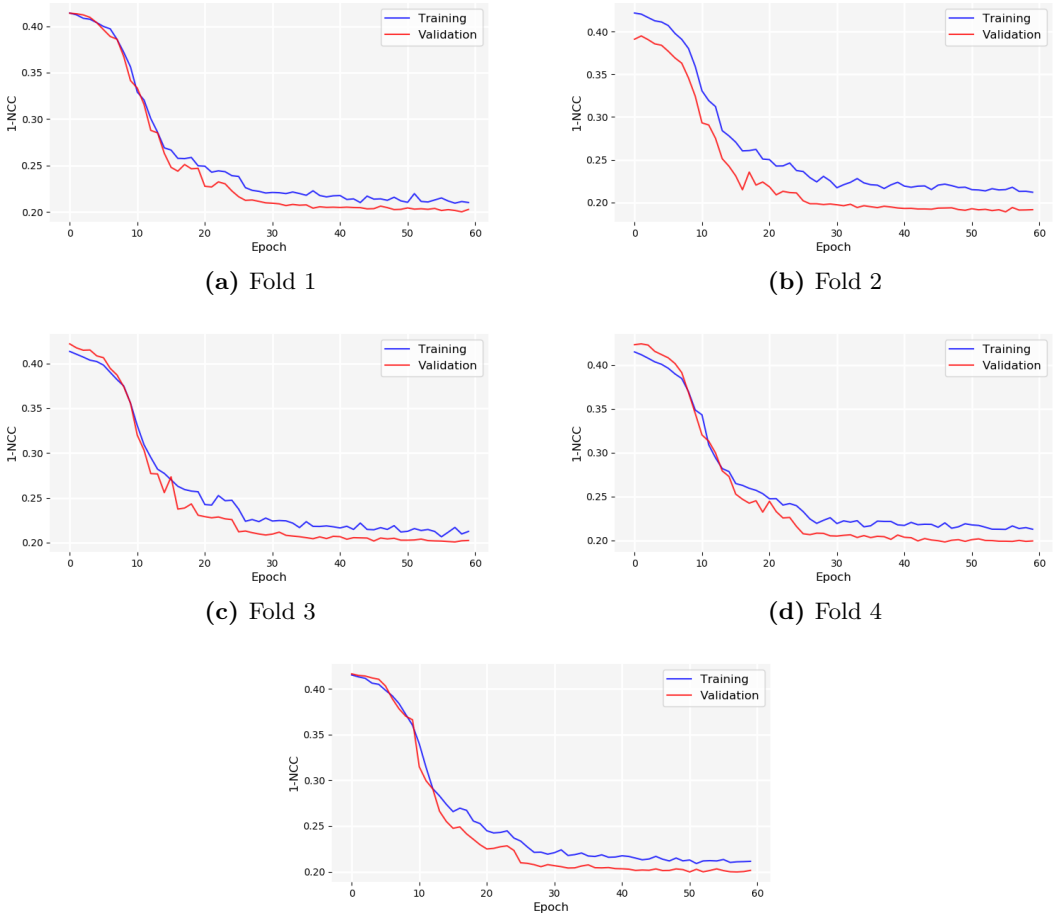
(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.4: Pre-alignment end-systolic views of prediction set 3.

4.1 Cross-validation

4.1.1 Model Training

During training, the similarity metric L_{sim} was monitored for the training and validation sets. Validation was done at the end of every epoch. Figure 4.5 shows the L_{sim} on the training and validation samples throughout training for each of the five folds in the cross-validation.



(a) Fold 1 **(b)** Fold 2

(c) Fold 3 **(d)** Fold 4

(e) Fold 5

Figure 4.5: Learning curves for each fold in the five-fold cross-validation showing the similarity metric L_{sim} for training and validation.

All five folds have a rapid decrease over the first 30 epochs before they start converging around 0.20, and all five folds end up performing better on the validation set than on the training set. Compared to the other folds, fold 2 has a larger gap between the performance on the training and validation sets. In the other four folds, validation follows training close up until epoch 15 before the start to deviate slightly.

The inference time for the network to produce the affine transformation was estimated when

running on the same GPU that was used for training. Inference time was found to be on average 447 ms for processing one batch of paired volumes, where the batch size is the same that was used for training.

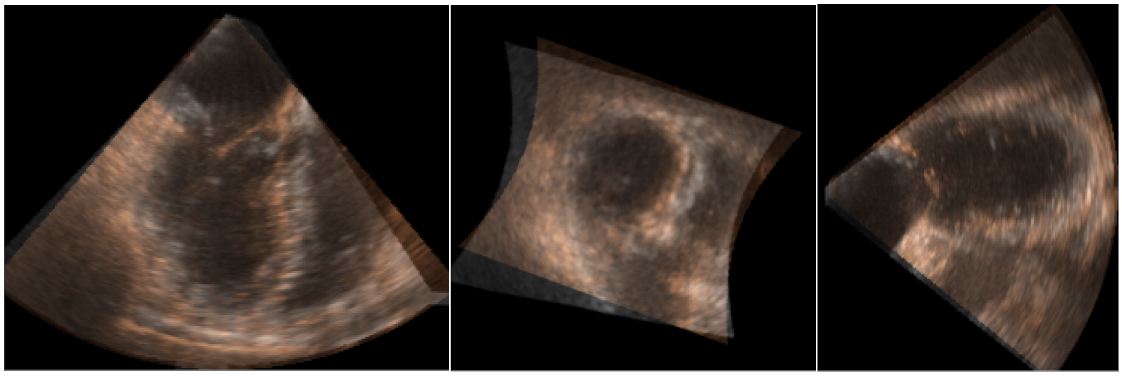
In total, the network requires 35.6M parameters. The encoder requires only 0.6M of these with the fully connected linear layers requiring the remaining parameters.

4.1.2 Similarity Metric and Visual Inspection

The results of the registration were obtained by computing NCC values for each prediction set using each of the five folds, and the results are summarized in Table 4.2. All of the post-alignment NCC values were computed using a masked NCC, where the similarity was only computed in overlapping regions to avoid any bias. Highlighted in fold 2 are the values that corresponds to the least and most improved registration in terms of increase in NCC post-alignment, and visualization of these predictions are shown in Figure 4.6 and 4.7 respectively.

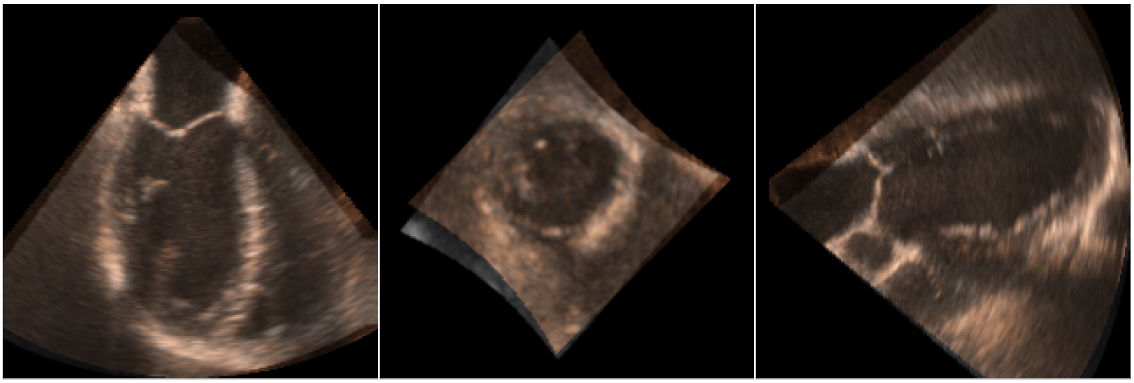
Table 4.2: Five-fold cross-validation NCC values for the three prediction sets.

	NCC						
Fold	1	2	3	4	5	**Total**	Improvement
Prediction set 1	0.8823	0.8800	0.8944	0.8821	0.8842	**0.8846**	0.0483
Prediction set 2	0.9264	**0.9177**	0.9285	0.9185	0.9316	**0.9245**	0.0461
Prediction set 3	0.9293	**0.9433**	0.9341	0.9317	0.9305	**0.9338**	0.0636



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.6: Post-alignment end-diastolic views of prediction set 2 for fold 2.



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.7: Post-alignment end-diastolic views of prediction set 3 for fold 2.

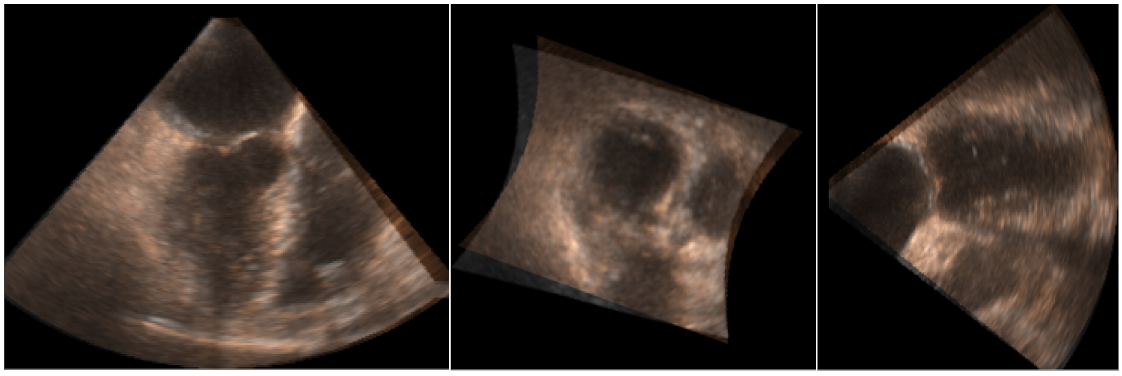
The visual inspection for prediction set 2 shows improved alignment of the septum and the right ventricle compared to the baseline. In prediction set 3, improvement and good alignment of all segments can be observed.

4.2 Cardiac Frame

Registration results were also obtained from the end-systolic frame, and the results are summarized in Table 4.3. Highlighted in fold 2 and 5 are respectively the highest NCC value and the most improved value in terms of increase from the baseline, and a visualization of these predictions are shown in Figure 4.8 and 4.9.

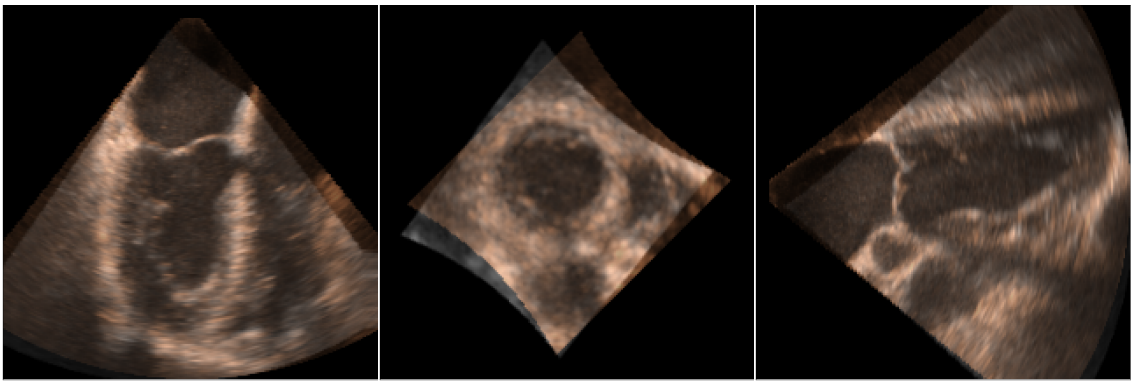
Table 4.3: End-systolic post-alignment NCC values for the three prediction sets.

	NCC						
Fold	1	2	3	4	5	**Total**	Improvement
Prediction set 1	0.9204	0.9121	0.9181	0.9034	0.9070	**0.9122**	0.0523
Prediction set 2	0.9402	**0.9449**	0.9404	0.9448	0.9391	**0.9419**	0.0391
Prediction set 3	0.9233	0.9205	0.9378	0.9292	**0.9390**	**0.9299**	0.0583



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.8: Post-alignment end-systolic views of prediction set 2.



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

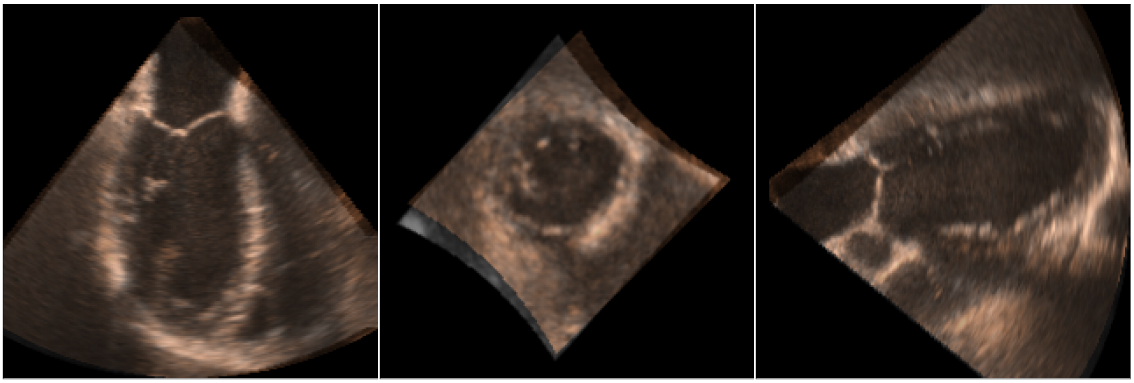
Figure 4.9: Post-alignment end-systolic views of prediction set 3.

4.3 Preprocessing

Registration results were obtained using NLMF as the filtering method, and Table 4.4 summarizes the values. The highlighted value in fold 2 corresponds to the most improved value in terms of increase from the baseline, and a visualization of the prediction is shown in Figure 4.10.

Table 4.4: NLMF post-alignment NCC values for the three prediction sets.

	NCC						
Fold	1	2	3	4	5	**Total**	Improvement
Prediction set 1	0.8836	0.8946	0.8785	0.8816	0.8787	**0.8834**	0.0471
Prediction set 2	0.9260	0.9260	0.9221	0.9210	0.9337	**0.9258**	0.0474
Prediction set 3	0.9381	**0.9459**	0.9297	0.9226	0.9408	**0.9354**	0.0652



(a) Long axis x-slice **(b)** Short axis **(c)** Long axis z-slice

Figure 4.10: Post-alignment NLMF views of prediction set 3.

4.4 GPU Mixed vs. Full Precision

The results from mixed precision training are summarized in Table 4.5. Mixed precision training requires 4.2 GB of GPU memory, which is a 45% reduction in required memory compared to full precision training. It reduces the training time per epoch by 2.6%. At inference, a 5.5% reduction in inference time and 12% reduction in memory consumption is observed.

Table 4.5: Mixed precision post-alignment NCC values for the three prediction sets.

	NCC						
Fold	1	2	3	4	5	**Total**	Improvement
Prediction set 1	0.8998	0.8985	0.8964	0.8775	0.8840	**0.8912**	0.0549
Prediction set 2	0.9234	0.9239	0.9209	0.9130	0.9210	**0.9204**	0.0420
Prediction set 3	0.9340	0.9325	0.9431	0.9318	0.9320	**0.9347**	0.0645

5 | Discussion

5.1 Cross-validation

The ultrasound volume registration task was performed by a neural network consisting of both convolutional layers and fully connected linear layers, which was trained in an unsupervised manner. Unsupervised training does not require manually annotated data, which is both time consuming and costly to obtain. The large number of parameters required to train the network is potentially problematic, as a higher number of parameters requires a larger dataset to effectively optimize all the parameters. It also highlights the limitations of using several fully connected linear layers in terms of image registration problems, as these layers require the majority of parameters. A solution to reduce the number of parameters would normally be to flatten the output of the convolutional layers and only use one linear layer to directly output the transformation parameters, as was done in [34]. However, testing showed that this was not the case for this particular problem and only made the predictions worse. In addition, the use of multiple linear layers effectively eliminates the possibility to train on larger volumes without aggressively downsampling the output of the convolutions.

During training of the network, no signs of overfitting were observed. For all five folds, the validation loss ends up lower than the training loss, indicating that the model is able to generalize well. In PyTorch, the batch normalization layers keep running estimates of its computed mean and variance during training which are then used for normalization during validation, and is most likely why validation ends up lower than the training loss. This behaviour could be disabled but would lead to worse performance on the validation set. For the cross-validation procedure, the patient data was split into separate folds after they were subdivided and shuffled. Ideally this split should happen on patient level to ensure that the model generalizes to an entirely unseen dataset. This approach was tested, but due to the large differences in a relatively small dataset, the models were not able to improve similarity from the baseline values.

The estimated inference time is quite high, about 40 times higher than what Zhao et al. achieves with affine registration on their liver CT registration [34], which also has the same input size to the network. However, the comparison is not fair as they only need to process one sample during inference as they do registration on the full volume. Inference time increases linearly with batch size in PyTorch, which makes inference with our model on one sample only three times slower than Zhao et al. Moreover, the implementation of the network is done in Python using the PyTorch library, and the Python interpreter is used at runtime. Implementing the method in a compiled language such as C++ will make the proposed method much faster. Another way of reducing inference time would be to parallelize the two encoder pipelines and run them on separate GPUs if hardware is available.

As for the visual inspection, it is clear that the achieved registration on prediction set 3 is significantly better than what is achieved in prediction set 2. For prediction set 2, a larger transformation was required to successfully align the volumes. The overall image quality is also lower than for prediction set 3, the myocardium is not as visible, and the valves and septum are less clear. However, the successful registration in Figure 4.7 show that the method can achieve good alignment in both long and short axis view and for all segments of the heart.

The network was found to consistently underestimate large movements and overestimate small movements. Underestimation of large movements could be handled by daisy-chaining several networks such as de Vos et al. did in their method [13], where several networks are stacked to enable a coarse-to-fine registration. With the high inference time in this method however, this becomes infeasible in terms of real-time usage as the network would become to slow. In addition, the constant overestimation of small movements would make convergence to the optimum difficult for the stacked network. It is assumed that the overestimation of small movements would be eliminated if a larger dataset, containing such small transformations between the images, was available.

Registration on patches that do not cover data-rich structures such as the myocardium, valvular areas or the septum was found to be difficult. If a patch ends up inside the ultrasound sector without covering these structures, it is still kept for training as it contains more than 70% non-zero data. With the lack of relation to important surrounding structures, spatial information is lost during downsampling in the encoder and the network is not able to make a good prediction. This would also be the case in volumes where these structures are only partially imaged, such as in prediction set 2. The lack of a clear myocardium could be a contributing factor to the unsuccessful alignment, but not the main cause as underestimation of patches containing such structures were still observed.

Due to the limited amount of training data, a patch-based method was proposed. Although this method enables generation of a sufficient amount of training data, it has some drawbacks. The quality of each patch is the same as the volume it was taken from, which keeps the overall data quality the same. The small stride causes a big overlap between patches, which could make subsequent patches to similar. This could potentially lead to the network not generalizing well to unseen data, however that is not the case as was seen in the plots from the model training. The big difference in volume size makes the bigger volumes contribute with much more samples towards the data set, which could lead the model to be biased towards these datasets. Upsampling of the volumes to the largest volume-size was attempted, but the overall performance of the network decreased.

The fact that some of the volumes showed different field of views was found to significantly impact the results of the registration. Each volume was therefore realigned to enable the same field of view. This was done subjectively, and similar anatomical orientation cannot be guaranteed. This should however be a feature in the DICOM format soon, such that the angle it was rotated with can be extracted and used to realign the volumes.

5.2 Cardiac Frame

In the ES frame, the ventricle volume is at its lowest in the heart cycle and the heart is fully contracted. This frame contains more heart structures compared to the ED frame, as can be seen in the pre-alignment visualizations of Chapter 4. This makes a direct comparison between registration results for ED and ES difficult. Table 4.3 shows that NCC values for prediction set 1 and 2 is significantly higher than those of ED registration. This is expected as the baseline NCC values are higher. In total, the overall improvement of the ES is slightly lower than ED.

Looking at the visual alignment for prediction set 3, it shows the same good alignment in all segments as for ED. Prediction set 2 is significantly better aligned in the septum, mitral valve and the ventricles. This could be due to the smaller transformation required in the ES frame, or the fact that the ES frame contains more structures that can be aligned. However, the results are inconclusive before testing has been done on more datasets.

It is worth noting that prediction set 2 has a higher post-alignment NCC value than prediction set 3, although the latter is clearly better aligned. This indicates that NCC cannot be used as the only evaluation and visual inspection also needs to be used actively.

5.3 Preprocessing

Table 4.4 shows that preprocessing using NLMF as the filter compares closely with the results in Table 4.2 where a bilateral filter was used. It was assumed that as the NLMF keeps a greater clarity and less loss of details in the filtered image, it would lead to improved registration results compared to simpler filtering algorithms. The overall improvement over the bilateral filter is however very low. With the additional significantly lower filtering speed, this discourages the use of the NLMF over the bilateral filter as the preferred filtering method. The visual inspection of the prediction with NLMF also show no significant improvement in registration results.

5.4 GPU Mixed vs. Full Precision

If the results in Table 4.5 are compared to those of the cross-validation in Table 4.2, a slight increase of improvement is observed for prediction set 1 and 3 and a slight decrease is observed for prediction set 2. The increase in NCC for prediction set 1 and 3 is unexpected but could be a result of the difficulty in achieving complete determinism in some of the functions that are used. However, the results show that the model accuracy is very close to that of full precision, even though memory consumption is reduced by 45% during training and 12% during inference. This makes mixed precision suitable for prediction on full volumes, which would make the overall registration method much faster.

A speed-up in training and inference is also observed. The use of 16-bit floating points (FP16) speeds up data transfers across layers due to lower memory bandwidth requirements. However,

the speed-up is only observed for large batch sizes, and a reduction in batch size will level the inference time between full- and mixed precision. By implementing the method on hardware that is more compatible with FP16, speed-ups are expected also in small batch sizes.

5.5 Limitations of Study and Future Work

Predictions in this thesis are made on preprocessed volumes and not on the raw TEE ultrasound volumes. Making predictions on preprocessed images is not a limitation itself but is nevertheless a shortcoming of the proposed pipeline. The implementations of the preprocessing algorithms presented are time-consuming and prevent real-time use. It is necessary to either implement the algorithms faster or develop a more robust registration network that handles registration on minimally preprocessed volumes.

Due to the limited amount of available data, only 3 prediction sets were set aside for testing to keep a sufficient amount for training and validation. This makes it hard to quantify the results, and a comparison between ED and ES prediction is particularly difficult. If more datasets are acquired, testing should be extended to fully evaluate the possibilities in the method.

Reference values should be acquired and compared using commercially available, iterative methods, for a more accurate assessment of the predictions. Ground truth manual alignments could also be obtained for comparison of the results.

The long inference time of the network is problematic, as use during interventions would require registration in real-time, and subdividing volumes and global alignment brings additional overhead. Efforts should be made to parallelize the pipelines of the network to improve performance. A more efficient algorithm to subdivide volumes should be developed, and a GPU implementation of the Procrustes method would significantly speed up predictions, as no time would get lost transferring from GPU to CPU and back.

To improve the registration, more experiments should be done with regard to network architecture. Experiments could be done with different downsampling rates, less fully connected linear layers or combination of multiple simpler networks. Combination of networks in particular have been shown to perform well in image registration methods [13, 34]. Each hyperparameter of the model should also be carefully tuned, including learning rate and batch size.

Much effort should be made to gather more high-quality TEE ultrasound volumes, both to improve the results but also to more accurately quantify the results. In addition, the potential that lies in memory efficient implementation of the network and mixed precision training, could be utilized to make predictions on full ultrasound volumes if more data is available.

6 | Conclusion

In this thesis, a deep learning based approach to ultrasound volume registration of 3D transesophageal echocardiographic recordings was presented. The method is an unsupervised end-to-end automatic pipeline consisting of an affine registration network and a global transformation method, and is, to the best of our knowledge, the first attempt to do this on ultrasound volumes. To overcome the complexity of deep learning on 3D data, and due to a general lack of datasets, the registration network is patch-based, and Procrustes analysis was used to globally align the point clouds. The network was built up using an adaptation of the dilated residual dense block introduced by Lee et al. [24].

Five-fold cross-validation showed that the model was able to generalize well to previously unseen data. The most successful registration was found on samples of good quality and with clearly visible structures, while performance decreased in samples that required large transformations to successfully align. Registration on the end-systolic frame showed promising results compared to the end-diastolic frame, but results were inconclusive due to the limited amount of test sets. Results from comparison of preprocessing algorithms showed that the advanced non-local means filtering algorithm only achieved slightly better results compared to the simpler bilateral filtering algorithm. Mixed precision achieved almost the same results as full precision, at a 45% reduction in memory consumption. With the low memory requirement of mixed precision, training and prediction on larger volumes is feasible using this method. Some sources of errors were identified with suggested improvements. The current inference time rules out real-time applications, but simple measures can be made to significantly reduce it.

In conclusion, the results showed that the fully automated ultrasound registration pipeline that was proposed, can successfully register successive echocardiographic recordings. This indicates that ultrasound volume registration is a learnable task that can be solved using deep learning. Further efforts should be focused towards gathering of more ultrasound data, reduce inference time and improve robustness of the method.

Bibliography

[1] R. B. Hawkins *et al.*, "Minimally invasive mitral valve surgery is associated with excellent resource utilization, cost, and outcomes", *The Journal of Thoracic and Cardiovascular Surgery*, vol. 156, no. 2, 611–616.e3, Aug. 2018, ISSN: 00225223. DOI: 10.1016/j.jtcvs.2018.03.108.

[2] E. A. Downs *et al.*, "Minimally invasive mitral valve surgery provides excellent outcomes without increased cost: A multi-institutional analysis", *The Annals of Thoracic Surgery*, vol. 102, no. 1, pp. 14–21, Jul. 2016, ISSN: 00034975. DOI: 10.1016/j.athoracsur.2016.01.084.

[3] F. Lucà *et al.*, "Minimally invasive mitral valve surgery: A systematic review", *Minimally Invasive Surgery*, vol. 2013, pp. 1–10, 2013, ISSN: 2090-1445, 2090-1453. DOI: 10.1155/2013/179569.

[4] P. Modi *et al.*, "Minimally invasive mitral valve surgery: A systematic review and meta-analysis", *European Journal of Cardio-Thoracic Surgery*, vol. 34, no. 5, pp. 943–952, Nov. 2008, ISSN: 10107940. DOI: 10.1016/j.ejcts.2008.07.057.

[5] C. D. Flynn and T. D. Yan, "Minimally invasive aortic surgery", in *New Approaches to Aortic Diseases from Valve to Abdominal Bifurcation*, Elsevier, 2018, pp. 383–392, ISBN: 978-0-12-809979-7. DOI: 10.1016/B978-0-12-809979-7.00033-X.

[6] F. Gumus *et al.*, "Multiple valve implantation through a minimally invasive approach: Comparison of standard median sternotomy and right anterior thoracotomy", *Heart, Lung and Circulation*, S144395062030041X, Feb. 2020, ISSN: 14439506. DOI: 10.1016/j.hlc.2020.01.012.

[7] G. Perk *et al.*, "Use of real time three-dimensional transesophageal echocardiography in intracardiac catheter based interventions", *Journal of the American Society of Echocardiography*, vol. 22, no. 8, pp. 865–882, Aug. 2009, ISSN: 08947317. DOI: 10.1016/j.echo.2009.04.031.

[8] J. K. Dave *et al.*, "Recent technological advancements in cardiac ultrasound imaging", *Ultrasonics*, vol. 84, pp. 329–340, Mar. 2018, ISSN: 0041624X. DOI: 10.1016/j.ultras.2017.11.013.

[9] H. Patel *et al.*, "Echocardiography in transcatheter structural heart disease interventions", *Progress in Cardiovascular Diseases*, vol. 61, no. 5, pp. 423–436, Nov. 2018, ISSN: 00330620. DOI: 10.1016/j.pcad.2018.11.009.

[10] A. Danudibroto *et al.*, "Anatomical view stabilization of multiple 3d transesophageal echocardiograms", in *2016 IEEE International Ultrasonics Symposium (IUS)*, Tours, France: IEEE, Sep. 2016, pp. 1–4, ISBN: 978-1-4673-9897-8. DOI: 10.1109/ULTSYM.2016.7728596.

[11] H. T. van den Broek *et al.*, "3d hybrid imaging for structural and congenital heart interventions in the cath lab", *Structural Heart*, vol. 2, no. 5, pp. 362–371, Sep. 3, 2018, ISSN: 2474-8706, 2474-8714. DOI: 10.1080/24748706.2018.1490841.

[12] B. M. Wiley *et al.*, "Fusion imaging for procedural guidance", *Revista Española de Cardiología (English Edition)*, vol. 71, no. 5, pp. 373–381, May 2018, ISSN: 18855857. DOI: 10.1016/j.rec.2017.10.029.

[13] B. D. de Vos *et al.*, "A deep learning framework for unsupervised affine and deformable image registration", *arXiv:1809.06130 [cs]*, Dec. 5, 2018. arXiv: 1809.06130.

[14] B. Rigaud *et al.*, "Deformable image registration for radiation therapy: Principle, methods, applications and evaluation", *Acta Oncologica*, vol. 58, no. 9, pp. 1225–1237, Sep. 2, 2019, ISSN: 0284-186X, 1651-226X. DOI: 10.1080/0284186X.2019.1620331.

[15] A. Sotiras *et al.*, "Deformable medical image registration: A survey", *IEEE Transactions on Medical Imaging*, vol. 32, no. 7, pp. 1153–1190, Jul. 2013, ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2013.2265603.

[16] A. Khalil *et al.*, "An overview on image registration techniques for cardiac diagnosis and treatment", *Cardiology Research and Practice*, vol. 2018, pp. 1–15, Aug. 8, 2018, ISSN: 2090-8016, 2090-0597. DOI: 10.1155/2018/1437125.

[17] B. C. Lowekamp *et al.*, "The design of SimpleITK", *Frontiers in Neuroinformatics*, vol. 7, 2013, ISSN: 1662-5196. DOI: 10.3389/fninf.2013.00045.

[18] A. Danudibroto *et al.*, "Spatiotemporal registration of multiple three-dimensional echocardiographic recordings for enhanced field of view imaging", *Journal of Medical Imaging*, vol. 3, no. 3, p. 037 001, Jul. 8, 2016, ISSN: 2329-4302. DOI: 10.1117/1.JMI.3.3.037001.

[19] G. Farnebäck and C.-F. Westin, "Affine and deformable registration based on polynomial expansion", in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*, R. Larsen *et al.*, Eds., vol. 4190, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 857–864, ISBN: 978-3-540-44707-8. DOI: 10.1007/11866565_105.

[20] D. Forsberg *et al.*, "Multi-modal image registration using polynomial expansion and mutual information", in *Biomedical Image Registration*, vol. 7359, Series Title: Lecture Notes in Computer Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 40–49, ISBN: 978-3-642-31339-4. DOI: 10.1007/978-3-642-31340-0_5.

[21] B. Heyde *et al.*, "Anatomical image registration using volume conservation to assess cardiac deformation from 3d ultrasound recordings", *IEEE Transactions on Medical Imaging*, vol. 35, no. 2, pp. 501–511, Feb. 2016, ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2015.2479556.

[22] A. H. Pham *et al.*, "Fast ultrasound to ultrasound auto-registration for interventional cardiology", in *2019 IEEE International Ultrasonics Symposium (IUS)*, Glasgow, United Kingdom: IEEE, Oct. 2019, pp. 16–19, ISBN: 978-1-72814-596-9. DOI: 10.1109/ULTSYM.2019.8925750.

[23] Z. Sobhaninia *et al.*, "Brain tumor segmentation using deep learning by type specific sorting of images", *arXiv:1809.07786 [cs, eess]*, Sep. 20, 2018. arXiv: 1809.07786.

[24] H. Lee *et al.*, "Efficient 3d fully convolutional networks for pulmonary lobe segmentation in CT images", *arXiv:1909.07474 [cs, eess]*, Sep. 16, 2019. arXiv: 1909.07474.

[25] A. Sheikhjafari *et al.*, "Unsupervised deformable image registration with fully connected generative neural network", 2018.

[26] G. Balakrishnan *et al.*, "An unsupervised learning model for deformable medical image registration", *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9252–9260, Jun. 2018. DOI: 10.1109/CVPR.2018.00964. arXiv: 1802.02604.

[27] M. W. Lafarge *et al.*, "Deformable image registration using convolutional neural networks", in *Medical Imaging 2018: Image Processing*, E. D. Angelini and B. A. Landman, Eds., Houston, United States: SPIE, Mar. 2, 2018, p. 27, ISBN: 978-1-5106-1637-0. DOI: 10.1117/12.2292443.

[28] S. Shan *et al.*, "Unsupervised end-to-end learning for deformable medical image registration", *arXiv:1711.08608 [cs]*, Jan. 19, 2018. arXiv: 1711.08608.

[29] L. Sun and S. Zhang, "Deformable MRI-ultrasound registration using 3d convolutional neural network", in *Simulation, Image Processing, and Ultrasound Systems for Assisted Diagnosis and Navigation*, vol. 11042, Cham: Springer International Publishing, 2018, pp. 152–158, ISBN: 978-3-030-01044-7. DOI: 10.1007/978-3-030-01045-4_18.

[30] Y. Hu *et al.*, "Weakly-supervised convolutional neural networks for multimodal image registration", *Medical Image Analysis*, vol. 49, pp. 1–13, Oct. 2018, ISSN: 13618415. DOI: 10.1016/j.media.2018.07.002. arXiv: 1807.03361.

[31] A. Kori and G. Krishnamurthi, "Zero shot learning for multi-modal real time image registration", *arXiv:1908.06213 [cs]*, Aug. 16, 2019. arXiv: 1908.06213.

[32] M. Jaderberg *et al.*, "Spatial transformer networks", *arXiv:1506.02025 [cs]*, Feb. 4, 2016. arXiv: 1506.02025.

[33] E. Chee and Z. Wu, "AIRNet: Self-supervised affine registration for 3d medical images using neural networks", *arXiv:1810.02583 [cs]*, Oct. 14, 2018. arXiv: 1810.02583.

[34] S. Zhao *et al.*, "Unsupervised 3d end-to-end medical image registration with volume tweening network", *IEEE Journal of Biomedical and Health Informatics*, pp. 1–1, 2019, ISSN: 2168-2194, 2168-2208. DOI: 10.1109/JBHI.2019.2951024.

[35] H. Sokooti *et al.*, "Nonrigid image registration using multi-scale 3d convolutional neural networks", in *Medical Image Computing and Computer Assisted Intervention - MICCAI 2017*, vol. 10433, Cham: Springer International Publishing, 2017, pp. 232–239, ISBN: 978-3-319-66181-0. DOI: 10.1007/978-3-319-66182-7_27.

[36] X. Cao *et al.*, "Deep learning based inter-modality image registration supervised by intra-modality similarity", *arXiv:1804.10735 [cs]*, Apr. 27, 2018. arXiv: 1804.10735.

[37] Z. Shen *et al.*, "Networks for joint affine and non-parametric image registration", *arXiv:1903.08811 [cs]*, Mar. 20, 2019. arXiv: 1903.08811.

[38] Z. Zhang *et al.*, "Road extraction by deep residual u-net", *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 749–753, May 2018, ISSN: 1545-598X, 1558-0571. DOI: 10.1109/LGRS.2018.2802944. arXiv: 1711.10684.

[39] OpenStax College, *Anatomy and physiology*. Houston, Texas: Rice University, 2013, OCLC: 911561117, ISBN: 978-1-938168-13-0.

[40] R. L. Drake *et al.*, *Gray's anatomy for students*, 4th edition. Philadelphia, MO: Elsevier, 2019, ISBN: 978-0-323-39304-1.

[41] L. S. Athanasiou *et al.*, "Introduction", in *Atherosclerotic Plaque Characterization Methods Based on Coronary Imaging*, Elsevier, 2017, pp. 1–21, ISBN: 978-0-12-804734-7. DOI: 10.1016/B978-0-12-804734-7.00001-4.

[42] R. G. Carroll, "The heart", in *Elsevier's Integrated Physiology*, Elsevier, 2007, pp. 65–75, ISBN: 978-0-323-04318-2. DOI: 10.1016/B978-0-323-04318-2.50013-3.

[43] K. Najarian, *Biomedical signal and image processing*. 2016, OCLC: 991528045, ISBN: 978-1-4398-7034-1.

[44] P. R. Hoskins *et al.*, Eds., *Diagnostic ultrasound: physics and equipment*, 2nd ed, Cambridge medicine, OCLC: ocn573196525, Cambridge, UK ; New York: Cambridge University Press, 2010, 263 pp., ISBN: 978-0-521-75710-2.

[45] K. K. Shung, *Diagnostic ultrasound: imaging and blood flow measurements*, Second edition. Boca Raton: CRC Press, Taylor & Francis Group, 2015, 273 pp., ISBN: 978-1-4665-8264-4.

[46] R. T. Hahn *et al.*, "Guidelines for performing a comprehensive transesophageal echocardiographic examination: Recommendations from the american society of echocardiography and the society of cardiovascular anesthesiologists", *Journal of the American Society of Echocardiography*, vol. 26, no. 9, pp. 921–964, Sep. 2013, ISSN: 08947317. DOI: 10.1016/j.echo.2013.07.009.

[47] A. B. Freitas-Ferraz *et al.*, "Transesophageal echocardiography complications associated with interventional cardiology procedures", *American Heart Journal*, vol. 221, pp. 19–28, Mar. 2020, ISSN: 00028703. DOI: 10.1016/j.ahj.2019.11.018.

[48] W. Wein *et al.*, "Automatic CT-ultrasound registration for diagnostic imaging and image-guided intervention", *Medical Image Analysis*, vol. 12, no. 5, pp. 577–585, Oct. 2008, ISSN: 13618415. DOI: 10.1016/j.media.2008.06.006.

[49] A. Geron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*, 1st ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Inc, 2017, ISBN: 978-1-4919-6229-9.

[50] Y. LeCun *et al.*, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature14539.

[51] R. C. Gonzalez and R. E. Woods, *Digital Image Processing: Global Edition*, 4th Edition. Pearson Education Inc, 2018, ISBN: 978-93-5306-298-9.

[52] I. Goodfellow *et al.*, *Deep Learning*. MIT Press, 2016.

[53] Y. Lecun *et al.*, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, ISSN: 00189219. DOI: 10.1109/5.726791.

[54] A. Krizhevsky *et al.*, "ImageNet classification with deep convolutional neural networks", in *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., 2012, pp. 1097–1105.

[55] A. Boulch, "ShaResNet: Reducing residual network parameter number by sharing weights", *arXiv:1702.08782 [cs]*, Mar. 6, 2017. arXiv: 1702.08782.

[56] D. Strigl *et al.*, "Performance and scalability of GPU-based convolutional neural networks", in *2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing*, Pisa: IEEE, Feb. 2010, pp. 317–324, ISBN: 978-1-4244-5672-7. DOI: 10.1109/PDP.2010.43.

[57] K. He *et al.*, "Deep residual learning for image recognition", *arXiv:1512.03385 [cs]*, Dec. 10, 2015. arXiv: 1512.03385.

[58] G. Huang *et al.*, "Densely connected convolutional networks", *arXiv:1608.06993 [cs]*, Jan. 28, 2018. arXiv: 1608.06993.

[59] D. Eggert *et al.*, "Estimating 3-d rigid body transformations: A comparison of four major algorithms", *Machine Vision and Applications*, vol. 9, no. 5, pp. 272–290, Mar. 1, 1997, ISSN: 0932-8092, 1432-1769. DOI: 10.1007/s001380050048.

[60] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images", in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, Bombay, India: Narosa Publishing House, 1998, pp. 839–846, ISBN: 978-81-7319-221-0. DOI: 10.1109/ICCV.1998.710815.

[61] A. Buades *et al.*, "A non-local algorithm for image denoising", in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, San Diego, CA, USA: IEEE, 2005, pp. 60–65, ISBN: 978-0-7695-2372-9. DOI: 10.1109/CVPR.2005.38.

[62] X. Yang *et al.*, "Quicksilver: Fast predictive image registration - a deep learning approach", *arXiv:1703.10908 [cs]*, Jul. 19, 2017. arXiv: 1703.10908.

[63] K. He *et al.*, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification", *arXiv:1502.01852 [cs]*, Feb. 6, 2015. arXiv: 1502.01852.

[64] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *arXiv:1412.6980 [cs]*, Jan. 29, 2017. arXiv: 1412.6980.

[65] F. Banterle *et al.*, "A low-memory, straightforward and fast bilateral filter through subsampling in spatial domain", *Computer Graphics Forum*, vol. 31, no. 1, pp. 19–32, Feb. 2012, ISSN: 01677055. DOI: 10.1111/j.1467-8659.2011.02078.x.

A | Bilateral Filtering

In a two-dimensional image, the bilateral filter is defined as [60, 65]

$$I^{\text{filtered}}(x) = \frac{1}{W_p} \sum_{x_i \in \Omega} I(x_i) f_r(||I(x_i) - I(x)||) g_s(||x_i - x||) \tag{A.1}$$

with the normalization term W_p defined as

$$W_p = \sum_{x_i \in \Omega} f_r(||I(x_i) - I(x)||) g_s(||x_i - x||) \tag{A.2}$$

where

I^{filtered} is the filtered image;

I is the input image that is to be filtered;

x are the coordinates of the current pixel to be filtered;

Ω is the window centered in x, so $x_i \in \Omega$ is another pixel;

f_r and g_r is the range and spatial kernel respectively for smoothing differences in coordinates.

Given a pixel located at (i, j) that needs to be denoised by a neighboring pixel at (k, l), assuming f_r and g_r to be Gaussian kernels, the weight assigned for pixel (k, l) to denoise (i, j) is

$$w(i, j, k, l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2} - \frac{||I(i, j) - I(k, l)||^2}{2\sigma_r^2}\right) \tag{A.3}$$

where σ_d and σ_r are smoothing parameters, and $I(i, j)$ and $I(k, l)$ are the intensity values of pixels (i, j) and (k, l). The weights are normalized after calculation as

$$I_D(i, j) = \frac{\sum_{k,l} I(k, l) w(i, j, k, l)}{\sum_{k,l} w(i, j, k, l)} \tag{A.4}$$

where I_D is the denoised intensity of pixel (i, j).

As the range parameter σ_r increases, the bilateral filter gradually approaches Gaussian blur. The larger the spatial parameter σ_d gets, the larger features get smoothened.

B | NLM Filtering

Given a discrete noisy two-dimensional image $v = \{v(i) \mid i \in I\}$, the estimated non-local (NL) value $NL[v](i)$ for a pixel (i), is computed as a weighted average of all pixels in the image and is given by [61]

$$NL[v](i) = \sum_{j \in I} w(i,j)v(j), \tag{B.1}$$

where the weights $\{w(i,j)\}_j$ depend on the similarity between pixels i and j and satisfy $0 \leq w(i,j) \leq 1$ and $\sum_j w(i,j) = 1$.

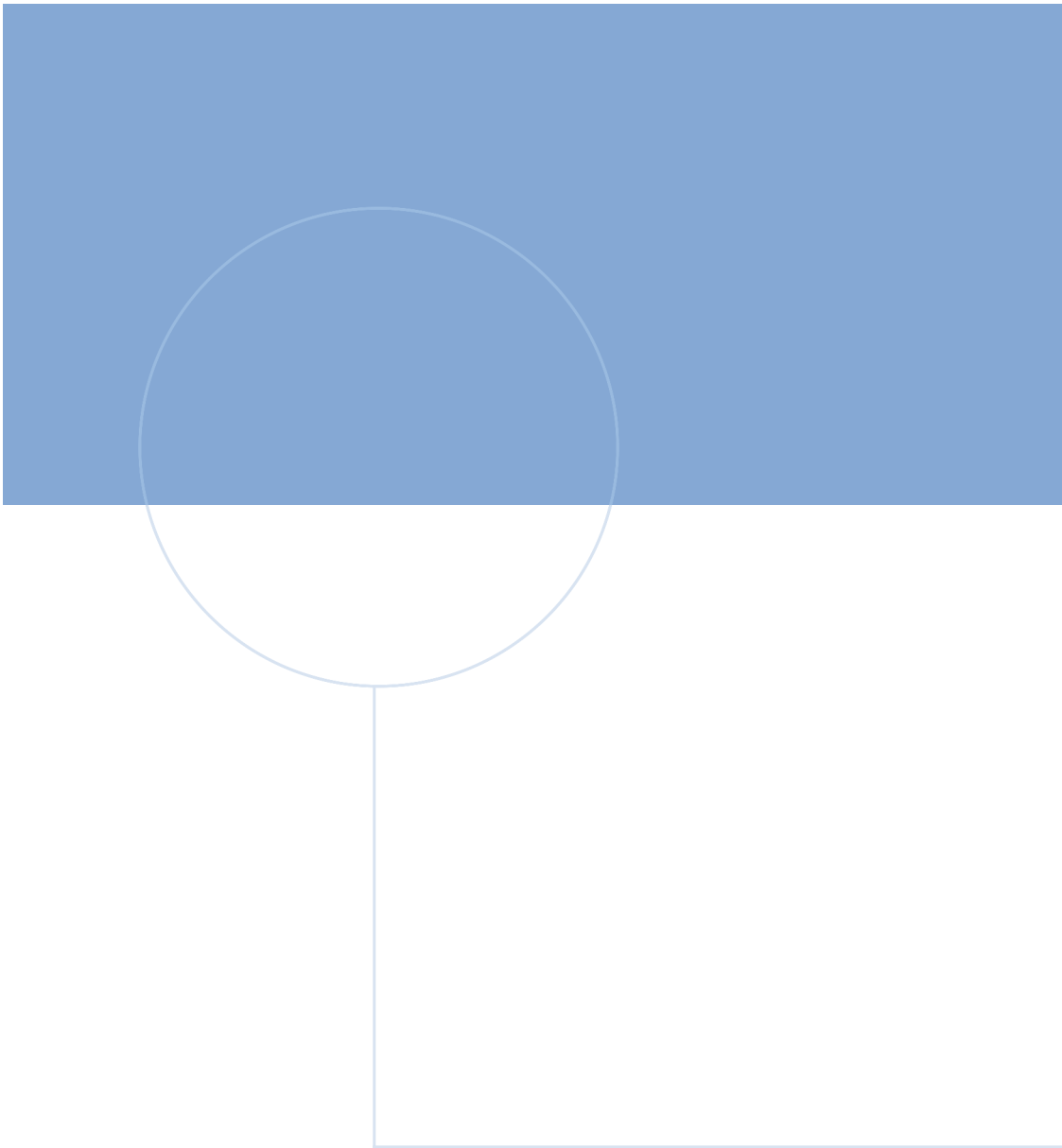
The similarity between the two pixels i and j depends on the similarity of the intensity gray level vectors $v(\mathcal{N}_i)$ and $v(\mathcal{N}_j)$, where $\mathcal{N}_k$ denotes a square neighborhood of fixed size and centered at pixel k. This similarity is measured as a decreasing function of the weighted Euclidean distance $||v(\mathcal{N}_i) - v(\mathcal{N}_j)||^2_{2,a}$, where $a > 0$ is the standard deviation of the Gaussian kernel. Pixels with a similar grey level neighborhood to $v(\mathcal{N}_i)$ have larger weights on average. The weights are defined as

$$w(i,j) = \frac{1}{Z(i)} \exp - \frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||^2_{2,a}}{h^2}, \tag{B.2}$$

where $Z(i)$ is the normalization constant defined as

$$Z(i) = \sum_j \exp - \frac{||v(\mathcal{N}_i) - v(\mathcal{N}_i)||^2_{2,a}}{h^2} \tag{B.3}$$

and the parameter h acts as a degree of filtering. It controls the decay of the exponential function and therefore the decay of the weights as a function of the Euclidean distances.



NTNU

Norwegian University of
Science and Technology