UNIVERSITÄT ZU LÜBECK
INSTITUTE FOR ROBOTICS
AND COGNITIVE SYSTEMS

ADVANCING ULTRASOUND IMAGE GUIDANCE: REAL-TIME TARGET TRACKING IN 4D ULTRASOUND USING DEEP REPRESENTATION LEARNING

Dissertation

Daniel Wulff

University of Lübeck
Institute for Robotics and Cognitive Systems

UNIVERSITÄT ZU LÜBECK

From the Institute for Robotics and Cognitive Systems
of the University of Lübeck
Director: Prof. Dr.-Ing. Achim Schweikard

Advancing ultrasound image guidance: Real-time target tracking in 4D ultrasound using deep representation learning

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

from the Department of Computer Sciences and Engineering

Submitted by

Daniel Wulff

from Lübeck

Lübeck, 2024

First Referee: Prof. Dr. rer. nat. habil. Floris Ernst

Second Referee: Prof. Dr. rer. nat. Franziska Mathis-Ullrich

Date of oral examination: November 7, 2024

Approved for printing. Lübeck, November 14, 2024

„Es scheint immer unmöglich, bis es vollbracht ist."
- Nelson Mandela

Für Sonja und Malea

Meine wundervolle Familie

KURZFASSUNG

Das Ziel der Strahlentherapie ist es Tumorgewebe zu zerstören, indem ionisierende Strahlung appliziert wird. Durch Patientenbewegungen, z.B. bedingt durch die Atmung, wird die Bestrahlung des Tumors erschwert, sodass auch gesundes Gewebe geschädigt wird. Dies wird bei der Planung der Bestrahlung durch Sicherheitsränder berücksichtigt. Durch eine kontinuierliche und genaue Lagekontrolle des Tumors können diese Sicherheitsränder und somit das Risiko gesundes Gewebe zu schädigen, minimiert werden. Dafür werden heute Röntgen-basierte Bildgebungsverfahren und implantierte Marker eingesetzt.
Die 3D Ultraschallbildgebung hat verschiedene Vorteile gegenüber anderen medizinischen Bildgebungsmodalitäten wie der Röntgenbildgebung: Sie ist in der Lage, Weichteilstrukturen in volumetrischen Bildern in Echtzeit zu visualisieren, ohne den Patienten durch ionisierende Strahlung zu schädigen. Diese Aspekte machen die 3D Ultraschallbildgebung zu einer vielversprechenden Bildgebungsmodalität für Anwendungen zur Therapieführung, z. B. in der Strahlentherapie. Nachteile wie eine handgehaltene Ultraschallsonde, ein begrenztes Sichtfeld und eine geringe Bildqualität stellen jedoch Herausforderungen dar, die es zu bewältigen gilt, um ein Ultraschallführungssystem zu realisieren. Es werden derzeit robotische Ultraschallsysteme erforscht, die in der Lage sein könnten, die handgehaltene Ultraschallsonde und das begrenzte Sichtfeld zu kompensieren, indem sie die Ultraschallsonde automatisch so ausrichten, dass das Ziel kontinuierlich visualisiert wird. Hierfür ist eine robuste und echtzeitfähige Zielverfolgung erforderlich, um kontinuierlich die genaue Position des Ziels zu ermitteln. In der Strahlentherapie kann dies sogar dazu verwendet werden, die Bestrahlung kontinuierlich anzupassen, um sicherzustellen, dass die Therapie wie geplant durchgeführt wird.
Aufgrund der geringen Bildqualität, der volumetrischen Bildeigenschaften und der hochdimensionalen Bewegungen des Weichgewebes ist die 4D Ultraschallverfolgung eine anspruchsvolle Aufgabe. Bisher wurden nur wenige Methoden untersucht, was darauf hinweist, dass es im Bereich des 4D Ultraschall Trackings eine Forschungslücke gibt. Die vorliegende Arbeit trägt dazu bei, diese Lücke zu schließen, indem sie die Anwendbarkeit des Repräsentationslernens mit tiefen neuronalen Netzen für die 4D-Ultraschallverfolgung untersucht. Das Training von tiefen neuronalen Netzen erfordert eine beträchtliche Datenmenge, aber bis heute ist nur eine begrenzte Menge von 4D Ultraschalldaten öffentlich zugänglich. Diese begrenzte Verfügbarkeit wurde in dieser Arbeit durch eine 4D Ultraschall Annotationsstudie erweitert. Es wurde ein neuer 4D Ultraschall Datensatz zur Verfügung gestellt, der Bilddaten und Landmarken enthält.
Es wurde gezeigt, dass lokale Bildmerkmale in 3D Ultraschallbildern mit Hilfe bi-

närer Merkmalsdeskriptoren auf eindeutige und aussagekräftige Weise erkannt und beschrieben werden können. Darüber hinaus wurde festgestellt, dass Autoenkoder in der Lage sind, 3D Ultraschallausschnitte in latente Repräsentationen zu überführen, die zur Identifizierung ähnlicher Weichteilstrukturen und zur Unterscheidung unähnlicher Strukturen verwendet werden können. Zu diesem Zweck wurden verschiedene Arten von Autoenkodern entwickelt und untersucht.
Um das Training von tiefen neuronalen Netzen zu unterstützen, wurden Methoden zur realistischen Veränderung von Ultraschallbildern entwickelt, die auf Repräsentationslernen basieren. Es wird eine neuartige Methode zur deformierbaren Datenerweiterung vorgestellt und gezeigt, dass sie die Genauigkeit eines neuronalen Netzes verbessert, das für die Ziellokalisierung in Ultraschallbildern trainiert wurde.
Es wurden verschiedene Algorithmen zur Zielverfolgung entwickelt. Algorithmen wurden implementiert und ausgewertet, die sowohl im Ultraschallbildraum als auch im Repräsentationsraum, der von Autoenkodern erzeugt wird, arbeiten. Es wurde gezeigt, dass die 4D Ultraschallverfolgung im Repräsentationsraum die bildraumbasierte Verfolgung in Bezug auf die Laufzeit übertreffen kann, während eine vergleichbare Genauigkeit beibehalten wird. Die in dieser Arbeit vorgeschlagene Zielverfolgungsmethodik basiert auf unüberwachtem Lernen, ist echtzeitfähig, robust und kann über Patienten und Organe hinweg verallgemeinert werden, was sie für ultraschallgeführte Therapiezwecke vielversprechend macht. Die Anwendbarkeit der auf dem Repräsentationsraum basierenden Verfolgung wurde in einem Online roboterbasiertem Ultraschallverfolgungsexperiment nachgewiesen. Daher wird in dieser Arbeit eine neuartige Methode zur 4D Ultraschallverfolgung vorgestellt, die in jeden Therapiebereich integriert werden könnte.

ABSTRACT

The aim of radiation therapy is to destroy tumor tissue by applying ionizing radiation. Patient movements, e.g. due to breathing, make irradiation of the tumor challenging, so that healthy tissue is also damaged. This is taken into account when planning the irradiation using safety margins. These safety margins and thus the risk of damaging healthy tissue can be minimized by continuously and precisely monitoring the position of the tumor. Today, X-ray-based imaging techniques and implanted markers are used for this purpose.
3D ultrasound imaging has different advantages over other medical imaging modalities such as X-ray imaging: It is capable of visualizing soft tissue structures in volumetric images in real-time without harming the patient with ionizing radiation. These aspects make 3D ultrasound imaging to a promising imaging modality for therapy guidance applications such as guiding radiation therapy. However, drawbacks like a hand-held ultrasound transducer, limited field of view and low image quality bring challenges that need to be addressed to be able to realize an ultrasound guidance system. Robotic ultrasound systems are being explored that may be able to compensate for the hand-held characteristic and the limited field of view by automatically aligning the ultrasound transducer so that the target is visualized continuously. Therefor, robust and real-time capable target tracking is required to be able to get continuous knowledge about the exact target position. In applications such as radiation therapy, it can even be used to continually optimize the treatment to ensure that the therapy is delivered as planned.
Due to low image quality, volumetric image characteristics and high-dimensional soft-tissue motion, 4D ultrasound tracking is a challenging task. Only a few methods have been investigated so far, showing that there is a lack in 4D ultrasound tracking research. This work contributes to fill this gap by investigating the usability of representation learning using deep neural networks for the purpose of 4D ultrasound tracking. Training deep neural networks requires a substantial amount of data, but to date, only a limited amount of 4D ultrasound data is publicly accessible. This limited amount of data has been extended in this work in a 4D ultrasound labeling study. A novel 4D ultrasound data set has been made available containing image and landmark data.
It has been shown that local image features can be detected and described in a unique and meaningful way in 3D ultrasound images using binary feature descriptors. In addition, it has been investigated that autoencoders are able to map 3D ultrasound patches into latent representations that can be used to identify similar soft-tissue structures and differentiate dissimilar ones. Therefor, different types of autoencoders were developed and investigated.

In order to support training deep neural networks, methods for realistic ultrasound image augmentation were developed based on representation learning. A novel deformable data augmentation method is proposed and was shown to improve the performance of a neural network trained for target localization in ultrasound images.
Different target tracking algorithms were developed. Algorithms working in ultrasound image space as well as in representation space created by autoencoders were implemented and evaluated. It has been shown that 4D ultrasound tracking in representation space can outperform image space-based tracking in terms of runtime while maintaining comparable accuracy. The target tracking methodology proposed in this work is based on unsupervised learning, is real-time capable, robust, and can be generalized across patients and organs, making it promising for ultrasound guided therapy purposes. The applicability of the representation space-based tracking has been shown in an online robotic ultrasound tracking experiment. Hence, this work proposes a novel method for 4D ultrasound tracking that could be integrated into any therapy domain.

CONTENTS

NOTATION

$\hat{\mathbf{I}}$	Ultrasound image predicted from a neural network.
λ	Weighting parameter.
$[a, b]$	Range of real numbers from a to b.
$\mathbb{G}$	Gray scale intensity value space of ultrasound images containing real numbers in the range $[0, 1]$.
$\mathbf{I} \circ \mathbf{F}$	Warping operation applying the deformation field $\mathbf{F}$ to the image $\mathbf{I}$.
$\mathbf{I}^t$	Ultrasound image with temporal index t from a sequence of ultrasound images.
$\mathbf{I}_i = \mathbf{I}(i)$	Gray scale value of the voxel element i of an ultrasound image $\mathbf{I} \in \mathbb{G}^d$, where d is the image size.
$\mathbf{P}$	3D ultrasound patch taken from a 3D ultrasound image $\mathbf{I}$.
$\mathbf{x} \odot \mathbf{y}$	Element-wise multiplication of vectors $\mathbf{x}$ and $\mathbf{y}$, also known as Hadamard product.
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2.
$\mathcal{T}$	Homogeneous transformation matrix.
$\mathcal{U}_{[0,1]}$	Uniform distribution in the range [0,1].
$\nabla \mathbf{I}$	Nabla operator determines all partial derivatives in $\mathbf{I}$ along all spatial axes.
L-Vi-j	A short-term labeled 4D ultrasound sequence of the liver (L; P for prostate also exists) from the data set used in this work. This is the jth sequence acquired on volunteer Vi.
L-Vi	A long-term 4D ultrasound sequence of the liver (L; P for prostate also exists) acquired on volunteer Vi from the data set used in this work.
Lia	An anatomical landmark (a; b and c also exist) labeled by an observer in ultrasound sequences of the liver (L) of volunteer Vi.

1 INTRODUCTION

Radiation therapy treats tumors by exposing them to ionizing radiation, which damages cells and subsequently destroys the tissue. However, the radiation does not only destroy the tumor, but also healthy tissue. This is disadvantageous and can be risky, especially when treating tumors close to vital organs as these must be spared. When a radiation treatment plan is created, margins are added to the actual tumor tissue to ensure that the entire tumor is treated, even if the tumor moves during treatment [1]. Abdominal organs, such as the liver, and tumors are in constant motion due to intrafractional motion, which occurs during treatment and is caused by processes such as breathing or arbitrary movements of the patient. Because these organs and tumors are soft tissue structures, the motion can be high dimensional, involving translation, rotation, and even deformation, which greatly complicates the planning and delivery of radiation therapy. With the aim of minimizing the irradiation of healthy tissue, methods of motion compensation have been developed [2]. If the exact location of the tumor could be determined during treatment, the radiation beam could be adjusted accordingly, resulting in less damage to healthy tissue.
Different approaches have been developed to track the tumor or defined fiducial markers during treatment and to measure or estimate the target position [3], [4]. A state-of-the-art method is using Cone-Beam Computed-Tomography (CBCT) imaging to visualize implanted fiducial markers during treatment. The drawback of this approach is two-fold: The patient is exposed to additional harmful radiation due to CBCT imaging and fiducial markers have to be implanted in an invasive procedure that involves additional risk. The CyberKnife system (Accuracy, Madison, WI, US) is a robotic arm equipped with a linear accelerator (LINAC) that is guided under repeated X-ray imaging and a stereo camera system that tracks external markers attached to the patient's surface. This system is expensive and also exposes the patient with additional harmful radiation due to the X-ray imaging process. In contrast, the Calypso 4D localization system (Varian Medical Systems, Palo Alto, CA, US) uses a magnetic marker tracking approach by generating and measuring a resonant response from implanted markers utilizing a magnetic array placed near the patient's skin [5]. In addition, systems based on marker tracking rely on the assumption that the motion of the marker and the tumor are correlated. However, this is not always the case, especially for external markers [6], [7].
A promising approach to overcome the drawbacks of additional radiation and the limitation to fiducial tracking is using ultrasound imaging. Its greatest benefit is

providing real-time soft tissue visualization without the use of harmful radiation. It provides the ability to locate and track the tumor directly in the ultrasound images. Furthermore, ultrasound imaging is widely used in clinics and is an inexpensive and mobile imaging modality. Drawbacks of ultrasound imaging include a low signal-to-noise ratio, a small field of view, and the fact that it cannot be used for air-containing areas, such as the lungs, due to physical limitations. In addition, no physician can hold the ultrasound transducer during treatment due to radiation exposure. Hence, the ultrasound transducer must be attached to a robotic arm that guides the transducer on the patient's skin. Furthermore, a robust tracking algorithm is required to locate the target and extract target motion information. For this purpose, different robotic ultrasound setups have been proposed [8], [9]. One commercially available ultrasound system is Clarity Autoscan (Elekta, Stockholm, Sweden), which is designed to guide radiation therapy of the prostate. The ultrasound transducer in this system is attached to a static platform, which limits its use to tumors that move little [10], [11]. Since this work focuses on tracking targets in the liver, where larger target motions occur, this system is not yet sufficiently suitable.
Another advantage of ultrasound imaging is the capability of providing volumetric (3D) images in real-time (3D+t or 4D). In a therapy guidance application where a 3D target, such as a tumor, needs to be tracked, a 3D imaging modality is required to correctly determine the full motion of the target in the 3D space [12]. Although 2D ultrasound imaging is more commonly used in clinical practice for diagnostic purposes, it is not optimal for therapy guidance. In 2D ultrasound images, motion that occurs in the direction of the missing spatial axis, known as out-of-plane motion, cannot be reliably detected because only 2D slices of a 3D structure are provided [13]. However, previously proposed target tracking approaches for ultrasound imaging are mainly developed and evaluated in 2D+t ultrasound showing a lack of 4D ultrasound tracking research [14].

1.1 Purpose

When considering 3D ultrasound images instead of 2D, the complexity of motion tracking and the computational cost increase due to the additional image dimension. This may be one reason for the lack of development of 4D tracking algorithms for ultrasound imaging. Other reasons could be that 3D ultrasound imaging is less common than 2D in clinical practice and that 4D ultrasound data sets are rarely available, especially with labels. 4D ultrasound labeling is a time-consuming task that requires expertise which is often hard to come by. This complicates training of supervised deep learning-based approaches such as neural networks. Previously proposed approaches for 4D ultrasound tracking are either based on neural networks trained in a supervised way [15], [16] or on approaches that do not require any model training such as template matching or deformable transformation models [17], [18]. One approach to overcome the problem of rare labeled data is using unsupervised machine learning methods as these can be trained with unlabeled data. For the purpose of 4D ultrasound tracking, there is a lack of unsupervised learning-based method development in literature.
Hence, this work focuses on the use of autoencoders to learn features in 3D ultra-

sound images for the purpose of target tracking. The tracking process is shifted from the image space into a representation space that is learned in an unsupervised training process. The knowledge and insights gained in this work are intended to contribute to the goal of integrating 4D robotic ultrasound guidance into the clinical workflow. For this purpose, this work provides different insights and methods. A labeled 4D ultrasound data set is generated, evaluated for motion range and labeling variability, and made available[1]. Different unsupervised methods for 3D ultrasound image feature extraction are developed that map 3D ultrasound patches into latent representation vectors. These methods could be used for different applications such as tracking. To facilitate training deep neural networks, two data augmentation techniques based on representation learning are developed for ultrasound images. The first 3D ultrasound augmentation technique that aims to simulate breathing-induced motion patterns is proposed and made available[2] in this work. Finally, different novel methods for performing real-time target tracking in representation space are developed. A real-time capable greedy tracking algorithm is proposed and evaluated in experiments. In addition, a novel strategy for rearranging 3D ultrasound patches to 2D is developed that is shown to be capable of outperforming a state-of-the-art strategy.

1.2 Structure

Four research questions are addressed in this work that aim to contribute to the development of a 4D ultrasound tracking algorithm that works in representation space. Each research question is investigated and answered in a separate chapter. First, basics about the methods used in this work are introduced in chapter 2.
In chapter 3,

Research Question 1
How much does a target in the liver move and how precisely can it be tracked in 3D ultrasound sequences?

is addressed, which aims to show the necessity and the maximum accuracy that can be achieved by a 4D tracking algorithm evaluated with expert-labeled data. This information is considered when evaluating the robustness of the tracking algorithms developed in this work in section 7.

To be able to perform target tracking in representation space, features or representations must be extracted from ultrasound images first. In order to to this,

Research Question 2
How can meaningful and unique features be extracted from 3D ultrasound images?

[1] The labeled 4D ultrasound data set can be made available upon request by emailing rob@uni-luebeck.de.

[2] `https://gitlab.rob.uni-luebeck.de/robPublic/3dus-deformable-augmentation`

is investigated in chapter 4. Two different approaches for feature extraction and description are implemented and tested.

As mentioned above, ultrasound image data sets are rarely publicly available. However, when training deep learning-based methods, such as neural networks, a high amount of training data is required. Therefore,

Research Question 3
Is it possible to perform realistic 3D ultrasound image augmentation using unsupervised deep representation learning and is this augmentation beneficial for a target tracking application?

is researched in chapter 5. Two approaches to perform data augmentation for ultrasound images are developed and evaluated, focusing on image domain transfer and realism.

Finally,

Research Question 4
Can real-time target tracking in 4D ultrasound be performed in representation space and can it outperform image space-based tracking algorithms?

is addressed in chapter 6, representing the main part of this work. Both image space-based and representation space-based tracking algorithms are implemented and evaluated in tracking experiments. In addition, the first online applicability of representation learning-based 4D ultrasound target tracking is demonstrated in a proof-of-concept study. Several methods are proposed that support the performance of real-time 4D ultrasound tracking in representation space, representing a valuable contribution to research.
In order to measure the usability of the tracking algorithms in the therapy guidance domain, requirements are defined that the tracking algorithms must meet, inspired by De Luca et al. [13]:

Type of learning As mentioned before, this work focuses on the use of unsupervised learning-based approaches for target tracking. Therefore, the first requirement is that the tracking algorithm must be fully unsupervised. Methods that do not require any training, such as template matching, also meet this requirement. The advantage of unsupervised approaches in the medical domain is that physicians do not have to spent time on time-consuming data labeling. This can reduce treatment planning effort, especially when training individual patient models or tuning models with individual patient data.

Generalizability The tracking algorithm should be able to cover a wide range of tissue variances in order to generalize across patients. With a generalized tracking algorithm, patient-individual model training or tuning would not be required as the algorithm can be applied to different patients. Such an algorithm can save time and effort in treatment planning.

Real-time capability The runtime of the tracking algorithm is a critical factor. In a robotic ultrasound setup for the purpose of radiation therapy guidance, different systems, such as the ultrasound machine, the robot, and the LINAC, have to work in parallel and share information. The tracking algorithm must not slow down the entire system or cause latency, as this may result in a suboptimal treatment outcome or risk to the patient. Since the purpose of the tracking algorithm is to locate the target in the current ultrasound image without causing system latency, the algorithm must be at least as fast as the imaging frame rate of the ultrasound machine.

Robustness The purpose of the tracking algorithm is to track the target over time with high accuracy and without any outliers or failures. The robustness of the algorithm is quantified by determining an average tracking error. According to the American Association of Physicists in Medicine (AAPM), motion management is required in radiation therapy if the target moves more than 5 mm [19]. Hence, the robustness requirement is that the tracking algorithm must have a tracking error less than 5 mm. However, the goal is to achieve the lowest possible tracking error, as a low tracking error can minimize the risk of damaging healthy tissue during treatment [20].

The algorithms developed in this work are compared and evaluated in terms of these requirements in chapter 7. In addition, state-of-the-art image space-based tracking algorithms proposed in literature are compared to the best performing algorithm of this work according to the requirements in chapter 8.

2 BASICS OF REPRESENTATION LEARNING

In this work, methods from the field of Artificial Intelligence (AI) and its sub-disciplines are used. For orientation, these fields and their interrelationships are illustrated in Figure 2.1. This chapter introduces the basics of methods from the fields of machine learning and deep learning, sub-disciplines of AI, which will be used in this thesis to learn representations and extract image features from 3D ultrasound images. Like illustrated in Figure 2.1, machine learning is a sub-discipline of artificial intelligence. Artificial intelligence is the generic term for systems that make their own decisions. This can be an algorithm following defined rules such as image feature extraction algorithms or a machine learning method. Such a method is characterized by the fact that it makes decisions based on knowledge it has extracted from prior observations. Therefore, for creating a machine learning model, a training data set is required. A sub-discipline of machine learning is representation learning. Methods from this field aim to generate representations from raw data in order to reduce the dimensionality and to extract features. However, other than in tasks such as classification where the objective is obviously defined, a difficulty in representation learning is to define a clear objective or target function for training [21]. Typically, representation learning is applied before the targeted task to improve the performance of algorithms or to facilitate machine learning methods. A basic method from the field of representation learning is the principal component analysis (PCA) [22]. A sub-discipline of representation learning is deep learning. Methods in this field are able to extract features from raw data and directly make decisions based on these features. Commonly used deep learning methods are neural networks. For processing medical image data convolutional neural networks (CNNs) are commonly used in literature [23]. These are introduced in section 2.3. In addition, in the following methods are presented that are used in this work to learn representations for 3D ultrasound image data. An ultrasound image is typically a gray scale image of a given size, such as $m \cdot n \cdot l$ vx in the case of 3D. The gray scale values are integers in the range $[0, 255]$. However, in this work the gray scale value range of ultrasound images is normalized and defined by

$$\mathbb{G} = \{g_i \in \mathbb{R} \mid g_i \geq 0 \wedge g_i \leq 1\}. \tag{2.1}$$

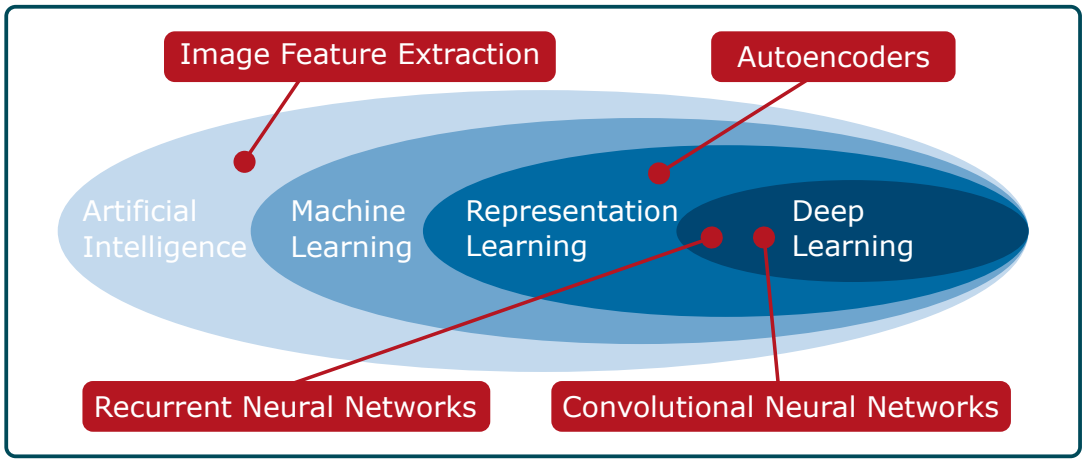

Figure 2.1: A Venn diagram illustrating machine learning is a sub-discipline of artificial intelligence, representation learning is a sub-discipline of machine learning and deep learning is a sub-discipline of representation learning. Methods that are used in this work for feature extraction are assigned to their sub-discipline.

Figure 2.2: The three steps performed by image feature extraction algorithms. Keypoints are detected in an image and described by a unique description vector to form a feature. In the last step, features are matched with features from another image to determine correspondences between these images.

2.1 Image Features

Image feature extraction is a category of algorithms from the area of artificial intelligence (see Figure 2.1). These algorithms follow defined rules without the need of model training. The purpose of such an algorithm is to detect and describe meaningful features in image data. Features can be extracted from whole images leading to global features representing these images in an encoded form. Such features can be used, e.g., for image classification. However, for tasks that require local image information, such as target tracking, local image features can be extracted [24]. Using image features instead of the image itself for a downstream task can provide benefits such as reducing the dimensionality of the data, resulting in reduced storage requirements and increased algorithm speed. In addition, data dimensionality reduction can remove noise as well as irrelevant and redundant information leading to improved data quality [25]. There are different local image feature extraction algorithms proposed in literature. Basically, all of them follow the same steps that are illustrated in Figure 2.2. First, points of interest, so called *keypoints*, are detected in the image. In a second step, description vectors are determined for the keypoints in order to equip them with meaningful and unique descriptions. The descriptions

are determined based on local image information in the area around the keypoint. The combination of the keypoint location and the corresponding description is called *feature*. When using the features in a downstream task such as image registration, feature matching is performed in a third step. Correspondences between sets of features from two different images are determined by finding those features that have a minimum distance between the description vectors.

2.1.1 Feature Detection

The goal of feature detection is to find keypoints, i.e. interesting points in an image. Such keypoints typically are edges or corners. Therefore, feature detection algorithms are often based on determining image gradients, which is often implemented using the sobel operator [26]. A basic feature detection algorithm is the Harris corner detector [27]. Corners are interesting points as they can simply be defined as a location $\mathbf{x}$ where the intensity variation in any direction is maximal [28]. Determining whether a location $\mathbf{x}$ in a 2D image $\mathbf{I} \in \mathbb{G}^{m \times n}$ is a corner is done by first calculating the matrix

$$\mathbf{M}(\mathbf{x}) = \sum w(\mathbf{x}) \left(\nabla \mathbf{I}(\mathbf{x}) \nabla \mathbf{I}(\mathbf{x})^T \right) = \begin{pmatrix} \sum w(\mathbf{x}) \mathbf{I}_x^2 & \sum w(\mathbf{x}) \mathbf{I}_x \mathbf{I}_y \\ \sum w(\mathbf{x}) \mathbf{I}_x \mathbf{I}_y & \sum w(\mathbf{x}) \mathbf{I}_y^2 \end{pmatrix} \tag{2.2}$$

where $w(\mathbf{x})$ is the window function, ∇ is the gradient operator and $\mathbf{I}_x$ and $\mathbf{I}_y$ are the partial derivatives of $\mathbf{I}$ in the x- and y-direction of the image, respectively. Second, for deciding whether the point $\mathbf{x}$ is a corner, the eigenvalues λ_i with $i \in \{1, 2\}$ of $\mathbf{M}$ are considered. Three cases are defined for differentiating between homogeneous regions ($\lambda_i \approx \lambda_j \approx 0$), edges ($\lambda_i \gg \lambda_j \approx 0$) and corners ($\lambda_i > \lambda_j \gg 0$) with $i \neq j$ [28]. Another feature detector is the Difference of Gaussian (DoG) detector that aims to locate blobs in the image [29]. The DoG detector has the benefit that it combines different scale spaces of the image providing the possibility to detect features at different scales.

2.1.2 Feature Description

The objective of a feature descriptor is to find description vectors for keypoints in order to robustly distinguish between different features. To be usable for a downstream task, a feature descriptor must cover multiple requirements. Considering the target tracking task, feature descriptions must be unique, reproducible and comparable to be able to robustly distinguish between different features and match similar features. Common image feature descriptors are the *Scale Invariant Feature Transform* (SIFT) [29] and *Speeded-Up Robust Features* (SURF) [30]. These algorithms contain both, a feature detector and descriptor. The detectors of both algorithms are based on the Harris and the DoG feature detectors shown above. Feature description is done based on image intensity information in the local region of the keypoint. The description vectors of both algorithms provide rotation and scale invariance due to considering the orientation of the features calculated based on the intensities in the local neighborhood and considering several image scale spaces. However, they differ in the way they are created and in length as SIFT produces a vector $\mathbf{z} \in \mathbb{R}^{128}$ and SURF $\mathbf{z} \in \mathbb{R}^{64}$ [29], [30].

2.1.3 Feature Matching

The process of feature detection and description corresponds to a mapping of local image information into a representation space. In this representation space, feature matching can be performed. This step is required for different tasks such as image registration, stitching or target tracking. The objective is to find feature correspondences between two sets of features extracted from two different images. For this, distance metrics such as the Euclidean distance can be used. Based on the feature correspondences, the motion that occurred between the images can be determined.

2.2 Machine Learning

Machine learning models can be categorized, based on the way of training them, into the categories *supervised* and *unsupervised*. In supervised learning, the data set used for training must contain labels that provide a ground truth for the targeted task. Hence, a data sample (x, y) must contain a data point $x \in X$ and the corresponding label $y \in Y$. A machine learning model $f : X \mapsto Y$ is basically trained by minimizing the prediction error of the model with regards to the label by

$$\min_f f(\mathbf{x}) - y. \tag{2.3}$$

A supervised learning method is, e.g., a basic neural network trained for a classification task. In contrast, in unsupervised learning no ground truth is required for model training. Unsupervised learning methods aim to learn useful features from the training data on their own. Examples for unsupervised learning methods are the autoencoder (presented in section 2.4) and the k-means clustering algorithm (presented in section 2.2.2). However, both learning approaches have in common that a training data set is required. In the following, common data preparation steps are presented that are performed before model training is applied.

2.2.1 Data Preparation

When preparing a given data set for training a machine learning model, typically, the data set is split into three subsets for training, validation and test. The training and validation sets are used in the training process. Both data sets are passed through the model (feed forward) to generate an output and fed into a cost or loss function to determine the model performance based on the training and validation data sets. However, the model weights are only updated based on the training loss. The validation loss is determined to be able to rate the current performance of the model on new data. This enables to measure whether the model generalizes or whether it is overfitting to the training data. The test data is used after training to evaluate the performance of the model.
Splitting the data set into fixed training, validation and test subsets can be problematic when the data set is small. Small data sets are often limited in their ability to cover the full range of possible data variance. Splitting such a data set can lead to an imbalance in data variance between the training and test subsets, making the

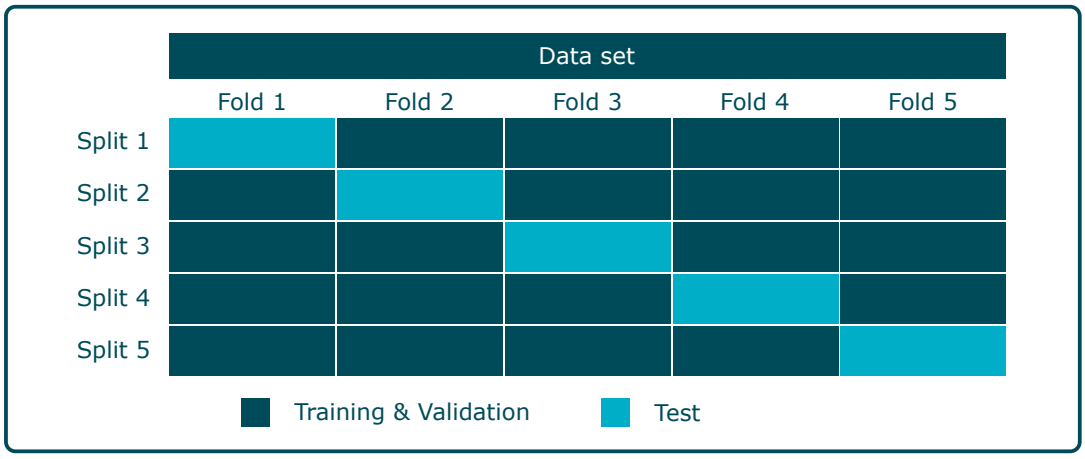

Figure 2.3: The procedure of data splitting in the 5-fold cross-validation. The data set is split into five non-overlapping folds. In five splits, four folds are used for model training and evaluation and one fold is used for model testing. In each split another fold is defined as the test fold.

evaluation of model performance unreliable. To prevent this issue during model evaluation, a k-fold cross-validation can be applied. Here, the data set is split into $k \in \mathbb{N}$ non-overlapping subsets, so called *folds*. In Figure 2.3 the 5-fold cross-validation is shown as an example. Based on the k folds, k different *splits* are generated, where each split defines a different fold for testing, while the remaining folds are used for training and validation. Hence, the model is trained and tested k times. When evaluating the model performance, the k sets of performance test results are averaged [31].

2.2.2 k-Means Clustering

Clustering algorithms belong to the field of unsupervised machine learning that aim to partition a set of data points into several subsets of data points, so called *clusters*. Thereby, the partitioning shall be done in a meaningful way. After clustering data, such as a set of one-dimensional image feature descriptions, the clustering result can be used for classifying new feature descriptions in an unsupervised way. One commonly used method for performing clustering is the k-means clustering algorithm [32].
For a given set of data points $\mathcal{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ k-means clustering aims to find $k \leq n$ clusters $\mathcal{C} = \{\mathcal{X}_1, ..., \mathcal{X}_k\}$ by finding

$$\arg\min_{\mathcal{C}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}_i} \|\mathbf{x} - \mathbf{c}_i\|_2 \tag{2.4}$$

where $\|\cdot\|_2$ is the Euclidean distance and $\mathbf{c}_i$ denotes the centroid of the i-th cluster with the size $|\mathcal{C}_i|$ defined by

$$\mathbf{c}_i = \frac{1}{|\mathcal{C}_i|} \sum_{\mathbf{x} \in \mathcal{C}_i} \mathbf{x}. \tag{2.5}$$

The number of clusters k is a hyperparameter of the algorithm that must be set manually. Therefore, the k-means algorithm assumes that the number of clusters to

be found is known in advance.

2.3 Convolutional Neural Networks

The neural network is a method from the field of deep learning that is basically based on the biological brain. It is built in layers, where the first layer is called *input layer*, the last layer is called *output layer* and all other layers in between are called *hidden layers*. Neural networks are commonly used for various tasks such as classification and regression, as they often outperform other machine learning methods [33]. In order to extract features from images, convolutional neural networks (CNNs) are shown to be effective [34]. Hence, CNNs are used in this work to process 3D ultrasound image data. The main components of a CNN are introduced in the following.
As the name suggests, convolutional neural networks contain convolutional layers. These are used to extract spatial features from images. The discrete convolution between an image $\mathbf{I}$ and a filter kernel $\mathbf{K}$ is defined as

$$y(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{K}(\mathbf{p}_n) \cdot \mathbf{I}(\mathbf{p}_0 + \mathbf{p}_n) \tag{2.6}$$

where $\mathbf{p}_0$ and $\mathbf{p}_n$ are the position in the image $\mathbf{I}$ and a position in the filter kernel $\mathbf{K}$, respectively. The size of the set of neighbor positions $\mathcal{R}$ considered depends on the size of the filter kernel. The elements of the kernel $\mathbf{K}$ are the weights that are learned during model training [31]. The first layer of a convolutional neural network typically is a convolutional layer. The output of such a layer is called *feature map*.
Another component often used in CNNs is the pooling layer. Pooling layers do not have learnable parameters but a kernel of a defined size. This type of layer is for reducing the size of the input feature map by passing only the maximum (max-pooling layer) or average value (average-pooling layer) of the input feature map elements covered by the kernel. The functionality of the max- and average-pooling layers for a 2D feature map are illustrated in Figure 2.4 for a kernel size of 2. In a CNN, convolutional and pooling layers often are arranged in sequence in order to extract spatial features and reduce the size of the image. It may also be necessary to increase the size of a feature map in a CNN such as in an autoencoder that is presented in section 2.4 in detail. In this case, the upsampling layer can be used. As can be seen in Figure 2.4, the upsampling layer basically aims to be the inversion of the pooling layer. It increases the size and sets the elements of the feature map by interpolating them based on the previous feature map.
Last, fully-connected layers are used in order to map the feature map into a latent representation and to form it into the required output size of the CNN. Basically, a set of fully-connected layers is a multi-layer perceptron (MLP) which consists of a set of perceptrons. A perceptron is defined by

$$y = \sigma\left(\mathbf{w}^T\mathbf{x} + b\right) \tag{2.7}$$

determining an output y based on a weighted sum of the input $\mathbf{x}$ with the learnable weights $\mathbf{w}$, the learnable bias b and the activation function $\sigma : \mathbb{R} \mapsto \mathbb{R}$. Convolutional and fully-connected layers are typically equipped with activation functions.

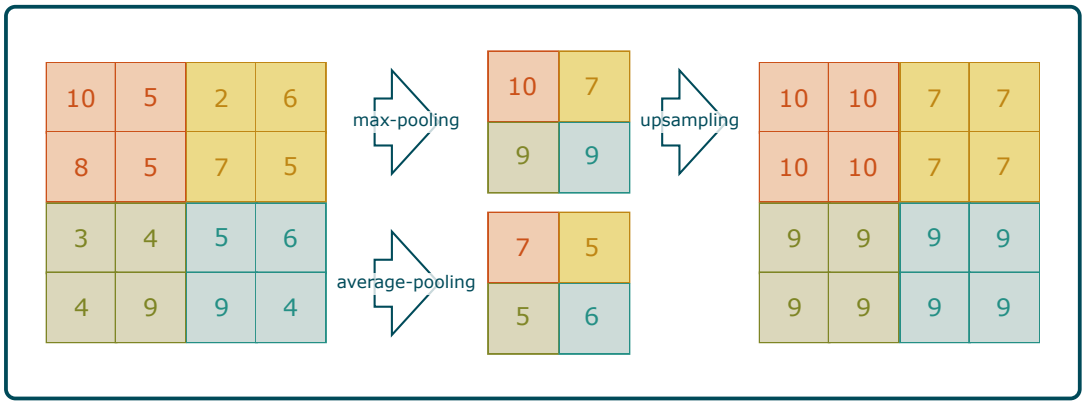

Figure 2.4: Illustration showing the functionality of the max- and average-pooling layers as well as the upsampling layer in the 2D case for a kernel with a side length of 2. In pooling, the feature map size is reduced by the length of the kernel and the elements contain the maximum or the average values of the feature map area covered by the pooling kernel. In upsampling, the feature map size is increased by the length of the kernel and the missing elements in the new feature map are filled by the known values.

Common activation functions are sigmoid:

$$\sigma_{\text{sigmoid}}(x) = \frac{1}{1+e^{-x}}, \tag{2.8}$$

tangens hyperbolicus (tanh):

$$\sigma_{\tanh}(x) = 1 - \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}, \tag{2.9}$$

rectified linear unit (ReLU):

$$\sigma_{\text{ReLU}}(x) = \max(0, x), \tag{2.10}$$

or slightly adapted ReLU versions like Exponential Linar Unit (ELU)

$$\sigma_{\text{ELU}}(x) = \begin{cases} x, & \text{if } x > 0 \\ \lambda \cdot (e^{x}-1), & \text{otherwise,} \end{cases} \tag{2.11}$$

as well as Leaky ReLU (LReLU) or Parametric ReLU (PReLU)

$$\sigma_{\text{adapted ReLU}}(x) = \begin{cases} x, & \text{if } x > 0 \\ \lambda \cdot x, & \text{otherwise.} \end{cases} \tag{2.12}$$

In PReLU, the weighting parameter λ is learnable, while in LReLU and ELU, it is manually set and fixed.

Neural Network Training

The training of a neural network is performed iteratively by using gradient descent to update the weights of the network. Gradient descent is a way to minimize an

objective function $J(\theta)$ (also called loss function) depending on parameters θ by updating them using the rule

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta) \tag{2.13}$$

where η is the learning rate which denotes the step size to reach the local minimum [35]. There are different algorithms, so called optimizers, proposed in literature that aim to perform gradient descent for training neural networks. In this work, the *RMSprop* [35] and the *Adaptive Moment Estimation* (Adam) [36] optimizers are used. To prevent the neural network from overfitting, regularization methods can be applied in neural network training. The most common methods are dropout, batch normalization and early stopping [37]. In dropout, a number of layer outputs are randomly ignored in every hidden layer during the training process. The rate of outputs to be ignored is a hyperparameter that needs to be set manually for every layer [38]. However, when using the neural network for prediction, all layer outputs are used, meaning dropout is only applied during training. In batch normalization, the values of the layer outputs are normalized by scaling them into the range $[-1, 1]$. This is done by applying

$$\begin{aligned} y_i = \mathrm{BN}_{\gamma\beta}(x_i) &= \gamma\hat{x}_i + \beta \quad \text{with} \\ \hat{x}_i &= \frac{x_i - \mu}{\sqrt{\sigma^2}} \\ \mu &= \frac{1}{m}\sum_{i=1}^{m} x_i \\ \sigma &= \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu)^2 \end{aligned} \tag{2.14}$$

to each data sample x_i in a batch separately. The parameters γ and β are learnable weights [39]. In contrast to dropout, batch normalization is also applied when using the neural network for prediction as the calculations directly affect the output of the layers. Early stopping aims to stop the training process if the validation loss reached the minimum. This is done by monitoring the validation loss over time. As long as the validation loss decreases, neural network training is continued. If the validation loss does not decrease or even increases for a certain number of training epochs, the training process is stopped [31].

2.4 Autoencoders

The autoencoder is a method from the field of representation learning. Since this work aims to process ultrasound image data, CNNs from the field of deep learning are used here to form autoencoders. Therefore, the autoencoder described in the following can be classified into the field of deep representation learning (see Figure 2.1). The key idea of the autoencoder is to learn the ability to map an image data sample into a meaningful representation vector that only contains the most important features of the image data. Hence, redundant or unimportant image information such as noise is reduced in representation space. For training an autoencoder, no

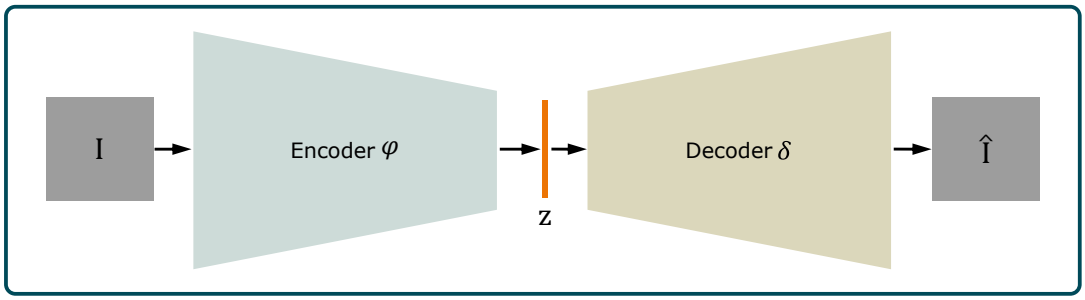

Figure 2.5: The general architecture of an autoencoder consists of an encoder and a decoder with the latent representation space in the bottleneck between them. The encoder maps the image $\mathbf{I}$ into a representation vector $\mathbf{z}$ and the decoder maps $\mathbf{z}$ into the image space by predicting an image reconstruction $\hat{\mathbf{I}}$.

labels are required making it an unsupervised learning method [40]. The general structure of an autoencoder is an encoder-decoder type as illustrated in Figure 2.5. It consists of an encoder $\varphi : \mathbb{G}^d \mapsto \mathbb{R}^k$ that maps an input image $\mathbf{I} \in \mathbb{G}^d$ into a latent representation $\mathbf{z} \in \mathbb{R}^k$, the so called bottleneck ($k \ll d$), and a decoder $\delta : \mathbb{R}^k \mapsto \mathbb{G}^d$ that reconstructs the image from the representation vector [41]. Training an autoencoder is done in an unsupervised way as the autoencoder is evaluated on its ability to reconstruct the input data sample $\mathbf{I}$. Therefor, the input data itself serves as ground truth and no further labels are required so that the autoencoder is trained by

$$\arg\min_{\varphi,\delta} d\left(\delta(\varphi(\mathbf{I})), \mathbf{I}\right). \tag{2.15}$$

However, since in a real world scenario the autoencoder will not be able to reconstruct the input image perfectly due to information loss in the bottleneck, the autoencoder predicts an image $\hat{\mathbf{I}} = \delta(\varphi(\mathbf{I})) \approx \mathbf{I}$ that aims to be as similar as possible to the input image $\mathbf{I}$. The autoencoder is trained by minimizing the difference between the input and the output image by using a similarity metric as a reconstruction loss function. A common loss is the mean-squared-error (MSE) loss defined by

$$L_{\text{Reconstruction}}(\mathbf{I}, \hat{\mathbf{I}}) = L_{MSE}(\mathbf{I}, \hat{\mathbf{I}}) = \frac{1}{N}\sum_{i=1}^{N}(I_i - \hat{I}_i)^2 \tag{2.16}$$

where N is the number of elements in the images $\mathbf{I}$ and $\hat{\mathbf{I}}$ and i denotes the elements of the image.
Besides the simple autoencoder described above, other variants of autoencoders exist as well providing some kind of benefit or aim to solve a certain task. For instance, a denoising autoencoder aims to remove noise from input data to reconstruct a cleaned version of the input data [40], [42]. In the following, two kinds of autoencoder, the *variational autoencoder* (VAE) and the *sliced-wasserstein autoencoder* (SWAE) are presented. These autoencoders aim to form the latent representation space in a continuous way. With this, the autoencoder gets the capability to be generative. The simple conventional autoencoder described above does not necessarily have this characteristic. This means that a randomly sampled latent vector taken from the representation space of a conventional autoencoder can lead to an unrealistic output when fed into the decoder. Thus, in order to get a generative model and a

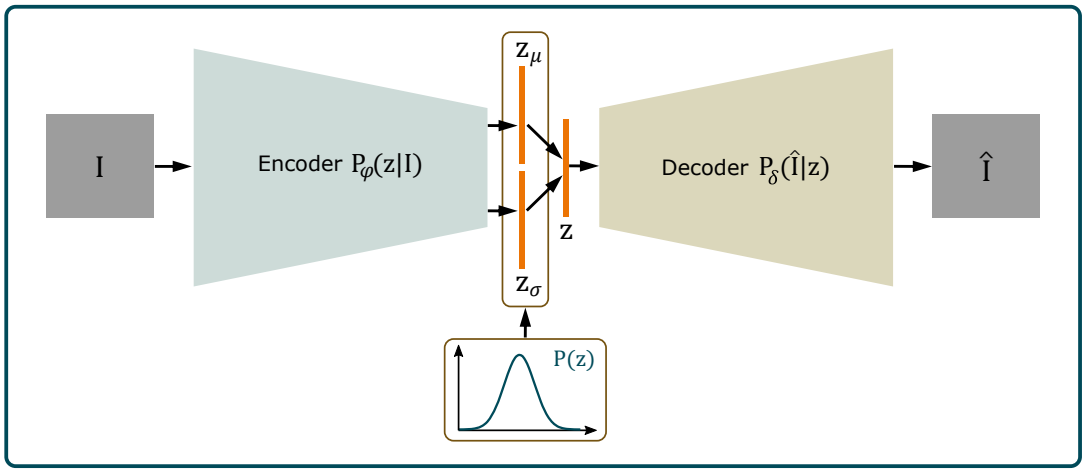

Figure 2.6: The general architecture of a VAE consisting of an encoder-decoder structure. Getting an input image, the encoder predicts two latent representation vectors $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$ denoting the mean and standard deviation of a normal distribution. The final representation vector $\mathbf{z}$ is sampled from the distribution defined by these parameters. The decoder predicts an image reconstruction from the representation vector $\mathbf{z}$.

continuous representation space, autoencoder types such as the VAE and the SWAE are proposed in literature. In the following, the basic principles of the VAE and the SWAE are presented and the main differences are highlighted.

2.4.1 Variational Autoencoders

The VAE is a probabilistic model and was proposed by Kingma and Welling [43]. To provide the ability to be generative, the VAE is trained to map an input data sample $\mathbf{I} \in \mathbb{G}^d$ into a prior distribution $P(\mathbf{z})$ instead of a discrete point in representation space. For this, the standard Normal distribution is commonly used as prior, meaning $P(\mathbf{z}) = \mathcal{N}(0, 1)$ [44], [45]. Therefore, the encoder and decoder are not deterministic but probabilistic mapping functions sampling the representation vector $\mathbf{z}$ from the distribution $P_\varphi(\mathbf{z}|\mathbf{I})$ and the reconstructed image $\hat{\mathbf{I}}$ from $P_\delta(\hat{\mathbf{I}}|\mathbf{z})$. The general architecture of a VAE is illustrated in Figure 2.6. The encoder maps the input image $\mathbf{I}$ into the latent representation vectors $\mathbf{z}_\mu$, $\mathbf{z}_\sigma \in \mathbb{R}^k$ representing the mean and standard deviation of a normal distribution that denotes the latent representation of the input sample. To get a single representation vector, the latent representation of the input image $\mathbf{I}$ is sampled from the normal distribution defined by the mean and standard deviation vectors by

$$\mathbf{z} \sim \mathcal{N}\left(\mathbf{z}_\mu, \mathbf{z}_\sigma^2\right). \tag{2.17}$$

When training a VAE, the loss function given in equation 2.16 is expanded by a regularization term in order to enforce the latent distribution to be a standard Normal distribution $P_\varphi(\mathbf{z}|\mathbf{I}) \stackrel{!}{=} P(\mathbf{z}) = \mathcal{N}(0, 1)$. For this, the Kullback-Leibler Divergence (KLD) is used. KLD is a metric for comparing two distributions and quantifying the distance between them [31]. For the VAE presented here KLD is

defined as

$$\mathrm{KLD}\left(P_\varphi(\mathbf{z}|\mathbf{I}) \parallel P(\mathbf{z})\right) = \mathbb{E}_{\mathbf{z}\sim P}\left[\frac{P(\mathbf{z})}{P_\varphi(\mathbf{z}|\mathbf{I})}\right] = \mathbb{E}_{\mathbf{z}\sim P}\left[\log P(\mathbf{z}) - \log P_\varphi(\mathbf{z}|\mathbf{I})\right]. \quad (2.18)$$

Due to the use of the standard Normal distribution as prior ($P(\mathbf{z}) = \mathcal{N}(0,1)$), the KLD can be calculated by

$$\mathrm{KLD}\left(\mathbf{z}_\mu, \mathbf{z}_\sigma\right) = \frac{1}{2}\sum_{i=1}^{k}\left(\mathbf{z}_{\sigma,i}^2 + \mathbf{z}_{\mu,i}^2 - 1 - \log(\mathbf{z}_{\sigma,i}^2)\right) \quad (2.19)$$

in a closed form where k is the dimensionality of the representation space [41], [45]. With this, the loss function of the VAE can be defined as

$$L_{\mathrm{VAE}}\left(\mathbf{I}, \hat{\mathbf{I}}, \mathbf{z}_\mu, \mathbf{z}_\sigma\right) = \lambda_{\mathrm{Reconstruction}} L_{\mathrm{Reconstruction}}(\mathbf{I}, \hat{\mathbf{I}}) + \lambda_{KL} L_{KL}(\mathbf{z}_\mu, \mathbf{z}_\sigma) \quad (2.20)$$

where $L_{KL}(\mathbf{z}_\mu, \mathbf{z}_\sigma) = \mathrm{KLD}\left(\mathbf{z}_\mu, \mathbf{z}_\sigma\right)$ and $\lambda_{\mathrm{Reconstruction}}$ and λ_{KL} are a weighting parameters. For the training process, however, an issue remains using this setup. Due to the fact that $\mathbf{z}$ is sampled from a normal distribution defined by $\mathbf{z}_\mu$ and $\mathbf{z}_\sigma$, the representation vector $\mathbf{z}$ is not deterministic but stochastic. Therefore, during training the backpropagation cannot be executed at this point since there is no derivation for this stochastic function. To find a solution for this, the *reparameterization trick* is applied [41]. The trick is to put the stochastic component to a place where no derivative is required during training. This ensures that all functions requiring a derivative for backpropagation are deterministic and therefore derivable. This is done by replacing the sampling procedure from equation 2.17. A random variable

$$\varepsilon \sim \mathcal{N}(0,1) \quad (2.21)$$

is introduced so that sampling the latent representation vector $\mathbf{z}$ can be performed by

$$\mathbf{z} = \mathbf{z}_\mu + \mathbf{z}_\sigma \odot \varepsilon \quad (2.22)$$

where $\odot$ denotes the Hadamard (element-wise) product.

2.4.2 Sliced-Wasserstein Autoencoders

The SWAE proposed by Kolouri et al. [46] is another type of autoencoder with similar characteristics as the VAE. It also has generative capability due to a continuous latent representation space formed according to a prior distribution [47]. However, in contrast to the VAE, the representation space is not formed by sampling each data sample from an own distribution but by forming the whole representation space according to a prior distribution. This difference between the VAE and the SWAE is illustrated in Figure 2.7. In VAE, the reconstructed images of two different but similar input images $\mathbf{I}^1$ and $\mathbf{I}^2$ can be equal due to the sampling procedure in the representation space of the VAE so that

$$\delta\left(\varphi\left(\mathbf{I}^1\right)\right) = \hat{\mathbf{I}} = \delta\left(\varphi\left(\mathbf{I}^2\right)\right). \quad (2.23)$$

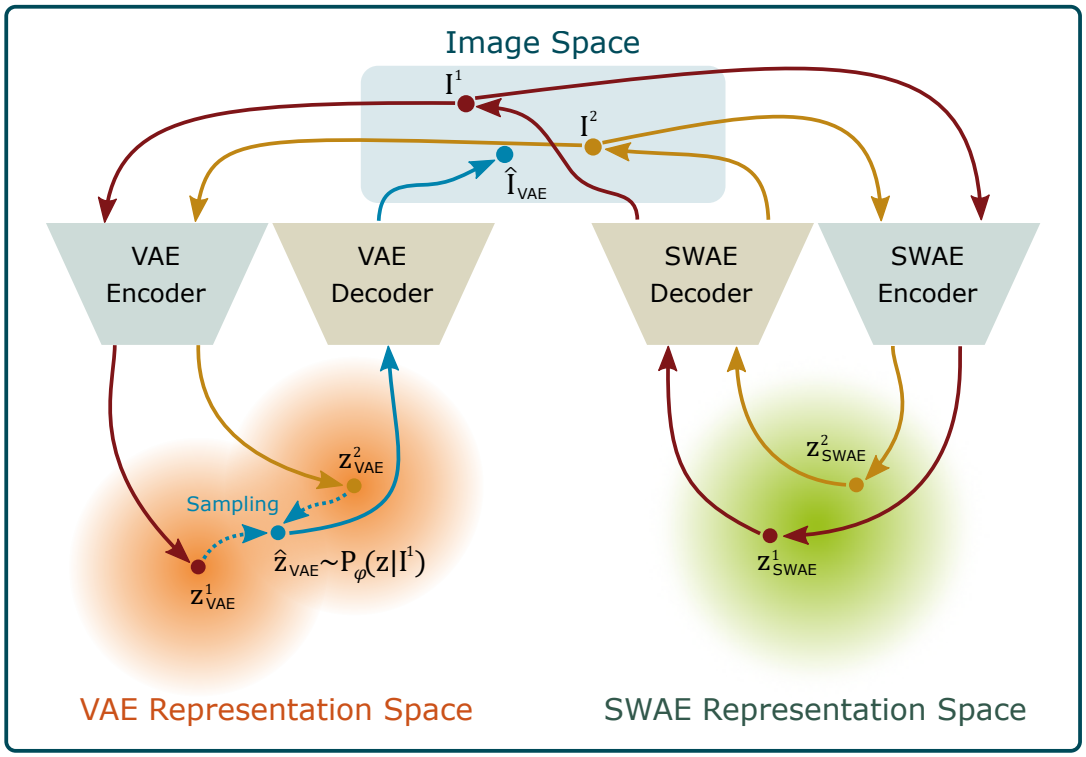

Figure 2.7: Illustration of the process of encoding and decoding data from image space into the representation space of a VAE (left) and a SWAE (right). The representation spaces are formed in different ways as the VAE representation space consists of a set of prior distributions and the SWAE representation space is formed according to one instance of a prior distribution.

In SWAE, there is no such sampling procedure making the image reconstruction deterministic and unique.
The architecture of a SWAE corresponds to the architecture of a conventional autoencoder shown in Figure 2.5. Similar to the VAE, the SWAE is also trained by using a reconstruction and a regularization loss defined by

$$L_{\text{SWAE}}(\mathbf{I}, \hat{\mathbf{I}}, p_{\mathbf{z}}, p_\gamma) = \lambda_{\text{Reconstruction}} L_{\text{Reconstruction}}(\mathbf{I}, \hat{\mathbf{I}}) + \lambda_{SW} L_{SW}(p_{\mathbf{z}}, p_\gamma) \tag{2.24}$$

where $p_{\mathbf{z}}$ and p_γ are the latent and the prior distribution, respectively, and $\lambda_{\text{Reconstruction}}$ and λ_{SW} are weighting parameters. Here, the regularization loss $L_{SW}(\cdot)$ is the sliced-Wasserstein distance that is based on the Wasserstein distance which is defined by

$$W_c(p_X, p_Y) = \int_0^1 c\left(P_X^{-1}(\tau), P_Y^{-1}(\tau)\right) d\tau \tag{2.25}$$

for two one-dimensional distributions p_X and p_Y with the corresponding cumulative distributions P_X and P_Y and a distance metric function $c(\cdot)$. Metaphorically speaking, the Wasserstein distance is a measure for the transport costs required to bring all samples from one cumulative distribution into another cumulative distribution [46]. For one-dimensional distributions the Wasserstein distance has the closed-form solution shown in equation 2.25. For higher-dimensional distributions, however, the Wasserstein distance has no closed-form solution so that solving an optimization

problem is required for determining the Wasserstein distance between two high-dimensional distributions [48]. Therefore, as the latent representation space of an autoencoder typically has more than one dimension, using the Wasserstein distance would highly complicate the training process.
For this reason, the sliced-Wasserstein distance is used that basically determines the one-dimensional Wasserstein distance several times for two high-dimensional distributions. Due to this, the sliced-Wasserstein distance shares similar properties with the Wasserstein distance, but it is much easier to compute [46], [49]. For determining the sliced-Wasserstein distance, one-dimensional slices are randomly taken from the high-dimensional distributions and compared to each other. Slicing from a high-dimensional distribution can be formulated by the Radon transform defined by

$$\mathcal{R}_{p_X}(t,\theta) = \int_X p_X(x)\delta_{\text{Dirac}}(t-\theta\cdot x)dx, \quad \forall\theta\in\mathbb{S}^{k-1},\ \forall t\in\mathbb{R} \tag{2.26}$$

where $\mathbb{S}^{k-1}$ is the k-dimensional unit sphere and $\delta_{\text{Dirac}}(\cdot)$ is the Dirac distribution function [46]. This means, $\mathcal{R}_{p_X}(\cdot,\theta)$ is a one-dimensional slice of the distribution p_X for any fixed $\theta\in\mathbb{S}^{k-1}$. Combining the equations 2.25 and 2.26, the sliced-Wasserstein distance can be defined as

$$SW_c(p_X,p_Y) = \int_{\mathbb{S}^{k-1}} W_c\left(\mathcal{R}_{p_X}(\cdot,\theta),\mathcal{R}_{p_Y}(\cdot,\theta)\right)d\theta \tag{2.27}$$

which is used as sliced-Wasserstein loss when training an SWAE ($L_{SW}(p_{\mathbf{z}},p_\gamma) = SW_c(p_{\mathbf{z}},p_\gamma)$). Here, $p_{\mathbf{z}}$ and p_γ are the distribution of the representation space and the prior distribution, respectively. As distance metric function, $c(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|_2^2$ is chosen. The prior distribution p_γ can be any samplable distribution. Kolouri et al. [46], e.g., trained SWAEs with a ring-, square- and sphere-shaped latent representation space.

2.5 Recurrent Neural Networks

Recurrent neural networks (RNNs) are a category of deep neural networks designed for processing time-resolved data such as audio or video data. Therefore, the RNN can be used to extract temporal features from input data. In contrast to other types of neural networks, in an RNN the input data is passed in the correct temporal order, called sequence data, so that the temporal information can be taken into account by the RNN. This is done by using hidden states that basically have memory from previous steps and that affect the prediction in combination with the data from the current step [31]. The components of an RNN are called *cells*. To define the mathematical formulation for the standard RNN cell, the formulation for the perceptron given in equation 2.7 is extended by adding a hidden state $\mathbf{h}_t$ and learnable weights $\mathbf{W}_h$. The formulation for the standard RNN cell can be given as

$$\mathbf{y}_t = \mathbf{h}_t = \sigma\left(\mathbf{W}_h\mathbf{h}_{t-1} + \mathbf{W}_x\mathbf{x}_t + \mathbf{b}\right). \tag{2.28}$$

Here, t is the considered time step. However, the standard RNN cell is not capable of handling long-term dependencies in the data. This means, the performance of

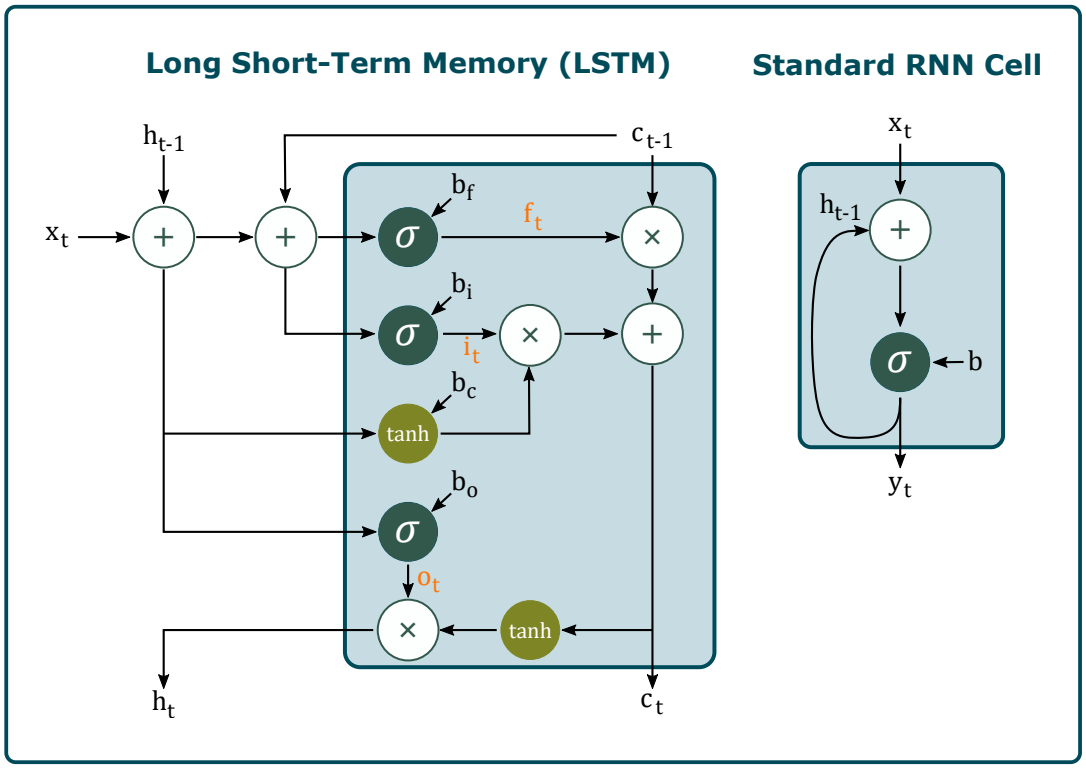

Figure 2.8: Comparison of the the LSTM (left) and the standard RNN cell (right) architectures. In LSTM, the intermediate steps $\mathbf{f}_t$, $\mathbf{i}_t$ and $\mathbf{o}_t$ given in equation 2.29 are marked in orange.

the RNN decreases as the temporal distance between relevant input increases [50]. Therefore, improved RNN cells such as *Long Short-Term Memory* (LSTM) [51] and *Gated Recurrent Unit* (GRU) [52] cells were proposed that are able to solve this issue.

2.5.1 Long Short-Term Memory

The most common type of RNN cell in literature is the LSTM proposed by Hochreiter and Schmidhuber [51] and improved by Gers et al. [53]. In LSTM, gates are inserted into the cell in order to improve the ability to remember previous data samples and to be able to forget unimportant information. To compare the standard RNN cell and the LSTM, the architectures are presented in Figure 2.8. Mathematically, the LSTM is defined as

$$\begin{aligned}
\mathbf{f}_t &= \sigma\left(\mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{W}_{fx}\mathbf{x}_t + \mathbf{W}_{fc} \odot \mathbf{c}_{t-1} + \mathbf{b}_f\right) \\
\mathbf{i}_t &= \sigma\left(\mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ix}\mathbf{x}_t + \mathbf{W}_{ic} \odot \mathbf{c}_{t-1} + \mathbf{b}_i\right) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{W}_{cx}\mathbf{x}_t + \mathbf{b}_c\right) \\
\mathbf{o}_t &= \sigma\left(\mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{ox}\mathbf{x}_t + \mathbf{W}_{oc} \odot \mathbf{c}_t + \mathbf{b}_o\right) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned} \tag{2.29}$$

where $\sigma(\cdot)$ is an activation function, $\odot$ denotes the Hadamard product, $\mathbf{f}_t$, $\mathbf{i}_t$ and $\mathbf{o}_t$ are the forget, input and output gate, respectively, and $\mathbf{c}_t$ and $\mathbf{h}_t$ are the cell output

and the hidden state at time step t [50].
The LSTM is widely used in literature for tasks where temporally ordered data sequences need to be analyzed. Tasks such as time series prediction, natural language processing and video data processing are tackled by using LSTM-based neural networks [54].

2.5.2 Convolutional Long Short-Term Memory

The LSTM cell shown above is designed for processing one-dimensional data. In order to process higher dimensional data such as images, Shi et al. [55] proposed the *Convolutional Long Short-Term Memory* (ConvLSTM) cell. Basically, ConvLSTM has the same structure as the LSTM but instead of multiplications, convolutions are performed in ConvLSTM. This characteristic provides the capability to use the ConvLSTM for extracting spatio-temporal features from the input data. Therefore, according to equation 2.29, the mathematical formulation for the ConvLSTM can be given as

$$\begin{aligned}
\mathbf{f}_t &= \sigma\left(\mathbf{W}_{fh} * \mathbf{H}_{t-1} + \mathbf{W}_{fx} * \mathbf{x}_t + \mathbf{W}_{fc} \odot \mathbf{C}_{t-1} + \mathbf{b}_f\right) \\
\mathbf{i}_t &= \sigma\left(\mathbf{W}_{ih} * \mathbf{H}_{t-1} + \mathbf{W}_{ix} * \mathbf{X}_t + \mathbf{W}_{ic} \odot \mathbf{C}_{t-1} + \mathbf{b}_i\right) \\
\mathbf{C}_t &= \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_{ch} * \mathbf{H}_{t-1} + \mathbf{W}_{cx} * \mathbf{X}_t + \mathbf{b}_c\right) \\
\mathbf{o}_t &= \sigma\left(\mathbf{W}_{oh} * \mathbf{H}_{t-1} + \mathbf{W}_{ox} * \mathbf{X}_t + \mathbf{W}_{oc} \odot \mathbf{C}_t + \mathbf{b}_o\right) \\
\mathbf{H}_t &= \mathbf{O}_t \odot \tanh(\mathbf{C}_t)
\end{aligned} \tag{2.30}$$

where $*$ denotes convolution and $\mathbf{C}$, $\mathbf{H}$ and $\mathbf{X}$ are the cell output, hidden state and input data, respectively [55]. When training an RNN for a task, the LSTM or ConvLSTM cells are used in combination with other layers such as fully-connected layers to build a deep neural network such as an autoencoder.

3 ROBOTIC ULTRASOUND

Medical ultrasound imaging plays an important role in clinics for diagnosis as well as for therapy guidance. While for diagnostic purposes the clinician can hold and guide the ultrasound transducer to the area of interest, this can be obstructive or even impossible for a therapy guidance purpose. In needle insertion procedures, e.g., the ultrasound transducer as well as the needle must be guided simultaneously by the clinician which is a challenging task requiring high skill level and experience. Thus, to facilitate these procedures, robotic ultrasound systems are under research either guiding the needle [56] or both the needle and the ultrasound transducer [57], [58]. In radiation therapy, hand-held ultrasound imaging cannot even be used to guide therapy because the clinician would be exposed to harmful ionizing radiation. For this reason, a robotic ultrasound system is required when bringing ultrasound guidance into the clinical application of radiation therapy. Also for other clinical domains like cardiac [59], vascular [60], obstetric [61] or spine [62], robotic ultrasound systems are under research [63], [64]. In a review study, robotic ultrasound systems proposed in literature are classified depending on the level of autonomy[1]. For this, the level of robot autonomy (LORA) scale is used ranging from *Teleoperation* (level 1) where the robot is completely controlled by a user to *Full autonomy* (level 9) where the robot performs all tasks autonomously [65]. The study showed that no robotic ultrasound system has reached a LORA level higher than *Executive control* (level 7) yet. This is mainly due to missing robust, real-time capable and reliable ultrasound image processing methods and safety strategies [64].
This work aims to contribute to closing the gap of missing real-time tracking algorithms for 4D ultrasound. In order to do this, in this chapter

Research Question 1
How much does a target in the liver move and how precisely can it be tracked in 3D ultrasound sequences?

is addressed. The basics for a robotic 4D ultrasound system usable for performing target tracking in the liver are introduced and evaluated. The robotic ultrasound setup and the procedure for acquiring a 4D ultrasound data set are presented. Subsequently, the data set is labeled and analyzed in terms of labeling accuracy and

[1]The review study was published in [64].

motion amplitude[2].

3.1 System Setup

In contrast to other imaging modalities like X-ray or Computed Tomography (CT), ultrasound imaging has the disadvantage of a limited field of view. Thus, the ultrasound transducer position must be adequately selected and constantly monitored. In addition, the ultrasound transducer must be in continuous contact with the patient's surface during the imaging procedure to ensure high quality images [68]. Due to these limitations and the fact that it is not possible in clinical practice for the ultrasound transducer to be guided by a clinician in interventional procedures, ultrasound robots are being researched. Several research groups are investigating ultrasound robot setups, methods to calibrate the systems, and algorithms to safely control them [63], [64], [69]. For instance, Schlosser et al. [9] developed a custom robot design with nine degrees of freedom (9-DoF) for radiation therapy. In this design, the ultrasound transducer as well as the linear accelerator (LINAC) are integrated making it a space saving setup. Sen et al. [70] proposed a custom 3-DoF robot in combination with a passive arm providing two further DoF to guide an ultrasound transducer during radiation therapy treatment. However, in the study they reported difficulties caused by the robot design and, thus, decided to use a commercially available 6-DoF robot arm in following studies. In contrast to custom robot designs, most robotic ultrasound setups under research use commercially available robot arms with six or seven DoF [66], [71]–[73].
In this work, the setup illustrated in Figure 3.1 is used. The setup consists of a 7-DoF robot arm placed near the patient table, an ultrasound machine with an ultrasound transducer that is attached to the robot's end-effector and a workstation that is connected to both the ultrasound machine and the robot to stream and process the ultrasound images and control the robot arm. As presented in Figure 3.1, the whole system can be split into four transformations defined by

$$^{B}\mathcal{T}_{L} = {}^{B}\mathcal{T}_{E} \cdot {}^{E}\mathcal{T}_{US} \cdot {}^{US}\mathcal{T}_{L} \tag{3.1}$$

[2]Parts of this chapter were published in:

[64] F. von Haxthausen, S. Böttger, D. Wulff, J. Hagenah, V. García-Vázquez and S. Ipsen, "Medical Robotics for Ultrasound Imaging: Current Systems and Future Trends," Current Robotics Reports, vol. 2, no. 1, pp. 55-71, 2021.

[66] S. Ipsen, D. Wulff, I. Kuhlemann, A. Schweikard and F. Ernst, "Towards automated ultrasound imaging - robotic image acquisition in liver and prostate for long-term motion monitoring." Physics in Medicine and Biology, vol. 66, p. 094002, 2021

[67] J. Osburg, D. Wulff and F. Ernst, "Generalized Automatic Probe Alignment based on 3D Ultrasound," Current Directions in Biomedical Engineering, vol. 8, no. 1, pp. 58-61, 2022

and have been submitted to:

BSPC D. Wulff and F. Ernst, "Analysis of Intra- and Inter-Observer Variability in 4D Ultrasound Landmark Labeling," Biomedical Signal Processing and Control, 2024, under review.

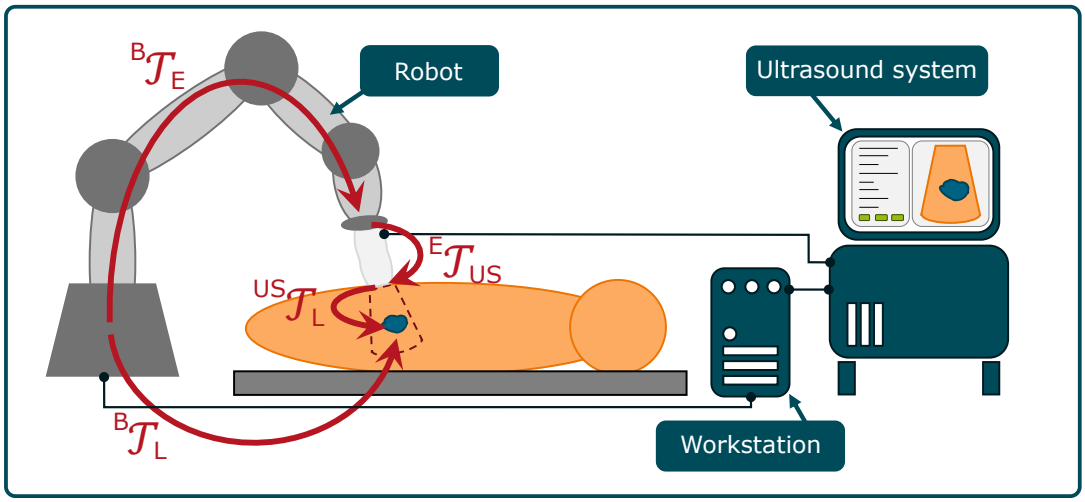

Figure 3.1: Robotic ultrasound setup consisting of a robot arm, an ultrasound system (transducer and machine) and a workstation. The transducer is connected to the ultrasound machine where the ultrasound images are visualized. The ultrasound machine as well as the robot arm are connected to the workstation processing the ultrasound images and controlling the robot accordingly. The whole setup is defined by the transformations ${}^{B}\mathcal{T}_{E}$, ${}^{E}\mathcal{T}_{US}$ and ${}^{US}\mathcal{T}_{L}$.

where B, E, US and L denote the robot basis, the robot end-effector, the ultrasound transducer and the target landmark, respectively. The transformation ${}^{B}\mathcal{T}_{E}$ denotes the robot end-effector position in the robot base coordinate system and is known from the robot itself. ${}^{E}\mathcal{T}_{US}$ is a fix but unknown transformation from the end-effector to the ultrasound transducer including the transducer holder geometry that needs to be determined by a calibration step. System calibration is required to enable mapping the position of the target landmark from the ultrasound image coordinate system into the robot coordinate system (${}^{B}\mathcal{T}_{L}^{-1}$). Therefor, hand-eye calibration algorithms can be used that are based either on using an ultrasound phantom [74] or an optical tracking system [75]. The transformation ${}^{US}\mathcal{T}_{L}$ is the unknown position of the target within the ultrasound image coordinate system and the main focus of this work. Tracking the target in time-resolved ultrasound image data as precise and fast as possible is a challenging task that will be investigated in detail in this work.
Moreover, to be able to acquire high quality ultrasound images that contain the target, further steps are required that are part of future work or are already under research. The ultrasound transducer must be placed on an optimal viewport to ensure target visibility and, at the same time, reduce the impact of the robot system to the treatment procedure [8], [76], [77]. Furthermore, the ultrasound transducer must be properly and continuously aligned with the curved human surface. For this, a neural network is developed and trained so that it predicts the rotation angles the robot has to apply in order to ensure the ultrasound transducer keeps full surface contact[3]. This is done based on the 3D ultrasound image and the force applied by the transducer on the surface provided by an *LBR iiwa 7 R800* robot arm (KUKA,

[3]The methodology and evaluation of the ultrasound probe alignment approach was published in [67].

Augsburg, Germany). In phantom experiments, a low alignment error of $5.0° \pm 2.8°$ was measured indicating that autonomous transducer alignment can be achieved in future work.
In the following sections, the 4D ultrasound imaging modality as well as the data set acquirement and annotation are presented in detail.

3.2 4D Ultrasound Imaging

Ultrasound imaging is a radiation-free imaging modality based on ultrasound waves with frequencies in the range of of 1-20 MHz [78]. An ultrasound imaging system consists of an ultrasound machine and a transducer that is connected to the machine. The ultrasound machine performs all required computations, offers a graphical user interface (GUI) that visualizes the ultrasound image and allows to change settings. The ultrasound transducer is directly placed onto the human skin in order to emit and receive ultrasound waves that are used to create the ultrasound image. To ensure a high rate of the ultrasound waves entering the human tissue, ultrasound gel is placed between the human skin and the transducer.
Ultrasound waves emitted into human tissue transmit through homogeneous tissue parts at a tissue-specific speed of sound c_{tissue}. At the boundaries of tissues with different acoustic impedances, the ultrasound waves are partially scattered and reflected back to the transducer. The tissue-specific acoustic impedance corresponds to the product between the tissue-specific density ρ_{tissue} and speed of sound c_{tissue}. The amount of ultrasound waves reflected depends on the acoustic impedance difference between the tissues. The higher the difference, the more ultrasound waves are reflected. The ultrasound waves reflected back to the transducer are used to form the ultrasound image by determining the location of the boundary from which the waves were reflected. For this, the time Δt between emitting and receiving the ultrasound waves and an average speed of sound in human tissue of $c_{avg} = 1540\,\frac{m}{s}$ are taken to determine the boundary location $s = \frac{1}{2}\Delta t \cdot c_{avg}$. By the use of this procedure, a so called brightness-mode (B-mode) ultrasound image is generated. Ultrasound wave generation and measurement is realized by using piezoelectric crystal elements placed inside the ultrasound transducer. A piezoelectric crystal deforms and, consequently, emits ultrasound waves when a voltage is applied, and vice versa, it creates a measurable voltage when arriving ultrasound waves deform the material [78], [79]. Today, many different ultrasound transducers are available such as linear, curved or phased array transducers developed for different clinical domains. These transducers consist of an array of piezoelectric crystal elements that are used in combination to create a B-mode 2D ultrasound image.
By acquiring several 2D ultrasound images over time, 2D ultrasound sequences (2D+t) are generated. This is sometimes referred to as 3D ultrasound. Furthermore, modern ultrasound stations are also capable to acquire volumetric (3D) ultrasound images and, by acquiring multiple 3D images over time, 3D ultrasound sequences, meaning 4D ultrasound data (3D+t). For the sake of clear terminology, the term "3D ultrasound" in this work refers to the acquisition of a volumetric 3D ultrasound image, and "2D+t ultrasound" refers to the acquisition of time-resolved 2D ultrasound images to form a sequence. The term "4D ultrasound" is used to describe a

sequence of volumetric 3D ultrasound images, which could also be called 3D+t. The elements of an ultrasound image containing the gray scale intensity values are called pixels (px) and voxels (vx) in 2D and 3D, respectively. By using modern matrix array ultrasound transducers, 4D ultrasound imaging frame rates of 2 Hz up to more than 20 Hz can be achieved [12], [80]. However, high frame rates often lead to decreased ultrasound image quality or low ultrasound volume sizes. Thus, the optimal ultrasound settings must be chosen according to the application. In this work, 3D ultrasound images are considered. 3D ultrasound images are advantageous as they provide volumetric information about the target region instead of only visualizing a 2D image slice. This is especially essential for measuring high dimensional motion in a 3D space. Therefore, for the therapy guidance purpose, 4D ultrasound imaging is promising [81], [82].

3.3 4D Ultrasound Data Set

In order to develop and test the target tracking approach presented in this work, a long-term 4D ultrasound data set acquired under robotic motion compensation is used[4]. For data acquisition, the robotic ultrasound setup described in section 3.1 was applied. In detail, the setup consisted of an *EPIQ7* ultrasound machine with an abdominal *X6-1* matrix array transducer (Philips Healthcare, Best, Netherlands) in combination with an *LBR iiwa 7 R800* light-weight robot arm (KUKA, Augsburg, Germany). From five different healthy volunteers (V1, V2, V3, V4, V5) long-term 3D ultrasound sequences with an average duration of 30 min were acquired from the liver as well as from the prostate. For this, the ultrasound transducer was placed on an intercostal liver viewport and a transabdominal prostate viewport. The volunteers were all male at an age of 27 to 38 years. During data acquisition, the volunteers lay on the back on a treatment table under free breathing. The ultrasound transducer attached to the robot was placed onto the volunteer using a remote game controller and a force-sensitive robot control mode. In this mode, the robot was controlled to keep the pressure force of the ultrasound transducer onto the volunteer to the target force $F_{target} = 10\,\mathrm{N}$. This force value was chosen based on other studies and preliminary tests in order to find a balance between ultrasound image quality and volunteer comfort and safety [73], [83]. If the force measured by the robot is $F_{meas} < F_{target}$, the robot increases the pressure by pushing the ultrasound transducer deeper into the volunteer until $F_{meas} = F_{target}$ was achieved and vice versa. With this procedure, the respiratory motion of the volunteer was compensated ensuring continuous transducer contact [66].
In total, ten scan sessions were performed leading to five liver sequences and five prostate sequences (L-Vi and P-Vi with $i \in \{1, 2, 3, 4, 5\}$, respectively). This was the first time that a long-term 4D ultrasound study was performed acquiring sequences at a length of about 30 min. In order to use the sequences for developing and validating a target tracking algorithm, the data set is labeled in the following.

[4]The 4D ultrasound data acquisition, analysis and labeling procedure presented here was partially published in [66]. A more extensive labeling study, the labeling tool and the motion analysis have been submitted to *Biomedical Signal Processing and Control* and is still under review.

Furthermore, an inter- and intra observer variability study as well as a motion range analysis were performed.

3.3.1 Data Labeling

Labeling medical 4D ultrasound data is a challenging task due to higher complexity of time-resolved 3D images compared to 2D images. Additionally, high expert knowledge is required to be able to interpret ultrasound images reliably which is rarely available. These aspects are some of the reasons for the lack of publicly available labeled 4D ultrasound data sets. Apart from the data set described in this study, only one other 4D ultrasound data set, the *Challenge on Liver Ultrasound Tracking* (CLUST) data set, is available [14]. The CLUST data set was published in 2015 in the MICCAI CLUST workshop[5] that focused on target tracking in liver ultrasound sequences. The data set contains 2D+t as well as 4D ultrasound data of the liver. In the 4D data set, 22 sequences acquired with three different ultrasound machines and transducers are available. However, this data set holds some drawbacks. The sequences have a limited duration of 14.4 s on average ([min, max]=[6 s, 27 s]). Furthermore, the sequences are sparsely labeled with an average annotation rate of 47 % ([min, max]=[10.1 %, 100 %]). In total, more than 50 % of all sequences have an annotation rate of less than 15 %. The amount of available labeled ultrasound sequences for training is even further reduced due to the data splitting performed by the authors. The data set is split into training, validation and test so that 8 sequences remain for training while 6 are for validation and 8 for testing. Furthermore, only the labels from the training set are publicly available.
Due to these limitations, the 4D ultrasound data acquired in section 3.3 are labeled in this study to provide a data set of an adequate size for tracking algorithm development and evaluation. In order to prepare a simple and easy-to-use labeling pipeline, a labeling tool with a GUI is implemented that aims to support an observer setting landmark labels within a 3D ultrasound sequence. The GUI, visualized in Figure 3.2, as well as the functionality of the tool are implemented in Python using Qt5[6]. The tool consists of seven main parts that are marked in Figure 3.2 and described in the following.

(1) For loading one of the five liver 3D ultrasound sequences easily, sequence shortcuts are placed in the upper part of the tool GUI. By pushing one of the sequence buttons, the first image of the liver sequence from the corresponding volunteer is loaded and visualized.

(2) The 3D ultrasound image is displayed in a Three-Plane-View showing the xy-, xz- and yz-plane of the image. As a result, the user is able to examine the image along all three spatial volume axes, supporting the user in analyzing the image and locating the anatomical landmark to be labeled. Furthermore, the user performs landmark labeling in the displayed views by clicking directly at the position the user wants to label. The labeled landmark can be adjusted by clicking at a new location.

[5] `https://clust.ethz.ch/`
[6] `https://doc.qt.io/qtforpython-5/`

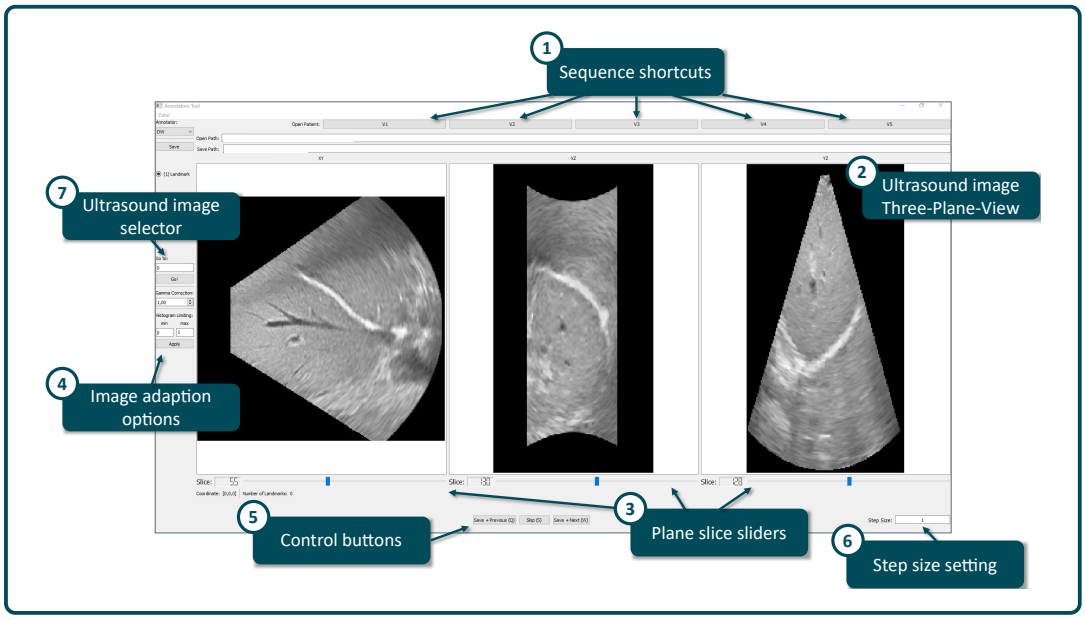

Figure 3.2: Screenshot of the labeling tool GUI used for 4D ultrasound labeling. The most important parts of the tool including image loading (1), ultrasound image visualization (2), control (3, 5, 6, 7) and image adaptation (4) are labeled.

③ To be able to adapt the slices visualized in the Three-Plane-View, for each plane, a slider is available located under the views. The current slice number is displayed next to the slider for easier orientation. Additionally, the slices can be changed using the mouse wheel.

④ Due to the fact that ultrasound image quality can be affected, e.g., by poor contrast or low brightness, two point-operation image adaptation methods are added to the labeling tool: gamma correction and histogram clipping. These can be used to adapt the ultrasound image visualization in order to increase the visibility of the anatomical landmark to be labeled. The gamma correction defined by

$$f_\gamma(\mathbf{I}, \gamma) = \mathbf{I}^\gamma \tag{3.2}$$

maps every gray scale voxel value in the image $\mathbf{I} \in \mathbb{G}$ to a new value in the range $[0, 1]$. For this, the user can set the parameter $\gamma \in \mathbb{R}$. By default, $\gamma = 1$. Histogram clipping is defined by

$$f_{clip}(\mathbf{I}, g_{min}, g_{max}) = \begin{cases} 0 & \text{if } 0 < \mathbf{I} < g_{min} \\ \dfrac{\mathbf{I} - g_{min}}{g_{max} - g_{min}} & \text{if } g_{min} \leq \mathbf{I} \leq g_{max} \\ 1 & \text{if } g_{max} < \mathbf{I} \leq 1 \end{cases} \tag{3.3}$$

where $g_{min}, g_{max} \in [0, 1]$, $g_{min} < g_{max}$ define the minimum and maximum gray scale values for clipping. By default, $g_{min} = 0$ and $g_{max} = 1$.

⑤ For labeling a complete 3D ultrasound sequence, the images are visualized one after another by the tool. The user can use the control buttons located at the

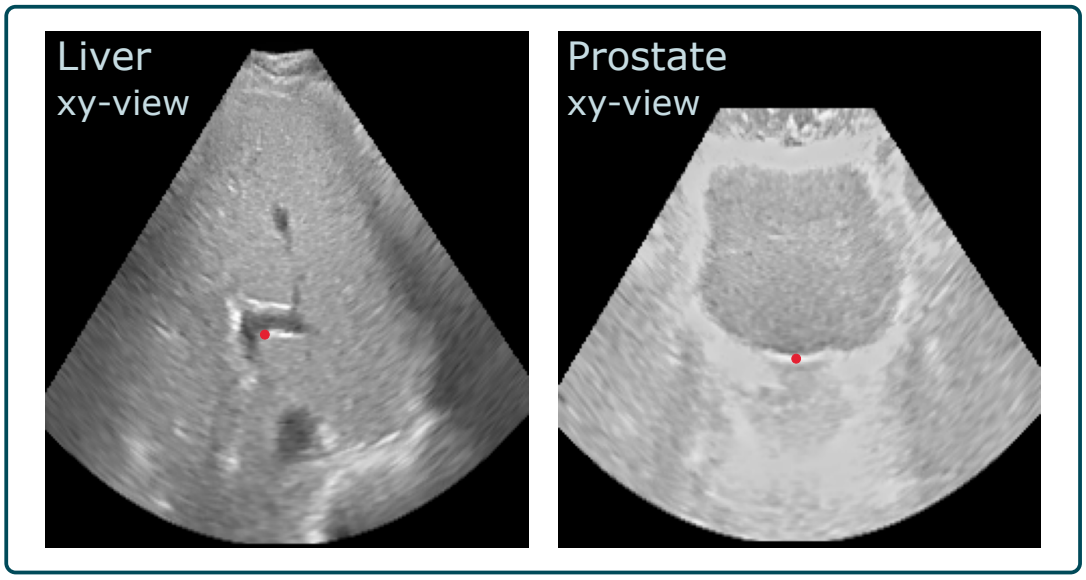

Figure 3.3: Examples for liver (left) and prostate (right) landmark labels visualized in the xy-plane of the 3D ultrasound images from the volunteer V3.

bottom of the GUI to save the label and go to the previous or next ultrasound image of the sequence. This provides an easy to use labeling procedure for time-resolved ultrasound data without the time-consuming need of opening every ultrasound image manually.

⑥ Labeling every 3D ultrasound image of a sequence is time-consuming. In order to reduce these costs, a step size parameter $\tau \in \mathbb{N}$ can be set at the bottom right of the GUI. This parameter defines the temporal step size that is applied when loading the next ultrasound image. By default, $\tau = 1$.

⑦ If the user wants to open a certain image of the ultrasound sequence, this can be done by typing the image number into the field on the left of the GUI and push the button "Go!". The label that has been set will be saved and the desired ultrasound image will be loaded and visualized.

This labeling tool was used to label the liver and prostate ultrasound sequences acquired above. A long-term as well as a short-term labeling session are performed that are described in the following.

Long-Term Labeling

In order to investigate long-term liver motion over the whole time period of the sequences, long-term labeling was performed. In every long-term sequence, an anatomical landmark was labeled by an expert observer. In the liver sequences, vessel bifurcations, and in the prostate sequences, the center of the prostate were chosen as landmarks to be labeled. Examples for these landmarks from the liver and prostate sequences from the volunteer V3 are presented in Figure 3.3. As illustrated in Figure 3.4, the sequences were sparsely labeled in 10 s intervals to reduce the time required for labeling the whole 30 min sequences. Thus, 187 and 172 labels were set

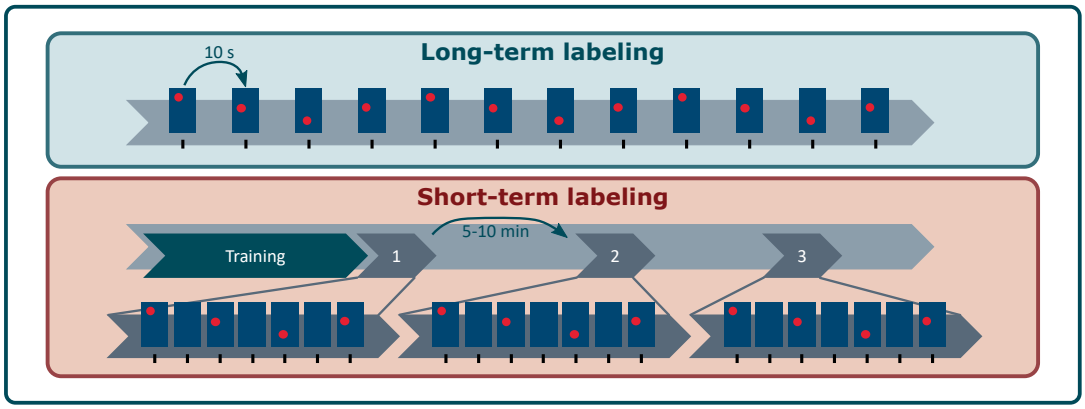

Figure 3.4: Illustration of the long-term (top) and short-term (bottom) labeling. In long-term labeling, an anatomical landmark is labeled in 10 s intervals for the whole 30 min long-term ultrasound sequence. In short-term labeling, an anatomical landmark is labeled in every second ultrasound image for 30 s short-term sequence parts (1, 2 and 3) of the long-term ultrasound sequence. The parts have a temporal distance of about 5 min to 10 min. The defined training phase lies right before the first labeled sequence part.

on average in liver and prostate sequences, respectively. Detailed information about the long-term labels is given in Table 3.1.

Short-Term Labeling

Since for target tracking short-term motion is important for both development and evaluation, short-term labeling was performed as well. During the process of this work, several short-term labeling phases were executed in order to increase the number of labels and labeled sequences. In short-term labeling, the same anatomical landmarks as in long-term labeling were used plus some additional landmarks. A visualization for all landmarks can be found in appendix section A.1. Additionally, in four long-term liver ultrasound sequences, a third sequence part was labeled. In contrast to long-term labeling, labels were set in every second ultrasound image in short-term labeling. This was done for different short sequence parts taken from the 30 min ultrasound sequences as illustrated in Figure 3.4. These sequence parts have a temporal distance of about 5 min to 10 min in order to simulate the clinical workflow of acquiring planning data in a first phase followed by the actual treatment session in a later phase. In Table 3.2, detailed information about all short-term labels is summarized. In general, one to three different anatomical landmarks were labeled in one to three sequence parts with a labeling rate of 50 % ([min, max]=[50 %, 50 %]). In total, 14 3D ultrasound liver sequences with an average duration of 36.6 s ([min, max]=[27 s, 59 s]) were labeled. In the prostate, ten 3D ultrasound sequences with an average duration of 30.8 s ([min, max]=[29 s, 34 s]) were labeled. Since this work focuses on tracking in liver ultrasound sequences, the labeling rate is higher. In order to prepare ultrasound images that can be used for training neural networks, a training phase was defined in every ultrasound sequence. As illustrated in Figure 3.4, the training phase lies right before the first labeled sequence part. This time-dependent splitting was done to avoid a bias in neural network training that would arise when using ultrasound images from the labeled sequences for training.

Table 3.1: The long-term 3D ultrasound sequences acquired under robotic motion compensation. Five sequences each were acquired at a liver and a prostate viewport under similar ultrasound machine settings. The sequences are sparsely labeled in 10 s intervals.

Organ	Sequence ID	No. frames	Frame rate (Hz)	No. landmarks	No. labeled frames	Duration (min:s)	Image size (vx)	Image resolution (mm^3)
Liver	L-V1	6,405	3.5	1	184	30:30	256×260×133	1.1×0.8×1.6
	L-V2	8,505	4.5	1	183	31:30	240×245×117	1.0×0.7×1.4
	L-V3	7,480	4.0	1	212	31:10	240×246×126	1.0×0.7×1.4
	L-V4	6,335	3.5	1	180	30:10	256×252×129	1.1×0.8×1.6
	L-V5	6,916	3.8	1	176	30:20	256×252×129	1.0×0.7×1.5
Prostate	P-V1	8,178	4.7	1	169	29:00	224×237×122	0.9×0.6×1.2
	P-V2	10,080	6.0	1	166	28:00	224×228×110	1.0×0.7×1.4
	P-V3	8,910	5.4	1	165	27:30	224×228×118	0.8×0.5×1.1
	P-V4	9,250	5.0	1	183	30:50	224×236×114	0.9×0.6×1.2
	P-V5	9,050	5.0	1	176	30:10	224×236×114	0.9×0.6×1.2

Table 3.2: The short-term sub-sequences extracted from the long-term sequences presented in Table 3.1 and labeled in every second ultrasound image. From every long-term sequence two to three short sub-sequences were extracted and labeled at one to three different anatomical landmarks.

Organ	Sequence ID	Sequence part ID	No. frames	Frame rate (Hz)	No. landmarks	No. labeled frames	Duration (s)	Image size (vx)	Image resolution (mm^3)
Liver	L-V1	L-V1-1	106	3.5	3	54	32	256×260×133	1.1×0.8×1.6
	L-V1	L-V1-2	108	3.5	3	55	31	256×260×133	1.1×0.8×1.6
	L-V1	L-V1-3	188	3.5	2	95	53	256×260×133	1.1×0.8×1.6
	L-V2	L-V2-1	136	4.5	3	69	31	240×245×117	1.0×0.7×1.4
	L-V2	L-V2-2	136	4.5	3	69	29	240×245×117	1.0×0.7×1.4
	L-V2	L-V2-3	198	4.5	2	100	45	240×245×117	1.0×0.7×1.4
	L-V3	L-V3-1	120	4.0	3	61	27	240×246×126	1.0×0.7×1.4
	L-V3	L-V3-2	120	4.0	3	61	28	240×246×126	1.0×0.7×1.4
	L-V3	L-V3-3	198	4.0	2	100	50	240×246×126	1.0×0.7×1.4
	L-V4	L-V4-1	106	3.5	2	54	31	256×252×129	1.1×0.8×1.6
	L-V4	L-V4-2	112	3.5	2	57	34	256×252×129	1.1×0.8×1.6
	L-V4	L-V4-3	198	3.5	2	100	59	256×252×129	1.1×0.8×1.6
	L-V5	L-V5-1	104	3.8	1	53	31	256×252×129	1.0×0.7×1.5
	L-V5	L-V5-2	112	3.8	1	57	31	256×252×129	1.0×0.7×1.5
Prostate	P-V1	P-V1-1	142	4.7	1	72	29	224×237×122	0.9×0.6×1.2
	P-V1	P-V1-2	140	4.7	1	71	29	224×237×122	0.9×0.6×1.2
	P-V2	P-V2-1	182	6.0	1	92	29	224×228×110	1.0×0.7×1.4
	P-V2	P-V2-2	184	6.0	1	93	33	224×228×110	1.0×0.7×1.4
	P-V3	P-V3-1	162	5.4	1	82	31	224×228×118	0.8×0.5×1.1
	P-V3	P-V3-2	162	5.4	1	82	30	224×228×118	0.8×0.5×1.1
	P-V4	P-V4-1	160	5.0	1	81	34	224×236×114	0.9×0.6×1.2
	P-V4	P-V4-2	150	5.0	1	76	30	224×236×114	0.9×0.6×1.2

Thus, ultrasound images used in the training process are taken temporally before the labeled sequences that will be used for test. The training phases of the liver sequences contain 1,803 3D ultrasound images, on average.

3.3.2 Inter- and Intra-Observer Variability Study

Expert labeling is a common method for creating ground truth data for ultrasound images. Other approaches as using implanted markers that can be tracked with an electromagnetic system like Calypso (Varian Medical Systems, Palo Alto, CA, US) or creating synthetic data, e.g., by using Generative Adversarial Networks (GANs) [84] harbor some drawbacks. Implanted markers are clearly visible in ultrasound images which can highly affect the tracking algorithms executed on the ultrasound sequences [80]. By using synthetic data, high quality ground truth data can be generated. Though, creating highly realistic synthetic ultrasound images is challenging and can be time-consuming. Furthermore, algorithm evaluation on real data is still required after testing on synthetic data in order to show real world usability of the developed algorithms. Due to these challenges, expert labeling is the gold standard for creating ground truth data for ultrasound images.
As introduced in section 3.3.1, the 2D+t and 4D ultrasound data sets created by the CLUST challenge organizers are equipped with expert labeled landmarks [14]. Another example is the data set provided by Al-Dhabany et al. [85] who published an 2D ultrasound data set containing benign, malign and normal breast 2D ultrasound images. In this data set, experts provided ground truth by segmenting tumors in the ultrasound images.
Even though expert labeling is common, this approach also brings some drawbacks. Experts in interpreting ultrasound images are rare and thus hard to get. If an expert is found for labeling, though, the expert is only human who makes mistakes and comes with minor inaccuracies. Hence, landmark labels provided by an expert might be affected by inaccuracies. Furthermore, it might be possible that another expert observer would set the labels slightly different when labeling a landmark in an ultrasound image. In order to determine the variability of landmark labeling in 4D ultrasound, an intra- and inter-observer variability study was performed in this work. Eight experts labeled a subset of the available sequence parts summarized in Table 3.2 using the labeling tool described in section 3.3.1. All experts labeled every sequence part three times in order to measure the intra-observer variability. Additionally, the observers were split into two groups ($A = \{A1, A2, A3, A4\}$, $B = \{B1, B2, B3, B4\}$) where all observers in a group had to label the same anatomical landmark to measure the inter-observer variability. Information about the sequence parts used in this observer study is summarized in Table 3.3. Additionally, the landmarks to be labeled by the observer groups A and B are illustrated in appendix section A.1.
The observers were asked to perform data labeling and fill in a survey afterwards in order to get some feedback information about the observers and their experience in labeling processes. In the following, the labeling variability as well as the feedback information from the survey are presented and evaluated.

Table 3.3: Overview of the liver sequences and landmarks that are part of the observer study.

Sequence part ID	Landmark	
	Group A	Group B
L-V1-1	L1a	L1b
L-V1-2	L1a	L1b
L-V2-1	L2a	L2b
L-V2-2	L2a	L2b
L-V3-1	L3a	L3b
L-V3-2	L3a	L3b
L-V4-1	L4a	L4b
L-V4-2	L4a	L4b

Variability Results

For measuring observer variability, different ways and metrics have been described in literature [86], [87]. Blanca et al. [88], e.g., used the intra-class correlation coefficient (ICC) for measuring the variability of labeling hemodynamic parameters in ultrasound images. However, ICC does only take 1D data into account making it difficult to use it for 3D data. De Luca et al. [14] used distance measurements between landmark labels in 3D ultrasound data for measuring observer variability in the CLUST data set. Thus, in order to achieve comparable results, the labeling error was measured using the Euclidean distance determined between two landmark labels $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$ defined by

$$d_{error}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}. \tag{3.4}$$

Quantification of the labeling accuracy was performed by measuring the intra- and inter-observer variability. Here, the intra-observer variability is a measurement for the repeatability and the inter-observer variability is a measurement for the reproducibility of labeling a landmark in 3D ultrasound images [88]. In order to get one variability measurement from a set of several observations $\mathcal{L} = \{\mathbf{l}_1, ..., \mathbf{l}_n\}$, a set of all possible pairwise distances was calculated by

$$\mathcal{D}_{error} = \{d_{error}(\mathbf{l}_i, \mathbf{l}_j) \mid i, j \in \{1, ..., n\}, i < j\}. \tag{3.5}$$

Based on this set of distances, the mean observer variability as well as the standard deviation (SD) were determined [89]. This procedure was done separately for both calculating the intra- as well as the inter-observer variability. For intra-observer variability, all observations from the same observer were used (three labels per observer) and for inter-observer variability, all observations from all observers who labeled the same landmark were used, i. e., four observers per landmark.

Intra-Observer Variability

The results of this study are summarized in Figure 3.5 (The detailed variability statistics determined in this study can be found in section A.2 in the appendix). The intra-observer variability results are presented in the top row, the

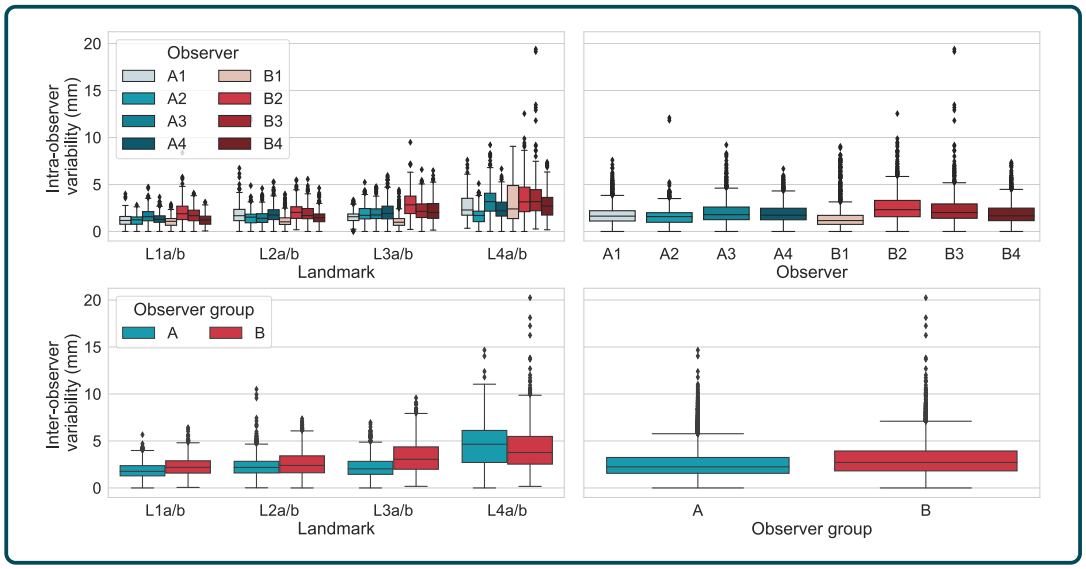

Figure 3.5: Intra-observer (top) and inter-observer (bottom) variability results measured in the labeling study. Eight observers split into two observer groups A and B labeled four landmarks each (Lia and Lib, $i \in \{1, 2, 3, 4\}$) three times.

inter-observer variability results are placed in the bottom row. As illustrated in the figure, the intra-observer variability varied depending on the observer ([min, max] = [1.58 mm ± 0.90 mm, 2.05 mm ± 1.22 mm] as well as on the landmark ([min, max] = [1.06 mm ± 0.54 mm, 3.53 mm ± 2.29 mm]). In the labels from observer $B3$, e.g., variability outliers of up to 19.40 mm were observed, while the observer $A4$ had a maximum variability value of 6.67 mm. The highest intra-variability values were measured for the landmarks L4a and L4b with mean intra-observer variability of 2.53 mm ± 1.38 mm and 3.28 mm ± 2.04 mm, respectively. In contrast to that, for the landmarks L1a and L1b mean intra-observer variability of 1.34 mm±0.68 mm and 1.52 mm ± 0.88 mm were measured, respectively. Thus, a mean intra-observer variability difference of up to 1.94 mm was measured depending on the landmark and the sequence to be labeled. During the labeling process, observers were free to choose the order in which they labeled the sequences. However, since the order was not tracked, it is not possible to examine any potential practice effects. This is a limitation of the study.

Inter-Observer Variability

Furthermore, it was observed that the average inter-observer variability with 2.68 mm± 1.69 mm in group A and 3.06 mm ± 1.74 mm in group B is higher than the intra-observer variability. This indicates that initially locating the landmark to be labeled was more challenging for the observers than repeating the landmark annotation. However, when considering the mean label position determined from the three label positions set by one observer, the mean inter-observer variability decreases significantly (p-value $\ll 0.01$), measured by performing a t-test. Mean inter-observer variabilities of 2.20 mm ± 1.49 mm and 2.51 mm ± 1.31 mm were measured in groups A and B, respectively. The inter-observer variability also depends on the landmark,

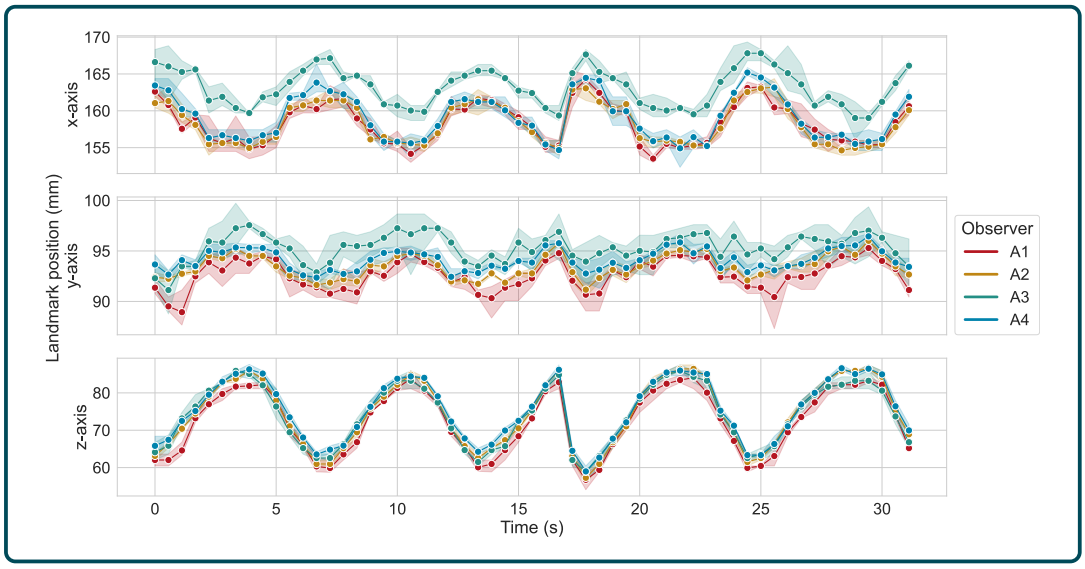

Figure 3.6: Landmark labels provided by the four observers of group A for the sequence L-V4-1 along the x-, y- and z-axes of the 3D ultrasound image. Observer $A3$ produced a label shift in the x- and y- axes in comparison to the remaining observers leading to high inter-observer variability of $4.54\,\text{mm} \pm 2.16\,\text{mm}$.

as can be seen in Figure 3.5. For instance, the mean inter-observer variability for landmark L1a is $1.85\,\text{mm} \pm 0.78\,\text{mm}$ while it is $4.54\,\text{mm} \pm 2.16\,\text{mm}$ for L4a. It could be observed that the observer $A3$ labeled the landmark 5.25 mm apart from the other observers, on average. This is caused by a labeling shift as presented in Figure 3.6. Here, it can be seen that the labels provided by observer $A3$ are shifted mainly along the x- and y-axes of the image in comparison to the labels set by the remaining observers of group A. In total, such a label shift could be observed for three out of eight landmarks, namely L2b, L3b and L4a. However, all of these label shifts were produced either by observers $A3$ or $B2$. This finding indicates that the landmark definition was not always clear enough for these observers. Vijayan et al. [90] reported the same issue in their study where four observers labeled landmarks in 4D ultrasound data. They figured out that some of the observers labeled slightly different anatomical landmarks producing an offset to the landmarks set by the remaining observers.

For evaluating the reliability of labeled ground truth in a target tracking task, the intra-observer variability is more convincing than inter-observer variability as it is a measure for how robust an observer can label a target. In contrast, in the CLUST challenge the inter-observer variability was used as a variability measure and it was even calculated in a different way [14], [91]. The authors determined inter-observer variability of $1.36\,\text{mm} \pm 1.14\,\text{mm}$, $1.27\,\text{mm} \pm 1.07\,\text{mm}$ and $1.19\,\text{mm} \pm 0.83\,\text{mm}$ for three observers. The reason for having three inter-observer values is due to the way of calculating it. Instead of determining the distances between the labels of the observers, the authors calculated the distance to the mean label position determined from the labels of the three observers [91]. Thus, for each observer an individual variability measure was calculated. When applying this calculation to the data set acquired in this work, a mean inter-observer variability range of ([min,

max] = [1.33 mm ± 0.79 mm, 2.05 mm ± 1.04 mm]) is determined which is in the same order of magnitude as in the CLUST data set. The highest inter-observer variability values were determined for the observers $A3$ (1.93 mm ± 1.49 mm) and $B2$ (2.05 mm ± 1.04 mm) which is caused by the label shifts shown before.

Observer Survey

The observers providing the landmark labels were technical experts in medical image analysis from the University of Lübeck. They reported experience in analyzing 3D ultrasound images of 1.8 years, on average. After labeling an ultrasound sequence, the observers were asked to complete a survey with the following questions, answered on a 5-point Likert scale (**Q1** to **Q6**) or multiple choice (**Q7**).

Q1: Rate the level of clarity according to the landmark definition.

- Scale from 1 (*very clear*) to 5 (*very unclear*)

Q2: Rate how much the target was clearly visible in the ultrasound sequence.

- Scale from 1 (*always visible*) to 5 (*rarely visible*)

Q3: Rate how much the level of labeling difficulty was affected by image noise.

- Scale from 1 (*not at all*) to 5 (*very much*)

Q4: Rate how much the level of labeling difficulty was affected by ultrasound artifacts.

- Scale from 1 (*not at all*) to 5 (*very much*)

Q5: Rate how much the level of labeling difficulty was affected by poor image quality.

- Scale from 1 (*not at all*) to 5 (*very much*)

Q6: Rate how much the level of labeling difficulty was affected by rotational target motion.

- Scale from 1 (*not at all*) to 5 (*very much*)

Q7: What image adaptation options did you use during the labeling procedure?

- Mark what fits: Gamma correction, histogram limiting, no adaptions used

The questions **Q1** to **Q6** had to be answered by rating on a scale from 1 to 5 while question **Q7** is a multiple choice question. Thus, every observer filled the survey once for the landmarks L1, L2, L3 and L4 each, meaning four times in total. The questions **Q1** and **Q2** focus on the target definition and visibility in the ultrasound sequences. The observers' answers are presented in Figure 3.7. It can be seen that only in one case (observer $A3$ for landmark L4a) the landmark definition was rated 5 (*very unclear*). This low level of landmark definition clarity could be a reason for

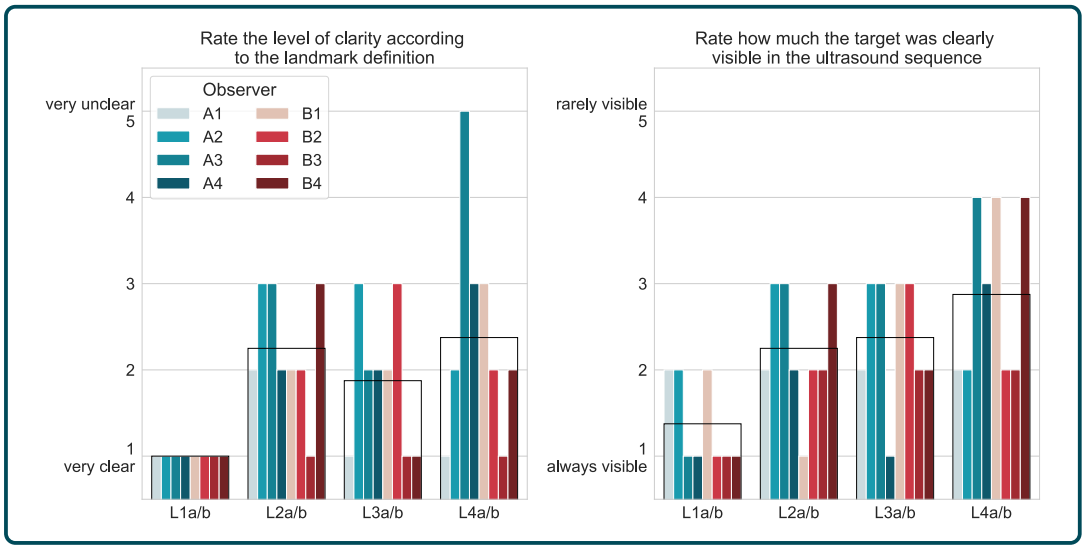

Figure 3.7: Observer ratings for the questions **Q1**: "Rate the level of clarity according to the landmark definition" and **Q2**: "Rate how much the target was clearly visible in the ultrasound sequence" on a scale from 1 to 5 for all ultrasound sequences.

the label shifts shown before. In total the landmark definition was rated 1 (*very clear*) or 2 in 75 % of the answers. This indicates that most of the landmarks to be labeled were intelligible for the observers. The visibility of the target within the ultrasound sequences (question **Q2**) was rated 1 (*always visible*) or 2 in 66 % of the cases. No observer rated a landmark to be *rarely visible*. Hence, the targets were predominantly visible to the observers. Though, it can be seen that the level of target visibility for the landmarks L4a and L4b is lower than for the other landmarks, indicating that these landmarks were harder to label for the observers. This finding could be a reason for the higher intra-observer variability values measured for the landmarks L4a and L4b. In the questions **Q3** to **Q6**, the observers had to rate how much image noise, ultrasound artifacts, image contrast and rotational target motion affected the level of labeling difficulty. The observers' answers are presented in Figures 3.8 and 3.9. It can be seen that image noise affects the labeling difficulty for all observers for all landmarks. Only two times the observers answered question **Q3** with 1 (*not at all*). In contrast, ultrasound artifacts highly affected the labeling difficulty for the landmarks L3a, L3b, L4a and L4b as the observers answered this question with an average rating of 2.4 for these landmarks. For the remaining landmarks, image artifacts only had a minor impact on the labeling difficulty with an average rating of 1.4. The observers reported a rib shadow artifact temporarily overlaying the landmarks L4a and L4b. This artifact occurred due to a rib moving into the field of view of the ultrasound transducer caused by breathing induced motion. In Figure 3.10 the landmark L4a is visualized in xy-slices of the ultrasound images from the sequence L-V4-1 acquired at 0 s (the first image of the sequence) and at 13 s. It can be seen that the landmark is close to the rib shadow artifact in both images. In the 13 s ultrasound image, the target structure is even partially obscured by the shadow artifact making accurate landmark labeling difficult. Since the landmarks were overlaid by this shadow artifact, labeling the landmarks became

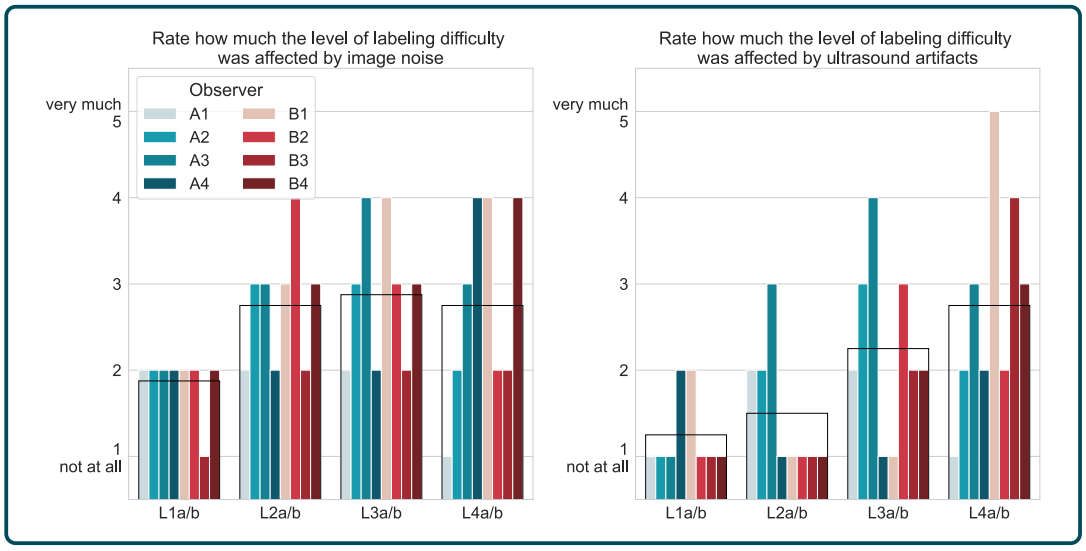

Figure 3.8: Observer ratings for the questions **Q3**: "Rate how much the level of labeling difficulty was affected by image noise" and **Q4**: "Rate how much the labeling difficulty was affected by ultrasound artifacts" on a scale from 1 to 5 for all ultrasound sequences.

very challenging due to limited visibility. This finding could be the reason for the high *rarely visible* ratings of the landmarks mentioned before and thus for the higher intra-variability values measured for these landmarks. The influence of the image quality (question **Q5**, Figure 3.9) was rated differently by the observers depending on the landmarks. A similar trend for the influence of ultrasound artifacts (question **Q4**) could be observed. However, for the influence of rotational target motion (question **Q6**) no trend was measured. The ratings range from 1 (*not at all*) to 4 and meet to 2.1 on average.
To determine the factors that most influenced the observers' landmark labels the correlation between the intra-observer variability and the survey answers was calculated for each observer. For this, the Pearson correlation, also known as the Normalized Cross Correlation (NCC), was used defined by

$$r(\mathbf{x},\mathbf{y}) = NCC(\mathbf{x},\mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\overline{y})^2}} \tag{3.6}$$

where $\overline{x}$ and $\overline{y}$ are the mean values of the data vectors $\mathbf{x}$ and $\mathbf{y}$. In total, four statistically significant correlations (p-value < 0.05) were determined. For observer $A4$, a correlation coefficient of 0.95 was measured between the landmark definition clarity (question **Q1**) and the intra-observer variability. Thus, the landmark definition clarity plays an important role in a labeling task. The intra-observer variability of the observers $B1$, $B3$ and $B4$ highly correlated with the influence of ultrasound artifacts (question **Q4**) with correlation coefficients of 0.96, 1.00 and 0.98, respectively. This confirms, that ultrasound artifacts have a major influence on the landmark labeling accuracy.
In the last question, the observers had to name what image adaptation method they used during labeling the ultrasound sequences. It turned out that they used histogram limiting the most in 56 % of the sequences. In 38 % of the sequences, they

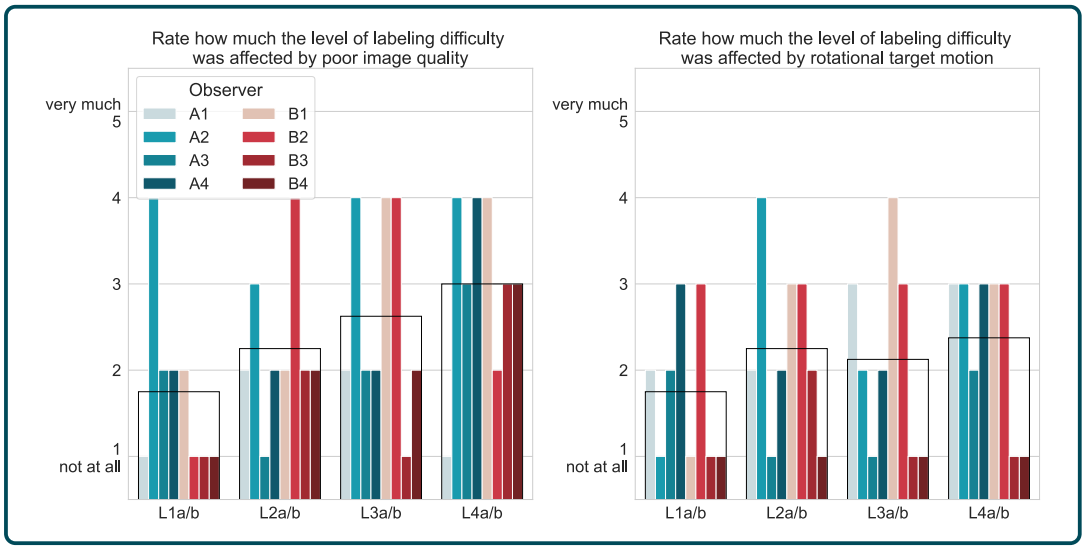

Figure 3.9: Observer ratings for the questions **Q5**: "Rate how much the level of labeling difficulty was affected by poor image contrast" and **Q6**: "Rate how much the labeling difficulty was affected by rotational target motion" on a scale from 1 to 5 for all ultrasound sequences.

did not use an image adaptation method and only in 6 % of the sequences they used gamma correction. In total, five out of eight observers used histogram limiting to adapt the ultrasound image visualization indicating that this method can be helpful when labeling ultrasound images.

3.4 Breathing Induced Motion

Abdominal organs such as the liver or kidneys are in continuous motion due to respiratory movement. This motion can consist of translation, rotation as well as deformation [92], [93]. Hence, high-precision treatment such as radiation therapy is challenging because a tumor located in the liver moves in a similar way. In literature, there are several studies that have investigated the respiration induced movement stretch in the liver. In 2006, the American Association of Physicists in Medicine (AAPM) published a report on the management of respiratory motion in radiation oncology giving an overview about breathing induced motion and how to deal with it [19]. In Table 3.4 the motion extent measured in other studies is summarized. In most studies, the liver motion was measured separately for left-right (LR), anterior-posterior (AP) and superior-inferior (SI) directions. However, in early studies, only the motion in SI direction was measured [94]–[97]. This was done due to the fact that the SI direction is the one where the highest motion occurs. In the LR- and AP-directions, no motion higher than 2 mm was expected [19]. Though, Park et al. [99] measured mean±SD displacement of 3.00±2.00 mm and 5.10±3.10 mm in LR- and AP-directions indicating that this assumption is not always true. In some studies, the total motion magnitude in 3D space was measured by determining the Euclidean norm of LR-, AP- and SI-directions. For instance, mean motion magnitude of 10.10 mm and a range of [min, max] = [2.40 mm, 19.40 mm] were measured by Park et al. [99]. It has also been shown that the liver does not just

Table 3.4: Overview of studies investigating breathing induced translation motion in the liver on the basis of n subjects. The motion extension is given along the left-right (LR), anterior-posterior (AP) and superior-inferior (SI) directions as well as in 3D space. All values are in mm in the form mean±SD (minimum-maximum).

Study	Modality	n	Motion (mm)			
			LR	AP	SI	3D
Weiss et al. [94]	Fluoroscopy	25	-	-	13.00±5.00	-
Harauz and Bronskill [95]	Scintigraphy	51	-	-	14.00	-
Suramo et al. [96]	US	50	-	-	25.00	-
			-	-	(10.00-40.00)	-
Davies et al. [97]	US	9	-	-	10.00±8.00	-
			-	-	(5.00-17.00)	-
Rohlfing et al. [93]	MRI	4	(1.00-3.00)	(1.00-12.00)	(12.00-26.00)	-
Case et al. [98]	CT	29	1.80	4.30	8.00	10.10
			(0.10-7.00)	(0.10-12.10)	(0.10-18.80)	(2.40-19.40)
Park et al. [99]	CT	20	3.00±2.00	5.10±3.10	17.90±5.10	-
Worm et al. [100]	CT	10	5.2	8.0	16.0	17.6
			(2.5-10.5)	(2.4-17.6)	(5.3-35.8)	(5.6-39.5)
Tsai et al. [92]	CT	20	0.60±3.00	2.30±2.40	5.70±3.40	-
Jupitz et al. [101]	CT	10	-	2.47±0.77	3.14±1.49	-
Sun et al. [20]	Cyberknife	12	0.56±0.29	1.24±0.69	3.38±1.25	3.98±1.28
			(0.22-1.14)	(0.62-2.65)	(1.36-5.46)	(1.61-6.18)
Own measurement	US	4	-	-	-	11.56±5.86
						(4.92-31.25)

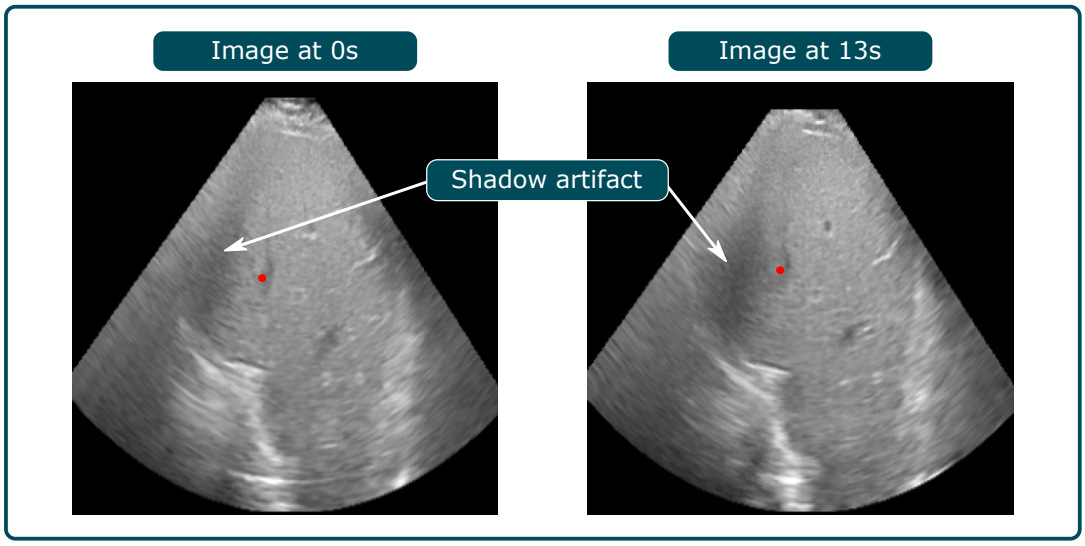

Figure 3.10: *xy*-slices of ultrasound images from the sequence L-V4-1 at 0 s (left) and 13 s (right). The landmark L4a is marked in red in both images. A rib shadow artifact is close to the landmark in the 0 s image. In the 13 s image the artifact partially obscures the landmark making accurate localization and labeling difficult.

move rigidly as a whole. Tsai et al. [92] studied breathing-induced liver motion separately for different liver segments and measured different motion extents for these segments. In addition to translation, the liver also rotates. Sun et al. [20] measured rotation extents of 0.88°±0.58°, 1.24°±0.53° and 1.12°±1.06° around LR-, AP- and SI-axes, respectively and Rohlfing et al. [93] reported a maximum total rotation of 1.5°. Even though the rotation angles are relatively low, this motion can highly affect the treatment accuracy in radiation therapy. The AAPM recommends to perform motion management in radiation therapy if the motion is > 5 mm [19]. This recommendation can also be considered as an upper bound requirement for the accuracy of a motion management system such as a tracking algorithm.
In order to quantify the short-term motion extent of the 4D ultrasound data set used in this study, the landmark labels set by observer $B1$ in the observer study (see section 3.3.2) were evaluated. This observer was chosen due to the fact that the lowest intra-observer variability is measured. As can be seen in Figure 3.6, the landmarks move in a periodic sinusoidal way due to the respiration process. Thus, for determining the motion extent, the peaks of the inhale and exhale motion phases were detected in the landmark labels. The motion extent is defined as the difference between the landmark positions of the inhale and exhale phases. The mean Euclidean difference between these landmark positions is 11.56±5.86 mm with a range of [min, max] = [4.92 mm, 31.25 mm]. It could be observed that the motion extent of landmark L1b is much larger than of the other landmarks. For L1b, a mean motion extent of 29.28±0.91 mm was measured while L4b has a mean motion extent of 12.77±2.49 mm. For the remaining landmarks, even lower motion was determined. Since the motion extent of the landmarks is different, a correlation analysis between the landmark motion and the intra-observer variability was performed in order to measure whether high landmark motion affects the labeling accuracy. However, no

correlation could be determined indicating that the landmark motion extent has no influence on labeling accuracy.
In addition to the short-term motion, the labels for the five liver long-term landmarks (Table 3.1) were used to measure long-term motion of the liver. For this, the median landmark position of the first and the last 30 s of the sequences were determined and compared by calculating the Euclidean difference between them. To this basis, a long-term liver motion range of [min, max] = [2.5 mm, 7.0 mm] was observed. This implies a drift in landmark positions over a time-period of about 30 min.
These findings about liver motion confirm the need of a target tracking method for radiation therapy guidance. Soft tissue structures in the liver, including tumors, can translate due to breathing-induced motion up to more than 30 mm in 3D space and rotate up to more than 1°. Since in radiation therapy the tumor must be irradiated as planned to ensure successful therapy outcome, treatment inaccuracy to this extent is unacceptable. For this purpose, the AAPM recommends to perform motion management if the motion range is > 5 mm [19].

3.5 Summary

In order to develop and evaluate a target tracking method for 4D ultrasound, in this chapter preliminary work was performed by investigating

Research Question 1
How much does a target in the liver move and how precisely can it be tracked in 3D ultrasound sequences?

For this purpose, the robotic ultrasound setup considered in this work was introduced and the main tasks required to make the system work were pointed out. In a preliminary phantom study investigating a method for automatic ultrasound transducer alignment, a rotational alignment error of $5.0° \pm 2.8°$ was achieved[7]. In addition, a 4D ultrasound data set acquired under robotic motion compensation using the presented robotic ultrasound setup was manually labeled by expert observers in this work. Therefor, an easy to use labeling tool with a GUI was implemented and provided to the observers. The created labeled data set highly enlarges the small amount of available labeled 4D ultrasound data in literature. Besides the CLUST data set [14], this novel data set is the only available densely labeled 4D ultrasound data set[8]. This is particularly useful for evaluating the performance of target tracking algorithms, as it increases the number of sequences to evaluate. Furthermore, for developing deep learning methods trained in a supervised way, the increased amount of labeled 4D ultrasound data is highly beneficial. In addition, as far as known, this work provides the first intra- and inter-observer variability study for landmark labeling in 4D ultrasound. It turned out that expert observers are able to label landmarks in 4D ultrasound with an intra-observer variability higher than 1 mm. This must be considered when evaluating a target tracking algorithm.

[7] The study on ultrasound transducer alignment was published in [67].
[8] The labeled 4D ultrasound data set can be made available upon request by emailing rob@uni-luebeck.de.

Tracking errors smaller than this would not be reliable because it would not be clear whether the errors were due to labeling or tracking inaccuracy. To reduce this limitation, labels that have been repeatedly set by an observer could be merged. Since the intra-observer variability is a measurement for the repeatability, the labeling inaccuracy could be overcome, e.g., by determining the mean label position. Based on the landmark labels, a liver motion analysis was conducted confirming the need of a target tracking algorithm as motion amplitudes of up to more than 30 mm were measured agreeing with motion amplitudes determined in other studies in literature. The findings presented in this chapter answer the given research question and provide the basis for developing a real-time target tracking algorithm in 4D ultrasound.

4 3D ULTRASOUND FEATURE EXTRACTION

In comparison to 2D ultrasound image data, 3D ultrasound image data contains much more information. When adding an additional dimension to an image, the number of voxels raises drastically leading to highly increased computational costs for image processing algorithms. This is especially true for intensity-based algorithms as these evaluate all image voxels [102]. Therefore, performing tasks like image registration or object tracking in 4D ultrasound is a challenging task. Such an algorithm does not only have to be real-time capable, but at the same time it must provide accurate and robust performance to be usable in the medical domain. While much research is in progress for 3D ultrasound to MRI or CT image registration, 3D ultrasound to 3D ultrasound image registration is rarely investigated [103], [104]. This could be due to limited fields of clinical applications for 3D ultrasound imaging, but also due to difficulties like high noise ratio, small field of view or high amount of possible artifacts that can strongly affect the quality of ultrasound images [12], [105], [106]. Due to these challenges and requirements such as real-time capability and deformation-awareness, this work focuses on developing a feature-based 4D ultrasound tracking algorithm. Using this approach prevents analyzing the whole 3D ultrasound image so that the real-time requirement can potentially be met. Such an algorithm is used, e.g., by Selmi et al. [107] who proposed a feature-based 2D to 3D ultrasound image registration approach with a mean target registration error (TRE) of $4.37 \pm 2.62\,\text{mm}$ but a total mean runtime of more than 10 s for registering two images. Even though this approach is not real-time capable yet, feature-based approaches are shown to be real-time capable as the computational complexity is highly reduced compared to intensity-based approaches [102], [108].
In order to develop a feature-based tracking algorithm for 4D ultrasound, this chapter investigates how to extract meaningful and robust features from 3D ultrasound images that can potentially be used in a feature-based tracking algorithm. Thus,

Research Question 2
How can meaningful and unique features be extracted from 3D ultrasound images?

is considered in this chapter. Two different approaches are presented and investi-

gated in terms of feature generation and evaluation[1].

4.1 Image Features

In the field of machine learning, various well-known algorithms for local feature extraction are available. In general, these algorithms were developed to be executed on 2D gray-scale or RGB images. Even though the algorithms differ in the way of determining and describing local features, they all follow the same scheme: Feature detection - description - matching (see section 2.2). In the detection phase, the image is analyzed and the locations of points-of-interest, so called keypoints, are determined. In the description phase, description vectors are generated to describe the keypoints. These description vectors should be as unique as possible to be able to distinguish between them. The combination of a keypoint and the description vector is called *feature*. In the matching phase, feature sets from different images are compared and corresponding features between the images are determined. This can be used, e.g., for image registration or object tracking.
There are several algorithms based on different approaches and with different complexity. The most common algorithms for feature detection and description are *Scale Invariant Feature Transform* (SIFT) [29] and *Speeded-Up Robust Features* (SURF) [30]. Both algorithms are scale as well as rotation invariant. However, in contrast to other algorithms, SIFT and SURF are computationally expensive. For instance, the *Oriented FAST and Rotated Brief* (ORB) algorithm is an alternative that is less complex but also provides rotation and scale invariance [113]. Karami et al. [114] found that ORB performs about ten times faster than SIFT and about two times faster than SURF in 2D images. In another study, Miksik and Mykolajczyk [115] evaluated the performance of different feature detectors and descriptors in 2D images. They figured out that SIFT and SURF required 448 ms and 117 ms, respectively, for describing 1,000 keypoints. To speed-up the SIFT algorithm, Acharya et al. [116] implemented the algorithm in a GPU-based way and achieved an average runtime of 55 ms for 2D images of the size $640 \cdot 480$ px. However, adding a third dimension to the image data would rapidly increase the runtime. Thus, providing real-time capable versions of SIFT and SURF for 3D ultrasound image data is very

[1] Parts of this chapter were published in:

[109] D. Wulff, I. Kuhlemann, F. Ernst, A. Schweikard, and S. Ipsen, "Robust motion tracking of deformable targets in the liver using binary feature libraries in 4D ultrasound," Current Directions in Biomedical Engineering, vol. 5, no. 1, pp. 601–604, 2019.

[110] D. Wulff and F. Ernst, "Feature description using autoencoders for fast 3D ultrasound tracking," Current Directions in Biomedical Engineering, vol. 10, no. 2, pp. 21-24, 2024.

[111] D. Wulff, J. Hagenah, S. Ipsen, and F. Ernst, "Learning local feature descriptions in 3D ultrasound," 2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE), pp. 323–330, 2020.

[112] D. Wulff, J. Hagenah, and F. Ernst, "Comparison of representation learning techniques for tracking in time resolved 3D ultrasound," International Conference on Medical Imaging with Deep Learning (MIDL), 2021.

challenging due to the fact that the computational costs even rise when considering such kind of data. Therefore, in order to perform real-time target tracking in 4D ultrasound, algorithms with lower complexity were selected in this work. In the following, 3D versions of a feature detector and two feature descriptors used in this work for performing target tracking (see section 6.2.1) in 4D ultrasound are described.

4.1.1 FAST-3D Detector

The *Features from Accelerated Segment Test* (FAST) algorithm was developed and proposed by Rosten and Drummond [117] for detecting keypoints in 2D images. The detector is searching for corners based on image intensity differences between a pixel to be examined and pixels in the local neighborhood. Due to the simplicity of the algorithm, FAST is an easy to implement and fast feature detector. Though, it is not scale invariant as it does not consider different scale spaces of the image. However, due to the fact that scale changes are not expected in ultrasound images once the ultrasound transducer is properly set, scale invariance can be waived for the targeted application. For 2D images, it has been shown that FAST performs faster than other feature detectors like Harris, Difference of Gaussians (DoG), *Smallest Univalue Segment Assimilating Nucleus* (SUSAN) and SURF making it a promising feature detector for real-time applications [115], [117]. Furthermore, compared to detectors like SURF or *Binary Robust Invariant Scalable Keypoints* (BRISK), FAST is able to find far more keypoints which increases the chance to find a sufficient amount of matches between images [118]. However, since the FAST feature detector was developed for 2D images, an extended FAST-3D algorithm is developed to be able to apply it to 3D ultrasound images in this work[2].
A 3D spherical mask with a radius r is created according to a 3D Bresenham circle. The voxel to examine lies in the center of the mask $\mathbf{c}$. For the original 2D FAST algorithm, a radius $r = 3$ is recommended so that the mask contains $n = 16$ elements [117]. However, due to higher dimensionality and due to the fact that ultrasound images have highly different characteristics compared to RGB images or photographs, the optimal radius, and thus the number of mask elements n, might be different for FAST-3D. For keypoint detection, the 3D ultrasound image $\mathbf{I} \in \mathbb{G}$ is sampled using the mask and in each step the test

$$\mathbf{s}_i^{\mathbf{c}}(\mathbf{x}_i) = \begin{cases} 1, & \text{if } \mathbf{I}(\mathbf{c}+\mathbf{x}_i) \leq \mathbf{I}(\mathbf{c}) - t & \text{(darker)} \\ 0, & \text{if } \mathbf{I}(\mathbf{c}) - t < \mathbf{I}(\mathbf{c}+\mathbf{x}_i) < \mathbf{I}(\mathbf{c}) + t & \text{(similar)} \\ -1, & \text{if } \mathbf{I}(\mathbf{c}) + t \leq \mathbf{I}(\mathbf{c}+\mathbf{x}_i) & \text{(brighter)} \end{cases} \tag{4.1}$$

is performed for all mask elements $\mathbf{x}_i$ with $i \in \{1, ..., n\}$. Here, t is a threshold defining the range of voxel intensity similarity. Thus, for the position $\mathbf{c}$ in the ultrasound image, a state vector $\mathbf{s}^{\mathbf{c}} \in \{1, 0, -1\}^n$ is created containing the states 1 (darker), 0 (similar) and -1 (brighter) for each element in the mask. A voxel is considered as a keypoint if the ratio of darker or brighter voxels determined by

$$v^{\mathbf{c}} = \frac{|\{x \in \mathbf{s}^{\mathbf{c}} \mid x \neq s\}|}{n} \tag{4.2}$$

[2]The FAST-3D feature detector presented here was published in [110]. A precursor was published in [109].

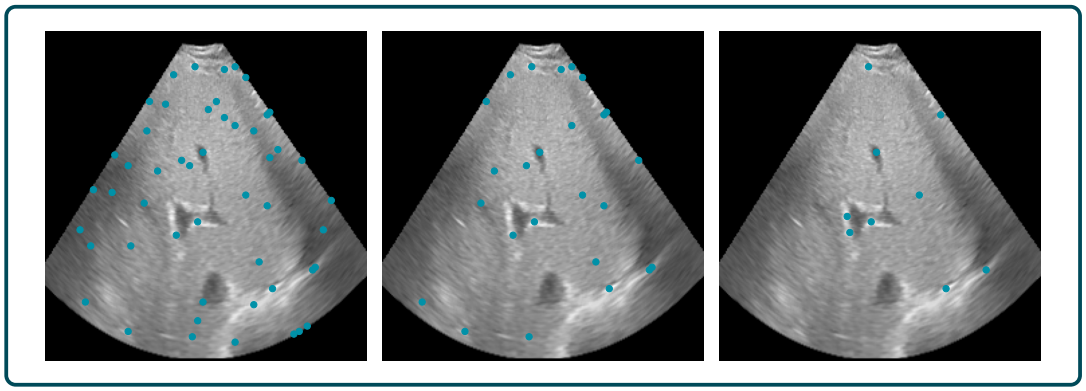

Figure 4.1: Examples for slices of a 3D ultrasound image with keypoints detected by FAST-3D at different parameter settings. Left: $(t, p, r) = (0.05, 0.5, 3)$, middle: $(t, p, r) = (0.05, 0.8, 3)$, right: $(t, p, r) = (0.1, 0.8, 3)$.

in $\mathbf{s^c}$ is greater than a threshold p. After detecting the keypoints, a non-maximum suppression is performed. This reduces the number of found keypoints. Keypoints located close to each other and probably indicating the same feature are reduced to the most prominent keypoint. Therefore, a state map $\mathbf{S}$ in the same size as the ultrasound image is created containing the values $v^{\mathbf{c}}$ at the positions where a keypoint is found and 0 else. For the whole state map $\mathbf{S}$, in a $5 \cdot 5 \cdot 5\,\mathrm{vx}$ neighborhood only the keypoint with the highest value $v^{\mathbf{c}}$ is kept. Keypoints with lower $v^{\mathbf{c}}$ are discarded.

Feature Detection in 3D Ultrasound Images

For applying FAST-3D to a 3D ultrasound image, the thresholds t and p as well as the mask radius r must be set manually. Finding the optimal parameter settings is challenging. Thus, in section 6.2.1 parameter optimization is performed in order to optimize the tracking performance. In Figure 4.1 examples for different parameter settings are visualized. It can be seen that adapting the thresholds t and p highly affects the amount of detected keypoints. In the examples, 2109 (left), 1108 (middle) and 308 (right) keypoints were detected in the same 3D ultrasound image using the parameter settings $(t, p, r) = (0.05, 0.5, 3)$, $(t, p, r) = (0.05, 0.8, 3)$ and $(t, p, r) = (0.1, 0.8, 3)$, respectively. For using the keypoints in a feature-based tracking algorithm, the goal of FAST-3D is not finding as many keypoints as possible, but rather finding meaningful and robust keypoints that can be relocated in consecutive 3D ultrasound images in a sequence.

4.1.2 BRISK-3D Descriptor

The binary feature descriptor *Binary Robust Invariant Scalable Keypoints* (BRISK) was proposed by Leutenegger et al. [119] offering an efficient algorithm for detecting and describing image features. In a performance study, including feature detection, description and matching, they found BRISK performed faster than SIFT and SURF. However, since FAST is able to perform feature detection faster than BRISK [115], and Patel et al.[118] determined that FAST is able to find about 20

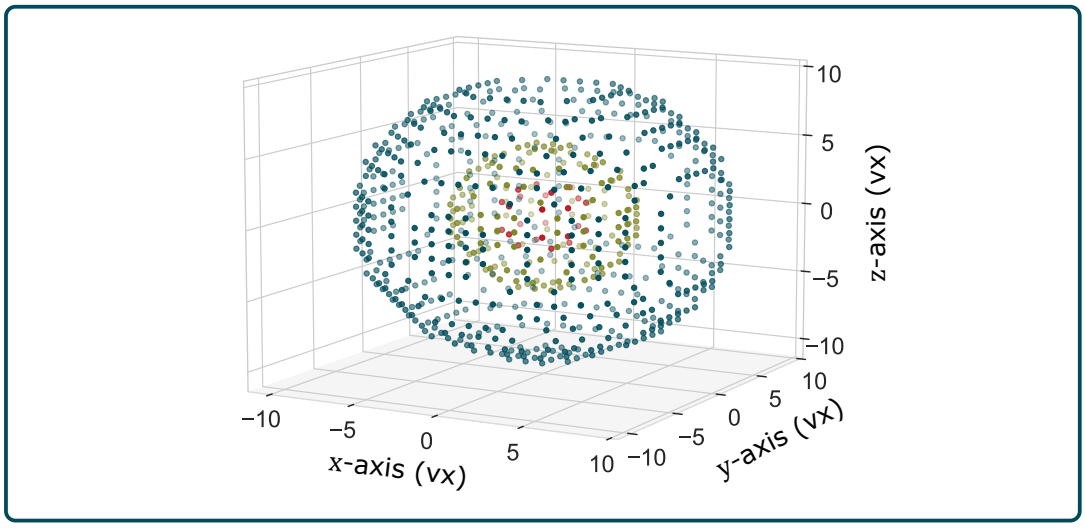

Figure 4.2: Illustration of the BRISK-3D mask $\mathcal{P}$ used for creating short- and long-pairs in a 3D scatter plot. The points in the plot denote voxels that are considered in the neighborhood of the keypoint. The mask consists of three nested spheres with radii of 2.5 vx (red), 5 vx (green) and 10 vx (blue). The keypoint is located in the center of the spheres.

times as many keypoints as BRISK, only the descriptor of BRISK is used in the following. However, since BRISK is an algorithm for 2D images as well, it is extended to BRISK-3D in this work[3]. For describing a keypoint, BRISK uses a mask $\mathcal{P}$ defining the intensity values from the local neighborhood of the keypoint to consider. The mask $\mathcal{P}$, visualized in Figure 4.2 consists of points $\mathbf{p}_i \in \mathcal{P}$ with $i \in \{1, ..., n\}$ arranged on three nested spheres centered on the keypoint. The spheres have radii of 2.5 vx, 5.0 vx and 10.0 vx which were determined empirically. This mask slightly differs from the 2D BRISK mask as it consists of four nested circles with different radii [119].
Generating the description vector, however, is performed in the same way as in 2D BRISK. This is done based on point-pairs $(\mathbf{p}_i, \mathbf{p}_j)$. Starting from the set

$$\mathcal{A} = \{(\mathbf{p}_i, \mathbf{p}_j) \in \mathcal{P} \mid i < n \wedge j < i \wedge i, j \in \mathbb{N}\} \tag{4.3}$$

defining all point-pairs in the BRISK-3D mask, two subsets are defined. The short-pairs

$$\mathcal{S} = \{(\mathbf{p}_i, \mathbf{p}_j) \in \mathcal{A} \mid \|\mathbf{p}_j - \mathbf{p}_i\| < \delta_{max}\} \tag{4.4}$$

contain an amount of s point-pairs where the distance between the points is less than the threshold δ_{max} and the long-pairs

$$\mathcal{L} = \{(\mathbf{p}_i, \mathbf{p}_j) \in \mathcal{A} \mid \|\mathbf{p}_j - \mathbf{p}_i\| > \delta_{min}\} \tag{4.5}$$

contain l point-pairs with a distance to each other greater than the threshold δ_{min}. While the short-pairs are used to build the description vector, the long-pairs are used to determine a general orientation of a keypoint based on the voxel intensities

[3]The BRISK-3D feature descriptor presented here was published in [109].

at the point positions to achieve rotation invariance. The mean keypoint orientation is determined by

$$\mathbf{g} = \begin{pmatrix} g_x \\ g_y \\ g_z \end{pmatrix} = \frac{1}{l} \cdot \sum_{(\mathbf{p}_i, \mathbf{p}_j) \in \mathcal{L}} g(\mathbf{p}_j, \mathbf{p}_i), \tag{4.6}$$

where l is the number of long-pairs and $g(\mathbf{p}_j, \mathbf{p}_i)$ is the local gradient of a long-pair computed by

$$g(\mathbf{p}_j, \mathbf{p}_i) = (\mathbf{p}_j - \mathbf{p}_i) \cdot \frac{\mathbf{I}(\mathbf{p}_j, \sigma) - \mathbf{I}(\mathbf{p}_i, \sigma)}{\|\mathbf{p}_j - \mathbf{p}_i\|^2}. \tag{4.7}$$

Here, $\mathbf{I}(\mathbf{p}_j, \sigma)$ is the gauss-filtered voxel intensity from the point $\mathbf{p}_j$. Originally, the value for σ, defining the gaussian kernel for image filtering, depends on the distance to the keypoint. However, due to the low image quality and high noise ratio of ultrasound images, and to prevent too strong filtering of the ultrasound image, $\sigma = 1.0$ is set fixed in BRISK-3D. To get a rotation invariant description vector, the determined orientation is applied to the short-pairs. For this, the angles

$$\begin{aligned} \alpha_1 &= \arctan2(g_x, g_y) \\ \alpha_2 &= \arctan2(g_x, g_z) \end{aligned} \tag{4.8}$$

are determined and the short-pairs are rotated accordingly in the corresponding planes. The description vector is built by applying

$$b = \begin{cases} 1, & \mathbf{I}(\mathbf{p}_j^\alpha, \sigma_j) > \mathbf{I}(\mathbf{p}_i^\alpha, \sigma_i) \\ 0, & \text{otherwise} \end{cases} \quad \forall (\mathbf{p}_i^\alpha, \mathbf{p}_j^\alpha) \in \mathcal{S} \tag{4.9}$$

to all rotated short-pairs $(\mathbf{p}_j^\alpha, \mathbf{p}_i^\alpha)$. With this, a description vector $\mathbf{d} \in \{0,1\}^s$ is set up, where s denotes the number of short-pairs and the length of the description vector. When applying BRISK-3D to a 3D ultrasound image, the parameters δ_{max} and δ_{min} need to be set manually.

4.1.3 BRIEF-3D Descriptor

Another binary feature descriptor is *Binary Robust Independent Elementary Features* (BRIEF) proposed by Calonder et al. [120]. Due to its simple principle, it is easy to implement and one of the fastest feature descriptors in literature. For 2D images, it has been shown that BRIEF outperforms well known descriptors like SURF, SIFT and BRISK in terms of runtime making it a promising approach for real-time applications in 3D ultrasound images [115]. This is mainly due to the fact that BRIEF does only perform intensity comparisons. In contrast, BRISK determines orientations as well leading to higher computational complexity than BRIEF. Due to this fact, BRISK is rotation invariant in contrast to BRIEF. Though, a *rotation-aware BRIEF* (rBRIEF) was proposed by Rublee et al. [113], e.g., used in the ORB algorithm.
Again, an extended BRIEF-3D needs to be implemented due to the fact that BRIEF is developed for 2D images[4]. Similar to BRISK-3D, BRIEF-3D creates a description

[4]The BRIEF-3D feature descriptor presented here was published in [110].

vector based on intensity comparisons. However, BRIEF-3D does not use a certain predefined mask for setting up point-pairs, but the point-pairs are created randomly. Thus, the point positions are not limited to a certain pattern but can lie anywhere in a defined region

$$\mathcal{P} = \{\mathbf{z} \in \mathbb{Z}^3 \mid z_i \leq r \wedge z_i \geq -r, \ \forall\, i \in \{1,2,3\}\}. \tag{4.10}$$

Therefore, a radius r is defined to create a spherical region where points $\mathbf{x}, \mathbf{y} \in \mathcal{P}$ with $\mathbf{x} \neq \mathbf{y}$ can randomly be sampled from. Based on the voxel intensities of the ultrasound image $\mathbf{I} \in \mathbb{G}$ and the keypoint position $\mathbf{c}$, the description vector is built by evaluating the test

$$\tau(\mathbf{c}, \mathbf{x}, \mathbf{y}) = \begin{cases} 1, & \text{if } \mathbf{I}(\mathbf{c}+\mathbf{x}) < \mathbf{I}(\mathbf{c}+\mathbf{y}) \\ 0, & \text{otherwise.} \end{cases} \tag{4.11}$$

The test τ is performed for all point-pairs sampled from the region. Thus, the length of the description vector l corresponds to the number of point-pairs generated. When performing BRIEF-3D, the parameters r and l must be set manually.

4.1.4 Feature Matching

In the feature matching process the goal is to find correspondences between feature sets of different images. For instance, in a tracking application, the images are two consecutive images of a time-resolved sequence. For this, the description vectors of the features are used. Depending on the kind of the description vector, a distance metric must be selected. For the binary descriptors BRISK-3D and BRIEF-3D, the Hamming distance is used. For two description vectors $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ the Hamming distance is defined by

$$d_{\text{hamming}}(\mathbf{x}, \mathbf{y}) = |\{i \in \{1, ..., n\} \mid x_i \neq y_i\}|. \tag{4.12}$$

Basically, this is the number of elements that differ between the description vectors. For non-binary description vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, e.g., the Euclidean distance

$$d_{\text{euclidean}}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4.13}$$

or the cosine distance

$$d_{\text{cosine}}(\mathbf{x}, \mathbf{y}) = \frac{\langle \mathbf{x}, \mathbf{y} \rangle}{\|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_2} \tag{4.14}$$

can be used. When matching two images, it is possible that for some features no correspondences are available in the other image. For consecutive 3D ultrasound images, this can be due to high dimensional motion patterns including rotation and deformation caused by internal procedures like breathing. Especially deformation can highly affect the appearance of anatomical structures in the ultrasound image so that features can get lost over time. Thus, the matching algorithm used in this work consists of three steps presented in Algorithm 1 and described in the following[5].

Algorithm 1 Matching algorithm to find corresponding image features in two 3D ultrasound images.

Input: $\mathbf{z}^1, \mathbf{z}^2, \delta$
Output: pairIDs

distmap = $\emptyset$ ▷ Initialize empty distance map
for j=1 to **len**($\mathbf{z}^1$) **do**
  for i=1 to **len**($\mathbf{z}^2$) **do**
    distmap $[j, i] = d\left(\mathbf{z}^1\left[j\right], \mathbf{z}^2\left[i\right]\right)$
  end for
end for
pairIDs = $\emptyset$ ▷ Initialize empty pairs-IDs array
for j=1 to **len**($\mathbf{z}^1$) **do**
  minIDy = arg min (distmap $[j, :]$)
  minIDx = arg min (distmap $[:,$ minIDy$]$)
  if j == minIDx && distmap $[j,$ minIDx$] \leq \delta$ **then**
    pairIDs.append ($[j,$ minIDx$]$)
  end if
end for

First, all distances between the feature descriptor sets $\mathbf{z}^1$, $\mathbf{z}^2$ of the two considered ultrasound images $\mathbf{I}^1 \in \mathbb{G}$ and $\mathbf{I}^2 \in \mathbb{G}$ are determined. Second, feature matches $(\mathbf{z}_j^1, \mathbf{z}_i^2)$ are determined by finding those features where $\min_{\mathbf{z}^1} d(\mathbf{z}^1, \mathbf{z}_i^2) = \min_{\mathbf{z}^2} d(\mathbf{z}_j^1, \mathbf{z}^2)$, meaning the minimum distances correspond. However, to reduce the number of wrong matches, third, a threshold $\delta \in [0, 1]$ is applied so that matches are neglected if $d(\mathbf{z}_j^1, \mathbf{z}_i^2) > n \cdot \delta$, where n denotes the length of the feature description vector. This threshold is a parameter that needs to be set manually when applying the algorithm.
In Figure 4.3, a slice of a matching example between two consecutive 3D ultrasound frames is visualized. The features were detected by FAST-3D ($(t, p, r) = (0.05, 0.8, 3)$) and described by BRIEF-3D ($(r, l) = (5, 256)$). In the matching algorithm, the threshold is set to $\delta = 0.4$. In this example, 493 features were matched correctly between the two 3D ultrasound images. In Figure 4.3, these features are marked green, if the match is located in the same displayed slice, and blue, if the matched feature is located in another slice of the 3D ultrasound image. However, the feature match marked in red indicates that mismatches can occur as this match is obviously incorrect. For performing target tracking in a 3D ultrasound sequence, the procedure consisting of feature detection, description and matching needs to be executed for every 3D ultrasound image. The motion between the previous and the current ultrasound image can be determined by considering the feature matches that have been found.

[5]The feature matching algorithm presented here was published in [110].

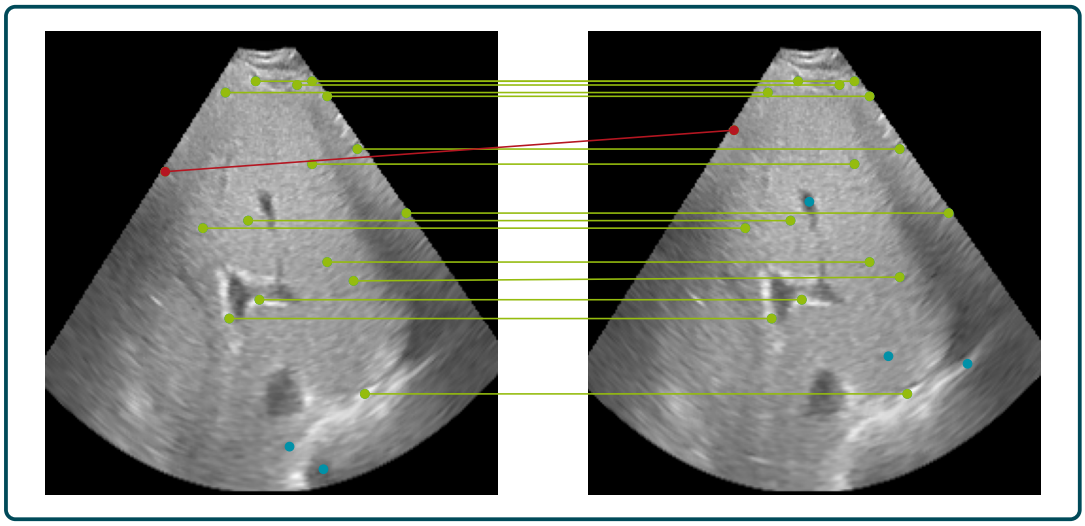

Figure 4.3: Slices of two consecutive 3D ultrasound images from the liver sequence V3 with detected, described and matched features. Feature detection is done by FAST-3D ($(t, p, r) = (0.05, 0.8, 3)$) and description by BRIEF-3D ($(r, l) = (5, 256)$). For feature matching, the threshold is set to $\delta = 0.4$. Features with found matches in this slice and in another slice are marked in green and blue, respectively. An incorrect match is marked in red.

4.2 Deep Representation Learning

Another approach for describing keypoints is deep representation learning. In contrast to the feature descriptors described above, in deep learning, the description vector is not created by following a certain algorithm. Instead, a mapping function is learned that maps an input data sample into a latent representation vector. The mapping function is learned by a deep neural network such as an autoencoder. Deep representation learning is used in various fields for different applications in the medical domain. Tabatabaei et al. [121], e.g., used an autoencoder for extracting features from 2D histological images for performing cancer detection. Anomaly detection in ultrasound data by using a variational autoencoder was proposed by Milkovic et al. [122] and Shvetsova et al. [123] performed anomaly detection in chest X-ray images using autoencoders. Even for processing higher dimensional data, like volumetric or time-resolved data, deep representation learning is used. Chen et al. [34] proposed using deep representation learning for extracting features from CT image data and Jiao et al. [124] proposed to extract features from 4D ultrasound data. The fact that deep representation learning can be performed in an unsupervised way is beneficial as no labels are required. In the medical domain, getting a large labeled data set is challenging but required for training a deep neural network in a supervised way. Labeling medical data requires high expertise which is often rare or even inaccessible. Furthermore, labeling high dimensional data like 4D ultrasound is complicated and time-consuming making it even more difficult to get a sufficient amount of labeled data. For this reason, deep representation learning is a promising approach for performing tasks in high dimensional medical data. For instance, Moriya et al. [125] proposed to use deep representation learning and clustering in representation space to perform unsupervised segmentation of 3D medical images.

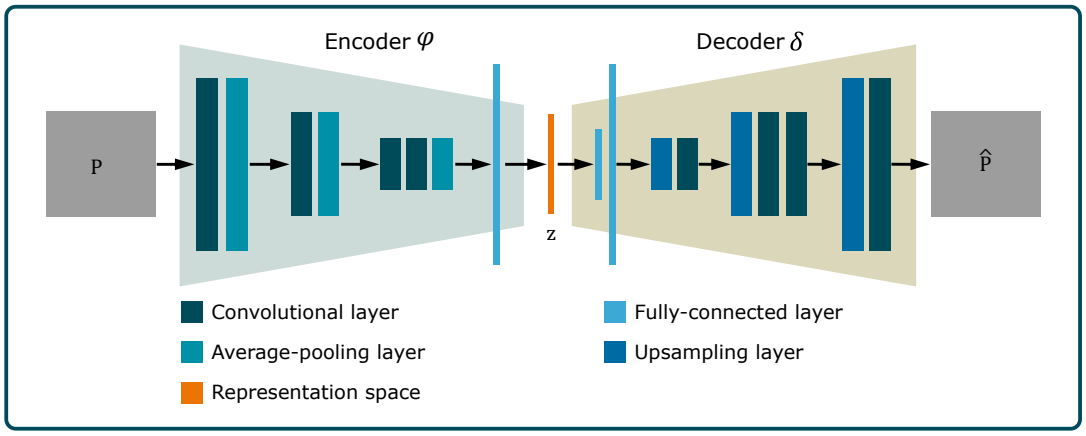

Figure 4.4: The architecture of the autoencoder used for feature extraction from 3D ultrasound patches $\mathbf{P} \in \mathbb{G}^{30\times30\times30}$. The ultrasound patches are mapped into a representation space $z \in \mathbb{R}^{50}$ by the encoder and reconstructed by the decoder.

However, using deep representation learning in time-resolved 3D ultrasound images has not been investigated yet. This work focuses on doing so in order to perform robust and real-time target tracking. Therefor, a patch-based approach is used. 3D ultrasound patches are cropped from 3D ultrasound images at certain locations in order to extract local image features. The location where to crop a patch can be defined manually, e.g., by a physician marking the tumor to track or detected automatically by a feature detector such as FAST-3D. In the following, autoencoders are presented that are used to map 3D ultrasound patches into a latent representation space. The performances of the autoencoders are analyzed and compared to each other.

4.2.1 Autoencoders for Feature Learning in 3D Ultrasound

As introduced in chapter 2.4, an autoencoder is an unsupervised representation learning technique. In order to train an autoencoder creating a latent representation space for 3D ultrasound patches, a 3D convolutional autoencoder is presented in the following[6]. The autoencoder was trained on the basis of patches of the size $30 \cdot 30 \cdot 30\,\text{vx}$ cropped from the liver ultrasound sequences L-Vi , $i \in \{1, 2, 3, 4, 5\}$. Details about data preparation for autoencoder training are presented in the following section. In order to evaluate the reconstruction performance of the autoencoder, different experiments were conducted that are explained and evaluated after the autoencoder is introduced.

Autoencoder Training

The architecture of the autoencoder used here is visualized in Figure 4.4. It consists of an encoder $\varphi : \mathbb{G}^{30\times30\times30} \mapsto [-1, 1]^{50}$ that maps a 3D ultrasound patch $\mathbf{P} \in \mathbb{G}^{30\times30\times30}$ into a latent representation vector $\mathbf{z} \in [-1, 1]^{50}$ and a decoder $\delta : [-1, 1]^{50} \mapsto \mathbb{G}^{30\times30\times30}$ that aims to generate a reconstruction of the 3D ultrasound

[6]The autoencoder presented here was published in [111].

patch $\hat{\mathbf{P}} \in \mathbb{G}^{30\times30\times30}$ so that $\hat{\mathbf{P}} \stackrel{!}{=} \mathbf{P}$. The autoencoder consists of convolutional and fully-connected layers as well as average-pooling in the encoder and upsampling layers in the decoder. In the layers, the PReLU activation function is used. However, in the last layer of the encoder the tanh activation function is used to limit the representation space to the range $[-1, 1]$. In the last layer of the decoder, the sigmoid activation is used to limit the range of the reconstructed 3D ultrasound patch to $[0, 1]$. In the training process, a custom loss function was used that aims to force the autoencoder focusing on dark appearing areas in the 3D ultrasound patch. In ultrasound images, anatomical structures like vessels appear darker than surrounding homogeneous tissue. Since such anatomical structures are highly relevant for motion tracking, the loss function

$$L_{\mathrm{LUS}}(\mathbf{P}, \hat{\mathbf{P}}) = \frac{1}{n}\sum_{i=1}^{N} \frac{1}{1 + e^{\alpha(P_i - \hat{P}_i)+\beta}} + e^{(P_i-\hat{P}_i)-1} \tag{4.15}$$

is used. Here, N is the number of voxels in the 3D ultrasound patches $\mathbf{P}$ and $\hat{\mathbf{P}}$, and i is the voxel index. Thus, the loss function L_{LUS} aims to make the autoencoder focus on anatomical structures like vessels. The values for the parameters $\alpha = 8$ and $\beta = 3$ were determined empirically. Training the autoencoder was performed for 100 epochs using the Adam optimizer and a batch size of 32.

Data Preparation

For evaluating the performance of the autoencoder, a study on quantifying the reconstruction accuracy was performed. The autoencoder was tested based on 3D ultrasound patches from five different subjects and two different organs (liver and prostate) using the ultrasound sequences L-Vi and P-Vi with $i \in \{1, 2, 3, 4, 5\}$ described in section 3.3. Different experiments were conducted in order to identify whether the autoencoder is capable of generalizing for different subjects and organs. For that, the training segments defined in the long-term 3D ultrasound sequences (see Figure 3.4) were used for training and the remaining parts are used for testing. 40,000 3D ultrasound patches of the size $30 \cdot 30 \cdot 30$ vx were cropped randomly from the training segments from all liver and prostate sequences to generate a training data set. To generate a test data set, 3D ultrasound patches were cropped randomly from each 25th ultrasound image from the test data. Thus, on average, 200 and 240 3D ultrasound patches were cropped from the liver and prostate sequences, respectively. Altogether, ten training as well as ten test data sets were generated based on 3D ultrasound sequences from five different subjects and two organs.

Reconstruction Experiments

Based on this data sets, four different types of autoencoders were trained, summarized in Table 4.1. The autoencoders AE1 were trained in a subject and organ specific way, meaning data only from one subject and one organ is used in the training process. For training the autoencoders AE2, training data from four out of five subjects, but only one organ was used. Thus, subject general and organ specific autoencoders were trained. The autoencoders AE3a and AE3b were based on the

Table 4.1: Overview of the autoencoders trained in different ways for the experiments.

Autoencoder type	Description		Amount of autoencoders
	Subject	Organ	
AE1	specific	specific	10
AE2	general	specific	10
AE3a	tuned	specific	10
AE3b	tuned	cross	10

Table 4.2: Overview of the experiments conducted to evaluate the 3D ultrasound patch reconstruction accuracy of the autoencoder.

Experiment ID	Description		Autoencoder type	Amount of experiments
	Subject	Organ		
E1	specific	specific	AE1	10
E2a	cross	specific	AE1	40
E2b	specific	cross	AE1	10
E3a	general	specific	AE2	10
E3b	general	cross	AE2	10
E4a	general	specific	AE3a	10
E4b	general	cross	AE3b	10

autoencoders AE2, but these were fine-tuned. For the autoencoders AE3a, training data from the remaining subject and the same organ was used. For AE3b, training data from the remaining subject and the other organ was used. Fine-tuning was performed for 25 epochs using the same hyperparameters as described above.
On the basis of these autoencoders, seven experiments were conducted summarized in Table 4.2. In the experiments, the autoencoders were applied to the 3D ultrasound patch test data so that the test data is encoded into the latent representation space and subsequently decoded in order to reconstruct the 3D ultrasound patches. The first experiment E1 is a subject and organ specific test. The autoencoders AE1 were applied to the test sets of the subject and the organ on which they were trained. Experiment E2 is divided into two parts, depending on whether the organ of training and test sets are identical. In Experiment E2a, a cross subject test was performed for the subject and organ-specific autoencoders AE1, using different subjects but the same organ they were trained on. Similarly, Experiment E2b performed a cross-organ test for AE1. Experiment E3 was also divided into E3a, where AE2 type autoencoders were applied to the test sets with the same organ of the subject that was excluded from training, and E3b, where AE2 type autoencoders were applied to the test sets with the other organ. Experiments E4a and E4b used autoencoders of type AE3a and AE3b, respectively. The autoencoders were applied to the test sets of the subjects with which they were fine-tuned. Experiment E4a is an organ-specific test and E4b is a cross-organ test. In general, experiments E2a, E3a, and E4a are organ-specific tests and experiments E2b, E3b, and E4b are cross-organ tests.
Quantifying the reconstruction accuracy was performed by comparing the original

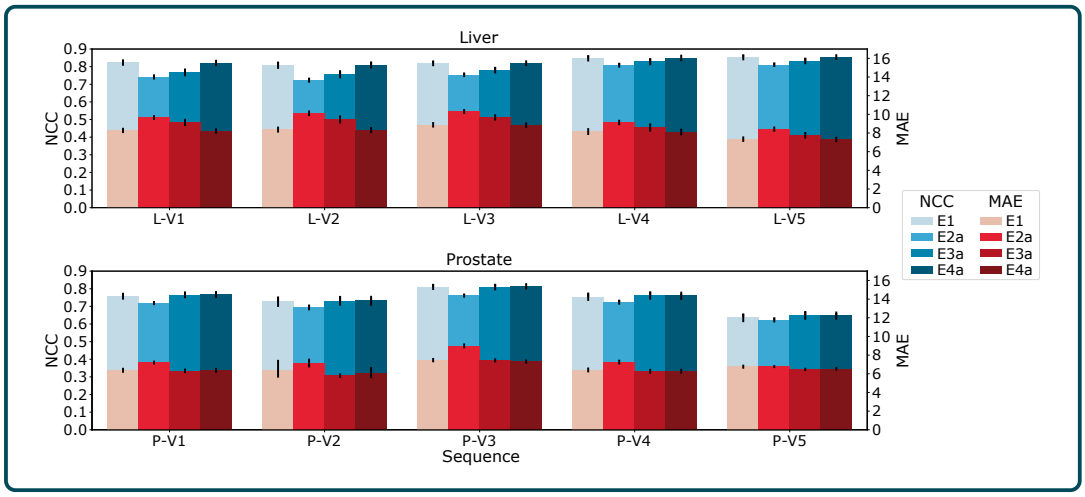

Figure 4.5: Reconstruction accuracy measured by MAE (red) and NCC (blue) observed in the organ specific experiments separated into the liver and prostate sequences.

3D ultrasound patch $\mathbf{P}$ and the reconstructed 3D ultrasound patch $\hat{\mathbf{P}}$ using the mean absolute error (MAE) defined by

$$\mathrm{MAE}(\mathbf{P}, \hat{\mathbf{P}}) = \frac{1}{N} \sum_{i=1}^{N} |P_i - \hat{P}_i|. \tag{4.16}$$

Here, $n = 27,000$ denotes the number of voxels in a 3D ultrasound patch. The MAE quantifies the absolute voxel intensity differences, but since it is not necessarily able to measure if an input patch and a decoded patch correlate, the normalized cross correlation $\mathrm{NCC}(\mathbf{P}, \hat{\mathbf{P}})$ defined in equation 3.6 is used as well. NCC is contrast invariant, although, it quantifies whether the voxel intensities in the input and output patches have a similar trend. In medical applications, the anatomical structures within an image are important. NCC can be used to verify that the autoencoder is able to correctly encode these important structures in latent representation space and reconstruct them, regardless of their voxel intensities.

Reconstruction Accuracy Results

The results of the experiments are visualized in Figures 4.5 and 4.6. The 3D ultrasound patch similarities are presented for all five subjects $i \in \{1, 2, 3, 4, 5\}$ and for the organ specific (Figure 4.5) as well as the cross organ (Figure 4.6) experiments separately. Furthermore, 2D slice examples of 3D ultrasound liver and prostate patches plus reconstructions from all experiments are shown in Figure 4.7.

Experiment E1 In experiment E1, the subject and organ specific autoencoders AE1 were applied to the corresponding test sets to determine a baseline. Therefore, the baseline was created using autoencoders that are trained specific for the organ and the subject. The mean NCC values of the baselines range from 0.81 to 0.85 for liver patches and 0.64 to 0.81 for prostate patches. Additionally, a mean filtered version of the original ultrasound patches are shown for comparison. It can be seen

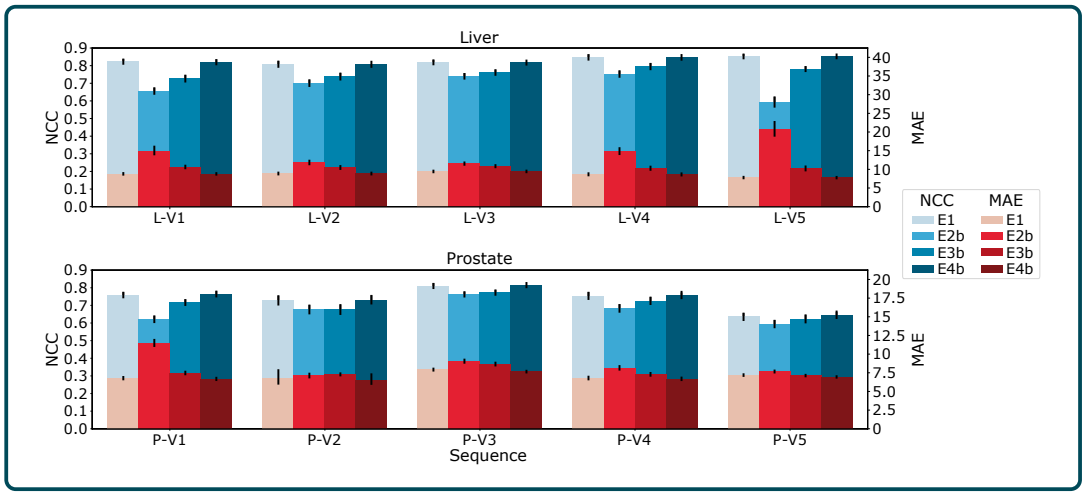

Figure 4.6: Reconstruction accuracy measured by MAE (red) and NCC (blue) observed in the cross organ experiments separated into the liver and prostate sequences.

that the reconstructed patches are smoother than the original patches. Ultrasound images are affected by noise and imaging artifacts such as speckle, which cannot be adequately learned by the autoencoder and, therefore, cannot be reconstructed. The missing noise results in a difference between the original and reconstructed patches that can be considered acceptable because no image information is lost, but it affects the accuracy of the reconstruction. The autoencoder smooths the patches in an unknown way. To approximate the difference between the original patches and the smoothed patches, a $3 \cdot 3 \cdot 3$ vx mean filter was applied to all patches in the test datasets by

$$\mathbf{I}_{\text{filtered}}(x, y, z) = \frac{1}{27} \cdot \sum_{i,j,k \in \{-1,0,1\}}^{27} \mathbf{I}(x + i, y + j, z + k) \tag{4.17}$$

where the image voxel intensity of the ultrasound image $\mathbf{I} \in \mathbb{G}$ at the location (x, y, z) is adapted by averaging the voxel intensities of the local neighborhood. The mean NCC and MAE between the original and filtered patches are 0.94 and 4.07, respectively. These numbers represent an upper and lower bound, respectively, for the reconstruction quality of the autoencoder that is not expected to be exceeded, and thus must be taken into account when interpreting the reconstruction accuracy of an autoencoder.

Experiment E2 Experiments E2a and E2b show that the application of the autoencoders in a cross-subject as well as in a cross-organ manner leads to a decrease of the NCC and an increase of the MAE compared to the baseline experiments E1. This could be explained by inter-subject variability, because although all 3D ultrasound sequences of an organ cover the same anatomical region, vessels and other fine structures differ between subjects. An autoencoder trained on one subject cannot reconstruct patches of another subject with high accuracy. The variability in a given data set used for training is too small to transfer the autoencoder to another subject or organ. Figure 4.7 illustrates that in the reconstructions of experiments E2a and

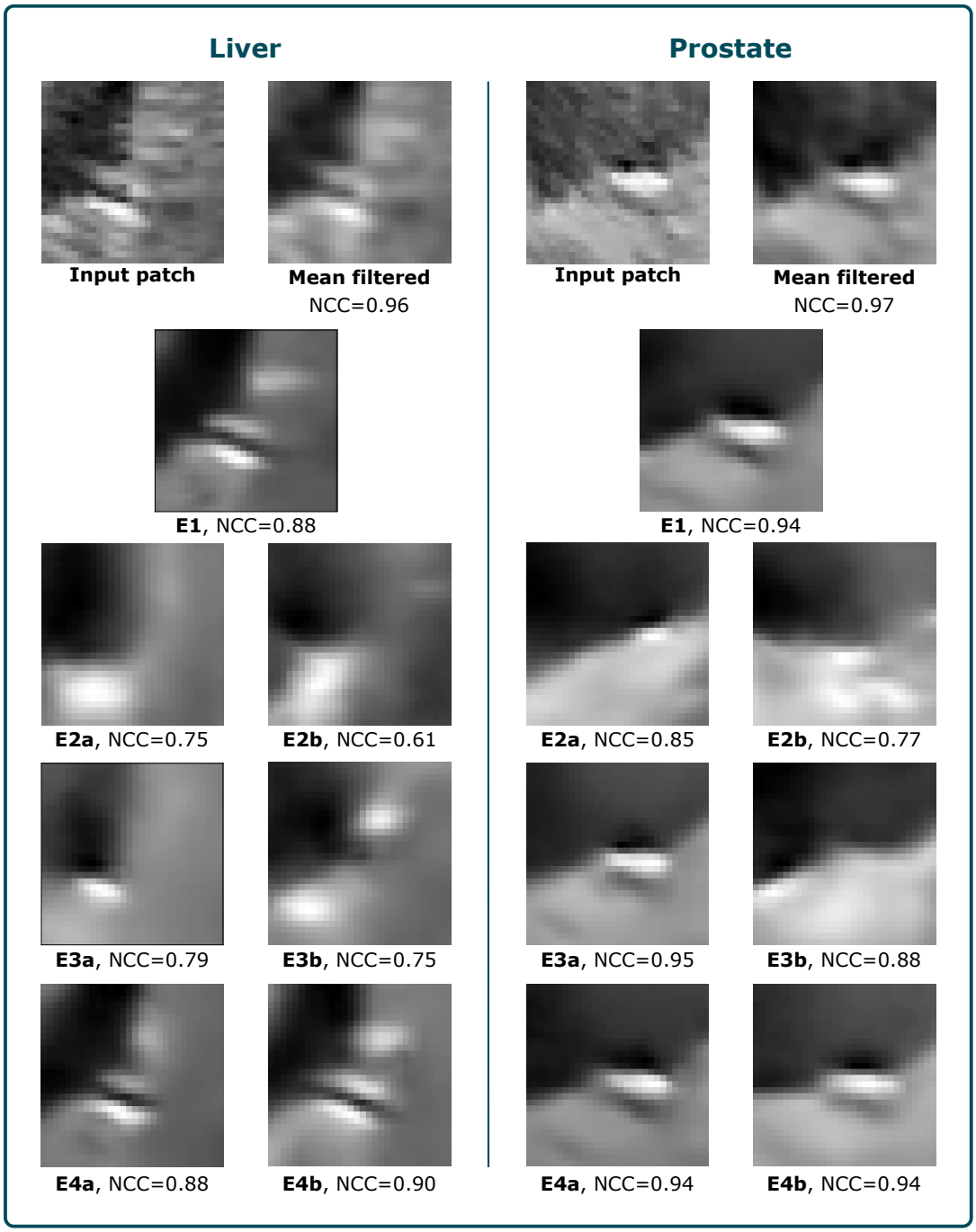

Figure 4.7: Comparison between slices of original and reconstructed patches of the liver (left) and prostate (right) depending on the experiments E1-4. Reconstructed patches of organ specific (left sub-blocks) as well as cross organ (right sub-blocks) autoencoders are presented. The NCC values determined in comparison to the input patch are given explicitly.

E2b, rough structures of the original patches can be identified, but the autoencoders could not recover finer structures.

Experiment E3 In experiment E3, the autoencoders were trained with data from an organ of different subjects. As shown in Figure 4.5, NCC and MAE enhanced

in experiments E3a and E3b compared to E2. While the liver NCC and MAE values in experiment E3a approached the baseline, the prostate results exceeded the baseline. The inter-subject variability in structure for prostate ultrasound images is lower than for liver images. Based on data from four different subjects, the prostate-specific autoencoders were able to reconstruct patches of an unknown subject with high accuracy, in contrast to the liver-specific autoencoders. Thus, an organ-specific autoencoder trained on multiple subjects performs with higher accuracy than the highly individual baseline trained on only one subject. Due to the higher structural variability, liver-specific autoencoders most likely need a larger number of different subjects to outperform the baseline with a subject-general autoencoder. This indicates that it is feasible to train a subject-general autoencoder if a sufficient amount of training data is available.

Experiment E4 Experiments E4 shows that fine-tuning can improve reconstruction accuracy if the autoencoder has not been pre-trained on a sufficient amount of data. In the organ-specific experiment E4a, the accuracy of the autoencoders trained and applied to the liver increased and reached baseline, while the accuracy of the autoencoders applied to the prostate remains constant compared to experiment E3a. These autoencoders could not be improved by fine-tuning. In addition, experiment E4b shows that the accuracy of all autoencoders increases with respect to experiments E3 and almost reaches the baseline. Fine-tuning an autoencoder that has been pre-trained on a different organ greatly improved accuracy, as can be seen in the example visualized in Figure 4.7.
The results show that there is no clear difference between fine-tuning an autoencoder that was pre-trained on the same organ or on a different organ. By fine-tuning the autoencoder on subject and organ specific data, the accuracy of the autoencoder was greatly increased. Furthermore, the results indicate that it is possible to train an autoencoder that does not need to be fine-tuned (E3a, prostate), if enough training data is available to overcome organ-specific subject variability. This is a promising finding for the purpose of 4D ultrasound target tracking for therapy guidance. A generalized autoencoder could be applied for different patients without the need of training patient individual autoencoders from scratch or fine-tuning a pre-trained autoencoder.

4.2.2 Autoencoder Comparison for 3D Ultrasound

In the previous section it has been shown that a latent representation space for 3D ultrasound patches can be learned using an autoencoder. However, so far only the ability to reconstruct the original ultrasound patch from a representation vector was investigated. Although this is an indicator that the latent representation vector is meaningful and contains the most important features of the ultrasound patch in an abstract way, it does not indicate usability for target tracking. In addition, it is still unclear whether the autoencoder used above is the best choice, or whether another type of autoencoder might perform better. To investigate these aspects, a comparative experiment between autoencoder types was conducted to determine whether the conventional autoencoder (AE), the variational autoencoder (VAE), or

the sliced-Wasserstein autoencoder (SWAE) is more advantageous for tracking a target in 4D ultrasound[7].
As shown in section 3.4, the motion a soft tissue target in the liver is exposed to can be divided into translation, rotation and deformation. This motion can also be observed in 3D ultrasound sequences. For the purpose of target tracking, target deformation and rotation are allowed as these transformations change the target appearance, but not the location. In contrast, translation changes the target location within the ultrasound image. In this case, in a tracking procedure, the actual target location must be relocated. Thus, the usability of the autoencoder for tracking can be measured by determining its ability to cluster ultrasound patches affected by allowed transformations while pushing out patches affected by not allowed transformations in the representation space. For simplicity, only deformation was considered as allowed transformation, and translation was considered as not allowed. Rotation is neglected here.

Autoencoder Training

The model architecture is the same for all autoencoder types to ensure comparability. It gets a 3D ultrasound patch $\mathbf{P} \in \mathbb{G}^{30\times30\times30}$ and encodes it into a latent representation vector $\mathbf{z} \in \mathbb{R}^{128} = \varphi(\mathbf{P})$ by the encoder consisting of four convolutional layers and two average-pooling layers. The decoder is built symmetrically to the encoder and predicts a reconstructed ultrasound patch $\hat{\mathbf{P}} \in \mathbb{G}^{30\times30\times30} = \delta(\mathbf{z})$. For model training, 20,000 3D ultrasound patches are randomly cropped from the training segment of the liver sequence L-V1. Autoencoder training was performed for 100 epochs using the Adam optimizer and a batch size of 32. The reconstruction loss used for all autoencoder types is the MSE loss. However, for the VAE the Kullback-Leibler loss is added by

$$L_{VAE} = L_{MSE}(\mathbf{P}, \hat{\mathbf{P}}) + \lambda_{KL} L_{KL}(P_\varphi(\mathbf{z}|\mathbf{P})) \tag{4.18}$$

where $P_\varphi(\mathbf{z}|\mathbf{P})$ denotes the distribution of the latent representation space and λ_{KL} is a weighting parameter. For the SWAE the sliced-Wasserstein loss is added by

$$L_{SWAE} = L_{MSE}(\mathbf{P}, \hat{\mathbf{P}}) + \lambda_{SW} L_{SW}(p_\mathbf{z}, p_\gamma). \tag{4.19}$$

During the training process, the SWAE learns to spread all samples into a representation space at a hyperbullet shaped distribution p_γ. This is done by using a 2D circle distribution for the sliced-Wasserstein loss implemented by randomly sampling an angle

$$\theta = 2\pi \cdot p, \quad p \sim \mathcal{U}_{[0,1]} \tag{4.20}$$

where $\mathcal{U}_{[0,1]}$ is a uniform distribution in the range $[0, 1]$. The 2D Cartesian coordinates $\mathbf{x} = (x_1, x_2)^T$ are determined by

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \cdot r, \quad r \sim \mathcal{U}_{[0,1]} \tag{4.21}$$

where r is a randomly sampled radius parameter. Therewith, $\mathbf{x}$ is a coordinate randomly sampled from a sphere-shaped distribution with the radius $r = 1$.

[7] The autoencoder comparison study presented here was published in [112].

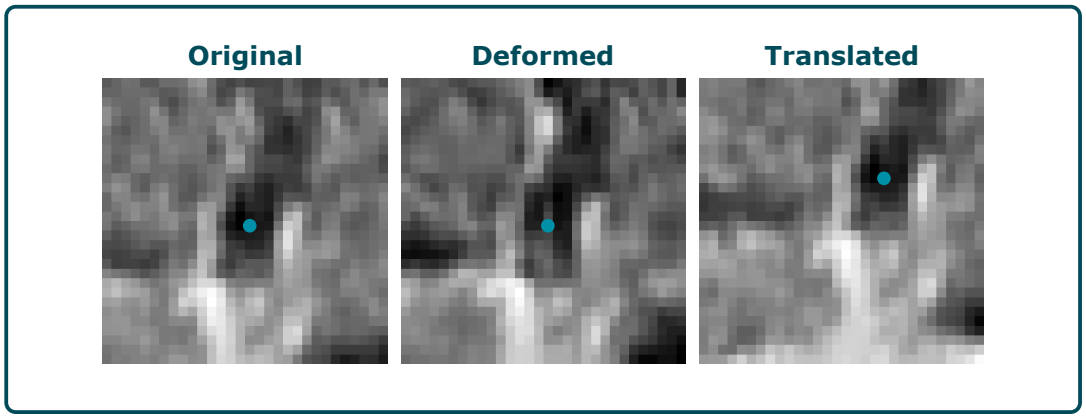

Figure 4.8: Slices of the xy-plane of a original 3D ultrasound patch (left) and a deformed (middle) as well as a translated (right) version. Deformation is performed randomly using the *elasticdeform* framework and translation is performed by adding a shift of $\mathbf{s} = (5, -4, 0)^T$ to the patch location. The blue marker denotes the center of the original ultrasound patch and shifted according to the transformations applied.

Evaluation

For evaluation, ten test ultrasound patches were randomly cropped from the 3D ultrasound liver sequence L-V1. For each test patch $\mathbf{P}$, 50 deformed and 50 translated patches were created, respectively, in order to generate a translation and a deformation test set containing 500 3D ultrasound patches each. A translated patch was generated by adding a random offset $\mathbf{s} \sim \mathcal{U}_{[(-10,-10,-10)^T,(10,10,10)^T]}$ sampled from a uniform distribution to the location of the test patch and cropping the patch from the new location. A deformed patch was generated by using the *elasticdeform* framework [126]. A sparse deformation field $\mathbf{F}_{\text{sparse}} \in \mathbb{R}^{5\times5\times5}$ was randomly created by sampling the deformation vectors of $\mathbf{F}_{\text{sparse}}$ from a standard normal distribution ($\mathbf{F}_{\text{sparse},i} \sim \mathcal{N}(0,1)$). The *elasticdeform* framework creates a dense deformation field $\mathbf{F}_{\text{dense}} \in \mathbb{R}^{30\times30\times30}$ by interpolating the missing deformation vectors. Subsequently, the deformation field $\mathbf{F}_{\text{dense}}$ was applied to the test patch to create a deformed ultrasound patch $\mathbf{P}_{\text{deformed}} = \mathbf{P} \circ \mathbf{F}_{\text{dense}}$ ($\circ$ denotes warping). In Figure 4.8, examples for an original ultrasound test patch, a deformed and a translated test patch are presented. The translation and deformation test sets were mapped into the representation spaces of the AE, the VAE and the SWAE. For evaluating the distribution of the latent representation vectors in the representation spaces, two tests were performed. First, k-means clustering was applied to the translation and deformation test sets in all representation spaces. The clustering results were evaluated by determining the precision defined by

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.22}$$

where TP and FP denote true positive and false positive classifications, respectively. Second, Calinski-Harabasz scores (CH-scores) were determined [127]. The CH-score is a metric for quantifying the ability to distinguish between clusters in a data set. It determines the ratio of the internal dispersion of clusters and the dispersion between the clusters. For a set of N data samples $\mathcal{D} = \{\mathbf{d}_1, ..., \mathbf{d}_N\}$ with K clusters

Table 4.3: Precision and CH score of clustering results in representation spaces of an autoencoder (AE), a variational autoencoder (VAE) and a sliced-Wasserstein autoencoder (SWAE) for 3D ultrasound patches containing translations and deformations.

Autoencoder	Translation		Deformation	
	Precision	CH-score	Precision	CH-score
AE	0.6	28	1.0	1197
VAE	0.5	8	0.8	60
SWAE	0.7	33	1.0	2385

the CH-score is defined as

$$\text{CH-score} = \frac{(N-K)\cdot\sum_{k=1}^{K} n_k\|\mathbf{c}_k-\mathbf{c}\|^2}{(K-1)\cdot\sum_{k=1}^{K}\sum_{i=1}^{n_k}\|\mathbf{d}_i-\mathbf{c}_k\|^2} \tag{4.23}$$

where n_k and $\mathbf{c}_k$ are the number of data samples and the centroid of the k-th cluster, respectively, and $\mathbf{c}$ is the global centroid of the whole data set [127], [128]. The higher the CH-score, the better the ratio and the easier to differentiate between the clusters.

Clustering Results

The results of both tests are presented in Table 4.3. It can be seen that all tested autoencoders were able to learn a meaningful representation space for the ultrasound patches, but clustering them by their representation vectors was less effective for the translation test set. This indicates that the locations of structures inside the patches are encoded in the representation vectors and these vectors could not be well clustered in a representation space. Patches with a translated target have a high distance in the representation space so the clusters could not be separated. This aspect was confirmed by the small CH-scores compared to the deformation experiment. Concerning the aim of performing tracking, this is a promising characteristic as translation is a not allowed transformation and should be far apart the target representation. In the deformation experiment the precision was even 1.0 in AE and SWAE, meaning, the clusters could be separated without errors. In the VAE representation space the precision is the lowest in both experiments. This could be caused by the sampling procedure from a normal distribution in the encoder part of the VAE. In the deformation experiment the CH-scores of both AE and SWAE were high. This indicates that the patches inside a cluster are close just as the distances between the clusters are high. For performing tracking, this is advantageous as patches showing the same target from different time-dependent images are close and patches showing different targets are far away from each other.
In this experiment, a method to analyze representation spaces for their usability for tracking was proposed. This study indicates that SWAEs and AEs are more suitable for tracking tasks than VAEs due to higher separation performance of a set of dissimilar patches by its representations. Due to the very high CH-score that is achieved in the SWAE representation space, this autoencoder seems to be a promising technique for tracking in 4D ultrasound.

4.2.3 Dimensionality of the Representation Space

In order to generate meaningful representation vectors of 3D ultrasound patches usable for target tracking, both the ultrasound patch size and the representation vector size must be chosen properly. In this work, the ultrasound patch size is chosen manually according to the ultrasound image resolution and the hepatic vessel sizes. Since the ultrasound image data set used in this work was acquired from healthy volunteers, the size of normal hepatic vessels is considered as an average. Due to missing tumorous tissue in the data set, the vessels are used as targets to be tracked. The major hepatic veins (right and left hepatic veins) have an average diameter of 15 mm [129]. The average diameter of hepatic arteries range from 2.09 mm (left hepatic arteries) to 5.07 mm (common hepatic arteries) [130]. Considering the ultrasound image resolution of $1.0 \cdot 0.7 \cdot 1.5\,\text{mm}^3$, a patch size of $24 \cdot 24 \cdot 24$ vx is sufficient for fully imaging a liver vessel part as such a patch covers an area of the size $24.0 \cdot 16.8 \cdot 36.0\,\text{mm}^3$. Hence, even a major hepatic vein part could fully be imaged as the diameter is smaller than the ultrasound patch size. Hepatic arteries have even smaller diameters and, thus, do also fully fit into such an ultrasound patch. Therefore, the patch size is set to $24 \cdot 24 \cdot 24$ vx for target tracking in this work.
In the experiments presented in sections 4.2.1 and 4.2.2 a larger patch size of $30 \cdot 30 \cdot 30$ vx was used. The patches were mapped into a 50-dimensional and 128-dimensional representation space meaning a data reduction by a factor of $\frac{1}{540}$ and $\frac{1}{211}$, respectively. However, in these experiments, the size of the representation space was chosen manually. So far, it is unclear what size the representation space should ideally have to be able to contain the most important features of a 3D ultrasound patch. The size of the representation space should be chosen as small as possible but as large as required. A balance must be found in order to be able to store the most important features, and, at the same time, to minimize the dimensionality of the representation space. Due to the *curse of dimensionality* simply enlarging the representation space dimensionality to offer as much space as possible can bear disadvantages [131]. For instance, the performance of similarity and distance measurements is highly affected as in high dimensions distances between close points and points far apart become similar making it impossible to differentiate between them [132]. In addition, too much space in representation space can lead to redundant features which highly affect clustering analyses or distance measurements [133]. These aspects would highly influence the performance of a tracking algorithm where finding the target by applying distance measurements is the key idea. In order to find the optimal size for the representation space, in the following, different representation space sizes are investigated by measuring the ultrasound patch reconstruction accuracy[8].

Data Preparation

In this experiment, the 3D ultrasound sequences L-Vi with the subjects $i \in I = \{1, 2, 3, 4, 5\}$ were used. For autoencoder training, 50,000 3D ultrasound patches from each sequence were cropped from images of the training phases at random

[8]The investigation of the optimal size of the representation space presented here was partially published in [134].

Table 4.4: Ultrasound patch data sets used for autoencoder training and reconstruction accuracy evaluation based on the subjects $i \in I = \{1, 2, 3, 4, 5\}$.

Name	Usage	Amount	Patch size
$\mathcal{T}_i$	Training	250,000	$24 \cdot 24 \cdot 24$ vx
$\mathcal{V}_i$	Test	25,000	$24 \cdot 24 \cdot 24$ vx

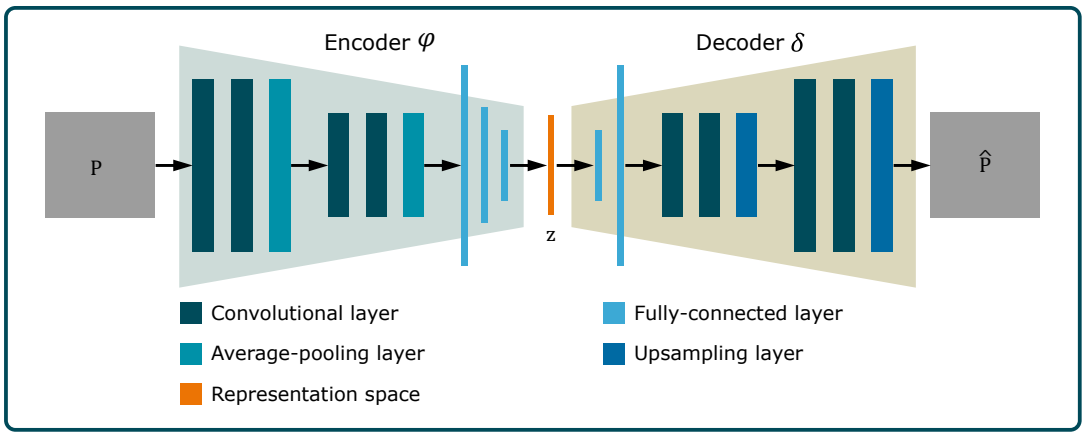

Figure 4.9: General architecture of the autoencoders. The encoder consists of four convolutional, two average-pooling and three fully-connected layers and the decoder consists of four convolutional, two upsampling and two fully-connected layers. Several autoencoders with different representation spaces $\mathbf{z} \in \mathbb{R}^k$ are used.

positions. In total, 250,000 3D ultrasound patches were generated for training. For evaluating the reconstruction accuracy a test set containing 25,000 ultrasound patches was generated similarly. In order to ensure a proper temporal data split between training and test data as well as the sequences used for tracking, the test patches were cropped from ultrasound images that are not part of the training segment or the labeled sequence parts. The patch data sets that were generated and used in this study are summarized in Table 4.4.

Autoencoder Training

In this study, a sliced-Wasserstein autoencoder (SWAE) was used. The autoencoder architecture consists of an encoder that encodes the ultrasound patch $\mathbf{P} \in \mathbb{G}^{24\times24\times24}$ into a latent representation vector $\mathbf{z} \in \mathbb{R}^k = \varphi(\mathbf{P})$ with $k = 128$ and a decoder that predicts the ultrasound patch reconstruction $\hat{\mathbf{P}} \in \mathbb{G}^{24\times24\times24} = \delta(\mathbf{z})$. The applied architectures of the encoder and decoder were determined empirically. The whole autoencoder architecture is presented in Figure 4.9. The encoder consists of four convolutional layers, two average-pooling layers and three fully-connected layers. The decoder consists of four convolutional layers, two upsampling layers and two fully-connected layers. The SWAE was trained using the loss function defined as

$$L_{SWAE}(\mathbf{P}, \hat{\mathbf{P}}, p_{\mathbf{z}}) = \lambda_{MSE} \cdot L_{\text{MSE}}(\mathbf{P}, \hat{\mathbf{P}}) + \lambda_{SW} \cdot L_{\text{SW}}(p_{\mathbf{z}}, p_{\gamma}) \tag{4.24}$$

consisting of the MSE and the sliced-Wasserstein loss. Here, $\mathbf{P}$ and $\hat{\mathbf{P}}$ are the original and reconstructed ultrasound patch, respectively, and p_z and p_γ are the distribution

Table 4.5: Overview of the three types of trained autoencoders. The index $i \in I = \{1, 2, 3, 4, 5\}$ defines the 3D ultrasound liver sequence L-Vi where training and validation data is taken from.

Approach	Autoencoder	Training Data	Test Data
Individual	$AE_{indiv,i}$	$\mathcal{T}_i$	$\mathcal{V}_i$
Generalized	$AE_{gen,i}$	$\mathcal{T}_h, h \in I \backslash i$	$\mathcal{V}_i$
Generalized fine-tuned	$AE_{genFT,i}$	$AE_{gen,i} + \mathcal{T}_i^*$	$\mathcal{V}_i$

* Fine-tuning was performed only with 5,000 patches randomly selected from the data set.

in representation space and the target sphere-shaped distribution (see equations 4.20 and 4.21), respectively. The weighting parameters $\lambda_{MSE} = 1.0$ and $\lambda_{SW} = 0.1$ were determined empirically.
Several autoencoders were generated in three different ways. Sets of subject individual autoencoders ($AE_{indiv,i}$, five sets), subject generalized autoencoders ($AE_{gen,i}$, five sets) as well as generalized fine-tuned autoencoders ($AE_{genFT,i}$, five sets) were trained. Each way of training was performed for all subjects i and for seven different representation space sizes $k \in \{8, 16, 32, 64, 128, 256, 512\}$ each. Reducing the 3D ultrasound patches into the representation space sizes from K corresponds to a data reduction by $96.30 - 99.94\%$. Training was performed with the training data sets $\mathcal{T}_i$, the Adam optimizer, a batch size of 32 and early stopping to avoid overfitting. For this, a validation split of 5% was chosen. The autoencoders $AE_{indiv,i}$ were trained with one data set of $\mathcal{T}_i$ each. Autoencoders $AE_{gen,i}$ were trained with four out of five data sets of $\mathcal{T}_h$ with $h \in I \backslash i$ so that subject generalized autoencoders were generated. In the last step, these generalized autoencoders were fine-tuned with 5,000 ultrasound patches of the remaining data set $\mathcal{T}_i$. In the fine tuning process, only the fully connected layers of the encoder and decoder were updated. Table 4.5 gives an overview about all trained autoencoders and the used data sets.

Reconstruction Accuracy Experiments

After training the three types of autoencoders, the reconstruction accuracy was evaluated based on the test data sets $\mathcal{V}_i$. The 3D ultrasound patches from the test data sets were fed into the SWAE models to predict a reconstructed version $\hat{\mathbf{P}}$. The reconstruction accuracy was measured by determining the Normalized Cross Correlation (see equation 3.6) between the original $\mathbf{P}$ and the reconstructed ultrasound patch. The results for the three autoencoder types are presented in Figure 4.10. It can be seen that the reconstruction accuracy (measured by NCC) increased with the representation space size and saturates at a mean accuracy of about 0.73 ± 0.02, 0.75 ± 0.02 and 0.74 ± 0.02 for individual, generalized and generalized fine-tuned autoencoders, respectively at a representation space size of 128. Increasing the representation space size to 256 or 512 dimensions did not bring a significant improvement in reconstruction accuracy (p-values > 0.05). By performing a t-test analysis, it turned out that the reconstruction accuracy improves significantly by increasing the representation space size from $k = 8$ to the size of $k = 128$ (p-value < 0.05) for the generalized and generalized fine-tuned autoencoders. For the individual autoencoders, no significant difference could even be measured between the per-

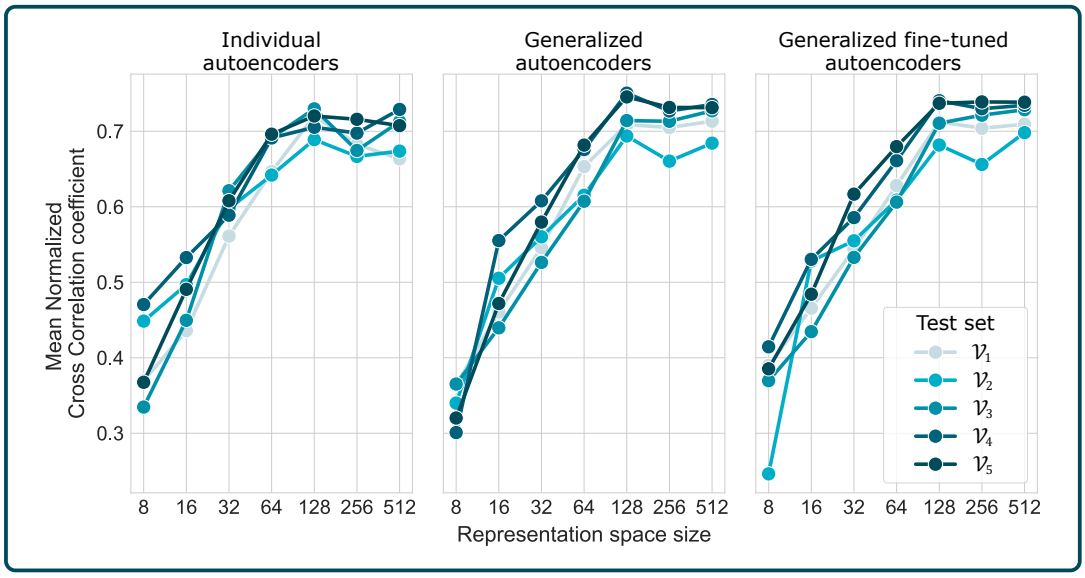

Figure 4.10: Reconstruction accuracy of the autoencoders $AE_{indiv,i}$ (left), $AE_{gen,i}$ (middle) and $AE_{genFT,i}$ (right) with different representation space sizes.

formances of the autoencoders with 64-, 256- and 512-dimensional representation spaces. However, also no significant difference could be measured between 128-, 256- and 512-dimensional representation spaces here (p-values > 0.05). The results indicate that a 128-dimensional representation space is of sufficient size to encode the most important features of an ultrasound patch of the size $24 \cdot 24 \cdot 24$ vx. In addition, the results show that the reconstruction accuracy is comparable for all three types of autoencoders. This is a promising finding for the target tracking purpose as it indicates that it is not necessary to train patient individual autoencoders. Fine-tuning a pre-trained autoencoder or even using a generalized pre-trained autoencoder could be sufficient. However, so far the autoencoder was evaluated only based on the reconstruction accuracy which can indicate that the autoencoder was trained successfully but it cannot show the usability for the target tracking purpose. In section 6.2.3, the tracking performances of these SWAEs are evaluated.

4.3 Summary

In order to develop a feature-based tracking algorithm, methods for feature detection, description and matching were investigated in this chapter. The main focus was on the most crucial part, the method to describe local image features as uniquely and meaningfully as possible. In a target tracking algorithm, the feature description is the main indicator for finding correspondences between ultrasound images, and thus, determining the motion occurred between these images. Therefore, determining robust and unique feature descriptions is essential for a feature-based tracking algorithm. In order to investigate the most promising methods for this purpose, this chapter addressed

Research Question 2
How can meaningful and unique features be extracted from 3D ultrasound images?

Applying a feature-based tracking algorithm can be done in two different ways. The target to track itself can be defined as a feature, e.g., by getting marked by a physician in an initial ultrasound image. The goal is to relocate the target feature as accurately as possible in the subsequent ultrasound images. Another approach is to use a feature detector to find several image features in both, the previous and the current ultrasound image. Here, the goal is to find feature correspondences between the ultrasound images, and subsequently, determine the motion between them. For this approach, a FAST-3D keypoint detector was developed and tested on 3D ultrasound images for the first time. It turned out that such a simple feature detector can be used for finding image features in 3D ultrasound images. Due to the real-time capability of the FAST algorithm, this is a promising finding for usage in a tracking algorithm that has a real-time requirement. However, in FAST-3D parameters need to be set manually. For now, it is unclear how these parameters should ideally be set and if the parameters need to be adapted if ultrasound machine settings are changed, or if another ultrasound probe is used.
For feature description, the image feature descriptors BRISK-3D and BRIEF-3D were developed and tested on 3D ultrasound images for the first time in this work. It was shown that such simple binary feature descriptors can be used for describing image features in 3D ultrasound images. Feature correspondences between consecutive ultrasound images could be found and matched successfully. However, like in FAST-3D, the feature descriptors require manual parameter settings which can complicate the usage since optimal parameters can differ, e.g., between ultrasound sequences acquired with different ultrasound machine settings.
In addition, a deep representation learning-based approach by using autoencoders was used for feature description. This approach has advantages such as being fully unsupervised, since no labels are required and no parameters need to be set in the prediction phase. These aspects make this approach very promising for the clinical workflow as the whole training process as well as the tracking algorithm has the potential to be fully unsupervised. Thus, the effort clinicians need to spend preparing the tracking algorithm for individual patients can be minimized.
There is a lack of research on determining abstract representation vectors for 3D ultrasound patches using autoencoders. This work presents the first study on extracting the most important features from 3D ultrasound patches using an autoencoder. The autoencoder was evaluated on determining the reconstruction accuracy. It was shown that such an autoencoder is able to generalize over different patients as well as over different organs. The results indicate that a robust generalized autoencoder could be trained if a data base of sufficient size was available. The space used for storing the information of an ultrasound patch could be reduced to less than 0.2 % of the patch size by using an autoencoder. There are different types of autoencoders known in literature. In order to determine the most promising type for the purpose of tracking in ultrasound sequences, three types, the conventional autoencoder (AE), the VAE and the SWAE were compared. A clustering analysis was executed

in the representation space to quantify the ability to distinguish between different ultrasound patches according to their representation vectors. The results indicate that the SWAE is the most promising type. In addition, the ideal dimensionality of the representation space of a SWAE is investigated based on the reconstruction accuracy. It turned out that for ultrasound patches of the size $24 \cdot 24 \cdot 24\,\mathrm{vx}$ a 128-dimensional representation space is sufficient as larger spaces did not result in higher reconstruction accuracy. This is a reduction of space for storing the ultrasound patch information by about 93 %. In contrast, smaller representation spaces resulted in significantly lower reconstruction accuracy.
The results obtained in this chapter fill the lack of research on local feature extraction in 3D ultrasound images mentioned in the research question. Local image features can be detected, described and matched in 3D ultrasound images using image feature extractors and deep learning approaches, which form the basis for feature-based target tracking algorithms in 4D ultrasound.

5 ULTRASOUND IMAGE AUGMENTATION

For training deep neural networks, a high amount of training data that covers the relevant variety of variability sufficiently is required to achieve robust results on unknown data. In the medical image domain, especially in the 3D ultrasound image domain, data sets of sufficient size are rare. A common method in deep learning to overcome this challenge is performing data augmentation [135]. The data set is extended synthetically by applying different methods, such as geometric transformations or color adaptions or by fully synthesizing new data samples. Methods for data augmentation can be divided into two groups: Transformation and Generation [136]. Different data transformation approaches are known such as affine or elastic transformations, or image adaptations like contrast or brightness changes in the pixel-level. All of them basically change original data in order to create new data samples. In contrast, data generation methods like generative neural networks or reconstruction-based approaches intend to fully synthesize new data samples, with the aim of sampling synthetic data from a highly similar data distribution. While transformation methods became a standard tool in the computer vision domain [135], their application on ultrasound images is not straight-forward.
In the ultrasound image domain, some transformation methods can lead to unrealistic images and thus, be disadvantageous for a deep learning application [137]. Martin et al. [138] investigated the effect of augmentation methods, such as flipping, noising and affine transformations for a segmentation task in 2D ultrasound images. The study revealed that only flipping led to a significant improvement in the segmentation task. This indicates that finding suitable augmentation methods can be challenging in the ultrasound image domain. Therefore, approaches to perform more realistic augmentation were proposed [139]–[141]. Typically, these approaches rely on learning a latent representation space of the data set from which new, synthetic images can be sampled. The representation space can be retrieved utilizing unsupervised deep learning methods, e.g., using a VAE [43] or a GAN [142]. For instance, an approach to synthesize 2D ultrasound images of breast tumors was proposed by Al-Dhabyani et al. [137]. They determined that using a GAN for augmentation instead of common affine augmentation methods lead to better results in a deep learning-based classification downstream task. An ultrasound physics-inspired augmentation approach was proposed by Tirindelli et al. [143] who trained a U-net to simulate the

effects of different transducer pressure forces and realistically augment ultrasound images. Most of these proposed augmentation techniques share the common limitation of being developed and tested for 2D ultrasound images. Even though these approaches could be extended to 3D, there is a lack of investigation for realistic 3D ultrasound augmentation methods. Additionally, 3D ultrasound data sets, e.g. [14], [66] are even rarer than 2D ultrasound data sets like [14], [85], [144], [145], which confirms the need for a realistic 3D ultrasound data augmentation method. For this reason, in this chapter

Research Question 3
Is it possible to perform realistic 3D ultrasound image augmentation using unsupervised deep representation learning and is this augmentation beneficial for a target tracking application?

is addressed. Both, an ultrasound image generation and a transformation augmentation approach are implemented and investigated. First, a qualitative analysis of the cross data set generalization performance of a generation augmentation approach in the 2D ultrasound image domain is presented. Therefore, two different learning-based augmentation models on three different ultrasound image data sets are trained and evaluated in different transfer scenarios. Second, a deformable 3D ultrasound image transformation augmentation approach is presented based on a conditional variational autoencoder (CVAE) that aims to extract and encode realistic breathing-induced deformation patterns from 3D ultrasound images[1].

5.1 Ultrasound Image Synthesis

In recent years, different deep learning-based medical image augmentation techniques were developed and applied to medical image data [139]. It is shown that deep learning-based approaches, like GANs or autoencoders are able to augment and even synthesize medical image data [139], [148], [149]. However, for training a neural network that is able to synthesize realistic data, a sufficient amount of training data with realistic variance is required. This is especially true when the data variance is high like in the ultrasound image domain, which represents one of the most user-dependent and least reproducible medical image domains. The appearance of ultrasound images can differ, e.g., in contrast, brightness, signal to noise ratio or in the size of the field of view. These effects can be due to setting changes on the ultrasound station, ultrasound transducer position adaptation or internal organ motion of the patient. Since in the medical ultrasound image domain the availability

[1] Parts of this chapter were published in:

[146] D. Wulff, M. Mehdi, F. Ernst and J. Hagenah, "Cross Data Set Generalization of Ultrasound Image Augmentation using Representation Learning: A Case Study," Current Directions in Biomedical Engineering, vol. 7, no. 2, pp. 755-758, 2021

[147] D. Wulff, T. Dohnke, N.T. Nguyen and F. Ernst, "Towards Realistic 3D Ultrasound Synthesis: Deformable Augmentation using Conditional Variational Autoencoders," IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS), pp. 821-826, 2023

Table 5.1: Information about the 2D ultrasound image data sets used for training and augmentation

Notation	Domain	Amount	Usage	Reference
$\mathcal{D}^{source}$	vessel phantom	1950	source	[150]
$\mathcal{D}_1^{target}$	female breast	780	target	[85]
$\tilde{\mathcal{D}}_1^{target}$	female breast	100	auxiliary	[85]
$\mathcal{D}_2^{target}$	fetus head	1334	target	[144]
$\tilde{\mathcal{D}}_2^{target}$	fetus head	100	auxiliary	[144]

of sufficiently large data sets is limited, a method is required to overcome this difficulty. For this, an augmentation approach based on deep representation learning is investigated in this section that aims to be generalized for different data domains. Such a generalized model could be used for different domains and also be trained using different domains. Both, model training and data augmentation would benefit from this as the model would not be limited to a specific domain.
In the following, a case study is presented that aims to investigate whether such an augmentation model can be developed for the ultrasound image domain[2]. A GAN as well as a VAE were trained in order to create a latent representation space that is able to represent 2D ultrasound images of different anatomical domains. Training and testing were performed based on three different ultrasound image domains in an intra- and inter- domain as well as a domain transfer way. For simplicity and due to a higher rate of 2D ultrasound image data sets available, the study focuses on 2D ultrasound images but the methods could be extended to 3D easily.

5.1.1 Data Preparation

In line with the goal of developing a model that is able to perform augmentation across domains with small data sets, three small data sets were used containing three different anatomical domains. As presented in Table 5.1, the data sets contain femoral artery images of a phantom leg [150], female breast images [85] and images of fetus head circumferences [144]. Note that the fetus images show full cone-shaped 2D ultrasound images while breast and phantom images show region-of-interest patches. For the experiments the data sets were divided into three data sets $\mathcal{D}^{source}$, $\mathcal{D}_1^{target}$ and $\mathcal{D}_2^{target}$, where $\mathcal{D}^{source}$ contains the phantom data, and $\mathcal{D}_1^{target}$ and $\mathcal{D}_2^{target}$ contain the breast and fetus data, respectively. To assess the influence of very small data sets in one experiment, auxiliary target data sets $\tilde{\mathcal{D}}_1^{target}$ and $\tilde{\mathcal{D}}_2^{target}$ were generated containing 100 randomly sampled images from the corresponding target data sets. In a preprocessing step all 2D ultrasound images were resized into a uniform shape of 128×128 px and the gray scale values are normalized to the range [0,1] for VAE and [-1,1] for GAN, respectively.

[2]The 2D ultrasound image augmentation study presented here was published in [146].

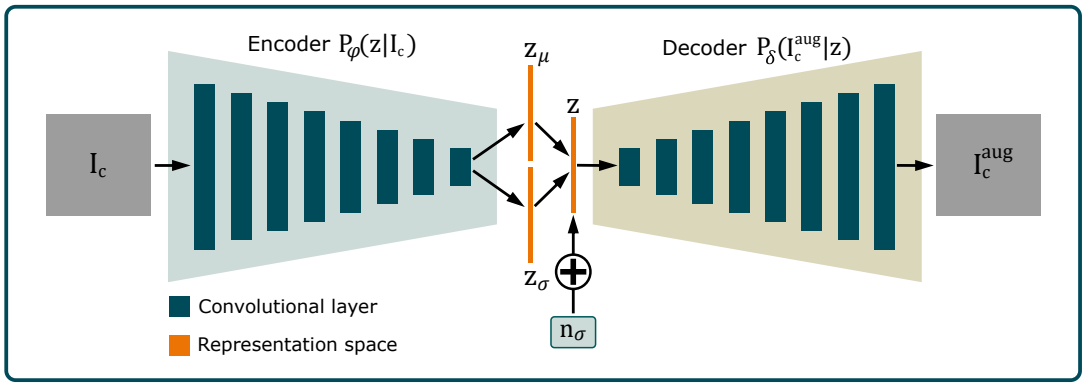

Figure 5.1: Architecture of the VAE trained for performing 2D ultrasound image augmentation. In the bottleneck a random noise vector $\mathbf{n}_\sigma$ is added to the latent representation vector $\mathbf{z}$ in order to change the representation and to decode an adjusted ultrasound image.

5.1.2 Neural Network Training

The key idea of this approach is to generate a latent representation space that is capable of encoding a wide range of different ultrasound images. With this, data augmentation can be performed by sampling from this representation space. For creating such a representation space, two approaches are proposed: A VAE as well as a GAN are trained based on the data set described above. In the following, the architectures of the VAE and GAN and the training procedure are described in detail. Both models basically aim to do the same: Generate an augmented ultrasound image $\mathbf{I}_c^{\text{aug}}$ for a given ultrasound image $\mathbf{I}_c$ by using a random augmentation vector $\mathbf{n}_\sigma$. This is done by mapping $\mathbf{I}_c$ into a latent representation vector $\mathbf{z}_c$, adapting the representation by adding the noise vector $\mathbf{z}_c^{\text{aug}} = \mathbf{z}_c + \mathbf{n}_c$ and synthesizing the corresponding augmented image $\mathbf{I}_c^{\text{aug}}$. The challenge is to balance between adding novel information to enhance the data set and constraining the difference to the original image to keep the synthetic image within the realistic data distribution. This is done by using a trained model M by

$$\mathbf{I}_c^{\text{aug}} = M(\mathbf{I}_c, \mathbf{n}_\sigma). \tag{5.1}$$

Here, $c \in \{\text{Vessel}, \text{Breast}, \text{Fetus}\}$ is the domain of the ultrasound image.

Variational Autoencoder

The architecture of the VAE used here consists of a symmetrically built encoder-decoder structure and is presented in Figure 5.1. The encoder $P_\varphi(\mathbf{z}|\mathbf{I}_c)$ gets an ultrasound image $\mathbf{I}_c \in \mathbb{G}^{128\times128}$ and maps it into a latent representation vector $\mathbf{z} \in \mathbb{R}^{64}$. It consists of seven blocks consisting of a convolutional layer, batch normalization and dropout with a rate of 0.2 each, followed by a convolutional layer used for flattening the feature map into the latent representation space. For downsampling the feature maps within the encoder, a stride length of 2 is chosen in every second block. In the decoder $P_\delta(\mathbf{I}_c^{\text{aug}}|\mathbf{z})$, convolutional layers are replaced by transpose convolutional layers in order to synthesize the ultrasound image $\mathbf{I}_c^{\text{aug}}$. However, the general architecture of the decoder is built symmetrically to the encoder. It

consists of a transpose convolutional layer followed by seven blocks consisting of a transpose convolutional layer, batch normalization and dropout with a rate of 0.2. In the whole VAE, LReLU activation function is used. For training, the loss function defined by

$$L_{VAE}(\mathbf{I}_c, \mathbf{I}_c^{\text{aug}}) = L_{MSE}(\mathbf{I}_c, \mathbf{I}_c^{\text{aug}}) + \lambda_{KL} L_{KL}(P_\varphi(\mathbf{z}|\mathbf{I}_c)) \tag{5.2}$$

with the weighting parameter $\lambda_{KL} = 10^{-4}$ is used. Training was performed using a batch size of 128, the RMSprop optimizer with a learning rate of $2 \cdot 10^{-4}$ for 5000 epochs. For ultrasound image augmentation using the VAE, the equation 5.1 is modified to

$$\mathbf{I}_c^{\text{aug}} = \delta\left(P_\varphi\left(\mathbf{z}|\mathbf{I}_c\right), \mathbf{n}_\sigma\right). \tag{5.3}$$

Generative Adversarial Network

A GAN is another method for learning a latent representation space. In contrast to an autoencoder, a GAN consists of two networks and is trained in an adversarial way. This is implemented by using a generator network $\gamma : (\mathbb{G}^d, \mathbb{R}^k) \mapsto \mathbb{G}^d$ that aims to synthesize new data samples and a discriminator network $\delta : (\mathbb{G}^d, \mathbb{G}^d) \mapsto \mathbb{R}$ that aims to distinguish between real and generator created data samples. The generator is forced to create data samples that appear as realistic as possible, while the discriminator is forced to identify synthetic data samples as accurately as possible [151]. The architecture of the GAN used here is presented in Figure 5.2. The generator gets an ultrasound image $\mathbf{I}_c$ and a random noise vector $\mathbf{n}_\sigma$ as input and predicts an augmented version of the image $\mathbf{I}_c^{\text{aug}}$. It is an U-net like network consisting of an encoder with five convolutional layers with a stride length of two to perform down-sampling and the LReLU activation function. The decoder consists of convolutional layers, ReLU activation function and upsampling layers. In the last layer of the decoder, tanh activation function is used. In addition, batch normalization is performed after all convolutional layers except for the first and last convolutional layer of the generator. Four skip-connections are added in between the encoder and the decoder to transfer spatial information from the encoder to the decoder. In the bottleneck of the generator the noise vector $\mathbf{n}_\sigma \in \mathbb{R}^{100}$ is added to the generator via a fully-connected layer enlarging the vector to the required size. Following, the vector is reshaped to the shape of the feature map coming from the encoder and concatenated with it to form the latent representation space. In the training phase, the noise vector is sampled from a standard normal distribution ($\mathbf{n}_\sigma \sim \mathcal{N}(0, 1)$), in the augmentation phase, this vector is used as a factor to adjust the augmentation intensity. As shown in Figure 5.2, the discriminator is an CNN with two branches receiving two images as input $(\mathbf{I}_c, \mathbf{I}_c^i)$ with $i \in \{\text{real}, \text{aug}\}$ denoting a real and an augmented image, and predicting a probability $p \in [-1, 1]$ for $\mathbf{I}_c^i$ being a real image from the domain of $\mathbf{I}_c$. The architecture of the two branches are equal. A branch consists of four convolutional layers with LReLU activation function and a stride length of two for down-sampling. Subsequently, the branches are concatenated and the probability output p is predicted via a fully-connected layer where dropout with a rate of 0.4 is performed. The complete GAN is trained by performing a zero-sum

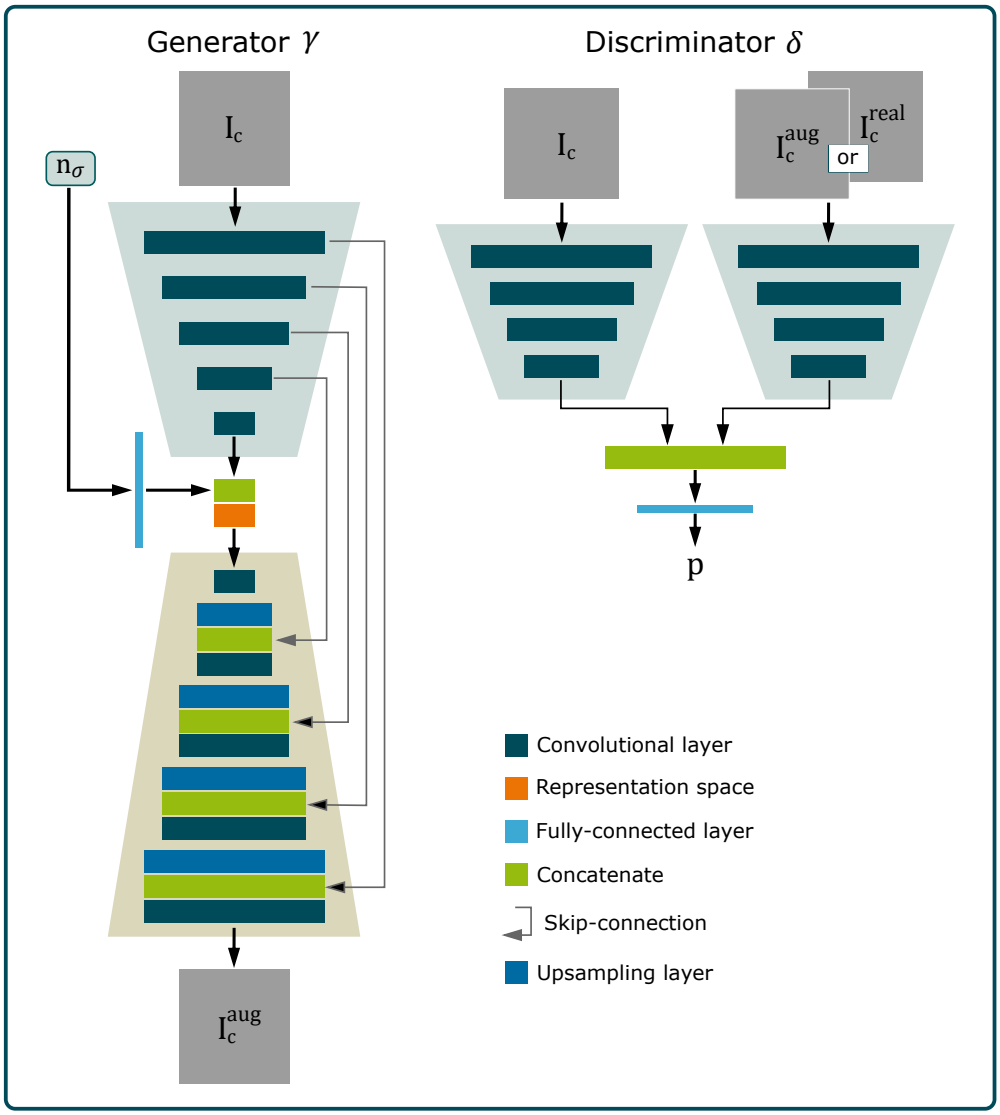

Figure 5.2: Architecture of the generator $\gamma(\mathbf{I}_c, \mathbf{n}_\sigma)$ (left) and the discriminator $\delta(\mathbf{I}_c, \mathbf{I}_c^i)$, $i \in \{\text{real}, \text{aug}\}$ (right) of the GAN trained for 2D ultrasound image augmentation. The generator generates an augmented ultrasound image based on a given image $\mathbf{I}_c$ and a random noise vector $\mathbf{z}_\sigma$ and the discriminator aims to classify whether a given image is real or created by the generator.

game using the loss function

$$\begin{aligned}\min_{\gamma}\max_{\delta} L_{GAN}(\delta,\gamma) &= L_{\text{Discriminator}} + L_{\text{Generator}} \\ &= \mathbb{E}_{\mathbf{I}_c\sim p_{\text{data}}(\mathbf{I}_c)} \log\left(\delta(\mathbf{I}_c, \mathbf{I}_c^i)\right) + \mathbb{E}_{\mathbf{I}_c\sim p_{\text{data}}(\mathbf{I}_c)} \log\left(1-\delta(\gamma(\mathbf{I}_c, \mathbf{n}_\sigma))\right)\end{aligned} \tag{5.4}$$

where p_{data} denotes the distribution of the ultrasound image space [31], [151]. Basically, the generator is trained by maximizing the classification error of the discriminator and the discriminator is trained by minimizing its error. The GAN was trained for 2,000 epochs with a batch size of 64 and the Adam optimizer with a learning rate of 10^{-4}. For ultrasound image augmentation only the generator is used so that

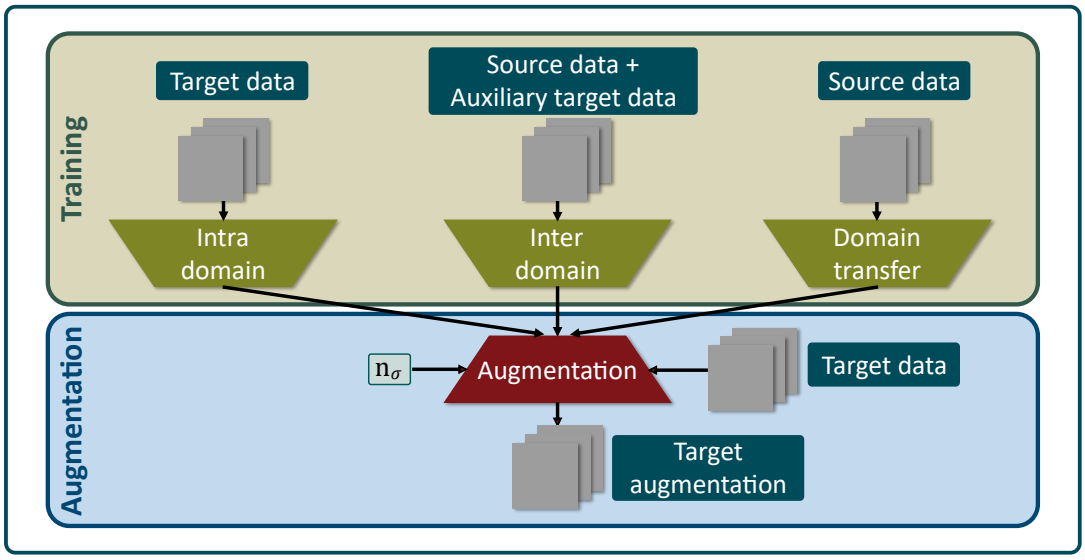

Figure 5.3: Visualization of training and augmentation process of the *intra domain* (left), *inter domain* (middle) and *domain transfer* (right) experiments. Ultrasound image augmentation is performed by adding a random noise vector $\mathbf{z}_\sigma$ to the generator (in GAN) or bottleneck (in VAE) to modify the latent representation of the image.

the equation 5.1 can be given as

$$\mathbf{I}_c^{\text{aug}} = \gamma(\mathbf{I}_c, \mathbf{n}_\sigma). \tag{5.5}$$

5.1.3 Augmentation Experiments

Three different experiments were performed whose general process is summarized in Figure 5.3. First, the VAE and GAN models were trained using three different approaches named *intra domain*, *inter domain* and *domain transfer* that are described in the following. Then, the trained models were used in combination with a target data set to generate augmented ultrasound images of this data set. For this, the random noise vector $\mathbf{n}_\sigma \in \mathbb{R}^k$ is sampled from a normal distribution with variance σ^2 by

$$\mathbf{n}_\sigma \sim \mathcal{N}(0, \sigma^2). \tag{5.6}$$

The variance parameter σ^2 of the normal distribution describes the degree of similarity between the real and the augmented ultrasound image and is a hyperparameter of the method. In all experiments different $\sigma^2 \in \{0.6, 1.0, 1.6, 2.0\}$ were chosen and investigated.
First, a baseline was generated to investigate whether the models are able to generate meaningful ultrasound images performing an *intra domain* experiment. Both, training and augmentation were performed using the same domain which is $\mathcal{D}_1^{target}$ or $\mathcal{D}_2^{target}$. Second, an *inter domain* experiment is performed. It is tested whether a model can augment 2D ultrasound images of a target data set, though the model was trained mainly on the source data set. For model training, $\mathcal{D}^{source}$ plus an auxiliary target data set $\tilde{\mathcal{D}}_1^{target}$ or $\tilde{\mathcal{D}}_2^{target}$ were used. Augmentation was performed using the remainder of the corresponding target data set ($\mathcal{D}_1^{target} \backslash \tilde{\mathcal{D}}_1^{target}$ or $\mathcal{D}_2^{target} \backslash \tilde{\mathcal{D}}_2^{target}$).

The third experiment is a *domain transfer* experiment where training was performed only using $\mathcal{D}^{source}$ and augmentation was done using the target data sets. This experiment investigates whether the models are able to augment images of a target data set though this model was trained on another domain. The image augmentations generated in the experiments were evaluated qualitatively, examples for each experiment and model are visualized in Figure 5.4.

Intra Domain

In the *intra domain* experiment both models were capable of generating variations of the origin ultrasound image. In general, the VAE created images that are strongly blurred. With high σ^2 values of 1.6 and 2.0 the VAE was able to generate breast augmentations that appear highly different than the origin, however, these augmentations seem to be realistic (row (a) inFigure 5.4). Though, fetus augmentations with high σ^2 values were not realistic due to the fact that the fetus head is distorted (row (b) in Figure 5.4). In contrast to VAE, GAN augmentations were less blurry but they appear more similar to the origin even with high σ^2 values. Although, in GAN augmentations several structures appeared clearer and more distinct compared to the origin. In addition, GAN fetus augmentations appeared realistic as there are no unrealistic distortions. The *intra domain* experiment serve as a baseline and indicate that this augmentation approach can lead to meaningful augmentations. The VAE generates blurry images but it can create augmentations that are changed in a larger scale compared to the GAN. However, this characteristic does also lead to unrealistic augmentations which is not observed in GAN augmentations.

Inter Domain

In the *inter domain* experiment VAE augmentations generated with σ^2 values of 0.6 and 1.0 appeared similar to the ones in the *intra domain* experiment. However, with higher σ^2 values the VAE generated augmentations showing a source data image instead of a target data image in multiple cases in both target data sets (rows (c) and (d) in Figure 5.4). As the VAE was mainly trained with the source data set, this was expected. When the distance to a target sample in latent space gets too high, the augmentation became a source data sample. Thus, in this approach $\sigma^2 = 1.0$ should be chosen at maximum to get meaningful target data augmentations. Considering the breast augmentations, changes like structure appearance and deformation of existing structures could be identified (row (c) in Figure 5.4). In the GAN augmentations, the image contrast clearly changed compared to the origin. Though, the breast augmentations appeared realistic and even fine new structures could be identified in some augmentations (row (g) in Figure 5.4). However, the fetus augmentations did not appear to be realistic, even with low σ^2 values as the fetus head is distorted or the structure is completely lost (row (j) in Figure 5.4). This could be explained with the different field of view in the ultrasound images and the different image contrast. Fetus images show a whole ultrasound image whereas phantom and breast images only show a section of an ultrasound image. Thus, phantom and breast images are more similar to each other than to the fetus images. However, even with high σ^2 values the GAN generated target data augmentations.

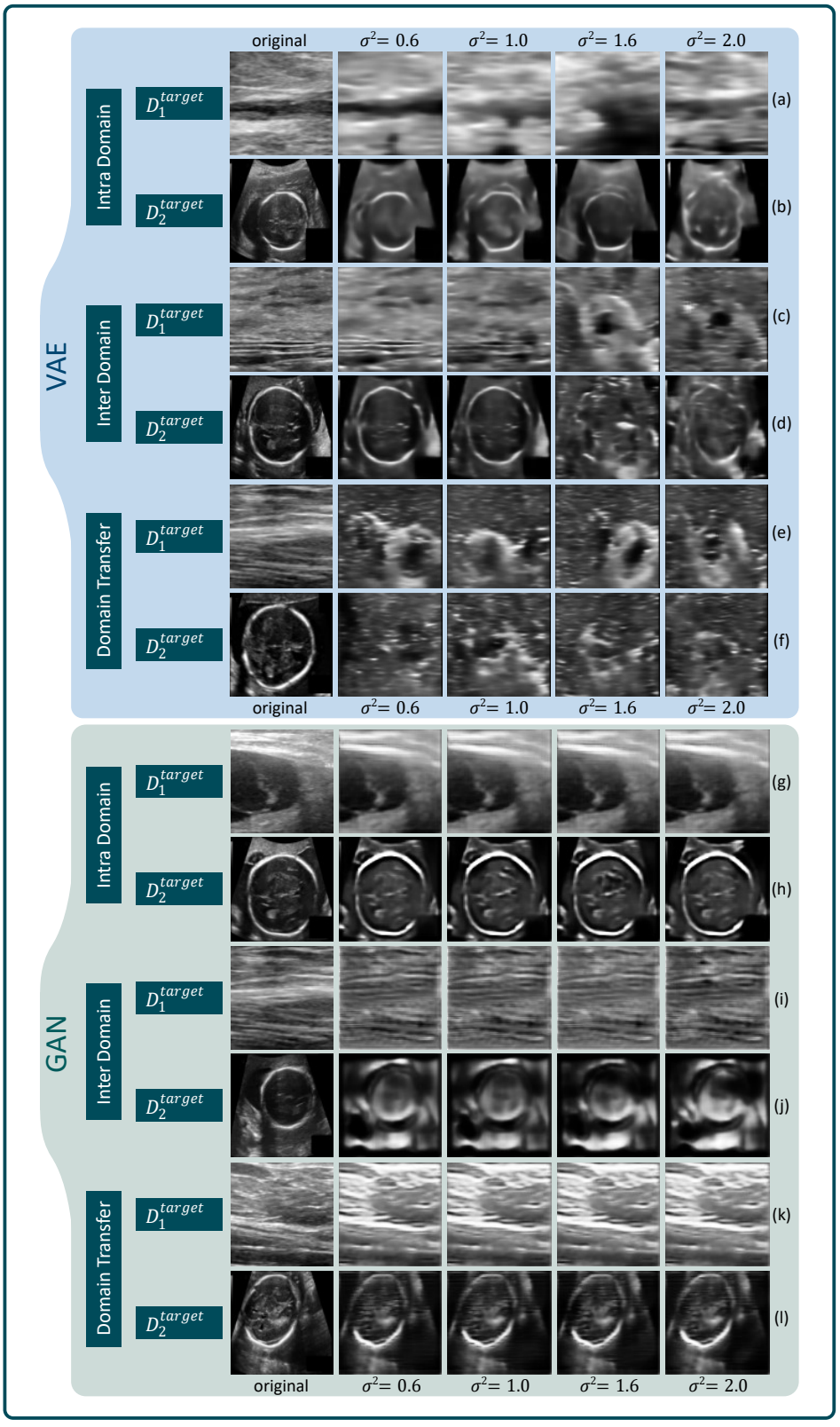

Figure 5.4: Examples for the augmentations generated in the *intra domain*, *inter domain* and *domain transfer* experiments for the target data sets $\mathcal{D}_1^{target}$ (breast) and $\mathcal{D}_2^{target}$ (fetus) using VAE (top) and GAN (bottom) at different σ^2 values.

Other than the VAE, the GAN tries to generate an augmentation that is similar to the origin so that it is not possible to get source data augmentations as observed when using the VAE.

Domain Transfer

In the *domain transfer* experiment, the VAE was not capable to generate augmentations of any target data set. Instead, at all σ^2 values the model generated augmentations of the source data set (rows (e) and (f) in Figure 5.4). As the VAE learned a latent representation space representing variations of the source data set, using target data for augmentation equates to sampling randomly in representation space. Thus, the VAE is not suitable for *domain transfer* if training data only contains one domain. Training data should rather contain various domains to make the model capable of generalizing over several domains. In contrast, the GAN was able to generate realistic augmentations of the origin for both target data sets. In breast augmentations the image contrast changes on a small scale. In addition, in some breast augmentations new structures appeared (row (k) in Figure 5.4) which is promising in terms of augmenting meaningful synthetic images. The fetus augmentations generated by the GAN appeared similar to the origin, but the images were strongly blurred (row (l) in Figure 5.4). The changes in breast augmentations were clearly higher than in fetus augmentations which again could be explained by the higher similarity between phantom and breast data compared to phantom and fetus data.
The results indicate that both models have the potential to become generalized. However, for optimal results, training should be performed on a larger and more diverse data set. Though, promising behavior as new structure appearance and structure deformation could be observed in this study. The results in the *inter domain* and *domain transfer* experiments indicate that it might be possible to use the proposed methods for augmenting an unknown data set from another domain only by performing fine tuning or even without any further model training. These are promising results as ultrasound data sets often are small and common augmentation techniques can lead to unrealistic ultrasound images not supporting deep learning downstream tasks [137], [138].

5.2 Deformable Ultrasound Augmentation

As could be seen in section 5.1, generating fully synthesized data samples is challenging as a large amount of data with a high level of variance is required to train a model that is able to create realistic synthetic data samples. Furthermore, as known from literature, simple transformation-based augmentation methods are easy to apply, but can result in unrealistic ultrasound image data [137]. For this reason, a transformation augmentation method is developed that aims to perform deformable (elastic) transformations to 3D ultrasound patches, which combines the simplicity of a transformation method and the complexity of a learning-based generation method.. The patch-based approach is chosen due to the fact that this work inves-

Table 5.2: The training, validation and test data sets prepared for training the CVAE and performing the evaluation experiments described in section 5.2.3. The data sets consist of 3D ultrasound patch pairs that were cropped from the sequences L-Vi with $i \in \{1, 2, 3, 4, 5\}$ and that were rigidly aligned.

Usage	Notation	Size	Origin
Training	$\mathcal{T}$	12,648	Training segments of L-Vi, $i \in \{1, 2, 3, 4, 5\}$
Validation	$\mathcal{V}$	1,116	Not-labeled images of L-Vi, $i \in \{1, 2, 3, 4, 5\}$
Test	$\mathcal{E}$	1,116	labeled images of L-Vi-1, $i \in \{1, 2, 3, 4\}$

tigates feature- and patch-based tracking algorithms in 4D ultrasound. Therefore, the proposed augmentation method can directly be used for data augmentation in a neural network training procedure in this work.
The main motion liver ultrasound sequences contain is due to respiratory movements [92]. Hence, realistic data augmentation could be achieved by being able to deform ultrasound image data in the same way as breathing-induced motion does. For this, however, the scope of potential motion must be known. This is done by using a conditional variational autoencoder (CVAE) that is trained as a deformable registration model to learn a representation space that contains the range of allowed deformation fields derived from 3D ultrasound sequences[3]. Krebs et al. [152] have shown that a CVAE can be used for CT image registration and claim that the model could be used for augmentation. However, they did not investigate the augmentation capability. In the following, the CVAE architecture, the training procedure and the evaluation experiments are presented.

5.2.1 Ultrasound Training Data

The proposed augmentation method is applied on 3D ultrasound patches with a size of $32 \cdot 32 \cdot 32$ vx and $N = 32{,}768$ voxels. For model training, validation and test the 3D ultrasound sequences L-Vi with $i \in \{1, 2, 3, 4, 5\}$ were used to prepare a data set that is summarized in Table 5.2. The training segments and the remaining unlabeled ultrasound images were used for generating a training set $\mathcal{T}$ and a validation set $\mathcal{V}$, respectively. The labeled sequences L-Vi-1 with $i \in \{1, 2, 3, 4\}$ were used for creating a test set $\mathcal{E}$ that is used for determining the augmentation performance. From the ultrasound sequences, 14,880 patch pairs were randomly cropped from different ultrasound images, where 12,648 (85 %) patches come from the training segment and 1,116 (7.5 %) patches each come from the validation and test parts. When generating the pairs, it was ensured that the patches are not coming from consecutive ultrasound frames to consider a wide range of deformations. To ensure that the pairs correspond in terms of translation and rotation, a rigid registration was applied to the 3D ultrasound images beforehand using the *SimpleITK* module [153]. The rigid registration is based on a maximization of the mutual information between the image pairs. Thus, the patch pairs only differ in terms of deformation which is mainly caused by breathing-induced motion.

[3]The 3D ultrasound augmentation approach presented here was published in [147]

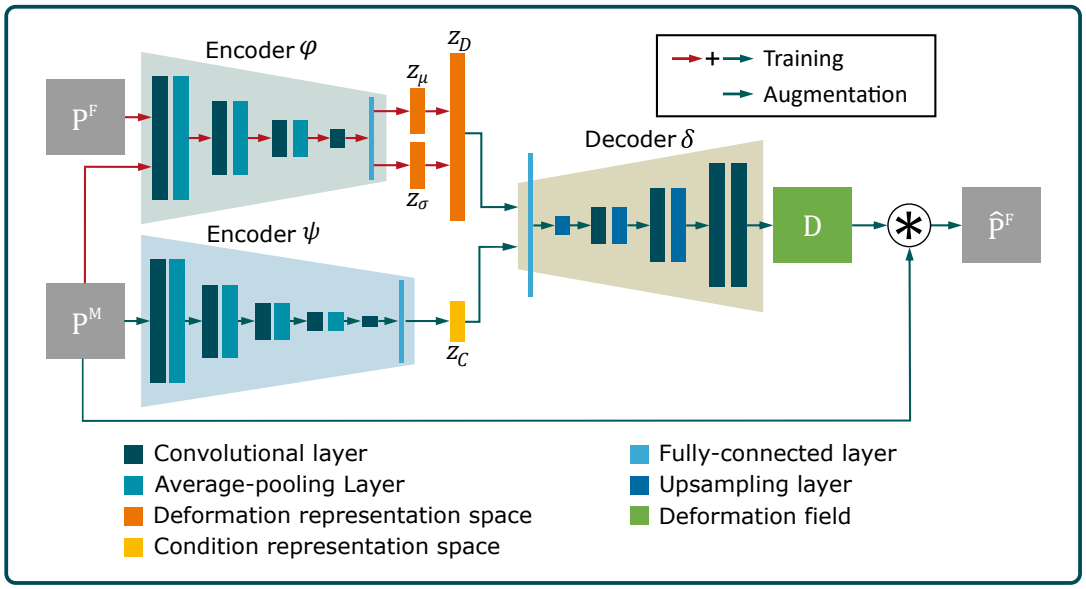

Figure 5.5: Architecture of the CVAE model. Training is performed using the variational encoder φ, the encoder ψ as well as the decoder δ. For augmentation, φ is omitted and a random vector $\mathbf{z}_D \sim \mathcal{N}(0,1)$ is given into the model.

5.2.2 Conditional Variational Autoencoder

To learn the scope of breathing-induced deformation patterns from the 3D ultrasound data, a deep neural registration network is used that is implemented as a CVAE. The architecture is illustrated in Figure 5.5. The CVAE consists of two encoders φ and ψ as well as one decoder δ. The CVAE gets a moving ultrasound patch $\mathbf{P}^M \in \mathbb{G}^{32\times32\times32}$ and a fixed ultrasound patch $\mathbf{P}^F \in \mathbb{G}^{32\times32\times32}$ as input and returns a deformation field $\mathbf{D} \in \mathbb{R}^{32\times32\times32\times3}$ so that $\mathbf{P}^F = \mathbf{D} \circ \mathbf{P}^M$ where $\circ$ denotes the warping operation. The registration encoder $\varphi(\mathbf{P}^M, \mathbf{P}^F)$ gets both ultrasound patches and maps them into a latent representation vector $\mathbf{z}_D \in \mathbb{R}^{256}$ which encodes the deformation between $\mathbf{P}^F$ and $\mathbf{P}^M$. This latent size is determined empirically in preliminary tests. The encoder $\varphi(\mathbf{P}^M, \mathbf{P}^F)$ is implemented as a variational encoder to ensure a generative capability [43]. Thus, the representation vector is a sample from a normal distribution $\mathbf{z}_D \sim \mathcal{N}(\mathbf{z}_\mu, \mathbf{z}_\sigma^2)$. The condition encoder $\psi(\mathbf{P}^M)$ maps the moving ultrasound patch $\mathbf{P}^M$ into a latent representation vector $\mathbf{z}_C \in \mathbb{R}^2$. This encoder is used to create a condition that describes the content of the ultrasound patch $\mathbf{P}^M$ so that the deformation representation encoding $\mathbf{z}_D$ is learned depending on the condition $\mathbf{z}_C$. Hence, it is ensured that the deformation extracted from an ultrasound patch pair depends on the underlying anatomical structure. This prevents a structure from being deformed in an unrealistic way when performing augmentation. After encoding, the representation vectors $\mathbf{z}_D$ and $\mathbf{z}_C$ are fed into the decoder to get the deformation field $\mathbf{D} = \delta(\mathbf{z}_D, \mathbf{z}_C)$. Subsequently, the deformation field $\mathbf{D}$ is applied to the moving patch $\mathbf{P}^M$ using the *VoxelMorph* framework [154] to arrive at the predicted patch $\hat{\mathbf{P}}^F$.

The CVAE is trained in an unsupervised way using the loss function given by

$$\begin{aligned} L(\mathbf{P}^F, \hat{\mathbf{P}}^F, \mathbf{D}, \mathbf{P}^M) = & L_{MSE}(\mathbf{P}^F, \hat{\mathbf{P}}^F) \\ & + \lambda_L \cdot L_{Laplace}(\mathbf{D}) \\ & + \lambda_{KL} \cdot L_{KL}(\varphi(\mathbf{P}^M, \mathbf{P}^F)) \end{aligned} \tag{5.7}$$

where $\lambda_L = 10^{-3}$ and $\lambda_{KL} = 10^{-5}$ were empirically determined weighting parameters and L_{MSE} denotes the MSE loss determined between the fixed ultrasound patch $\mathbf{P}^F$ and the prediction $\hat{\mathbf{P}}^F$. To ensure smooth and realistic deformation fields without any disruptions, a regularization loss is applied to $\mathbf{D}$ given by

$$L_{Laplace}(\mathbf{D}) = \frac{1}{N} \sum_{i=1}^{N} \Delta D_i \tag{5.8}$$

where Δ denotes the Laplace operator [155]. Since a variational autoencoder is implemented, the Kullback-Leibler loss $L_{KL}(\varphi(\mathbf{P}^M, \mathbf{P}^F))$ is also applied. Training was performed using the Adam optimizer, with a batch size of 128 and a learning rate of 10^{-4} until the validation loss stagnates. The CVAE was trained using the training data set $\mathcal{T}$ and the validation set $\mathcal{V}$.
When performing augmentation, only the encoder ψ and the decoder δ are required. For a 3D ultrasound patch sample $\mathbf{S}$, the condition vector $\mathbf{z}_C^{\mathbf{S}} = \psi(\mathbf{S})$ is determined, a random deformation vector $\mathbf{z}_D^{\mathbf{S}} \sim \mathcal{N}(0, 1)$ is generated and both are fed into the decoder to obtain a valid deformation field $\mathbf{D}^{\mathbf{S}} = \delta(\mathbf{z}_D^{\mathbf{S}}, \mathbf{z}_C^{\mathbf{S}})$. An augmented ultrasound patch is generated by applying the deformation field to the patch sample: $\mathbf{S}^{\text{aug}} = \mathbf{D}^{\mathbf{S}} \circ \mathbf{S}$.
For evaluating the registration performance, a baseline VAE was trained using the same training procedure and data but the encoder ψ is omitted. The VAE consists of an encoder φ and a decoder δ and is trained using the loss

$$L(\mathbf{P}^F, \hat{\mathbf{P}}^F, \mathbf{P}^M) = L_{MSE}(\mathbf{P}^F, \hat{\mathbf{P}}^F) + \lambda_{KL} \cdot L_{KL}(\varphi(\mathbf{P}^M, \mathbf{P}^F)). \tag{5.9}$$

The source codes for model training and testing as well as the trained VAE and CVAE models were made publicly available[4].

5.2.3 Augmentation Quality Study

The performance of the CVAE for the purpose of data augmentation is evaluated in three experiments that are presented in the following sections. Here, the CVAE's ability to register two 3D ultrasound patches and then realistically augment a 3D ultrasound patch is evaluated and compared to the VAE's performance.

Registration

The registration performances of the trained autoencoders are evaluated using the test data set $\mathcal{E}$ and the NCC (see equation 3.6) to quantify the ultrasound patch similarity. The results are presented in Table 5.3. Compared to the no registration

[4] `https://gitlab.rob.uni-luebeck.de/robPublic/3dus-deformable-augmentation`

Table 5.3: Results of the registration experiment using the test data set $\mathcal{E}$ containing 1,116 3D ultrasound patch pairs.

Registration type	**NCC (Mean ± SD)**	**Improvement of NCC (%)**
No Registration	0.57 ± 0.25	-
VAE	0.69 ± 0.24	21
CVAE	0.70 ± 0.23	23

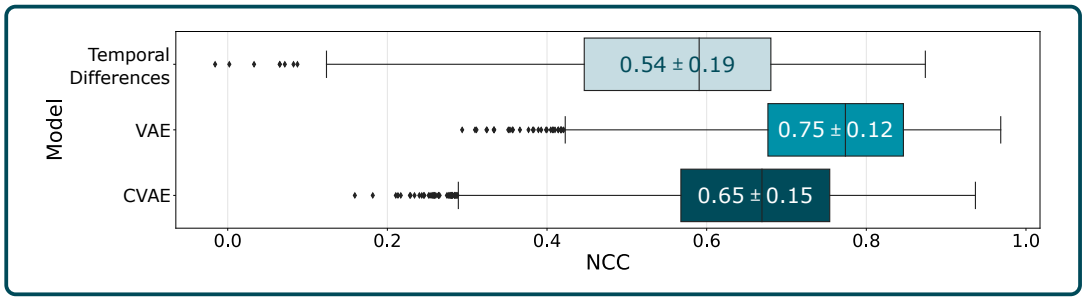

Figure 5.6: Range of NCC between augmented patches generated using VAE and CVAE as well as between time-dependent realistic patches (Temporal differences). Five targets are evaluated with an average of 50 patches with temporal differences each and 400 augmented patches per method.

case, by using the VAE and CVAE models the image similarity could be improved by 21 % and 23 %, respectively. Thus, both models were able to register two 3D ultrasound patches.

Augmentation

For evaluating the augmentation performance, 400 augmented patches each were generated using the VAE and the CVAE for five different ultrasound patches. For this, patches were cropped from the labeled landmark positions of the first image from the sequences L-Vi-1 with $i \in \{1, 2, 3, 4, 5\}$. The amount of change observed in the ultrasound patch after applying the deformation field was measured by comparing the augmented and original ultrasound patches using NCC. In addition, a realistic time-dependent ultrasound patch NCC range was identified representing the temporal differences in the target shape. This was done based on patches cropped from the remaining labeled landmark positions in the mentioned sequences. For each patch in a sequence, the NCC was determined compared to the first patch of the sequence. The results are illustrated in Figure 5.6. The mean NCC is 0.75 ± 0.12 and 0.65 ± 0.15 for the VAE and CVAE, respectively. The mean realistic time-dependent ultrasound patch NCC range is 0.54 ± 0.19. It turned out that augmented patches generated by the VAE are 38.9 % more similar to each other than to realistic ultrasound patches. In contrast, the mean NCC of CVAE augmented patches, is only 20.4 % higher than the mean NCC of realistic ultrasound patches which indicate that the CVAE is able to generate more realistic augmentations than the VAE when considering ultrasound patch similarity. The mean NCC of CVAE augmented patches is 13.3 % lower than VAE augmented patches which

shows that the CVAE is able to perform augmentations at a higher magnitude than the VAE. Furthermore, the level of deformation magnitude applied by the CVAE is 18.5 percentage points closer to realistic deformations compared to the VAE. This indicates that the range of deformation of CVAE augmented patches is more similar to realistic changes caused by physiological motion due to breathing than VAE augmented patches. This finding is promising for realistically enlarging a data set for deep learning purposes. The model architectures of the CVAE and VAE differ in the second encoder ψ of the CVAE which is used to create a condition depending on the anatomical structure. Thus, learning the deformation range depending on the structure type shown in the ultrasound patch is beneficial for realistic augmentation.

5.2.4 Expert Study

To evaluate the level of realism of the augmentations, an expert study was conducted where six experienced radiologists (4.3 years of experience on average) were asked to classify ultrasound patches into the classes *real*, *unsure* or *fake*. This rating scale was chosen according to recommendation by the radiologists as it is a common rating scale for them. The experts were provided with a set of 75 different ultrasound patches with a size of $32 \cdot 32 \cdot 32$ vx containing realistic patches as well as CVAE and random augmented patches (25 each). Random augmented patches were generated by sampling the deformation field from a normal distribution $\mathbf{D} \sim \mathcal{N}(0, 1)$, applying it to an ultrasound patch $\mathbf{S}^{\text{aug}} = \mathbf{D}^{\mathbf{S}} \circ \mathbf{S}$ and filtering the augmentation with a Gaussian kernel ($\sigma^2 = 1$) to reduce additional noise. The magnitude of random augmentation needed to be chosen smaller than CVAE augmentation since random deformation does not underlie any rules and thus, the patches are easily identifiable if the magnitude is as large as in CVAE augmentation. For each ultrasound patch, the experts had to decide whether it is augmented in any way (*fake*) or whether it is an original ultrasound patch (*real*). In cases where they could not decide for *real* or *fake*, they had the possibility to choose the class *unsure*. Since rating ultrasound patches instead of whole volumes is an unfamiliar task for the experts, four volumes were provided additionally to facilitate general orientation. The data set used in the expert study contained manually selected ultrasound patches to ensure that the patches show identifiable anatomical structures like vessels. The mean NCC between the augmented and the original patches is 0.72 ± 0.12 and 0.96 ± 0.19 for CVAE and random augmentations, respectively. In Figure 5.7, examples for a CVAE as well as a random augmentation and the original patch are illustrated. It can be seen that the random augmentation contains additional noise caused by the random deformation field while the CVAE augmentation only smoothly deforms the structure. For this reason random augmented patches were smoothed after applying the deformation. The results of the expert study are summarized in Figure 5.8. Real ultrasound patches are rated 44.7 % *real*, CVAE and random augmented patches are rated 50.0 % and 42.0 % *real*, respectively. This indicates that CVAE augmented patches predominantly appeared realistic to the experts. In contrast, the results show that random augmented patches appeared less realistic for the experts as they are rated 46.7 % *fake* which is 13.4 percentage points more than CVAE augmentations. For better comparison, in Figure 5.8, the classification ratios of CVAE

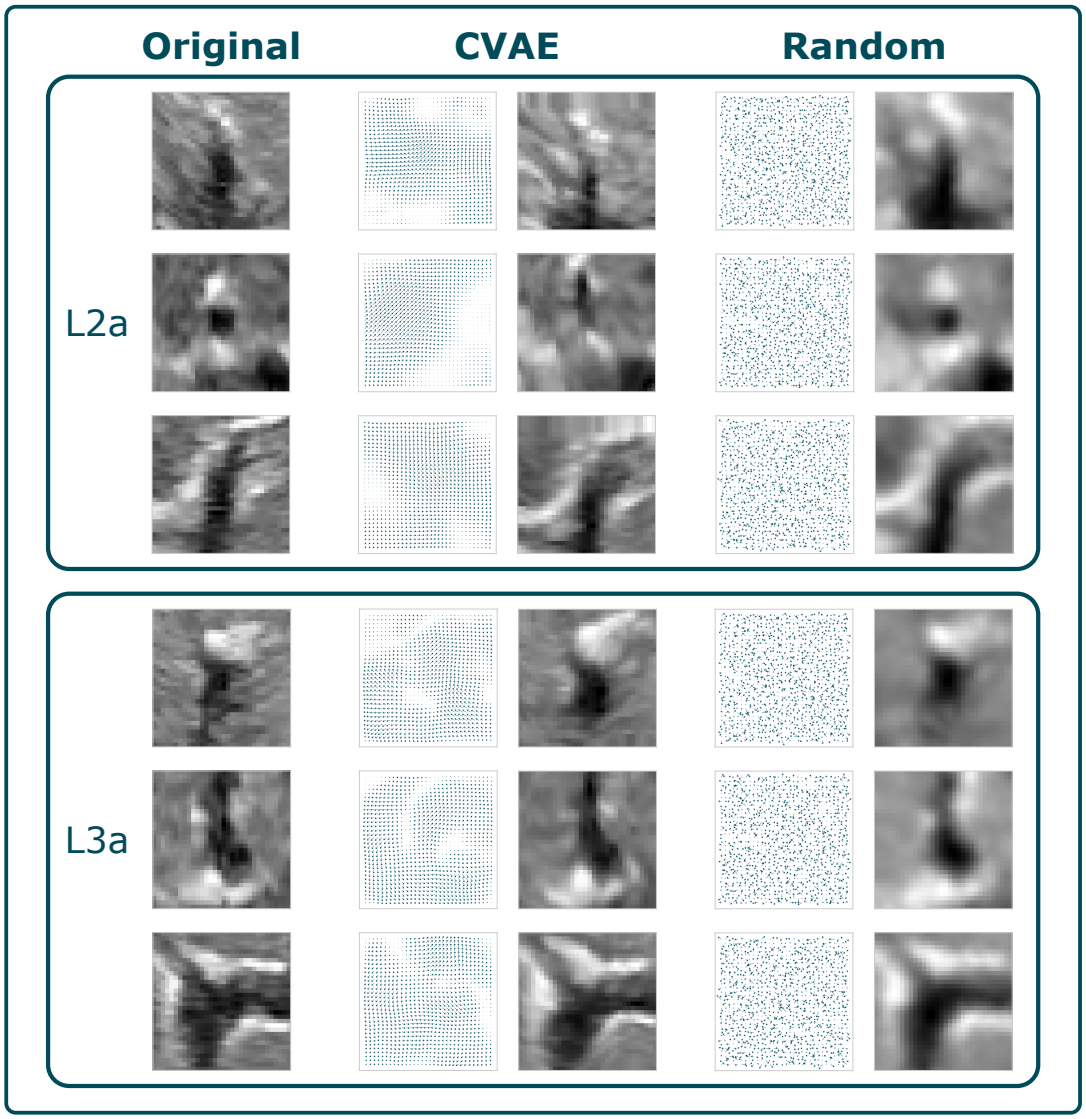

Figure 5.7: Two examples (landmarks L2a and L3a) for a 3D ultrasound patch (left) augmented using the CVAE (middle) and a random deformation field (right). The used deformation fields as well as the resulting augmented patches are visualized by showing three orthogonal slices. Random augmented patches are filtered by using a Gaussian kernel after deformation to reduce additional noise.

augmented patches are grouped into the three NCC ranges **A** (NCC $\in [0.8, 1]$), **B** (NCC $\in [0.6, 0.79]$) and **C** (NCC $\in [0.4, 0.59]$), where the NCC is determined between the augmented and the original patches. All random augmented patches lie in the NCC range **A**. Although the augmentation magnitude is the lowest in this NCC range, 58.0 % of the random augmented patches were not classified *real*. In the same range, 59.0 % of the CVAE augmented patches were rated *real*. Even in the NCC range **B**, 50.0 % of CVAE augmented patches were rated *real*, which is 8 percentage points higher than random augmented patches.
The results of the expert study show that the experts could not differentiate between real and CVAE augmented patches as the ratios of *real* ratings are comparable. In contrast, the experts could identify random augmented patches more reliably even

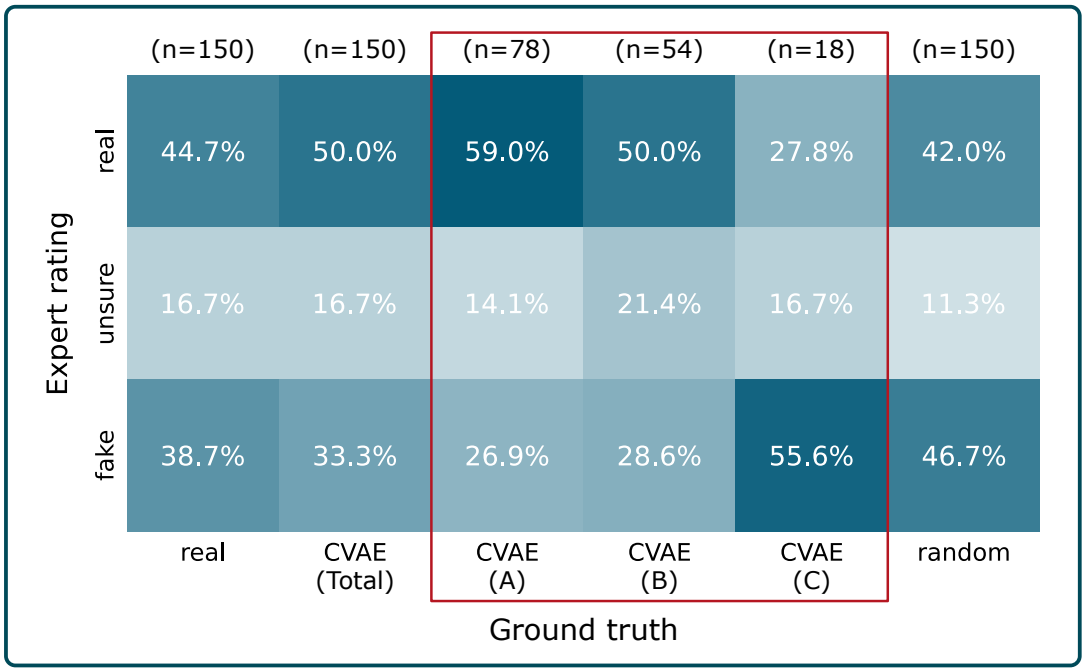

Figure 5.8: Confusion matrix for the results of the expert ratings into the classes *real*, *unsure* and *fake*. The CVAE results are additionally grouped by the deformation magnitude into the groups **A** $\equiv$ NCC $\in [0.8, 1]$, **B** $\equiv$ NCC $\in [0.6, 0.79]$ and **C** $\equiv$ NCC $\in [0.4, 0.59]$ marked by the red box.

though the deformation range is 25 % smaller than CVAE augmented patches. It turned out, that the lower the deformation, the more false *real* ratings could be observed. However, randomly augmented patches could predominantly be identified correctly which indicates that using random deformation fields leads to potentially unrealistic ultrasound patches. CVAE augmented patches, turned out to be realistic as long as the deformation magnitude is not too high. Thus, a balance between deformation magnitude and level of realism must be found as the magnitude should be as high as possible to generate a high data variance, but simultaneously, the data should be as realistic as possible. A limitation of the expert study is the fact that the experts had to rate 3D ultrasound patches instead of whole ultrasound images. The experts pointed out that they are unfamiliar with ultrasound patches as only a very limited anatomical area is shown. Although whole ultrasound volumes were also provided for general orientation, the task was challenging for the experts which could be a reason for the 38.7 % ratio of false *fake* ratings of real ultrasound patches.

5.2.5 Application Study

The results achieved in the augmentation quality and expert studies indicate that the CVAE is able to augment 3D ultrasound patches in a realistic way. In order to evaluate the usability of the proposed augmentation method in a deep learning downstream task, target detection in 3D ultrasound was performed. The task was to locate the position of an ultrasound target patch within a ROI.

Table 5.4: Results of the 5-fold cross-validation target detection experiment including 1,912 ROI patch pairs per fold. The error corresponds to the Euclidean distance between the ground truth and the predicted target location. In addition, the motion amplitude of the ROI patch pairs is given in *No tracking* created by adding an offset of ±20 vx per image axis. The best results are highlighted in bold.

Augmentation method	**Error (vx)**			**Mean improvement (%)**
	Mean ± SD	**95%-tile**	**Max**	
No tracking	17.32 ± 10.00	32.91	34.64	-
None	2.40 ± 1.24	4.75	15.33	-
Random	2.26 ± 1.10	4.30	12.93	7
CVAE	$\mathbf{1.43 \pm 0.68}$	**2.67**	7.73	**41**
Rotation	1.75 ± 0.88	3.37	11.26	28
CVAE+Rotation	$\mathbf{1.43} \pm 0.70$	2.72	**7.65**	**41**

Neural Network Training

For creating a data set, one labeled landmark each from the sequences L-Vi-1 with $i \in \{1, 2, 3, 4, 5\}$ was used to generate 9,560 pairs of ROIs of the size $48 \cdot 48 \cdot 48$ vx and patches of the size $32 \cdot 32 \cdot 32$ vx. A pair consisted of a ROI and a patch containing the same structure but from different ultrasound images. Thus, the targets in the ROI and the patch differed in shape and appearance due to breathing-induced motion. Additionally, a random shift of ±20 vx per axis was added to the target structure position to ensure the target was located at a random position within the ROI.
A convolutional neural network (CNN) was trained in a supervised way to predict the location of the given target. The model consists of two encoders (for ROI and target patches) containing ResNet-blocks for image feature extraction and a fully-connected block for the target location prediction. For training, the Adam optimizer, a batch size of 64, a learning rate of 10^{-5} and early stopping with a validation split of 20 % for avoiding overfitting was used. The CNN was trained using five different modalities. First, training was performed on the basis of the raw generated data set without data augmentation to generate a baseline. Furthermore, training was performed adding augmented ultrasound patches using random deformation, CVAE deformation, or rotation. Additionally, a CNN was trained where both CVAE deformation and rotation were used. When using augmentation, each target patch was augmented once so that the size of the data set was doubled to 19,120 pairs. In the CVAE plus rotation case, the data set size was 28,680.

Localization Experiments

A 5-fold cross-validation experiment was conducted where in each fold a subset of 1,912 ROI patch pairs was omitted and used for testing. The results of the 5-fold cross-validation experiment are presented in Table 5.4. For reference, the *no tracking* case is given as well representing the motion amplitude of the target within the ROI generated by adding a random offset of ±20 vx per image axis. The error was measured by determining the Euclidean distance between the ground truth and the predicted target location. In the test sets, only original ultrasound patches were considered to measure the performance of the CNN on real data. The results show

that all tested augmentation techniques improved the target detection performance of the CNN. Using random augmentation led to an improvement of 0.17 vx (7 %) compared to no augmentation case. However, performing a t-test showed that this improvement is not significant with p-value = 0.95. The CNN trained on rotational augmented ultrasound patches achieved a significant improvement of 0.68 vx (28 %, p-value $\ll$ 0.001) and using CVAE augmentation improved the performance significantly by 1.00 vx (41 %, p-value $\ll$ 0.001). The CNN trained on both CVAE and rotational augmentation could not improve the performance further as it also led to an improvement of 1.00 vx (41 %, p-value $\ll$ 0.001) compared to the no augmentation case. Comparing the results achieved by using CVAE and CVAE plus rotation showed no significant difference in performance with p-value = 0.61.
The results show that applying the proposed augmentation method in a target detection application improved its performance by 41 % compared to not using data augmentation. Furthermore, using CVAE augmentation outperformed the use of random deformation as well as rotation. This indicates that the proposed CVAE is a realistic deformation augmentation method that is promising for a target detection task in 3D ultrasound as the realistic breathing-induced motion pattern can be simulated. Another benefit of this transformation augmentation approach is its capability of label preserving as the applied deformation field is known. In the approach by Al Dhabyani et al. [137], e.g., fully synthesized ultrasound images were generated so that the resulting augmentations can contain label-violating structures. Similar to the findings in this study, they also found that deep learning-based augmentation methods lead to better performance in a downstream task compared to geometric augmentation techniques such as rotation.

5.3 Summary

Data augmentation is known to be beneficial for enlarging small data sets in order to train robust deep neural networks [135]. In the ultrasound image domain, data sets are often small and even sparsely available. It is shown that traditional augmentation techniques such as rotation, or flipping can lead to unrealistic ultrasound images and even be disadvantageous for training deep neural networks for a downstream task such as target tracking [137]. Since there is a lack of generalized augmentation methods for the ultrasound image domain, this chapter addressed

Research Question 3
Is it possible to perform realistic 3D ultrasound image augmentation using unsupervised deep representation learning and is this augmentation beneficial for a target tracking application?

Two unsupervised deep representation learning-based augmentation approaches were investigated in this work. A data generation- as well as a transformation-based approach were considered. Based on available information, the first case study on domain generalized ultrasound image augmentation has been conducted. It was shown that a data generation technique using a VAE or a GAN is able to synthesize ultrasound images even containing new structures. The results indicate that a GAN

is more effective at domain transfer, but less effective at making high-level data changes compared to a VAE. However, the study presented in this work is a first case study limited to three anatomical domains and to small data sets. In future studies, the amount of anatomical domains and data sets should be increased.
In addition, a novel data transformation technique for deformable 3D ultrasound patch augmentation was proposed. A CVAE was used to learn a latent representation space that contains abstract representations of realistic motion fields extracted from realistic breathing-induced motion patterns. In contrast to Tirindelli et al. [143] who focused on deformable motion caused by different ultrasound transducer forces and 2D ultrasound images, the first study on breathing-induced motion and 3D ultrasound image patches was conducted in this work. It could be shown that clinical experts could not differentiate between real and CVAE augmented 3D ultrasound patches. Furthermore, the performance of a deep neural network performing target localization in 3D ultrasound ROIs could significantly (p-value $\ll 0.001$) be improved by using the proposed augmentation technique.
The findings achieved in this chapter can be used to answer the research question: A deep-learning based realistic deformable 3D ultrasound image augmentation method was proposed that was shown to be able to improve the target localization accuracy of a neural network. Even though generating fully new ultrasound images is still challenging and needs further research, it is already possible to perform realistic ultrasound transformation, e.g., by using the proposed CVAE. This data transformation approach was shown to be potentially beneficial for training a deep neural network for a downstream task.

6 REAL-TIME TARGET TRACKING

In the domain of liver ultrasound imaging, different target tracking approaches have been proposed in literature [14]. Many of these approaches were developed, trained and tested with the CLUST data set that was described in section 3.3.1. However, as mentioned earlier, most of these approaches were proposed for 2D+t ultrasound sequence tracking considering only 2D images sliced from 3D structures. On the CLUST website[1], 22 participants are reported for 2D+t and 7 are reported for 4D ultrasound tracking. This shows there is a lack of 4D ultrasound tracking research although 3D imaging is required for observing motion with six DoF or even more when considering deformation. When using 2D ultrasound imaging, out-of-plane motion cannot be noticed limiting the observable motion dimensionality, e.g., from six DoF (3D) to three DoF (2D) in a rigid scenario. This makes 2D ultrasound imaging insufficient for the purpose of robust and precise target tracking in the domain of medical interventions such as radiation therapy guidance. The average tracking error measured in the evaluation of the tracking approaches listed on the CLUST website are in a range [0.69, 3.35] mm for 2D+t and [1.70, 4.90] mm for 4D ultrasound tracking. However, the list is not complete as other studies, such as by Huang et al. [15] who performed 4D ultrasound tracking and measured an average tracking error of 1.66 mm, are not listed. The tracking error (TE) is defined as the difference between the tracked target position within an ultrasound image and the ground truth. According to De Luca et al. [14], the TE between a predicted position $\hat{\mathbf{p}}^t$ and a ground truth position $\mathbf{p}^t$ at time step t is determined using the Euclidean distance $\|\cdot\|_2$ by

$$\mathrm{TE}^t(\hat{\mathbf{p}}, \mathbf{p}) = \|\hat{\mathbf{p}}^t - \mathbf{p}^t\|_2. \tag{6.1}$$

To determine an average TE for a whole ultrasound sequence containing T images, the mean and standard deviation (SD) is calculated based on all TE^t with $t \in \{1, ..., T\}$ where a ground truth exists.
Target tracking methods can be categorized based on the information they use to determine the target position in intensity- and feature-based methods. intensity-based methods use the pixel (2D image) or voxel (3D image) intensity information to predict the target position. Such a method is used, e.g., by Banerjee et al. [17],

[1] `https://clust.ethz.ch/results.html`

[156] who proposed a template matching-approach to perform target tracking in 4D ultrasound. Another example is the approach proposed by Royer et al. [18] who used an deformable transformation model for the target that is optimized based on voxel intensity values of the ultrasound image. In contrast, in feature-based approaches the target position prediction relies on features extracted from the ultrasound image. For instance, Hallack et al. [157] used the SIFT feature detector and descriptor to extract and match features in consecutive ultrasound images to perform 2D+t ultrasound tracking. There is a lack of research on feature-based ultrasound tracking methods, especially for 4D ultrasound, as most of the approaches proposed in literature are intensity-based approaches.
In this chapter, feature-based target tracking approaches are developed where the tracking procedure is performed in a representation space generated by an autoencoder. Therefore, ultrasound patches are cropped from the ultrasound images and mapped into the representation space. Using a patch-based approach for 3D image data has the advantage that it is potentially fast and robust whereas processing complete 3D images can be computationally expensive [158]. In addition, the amount of available data is increased due to the fact that multiple patches can be generated from one ultrasound image. The approaches are tested and evaluated based on labeled 3D ultrasound sequences. The objective of this chapter is to investigate

Research Question 4
Can real-time target tracking in 4D ultrasound be performed in representation space and can it outperform image space-based tracking algorithms?

Therefor, besides approaches for representation space-based target tracking, also simple intensity-based tracking approaches are implemented to create an image space-based baseline[2].

[2]Parts of this chapter were published in:

[110] D. Wulff and F. Ernst, "Feature description using autoencoders for fast 3D ultrasound tracking," Current Directions in Biomedical Engineering, vol. 10, no. 2, pp. 21-24, 2024.

[134] D. Wulff, J. Hagenah, and F. Ernst, "Landmark tracking in 4D ultrasound using generalized representation learning," International Journal of Computer Assisted Radiology and Surgery, vol. 18, pp. 493–500, 2023.

[159] C. Krause, D. Wulff, and F. Ernst, "Landmark tracking in 4D ultrasound using localization networks," Current Directions in Biomedical Engineering, vol. 10, no. 2, pp. 29-32, 2024.

[160] D. Wulff, R. Sarau, and F. Ernst, "Target tracking in 4D ultrasound based on template matching and target forecasting using spatio-temporal autoencoders," 2023 IEEE 23th International Conference on Bioinformatics and Bioengineering (BIBE), pp. 113-120, 2023

and have been submitted to:

CIBM D. Wulff, F. Ernst, "Real-Time Deformable Structure Tracking in 3D Ultrasound Sequences using Deformable Convolutional Layers," Computers in Biology and Medicine, 2024, under review

6.1 Tracking in Image Space

Previous ultrasound tracking methods proposed in literature are mainly applied in image space [14], [91]. The main objective is to detect a target represented by a reference patch in a search image by using a similarity metric such as NCC or MSE. A simple method based on this approach is template matching which is explained in the following in more detail. Due to different drawbacks of such a simple tracking method like high runtime and the difficulty to define a proper similarity metric that is invariant to image changes and target motion, deep learning-based approaches have been investigated. The most common approach is to use a siamese neural network (SNN) that has two neural network branches that are fed with a target patch as well as a search image and that predicts the position of the target within the search image. Even though most of the methods proposed in literature are developed and evaluated in the 2D ultrasound image domain, Huang et al. [15] and He et al. [16] proposed SNNs for target tracking in 4D ultrasound.
In this section, two basic target tracking approaches are presented and evaluated in order to create a baseline. A template matching algorithm is implemented and used for performing target tracking in 4D ultrasound. To improve the performance of this simple algorithm, a deep representation learning-based approach is proposed that aims to predict the appearance of the target patch for the next time step. Following, a simple neural network is proposed that aims to locate a target in a 3D ultrasound image.

6.1.1 Template Matching

A naive target tracking approach in image space is template matching, also called block matching. Such an algorithm has the advantage that no model training, and thus, no labeled data is required. In addition, template matching is easy to implement and therefore, often a first approach to be tested when developing a tracking algorithm [161]. Even though template matching is commonly used in the RGB-image and -video domains [161], [162], it is also used in the medical ultrasound imaging domain. For instance, Koizumi et al. [163] proposed a template matching algorithm to track a target in the kidney in an invasive 2D ultrasound theragnostic system. In other studies, Carletti et al. [164] proposed a method combining template matching and a particle filter to track targets and De Luca et al. [165] investigated different template matching-based approaches for landmark tracking in 4D ultrasound. Template matching-based tracking has also be investigated in 4D ultrasound. Kondo et al. [166] performed template matching to track targets in the liver. In template matching, only the pattern to be found, a so called target reference patch and the search image or a defined ROI within the image are required [167]. The key idea is to pass through the search image $\mathbf{I}$ and locate the position $\hat{\mathbf{p}}$ where a similarity metric between the search image and the reference patch $\mathbf{P}$ maximizes by

$$\hat{\mathbf{p}} = \arg\max_{\mathbf{x}} d_{\text{Similarity}}(\mathbf{I}(\mathbf{x}), \mathbf{P}). \tag{6.2}$$

The template matching algorithm used in this work is presented in Algorithm 2. As can be seen, template matching is done in a limited ROI $\mathbf{R}_{\text{curr}}$ instead of the whole

Algorithm 2 Template Matching algorithm to find the target position $\hat{\mathbf{p}}^t$ in a ROI $\mathbf{R}_{\text{curr}}$ cropped at the position $\mathbf{s}$ in a 3D ultrasound image $\mathbf{I}_{\text{curr}}$ from a sequence $\mathbf{S}$ at time step t.

Input: $\mathbf{P}_{\text{ref}}, \mathbf{S}, \mathbf{s}$
Output: $\hat{\mathbf{p}}^t, t = 1$ to EndofSequence

1: $\mathbf{p}^0 = \mathbf{s}$
2: $\hat{\mathbf{p}}^{\text{seq}} = \emptyset$ ▷ Initialize empty predicted position sequence
3: **for** $t = 1$ to EndofSequence **do** ▷ Go through the Sequence $\mathbf{S}$
4: $\quad \mathbf{I}_{\text{curr}} = \mathbf{S}^t$
5: $\quad \mathbf{R}_{\text{curr}} = \text{cropROI}(\mathbf{I}_{\text{curr}}, \hat{\mathbf{p}}^{t-1})$
6: $\quad sim^t = 0$
7: $\quad$ **for** $\mathbf{x} = (0,0,0)^T$ to EndofROI **do** ▷ Slide through the ROI $\mathbf{R}_{\text{curr}}$
8: $\quad\quad \mathbf{P}_{\text{curr}} = \text{cropPatch}(\mathbf{I}_{\text{curr}}, \mathbf{x})$
9: $\quad\quad sim_{\text{curr}} = Similarity(\mathbf{P}_{\text{ref}}, \mathbf{P}_{\text{curr}})$
10: $\quad\quad$ **if** $sim_{\text{curr}} > sim^t$ **then**
11: $\quad\quad\quad sim^t = sim_{\text{curr}}$
12: $\quad\quad\quad \hat{\mathbf{p}}^t = \mathbf{x}$
13: $\quad\quad\quad \hat{\mathbf{p}}^{\text{seq}}.\text{append}(\hat{\mathbf{p}}^t)$ ▷ Collect the tracked target positions
14: $\quad\quad$ **end if**
15: $\quad$ **end for**
16: $\quad \mathbf{P}_{\text{ref}} = \text{cropPatch}\,(\mathbf{R}_{\text{curr}}, \mathbf{s})$ ▷ Update reference patch (If required only)
17: **end for**

image to reduce the computational costs.
Due to the simple approach, template matching also has some disadvantages. By using simple similarity metrics such as NCC or MSE, the algorithm is not invariant to changes like rotation and deformation. As shown in section 3.4, anatomical liver targets visualized in medical ultrasound images are subject to these types of changes. Especially, when the reference patch is taken from an ultrasound image that is temporally distant from the current ultrasound image, great changes to the target can occur. This can lead to the fact that the reference patch may not be sufficiently similar to the current target appearance to locate it. As a solution for this difficulty, the reference patch can be updated by cropping the reference patch from the previous ultrasound image at the located position as shown in line 14 in Algorithm 2. In the following, this approach is called *Updated Target Tracking* (UTT), and using a fixed reference patch is called *Fixed Target Tracking* (FTT)[3]. Furthermore, template matching is a time-consuming algorithm as the reference patch is slid through the whole image or ROI and the similarity metric must be determined at every position. Even though there are approaches to provide fast template matching algorithms in literature [168], [169], especially in large 3D image data, such as 3D ultrasound, the runtime of such a template matching algorithm is potentially high and not real-time capable.

[3]The template matching experiments presented here were published in [160].

Table 6.1: Tracking errors measured in the sequences L-Vi-3 with $i \in \{1, 2, 3, 4\}$ with the landmarks Lia and Lib using template matching algorithm with a fixed (FTT) and an updated (UTT) reference patch strategy. The best values are highlighted in bold.

Method	Tracking error (mm)			Mean TE range (mm)	landmarks (%) where	
	Mean ± SD	95%ile	Max		TE > 3 mm	TE > 5 mm
No tracking	9.67 ± 9.55	34.28	**42.15**	[4.18, 28.61]	81.32	61.75
FTT	**7.55** ± 12.31*	42.92	52.76	[**1.71**, 33.24]	**49.17**	**28.21**
UTT	8.00 ± **7.52**	**26.49**	43.81	[3.77, **17.95**]	87.67	60.23

*Only TE ≥ 0 was measured.

Tracking Experiments

In order to determine a baseline for this work, template matching was used to perform target tracking in the labeled 4D ultrasound data and evaluated using the landmarks as ground truth. For the template matching experiments, the sequences L-Vi-3 with $i \in \{1, 2, 3, 4\}$ and the landmarks Lia and Lib were used. For all landmarks, the reference patch $\mathbf{P}_{\text{ref}} \in \mathbb{G}^{24\times24\times24}$ was cropped from the first ultrasound image of the sequence at the labeled landmark position. This position served as the starting position for the template matching algorithm. Tracking was performed in the remaining images of the ultrasound sequence. To reduce the computational costs, template matching was performed for a ROI of the size $35 \cdot 35 \cdot 35$ vx.
The template matching algorithm presented in Algorithm 2 was implemented in *Python* and executed to the given ultrasound sequences. Two different experiments were conducted: FTT and UTT. The tracking accuracy was quantified by determining the TE between the labeled ground truth and the position found by the template matching algorithm.

Tracking Results

The results for the FTT and UTT experiments are presented in Table 6.1. High mean TEs of 7.55 ± 12.31 mm[4] and 8.00 ± 7.52 mm were measured in the FTT and UTT experiments, respectively, indicating that template matching is insufficient for robust target tracking in 4D ultrasound. Though, in FTT a low minimum mean TE of 1.71 mm was measured in sequence V3-1 tracking landmark L3a. Hence, for some sequences and landmarks FTT could achieve a low TE. However, this approach is not robust. As can be seen in the results in Table 6.1, high mean TE ranges of $[1.71, 33.24]$ mm and $[3.77, 17.95]$ mm, were observed and the TEs are higher than 3 mm in 49.17 % and 87.67 % of all ultrasound images in FTT and UTT, respectively.
It could be observed that these large tracking error ranges are caused by two different failure types. The first one occurred in the FTT algorithm. The TE alternated in correlation with the breathing-induced motion of the labeled landmark caused by a fixed reference target patch. This failure depicts a main challenge of target tracking in 4D ultrasound. Due to deformable motion patterns, the target appearance varies over time. The second one occurred in UTT. Here, the target got lost and could not be relocated in the following images so that the TE increases rapidly

[4]Only TE ≥ 0 was measured.

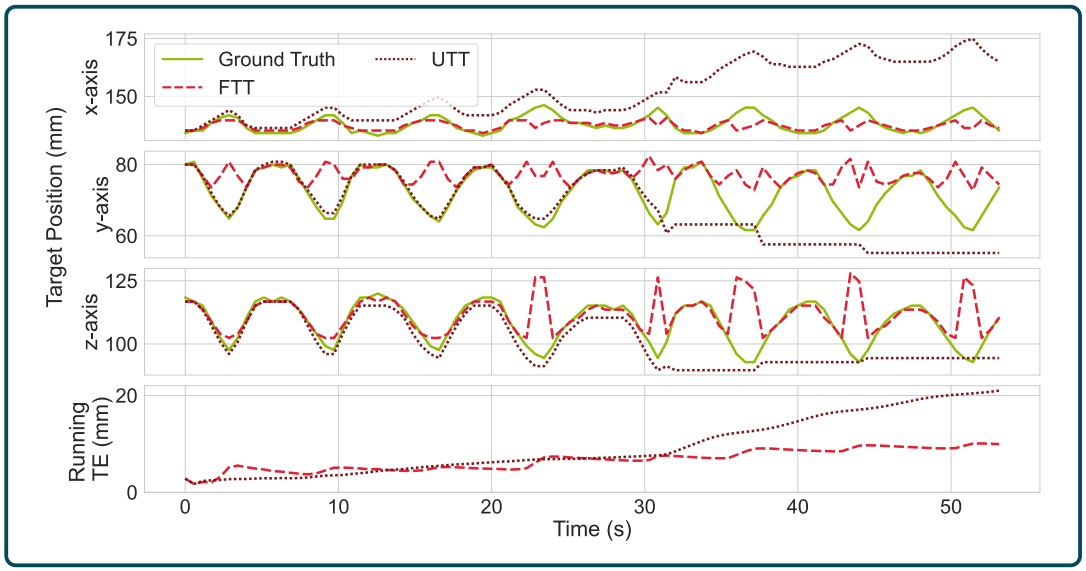

Figure 6.1: Examples for target tracking trajectories in x- (top), y- (second from the top) and z-axes (third from the top) measured after applying template matching using Fixed Target Tracking (FTT) and Updated Target Tracking (UTT) to the sequence L-V1-3 compared to the labeled landmark L1a trajectory. In addition, the running TE for FTT and UTT are visualized (bottom).

and stays high. This failure is caused by high target appearance variations between consecutive ultrasound images, so that the reference target is not sufficiently similar to the current target appearance. Both tracking failure types are illustrated in Figure 6.1 representing the ground truth landmark position as well as the tracked trajectories achieved by applying FTT and UTT template matching to the sequence L-V1-3 and landmark L1a. In FTT, the TE highly depends on the phase of the breathing-induced motion and in UTT, the TE increases with time. In this example, these tracking failures led to mean TEs of 8.37 ± 10.51 mm and 17.95 ± 14.72 mm in FTT and UTT, respectively. It can be seen that the TE in UTT increased rapidly after second 30, where the algorithm loses track of the target. In contrast, a step-wise increase in TE was observed in FTT, correlating with the temporal occurrence of tracking failure. De Luca et al. [165] reported similar issues when using template matching-based tracking algorithms in 4D ultrasound. They proposed to use a learning-based template matching approach to compensate for these failures by learning the relationship between the image appearance and the corresponding displacements from another more robust tracking method. In this work, a novel approach for compensating these failures based on deep representation learning is proposed and described in the following section.

6.1.2 Spatio-Temporal Autoencoder

Soft-tissue targets in the liver undergo high-dimensional motion caused by the respiratory process. This motion can be considered as quasi-periodic as the breathing process is periodic but the exact motion of structures differs from phase to phase in the breathing cycle, leading to even drastic changes in the target appearance over time. Hence, simple target tracking approaches such as template matching

can fail leading to total target loss or periodically alternating TE as shown in the previous section. Since an ultrasound sequence is time-resolved image data, target tracking approaches based on RNNs are proposed in literature in order to take time-dependent information into account. For instance, Rangamani et al. [170] proposed a supervised CNN-RNN framework for target tracking in 4D ultrasound consisting of an encoder-decoder structured CNN and an LSTM network placed in the bottleneck of the CNN. The network predicts the target position within a search image based on three previous ultrasound images. Another approach proposed by Zhang et al. [171] uses a combination of an attention network that predicts an ROI from an 2D ultrasound image and a Mask R-CNN network containing LSTM cells that predicts the target position. A similar approach was proposed by Huang et al. [172] using an attention-aware CNN and convolutional LSTM (ConvLSTM) cells to perform target tracking in 4D ultrasound.
In this work, another more modular method for target patch forecasting is proposed. In contrast to other studies in literature, this is the first time an RNN-based approach is applied for 4D ultrasound tracking. The key idea of this approach is to predict the target patch for the next ultrasound image based on the last three target patches found by a separate tracking algorithm. Therefore, a spatio-temporal autoencoder was developed and trained in a supervised way[5]. The spatio-temporal autoencoder architecture is inspired by the neural network proposed by Deepak et al. [173] who aimed to perform anomaly detection in video sequences. In order to investigate whether this target patch forecasting method is advantageous, template matching was chosen as tracking algorithm. Therefore, the tracking results can be compared to the results observed in the previous section.

Data Preparation

In this study, the landmarks Lia and Lib in the sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3, 4\}$ were used for training and the sequences L-Vi-3 are used for testing. In order to prepare the data set for autoencoder training, the sequences were split into a couple of time-series. Since only every second ultrasound image is labeled, and labels are required for the proposed approach, frames without any label are omitted. Therefore, the average frame rate is reduced from 4 Hz to 2 Hz. From the remaining fully-labeled sequences, all possible consecutive 4D ultrasound time-series containing four 3D ultrasound images each were extracted. A time-series consists of ultrasound image patches $\mathbf{P}^{\text{seq}} = (\mathbf{P}^{t-3}, \mathbf{P}^{t-2}, \mathbf{P}^{t-1}, \mathbf{P}^{t})$ cropped from the ultrasound images $(\mathbf{I}^{t-3}, \mathbf{I}^{t-2}, \mathbf{I}^{t-1}, \mathbf{I}^{t})$ where t represents the point of time. The target patches have a size of $24 \cdot 24 \cdot 24$ vx and were cropped at the location of the corresponding labeled landmark from the ultrasound images. For model training, one time-series containing four ultrasound target patches corresponds to one training sample. The patches from the past points of time $(t-3, t-2, t-1)$ were used for training and the patch from the following point of time (t) served as ground truth. Altogether, eight training and eight test sets for all landmarks and sequences were generated. The number of samples per set is represented in Table 6.2.
In addition, the position trajectories of the labeled landmarks within the ultrasound

[5]The spatio-temporal autoencoder presented here was published in [160].

Table 6.2: Overview of the time-series data sets generated from the landmarks Lia and Lib with $i \in \{1, 2, 3, 4\}$ from the sequences L-Vi-1 and L-Vi-2 for training and L-Vi-3 for test.

Landmark	No. of time-series samples	
	Training	Test
L1a	106	92
L1b	106	92
L2a	135	97
L2b	135	97
L3a	119	97
L3b	119	97
L4a	108	97
L4b	108	97

sequences were prepared in a similar way. First, the temporal differences of the position trajectories were determined to get relative position changes instead of the absolute landmark locations. Second, all possible time-series were generated containing four time steps each. Again, the last time step of a time-series served as ground truth. In contrast to the image data, all trajectory data sets were combined into one position data set. Meaning, one trajectory data set containing 946 samples was created. This can be done since all motion trajectories are caused by the same physiological process, namely breathing, and the data was used in a relative position change format.

Autoencoder Training

The autoencoder, illustrated in Figure 6.2, consists of a typical encoder-decoder architecture with the encoder φ and the decoder δ. It gets a sequence containing three 3D ultrasound patches $\mathbf{P}^{\text{seq}} \in \mathbb{G}^{3\times24\times24\times24}$, maps it into one latent representation vector $\mathbf{z} \in \mathbb{R}^{128}$ and reconstructs one 3D ultrasound patch $\hat{\mathbf{P}}^t \in \mathbb{G}^{24\times24\times24}$. The autoencoder was trained by minimizing the difference measurement d between the predicted patch and the ground truth by

$$\arg\min_{\varphi,\delta} d\left(\delta\left(\varphi(\mathbf{P}^{\text{seq}})\right), \mathbf{P}^t\right). \tag{6.3}$$

The representation vector size was chosen according to section 4.2.3 where it was determined this size being sufficient for the used patch size. The encoder consists of three time-distributed convolution blocks (TD-convolution blocks), a ConvLSTM and two Residual blocks. The ConvLSTM layer is an extension from a fully-connected LSTM layer having convolutional structures in the input and output states proposed by Shi et al. [55]. They found that ConvLSTM performed better compared to fully-connected LSTM in a spatio-temporal forecasting task. As visualized in Figure 6.2, a TD-convolution block contains three time-distributed convolutional layers and a max-pooling layer. A Residual block contains two convolutional layers and a skip connection. This architecture is chosen to make the encoder capable of extracting spatio-temporal features from the input. The decoder consists of three Deconvolution blocks that contain an upsampling layer and three

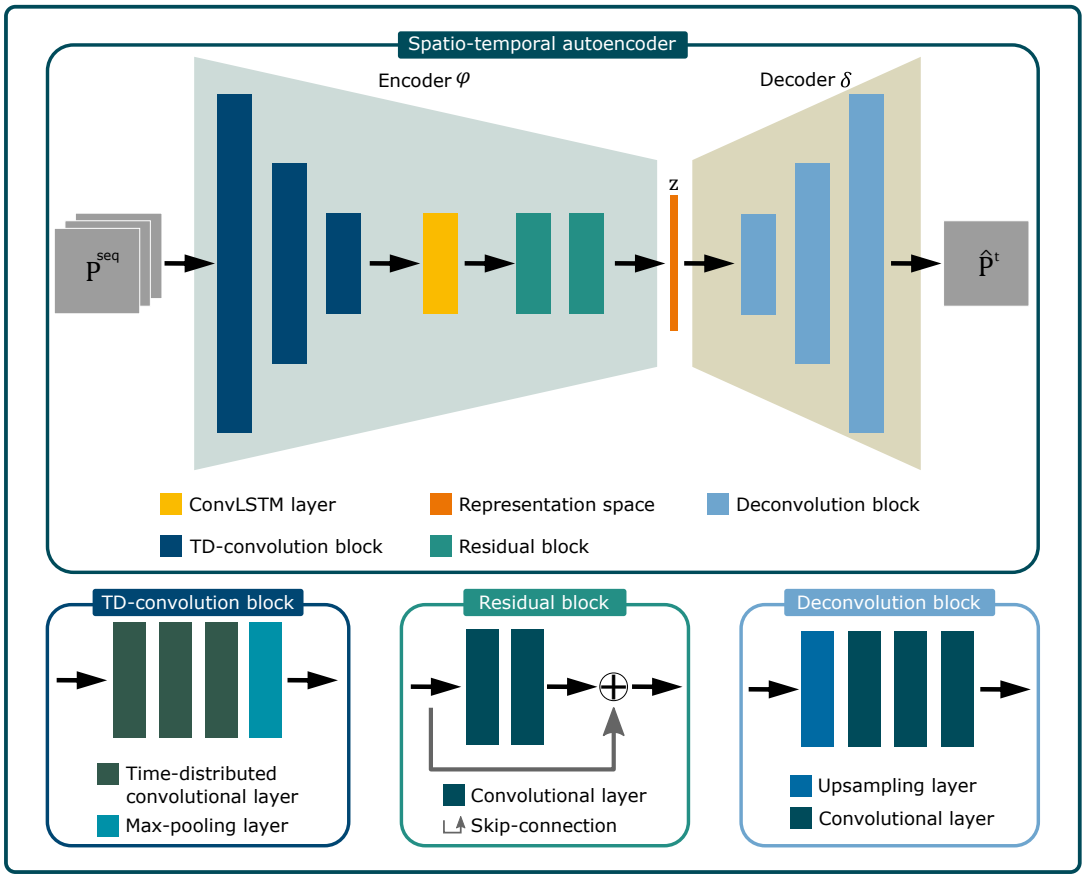

Figure 6.2: Overview of the spatio-temporal autoencoder architecture. A sequence of 3D ultrasound patches $\mathbf{P}^{\text{seq}} \in \mathbb{G}^{3\times24\times24\times24}$ is mapped into a latent representation vector $\mathbf{z} \in \mathbb{R}^{128}$ by the encoder. The decoder predicts an ultrasound patch $\hat{\mathbf{P}}^t \in \mathbb{G}^{24\times24\times24}$ containing the target of the next ultrasound frame. The encoder consists of three *TD-convolution blocks* (bottom left), a *ConvLSTM* layer and two *Residual blocks* (bottom center). The decoder consists of three *Deconvolution blocks* (bottom right). The layers are labeled with the number of used filter kernels c (under the blocks) and the sizes of the output feature maps (in the blocks).

convolutional layers each. The goal of the decoder is to reconstruct the 3D ultrasound patch showing the target from the ultrasound image of the point of time t using the latent representation vector.
Eight target-specific spatio-temporal autoencoders were trained based on the prepared time-series ultrasound data sets summarized in Table 6.2. For training, the Adam optimizer with weight decay of 10^{-5} and a batch size of 32, early stopping with a validation split of 20 % and the MSE loss (see equation 2.16) were used.

Recurrent Neural Network Training

For predicting the current landmark location, an RNN γ was used. An RNN consisting of two stacked LSTM layers with 100 units each and a following fully-connected layer forming the output vector $\mathbf{p} \in \mathbb{R}^3$ was implemented. The RNN gets time-series trajectory data $\mathbf{p}^{\text{seq}} = (\mathbf{p}^{t-3}, \mathbf{p}^{t-2}, \mathbf{p}^{t-1})$ containing the target motion vectors from the last three time steps and predicts the next target motion vector $\hat{\mathbf{p}}^t$. Model training was performed using the prepared trajectory data set. Again, the Adam

optimizer with a batch size of 32, early stopping with a validation split of 20 % and the MSE loss were used. Motion prediction for radiation therapy is a challenging problem under research. Different approaches have been proposed to compensate system latency and to estimate the target position based on real-time position data in order to optimize the irradiation beam [174]–[176]. However, since the purpose of this study is not to develop a novel target position predictor, but to estimate the target appearance, a simple RNN was used that provides sufficient accuracy for the purpose.

Target Forecasting Accuracy

The spatio-temporal autoencoder was trained for predicting the target appearance in the current ultrasound image given a time-series of ultrasound patches from the previous three images. For evaluating the accuracy of the predicted target patches, the time-series test data set presented in Table 6.2 was used. A predicted target patch $\hat{\mathbf{P}}^t$ was compared to the given ground truth patch of the current frame $\mathbf{P}^t$ using the root mean squared error (RMSE) defined by

$$\mathrm{RMSE}(\hat{\mathbf{P}}, \mathbf{P}^t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{P}_i - P_i^t\right)^2} \tag{6.4}$$

where N is the number of voxels in a patch and i denotes the voxel index. To create a baseline, the ground truth was also compared to the target patch $\mathbf{P}^0$ from the first ultrasound image of the sequence and from the previous ultrasound image ($\mathbf{P}^{t-1}$). This procedure corresponds to the FTT and UTT template matching-based tracking from section 6.1.1. Thus, it can be evaluated whether the target prediction brings an advantage over other target selection approaches in terms of image similarity.
The results are represented in Figure 6.3. As can be seen, the mean similarity between the ground truth and the target patch from the first ultrasound image ($\mathbf{P}^t$ vs. $\mathbf{P}^0$) is the lowest in all test sets. Only for landmark L4b a low mean RMSE was observed, which is the landmark tracked with the lowest TE of 1.71 mm using FTT (see Table 6.1). In contrast, the mean similarity between the ground truth and the prediction ($\mathbf{P}^t$ vs. $\hat{\mathbf{P}}^t$) is the highest in five out of eight test sets. In detail, in 85 % of all test time-series the prediction is more similar to the ground truth than the first image target patch (p-value $\ll 0.001$, according to t-test). Compared to the previous image target patch ($\mathbf{P}^t$ vs. $\mathbf{P}^{t-1}$), the prediction is more similar to the ground truth in 51 % of all time-series (p-value $= 0.053$).
In Figure 6.4, three examples from different time-series are illustrated. The lower similarity of the first image target patch to the ground truth in comparison to the prediction can also qualitatively be identified in all examples. The prediction, however, appears similar to the ground truth in all examples. This is even true for the bottom example, although, the RMSE is relatively high. In this example, the previous image patch is more similar to the ground truth according to the RMSE. Qualitatively however, the difference is low and hard to identify. Furthermore, it can be noticed that the prediction patches seem blurred compared to the other patches. This effect occurs due to the autoencoder that focuses on the most important features of the ultrasound patches, as the amount of information that can be stored

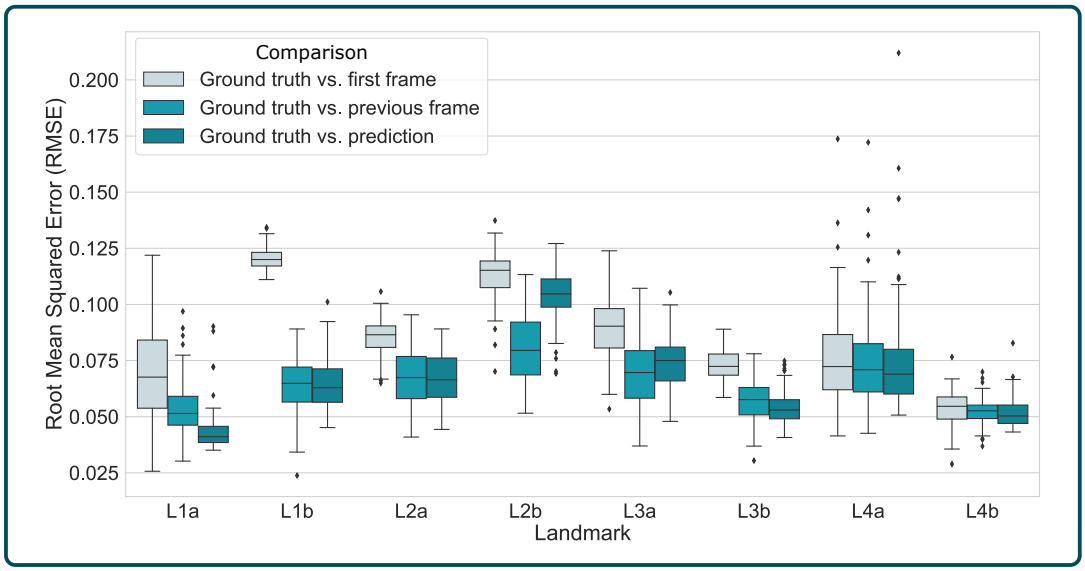

Figure 6.3: Comparison of similarity measurements using root mean squared error (RMSE) between ground truth target patches and target patches coming from the previous and the first ultrasound image as well as the predicted target patches.

in the latent representation is strongly limited. Therefore, noise and minor features are reduced.
It could be shown that the predicted target patches are significantly (p-value < 0.05) more similar to the ground truth than a fixed target patch in 85 % of the data set. This indicates that using a predicted reference target patch promises higher tracking accuracy than choosing a fixed reference target patch. In contrast, only in 51 % of the data set the predicted target patch was more similar to the ground truth than the target patch from the previous ultrasound frame. No significant difference could be observed here. However, since the predicted target patches do not contain any ultrasound noise, the quantitative analysis using RMSE might be inaccurate as the original ultrasound patches are noisy, and thus, might have an advantage over the predicted patches in image similarity metrics. Furthermore, the similarity difference between predicted and previous targets to the ground truth can depend on factors like target motion and deformation ranges. For instance, for the landmark L1a, the predictions are more similar to the ground truth than the previous target patch in 88 %, whereas for landmark L2b this rate is 10 %. The 3D landmark motion ranges of these landmarks are highly different: 45.1 mm for L1a and 23.9 mm for L2b. Considering similar frame rates of 1.8 Hz and 2.3 Hz as well as similar breathing rates, this indicates that predictions are fitting better to the ground truth in sequences with higher target motion between consecutive images. Obviously, in sequences with slow motion, the target appearance variation between consecutive images is low, and thus, the similarity between the target appearances in these images is high. However, for verifying this finding, a further investigation would be required.

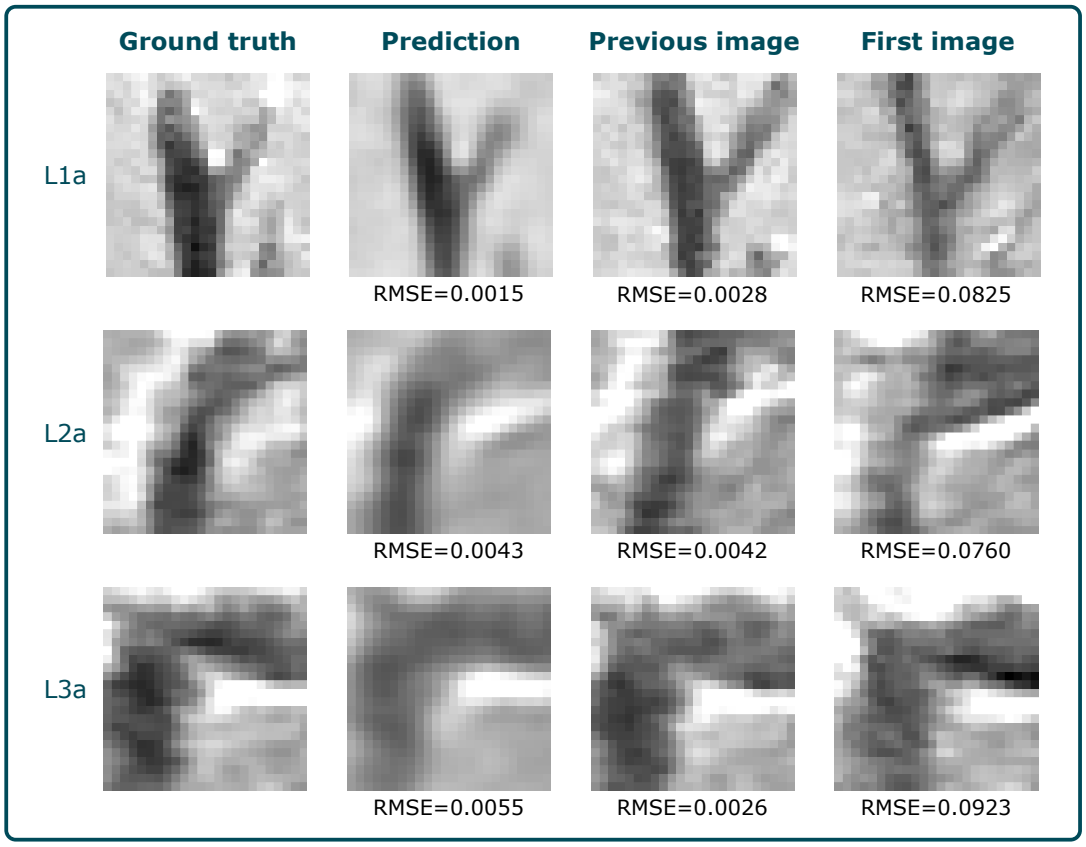

Figure 6.4: Comparison examples for slices from the target patches from the landmarks L1a (top), L2a (middle) and L3a (bottom). A ground truth target patch (left) compared to the spatio-temporal autoencoder prediction (middle-left), the previous frame (middle-right) and the first frame of the sequence (right). The RMSE scores compared to the corresponding ground truth is represented.

Tracking Experiments

This tracking study was conducted to evaluate the effects of target prediction in a tracking application. The tracking algorithm used in this study is the template matching algorithm presented in Algorithm 2 as it is a simple and easy to implement algorithm. However, the spatio-temporal autoencoder presented here can be integrated in any target tracking algorithm as it just provides a procedure for target appearance forecasting. Two different tracking approaches were implemented and investigated. The *Predicted Target Tracking* (PTT) and the *Weighted Predicted Target Tracking* (WPTT) approaches were performed using the trained models and compared to the naive template matching approaches presented in section 6.1.1. In general, the template matching algorithm worked the same in all approaches. However, the approaches differ in the way of updating the reference target patch $\mathbf{P}_{\text{ref}}$ used for performing template matching (see line 14 in Algorithm 2). In contrast to FTT and UTT, in PTT and WPTT the reference target patch was updated by predicting it for the next ultrasound image. Therefore, line 14 in Algorithm 2 is replaced by

$$\mathbf{P}_{\text{ref}} = \hat{\mathbf{P}}^t = \delta\left(\varphi(\mathbf{P}^{\text{seq}})\right) \tag{6.5}$$

where $\mathbf{P}^{\text{seq}} = \left(\hat{\mathbf{P}}^{t-3}, \hat{\mathbf{P}}^{t-2}, \hat{\mathbf{P}}^{t-1}\right)$ contains the patches identified as the target in the previous time steps of the ultrasound sequence $\mathbf{S}$. These patches are received by applying

$$\hat{\mathbf{P}}^{t-i} = \text{cropPatch}(\mathbf{S}^{t-i}, \hat{\mathbf{p}}^{\text{seq}}(t-i)), \quad \forall i \in \{1,2,3\}. \tag{6.6}$$

In addition, for cropping the ROI the position where to crop is defined using the RNN γ so that the line 4 in Algorithm 2 is replaced by

$$\begin{aligned} \hat{\mathbf{p}}^t &= \gamma\left(\mathbf{p}^{\text{seq}}\right) \\ \mathbf{R}_{\text{curr}} &= \text{cropROI}(\mathbf{I}_{\text{curr}}, \hat{\mathbf{p}}^t). \end{aligned} \tag{6.7}$$

Predicted Target Tracking In PTT the target reference patch as well as the ROI position are predicted using the spatio-temporal autoencoder and the RNN, respectively, as shown above. For this, the final positions from the previous three ultrasound images and the patches cropped from these positions are used.

Weighted Predicted Target Tracking In WPTT, PTT is extended. The position for the ROI is set not only based on the RNN prediction but also based on using two further approaches, to compensate potential prediction inaccuracies. For this, the final position from the previous ultrasound frame as well as an interpolated position determined by cubic regression are used. Thus, three ROIs are generated based on these three ROI positions. However, if ROIs spatially overlap, they are merged to avoid considering the same area more than once and to reduce computational costs. Thereby, for the ROIs based on the interpolated and the previous positions, weights are determined based on the differences between them and the predicted positions from the last $T = 3$ ultrasound images by

$$\lambda_{\text{test}} = 1 - \frac{1}{T}\sum_{i=1}^{T} \|\mathbf{p}_{\text{test}}^{t-i} - \hat{\mathbf{p}}^{t-i}\|_2, \quad \text{test} \in \{\text{previous}, \text{regression}\} \tag{6.8}$$

where $\mathbf{p}_{\text{test}}^{t-i}$ denotes the past positions estimated by either using the previous target position or the interpolated one. This is implemented by repeatedly performing the lines 6-13 in Algorithm 2 once for each test in test $\in \{\text{previous}, \text{regression}, \text{prediction}\}$ and determining the final predicted target position by

$$\hat{\mathbf{p}}^t = \arg\max_{\text{test}} \lambda_{\text{test}} \cdot sim_{test}^t \tag{6.9}$$

where $\lambda_{\text{prediction}} = 1$. This means, the more they differed from the predicted positions in previous frames, the less they are weighted. These weights are used for weighting the NCC similarity values within the corresponding ROIs.

Like in FTT and UTT, the NCC was used as a metric for measuring ultrasound patch similarity. For the tracking experiments, the 3D ultrasound sequences L-Vi-3 with $i \in \{1,2,3,4\}$ were used whereas the start position and the first reference patch corresponds to the labeled landmark from the first ultrasound image of the sequences.

Table 6.3: Tracking errors measured in the sequences L-Vi-3 with $i \in \{1, 2, 3, 4\}$ with the landmarks Lia and Lib using PTT and WPTT compared to the results achieved using FTT and UTT. The best values are highlighted in bold.

Method	Tracking error (mm)			Mean TE range (mm)	landmarks (%) where	
	Mean ± SD	95%ile	Max		TE > 3 mm	TE > 5 mm
No tracking	9.67 ± 9.55	34.28	42.15	[4.18, 28.61]	81.32	61.75
FTT	7.55 ± 12.31*	42.92	52.76	[1.71, 33.24]	49.17	28.21
UTT	8.00 ± 7.52	26.49	43.81	[3.77, 17.95]	87.67	60.23
PTT	6.82 ± 12.24*	45.21	53.84	[**0.98**, 28.83]	42.52	23.97
WPTT	**2.85 ± 2.04**	**6.45**	**19.90**	[1.31, **5.42**]	**36.29**	**15.10**

*Only TE ≥ 0 was measured.

Tracking Results

The two proposed tracking approaches PTT and WPTT were tested and compared to the naive template matching approaches FTT and UTT. For each ultrasound image, the TE was determined. The runtime of the tracking algorithm was not measured in the experiments as the template matching algorithm implemented here did not support real-time capability. In preliminary tests, runtimes > 1 s per 3D ultrasound image (< 1 Hz) were measured. The average tracking results are summarized in Table 6.3. It turned out that PTT and WPTT outperformed the baselines FTT and UTT. WPTT performed best with a mean tracking error of of 2.85 ± 2.04 mm, so that an improvement of 62 % between FTT and WPTT could be achieved. Even though, on average PTT performed worse than WPTT with a mean tracking error of 6.82 ± 12.24 mm[6] , the lowest mean tracking error per sequence of 0.98 mm was achieved using PTT.
When looking into the results in more detail, some aspects can be determined. The sequence-wise results are presented in Figure 6.5. It can be seen that higher TEs are measured for most of the landmarks when using FTT and UTT compared to PTT and WPTT. As shown in Figure 6.1, these high TEs were caused by the two failure types described in section 6.1.1. For instance, for the landmarks L1a and L1b, even clearly higher tracking errors of up to more than 50 mm and larger error ranges were measured when using FTT and UTT. The target loss failure was also observed in PTT for landmark L4a which is the reason for the high TE of 9.80 ± 15.62 mm[6] on average for this landmark. Here, the target gets lost and cannot be relocated in the following images so that the TE increases rapidly to up to 53.84 mm and stays high. However, in contrast to UTT, in PTT this failure was caused by rib shadow artifacts in the sequence V4-3 overlaying the landmark L4a, making it hard to precisely relocate the target. On average, PTT and WPTT performed better than the baselines. In Figure 6.6 the ground truth and tracking trajectories for landmark L1a achieved by the baselines (FTT and UTT) as well as PTT and WPTT are presented. It can be seen that by using PTT and WPTT the tracking failures observed in FTT and UTT could be prevented. In PTT and WPTT low mean tracking errors of 0.98 ± 0.50 mm and 1.31 ± 1.18 mm were measured in this example. When looking at Figure 6.6, it can be seen that PTT follows the ground truth precisely and no rapid jumps were measured in TE. However, as can be noticed in the y- and z-axes,

[6]Only TE ≥ 0 was measured.

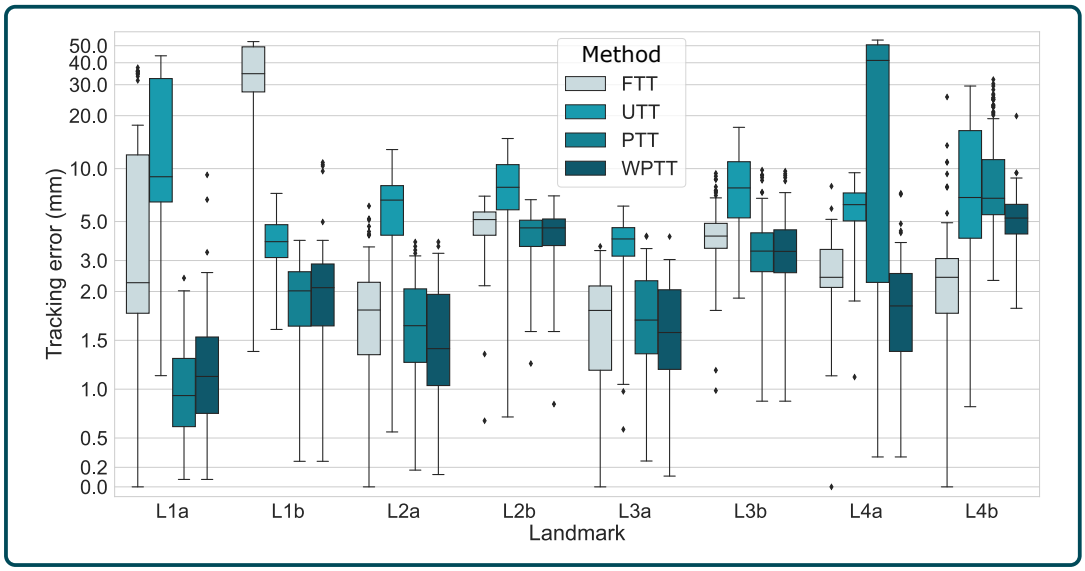

Figure 6.5: Results of the tracking experiments in the eight test sequences using Fixed Target Tracking (FTT), Updated Target Tracking (UTT), Predicted Target Tracking (PTT) and Weighted Predicted Target Tracking (WPTT). The y-axis is in a logarithmic scale.

in WPTT the target got lost in second 30, but it could be relocated in second 36 resulting in a small jump in TE. Due to this, WPTT performed worse than PTT in the whole sequence. Though, after relocating the target, the TE even slightly decreased indicating that WPTT is able to track the target precisely. In contrast to that, for the landmarks L4a and L4b, WPTT performed clearly better than PTT due to target loss occurring in PTT. This means, unlike PTT, WPTT is able to compensate tracking failures by considering more than one ROI.
The results indicate that using target patch and position predictions instead of the previously found or a fixed target patch (UTT and FTT) result in lower tracking errors. However, the RNN for position prediction used in this study is a simple model that could be replaced by a state-of-the-art method for achieving higher accuracy [176]. For the template matching-based approach, the RNN is sufficient as the required accuracy is ± 5 vx per spatial axis according to the ROI size. Though, the simple template matching algorithm brings some drawbacks. It is a time-consuming method, and, since it is a local search algorithm, large target displacements could not be compensated. However, since the target prediction process using the spatio-temporal autoencoder is independent of the tracking algorithm, it can be combined with any suitable tracking algorithm for improving the tracking accuracy. As the purpose of this study is to investigate the capability of improving a tracking algorithm, the results are promising. Drawback of the proposed method is its supervised learning characteristic so that it does not meet the learning type requirement. Labeling 4D ultrasound data is time-consuming and requires expert knowledge. Furthermore, the spatio-temporal autoencoders are trained target-specific, meaning, in a radiation therapy application an autoencoder must be trained for each patient individually. However, it was shown in section 4.2.1 that spatial autoencoders can be trained in a patient-generalized as well as an organ-generalized way. Therefore, it might also be possible to train a spatio-temporal autoencoder that way to avoid

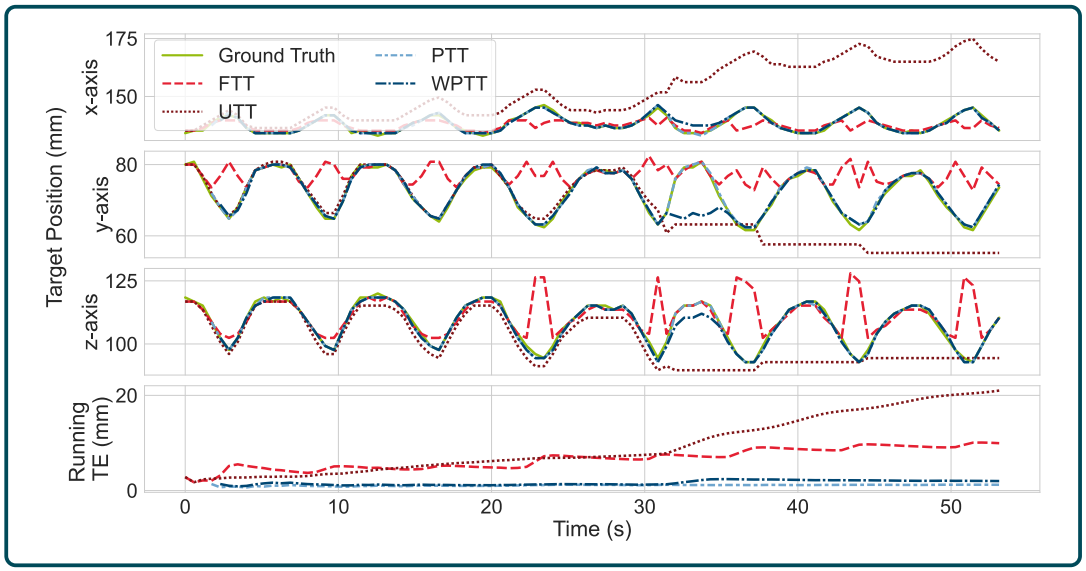

Figure 6.6: Examples for target tracking trajectories in x- (top), y- (second from the top) and z-axes (third from the top) measured after applying Fixed Target Tracking (FTT), Updated Target Tracking (UTT), Predicted Target Tracking (PTT) and Weighted Predicted Target Tracking (WPTT) to the landmark L1a compared to the labeled ground truth trajectory. In addition, the running TE for FTT, UTT, PTT and WPTT are visualized (bottom).

the need of patient-specific training.

6.1.3 Deep Localization Network

One of the most commonly used approaches for target tracking in ultrasound sequences is using siamese neural networks (SNNs). The key idea of an SNN is to feed the network with both, a target reference patch and a search image to predict the target position. In most approaches, the SNN predicts a heatmap containing pixel-wise probability information about the target position within the search image [177], [178]. While different approaches were developed and tested for tracking in 4D ultrasound, only He et al. [16] proposed and investigated a 3D SNN-based method for tracking in 4D ultrasound. The lack of such tracking methods for 4D ultrasound might be due to the lack of sufficiently sized labeled 4D ultrasound data sets. To train a robust and generalized neural network, a huge labeled data set would be required.
In this section, another simple target tracking approach is presented based on a deep localization model (DLN) trained in a supervised way. Since the 4D ultrasound data set used in this work contains a very limited number of different landmarks, it would be difficult to ensure that the DLN generalizes over a wide range of landmarks. Therefore, the DLN is trained in a target-specific way in this work[7]. This means, a DLN is trained on the basis of a small data subset containing labeled image data of only one specific target. Hence, the DLN does not need to have a two-branch architecture since it only predicts the position of the specific target anyway. Even

[7]The deep localization network presented here was published in [159].

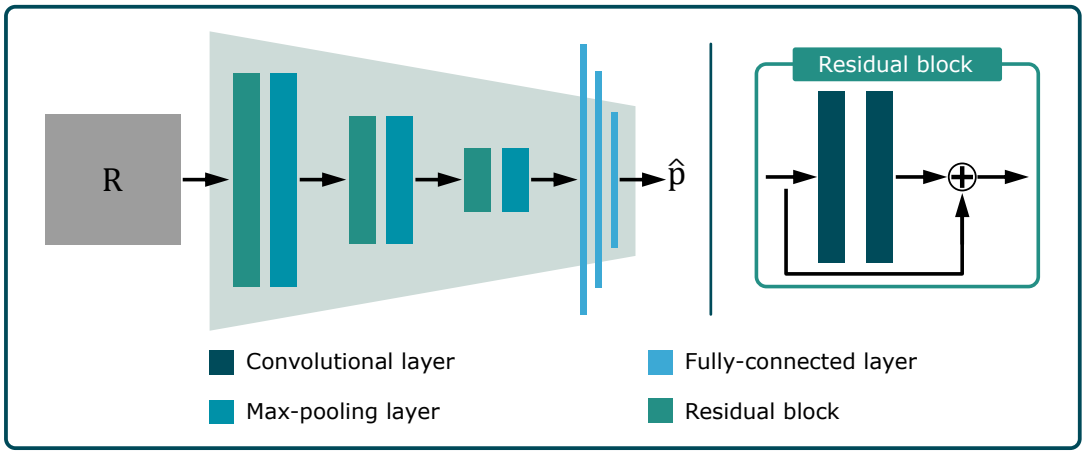

Figure 6.7: Architecture of the proposed DLN. Three residual blocks followed by max-pooling layers are used for feature extraction. Four fully-connected layers are used to obtain the output size of three in order to represent the coordinates of the target.

though this kind of neural network cannot meet the requirement for a tracking algorithm to be generalized, it provides a baseline for a target tracking algorithm in 4D ultrasound. In a second experiment, the DLN is trained based on four different targets to investigate whether such a DLN can also be trained for several targets without a loss in tracking accuracy.

Data Preparation

In this study, the ultrasound sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3, 4\}$ with the landmarks Lia and Lib were used. In total, eight 3D ultrasound sequences and eight different landmarks were considered. Since this is a supervised learning approach, the labeled landmarks were used as ground truth for training and evaluation of the DLN. To reduce the computational costs during the localization process, a ROI with a size of $48 \cdot 48 \cdot 48$ vx was used instead of the whole 3D ultrasound image. Data augmentation was used in order to represent rigid transformations in the training data that can occur due to respiration. Therefore, translation was integrated by cropping the ROI around the landmark with a random offset from the range $[-10, 10]$ vx per axis. This range was chosen according to the translational landmark motion determined in section 3.4. Additionally, 70 % of the data were rotated by random angles from the range $[-5, 5]^\circ$ per axis. Since the number of available labels per landmark is rather small with 120 on average, for each landmark, 60 ROIs with different offsets and different rotations were created to increase the number of training data samples. Hence, each data set contained about 7200 ROIs.

Neural Network Training

For the single-target as well as the multiple-target approach the DLN architecture illustrated in Figure 6.7 were used. It consists of three convolutional Residual blocks [179] followed by max-pooling and three fully-connected Layers. The DLN gets a 3D ultrasound ROI $\mathbf{R} \in \mathbb{G}^{48\times48\times48}$ as input where the searched target shall be found

and returns a vector $\hat{\mathbf{p}} \in \mathbb{R}^3$ defining the target position within the ROI. This vector represents the coordinates of the searched target with regard to the coordinate system of the ROI. Since image-wise localization is a numerical regression problem, MSE is chosen as loss function. The DLN was trained to minimize the distance between the predicted landmark position and the ground truth, i.e. the labeled landmark position. For this purpose, the Adam optimizer and a batch size of 42 was used. During training, regularization of the DLN was performed in order to prevent overfitting. Early stopping was used, whereby training is interrupted if the validation loss does not improve within 10 epochs. Furthermore, a stair-cased learning rate scheduler was applied. At the beginning of the training, the learning rate was set to 10^{-4}. If the validation loss has not improved for 3 epochs, the learning rate was multiplied by 10^{-1}. Thereby, a lower threshold of 10^{-6} was set. For this, a validation split of 10 % was chosen. DLN training was executed on an NVIDIA GeForce GTX 1070Ti GPU.
On the basis of the two ultrasound sequences that contain the same landmark (e.g., sequences L-V*i*-1 and L-V*i*-2 both contain the landmark L*i*a), a 2-fold cross-validation was performed. The generated ROIs of the one sequence were used for the training, the remaining sequence is used for evaluating the DLN in a tracking experiment. This was repeated twice so that each sequence is used once for evaluation. The single-target approach intends to find one specific target, so for this, one DLN is trained for each of the eight landmarks. The multiple-target approach aims to predict multiple specific targets using one DLN. Therefore, the landmarks were divided into two groups by the labels A and B, respectively, like it was done in section 3.3.2 in the labeling study. Thus, two DLNs in total were trained for the multiple-target approach.

Tracking Experiments

For tracking, an image-by-image localization of the searched target was performed within the ROIs. Therefore, the sequences were processed image by image in the correct temporal order. For the first image, the starting position was selected according to the labeled landmark position. Around this position, the ROI was cropped from the 3D ultrasound image. This ROI was then fed into the DLN to predict the position of the target. The resulting position was applied to the next ultrasound image in the sequence to extract the ROI. This procedure was repeated for all images in the sequence. To evaluate the tracking accuracy, the TE was determined in all images where a landmark label is available. Due to the 2-fold cross-validation, the two sets of TEs of the test sequences were merged to determine the mean TE of the DLN.
Table 6.4 shows the mean TEs measured in this study. In the single-target experiments, a mean TE of 2.28 ± 1.20 mm and a 95th percentile of 4.30 mm was measured. In the multiple-target experiments, a comparable mean TE of 2.23 ± 1.28 mm with a 95th percentile of 4.38 mm was obtained with no significant (p-value = 0.38, according to t-test) difference to the single-target results.
The mean processing time over all experiments was 68 ms (15 Hz). This processing time contains the extraction of the ROI, the normalization of the gray-scale intensities and the prediction of the target position. A post-processing step is not necessary in this approach, since the DLN predicts the coordinates of the target directly. The

Table 6.4: The mean TE results of the 2-fold cross-validation for the single-target and the multiple-target approaches. Tracking experiments were performed in the 3D ultrasound sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3, 4\}$ for the landmarks Lia and Lib. The best values are highlighted in bold.

Method	Tracking Error (mm)			Mean TE range (mm)	Landmarks (%) where	
	Mean ± SD	95%-tile	Max		TE > 3 mm	TE > 5 mm
No tracking	8.84 ± 7.90	28.08	31.15	[3.75, 20.33]	75.64	56.78
Single	2.28 ± **1.20**	**4.30**	**12.89**	[1.75, **2.76**]	20.33	2.92
Multiple	**2.23** ± 1.28	4.38	13.74	[**1.27**, 3.92]	**20.00**	**1.46**

processing time required for one image measured in the experiments was much faster than the frame rate of 4 Hz the 4D ultrasound data set was acquired with. Since the mean processing time of the DLNs was at most 76 ms, the tracking algorithm could be applied in real-time to ultrasound sequences acquired with a frame rate of up to 13 Hz.
The results show that translational motion can be tracked robustly in all sequences in real-time using a DLN. It was observed that even large target shifts of up to 20 mm could be compensated due to using a ROI of sufficient size. However, in the tracking of the landmarks L3b, L4a and L4b higher TE outliers of more than 5 mm were measured. In the L4a and L4b cases, these outliers occurred due to the fact that the landmarks were temporarily covered by a rib shadow artifact, causing the searched landmark to be invisible. This could be overcome by increasing the amount of labeled ultrasound images for training, to achieve more robust results. However, it has to be noticed that this would require more labeling effort which is time-consuming and, thus, impractical in a clinical setting. Due to the supervised training procedure, for the use, e.g., in radiation therapy, an additional planning ultrasound acquisition phase including labeling of the target would be required.

6.2 Tracking in Representation Space

The image space tracking algorithms presented in the previous sections have some disadvantages violating the requirements defined in chapter 1. The simple template matching-based algorithm is not real-time capable and not robust as different systematic failures were observed. Even after adding a method for target patch forecasting enhancing the tracking performance, the lack of real-time capability remains. In addition, due to the lack of a large amount of labeled data with sufficient variability, the spatio-temporal autoencoder was trained in a supervised and target specific way so that it did not generalize for different targets. This issue was also observed in the DLN approach. However, the DLN meets the real-time requirement for the 4D ultrasound data set.
The key idea of this work to meet all requirements of the tracking algorithm is to use feature-based tracking, e.g., by applying representation learning. Yet, the use of feature-based tracking in ultrasound imaging has rarely been investigated. Shepard et al. [180] used a feature-based rough displacement estimation before performing the actual tracking algorithm to better compensate large target shifts and Hallack et al. [157] used SIFT feature-based image registration to perform landmark tracking.

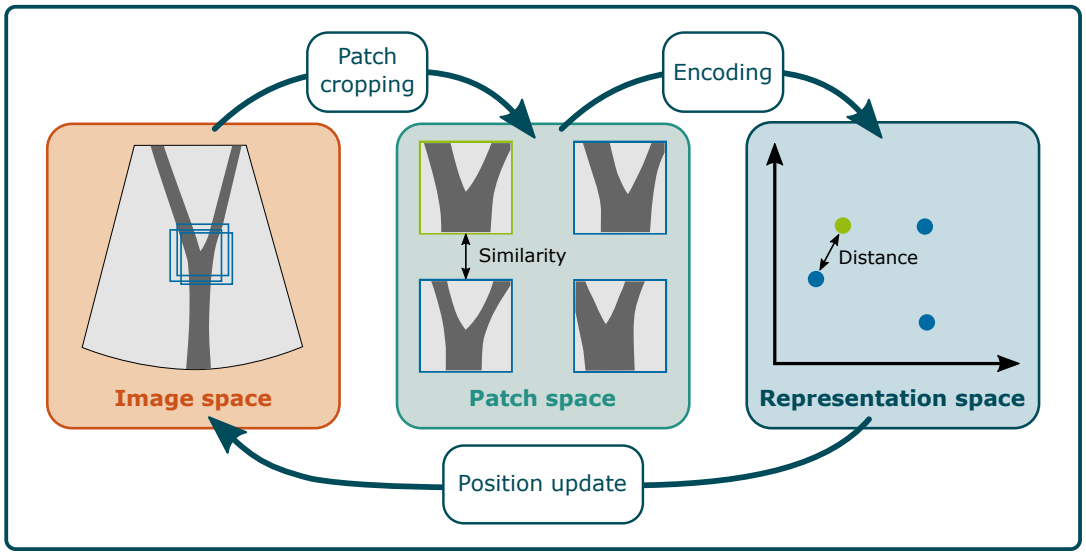

Figure 6.8: Illustration presenting the key idea of tracking a landmark in representation space. In the ultrasound image space (left) features are located, e.g., at previous landmark positions or by using a feature detector and mapped into the patch space (middle) by cropping patches from the located positions. The ultrasound patches are mapped (encoded) into the representation space (right) by using an autoencoder or a feature descriptor. Thus, the comparison of ultrasound patches (blue) with a target reference patch (green) shifts from measuring similarities in patch space to determining distances in representation space.

These approaches were developed and tested in 2D+t ultrasound. For 4D ultrasound, no feature-based tracking approach has been proposed yet. The tracking approach presented in this work is based on the key idea to map landmarks within a 3D ultrasound image into a latent representation space as illustrated in Figure 6.8. This can be done, e.g., by using a feature descriptor or an autoencoder as shown in chapter 4. Hence, comparing the appearance of landmarks is transferred from measuring similarities in image space to determining distances in representation space. This approach has the advantage that it can be learned in a fully unsupervised way, eliminating the need for labels in the training process. In addition, due to the reduction of data dimensionality by mapping 3D images into a latent representation vector, unimportant and redundant information as well as noise are reduced while important information is preserved. This characteristic can support a tracking algorithm locating the target position more reliably.
In this section, different approaches for extracting local feature representations from 3D ultrasound images are investigated for the purpose of target tracking. An image feature-based approach as well as two autoencoder-based approaches are proposed and executed in tracking experiments. In addition, to meet the real-time requirement, a greedy-based target search algorithm is proposed and integrated in the tracking experiments.

6.2.1 Image Feature-based Tracking

In literature, feature-based tracking is less commonly studied than image intensity-based algorithms. However, as shown in section 4.1, image features can be detected

and described in 3D ultrasound images using simple feature detectors and descriptors. Kubota et al. [181] detected features in limited ROIs of 2D ultrasound images using the Harris corner detector (see section 2.1) and performed tracking by estimating the average motion between the feature sets of two consecutive 2D ultrasound images only based on the features' location. In their study, they did not use a feature descriptor for feature matching. Since there is a lack of investigation for using feature descriptors in 4D ultrasound tracking, it is investigated in this work. Hence, the detector FAST-3D as well as the binary descriptors BRISK-3D and BRIEF-3D are implemented.
To investigate the applicability of these approaches to target tracking in 4D ultrasound, a tracking study is performed to quantify the achievable tracking accuracy[8]. In addition, using a SWAE for feature description is tested as well in order to compare the performance of binary feature descriptors and a deep learning-based approach. In total, three algorithm combinations are tested where FAST-3D is combined with either BRISK-3D, BRIEF-3D or a SWAE. For feature matching the same algorithm is used for all combinations. However, when applying the feature detector and descriptors, different parameters need to be set manually as described in section 4.1. Finding the optimal parameter setting is challenging due to the large number of parameters, e.g. up to six when using FAST-3D, BRISK-3D and the feature matching algorithm. Due to the fact that the parameters can influence each other, it is not possible to optimize each parameter separately by hand. Therefore, parameter optimization is performed to find the optimal parameter setting for target tracking.

Parameter Optimization

In this study, 12 sequences L-Vi-j with $i \in \{1,2,3,4\}$, $j \in \{1,2,3\}$ with two corresponding landmark labels each were used. In order to avoid that the tracking experiments were performed on the same sequences as parameter optimization, the sequences were split into a training set

$$\mathcal{S}_{\text{train}} = \{\text{L-V}i\text{-}1 \mid i \in \{1,2,3,4\}\} \tag{6.10}$$

containing four sequences and a test set

$$\mathcal{S}_{\text{test}} = \{\text{L-V}i\text{-}j \mid i \in \{1,2,3,4\}, j \in \{2,3\}\} \tag{6.11}$$

containing eight sequences.
Parameter optimization was done using Bayesian optimization [182], [183]. This is a machine learning-based optimization method that basically maximizes a given function by adapting the parameters of the function. Such an optimization algorithm is much faster than performing a grid-search as it requires by far less iterations to find the maximum [184]. Here, the function to maximize is the tracking algorithm $f_{track}(\mathbf{x}, \mathcal{S}_{\text{train}})$ that gets a set of d parameters $\mathbf{x} \in \mathbb{R}^d$ for the detector, the descriptor and the matching algorithm as well as the training set of ultrasound sequences $\mathcal{S}_{\text{train}}$ and returns the tracking error (the algorithm is described in the following section in

[8] The tracking study presented here was partially published in [110].

Table 6.5: Parameters for detector, descriptors and matching algorithm used in the tracking experiments optimized by using Bayesian optimization.

Approach	FAST-3D			BRISK-3D		BRIEF-3D	Matching
	r	t	p	δ_{min}	δ_{max}	r	δ
FAST-3D + BRISK-3D	2	0.05	0.8	13	15	-	0.4
FAST-3D + BRIEF-3D	2	0.05	0.8	-	-	5	0.4
FAST-3D + SWAE	3	0.09	0.8	-	-	-	0.002

detail). Since the TE needs to be minimized and Bayesian optimization performs a maximization, the maximization problem is defined by

$$\max_{\mathbf{x}} -f_{track}(\mathbf{x}, \mathcal{S}_{\text{test}}). \tag{6.12}$$

For finding the optimal parameters for the three detector-descriptor combinations investigated, tracking was performed for each combination individually. The parameter ranges were set as $r \in \{2, 3, 4\}$, $t \in [0.05, 0.20]$, $p \in [0.5, 0.8]$ for the FAST-3D radius, intensity threshold and state threshold, respectively. For BRISK-3D, the threshold parameter ranges $\delta_{min} \in [14, 20]$ and $\delta_{max} \in [8, 15]$ were set and for BRIEF-3D $r \in [5, 25]$ for the mask size. For the threshold parameter in the matching algorithm, the range $\delta \in [0, 2]$ was set. The length l of the description or representation vectors was set fix to 256 for all descriptors.
The optimal parameters determined by the Bayesian optimization are presented in Table 6.5. It can be seen that the optimal parameters for FAST-3D are the same for the combinations with BRISK-3D and BRIEF-3D. When using the SWAE for feature description, however, the parameters are slightly different. Furthermore, the matching parameter δ is different when using SWAE compared to BRISK-3D or BRIEF-3D. This is due to the fact that the Hamming distance is used to compare BRISK-3D and BRIEF-3D description vectors, since they are binary descriptors, but not for SWAE representation vectors since these are non-binary defined as $\mathbf{z} \in [-1, 1]^l$. For this reason, Euclidean distance is used here.

Autoencoder Training

The autoencoder used for feature description is an SWAE. The architecture of the SWAE is presented in Figure 6.9. It consists of an encoder φ and a decoder δ that maps a 3D ultrasound patch $\mathbf{P} \in \mathbb{G}^{24\times24\times24}$ into a latent representation vector $\mathbf{z} \in [-1, 1]^{256}$ and predicts a reconstructed patch $\hat{\mathbf{P}} \in \mathbb{G}^{24\times24\times24}$. Encoder and decoder are built symmetrically containing five convolutional layers, two max-pooling (encoder) or upsampling layers (decoder), respectively and two fully-connected layers. To limit the representation vector to the range $[-1, 1]$ the tanh activation function is used in the last layer of the encoder. In the remaining layers ReLU activation function is used. For regularization, batch normalization is performed before max-pooling and after upsampling layers. For SWAE training, 40,000 3D ultrasound patches were cropped randomly from the training segments of each of the sequences L-Vi with $i \in \{1, 2, 3, 4\}$ leading to a training data set containing

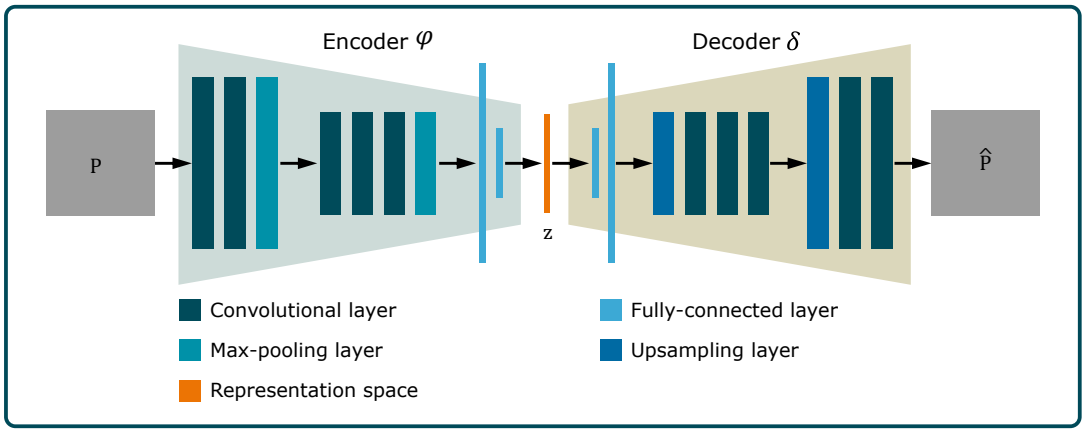

Figure 6.9: Architecture of the SWAE used for feature description. Encoder and decoder are built symmetrically consisting of convolutional, max-pooling or upsampling and fully-connected layers. Batch normalization is performed before every max-pooling and after every upsampling layer. The SWAE maps a 3D ultrasound patch $\mathbf{P} \in \mathbb{G}^{24\times24\times24}$ into a latent representation vector $\mathbf{z} \in [-1,1]^{256}$.

160,000 3D ultrasound patches. Training was executed on an NVIDIA RTX 2080Ti GPU using a batch size of 64, the Adam optimizer and early stopping regularization with a validation split of 10 % to prevent overfitting. The loss function is defined by

$$L_{SWAE}(\mathbf{P}, \hat{\mathbf{P}}, p_{\mathbf{z}}) = \lambda_{MSE} \cdot L_{\text{MSE}}(\mathbf{P}, \hat{\mathbf{P}}) + \lambda_{SW} \cdot L_{\text{SW}}(p_{\mathbf{z}}, p_{\gamma}) \tag{6.13}$$

where p_{γ} is the sphere-shaped distribution presented in section 4.2.2 and the weights were chosen as $\lambda_{MSE} = 1$ and $\lambda_{SW} = 0.1$.

Tracking Experiments

The tracking algorithm used here is illustrated in Figure 6.10. It aims to determine the motion between two consecutive 3D ultrasound images and consists of four steps: keypoint detection, feature description, feature matching, and motion estimation, which are executed sequentially in a loop and repeated for each new ultrasound image. Given two 3D ultrasound images $\mathbf{I}^{t-1}, \mathbf{I}^{t}$ from consecutive time steps $t-1$ and t, first, keypoints are detected in both images using FAST-3D generating two sets of keypoints $\mathcal{K}^{t-1} = \{\mathbf{k}_1^{t-1}, ..., \mathbf{k}_N^{t-1}\}$ and $\mathcal{K}^{t} = \{\mathbf{k}_1^{t}, ..., \mathbf{k}_N^{t}\}$. All detected keypoints are described using one of the three feature descriptors. Subsequently, feature matching is performed based on the feature description vectors to find corresponding features between the two ultrasound images. To estimate the motion between the ultrasound images a mean motion vector is determined by

$$\hat{\mathbf{m}} = \frac{1}{N} \cdot \sum_{i=1}^{N} \mathbf{k}_i^{t} - \mathbf{k}_i^{t-1}. \tag{6.14}$$

This procedure was executed for all ultrasound images within the 3D ultrasound sequences in the test set. The tracking experiments were executed on an Intel Core i5-4460 CPU, whereas SWAE prediction was performed on an NVIDIA GeForce GTX 970 GPU. The TE was determined for each landmark $\mathbf{l} \in \mathbb{R}^3$ separately.

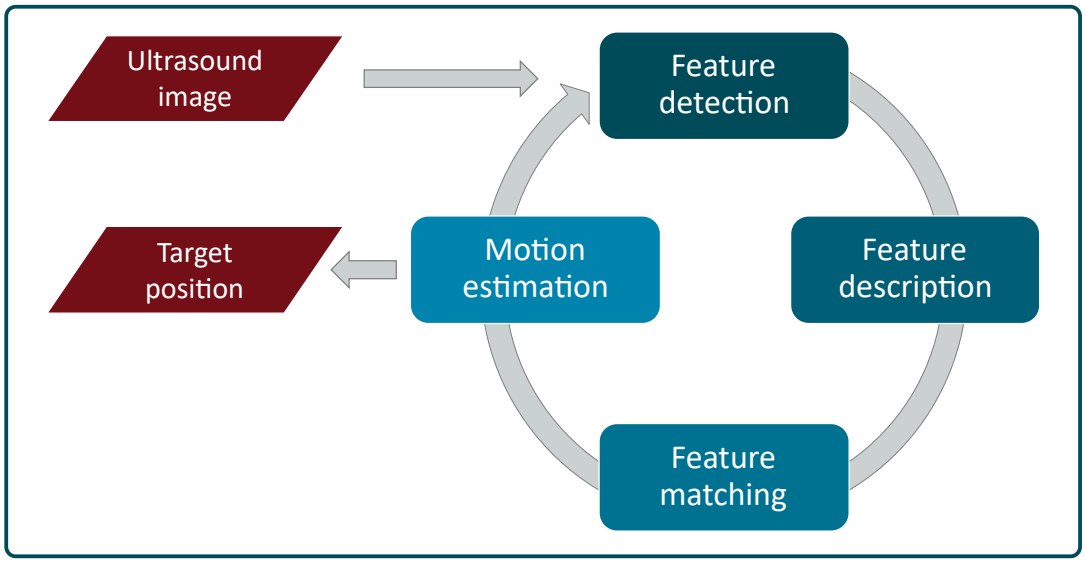

Figure 6.10: Flowchart representing the image feature-based tracking algorithm. In a new 3D ultrasound image, keypoint detection, feature description, feature matching and motion estimation are executed sequentially. Feature matching is performed based on the features from the previous and the new ultrasound image. Based on the estimated motion the current target position is determined. The process is repeated for every new ultrasound image.

Table 6.6: Results of the tracking experiments based on the 3D ultrasound sequences L-Vi-2 and L-Vi-3 with $i \in \{1, 2, 3, 4\}$ with the labeled landmarks Lia and Lib. For all descriptors, features are detected using FAST-3D algorithm. The best results are highlighted in bold.

Descriptor	**Tracking error (mm)**			**Mean TE**	**landmarks (%) where**	
	Mean ± SD	**95%ile**	**Max**	**range (mm)**	**TE > 3 mm**	**TE > 5 mm**
No tracking	8.32 ± 8.42	28.71	42.15	[3.75, 28.61]	76.49	55.38
BRISK-3D	2.58 ± 2.17	5.95	25.26	[1.85, 3.61]	27.04	8.32
BRIEF-3D	2.29 ± 1.59	5.03	25.30	[1.76, 2.96]	22.96	5.20
SWAE	**2.08 ± 1.50**	**4.36**	**24.10**	[**1.68**, **2.62**]	**16.80**	**3.28**

For this, the mean motion vector $\hat{\mathbf{m}}$ was added to the landmark position from the previous time step $t-1$ and the resulting estimated landmark position

$$\hat{\mathbf{l}}^t = \mathbf{l}^{t-1} + \hat{\mathbf{m}} \tag{6.15}$$

was compared to the ground truth by determining the TE between the prediction $\hat{\mathbf{l}}^t$ and the ground truth $\mathbf{l}^t$. In all test sequences the landmarks could be tracked robustly. The results of the tracking experiments are presented in Tables 6.6 and 6.7. In addition, for sequence comparison the TEs are illustrated for all test sequences for every descriptor separately in Figure 6.11. It can be seen that using the SWAE led to significant (p-value $\ll 0.01$, according to t-test) lower TE of 2.08±1.50 mm compared to 2.29 ± 1.59 mm and 2.58 ± 2.17 mm measured when using BRIEF-3D and BRISK-3D, respectively. However, BRISK-3D and BRIEF-3D performed significantly (p-value $\ll 0.01$) faster than SWAE. Since the feature detection and matching algorithm are fixed, the time difference is caused by the description method. BRIEF-3D had the lowest mean processing time of 137 ms (7.3 Hz) which is 46 % faster than SWAE with 256 ms (3.9 Hz). For the given test sequences,

Table 6.7: Mean Runtime and number of found features and matches in the tracking experiments based on the 3D ultrasound sequences L-Vi-2 and L-Vi-3 with $i \in \{1, 2, 3, 4\}$ with the labeled landmarks Lia and Lib. For all descriptors, features are detected using FAST-3D algorithm. The best values are highlighted in bold.

Descriptor	Mean runtime (ms)	Mean no. features	Mean no. matches	Matches per features (%)
BRISK-3D	205	**168.2**	24.5	15
BRIEF-3D	**137**	**168.2**	**62.2**	37
SWAE	256	58.0	25.1	**43**

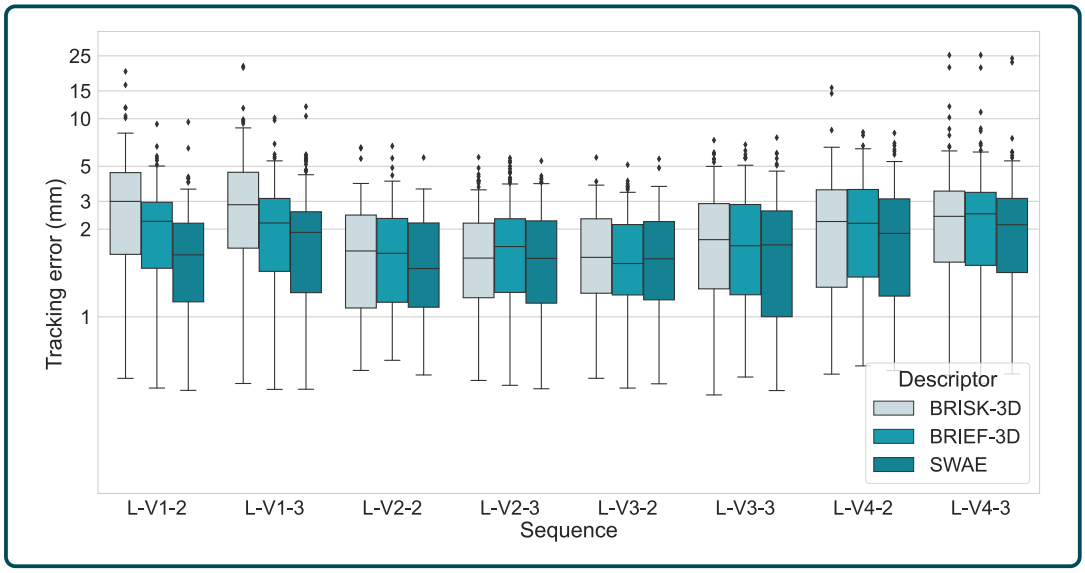

Figure 6.11: Mean TEs measured in the experiments using BRISK-3D, BRIEF-3D and SWAE for feature description for all test sequences. The y-axis is in a logarithmic scale.

that were acquired at a mean frame rate of 4.0 Hz, the SWAE approach is nearly real-time capable. In contrast, the BRISK-3D and BRIEF-3D approaches could be performed in real-time. However, the hardware used in the tracking experiments has limited power and the implementation for FAST-3D, BRIEF-3D, BRISK-3D and feature matching is CPU-based. By using more powerful hardware and implementing GPU-based algorithms, there is high potential for improving the runtime so that the proposed approach is a promising real-time capable tracking method.
The number of features detected varied between the approaches, which is caused by using different parameters in FAST-3D. It turned out that the optimal FAST-3D sphere radius differs between the approaches. Furthermore, by using BRISK-3D or BRIEF-3D, a FAST-3D threshold $t = 0.05$ was used. This leads to the fact that more features were detected than when using SWAE ($t = 0.09$) since the intensity ranges for the non-similar states were increased. On average, three times more features were detected when using BRIEF-3D or BRISK-3D. Features could be found in all 3D ultrasound test images as the minimum number of features detected was 36 and 111 when using the SWAE and BRIEF-3D or BRISK-3D combination, respectively. Furthermore, the minimum number of found matches was 3 (BRISK-3D), 6

(SWAE) and 16 (BRIEF-3D). However, the ratio between the number of matches and the number of detected features was the highest when using the SWAE. This indicates that the SWAE provides more robust feature descriptions than BRIEF-3D and BRISK-3D. In the BRISK-3D approach the minimum number of found matches is low with 3 matches. However, such a low number of matches only occurred in the sequences V1-2 and V1-3 which could be the reason for the high mean tracking errors of $3.61 \pm 3.09\,\text{mm}$ and $3.59 \pm 2.92\,\text{mm}$ in these sequences.
The results indicate that a learning-based descriptor such as a SWAE is more robust and leads to lower tracking errors than using binary feature descriptors. However, there is still room for improvement in the approach. The parameters were optimized for the given data set. It is still unclear whether the parameters need to be adjusted when using a different data set acquired with different ultrasound machine settings. This needs to be investigated in future studies. In addition, determining a mean motion vector is a very simple method that needs to be improved. As mentioned in section 3.4, the liver is also exposed to deformation which shows the need of considering local deformation vectors such as in a deformation field [154]. This could be done, e.g., by generating a sparse deformation field from the matched image features [185].

6.2.2 Greedy-Based Tracking Algorithm

When using representation learning, the tracking process is transferred from image space into representation space. However, an algorithm to locate the representation vector closest to the targets' representation in a new ultrasound image is still required. A naive approach is using template matching as shown in section 6.1.1. Template matching could also be executed in representation space where the target representation vector is compared to all representation vector candidates within an ROI. Since this approach is time consuming, especially in 3D image data, another approach is developed in this work to provide a fast algorithm.
Due to the slow and continuous breathing-induced motion pattern of the liver tissue, the target position does not move much between consecutive images in an ultrasound sequence. On average, an adult performs 12-20 respiration cycles per minute at rest which corresponds to a respiration rate range of $[0.2, 0.3]\,\text{Hz}$ [186]. Assuming a mean motion range of $11.56 \pm 5.86\,\text{mm}$ according to the findings in section 3.3.1, liver structures move at an average speed range of about $11.56\,\text{mm} \cdot [0.2, 0.3]\,\text{Hz} = [2.31, 3.47]\,\frac{\text{mm}}{\text{s}}$. Since the 4D ultrasound data set was acquired with an average frame rate of $4\,\text{Hz}$, the range of liver tissue motion between two consecutive images is about $[2.31, 3.47]\,\frac{\text{mm}}{\text{s}} \cdot 4\,\text{Hz} = [0.58, 0.87]\,\text{mm}$. When taking the mean spatial resolution of the 3D ultrasound images of $1.0 \cdot 0.7 \cdot 1.5\,\text{mm}^3$ into account, liver structures move about $0.39\,\text{vx}$ to $1.24\,\text{vx}$ between consecutive ultrasound images depending on the image axis. Therefore, when neglecting large motion events such as coughing or arbitrary patient movements, it can be assumed that the target moves continuously at a limited speed of about $1\,\text{vx} - 2\,\text{vx}$ per time step per image axis. Based on this assumption, a local greedy-based search algorithm is developed in this work.
To locate the landmark position in the next ultrasound image of a sequence, an

Algorithm 3 Greedy algorithm to find the target position $\mathbf{p}^t$ in a 3D ultrasound image $\mathbf{I}_{\text{curr}}$ taken from a sequence $\mathbf{S}$ at time step t starting at the position $\mathbf{s}$.

Input: $\mathbf{P}_{\text{ref}}, \mathbf{S}, \mathbf{s}$
Output: $\hat{\mathbf{p}}^{\text{t}}, t = 1$ to EndofSequence

1: $\mathbf{p}^0 = \mathbf{s}$
2: **for** $t = 1$ to EndofSequence **do** ▷ Go through the Sequence $\mathbf{S}$
3: $\quad \mathbf{I}_{\text{curr}} = \mathbf{S}^t$
4: $\quad \mathbf{p}_{\text{curr}} = \hat{\mathbf{p}}^{t-1}$
5: $\quad \mathbf{P}_{\text{curr}} = \text{cropPatch}(\mathbf{I}_{\text{curr}}, \mathbf{p}_{\text{curr}})$
6: $\quad \text{dist}^t = \mathit{Distance}\,(\varphi(\mathbf{P}_{\text{ref}}), \varphi(\mathbf{P}_{\text{curr}}))$
7: $\quad \text{OptimumFound} = \mathit{False}$
8: $\quad$ **while not** OptimumFound **do**
9: $\quad\quad \mathbf{P}_{\text{curr},i} = \text{cropPatch}(\mathbf{I}_{\text{curr}}, \mathbf{n}_i \in \text{Neighborhood}(\mathbf{p}_{\text{curr}}))$
10: $\quad\quad \mathit{dist}_{\text{curr}} = \min_i \mathit{Distance}\,(\varphi(\mathbf{P}_{\text{ref}}), \varphi(\mathbf{P}_{\text{curr},i}))$
11: $\quad\quad$ **if** $\mathit{dist}_{\text{curr}} < \mathit{dist}^t$ **then**
12: $\quad\quad\quad \mathit{dist}^t = \mathit{dist}_{\text{curr}}$
13: $\quad\quad\quad \mathbf{p}_{\text{curr}} = \arg\min_{\mathbf{n}_i \in \text{Neighborhood}(\mathbf{p}_{\text{curr}})} \mathit{dist}_{\text{curr}}$
14: $\quad\quad$ **else**
15: $\quad\quad\quad$ **if** Sub-voxel accuracy is required **then**
16: $\quad\quad\quad\quad \mathbf{p}_{\text{curr},i} = \arg\min_{\mathbf{n}_i \in \text{Neighborhood}(\mathbf{p}_{\text{curr}})} \mathit{dist}_{\text{curr}}$
17: $\quad\quad\quad\quad \hat{\mathbf{p}}^t = \text{SubVoxelPosition}(\mathbf{p}_{\text{curr}}, \mathbf{p}_{\text{curr},i},\ i \in \{1, 2, 3\})$
18: $\quad\quad\quad$ **else**
19: $\quad\quad\quad\quad \hat{\mathbf{p}}^t = \mathbf{p}_{\text{curr}}$
20: $\quad\quad\quad$ **end if**
21: $\quad\quad\quad \text{OptimumFound} = \mathit{True}$
22: $\quad\quad$ **end if**
23: $\quad$ **end while**
24: **end for**

iterative hill-climbing greedy search algorithm is applied starting at the target position of the previous ultrasound image [187]. The greedy tracking algorithm is presented in Algorithm 3[9]. All patches in the three-dimensional neighborhood of the current position are encoded and the representations are compared to the target representation. The neighborhood taken into account can be defined manually. In preliminary experiments, the 6- and 14-neighborhood illustrated in Figure 6.12 showed most promising results. Hence, these will be used in the following tracking studies. If there is a representation in the local neighborhood that is closer to the target representation than the current one, the position is updated. The distance between two representations is measured using a distance metric such as the Euclidean distance (see equation 4.13) or the cosine distance (see equation 4.14) so the greedy search is looking for the local minimum. This procedure is repeated until the global optimum is obtained. Additionally, to provide sub-voxel accuracy, a method for determining the final target position is implemented (see lines 15-18 in Algorithm 3). Without performing sub-voxel accuracy, the final target position $\hat{\mathbf{p}}^t$

[9]The greedy tracking algorithm presented here was partially published in [134].

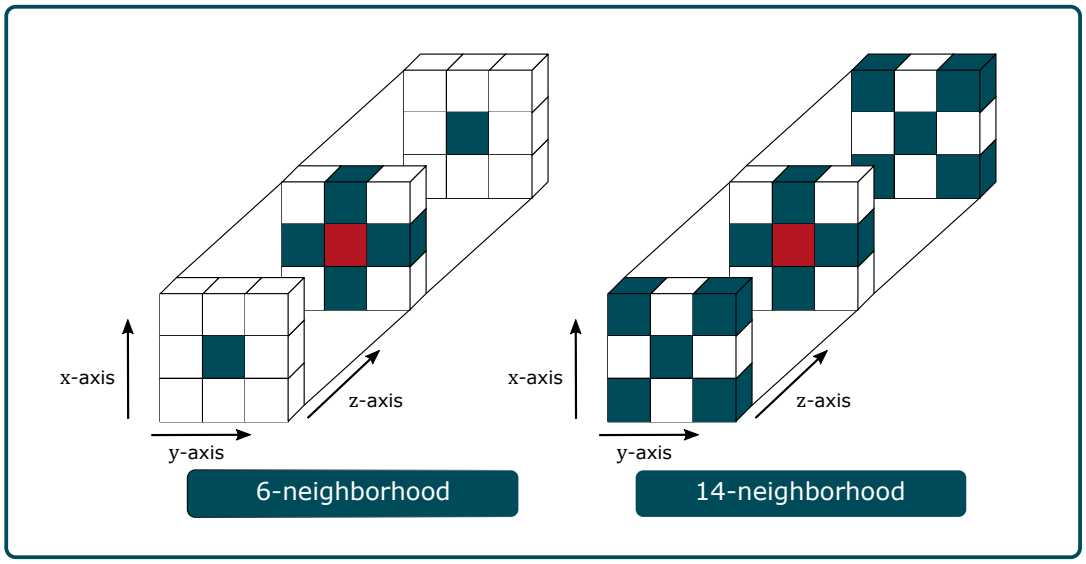

Figure 6.12: Illustration showing the 6- and 14-neighborhoods used in the greedy tracking algorithm. The current position is located in the center of the 3D patch (red) and the considered neighbors are marked in blue.

corresponds to the position of the representation vector with the minimum distance to the reference representation vector (see line 9 in Algorithm 3). When sub-voxel accuracy is applied, the final target position $\hat{\mathbf{p}}^t$ is linearly interpolated based on the final position and three neighbor voxel positions with the smallest distance $dist_i$ to the reference patch in representation space ($\mathbf{p}_{\text{curr},i}$ with $i \in \{0,1,2,3\}$ where $\mathbf{p}_{\text{curr},0} = \mathbf{p}_{\text{curr}}$). Based on the distances $dist_i$, a weighting is determined by

$$\lambda_i = \frac{\frac{1}{dist_i}}{\sum_i \frac{1}{dist_i}} \tag{6.16}$$

so that $\sum_i \lambda_i = 1$. As a result, positions with a smaller distance in representation space will be weighted higher, and vice versa. Subsequently, the final target position is determined by

$$\hat{\mathbf{p}}^t = \sum_i \mathbf{p}_{\text{curr},i} \cdot \lambda_i. \tag{6.17}$$

In template matching, the runtime depends on the size and dimensionality of the image or ROI to be searched through since all candidates are evaluated before a final decision is made. In contrast, the runtime of the greedy tracking algorithm depends on the motion speed of the target. Since the algorithm performs an iterative hill-climbing search following the descending distance values until the global minimum is reached, the runtime depends on the number of iterations required to reach the global minimum. The general process of the greedy search algorithm for the 2D case is illustrated in Figure 6.13. It can be seen that the current target position prediction follows the measured distances in representation space until the minimum is reached. In this example, the greedy algorithm required three iterations to reach the minimum. Another advantage of the greedy algorithm is the runtime invariance

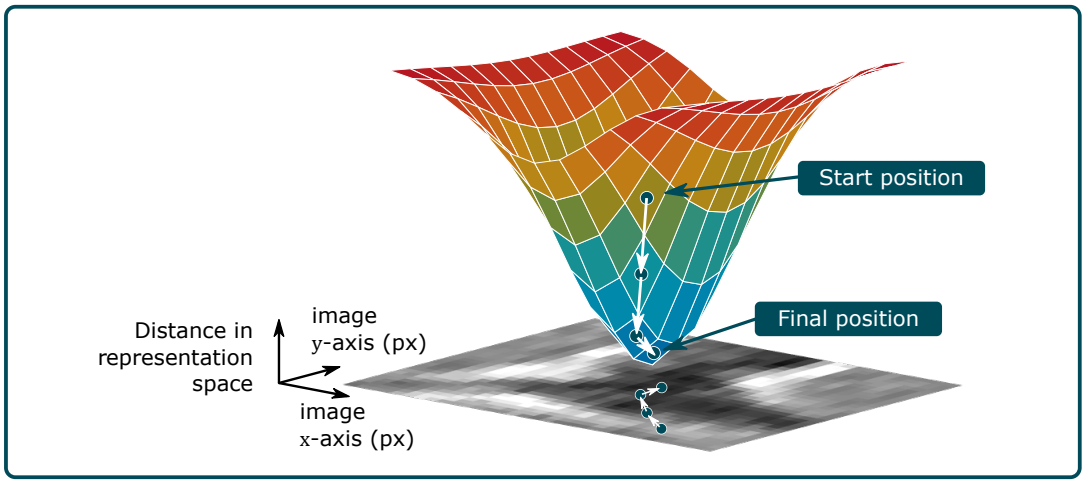

Figure 6.13: Illustration showing the process of the hill-climbing greedy search algorithm used for target tracking in 4D ultrasound. For simplicity, an example for a 2D image is presented here. The algorithm follows the lowest distance measurements in representation space in an iterative way until the global minimum is reached.

to the ultrasound image size. Due to the local search characteristic, the image size has no influence on the runtime.
This greedy tracking algorithm is used in the following to perform target tracking in 4D ultrasound.

6.2.3 Sliced-Wasserstein Autoencoder

As shown in section 4.2.2, the SWAE showed promising results in distinguishing dissimilar and clustering similar ultrasound patches. Therefore, the SWAE is used for performing target tracking in this section. 3D ultrasound patches are mapped into the SWAE representation space and compared to reference patches by determining distances in order to locate the target within the ultrasound image. Unlike the image feature-based approach presented in section 6.2.1, here the landmark is tracked directly by applying the greedy tracking algorithm presented in section 6.2.2 without performing feature detection. The process of the tracking algorithm is illustrated in Figure 6.14. As can be seen, target selection and encoding are performed once initially, and the tracking process corresponds to the execution of the greedy tracking algorithm.
In section 4.2.3 the optimal size of the representation space was investigated for 3D ultrasound patches of the size $24 \cdot 24 \cdot 24$ vx on the basis of the reconstruction accuracy. For this, different SWAE were trained on the basis of different training data subsets. In the following, these trained models are utilized to perform target tracking. It is investigated whether the optimal representation space size can be confirmed by measuring the tracking accuracy and how precise a target can be tracked using this approach[10].

[10]The SWAE-based target tracking study presented here was published in [134].

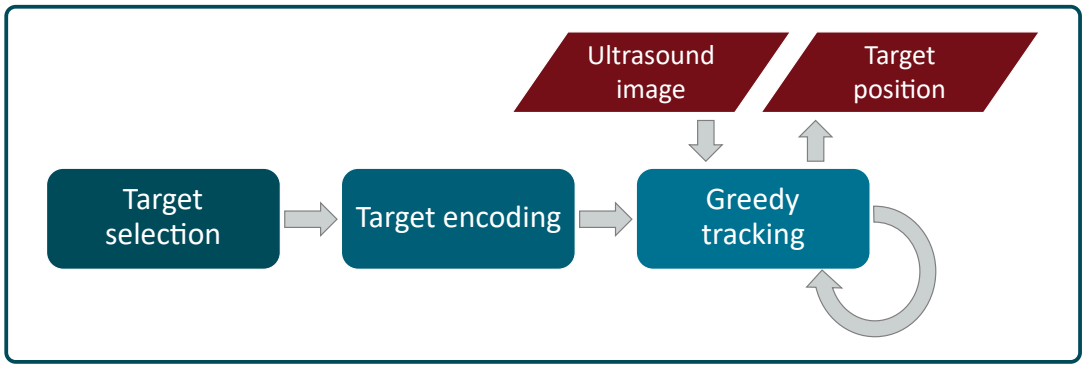

Figure 6.14: Flowchart representing the representation learning- and greedy-based tracking procedure. Initially, the target is selected and encoded. The greedy tracking algorithm is then applied in a loop that is repeated for each new ultrasound image and that predicts the current target position.

Table 6.8: Overview of the three types of trained autoencoders. $i \in I = \{1, 2, 3, 4\}$ define the subject with the corresponding ultrasound liver sequence L-Vi-j and $j \in J = \{1, 2\}$ denote the short-term ultrasound sequence number. In every sequence, one landmark Lia is used as ground truth.

Approach	Autoencoder	Training Data	Tracking Data
Individual	$\text{SWAE}_{indiv,i}$	$\mathcal{T}_i$	L-Vi-j, Lia
Generalized	$\text{SWAE}_{gen,i}$	$\mathcal{T}_h, h \in I \backslash i$	L-Vi-j, Lia
Generalized fine-tuned	$\text{SWAE}_{genFT,i}$	$\text{SWAE}_{gen,i} + \mathcal{T}_i^*$	L-Vi-j, Lia

* Fine-tuning is performed only with 5,000 patches selected randomly from the data set.

Data Preparation

The data preparation and autoencoder training has already been presented in the study investigating the optimal size of the latent representation space in section 4.2.3. Here, these trained autoencoders are used in a tracking study. As a reminder, the three ways of autoencoder training are shown in Table 6.8. In addition, information about the sequences and landmarks used for performing target tracking is presented. Note that only ultrasound images not taken from the test sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3, 4\}$ were used to avoid bias in tracking. As presented in section 4.2.3, each autoencoder was trained with different latent representation space sizes $k \in \{8, 16, 32, 64, 128, 256, 512\}$ in order to investigate the optimal size for ultrasound patches with a size of $24 \cdot 24 \cdot 24$ vx. In the following, tracking is performed using all these models to investigate whether the results observed in the image reconstruction accuracy study can be confirmed.

Tracking Experiments

Deformation turns tracking into a challenging task due to changes of the target shape and appearance over time. As shown in section 3.4, liver motion is mainly caused by respiration and has translation, rotation and deformation parts. This leads to the fact that the target shape in an inhale phase can highly differ from the target

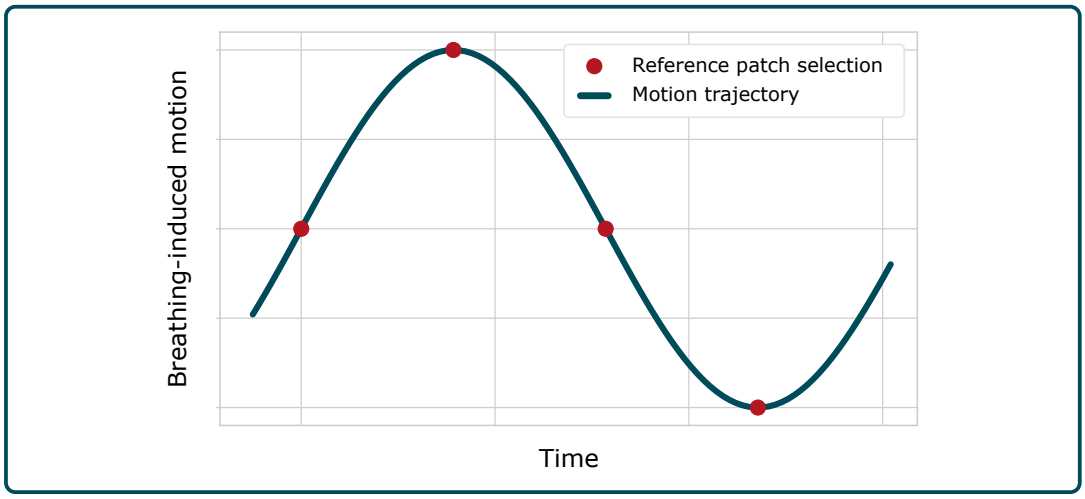

Figure 6.15: Illustration showing the points of time in the breathing-induced motion cycle where the four reference target patches are taken from. Two are taken after the inhale and exhale phases (upper and lower peak) and two are taken during the inhale and exhale phases.

shape in an exhale phase. Therefore, four target reference patches were defined for tracking in this study in order to better cover the range of target shape variability. The target patches were chosen manually in four different breathing cycle phases like illustrated in Figure 6.15. The first and second patch were taken after an inhale and exhale phase, respectively and the third and fourth patch were taken during an inhale and exhale phase, respectively. Target patches were taken from the other short-term sequence part of the same long-term sequence. This means, the target patches for tracking in L-V*i*-1 were taken from L-V*i*-2 and vice versa. Tracking was performed by using the greedy tracking algorithm presented in Algorithm 3. Here, the sub-voxel accuracy was not applied. During the tracking procedure, the distances between the neighborhood candidates $\mathbf{z}^{j}_{\text{candidate}}$ and all target references $\mathbf{z}^{i}_{\text{ref}}$ were measured in the representation space using the Euclidean distance (see equation 4.13). The neighborhood candidate with the smallest distance to any of the four target references was selected by

$$\mathbf{p} = \arg\min_{i,j} d_{\text{euclidean}}\left(\mathbf{z}^{i}_{\text{ref}}, \mathbf{z}^{j}_{\text{candidate}}\right) \tag{6.18}$$

where $j \in \{1, ..., 14\}$ denotes the candidate from the local neighborhood and $i \in \{1, 2, 3, 4\}$ denotes a target reference.

Tracking Results

The tracking accuracy was evaluated using the labeled landmarks L*i*a. In ultrasound images where a landmark exists, the TE between the tracking position and the ground truth landmark was determined. The mean tracking results of all experiments are illustrated in Figure 6.16. At lower representation space sizes the TE is high and it decreased when the representation space size is set higher. However, the TE saturated at a mean error of 4.54 ± 1.46 mm, 3.48 ± 1.63 mm and 3.30 ± 1.39 mm in individual, generalized and generalized fine-tuned autoencoders,

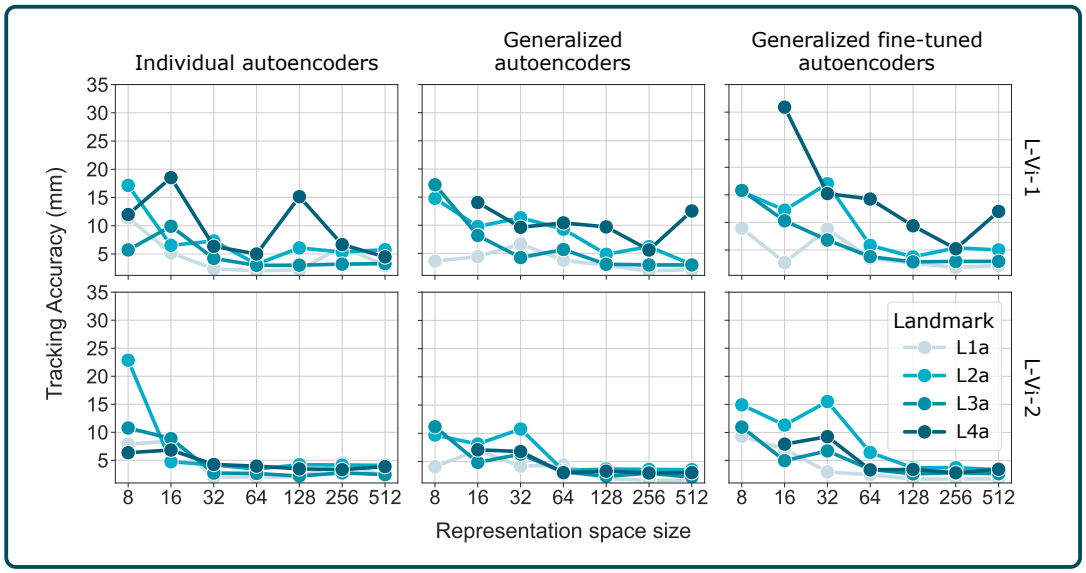

Figure 6.16: Tracking accuracy of the autoencoders $\text{SWAE}_{indiv,i}$ (left), $\text{SWAE}_{gen,i}$ (middle) and $\text{SWAE}_{genFT,i}$ (right) in the sequences L-Vi-1 (top) and L-Vi-2 (bottom) with different representation space sizes $k \in \{8, 16, 32, 64, 128, 256, 512\}$ for the landmark Lia.

Table 6.9: Results of the tracking experiments using the autoencoders with representation space size of 256 performed in the sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3, 4\}$ to track the landmarks Lia. The best results are highlighted in bold.

Auto-encoder	Tracking error (mm)			Mean TE range (mm)	landmarks (%) where	
	Mean ± SD	95%ile	Max		TE > 3 mm	TE > 5 mm
No tracking	8.73 ± 8.02	28.54	31.15	[4.82, 19.91]	78.18	56.78
$\text{SWAE}_{indiv,i}$	4.08 ± 4.59*	9.39	35.94	[2.03, 6.61]	49.47	19.96
$\text{SWAE}_{gen,i}$	2.81 ± 4.03*	5.35	29.41	[**1.33**, 5.55]	17.41	5.10
$\text{SWAE}_{genFT,i}$	**2.58 ± 3.15***	**4.72**	**27.11**	[1.38, **4.58**]	**15.92**	**4.67**

*Only TE ≥ 0 was measured.

respectively at a representation space size of 256. Detailed results of the tracking experiments with the autoencoders of this representation space size are summarized in Table 6.9. The results show that the performances of generalized and generalized fine-tuned autoencoders significantly (p-value $\ll 0.01$, according to t-test) outperformed the individual autoencoders with an average improvement of 1.27 mm and 1.50 mm, respectively. However, the use of generalized fine-tuned autoencoders did not show a significant (p-value = 0.17) improvement compared to the generalized autoencoders even though the TE decreases by 0.23 mm, on average. For the clinical workflow this finding is promising as it indicates that it is possible to train an autoencoder that generalizes for different patients and no further training is required. However, since the SWAE was trained unsupervised, requiring no data labeling, the entire fine-tuning process could be automated, making it an easy step to implement. In addition, in the fine-tuning process only the fully-connected layers were updated so the computing time is reduced compared to training from scratch.
In this study, a similar trend in the representation space size experiments could be observed as in section 4.2.3 where the reconstruction accuracy at different representation space sizes was investigated. Representation space sizes of $k \in \{128, 256, 512\}$

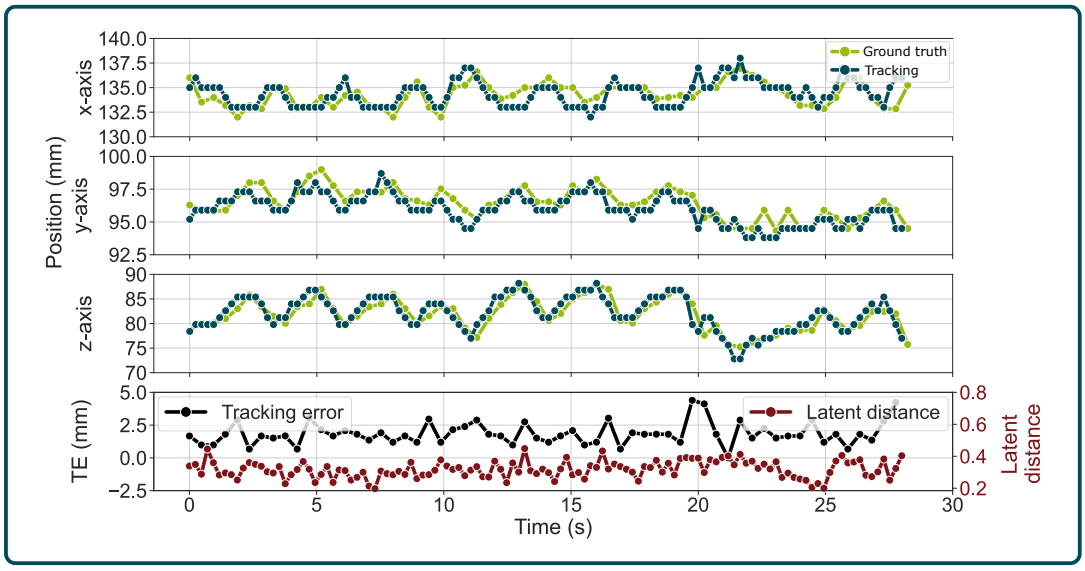

Figure 6.17: Result of tracking the landmark L3a (blue) and the labeled ground truth (green) for the complete ultrasound sequence L-V3-1 in x-, y- and z-axes (top three plots). In addition, the TE for every labeled ultrasound image (black) and the minimal distance measured in the latent representation space for every ultrasound image (red) are illustrated. In this example, a TE of 1.86 ± 0.81 mm was measured.

led to robust target tracking behaviors with similar TEs, while smaller representation space sizes led to higher TEs. In Figure 6.17 the tracking result along all three axes for landmark L3a in sequence L-V3-1 with the autoencoder $\text{SWAE}_{genFT,1}$ is illustrated. In this example, a low mean TE of 1.86 ± 0.81 mm was measured indicating robust tracking of the landmark. However, as can be seen in the outliers in Figure 6.16, some tracking experiments led to a high TE, even at larger representation space sizes. An example for these cases is presented in Figure 6.18 where the ground truth and tracked landmark trajectories using the autoencoder $\text{SWAE}_{genFT,4}$ with the representation space size of 512 applied in the sequence L-V4-1 are visualized. In this experiment a TE of 12.54 ± 10.19 mm was measured. It can be seen that the algorithm loses the target after about 17 s and does never relocate it. Instead, the tracking algorithm seems to track another structure nearby the target as the tracked motion trajectory still correlates with the landmark trajectory. Such failures were caused by position shifts of the annotated target like shown in the z-axis in Figure 6.18. Here, the target shifted about 20 mm along the z-axis between two annotated images. Such shifts lead to a high TE because the assumption that the next target position lies nearby the previous target position is violated. In these cases the greedy tracking algorithm was not able to recover the target as it is a local search algorithm. Apart from these cases, tracking with the greedy algorithm was successful which means that the target was not lost.
In some experiments another failure was observed. the TE increased repeatedly in the same phase in the motion pattern. In Figure 6.18 this can be seen at the time steps 0 s, 7 s and 14 s. The tracking algorithm was not able to find the exact target position in these phases. This is caused by the fact that only four fixed target references are used. In addition, these references were taken from a sequence that

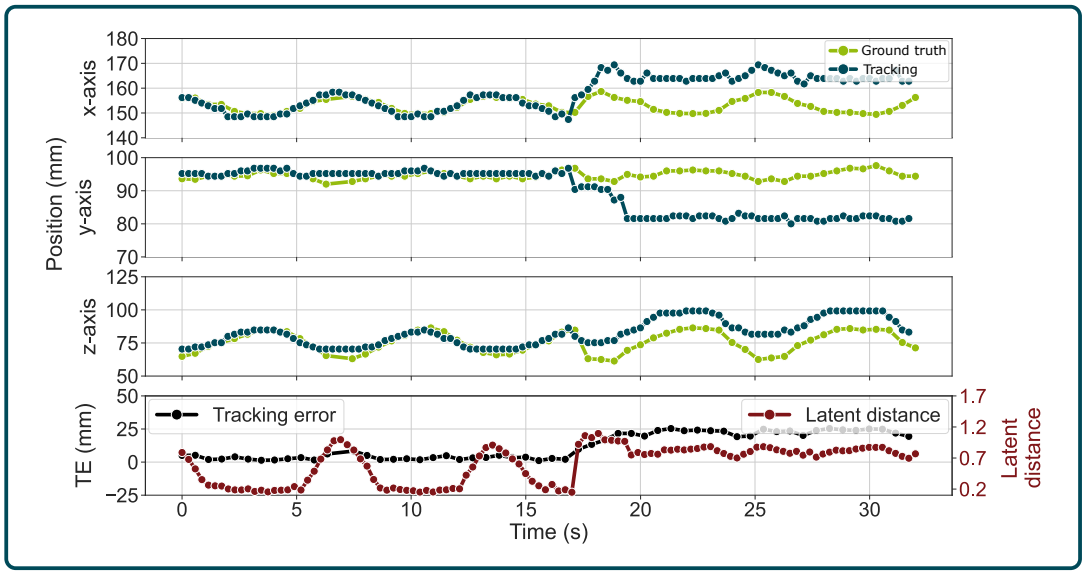

Figure 6.18: Result of tracking the landmark L4a (blue) and the labeled ground truth (green) for the complete ultrasound sequence L-V4-1 in x-, y- and z-axes(top three plots). In addition, the TE for every labeled ultrasound image (black) and the minimal distance measured in the latent representation space for every ultrasound image (red) are illustrated. In this example, a TE of 12.54 ± 10.19 mm was measured.

has a temporal distance to the tracked sequence. Although the motion is periodic, the target does not necessarily deform in the same way all the time. This means the target references taken from L-V1-1 can be inadequate for an exact tracking in L-V1-2. However, this approach is realistic as therapy planning is done some time before the treatment in the clinical workflow.

6.2.4 Deformable Convolutional Autoencoder

In the previous section it was shown that landmarks can be tracked in the representation space of an autoencoder. However, some challenges still remain. It was observed that the greedy algorithm fails tracking the target when large target shifts occur due to the local search characteristic of the algorithm. This problem could be solved, e.g., by performing a rough global image registration using the image feature-based method presented in section 6.2.1 before applying the local greedy tracking algorithm. Such a two-step approach was also used by Shepard et al. [180]. So far, four target reference patches are used in order to cover the range of target variability caused by high-dimensional target motion. Even though this approach is common [180], [188], it requires manual breathing phase-dependent target labeling from a physician in the clinical workflow. In addition, the four target reference patches were found to be insufficient in some test sequences.
In the following, a method is proposed that aims to maximize the ability of the autoencoder to map similar ultrasound patches closer to each other in the representation space. A deformable convolutioal autoencoder (DCAE) is proposed that contains deformable convolution layers to better deal with deformable soft-tissue

targets[11]. For 2D+t ultrasound tracking, deformable convolutional layers were used by Liu et al. [189] in an SNN that achieved a low TE in their target tracking study indicating that this layer is promising. In this work, deformable convolution for target tracking in 4D ultrasound is investigated for the first time. It will be determined if adding deformable convolutional layers is advantageous for this purpose. In addition, a novel data dimensionality reduction technique is proposed to reduce 3D ultrasound patches into a set of 2D ultrasound patches. This kind of data format will be called 2.5D ultrasound patches.

Data Preparation

For this study, the 3D ultrasound sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3\}$ with the landmarks Lia, Lib and Lic were used for evaluation. For training the autoencoder, 70,000 patches sized $24 \cdot 24 \cdot 24$ vx were extracted randomly from each ultrasound sequence L-Vi in the training segments. Thus, in total a training data set with 210,000 ultrasound patches was created. In addition, another data dimensionality reduction technique is proposed in this study. For further data dimensionality reduction, 2D ultrasound slices were extracted from the 3D ultrasound patches and arranged in three different ways as illustrated in Figure 6.19 and described in the following to create 2.5D ultrasound patches.

3-channel Data In a first step, the three orthogonal midline slices were extracted similar to the procedure performed by Huang et al. [15]. This procedure is illustrated on the left in Figure 6.19. The 2D ultrasound slices were stacked so that a sample size of $24 \cdot 24$ px with three channels is generated (2.5D data). As can be seen in Figure 6.19, by limiting the data to the midline slices of the 3D patch, a large part containing potentially important volumetric information is neglected. Especially, information located in the corners of the 3D ultrasound patches get lost using this approach.

9-channel Data Thus, for considering more volumetric information, in another step six further 2D ultrasound slices were extracted from the diagonals of the 3D ultrasound patches as illustrated in the middle in Figure 6.19 and stacked to a sample of the size $24 \cdot 24$ px with nine channels (2.5D data).

24-channel Data Since the 9-channel approach results in a data reduction by only 62.5 %, a further 3D to 2.5D image rearrangement approach was tested where a $24 \cdot 24 \cdot 24$ vx 3D ultrasound patch sample was considered as a 2.5D ultrasound patch sample with a size of $24 \cdot 24$ px with 24 channels. As can be seen on the right in Figure 6.19, this slicing procedure can be applied to all three ultrasound volume axes (x-, y- and z-axis) so that three types of 24-channel data are generated: 24^x-, 24^y- and 24^z-channel.

[11] The deformable convolutional autoencoder and the target tracking study presented here has been submitted to *Computers in Biology and Medicine* and is still under review.

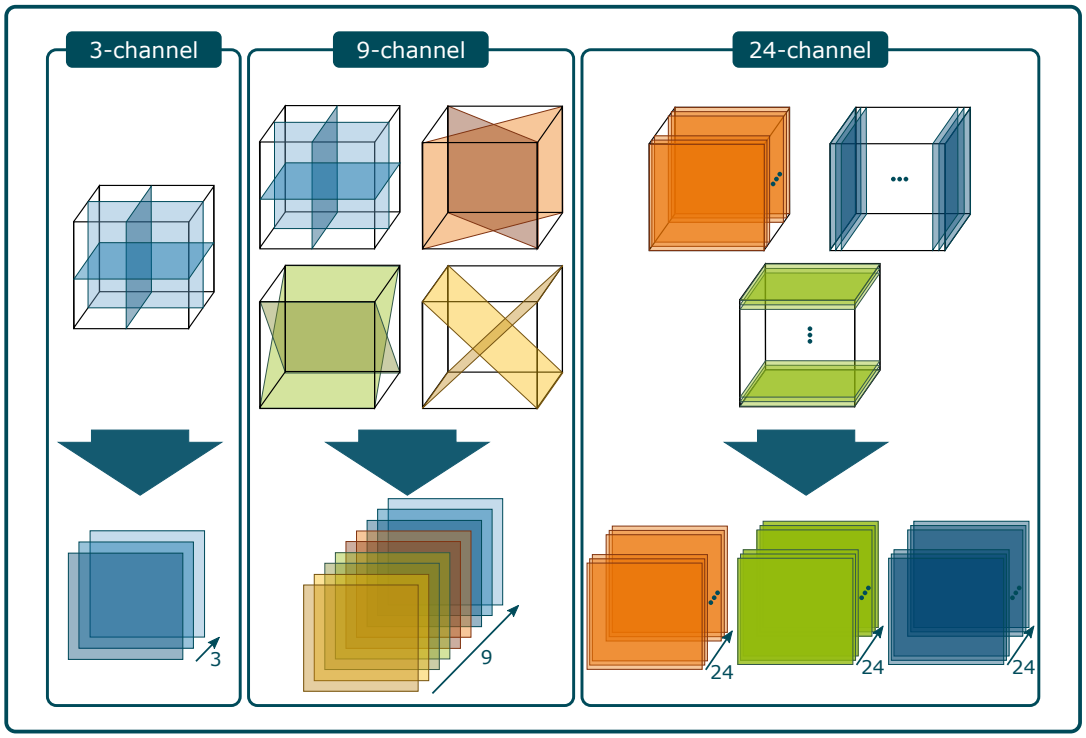

Figure 6.19: Illustration showing the slicing procedures for creating the 3-channel (left), 9-channel (middle) and 24-channel (right) 2.5D ultrasound patch data from 3D ultrasound patches of the size $24 \cdot 24 \cdot 24$ vx. In the 24-channel technique three different ways of slicing are used according to the axis where to slice producing 24^x-, 24^y- and 24^z-channel 2.5D ultrasound patches.

These three approaches are considered in parallel in the following. Altogether, five training data sets $\mathcal{T}_c$, where $c \in \{3, 9, 24^x, 24^y, 24^z\}$ is the number of channels, containing 210,000 2D ultrasound patch samples each were generated.

Autoencoder Training

In this study, it is investigated whether using deformable convolutional layers is beneficial for tracking deformable anatomical structures in 4D ultrasound data. The deformable convolution, proposed by Dai et al. [190], is an extended form of the conventional convolution (given in equation 2.6) between an input image $\mathbf{I}$ and a convolution kernel $\mathbf{K}$. For each element in the convolution kernel, a trainable offset $\Delta\mathbf{p}_n \in \mathbb{R}^2$ is added that allows to shift the kernel elements independently in the image. The formulation for the deformable convolution is given by

$$y(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in R} \mathbf{K}(\mathbf{p}_n) \cdot \mathbf{I}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n) \tag{6.19}$$

where R denotes the region to be considered in the image $\mathbf{I}$. Since a 2D convolutional autoencoder is used in this study, the offset consists of two values for each kernel element defining the shift along the two image axes x and y. Therefore, in comparison to the conventional convolution the number of trainable weights increases when

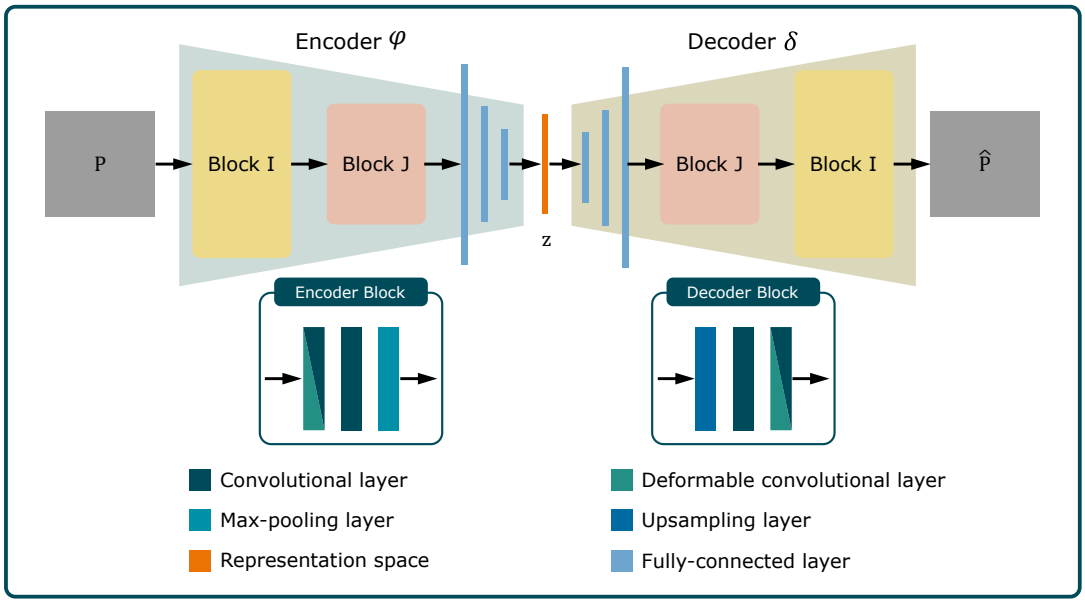

Figure 6.20: General architecture of the autoencoder consisting of two convolution blocks each in the encoder and decoder part. Each block consists of a convertible convolution layer, which can either be a deformable convolution or a conventional convolution, and a conventional convolution layer plus a max-pooling layer or an upsampling layer.

using deformable convolution[12].
The general architecture of the autoencoder used in this study is illustrated in Figure 6.20. The encoder φ maps the input data $\mathbf{P} \in \mathbb{G}^{24\times24\times c}$, where c denotes the number of channels, into a latent representation vector $\mathbf{z} \in \mathbb{R}^{128}$, containing only the most important features, and the decoder δ reconstructs the input data. Here, the encoder and decoder are constructed symmetrically as illustrated in Figure 6.20, containing two convolutional blocks each. A block consists of a conventional convolution layer, a convertible convolution layer and a max-pooling or upsampling layer. A convertible convolution layer means that this layer is either a conventional convolution layer or a deformable convolution layer. In the experiments, four architectures $M^{i,j}$ with different layer arrangements are investigated where $i, j \in I = J = \{0, 1\}$ indicate whether the convertible convolution layers in the blocks I and J are a deformable convolution $(i, j = 1)$ or a conventional convolution $(i, j = 0)$. In the bottleneck, the autoencoder encodes the input data into a representation vector with a size of 128. The whole training procedure is unsupervised and was based on the training data $\mathcal{T}_c$. However, for ensuring unbiased autoencoders in the tracking experiments, ultrasound patches coming from the same sequence as the labeled test sequence were omitted in autoencoder training. Training was performed using the Adam optimizer, a batch size of 512, a validation split of 20 % and the MSE loss. Early stopping regularization was used to avoid overfitting so that the training process is stopped if no improvement can be measured on the validation data for 10 epochs. The autoencoder training process was executed on an NVIDIA RTX 3060 GPU. This procedure was done on the basis of 3-channel, 9-channel as well as all

[12]The implementation of the deformable convolution layer used in this work is available in `https://github.com/developer0hye/PyTorch-Deformable-Convolution-v2`

24-channel ultrasound data separately. To perform target tracking in representation space, only the encoder part of the autoencoders was used as this model part maps the ultrasound patches into the representation space.

Tracking Experiments

For evaluating the proposed tracking approach, experiments were performed where the TE as well as the runtime per ultrasound image were measured. The tracking experiments were executed for the 3-channel, 9-channel and all 24-channel data approaches. The greedy target tracking algorithm presented in Algorithm 3 was used here including the sub-voxel accuracy. For determining the distance between latent representation vectors the cosine distance was applied (see equation 4.14). Tracking was performed on an Intel i5 12400F CPU, while encoder prediction was performed on an NVIDIA RTX 3060 GPU. The required runtime between getting a new 3D ultrasound image from the sequence until finding the target position was measured. Since the experiments were performed in an offline setup, meaning the ultrasound images were not streamed from the ultrasound station but loaded from the storage, the latency arising from ultrasound image acquisition and transfer between the ultrasound station and the work station was excluded from the measurements. In an online setup, this latency defines the runtime requirement to the target tracking algorithm to be real-time capable. Altogether, 360 tracking experiments were performed where four different autoencoder architectures ($M^{i,j}$ with $(i,j) \in \{(0,0),(0,1),(1,0),(1,1)\}$) and five different 2.5D data formats (3-channel, 9-channel, 24^x-channel, 24^y-channel and 24^z-channel) were investigated.

Tracking Results

The TEs were measured and summarized over all test sequences for the four different autoencoder architectures separately. The results of the tracking experiments are summarized in Table 6.10. To get a general baseline, a TE of 7.73 ± 8.23 mm was determined for the *no tracking* case. Here, the initial target position was considered to be stable in all following ultrasound images. Thus, these results provide an indicator for the error affecting the treatment when no tracking is applied.

3-channel Results In the 3-channel experiments, the autoencoders $M^{0,1}$ and $M^{1,0}$ performed similarly to the autoencoder $M^{0,0}$ with no significant improvement (p-value = 0.64 and p-value = 0.90, according to t-test). However, the autoencoder $M^{1,1}$ performed significantly better than $M^{0,0}$ (p-value < 0.05) with a mean TE of 1.73 ± 1.06 mm. The results show that reducing the data dimensionality from 3D to 2.5D by using three orthogonal midline slices can improve the target tracking performance. The performance could even be significantly (p-value < 0.05) further improved by 6.5 % by adding deformable convolution layers when comparing the $M^{0,0}$ and $M^{1,1}$ autoencoders in the 3-channel approach. The results achieved in these experiments are comparable to the TE of 1.63 ± 1.04 mm obtained by Huang et al. [15] who proposed the 3-channel procedure. In addition, it could be shown that using the 2.5D approach outperforms the mean TE of 2.58 ± 3.15 mm achieved in section 6.2.3 using the same tracking algorithm but 3D autoencoders.

Table 6.10: Overview over the TEs measured in the tracking experiments using DCAE with different architecture configurations. Tracking was performed in the sequences L-Vi-1 and L-Vi-2 with $i \in \{1, 2, 3\}$ for the landmarks Lia, Lib and Lic. The best values are highlighted in bold.

Architecture (i, j)/ Method	**Tracking error (mm)**			**Mean TE range (mm)**	**landmarks (%) where**		**Mean runtime per image (ms)**
	Mean ± SD	**95%ile**	**Max**		**TE > 3 mm**	**TE > 5 mm**	
No tracking	7.73 ± 8.23	28.48	35.25	[3.81, 23.59]	67.72	48.37	-
3-channel							
(0,0)	1.85 ± 1.23	4.19	11.67	[1.40, 2.48]	11.96	2.80	5.10
(0,1)	1.86 ± 1.44	3.76	14.99	[1.50, 2.39]	12.05	2.60	3.20
(1,0)	1.81 ± 1.52	3.53	23.14	[1.49, 2.14]	9.07	1.74	**3.03**
(1,1)	1.73 ± 1.06	3.56	9.44	[1.40, 2.09]	9.45	1.45	3.50
9-channel							
(0,0)	1.77 ± 1.36	3.61	11.77	[1.41, 2.41]	9.06	2.22	5.85
(0,1)	1.74 ± 1.40	3.60	14.23	[1.36, 2.24]	8.10	2.51	4.06
(1,0)	1.78 ± 1.52	3.72	12.06	[1.38, 2.72]	9.06	2.41	4.12
(1,1)	**1.58 ± 0.87**	**3.17**	**6.14**	[**1.37**, 2.04]	**6.46**	**0.39**	4.00
24^{x}-channel							
(0,0)	1.84 ± 1.23	4.29	10.51	[1.50, 2.26]	13.89	2.70	5.84
(0,1)	1.84 ± 1.24	4.35	9.91	[1.55, 2.50]	14.27	2.60	3.38
(1,0)	1.70 ± 1.08	3.55	7.92	[1.45, 2.22]	9.64	1.64	3.56
(1,1)	1.70 ± 1.02	3.49	7.48	[1.49, 2.22]	9.35	1.35	3.81
24^{y}-channel							
(0,0)	1.78 ± 1.15	4.07	8.65	[1.54, 2.00]	11.57	1.74	6.46
(0,1)	1.75 ± 1.13	3.86	9.94	[1.50, 2.09]	10.61	1.54	3.85
(1,0)	1.72 ± 1.09	3.66	9.18	[1.53, 1.99]	10.22	1.64	3.39
(1,1)	1.72 ± 1.13	3.52	9.57	[1.54, **1.97**]	10.13	1.64	3.69
24^{z}-channel							
(0,0)	1.75 ± 1.08	3.93	7.05	[1.38, 2.81]	12.25	1.25	6.92
(0,1)	2.00 ± 1.60	4.78	13.53	[1.39, 4.04]	16.97	4.44	3.94
(1,0)	1.77 ± 1.20	3.95	11.51	[1.44, 2.55]	12.34	2.22	3.58
(1,1)	1.96 ± 1.28	4.71	9.49	[1.41, 3.36]	17.45	3.86	4.19

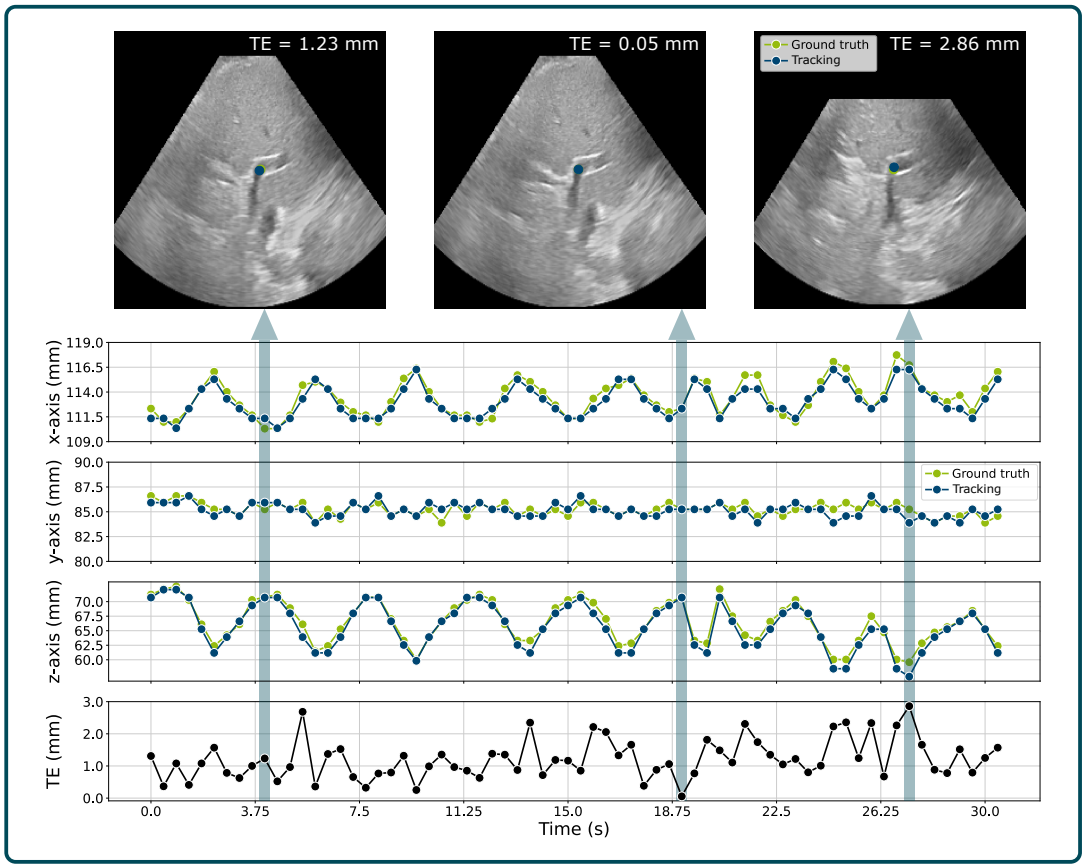

Figure 6.21: The tracking performance on the test sequence L-V2-1 using the 9-channel autoencoder $M^{1,1}$ along the x-, y- and z-axes (top three plots) and the measured TE (bottom plot). In this example, a TE of 1.21 ± 0.62 mm, a maximum TE of 2.86 mm and a 95%-tile of 2.34 mm were measured. On the top, the tracking result and the landmark are visualized in x-y-slices from three different time steps.

9-channel Results The same trend could be seen in the 9-channel experiments. The autoencoders $M^{0,1}$ and $M^{1,0}$ did not perform significantly better than $M^{0,0}$ (p-value = 0.64 and p-value = 0.90), but $M^{1,1}$ did (p-value < 0.05). By using the autoencoder $M^{1,1}$ and the 9-channel approach, the best tracking performance of all experiments was measured with a mean TE of 1.58 ± 0.87 mm. This autoencoder performed significantly better than the 9-channel $M^{0,0}$ autoencoder (p-value < 0.05). It also performed significantly better than the 3-channel and all 24-channel $M^{1,1}$ autoencoders (p-value < 0.05 for all). Additionally, the lowest TE > 3 mm and TE > 5 mm ratios of 6.46 % and 0.39 % were observed using $M^{1,1}$ in the 9-channel experiments. In Figure 6.21, the tracking performance of $M^{1,1}$ using the 9-channel approach in the test sequence V2-1 is visualized. In this example, a TE of 1.21 ± 0.62 mm and a maximum TE of 2.86 mm were measured. Additionally, the landmark (green) and the tracking result (blue) in slices of three different 3D ultrasound images from different time steps with the corresponding TE are illustrated. The results indicate that using the 9-channel approach outperforms the 3-channel approach. This could be caused by the high ratio of neglected volumetric information in the 3-channel approach. Using only three 2D slices instead of fully

$24 \cdot 24 \cdot 24$ vx 3D ultrasound patches corresponds to an image voxel loss of 87.5 %. Especially information in the corners of the volumetric patches is neglected. Due to the low volumetric information content, it gets difficult to identify deformable volumetric motion. By using nine 2D slices, more volumetric information is considered as the image voxel loss is reduced to 62.5 %. However, the procedure to get the 9-channel data leads to an over-representation of central voxels of the 3D ultrasound patch compared to outer voxels due to the fact that all cut slices intersect in the 3D patch center (see Figure 6.19). Although no drawbacks could be observed in the experiments due to this over-representation, it might be possible that a different 3D to 2.5D patch slicing strategy leads to better results. In addition, compared to the baseline autoencoder $M^{0,0}$, the tracking performance could be improved significantly by 10.7 % (p-value < 0.05) using the autoencoder $M^{1,1}$ in the 9-channel approach. This confirms that adding deformable convolution layers is beneficial for tracking deformable targets.

24-channel Results Considering the 24-channel experiments, the results vary. In the 24^y-channel experiments, no significant difference could be identified between the results when using different autoencoders (p-values > 0.05 for all). Furthermore, in the 24^z-channel experiments the highest mean TE of 2.00 ± 1.60 mm for $M^{1,0}$ and 1.96 ± 1.28 mm for $M^{1,1}$ were measured. It can be seen that the highest TE > 3 mm ratios of up to 17.45 % were measured in the 24-channel experiments. The results of the 24^x-, 24^y- and 24^z-channel experiments show different trends. In the 24^z-channel experiment, the highest mean tracking errors were observed indicating that slicing along the $z-$axis of the ultrasound volumes is disadvantageous. Furthermore, no tracking performance improvement could be observed by adding deformable convolutional layers. This was also observed in the 24^y-channel experiments. In contrast to that, in the 24^x-channel experiments, the autoencoder $M^{1,1}$ significantly outperformed the baseline autoencoder $M^{0,0}$ (p-value < 0.05).
These findings are disadvantageous for a tracking application as the ultrasound transducer must be placed properly to get a low TE when using a 24-channel approach. This can be caused by the fact that liver motion varies in amplitude in the AP-, LR- and SI-directions of the human body. The 24-channel approach always neglects motion along one axis, the axis where the slicing was performed, because the autoencoder cannot learn motion in that direction since there is no filter kernel covering it. Using the 9-channel approach, proper transducer placement is not required as this approach equally considers all spatial axes in combination. Thus, the results indicate that using the 9-channel approach is more promising than the 24-channel approach for dimensionality reduction in a 4D target tracking task. When using the $M^{1,1}$ autoencoders, outliers greater than 10 mm were not observed in the experiments. Hence, it turned out that using nine instead of only three 2D slices in combination with using deformable convolution layers is promising for tracking of deformable targets.

Runtime The mean tracking times measured during the experiments range from 3.03 ms to 6.92 ms (144 Hz to 333 Hz). The highest tracking times were measured when using the baseline autoencoders $M^{0,0}$. This corresponds to a speed-up of up to

100 times faster than the state-of-the-art method proposed by Huang et al. [15] who reported a mean runtime of 300 ms (3 Hz) per 3D ultrasound image. The tracking algorithm proposed by He et al. [16] could run at a frame rate of up to 125 ms (8 Hz) so that the DCAE tracking algorithm is 42 times faster even though He et al. used more powerful hardware with two NVIDIA RTX 2080Ti GPUs for neural network prediction. This is especially beneficial as the runtime of the proposed method is independent from the ultrasound image size due to the local search approach. Since a target tracking task must be real-time capable, the low runtime is very promising. Due to the fact that modern 3D ultrasound machines have frame rates of 2 Hz up to more than 30 Hz [191], [192] and the proposed approach has a frame rate of up to more than 300 Hz, there is still time left that could be used for improvements or regularization.

6.3 Online Phantom Study

In the previous sections, the tracking algorithms developed in this work were tested offline with the acquired 4D ultrasound data set. Although the tracking performance could be evaluated in terms of accuracy and runtime, the usability in an online robotic ultrasound setup had not been demonstrated until to date. Hence, this section represents a first proof-of-concept study in an online application that aims to show the applicability of the DCAE-based target tracking method developed in section 6.2.4. In this experiment, a 9-channel $M^{1,1}$ DCAE trained with the 4D ultrasound liver data set was used and no fine-tuning was executed. The robotic ultrasound setup introduced in section 3.1 was assembled and combined with an ultrasound phantom that was moved by a hexapod robot.

6.3.1 Experiment Setup

The experiment setup is visualized in Figure 6.22. It consists of two different robots, a 7-DoF robot arm and a parallel hexapod robot, as well as an ultrasound machine, an ultrasound phantom and a workstation.

Ultrasound Machine

An *EPIQ 7* ultrasound machine with an *X6-1* matrix array transducer (Philips Healthcare, Best, Netherlands) was used for 4D ultrasound imaging. The ultrasound transducer was attached to an *LBR iiwa 14 R820* robot arm (KUKA, Augsburg, Germany) with a custom transducer holder mounted on the robot end-effector. The ultrasound machine settings such as gain and depth were determined empirically in preliminary tests and set fixed during the experiment. The imaging frame rate was 4 Hz and the ultrasound images had a size of $224 \cdot 243 \cdot 167$ vx with a voxel spacing of $0.7 \cdot 0.55 \cdot 0.94\,\mathrm{mm}^3$.

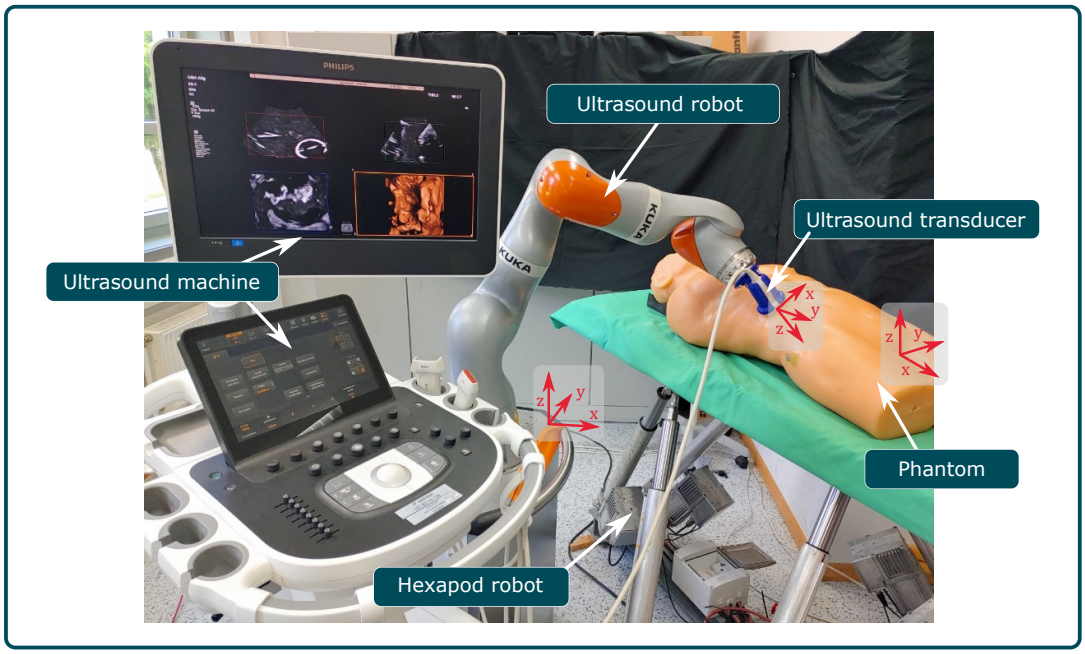

Figure 6.22: The setup used for the online ultrasound tracking experiment. The *EPIQ 7* ultrasound machine shows 3D ultrasound images in real-time acquired with an *X6-1* ultrasound transducer that was placed and guided on the phantom by a 7-DoF *LBR iiwa 14 R820* robot arm. The phantom was placed on a custom hexapod robot that moved the phantom in a breathing-induced motion pattern.

Workstation

For getting access to the acquired 3D ultrasound images, the ultrasound machine was connected to a workstation via Ethernet connection. The workstation was equipped with an Intel i7-12700 CPU and an NVIDIA RTX 3060 GPU. On the workstation, the *PLUS Toolkit* application [193], [194] using *OpenIGTlink* [195] was executed to stream the ultrasound image data in real-time. The ultrasound images were processed by performing target tracking on the workstation. In addition, the workstation was used for controlling the *LBR iiwa 14 R820*.

Phantom

The phantom used in this experiment was a human torso *Blue Phantom* (Elevate Healthcare, Florida, US) manufactured for ultrasound imaging practice. The phantom contains some organs such as heart, prostate and kidneys but it does not contain a liver. Hence, target tracking was performed at the right kidney in the experiment.

Hexapod

In order to make the phantom move in a pattern similar to breathing-induced motion, it was placed on a custom hexapod robot consisting of six actors working in a parallel kinematic [196]. The ultrasound phantom was placed on a plate mounted on top of the actors so that the AP-, LR- and SI-directions of the phantom approximately corresponded to the z-, y- and x-axes of the hexapod robot, respectively.

The hexapod robot has a workspace of size $27,000\,\mathrm{cm}^3$ and can apply force of up to 2,000 N which is sufficient for the purpose. It was controlled to execute a movement that is similar to a breathing-induced motion pattern. Therefore, a 3D sinus pattern was implemented that is defined by

$$g_{motion}(t) = \mathbf{a} \cdot \sin(2\pi \cdot f \cdot t) \tag{6.20}$$

where $\mathbf{a} = (-4.0, 1.8, 8.0)^T$ denotes the amplitude of the motion trajectory (in mm), $f = 0.25$ is the frequency (in Hz) and t is the time step (in s). As can be seen in Figure 6.22, the x-, y- and z-axes of the hexapod robot corresponded to the SI-, LR- and AP-direction of the phantom, respectively.

Ultrasound Robot

The *LBR iiwa 14 R820* robot arm (ultrasound robot) was used to hold the ultrasound transducer in continuous contact to the phantom surface. Due to internal force torque sensors, the robot is able to estimate the force applied to the end-effector and to the ultrasound transducer. On the basis of this force information, the robot was controlled in an impedance control mode during the experiment. It was forced to press the ultrasound transducer to the phantom body with a target force F_{target}, but to be compliant if it was pushed with a higher force or if the applied force decreased below this value. With this behavior the ultrasound transducer could be held in continuous contact to the phantom with a target force of $F_{target} = 10\,\mathrm{N}$. This target force was chosen according to the force applied in the data set acquisition study (see section 3.3).

6.3.2 Experiment Workflow

The experiment was initiated by starting the hexapod motion trajectory. Thus, the phantom moved continuously in a simulated breathing-induced motion pattern. The ultrasound image streaming procedure was then activated and the ultrasound transducer initially placed on the phantom surface by using the hand-guidance-mode of the ultrasound robot. Therefor, an operator pushed the robot arm to the required position. When the ultrasound transducer was correctly aligned, the ultrasound robot was brought into the impedance control mode. In this mode, the ultrasound transducer was pressed on the phantom surface compensating the motion pattern induced by the hexapod robot. Finally, the target tracking procedure was started. First, a target was manually selected by an operator in the streamed 3D ultrasound images on the workstation. The landmark selected in this experiment is visualized in Figure 6.23. After target selection, the DCAE-based tracking algorithm proposed in section 6.2.4 was applied to track the selected target within the ultrasound images. The complete tracking procedure, including target selection and target tracking, was limited to a 10 s interval. The tracked target positions and the ultrasound images, as well as the ultrasound robot positions and force measurements were stored in order to evaluate the tracking experiment. In addition, the runtime required for ultrasound image acquisition and streaming as well as for target tracking was stored.

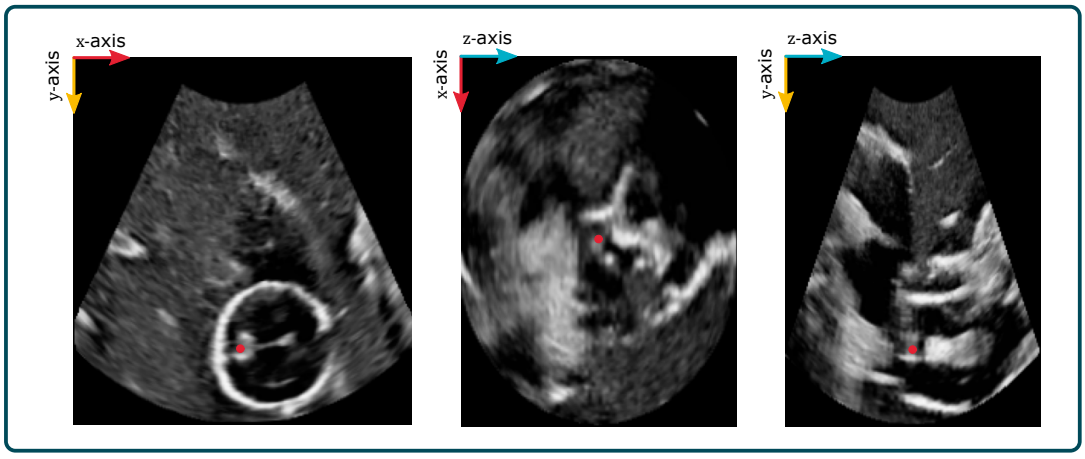

Figure 6.23: Slices in the xy-, xz- and yz-planes of the ultrasound image in which the target to track (marked in red) was selected and labeled by the observer in the experiment.

6.3.3 Results

As illustrated in Figure 6.22, the ultrasound transducer, the ultrasound robot and the hexapod robot have their own coordinate systems that were not calibrated to each other in this experiment. Hence, the motion data acquired and stored are not one-to-one comparable. However, since the purpose of this experiment was to show the applicability of the tracking algorithm in an online setup, the evaluation can be done with the acquired data. In the 10 s interval, about 2 s were spent on target selection, while the remaining 8 s were spent on target tracking. In this time span, 29 3D ultrasound images were acquired, streamed and processed.

Ultrasound Robot Motion

The motion trajectory of the ultrasound robot as well as the force applied to the end-effector that were measured during the experiment are visualized in Figure 6.24. It can be seen that the force the transducer was applied with to the phantom could be kept in an acceptable range around the target force $F_{\text{target}} = 10\,\text{N}$. For the 8 s interval, the force was $11.2 \pm 1.6\,\text{N}$, on average. The z-axis of the ultrasound transducer points into the phantom body (approximately AP-direction of the phantom body), while the x- and y-axes point approximately into the LR- and SI-directions of the phantom body, respectively. The movement of the ultrasound transducer follows a sinusoidal trajectory in all spatial axes, which can be seen in Figure 6.24. The highest motion range was observed in the z-axis of the ultrasound transducer. This indicates that the motion applied by the hexapod robot was successfully compensated by the ultrasound robot. In addition, a drift can be observed in the movement in all spatial axes. The ultrasound robot consistently caused the transducer to drift in one direction on the phantom. This was due to the fact that the ultrasound robot was only controlled to keep the force at the end-effector at F_{target}, but it was not controlled to stay at a target position. This behavior could be avoided by adding a target following procedure to the ultrasound robot control algorithm, e.g., by following the tracked target.

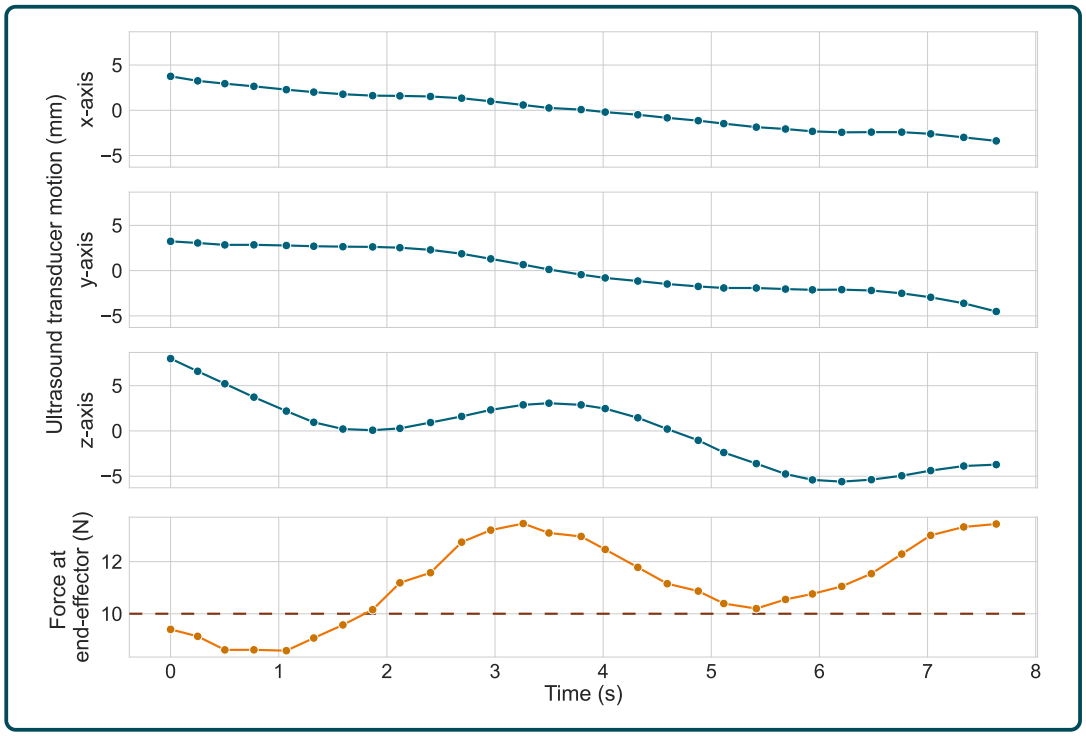

Figure 6.24: The motion trajectory of the ultrasound transducer in all spatial axes (top three plots) and the force measured at the end-effector of the ultrasound robot (bottom plot). The target force $F_{\text{target}} = 10\,\text{N}$ is marked with the dashed line.

Landmark Motion

To evaluate the tracking performance, the target selected and tracked in the online experiment was retrospectively labeled by a 3D ultrasound expert observer. The observer labeled each 3D ultrasound image in the sequence four times. Based on these labels, an intra-observer variability of $1.59 \pm 0.39\,\text{mm}$ was determined in the same way as presented in section 3.3.2. To overcome this labeling variability, the ground truth target position for each 3D ultrasound image was selected by taking the average of all four landmark annotations. The resulting ground truth target trajectory and the tracked trajectory are visualized in Figure 6.25. The TE was determined based on the tracked and the ground truth target position and is also presented in Figure 6.25 for each ultrasound image in the sequence. On average, a low TE of $0.92 \pm 0.38\,\text{mm}$ with a range of [min, max]=[0.20 mm, 1.75 mm] was measured in the 8 s sequence. The results indicate that the DCAE-based tracking algorithm was able to accurately track a kidney target in the phantom even though the DCAE was not trained for this domain. Furthermore, the drift applied by the ultrasound robot can also be observed in the motion trajectory of the landmark in Figure 6.25. This could be expected since the target position within the ultrasound image depends on the ultrasound transducer position on the phantom.
To evaluate the runtime of the tracking algorithm, the time required to locate the target within the 3D ultrasound image and the time required for the streaming procedure to deliver the next ultrasound image was measured. The time span between two consecutive ultrasound images was 273 ms (3.7 Hz), on average. In contrast,

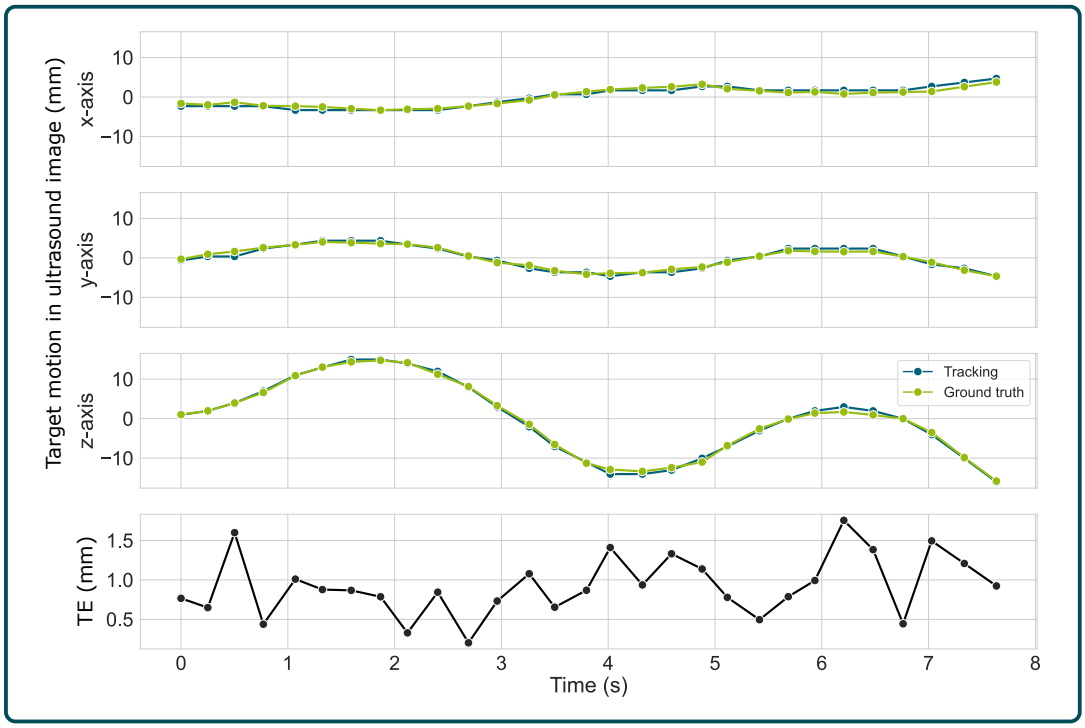

Figure 6.25: The target motion in all spatial axes in the 8 s 3D ultrasound sequence acquired in the online experiment (top three plots). The ground truth target motion (green) was labeled by an expert observer. The tracked target motion (blue) was generated by applying the DCAE-based target tracking algorithm. In addition, the TE for each ultrasound image is presented in the bottom plot.

the tracking algorithm performed at an average runtime of 6.2 ms (161 Hz). Hence, the DCAE-based tracking algorithm could be performed in real-time in this online experiment.

In this study, the applicability of the DCAE-based tracking algorithm was shown in an online proof-of-concept experiment. It was shown that tracking can be applied in real-time with a low tracking error. The setup presented in this experiment can be used to perform a volunteer study where the hexapod robot and the phantom are replaced by healthy humans or even by patients.

6.4 Summary

Target tracking in 4D ultrasound is a challenging task due to different requirements such an algorithm must meet to be usable in a therapy guidance application. In this chapter, different tracking approaches have been developed and implemented in order to investigate the performance of representation learning-based tracking algorithms in comparison to image intensity-based tracking algorithms. Hence,

Research Question 4
Can real-time target tracking in 4D ultrasound be performed in representation space and can it outperform image space-based tracking algorithms?

has been addressed in this chapter. A naive but simple approach for target tracking is template matching which is commonly used in literature [163], [166], [180]. Though, it has been reported that tracking deformable targets such as soft tissue structures in ultrasound images cannot be reliably tracked using a template matching algorithm [165]. This was confirmed in the experiments conducted in this work. It has been observed that a simple template matching algorithm produces tracking failures that lead to temporal or even complete loss of the target. To overcome these failures, a novel target appearance forecasting approach has been developed based on a spatio-temporal autoencoder consisting of convLSTM cells. It could be observed that forecasting the appearance of the target and using this as reference in template matching improves the tracking performance. The findings indicate that a target forecasting approach such as using the proposed spatio-temporal autoencoder can be useful for a target tracking algorithm. An advantage is that the approach can be combined with any target tracking algorithm, and is not limited to template matching.
Another tracking approach in image space was implemented using a DLN. The DLN was trained in a supervised and target specific way in this work due to the limited variance of the labeled target data. However, in literature similar but more complex approaches, such as SNNs, are proposed that are trained in a generalized way [15], [178], [197]. The results showed that a DLN is able to track a target in 4D ultrasound in real-time with a TE lower than using template matching. However, the problems of non-generalizability and the fact that such networks are trained in a supervised way remain.
The main purpose of this work was to develop and investigate representation learning-based target tracking. Hence, the first image feature-based 4D target tracking algorithm based on binary feature descriptors was developed. It was observed that the algorithm could run in real-time and track the target with high accuracy. The tracking accuracy could even be improved by replacing the binary descriptor with a SWAE trained in an unsupervised way.
In order to provide a fast algorithm that is real-time capable even at imaging frame rates higher than 4 Hz, a novel greedy tracking algorithm was developed and combined with autoencoder-based feature learning approaches in this work. Due to the local search characteristic implemented in a hill-climbing way, the greedy tracking algorithm could be executed at a frame rate of up to 250 Hz in 4D ultrasound tracking experiments. Additionally, the algorithm runtime is independent from the 3D ultrasound image size due to the local search characteristic.
Similarly to the findings in chapter 4, it was observed that SWAEs trained in a generalized way outperformed individually trained SWAEs. The tracking performance could be improved by fine-tuning the generalized SWAEs. This is a promising finding, as it suggests that with a sufficiently large data set, it is possible to train a generalized autoencoder for a wide range of patients, eliminating the need to train

Table 6.11: Summary of the characteristics of the 4D ultrasound tracking approaches presented in this work according to the requirements defined for a target tracking algorithm to be usable for therapy guidance purposes in radiation therapy.

Method	Type of learning	Generalizability	Real-time capability (Hz)	Robustness	
				Mean TE (mm)	TE>5 mm (%)
Requirement	Unsupervised	Yes	>4	< 5.00	0.00
Image space					
Template matching	None	Yes	<1	7.55	28.21
Spatio-temporal AE	Supervised	No	<1	2.85	15.10
DLN	Supervised	No	15	2.23	1.46
Representation space					
Image features	None	No	3.9	2.08	3.28
SWAE	Unsupervised	Yes	-	2.58	4.67
DCAE	Unsupervised	Yes	250	1.58	0.39

an individual autoencoder for each patient. However, it may be advantageous to individually fine-tune such an autoencoder with a small patient-specific data set. In these experiments four manually selected target reference patches were used during tracking to cover the full range of target appearance changes due to deformation. To eliminate this need, a DCAE containing deformable convolutional layers has been developed in this work, which aims to learn compensating deformable target motion. It was observed that replacing conventional convolutional layers with deformable convolutional layers improves the tracking performance. In addition, a novel 3D to 2.5D ultrasound patch slicing strategy was developed where a 3D ultrasound patch is mapped into a 2D ultrasound patch with nine channels. The results showed that this novel approach outperforms the approach proposed by Huang et al. [15] who only used three 2D ultrasound slices to form a 2.5D ultrasound patch with three channels.
In addition, it was shown that the DCAE-based tracking algorithm can be executed in an online application using an *EPIQ 7* ultrasound machine and an *iiwa 14 R820* robot arm in real-time with high tracking accuracy. The first proof-of-concept online 4D ultrasound tracking experiment was performed in this work showing that representation space-based 4D ultrasound tracking is promising and realizable.

6.4.1 Method Comparison

Different tracking approaches executed in image or in representation space have been developed and tested in this work. To evaluate whether these approaches meet the defined requirements, their properties are summarized in Table 6.11. Here, the variants with the best performance, meaning the lowest mean TE measured in the experiments are presented. In the following, these results are evaluated in order to answer the research question.

Type of Learning

As can be seen in Table 6.11, the tracking algorithms performed in image space were trained in a supervised way (spatio-temporal autoencoder and DLN) or even did not

need any training (template matching). This conformity can also be observed when considering other approaches proposed in literature. Using non-training approaches such as template matching [17], [180], [198] or supervised learning approaches such as SNNs [15], [16], [177], [178], [188], [197], [199], [200] are much more common than using unsupervised ones like image features [18], [157], [166]. Using an unsupervised learning approach has the advantage of not requiring data labeling, potentially saving time that physicians spend on labeling in a supervised learning approach. This fact is advantageous for implementing the approach into the clinical workflow as the training procedure can be implemented in a fully automatic way, e.g., for performing fine-tuning or generating a patient individual neural network.

Generalizability

The generalizability requirement could not be met by the spatio-temporal autoencoder, the DLN and the image feature-based approach for different reasons. The spatio-temporal autoencoder was trained specifically for every target due to the small amount of different available labeled targets. However, since a generalized spatial autoencoder could be generated in this work (see section 4.2.1), training a generalized spatio-temporal autoencoder might also be possible if a sufficiently sized training data set was available. The DLN was trained in a target specific way in this work due to a limited variance in labeled target data for training. Though, it has been shown in literature that such a neural network can be trained in a generalized way [15], [178]. In contrast, the image feature-based tracking approach could not meet the generalizability requirement due to the fact that different hyperparameters need to be set manually. These hyperparameters are highly dependent on the ultrasound image itself, so changes, e.g., in image brightness or contrast, can lead to feature detection failures, as FAST-3D requires an image intensity threshold to be set. To be able to eliminate this issue, it needs to be investigated how to determine the optimal hyperparameter settings from the ultrasound image. Another approach could be using the ultrasound machine settings to derive the optimal hyperparameter settings. However, this needs to be investigated in future studies.

Real-Time Capability

The real-time capability requirement is one of the most critical ones as it clearly defines whether the tracking algorithm is even usable for the purpose of therapy guidance. The runtime of the algorithm must be less than the time required for the ultrasound machine to generate the next ultrasound image. The 4D ultrasound data used in this work were acquired at a frame rate of 4 Hz denoting the lower bound for the tracking algorithm to be real-time capable. However, ultrasound machines that are able to acquire 4D data at frame rates of up to more than 20 Hz are already available [12], [201]. Even though the runtime of an algorithm can also be improved, e.g., by increasing the hardware power or optimizing the implementation, the runtime measured in this work is an indicator for real-time capability of the approaches in general. In addition, simply adding more powerful hardware to such a system would also increase the costs of the system.
It was found that template matching-based tracking approaches could not meet the

real-time requirement. This could be caused by the CPU-based implementation used in this work. Approaches to make template matching real-time capable have already been proposed for 2D images [163], [167]. This indicates that performing template matching in 3D images could potentially be achieved by using a GPU-based optimized implementation and powerful hardware. In contrast, the feature-based tracking algorithms proposed in this work could be executed much faster. The image feature-based approach using a SWAE for feature description could be performed at an average frame rate of 3.9 Hz which is nearly real-time for the data set. Hence, by implementing a GPU-based version of the algorithm, real-time capability could easily be achieved.
The DCAE-based approach is clearly real-time capable with a frame rate of 250 Hz due to the local greedy-based tracking algorithm developed in this work. Due to this very low runtime there is time left to perform regularization, e.g., for compensating large target displacements as observed in some experiments, or to track multiple targets in parallel.

Robustness

According to the AAPM, motion management is required in radiation therapy if the target moves for more than 5 mm [19]. Therefore, this is set as the upper TE bound for the robustness requirement. However, this boundary should not be considered a limit for defining a well-performing tracking algorithm, as it only indicates a motion magnitude at which motion management should be applied to ensure high quality radiation therapy. A tracking algorithm should be more reliable than this bound to improve the treatment accuracy. In radiation therapy, the area to irradiate is planned according to different factors such as geometrical deviations. When a tracking algorithm is available that is able to track the tumor precisely, the geometrical deviation can be reduced and, subsequently, the area to irradiate can be minimized [202]. Hence, less healthy tissue would be damaged. In addition, in radiation therapy, the treatment system has a latency that must be compensated for by predicting the actual target position based on the tracked position, which can have uncertainties. [203]. To minimize this uncertainty in target position prediction, the tracking error must be minimized.
In the experiments, it could be observed that only the template matching approach could not meet this robustness requirement considering the average TE. By using the spatio-temporal autoencoder for target appearance prediction, the TE could be decreased indicating that proper target reference selection is a critical aspect for this approach. This has also been determined by De Luca et al. [165] who proposed a learning-based template matching algorithm. Even though the approaches could meet the robustness requirement according to the mean TE adequately, the TE>5 mm rate indicates there is space for improvement left in most approaches. Template matching and the spatio-temporal autoencoder-based approach have the highest TE>5 mm rates with 28.21 % and 15.10 %, respectively, indicating they are not reliably robust. This could be due to the fact that template matching is image based and finding a proper target reference is difficult due to high dimensional motion patterns and low image quality. In contrast, the DLN had a low TE>5 mm rate of 1.46 % which might be the reason why localization networks such as SNNs

are often used in literature. In representation space tracking, the image feature-based approach has a TE>5 mm rate of 3.28 % indicating it is more robust than the SWAE-based tracking approach with a TE>5 mm rate of 4.67 %. In the DCAE-based tracking approach the smallest TE>5 mm rate of 0.39 % was measured indicating this is a promising robust tracking approach with a low failure rate.

Considering these findings, the research question could be answered. 4D ultrasound target tracking can be performed in real-time in frame rates of up to 250 Hz using the proposed DCAE approach. The representation space-based target tracking approaches could partially outperform the image space-based approaches implemented in this work. Therefore, representation space-based target tracking is a promising approach to further develop a robotic ultrasound system for therapy guidance in radiation therapy.

7 DISCUSSION

The purpose of this work was to investigate whether feature-based target tracking in 4D ultrasound can be performed precisely and to develop such approaches based on representation learning. It has been investigated if 3D ultrasound patches can be mapped into representation vectors that contain abstract information describing the most important features of the patch. In different experiments, the performance of image space and representation space-based target tracking algorithms were evaluated and, subsequently, compared to each other. The focus of this work was on the applicability of representation learning for 4D ultrasound target tracking. Hence, the image space-based methods implemented in this work are simple approaches that serve as a baseline. There are other image space-based approaches proposed in the literature that are more complex and may perform better. However, most of them have been developed and evaluated only for 2D+t ultrasound. 4D ultrasound tracking approaches proposed in the literature are also solely image space-based. Huang et al. [15] and He et al. [16] both proposed SNNs, Royer et al. [18] used a deformable transformation model and Banerjee et al. [17], [156] used a template matching-based approach. In Table 7.1, the characteristics of these methods are summarized according to the requirements of the tracking algorithm defined in this work. As can be seen, the DCAE-based tracking approach proposed in this work is comparable to state-of-the-art methods in terms of robustness, as a low TE of less than 2.00 mm was determined for all methods. In the type of learning and the real-time capability requirements, the DCAE-based tracking approach even outperforms

Table 7.1: Overview of state-of-the-art methods for 4D ultrasound tracking performing in image-space according to the requirements defined for a target tracking algorithm to be usable for therapy guidance purposes in radiation therapy.

Method	**Type of learning**	**Generalizability**	**Real-time capability (Hz)**	**Robustness**	
				Mean TE (mm)	**TE>5 mm (%)**
Requirement	Unsupervised	Yes	>4	< 5.00	0.00
Banerjee et al. [17]	None	Yes	8	1.83	-
Royer et al. [18]	None	No*	2.9	1.74	-
Huang et al. [15]	Supervised	Yes	3.3	1.63	-
He et al. [16]	Supervised	Yes	8	1.60	-
DCAE	Unsupervised	Yes	250	1.58	0.39

*Requires manual segmentation for each target individually.

the state-of-the-art methods. In this work, the first fully unsupervised learning-based target tracking approach for 4D ultrasound was proposed. Training the SNNs require labeled data [15], [16], which is limited and difficult to obtain because it requires expert knowledge and it is time-consuming to label volumetric image data. In terms of runtime, the DCAE-based approach clearly outperforms state-of-the art methods with 250 Hz, as it is about 30 times faster than the fastest state-of-the-art methods running at 8 Hz. The approaches proposed by Royer et al. [18] and Huang et al. [15] did not even run in real-time for sequences acquired at a frame rate of 4 Hz like the sequences used in this work. Also the approaches by Banerjee et al. [17] and He et al. [16] are limited in usability according to their runtime since ultrasound machines providing frame rates higher than 8 Hz are already available [201]. The generalizability requirement could be met by all approaches except the one proposed by Royer et al. [18], as it requires manual segmentation of the target to generate a target-specific model. This comparison shows that the representation learning-based approach developed in this work performs comparable with state-of-the-art methods but brings two major advantages over them: Unsupervised learning type and very fast runtime.
The main findings of this work are discussed in detail in the following. Open questions are identified and possible solutions are suggested.

Feature Extraction

It has been identified that features can be detected and described in 3D ultrasound images using an image feature detector and descriptor such as FAST-3D and BRIEF-3D or BRISK-3D, respectively. These methods do not require learning, but hyper-parameters must be set manually which are not obvious to find. Using an SWAE instead of a feature descriptor could eliminate this issue for the description process. In a tracking experiment, the SWAE descriptor even outperformed the binary descriptors indicating autoencoders are producing representations more suitable for a tracking application. However, in literature other feature descriptors are available that are more complex than the simple binary ones applied in this work. In future studies, commonly used descriptors in 2D, such as SIFT and SURF should be extended to 3D and evaluated for use in 4D ultrasound target tracking. One challenge will be to make the algorithms real-time capable. It has been shown that these algorithms are computationally much more expensive than binary descriptors in 2D images [115].
When creating an autoencoder for extracting local features from an ultrasound image, the patch size and the size of the representation space must be defined. In this work, the optimal size of the representation space for an ultrasound patch with the size $24 \cdot 24 \cdot 24$ vx has been identified in reconstruction and tracking experiments. However, the patch size was selected manually according to the spatial resolution of the ultrasound images and the size of liver vessels. Although the autoencoders were able to reconstruct the patches with high accuracy and target tracking could be performed with low TE, it is still unclear whether this patch size is optimal. To identify the optimal patch size, a study should be conducted where different patch sizes are evaluated in tracking experiments. This should be done in combination with varying the representation space size to identify the optimal combination of

patch and representation space size.

Tracking Accuracy

An observer variability study was conducted to quantify the labeling error that defines a lower bound for the performance of the tracking algorithms. Based on the intra-observer variability measurements, labeled landmarks from the observers A2 and B1 are used as ground truth in the tracking experiments as they had the lowest intra-observer variability of $1.58 \pm 0.90\,\text{mm}$ and $1.60 \pm 1.51\,\text{mm}$, respectively ($1.59 \pm 1.24\,\text{mm}$ on average). Even though the landmarks Lic with $i \in \{1, 2, 3\}$ and the sequences L-Vi-3 with $i \in \{1, 2, 3, 4, 5\}$ were not part of the observer variability study, but used in some experiments of the tracking approaches, these variability measurements serve as an indicator for the limitation of the ability to evaluate the tracking performance.
For instance, template matching and spatio-temporal autoencoder-based tracking was evaluated based on the sequences L-Vi-3 with $i \in \{1, 2, 3, 4\}$ where no variability measure is available. However, the average TEs of $7.55 \pm 12.31\,\text{mm}$[1] and $2.85 \pm 3.15\,\text{mm}$[1] measured in these studies are clearly greater than the average labeling variability indicating that these tracking algorithms have inaccuracies. This becomes even more obvious considering the failures observed in the experiments.
In contrast, the DLN was evaluated based on the sequences and landmarks where variability measurements are available. It could be observed that the DLN outperformed template matching and the spatio-temporal autoencoder-based approach with an average TE of $2.23 \pm 1.28\,\text{mm}$. This TE is 0.64 mm greater than the labeling variability, on average.
The representation space-based tracking algorithms using image features as well as using an SWAE in combination with the greedy algorithm performed with a TE of $2.08 \pm 1.50\,\text{mm}$ and $2.58 \pm 3.15\,\text{mm}$[1] which are 0.49 mm and 0.99 mm greater than labeling variability, respectively. However, it should be noted that in the image feature-based approach the sequences L-Vi-2 and L-Vi-3 with $i \in \{1, 2, 3, 4\}$ were used for evaluation but no variability measure is available in the sequences L-Vi-3.
In the DCAE-based target tracking experiments a TE of $1.58 \pm 0.87\,\text{mm}$ was measured. Since this TE is 0.01 mm less than the intra-observer variability, it cannot be decided whether the error is due to inaccuracy of the tracking algorithm or due to labeling inaccuracy. Therefore, for the used data set this is the smallest reliable TE that could be achieved. Given the average voxel size of the data set of $1.0 \cdot 0.7 \cdot 1.5\,\text{mm}^3$, a TE of 1.58 mm means that the target is located with an error of 1 vx to 2 vx, depending on the image axis. This is a low error that is close to the lower error bound implied by the resolution of the ultrasound image, defined by the voxel size.

Failures

One reason that this low TE could be achieved in the DCAE-based approach in contrast to other approaches might be that the sequences L-V4-j with $j \in \{1, 2, 3\}$

[1] Only TE ≥ 0 was measured.

were not part of the evaluation experiments. Here, they have been neglected due to the fact that the landmarks are affected by a rib shadow artifact that causes them to temporarily disappear in the ultrasound images. This even highly complicated the labeling procedure as reported by the observers in the labeling study (see section 3.3.2). With such an unreliable ground truth evaluating the tracking performance would also not be reliable. In addition, a tracking algorithm based on a local search algorithm such as a greedy search, is not able to track a landmark that is not visible in the image. An algorithm that is based on a more global approach might be able to compensate partially disappearing parts, but this was not part of the greedy algorithm used here.
The greedy algorithm developed in this work has the advantage of being fast, providing real-time capable tracking in 4D ultrasound due to its local search characteristic. However, this also has the drawback that artifact-induced target disappearances and large target displacement events can lead to target loss failures. For these cases, solutions need to be developed to ensure robust tracking. One possible solution for both cases could be adding a global tracking method such as the image feature-based approach or a DLN or SNN to the tracking algorithm. By taking full ultrasound images or large ROIs into account, these approaches can compensate for large target displacements as well as partially disappearing structures. This could be observed in the experiments (see sections 6.1.3 and 6.2.1), where no drastically increasing TE was measured. Another approach to compensate partially disappearing targets was proposed by Royer et al. [18] who used a deformable transformation model to perform target tracking. However, this approach is computationally expensive, resulting in a low runtime (2.9 Hz on average), and it would also fail tracking if the target completely disappeared. Ultrasound image artifacts, such as rib shadows, can cause tracking failures, because they greatly affect ultrasound images. To avoid such failures, a proper ultrasound transducer alignment strategy should be applied such as proposed by Osburg et al. [67]. This could remove the task of dealing with ultrasound shadow artifacts from the tracking algorithm and simplifies the tracking process.
Another approach for compensating target displacements that are caused by patient movements could be using external motion information such as from a camera system that tracks the surface motion of the patient during treatment [204]. If this system detects a large target displacement, this motion information could be used, e.g., to adapt the ultrasound tracking algorithm or to reposition the ultrasound transducer.

Motion Invariance

It has been found that the choice of reference patch can also affect the tracking performance. One solution is to use multiple reference patches to be able to cover a wide range of target shape variance like performed in other studies in literature [180], [188]. This method has been used in section 6.2.3 using an SWAE for target tracking. Four reference patches taken from different breathing phases were selected manually. However, in some test sequences, the target motion could not be completely tracked resulting in temporally higher TEs. This could be due to long-term motion affecting the target appearance, as the target reference patches were taken

from ultrasound images that were about 10 min apart from the test sequences. This indicates that the use of multiple reference patches may not be sufficient for target tracking when there is a large time lag between planning and tracking. In the clinical workflow of radiation therapy such a time lag can occur between the planning and the actual treatment phase. In addition, it is even still unclear how many reference patches would be required to ensure robust target tracking. A better solution would be to make the tracking algorithm itself invariant to this motion, so that only one reference patch is needed. One approach doing this was applied in section 6.2.4 with the DCAE-based target tracking approach. Here, the target reference was taken from the first ultrasound image of the test sequence according to an expert label. This eliminates the time lag between planning and tracking as no planning is required. However, this procedure requires a physician to mark the target to track in the beginning of the treatment session which could be impractical for the clinical workflow. To eliminate this need, the process could be automated. A method for finding the optimal target to track and to define the reference patch could be developed. A simple approach was proposed by Lin et al. [205], who performed different preprocessing steps based on multiple 2D ultrasound images to identify regions with the lowest motion magnitude. Another approach could be using a feature detection algorithm, such as FAST-3D, to identify robust landmarks near the target that can be detected in multiple ultrasound images.

In addition, in the DCAE-based approach deformable convolutional layers were added to the autoencoder to compensate for deformable target motion and make the autoencoder invariant to them. The results show an improvement in TE when adding these layers to the autoencoder architecture indicating this is a promising approach. This finding was confirmed by Liu et al. [189] who also used deformable convolutional layers in their neural network for target tracking. Even though they only performed target tracking in 2D+t ultrasound, they measured a low TE of 0.69 ± 0.67 mm, on average, which is the lowest TE reported on the CLUST website[2] for 2D+t ultrasound tracking. In this work, 2D deformable convolutional layers were used for processing 3D ultrasound patches that has been rearranged to 2.5D patches. Although promising tracking results were observed, a future study should investigate whether the use of 3D deformable convolution can further improve the tracking accuracy. This would spare the slicing procedure, but could potentially increase the computational cost of autoencoder prediction, since more parameters are required in the autoencoder due to the higher data dimensionality.

Another approach to improve the ability to discriminate dissimilar ultrasound patches from similar patches in the representation space could be contrastive learning. Contrastive learning is based on the idea of moving similar samples closer together and dissimilar samples further apart in the representation space [206]. This can even be done in a self-supervised way by using data augmentation in order to eliminate the need of labeled data for training [207]. The deformable ultrasound augmentation technique developed in this work (see section 5.2) could be used for this to generate realistic similar ultrasound patches whereas translation could be used for generating dissimilar patches. This approach could be used, e.g., for fine-tuning the encoder after training an autoencoder in an unsupervised way [208]. In future studies, it

[2]Results on CLUST data are summarized in: `https://clust.ethz.ch/results.html`

should be investigated whether such a contrastive learning approach can improve the performance of a tracking algorithm executed in representation space.

Ultrasound Imaging

Besides geometrical transformations, ultrasound images can be affected by other effects as well, e.g., caused by the imaging procedure itself. Depending on the ultrasound transducer characteristics and the settings of the ultrasound machine, the ultrasound images can highly differ, e.g., in contrast, brightness, depth and size [209], [210]. For the purpose of tracking, patches that differ in such aspects are considered similar as the image appearance but not the target itself has changed. These changes were not considered in the experiments in this work as the main objective was to analyze the target motion independently from ultrasound machine settings. In addition, the 3D ultrasound sequences in the data set used in this work were acquired with similar settings so that this type of change did not affect the data set. In the clinic, physicians adjust the settings of the ultrasound machine individually for each examination, which can result in different looking ultrasound images. In fact, this process is highly user-dependent and requires a high level of skill. In future work, the influence of this type of ultrasound image change to the ultrasound tracking performance should be investigated to be able to make the tracking algorithm more robust. For instance, when using an image feature-based tracking algorithm that requires setting hyperparameters that depend on image gray scale values, knowing the relationship between ultrasound machine settings and the resulting ultrasound image appearance would be required to automate finding the optimal set of hyperparameters. This knowledge could overcome the need of hyperparameter optimization like performed in the image feature-based approach in section 6.2.1. In learning-based approaches such as using networks like a DLN or an autoencoder, these changes could be overcome, e.g., by simulating them using data augmentation. However, it can be questioned whether this type of changes even needs to be considered for an autonomous ultrasound guided robotic system. For this purpose, the settings of the ultrasound machine could be set to a standard setting that are optimized for the objective and kept fixed. Consequently, the neural network used only needs to be trained based on data collected with these standard settings.

8 CONCLUSION

Robotic ultrasound guidance systems have the potential to ensure and improve the quality of treatment outcomes. For the radiation therapy domain, robotic ultrasound guidance systems are under research as they can remove the need for using harmful ionizing imaging and implanted fiducial markers to track the target [4], [9], [73]. Using 3D ultrasound imaging, the target can be directly visualized and tracked in real-time during treatment. In order to contribute to the development of such a robotic ultrasound system, an approach for performing robust and real-time target tracking in 4D ultrasound was investigated in this work. The goal was to develop representation learning-based target tracking algorithms, as there is a lack of feature-based tracking methods for 4D ultrasound in literature. For this purpose, four research questions have been investigated and answered in this work:

Research Question 1
How much does a target in the liver move and how precisely can it be tracked in 3D ultrasound sequences?

could be answered with a target motion range of 11.56 ± 5.86 mm and an intra-observer variability of 1.59 ± 1.24 mm, on average. The intra-observer variability was determined based on the labels set by the two expert observers with the lowest intra-observer variability in the labeling study. These results show that motion management in radiation therapy of liver targets is required to ensure high quality treatment as large motion can occur. Furthermore, the results provide a lower bound for evaluating tracking accuracy using the labeled data set.

Research Question 2
How can meaningful and unique features be extracted from 3D ultrasound images?

has been answered by proposing two different approaches. First, an image keypoint detector, FAST-3D, and two binary feature descriptors, BRIEF-3D and BRISK-3D, were developed that provide the ability to locate and describe local features in 3D ultrasound images in a simple and fast way. Second, convolutional autoencoders are proposed that are able to map a 3D ultrasound patch into a latent representation vector that contains information sufficient for patch reconstruction and comparison.

Research Question 3
Is it possible to perform realistic 3D ultrasound image augmentation using unsupervised deep representation learning and is this augmentation beneficial for a target tracking application?

could be answered affirmatively. Both, a generation and a transformation-based augmentation method based on representation learning were proposed. It was shown that synthetic ultrasound images can be created that are similar to a real one by sampling from a representation space created across different anatomical domains. Furthermore, a deformable autoencoder was proposed that is able to augment 3D ultrasound patches in a realistic deformable way, learned from real breathing-induced motion patterns. This approach has been shown to improve the accuracy of a neural network to localize a target in 3D ultrasound images.

Research Question 4
Can real-time target tracking in 4D ultrasound be performed in representation space and can it outperform image space-based tracking algorithms?

could also be answered positively. Two approaches for performing tracking in representation space were proposed and evaluated in comparison to basic image space-based tracking approaches. Image feature detection and description methods were used to perform fast target tracking in 4D ultrasound. Using autoencoders and the greedy tracking algorithm proposed in this work, it was possible to run faster than any state-of-the-art 4D ultrasound tracking algorithm with an average frame rate of 250 Hz and achieve a comparable TE of 1.58 ± 0.87 mm. Hence, it was shown that representation space-based target tracking is promising for the purpose of robotic 4D ultrasound therapy guidance. In addition, the online applicability of the tracking algorithm was shown in a first proof-of-concept experiment.
Considering the findings observed in this work, a general procedure for performing target tracking in 4D ultrasound is suggested in Figure 8.1. The procedure is presented in a loop starting with ultrasound image acquisition denoting the imaging process on the ultrasound machine. Subsequently, target localization is split into two consecutive processes performing coarse and global target localization followed by precise and local target localization. As shown in this work, a global tracking approach such as the proposed image feature-based method is able to compensate for large target displacements or partially disappearing targets caused, e.g., by shadow artifacts. To reduce the required computation time, this could be done on subsampled ultrasound images resulting in coarse target localization. For precise and local target localization, the greedy tracking algorithm and the DCAE-based approach proposed in this work could be used as they showed promising results in accuracy and runtime. After target localization, the target motion is determined and the motion information can be used, e.g., to adjust the ultrasound robot position or the treatment procedure to ensure optimal treatment outcome. To prevent subsequent ultrasound images from being severely affected by ultrasound image artifacts, such as shadow artifacts, e.g., caused by ribs, a transducer alignment approach should be included. Based on the ultrasound image and other measurements, such as

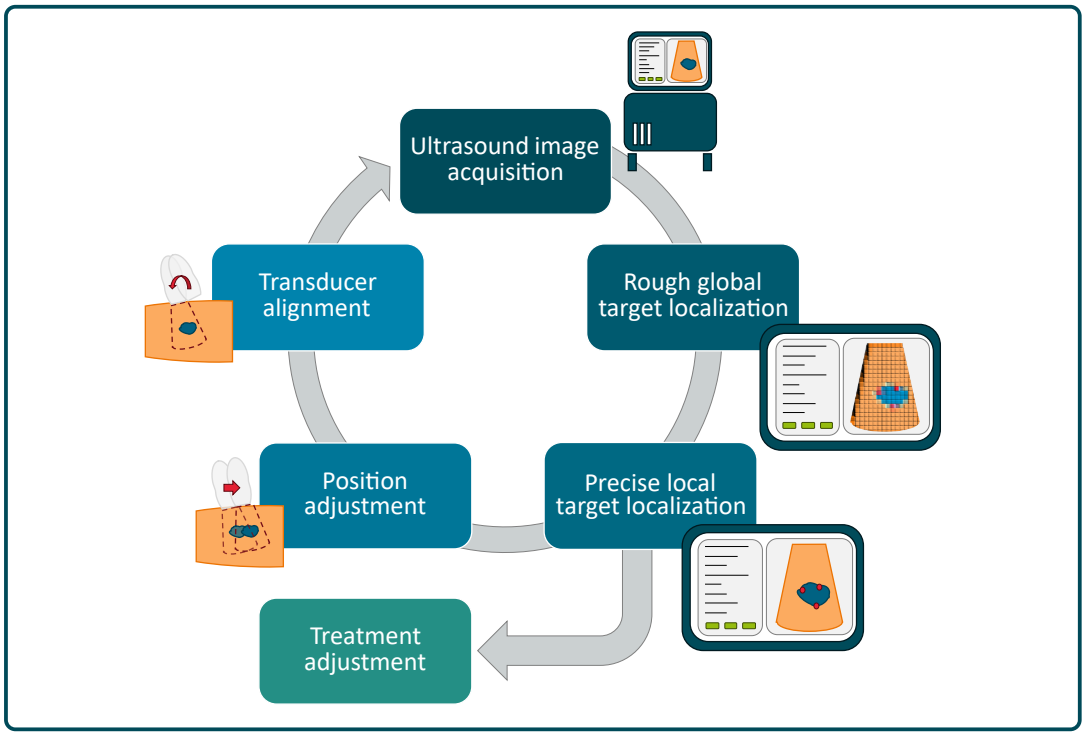

Figure 8.1: The procedure suggested for performing target tracking in 4D ultrasound based on the findings in this work. After acquiring a 3D ultrasound image, a coarse and global target localization is performed to compensate for large target displacements, followed by a precise and local target localization to pinpoint the target position. The target motion is determined and can be used to reposition the ultrasound robot and for treatment adjustment. Finally, the ultrasound transducer is properly aligned on the patient to ensure high quality ultrasound images and to avoid ultrasound image artifacts.

the pressure force of the ultrasound robot, the ultrasound transducer can be aligned to avoid artifacts. Therefor, different approaches are under research based on deep learning [67], [211].

This work contributes to the research on real-time and robust 4D ultrasound target tracking methods required to integrate robot-guided ultrasound imaging into therapy guidance applications. Although it was found that there are still some challenges that need to be addressed in future studies to develop a highly reliable robotic ultrasound system, deep representation learning has proven to be a promising approach for this purpose, as it provides the ability to perform real-time tracking in a robust way without the need of labels for training.

BIBLIOGRAPHY

[1] J. A. Antolak and I. I. Rosen, "Planning target volumes for radiotherapy: how much margin is needed?", *International Journal of Radiation Oncology Biology Physics*, vol. 44, no. 5, pp. 1165–1170, 1999. DOI: 10.1016/S0360-3016(99)00117-0.

[2] J. Bertholet, A. Knopf, B. Eiben, J. McClelland, A. Grimwood, E. Harris, M. Menten, P. Poulsen, D. T. Nguyen, P. Keall, and U. Oelfke, "Real-time intrafraction motion monitoring in external beam radiotherapy", *Physics in Medicine and Biology*, vol. 64, no. 15, 15TR01, 2019. DOI: 10.1088/1361-6560/ab2ba8.

[3] M. Guckenberger, A. Richter, J. Boda-Heggemeann, and F. Lohr, "Motion Compensation in Radiotherapy", *Critical Reviews in Biomedical Engineering*, vol. 40, no. 3, pp. 187–197, 2012. DOI: 10.1615/CritRevBiomedEng.v40.i3.30.

[4] C. Western, D. Hristov, and J. Schlosser, "Ultrasound Imaging in Radiation Therapy: From Interfractional to Intrafractional Guidance", *Cureus*, vol. 7, no. 6, e280, 2015. DOI: 10.7759/cureus.280.

[5] R. D. Foster, T. D. Solberg, H. S. Li, A. Kerkhoff, C. A. Enke, T. R. Willoughby, and P. A. Kupelian, "Comparison of transabdominal ultrasound and electromagnetic transponders for prostate localization", *Journal of Applied Clinical Medical Physics*, vol. 11, no. 1, pp. 57–67, 2010. DOI: 10.1120/jacmp.v11i1.2924.

[6] D. P. Gierga, J. Brewer, G. C. Sharp, M. Betke, C. G. Willett, and G. T. Chen, "The correlation between internal and external markers for abdominal tumors: Implications for respiratory gating", *International Journal of Radiation Oncology Biology Physics*, vol. 61, no. 5, pp. 1551–1558, 2005. DOI: 10.1016/j.ijrobp.2004.12.013.

[7] A. S. Beddar, K. Kainz, T. M. Briere, Y. Tsunashima, T. Pan, K. Prado, R. Mohan, M. Gillin, and S. Krishnan, "Correlation between internal fiducial tumor motion and external marker motion for liver tumors imaged with 4D-CT", *International Journal of Radiation Oncology Biology Physics*, vol. 67, no. 2, pp. 630–638, 2007. DOI: 10.1016/j.ijrobp.2006.10.007.

[8] S. Gerlach, I. Kuhlemann, P. Jauer, R. Bruder, F. Ernst, C. Fürweger, and A. Schlaefer, "Robotic ultrasound-guided SBRT of the prostate: feasibility with respect to plan quality", *International Journal of Computer Assisted Radiology and Surgery*, vol. 12, no. 1, pp. 149–159, 2017. DOI: 10.1007/s11548-016-1455-7.

[9] J. Schlosser, R. H. Gong, R. Bruder, A. Schweikard, S. Jang, J. Henrie, A. Kamaya, A. Koong, D. T. Chang, and D. Hristov, "Robotic intrafractional US guidance for liver SABR: System design, beam avoidance, and clinical imaging", *Medical Physics*, vol. 43, no. 11, pp. 5951–5963, 2016. DOI: 10.1118/1.4964454.

[10] A. Grimwood, H. A. McNair, T. P. O'Shea, S. Gilroy, K. Thomas, J. C. Bamber, A. C. Tree, and E. J. Harris, "In Vivo Validation of Elekta's Clarity Autoscan for Ultrasound-based Intrafraction Motion Estimation of the Prostate During Radiation Therapy", *International Journal of Radiation Oncology Biology Physics*, vol. 102, no. 4, pp. 912–921, 2018. DOI: 10.1016/j.ijrobp.2018.04.008.

[11] A. Grimwood, H. Rivaz, H. Zhou, H. A. McNair, K. Jakubowski, J. C. Bamber, A. C. Tree, and E. J. Harris, "Improving 3D ultrasound prostate localisation in radiotherapy through increased automation of interfraction matching", *Radiotherapy and Oncology*, vol. 149, pp. 134–141, 2020. DOI: 10.1016/j.radonc.2020.04.044.

[12] Q. Huang and Z. Zeng, "A Review on Real-Time 3D Ultrasound Imaging Technology Qinghua", *BioMed Research International*, vol. 2017, no. 1, 2017. DOI: 10.1155/2017/6027029.

[13] V. De Luca, G. Székely, and C. Tanner, "Estimation of Large-Scale Organ Motion in B-Mode Ultrasound Image Sequences: A Survey", *Ultrasound in Medicine and Biology*, vol. 41, no. 12, pp. 3044–3062, 2015. DOI: 10.1016/j.ultrasmedbio.2015.07.022.

[14] V. De Luca, J. Banerjee, A. Hallack, S. Kondo, M. Makhinya, D. Nouri, L. Royer, A. Cifor, G. Dardenne, O. Goksel, M. J. Gooding, C. Klink, A. Krupa, A. Le Bras, M. Marchal, A. Moelker, W. J. Niessen, B. W. Papiez, A. Rothberg, J. Schnabel, T. van Walsum, E. Harris, M. A. Lediju Bell, and C. Tanner, "Evaluation of 2D and 3D ultrasound tracking algorithms and impact on ultrasound-guided liver radiotherapy margins", *Medical Physics*, vol. 45, no. 11, pp. 4986–5003, 2018. DOI: 10.1002/mp.13152.

[15] Y. Huang, J. He, X. Wu, X. Zhao, and J. Wu, "Tracking 3D ultrasound anatomical landmarks via three orthogonal plane-based scale discriminative correlation filter network", *Medical Physics*, vol. 48, no. 5, pp. 2127–2135, 2021. DOI: 10.1002/mp.14798.

[16] J. He, C. Shen, Y. Chen, Y. Huang, and J. Wu, "FPSN-FNCC: an accurate and fast motion tracking algorithm in 3D ultrasound for image-guided interventions", *Physics in Medicine and Biology*, vol. 66, no. 15, p. 155 012, 2021. DOI: 10.1088/1361-6560/abffef.

[17] J. Banerjee, C. Klink, E. D. Peters, W. J. Niessen, A. Moelker, and T. van Walsum, "Fast and robust 3D ultrasound registration – Block and game theoretic matching", *Medical Image Analysis*, vol. 20, no. 1, pp. 173–183, 2015. DOI: 10.1016/j.media.2014.11.004.

[18] L. Royer, A. Krupa, G. Dardenne, A. Le Bras, E. Marchand, and M. Marchal, "Real-time target tracking of soft tissues in 3D ultrasound images based on robust visual information and mechanical simulation", *Medical Image Analysis*, vol. 35, pp. 582–598, 2017. DOI: 10.1016/j.media.2016.09.004.

[19] P. J. Keall, G. S. Mageras, J. M. Balter, R. S. Emery, K. M. Forster, S. B. Jiang, J. M. Kapatoes, D. a. Low, M. J. Murphy, B. R. Murray, C. R. Ramsey, M. B. Van Herk, S. S. Vedam, J. W. Wong, and E. Yorke, *The Management of Respiratory Motion in Radiation Oncology*. American Association of Physicists in Medicine, 2006, ISBN: 978-1-888340-61-7.

[20] X. Sun, Z. Dai, M. Xu, X. Guo, H. Su, and Y. Li, "Quantifying 6D tumor motion and calculating PTV margins during liver stereotactic radiotherapy with fiducial tracking", *Frontiers in Oncology*, vol. 12, 2022. DOI: 10.3389/fonc.2022.1021119.

[21] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013. DOI: 10.1109/TPAMI.2013.50.

[22] M. Greenacre, P. J. F. Groenen, T. Hastie, A. I. D'Enza, A. Markos, and E. Tuzhilina, "Principal component analysis", *Nature Reviews Methods Primers*, vol. 2, no. 1, p. 100, 2022. DOI: 10.1038/s43586-022-00184-w.

[23] M. Puttagunta and S. Ravi, "Medical image analysis based on deep learning approach", *Multimedia Tools and Applications*, vol. 80, no. 16, pp. 24 365–24 398, 2021. DOI: 10.1007/s11042-021-10707-4.

[24] A. I. Awad and M. Hassaballah, Eds., *Image Feature Detectors and Descriptors* (Studies in Computational Intelligence). Cham: Springer International Publishing, 2016, vol. 630, ISBN: 978-3-319-28852-9. DOI: 10.1007/978-3-319-28854-3.

[25] L. Ladha and T. Deepa, "Feature Selection Methods and Algorithms", *International Journal on Computer Science and Engineering*, vol. 3, no. 5, pp. 1787–1797, 2011.

[26] L. Han, Y. Tian, and Q. Qi, "Research on edge detection algorithm based on improved sobel operator", *MATEC Web of Conferences*, vol. 309, p. 03 031, 2020. DOI: 10.1051/matecconf/202030903031.

[27] C. G. Harris and M. J. Stephens, "A Combined Corner and Edge Detector", in *Alvey Vision Conference*, 1988. [Online]. Available: https://api.semanticscholar.org/CorpusID:1694378.

[28] J. Sánchez, N. Monzón, and A. Salgado, "An Analysis and Implementation of the Harris Corner Detector", *Image Processing On Line*, vol. 8, pp. 305–328, 2018. DOI: 10.5201/ipol.2018.229.

[29] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004. DOI: 10.1023/B:VISI.0000029664.99615.94.

[30] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)", vol. 110, no. 3, pp. 346–359, 2008. DOI: 10.1016/j.cviu.2007.09.014.

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, T. Dietterich, Ed. MIT Press, 2016, http://www.deeplearningbook.org, ISBN: 978-0-262-03561-3.

[32] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means Algorithm: A Comprehensive Survey and Performance Evaluation", *Electronics*, vol. 9, no. 8, p. 1295, 2020. DOI: 10.3390/electronics9081295.

[33] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions", *Journal of Big Data*, vol. 8, no. 1, p. 53, 2021. DOI: 10.1186/s40537-021-00444-8.

[34] M. Chen, X. Shi, Y. Zhang, D. Wu, and M. Guizani, "Deep Feature Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network", *IEEE Transactions on Big Data*, vol. 7, no. 4, pp. 750–758, 2021. DOI: 10.1109/TBDATA.2017.2717439.

[35] S. Ruder, "An overview of gradient descent optimization algorithms", 2016. arXiv: 1609.04747.

[36] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2014. arXiv: 1412.6980.

[37] R. Moradi, R. Berangi, and B. Minaei, "A survey of regularization strategies for deep models", *Artificial Intelligence Review*, vol. 53, no. 6, pp. 3947–3986, 2020. DOI: 10.1007/s10462-019-09784-7.

[38] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", 2012. arXiv: 1207.0580.

[39] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", 2015. arXiv: 1502.03167.

[40] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders", in *Machine Learning for Data Science Handbook*, L. Rokach, O. Maimon, and E. Shmueli, Eds. Springer International Publishing, 2023, pp. 353–374. DOI: 10.1007/978-3-031-24628-9_16.

[41] J. Ehrhardt and M. Wilms, "Autoencoders and variational autoencoders in medical image analysis", in *Biomedical Image Synthesis and Simulation*, 2022, pp. 129–162. DOI: 10.1016/B978-0-12-824349-7.00015-3.

[42] B. Li, K. Xu, D. Feng, H. Mi, H. Wang, and J. Zhu, "Denoising Convolutional Autoencoder Based B-mode Ultrasound Tongue Image Feature Extraction", in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7130–7134. DOI: 10.1109/ICASSP.2019.8682806.

[43] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes", *CoRR*, 2013. arXiv: 1312.6114.

[44] A. T. Cemgil, S. Ghaisas, K. Dvijotham, S. Gowal, and P. Kohli, "Autoencoding Variational Autoencoder", *Advances in Neural Information Processing Systems*, 2020. arXiv: 2012.03715.

[45] C. Doersch, "Tutorial on Variational Autoencoders", 2016. arXiv: 1606.05908.

[46] S. Kolouri, P. E. Pope, C. E. Martin, and G. K. Rohde, "Sliced-Wasserstein Autoencoder: An Embarrassingly Simple Generative Model", *arXiv*, 2018. [Online]. Available: http://arxiv.org/abs/1804.01947.

[47] J. Wu, Z. Huang, D. Acharya, W. Li, J. Thoma, D. P. Paudel, and L. Van Gool, "Sliced Wasserstein Generative Models", in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3708–3717. DOI: 10.1109/CVPR.2019.00383.

[48] S. Kolouri, S. R. Park, M. Thorpe, D. Slepcev, and G. K. Rohde, "Optimal Mass Transport: Signal processing and machine-learning applications", *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 43–59, 2017. DOI: 10.1109/MSP.2017.2695801.

[49] I. Deshpande, Z. Zhang, and A. Schwing, "Generative Modeling Using the Sliced Wasserstein Distance", in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3483–3491. DOI: 10.1109/CVPR.2018.00367.

[50] H. Salman, P. Singhal, T. Shankar, P. Yin, A. Salman, W. Paivine, G. Sartoretti, M. Travers, and H. Choset, "Learning to Sequence Robot Behaviors for Visual Navigation", 2018. arXiv: 1803.01446.

[51] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[52] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[53] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to Forget: Continual Prediction with LSTM", *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000. DOI: 10.1162/089976600300015015.

[54] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model", *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, 2020. DOI: 10.1007/s10462-020-09838-1.

[55] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting", *Advances in Neural Information Processing Systems*, vol. 2015-Janua, pp. 802–810, 2015. arXiv: 1506.04214.

[56] E. M. Boctor, M. A. Choti, E. C. Burdette, and R. J. W. III, "Three-dimensional ultrasound-guided robotic needle placement: an experimental evaluation", *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 4, no. 2, pp. 180–191, 2008. DOI: 10.1002/rcs.184.

[57] J. Esteban, W. Simson, S. Requena Witzig, A. Rienmüller, S. Virga, B. Frisch, O. Zettinig, D. Sakara, Y. M. Ryang, N. Navab, and C. Hennersperger, "Robotic ultrasound-guided facet joint insertion", *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 6, pp. 895–904, 2018. DOI: 10.1007/s11548-018-1759-x.

[58] S. Chen, F. Wang, Y. Lin, Q. Shi, and Y. Wang, "Ultrasound-guided needle insertion robotic system for percutaneous puncture", *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, no. 3, pp. 475–484, 2021. DOI: 10.1007/s11548-020-02300-1.

[59] M. Giuliani, D. Szcześniak-Stańczyk, N. Mirnig, G. Stollnberger, M. Szyszko, B. Stańczyk, and M. Tscheligi, "User-centred design and evaluation of a tele-operated echocardiography robot", *Health and Technology*, vol. 10, no. 3, pp. 649–665, 2020. DOI: 10.1007/s12553-019-00399-0.

[60] F. Langsch, S. Virga, J. Esteban, R. Gobl, and N. Navab, "Robotic Ultrasound for Catheter Navigation in Endovascular Procedures", in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 5404–5410. DOI: 10.1109/IROS40897.2019.8967652.

[61] S. J. Adams, B. E. Burbridge, A. Badea, N. Kanigan, L. Bustamante, P. Babyn, and I. Mendez, "A Crossover Comparison of Standard and Telerobotic Approaches to Prenatal Sonography", *Journal of Ultrasound in Medicine*, vol. 37, no. 11, pp. 2603–2612, 2018. DOI: 10.1002/jum.14619.

[62] O. Zettinig, B. Fuerst, R. Kojcev, M. Esposito, M. Salehi, W. Wein, J. Rackerseder, E. Sinibaldi, B. Frisch, and N. Navab, "Toward real-time 3D ultrasound registration-based visual servoing for interventional navigation", in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 945–950. DOI: 10.1109/ICRA.2016.7487226.

[63] Z. Jiang, S. E. Salcudean, and N. Navab, "Robotic Ultrasound Imaging: State-of-the-Art and Future Perspectives", *Medical Image Analysis*, vol. 89, p. 102 878, 2023. DOI: 10.1016/j.media.2023.102878.

[64] F. von Haxthausen, S. Böttger, D. Wulff, J. Hagenah, V. García-Vázquez, and S. Ipsen, "Medical Robotics for Ultrasound Imaging: Current Systems and Future Trends", *Current Robotics Reports*, vol. 2, no. 1, pp. 55–71, 2021. DOI: 10.1007/s43154-020-00037-y.

[65] J. M. Beer, A. D. Fisk, and W. A. Rogers, "Toward a Framework for Levels of Robot Autonomy in Human-Robot Interaction", *Journal of Human-Robot Interaction*, vol. 3, no. 2, p. 74, 2014. DOI: 10.5898/JHRI.3.2.Beer.

[66] S. Ipsen, D. Wulff, I. Kuhlemann, A. Schweikard, and F. Ernst, "Towards automated ultrasound imaging—robotic image acquisition in liver and prostate for long-term motion monitoring", *Physics in Medicine and Biology*, vol. 66, p. 094 002, 2021. DOI: 10.1088/1361-6560/abf277.

[67] J. Osburg, D. Wulff, and F. Ernst, "Generalized Automatic Probe Alignment based on 3D Ultrasound", *Current Directions in Biomedical Engineering*, vol. 8, no. 1, pp. 58–61, 2021. DOI: 10.1515/cdbme-2022-0015.

[68] K. Li, Y. Xu, and M. Q.-H. Meng, "An Overview of Systems and Techniques for Autonomous Robotic Ultrasound Acquisitions", *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 2, pp. 510–524, 2021. DOI: 10.1109/TMRB.2021.3072190.

[69] R. Elek, T. D. Nagy, D. Á. Nagy, B. Takács, P. Galambos, I. Rudas, and T. Haidegger, "Robotic platforms for ultrasound diagnostics and treatment", in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017, pp. 1752–1757. DOI: 10.1109/SMC.2017.8122869.

[70] H. T. Sen, M. A. L. Bell, Y. Zhang, K. Ding, E. Boctor, J. Wong, I. Iordachita, and P. Kazanzides, "System Integration and In Vivo Testing of a Robot for Ultrasound Guidance and Monitoring During Radiotherapy", *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 7, pp. 1608–1618, 2017. DOI: 10.1109/TBME.2016.2612229.

[71] P. Chatelain, A. Krupa, and M. Marchal, "Real-time needle detection and tracking using a visually servoed 3D ultrasound probe", in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 1676–1681. DOI: 10.1109/ICRA.2013.6630795.

[72] C. Hennersperger, B. Fuerst, S. Virga, O. Zettinig, B. Frisch, T. Neff, and N. Navab, "Towards MRI-Based Autonomous Robotic US Acquisitions: A First Feasibility Study", *IEEE Transactions on Medical Imaging*, vol. 36, no. 2, pp. 538–548, 2017. DOI: 10.1109/TMI.2016.2620723.

[73] P. K. Seitz, B. Baumann, W. Johnen, C. Lissek, J. Seidel, and R. Bendl, "Development of a robot-assisted ultrasound-guided radiation therapy (USgRT)", *International Journal of Computer Assisted Radiology and Surgery*, vol. 15, pp. 491–501, 2020. DOI: 10.1007/s11548-019-02104-y.

[74] J. Sarrazin, E. Promayon, M. Baumann, and J. Troccaz, "Hand-eye calibration of a robot - UltraSound probe system without any 3D localizers", in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 21–24. DOI: 10.1109/EMBC.2015.7318251.

[75] I. Kuhlemann, P. Jauer, A. Schweikard, and F. Ernst, "Patient localization for robotized ultrasound-guided radiation therapy", *Imaging and Computer Assistance in Radiation Therapy, ICART 2015, 18th International Conference on Medical Image Computing and Computer-Assisted Intervention - MICCAI'15*, pp. 105–112, 2015.

[76] S. Ipsen, R. Bruder, I. Kuhlemann, P. Jauer, L. Motisi, F. Cremers, F. Ernst, and A. Schweikard, "A visual probe positioning tool for 4D ultrasound-guided radiotherapy", in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 883–886. DOI: 10.1109/EMBC.2018.8512390.

[77] M. Schlüter, C. Fürweger, and A. Schlaefer, "Optimizing robot motion for robotic ultrasound-guided radiation therapy", *Physics in Medicine and Biology*, vol. 64, no. 19, p. 195 012, 2019. DOI: 10.1088/1361-6560/ab3bfb.

[78] V. Chan and A. Perlas, "Basics of Ultrasound Imaging", in *Atlas of Ultrasound-Guided Procedures in Interventional Pain Management*, S. N. Narouze, Ed. New York, NY: Springer New York, 2011, pp. 13–19, ISBN: 978-1-4419-1681-5. DOI: 10.1007/978-1-4419-1681-5_2.

[79] J. A. Jensen, "Medical ultrasound imaging", *Progress in Biophysics and Molecular Biology*, vol. 93, pp. 153–165, 2007. DOI: 10.1016/j.pbiomolbio.2006.07.025.

[80] S. Ipsen, R. Bruder, E. S. Worm, R. Hansen, P. R. Poulsen, M. Høyer, and A. Schweikard, "Simultaneous acquisition of 4D ultrasound and wireless electromagnetic tracking for in-vivo accuracy validation", *Current Directions in Biomedical Engineering*, vol. 3, no. 2, pp. 75–78, 2017. DOI: doi:10.1515/cdbme-2017-0016.

[81] A. Fenster, D. B. Downey, and H. N. Cardinal, "Three-dimensional ultrasound imaging", *Physics in Medicine and Biology*, vol. 46, no. 5, R67–R99, 2001. DOI: 10.1088/0031-9155/46/5/201.

[82] A. Fenster, J. Bax, H. Neshat, D. Cool, N. Kakani, and C. Romagnoli, "3D ultrasound imaging in image-guided intervention", in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2014, pp. 6151–6154. DOI: 10.1109/EMBC.2014.6945033.

[83] M. A. Lediju Bell, H. T. Sen, I. I. Iordachita, P. Kazanzides, and J. Wong, "In vivo reproducibility of robotic probe placement for a novel ultrasound-guided radiation therapy system", *Journal of Medical Imaging*, vol. 1, no. 2, p. 025 001, 2014. DOI: 10.1117/1.JMI.1.2.025001.

[84] A. Gilbert, M. Marciniak, C. Rodero, P. Lamata, E. Samset, and K. Mcleod, "Generating Synthetic Labeled Data From Existing Anatomical Models: An Example With Echocardiography Segmentation", *IEEE Transactions on Medical Imaging*, vol. 40, no. 10, pp. 2783–2794, 2021. DOI: 10.1109/TMI.2021.3051806.

[85] W. Al-Dhabyani, M. Gomaa, H. Khaled, and A. Fahmy, "Dataset of breast ultrasound images", *Data in Brief*, vol. 28, p. 104 863, 2020. DOI: 10.1016/j.dib.2019.104863.

[86] S. Tong, H. Cardinal, R. F. McLoughlin, D. B. Downey, and A. Fenster, "Intra- and inter-observer variability and reliability of prostate volume measurement via two-dimensional and three-dimensional ultrasound imaging", *Ultrasound in Medicine and Biology*, vol. 24, no. 5, pp. 673–681, 1998. DOI: 10.1016/S0301-5629(98)00039-8.

[87] I. Sarris, C. Ioannou, P. Chamberlain, E. Ohuma, F. Roseman, L. Hoch, D. G. Altman, and A. T. Papageorghiou, "Intra- and interobserver variability in fetal ultrasound measurements", *Ultrasound in Obstetrics and Gynecology*, vol. 39, no. 3, pp. 266–273, 2012. DOI: 10.1002/uog.10082.

[88] D. Blanca, E. C. Schwarz, T. J. Olgers, E. ter Avest, N. Azizi, H. R. Bouma, and J. C. Ter Maaten, "Intra-and inter-observer variability of point of care ultrasound measurements to evaluate hemodynamic parameters in healthy volunteers", *The Ultrasound Journal*, vol. 15, no. 1, p. 22, 2023. DOI: 10.1186/s13089-023-00322-9.

[89] Z. B. Popović and J. D. Thomas, "Assessing observer variability: a user's guide", *Cardiovascular Diagnosis and Therapy*, vol. 7, no. 3, pp. 317–324, 2017. DOI: 10.21037/cdt.2017.03.12.

[90] S. Vijayan, S. Klein, E. F. Hofstad, F. Lindseth, B. Ystgaard, and T. Langø, "Motion tracking in the liver: Validation of a method based on 4D ultrasound using a nonrigid registration technique", *Medical Physics*, vol. 41, p. 082 903, 2014. DOI: 10.1118/1.4890091.

[91] V. De Luca, T. Benz, S. Kondo, L. König, D. Lübke, S. Rothlübbers, O. Somphone, S. Allaire, M. A. Lediju Bell, D. Y. F. Chung, A. Cifor, C. Grozea, M. Günther, J. Jenne, T. Kipshagen, M. Kowarschik, N. Navab, J. Rühaak, J. Schwaab, and C. Tanner, "The 2014 liver ultrasound tracking benchmark", *Physics in Medicine and Biology*, vol. 60, no. 14, pp. 5571–5599, 2015. DOI: 10.1088/0031-9155/60/14/5571.

[92] Y.-L. Tsai, C.-J. Wu, S. Shaw, P.-C. Yu, H.-H. Nien, and L. T. Lui, "Quantitative analysis of respiration-induced motion of each liver segment with helical computed tomography and 4-dimensional computed tomography", *Radiation Oncology*, vol. 13, no. 1, p. 59, 2018. DOI: 10.1186/s13014-018-1007-0.

[93] T. Rohlfing, C. R. Maurer, W. G. O'Dell, and J. Zhong, "Modeling liver motion and deformation during the respiratory cycle using intensity-based nonrigid registration of gated MR images", *Medical Physics*, vol. 31, no. 3, pp. 427–432, 2004. DOI: 10.1118/1.1644513.

[94] P. H. Weiss, J. M. Baker, and E. J. Potchen, "Assessment of Hepatic Respiratory Excursion", *Journal of Nuclear Medicine*, vol. 13, no. 10, pp. 758–759, 1972.

[95] G. Harauz, M. J. Bronskill, and T. Ontario, "Comparison of the liver's respiratory motion in the supine and upright positions: concise communication", *Journal of Nuclear Medicine*, vol. 20, no. 7, pp. 733–735, 1979.

[96] I. Suramo, M. Päivänsalo, and V. Myllylä, "Cranio-Caudal Movements of the Liver, Pancreas and Kidneys in Respiration", *Acta Radiologica. Diagnosis*, vol. 25, no. 2, pp. 129–131, 1984. DOI: 10.1177/028418518402500208.

[97] S. C. Davies, A. L. Hill, R. B. Holmes, M. Halliwell, and P. C. Jackson, "Ultrasound quantitation of respiratory organ motion in the upper abdomen", *The British Journal of Radiology*, vol. 67, no. 803, pp. 1096–1102, 1994. DOI: 10.1259/0007-1285-67-803-1096.

[98] R. B. Case, D. J. Moseley, J. J. Sonke, C. L. Eccles, R. E. Dinniwell, J. Kim, A. Bezjak, M. Milosevic, K. K. Brock, and L. A. Dawson, "Interfraction and Intrafraction Changes in Amplitude of Breathing Motion in Stereotactic Liver Radiotherapy", *International Journal of Radiation Oncology Biology Physics*, vol. 77, no. 3, pp. 918–925, 2010. DOI: 10.1016/j.ijrobp.2009.09.008.

[99] J. C. Park, S. H. Park, J. H. Kim, S. M. Yoon, S. Y. Song, Z. Liu, B. Song, K. Kauweloa, M. J. Webster, A. Sandhu, L. K. Mell, S. B. Jiang, A. J. Mundt, and W. Y. Song, "Liver motion during cone beam computed tomography guided stereotactic body radiation therapy", *Medical Physics*, vol. 39, no. 10, pp. 6431–6442, 2012. DOI: 10.1118/1.4754658.

[100] E. S. Worm, M. Høyer, W. Fledelius, and P. R. Poulsen, "Three-dimensional, Time-Resolved, Intrafraction Motion Monitoring Throughout Stereotactic Liver Radiation Therapy on a Conventional Linear Accelerator", *International Journal of Radiation Oncology Biology Physics*, vol. 86, no. 1, pp. 190–197, 2013. DOI: 10.1016/j.ijrobp.2012.12.017.

[101] S. A. Jupitz, A. J. Shepard, P. M. Hill, and B. P. Bednarz, "Investigation of tumor and vessel motion correlation in the liver", *Journal of Applied Clinical Medical Physics*, vol. 21, no. 8, pp. 183–190, 2020. DOI: 10.1002/acm2.12943.

[102] S. Arslan, A. Yazıcı, A. Saçan, I. H. Toroslu, and E. Acar, "Comparison of feature-based and image registration-based retrieval of image data using multidimensional data access methods", *Data and Knowledge Engineering*, vol. 86, pp. 124–145, 2013. DOI: 10.1016/j.datak.2013.01.007.

[103] G. Haskins, U. Kruger, and P. Yan, "Deep learning in medical image registration: a survey", *Machine Vision and Applications*, vol. 31, no. 8, 2020. DOI: 10.1007/s00138-020-01060-x.

[104] H. R. Boveiri, R. Khayami, R. Javidan, and A. Mehdizadeh, "Medical image registration using deep neural networks: A comprehensive review", *Computers and Electrical Engineering*, vol. 87, p. 106 767, 2020. DOI: 10.1016/j.compeleceng.2020.106767.

[105] F. W. Kremkau and K. J. Taylor, "Artifacts in ultrasound imaging.", *Journal of Ultrasound in Medicine*, vol. 5, no. 4, pp. 227–237, 1986. DOI: 10.7863/jum.1986.5.4.227.

[106] M. M. Quien and M. Saric, "Ultrasound imaging artifacts: How to recognize them and how to avoid them", *Echocardiography*, vol. 35, no. 9, pp. 1388–1401, 2018. DOI: 10.1111/echo.14116.

[107] S.-y. Selmi, E. Promayon, and J. Troccaz, "Hybrid 2D–3D ultrasound registration for navigated prostate biopsy", *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 7, pp. 987–995, 2018. DOI: 10.1007/s11548-018-1736-4.

[108] S.-Y. Guan, T.-M. Wang, C. Meng, and J.-C. Wang, "A Review of Point Feature Based Medical Image Registration", *Chinese Journal of Mechanical Engineering*, vol. 31, no. 1, p. 76, 2018. DOI: 10.1186/s10033-018-0275-9.

[109] D. Wulff, I. Kuhlemann, F. Ernst, A. Schweikard, and S. Ipsen, "Robust motion tracking of deformable targets in the liver using binary feature libraries in 4D ultrasound", *Current Directions in Biomedical Engineering*, vol. 5, no. 1, pp. 601–604, 2019. DOI: 10.1515/cdbme-2019-0151.

[110] D. Wulff and F. Ernst, "Feature Description using Autoencoders for Fast 3D Ultrasound Tracking", *Current Directions in Biomedical Engineering*, vol. 10, no. 2, pp. 21–24, 2024. DOI: 10.1515/cdbme-2024-1057.

[111] D. Wulff, J. Hagenah, S. Ipsen, and F. Ernst, "Learning Local Feature Descriptions in 3D Ultrasound", *2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE)*, pp. 323–330, 2020. DOI: 10.1109/BIBE50027.2020.00059.

[112] D. Wulff, J. Hagenah, and F. Ernst, "Comparison of Representation Learning Techniques for Tracking in time resolved 3D Ultrasound", *International Conference on Medical Imaging with Deep Learning (MIDL)*, 2022. arXiv: 2201.03319.

[113] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF", in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[114] E. Karami, S. Prasad, and M. S. Shehata, "Image matching using sift, surf, brief and orb: Performance comparison for distorted images", 2017. arXiv: 1710.02726.

[115] O. Miksik and K. Mikolajczyk, "Evaluation of Local Detectors and Descriptors for Fast Feature Matching", *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 2681–2684, 2012.

[116] K. A. Acharya, R. Venkatesh Babu, and S. S. Vadhiyar, "A real-time implementation of SIFT using GPU", *Journal of Real-Time Image Processing*, vol. 14, no. 2, pp. 267–277, 2018. DOI: 10.1007/s11554-014-0446-6.

[117] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection", *European Conference on Computer Vision*, pp. 430–443, 2006. DOI: 10.1007/11744023_34.

[118] A. Patel, D. R. Kasat, S. Jain, and V. M. Thakare, "Performance Analysis of Various Feature Detector and Descriptor for Real-Time Video based Face Tracking", *International Journal of Computer Applications*, vol. 93, no. 1, pp. 37–41, 2014. DOI: 10.5120/16183-5415.

[119] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints", in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555. DOI: 10.1109/ICCV.2011.6126542.

[120] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features", in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4, vol. 6314, 2010, pp. 778–792. DOI: 10.1007/978-3-642-15561-1_56.

[121] Z. Tabatabaei, A. Colomer, K. Engan, J. Oliver, and V. Naranjo, "Residual block Convolutional Auto Encoder in Content-Based Medical Image Retrieval", in *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*, 2022. DOI: 10.1109/IVMSP54334.2022.9816325.

[122] F. Milković, B. Filipović, M. Subašić, T. Petković, S. Lončarić, and M. Budimir, "Ultrasound Anomaly Detection Based on Variational Autoencoders", in *2021 12th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2021, pp. 225–229. DOI: 10.1109/ISPA52656.2021.9552041.

[123] N. Shvetsova, B. Bakker, I. Fedulova, H. Schulz, and D. V. Dylov, "Anomaly Detection in Medical Imaging With Deep Perceptual Autoencoders", *IEEE Access*, vol. 9, pp. 118 571–118 583, 2021. DOI: 10.1109/ACCESS.2021.3107163.

[124] J. Jiao, R. Droste, L. Drukker, A. T. Papageorghiou, and J. A. Noble, "Self-Supervised Representation Learning for Ultrasound Video", in *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 2020, pp. 1847–1850. DOI: 10.1109/ISBI45749.2020.9098666.

[125] T. Moriya, H. R. Roth, S. Nakamura, H. Oda, K. Nagara, M. Oda, and K. Mori, "Unsupervised segmentation of 3D medical images based on clustering and deep representation learning", in *Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 10578, 2018, p. 1 057 820. DOI: 10.1117/12.2293414.

[126] G. van Tulder, *Elasticdeform: Elastic deformations for n-dimensional images*, version 0.4.9, 2021. DOI: 10.5281/zenodo.4569691.

[127] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis", *Communications in Statistics*, vol. 3, no. 1, pp. 1–27, 1974. DOI: 10.1080/03610927408827101.

[128] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650–1654, 2002. DOI: 10.1109/TPAMI.2002.1114856.

[129] F. Draghi, G. L. Rapaccini, C. Fachinetti, N. de Matthaeis, S. Battaglia, T. Abbattista, and P. Busilacchi, "Ultrasound examination of the liver: Normal vascular anatomy", *Journal of Ultrasound*, vol. 10, no. 1, pp. 5–11, 2007. DOI: 10.1016/j.jus.2007.02.002.

[130] A. Bilir, Z. Fazliogullari, M. Koplay, N. Unver-Dogan, and A. K. Karabulut, "Diameter Measurements and Variations of the Hepatic Arterial System in Multidetector Computed Tomography Images", *International Journal of Morphology*, vol. 39, no. 3, pp. 869–875, 2021. DOI: 10.4067/S0717-95022021000300869.

[131] N. Altman and M. Krzywinski, "The curse(s) of dimensionality", *Nature Methods*, vol. 15, no. 6, pp. 399–400, 2018. DOI: 10.1038/s41592-018-0019-x.

[132] L. Chen, "Curse of Dimensionality", in *Encyclopedia of Database Systems*, Boston, MA: Springer US, 2009, pp. 545–546. DOI: 10.1007/978-0-387-39940-9_133.

[133] D. Peng, Z. Gui, and H. Wu, "Interpreting the Curse of Dimensionality from Distance Concentration and Manifold Effect", 2023. arXiv: 2401.00422.

[134] D. Wulff, J. Hagenah, and F. Ernst, "Landmark tracking in 4D ultrasound using generalized representation learning", *International Journal of Computer Assisted Radiology and Surgery*, vol. 18, pp. 493–500, 2023. DOI: 10.1007/s11548-022-02768-z.

[135] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning", *Journal of Big Data*, vol. 6, no. 1, p. 60, 2019. DOI: 10.1186/s40537-019-0197-0.

[136] F. Garcea, A. Serra, F. Lamberti, and L. Morra, "Data augmentation for medical imaging: A systematic literature review", *Computers in Biology and Medicine*, vol. 152, p. 106 391, 2023. DOI: 10.1016/j.compbiomed.2022.106391.

[137] W. Al-Dhabyani, M. Gomaa, H. Khaled, and A. Fahmy, "Deep Learning Approaches for Data Augmentation and Classification of Breast Masses using Ultrasound Images", *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, pp. 618–627, 2019. DOI: 10.14569/IJACSA.2019.0100579.

[138] M. Martin, B. Sciolla, M. Sdika, P. Quetin, and P. Delachartre, "Segmentation of neonates cerebral ventricles with 2D CNN in 3D US data: suitable training-set size and data augmentation strategies", in *2019 IEEE International Ultrasonics Symposium (IUS)*, vol. 2019-Octob, 2019, pp. 2122–2125. DOI: 10.1109/ULTSYM.2019.8925799.

[139] P. Chlap, H. Min, N. Vandenberg, J. Dowling, L. Holloway, and A. Haworth, "A review of medical image data augmentation techniques for deep learning applications", *Journal of Medical Imaging and Radiation Oncology*, vol. 65, no. 5, pp. 545–563, 2021. DOI: 10.1111/1754-9485.13261.

[140] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "GAN-based synthetic medical image augmentation for increased CNN performance in liver lesion classification", *Neurocomputing*, vol. 321, pp. 321–331, 2018. DOI: 10.1016/j.neucom.2018.09.013.

[141] Z. Qin, Z. Liu, P. Zhu, and Y. Xue, "A GAN-based image synthesis method for skin lesion classification", *Computer Methods and Programs in Biomedicine*, vol. 195, p. 105 568, 2020. DOI: 10.1016/j.cmpb.2020.105568.

[142] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers, "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks", *Scientific Reports*, vol. 9, no. 1, p. 16 884, 2019. DOI: 10.1038/s41598-019-52737-x.

[143] M. Tirindelli, C. Eilers, W. Simson, M. Paschali, M. F. Azampour, and N. Navab, "Rethinking Ultrasound Augmentation: A Physics-Inspired Approach", in *Medical Image Computing and Computer Assisted Intervention - MICCAI 2021*, 2021, pp. 690–700. DOI: 10.1007/978-3-030-87237-3_66.

[144] T. L. A. van den Heuvel, D. de Bruijn, C. L. de Korte, and B. van Ginneken, "Automated measurement of fetal head circumference using 2D ultrasound images", *PLOS ONE*, vol. 13, no. 8, e0200412, 2018. DOI: 10.1371/journal.pone.0200412.

[145] Y. Chen, C. Zhang, L. Liu, C. Feng, C. Dong, Y. Luo, and X. Wan, "USCL: Pretraining Deep Ultrasound Image Diagnosis Model Through Video Contrastive Representation Learning", in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, 2021, pp. 627–637. DOI: 10.1007/978-3-030-87237-3_60.

[146] D. Wulff, M. Mehdi, F. Ernst, and J. Hagenah, "Cross Data Set Generalization of Ultrasound Image Augmentation using Representation Learning: A Case Study", *Current Directions in Biomedical Engineering*, vol. 7, no. 2, pp. 755–758, 2021. DOI: 10.1515/cdbme-2021-2193.

[147] D. Wulff, T. Dohnke, and F. Ernst, "Towards Realistic 3D Ultrasound Synthesis: Deformable Augmentation using Conditional Variational Autoencoders", *IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 821–826, 2023. DOI: 10.1109/CBMS58004.2023.00326.

[148] E. Goceri, "Medical image data augmentation: techniques, comparisons and interpretations", *Artificial Intelligence Review*, vol. 56, no. 11, pp. 12 561–12 605, 2023. DOI: 10.1007/s10462-023-10453-z.

[149] J. M. Wolterink, T. Leiner, and I. Isgum, "Blood Vessel Geometry Synthesis using Generative Adversarial Networks", *1st Conference on Medical Imaging with Deep Learning (MIDL)*, 2018. arXiv: 1804.04381.

[150] F. von Haxthausen, J. Hagenah, M. Kaschwich, M. Kleemann, V. García-Vázquez, and F. Ernst, "Robotized ultrasound imaging of the peripheral arteries – a phantom study", *Current Directions in Biomedical Engineering*, vol. 6, no. 1, pp. 1–4, 2020. DOI: 10.1515/cdbme-2020-0033.

[151] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks", pp. 1–9, 2014. arXiv: 1406.2661.

[152] J. Krebs, H. Delingette, B. Mailhe, N. Ayache, and T. Mansi, "Learning a Probabilistic Model for Diffeomorphic Registration", *IEEE Transactions on Medical Imaging*, vol. 38, no. 9, pp. 2165–2176, 2019. DOI: 10.1109/TMI.2019.2897112.

[153] Z. Yaniv, B. C. Lowekamp, H. J. Johnson, and R. Beare, "SimpleITK Image-Analysis Notebooks: a Collaborative Environment for Education and Reproducible Research", *Journal of Digital Imaging*, vol. 31, no. 3, pp. 290–303, 2018. DOI: 10.1007/s10278-017-0037-8.

[154] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "VoxelMorph: A Learning Framework for Deformable Medical Image Registration", *IEEE Transactions on Medical Imaging*, vol. 38, no. 8, pp. 1788–1800, 2019. DOI: 10.1109/TMI.2019.2897538.

[155] U. Semmelmann and G. Weingart, "The standard Laplace operator", *manuscripta mathematica*, vol. 158, pp. 273–293, 2019. DOI: 10.1007/s00229-018-1023-2.

[156] Banerjee, J., Klink, C., Vast, E., Niessen, W.J., Moelker, A. and van Walsum, T., "A combined tracking and registration approach for tracking anatomical landmarks in 4D ultrasound of the liver", *MICCAI workshop: Challenge on Liver Ultrasound Tracking*, pp. 43–50, 2015. [Online]. Available: https://clust.ethz.ch/opendownload/CLUST2015/banerjee_clust15.pdf.

[157] A. Hallack, B. W. Papie, A. Cifor, M. J. Gooding, and J. A. Schnabel, "Robust Liver Ultrasound Tracking using Dense Distinctive Image Features", 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:195817078.

[158] Y. Zheng, D. Liu, B. Georgescu, H. Nguyen, and D. Comaniciu, "3D Deep Learning for Efficient and Robust Landmark Detection in Volumetric Data", in *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015*, 9349, 2015, pp. 565–572. DOI: 10.1007/978-3-319-24553-9_69.

[159] C. Krause, D. Wulff, and F. Ernst, "Landmark Tracking in 4D Ultrasound using Localization Networks", *Current Directions in Biomedical Engineering*, vol. 10, no. 2, pp. 29–32, 2024. DOI: 10.1515/cdbme-2024-1059.

[160] D. Wulff, R. Sarau, and F. Ernst, "Target tracking in 4D ultrasound based on template matching and target forecasting using spatio-temporal autoencoders", *2023 IEEE 23th International Conference on Bioinformatics and Bioengineering (BIBE)*, pp. 113–120, 2023. DOI: 10.1109/BIBE60311.2023.00026.

[161] N. S. Hashemi, R. B. Aghdam, A. S. B. Ghiasi, and P. Fatemi, "Template Matching Advances and Applications in Image Analysis", 2016. arXiv: 1610.07231.

[162] W. Chantara, J.-H. Mun, D.-W. Shin, and Y.-S. Ho, "Object Tracking using Adaptive Template Matching", *IEIE Transactions on Smart Processing and Computing*, vol. 4, no. 1, pp. 1–9, 2015. DOI: 10.5573/IEIESPC.2015.4.1.001.

[163] N. Koizumi, T. Funamoto, J. Seo, D. Lee, H. Tsukihara, A. Nomiya, T. Azuma, K. Yoshinaka, N. Sugita, Y. Homma, Y. Matsumoto, and M. Mitsuishi, "A novel robust template matching method to track and follow body targets for NIUTS", in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1929–1936. DOI: 10.1109/ICRA.2014.6907114.

[164] M. Carletti, D. Dall'Alba, M. Cristani, and P. Fiorini, "A Robust Particle Filtering Approach with Spatially-dependent Template Selection for Medical Ultrasound Tracking Applications", in *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 3, 2016, pp. 522–531. DOI: 10.5220/0005725505220531.

[165] V. De Luca, M. Tschannen, G. Székely, and C. Tanner, "A Learning-Based Approach for Fast and Robust Vessel Tracking in Long Ultrasound Sequences", in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013. MICCAI 2013. Lecture Notes in Computer Science*, 2013, pp. 518–525. DOI: 10.1007/978-3-642-40811-3_65.

[166] S. Kondo, "Liver ultrasound tracking using long-term and short-term template matching", *MICCAI Challenge on Liver Ultrasound Tracking (CLUST' 14)*, vol. 1, pp. 13–20, 2014. [Online]. Available: https://clust.ethz.ch/opendownload/CLUST2014/Kondo_CLUST_2014.pdf.

[167] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation", in *Optical Pattern Recognition XII*, vol. 4387, 2001, pp. 95–102. DOI: 10.1117/12.421129.

[168] L. Di Stefano and S. Mattoccia, "Fast template matching using bounded partial correlation", *Machine Vision and Applications*, vol. 13, no. 4, pp. 213–221, 2003. DOI: 10.1007/s00138-002-0070-5.

[169] S. Korman, D. Reichman, G. Tsur, and S. Avidan, "FasT-Match: Fast Affine Template Matching", in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2331–2338. DOI: 10.1109/CVPR.2013.302.

[170] A. Rangamani, T. Xiong, A. Nair, T. D. Tran, and S. P. Chin, "Landmark Detection and Tracking in Ultrasound using a CNN-RNN Framework", *Conference on Neural Information Processing Systems (NIPS)*, 2016. [Online]. Available: https://www.researchgate.net/publication/316441246.

[171] Y. Zhang, X. Dai, Z. Tian, Y. Lei, J. F. Wynne, P. Patel, Y. Chen, T. Liu, and X. Yang, "Landmark tracking in liver US images using cascade convolutional neural networks with long short-term memory", *Measurement Science and Technology*, vol. 34, no. 5, p. 054 002, 2023. DOI: 10.1088/1361-6501/acb5b3.

[172] P. Huang, G. Yu, H. Lu, D. Liu, L. Xing, Y. Yin, N. Kovalchuk, L. Xing, and D. Li, "Attention-aware fully convolutional neural network with convolutional long short-term memory network for ultrasound-based motion tracking", *Medical Physics*, vol. 46, no. 5, pp. 2275–2285, 2019. DOI: 10.1002/mp.13510.

[173] K. Deepak, S. Chandrakala, and C. K. Mohan, "Residual spatiotemporal autoencoder for unsupervised video anomaly detection", *Signal, Image and Video Processing*, vol. 15, no. 1, pp. 215–222, 2021. DOI: 10.1007/s11760-020-01740-1.

[174] G. C. Sharp, S. B. Jiang, S. Shimizu, and H. Shirato, "Prediction of respiratory tumour motion for real-time image-guided radiotherapy", *Physics in Medicine and Biology*, vol. 49, no. 3, pp. 425–440, 2004. DOI: 10.1088/0031-9155/49/3/006.

[175] K. Ukon, Y. Arai, S. Takao, T. Matsuura, M. Ishikawa, H. Shirato, S. Shimizu, K. Umegaki, and N. Miyamoto, "Prediction of target position from multiple fiducial markers by partial least squares regression in real-time tumor-tracking radiation therapy", *Journal of Radiation Research*, vol. 62, no. 5, pp. 926–933, 2021. DOI: 10.1093/jrr/rrab054.

[176] A. Mylonas, J. Booth, and D. T. Nguyen, "A review of artificial intelligence applications for motion tracking in radiotherapy", *Journal of Medical Imaging and Radiation Oncology*, vol. 65, no. 5, pp. 596–611, 2021. DOI: 10.1111/1754-9485.13285.

[177] A. Gomariz, W. Li, E. Ozkan, C. Tanner, and O. Goksel, "Siamese Networks With Location Prior for Landmark Tracking in Liver Ultrasound Sequences", in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 1757–1760. DOI: 10.1109/ISBI.2019.8759382.

[178] C. Wu, T. Fu, Y. Wang, Y. Lin, Y. Wang, D. Ai, J. Fan, H. Song, and J. Yang, "Fusion Siamese network with drift correction for target tracking in ultrasound sequences", *Physics in Medicine and Biology*, vol. 67, no. 4, p. 045 018, 2022. DOI: 10.1088/1361-6560/ac4fa1.

[179] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", 2015. arXiv: 1512.03385.

[180] A. J. Shepard, B. Wang, T. K. Foo, and B. P. Bednarz, "A block matching based approach with multiple simultaneous templates for the real-time 2D ultrasound tracking of liver vessels", *Medical Physics*, vol. 44, no. 11, pp. 5889–5900, 2017. DOI: 10.1002/mp.12574.

[181] Y. Kubota, A. Matsumura, M. Fukahori, S.-i. Minohara, S. Yasuda, and H. Nagahashi, "A new method for tracking organ motion on diagnostic ultrasound images", *Medical Physics*, vol. 41, no. 9, 2014. DOI: 10.1118/1.4892065.

[182] F. Nogueira, *Bayesian Optimization: Open source constrained global optimization tool for Python*, 2014. [Online]. Available: https://github.com/fmfn/BayesianOptimization.

[183] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms", in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol. 2, 2012, pp. 2951–2959. arXiv: 1206.2944.

[184] A. H. Victoria and G. Maragatham, "Automatic tuning of hyperparameters using Bayesian optimization", *Evolving Systems*, vol. 12, no. 1, pp. 217–223, 2021. DOI: 10.1007/s12530-020-09345-2.

[185] M. Kim, G. Wu, Q. Wang, S.-W. Lee, and D. Shen, "Improved image registration by sparse patch-based deformation estimation", *NeuroImage*, vol. 105, pp. 257–268, 2015. DOI: 10.1016/j.neuroimage.2014.10.019.

[186] B. A. Chourpiliadis C. "Physiology, respiratory rate". version 2022-09-12, StatPearls. (2022), [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK537306/ (visited on 03/18/2024).

[187] C. Wilt, J. Thayer, and W. Ruml, "A Comparison of Greedy Search Algorithms", *Proceedings of the International Symposium on Combinatorial Search*, vol. 1, no. 1, pp. 129–136, 2010. DOI: 10.1609/socs.v1i1.18182.

[188] Y. Wang, T. Fu, Y. Wang, D. Xiao, Y. Lin, J. Fan, H. Song, F. Liu, and J. Yang, "Multi 3 : multi-templates siamese network with multi-peaks detection and multi-features refinement for target tracking in ultrasound image sequences", *Physics in Medicine and Biology*, vol. 67, no. 19, p. 195 007, 2022. DOI: 10.1088/1361-6560/ac9032.

[189] F. Liu, D. Liu, J. Tian, X. Xie, X. Yang, and K. Wang, "Cascaded one-shot deformable convolutional neural networks: Developing a deep learning model for respiratory motion estimation in ultrasound sequences", *Medical Image Analysis*, vol. 65, p. 101 793, 2020. DOI: 10.1016/j.media.2020.101793.

[190] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable Convolutional Networks", *International Conference on Computer Vision - ICCV 2017*, pp. 764–773, 2017. arXiv: 1703.06211.

[191] R. J. G. van Sloun, R. Cohen, and Y. C. Eldar, "Assessment of 4D Ultrasound Systems for Image-guided Radiation Therapy – Image Quality, Framerates and CT Artifacts", *Current Directions in Biomedical Engineering*, vol. 5, no. 1, pp. 245–248, 2019. DOI: 10.1515/cdbme-2019-0062.

[192] S. Ipsen, S. Böttger, H. Schwegmann, and F. Ernst, "Target tracking accuracy and latency with different 4D ultrasound systems – a robotic phantom study", *Current Directions in Biomedical Engineering*, vol. 6, no. 1, p. 20 200 038, 2020. DOI: 10.1515/cdbme-2020-0038.

[193] A. Lasso, T. Heffter, A. Rankin, C. Pinter, T. Ungi, and G. Fichtinger, "PLUS: Open-Source Toolkit for Ultrasound-Guided Intervention Systems", *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 10, pp. 2527–2537, 2014. DOI: 10.1109/TBME.2014.2322864.

[194] A. Lasso, *PLUS Toolkit*, version 2.8.0, 2024. [Online]. Available: https://plustoolkit.github.io/.

[195] J. Tokuda, *OpenIGTLink*, version 0.3.1, 2024. [Online]. Available: https://openigtlink.org/.

[196] R. Campa, J. Bernal, and I. Soto, "Kinematic modeling and control of the Hexapod parallel robot", in *2016 American Control Conference (ACC)*, 2016, pp. 1203–1208. DOI: 10.1109/ACC.2016.7525081.

[197] S. Bharadwaj, S. Prasad, and M. Almekkawy, "An Upgraded Siamese Neural Network for Motion Tracking in Ultrasound Image Sequences", *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 68, no. 12, pp. 3515–3527, 2021. DOI: 10.1109/TUFFC.2021.3095299.

[198] T. Williamson, W. Cheung, S. K. Roberts, and S. Chauhan, "Ultrasound-based liver tracking utilizing a hybrid template/optical flow approach", *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 10, pp. 1605–1615, 2018. DOI: 10.1007/s11548-018-1780-0.

[199] C. Shen, J. He, Y. Huang, and J. Wu, "Discriminative Correlation Filter Network for Robust Landmark Tracking in Ultrasound Guided Intervention", *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, pp. 646–654, 2019. DOI: 10.1007/978-3-030-32254-0_72.

[200] D. Nouri and A. Rothberg, "Liver Ultrasound Tracking using a Learned Distance Metric", *Proceedings of MICCAI workshop: Challenge on Liver Ultrasound Tracking*, 2015. [Online]. Available: https://clust.ethz.ch/opendownload/CLUST2015/nouri_clust15.pdf.

[201] S. Ipsen, R. Bruder, I. Kuhlemann, P. Jauer, L. Motisi, F. Cremers, F. Ernst, and A. Schweikard, "A visual probe positioning tool for 4D ultrasound-guided radiotherapy", in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 883–886. DOI: 10.1109/EMBC.2018.8512390.

[202] M. van Herk, P. Remeijer, C. Rasch, and J. V. Lebesque, "The probability of correct target dosage: dose-population histograms for deriving treatment margins in radiotherapy", *International Journal of Radiation Oncology Biology Physics*, vol. 47, no. 4, pp. 1121–1135, 2000. DOI: 10.1016/S0360-3016(00)00518-6.

[203] M. Pohl, M. Uesaka, H. Takahashi, K. Demachi, and R. Bhusal Chhatkuli, "Prediction of the position of external markers using a recurrent neural network trained with unbiased online recurrent optimization for safe lung cancer radiotherapy", *Computer Methods and Programs in Biomedicine*, vol. 222, p. 106 908, 2022. DOI: 10.1016/j.cmpb.2022.106908.

[204] T.-C. Çallar, E. Rueckert, and S. Böttger, "Efficient Body Registration Using Single-View Range Imaging and Generic Shape Templates", *Current Directions in Biomedical Engineering*, vol. 6, no. 3, pp. 119–122, 2020. DOI: 10.1515/cdbme-2020-3031.

[205] C.-K. Lin, F.-C. Lin, F.-L. Lian, K.-H. Chang, M.-C. Ho, J.-Y. Yen, and Y.-Y. Chen, "Ultrasound image-guided algorithms for tracking liver motion", in *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 2012, pp. 51–56. DOI: 10.1109/AIM.2012.6265916.

[206] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum Contrast for Unsupervised Visual Representation Learning", in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9726–9735. DOI: 10.1109/CVPR42600.2020.00975.

[207] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A Survey on Contrastive Self-Supervised Learning", *Technologies*, vol. 9, no. 1, 2020. DOI: 10.3390/technologies9010002.

[208] J. Aneja, A. Schwing, J. Kautz, and A. Vahdat, "A Contrastive Learning Approach for Training Variational Autoencoder Priors", *Advances in Neural Information Processing Systems*, vol. 1, pp. 480–493, 2020. arXiv: 2010.02917.

[209] H. Venables, "Practical applications of ultrasound: getting the best out of your ultrasound machine", *Ultrasound*, vol. 19, no. 1, pp. 50–55, 2011. DOI: 10.1258/ult.2010.010052.

[210] I. Matthias, N. L. Panebianco, M. G. Maltenfort, A. J. Dean, and C. Baston, "Effect of Machine Settings on Ultrasound Assessment of B-lines", *Journal of Ultrasound in Medicine*, vol. 40, no. 10, pp. 2039–2046, 2021. DOI: 10.1002/jum.15581.

[211] H. Hase, M. F. Azampour, M. Tirindelli, M. Paschali, W. Simson, E. Fatemizadeh, and N. Navab, "Ultrasound-Guided Robotic Navigation with Deep Reinforcement Learning", in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5534–5541. DOI: 10.1109/IROS45743.2020.9340913.

APPENDIX

A.1 Landmarks

The 4D ultrasound sequence data set used in this work was labeled by expert observers during the course of the work. The data set has been introduced in Tables 3.1 and 3.2. In the following Figures A.1, A.2, A.3, A.4 and A.5 all landmarks set in the liver ultrasound sequences L-Vi-1, L-Vi-2 and L-Vi-3 with $i \in \{1, 2, 3, 4, 5\}$ are visualized in a three-plane view. The slices are cut from the 3D ultrasound volumes in three different spatial planes so that the slices intersect at the landmark.

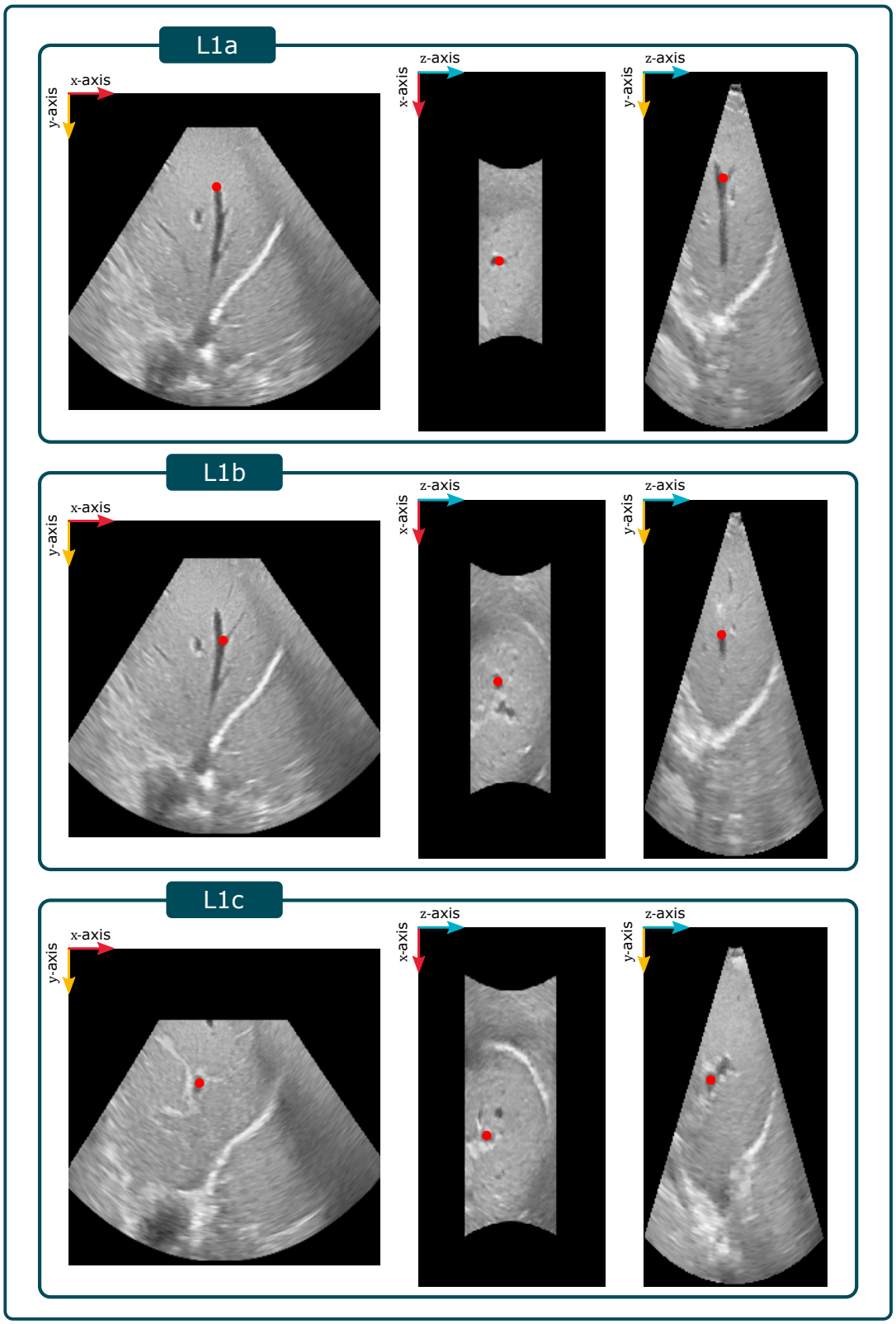

Figure A.1: Three-plane view of the landmarks L1a, L1b, and L1c set by an expert observer in the sequences L-V1-1 and L-V1-2. In the sequence L-V1-3 only the landmarks L1a and L1b were set.

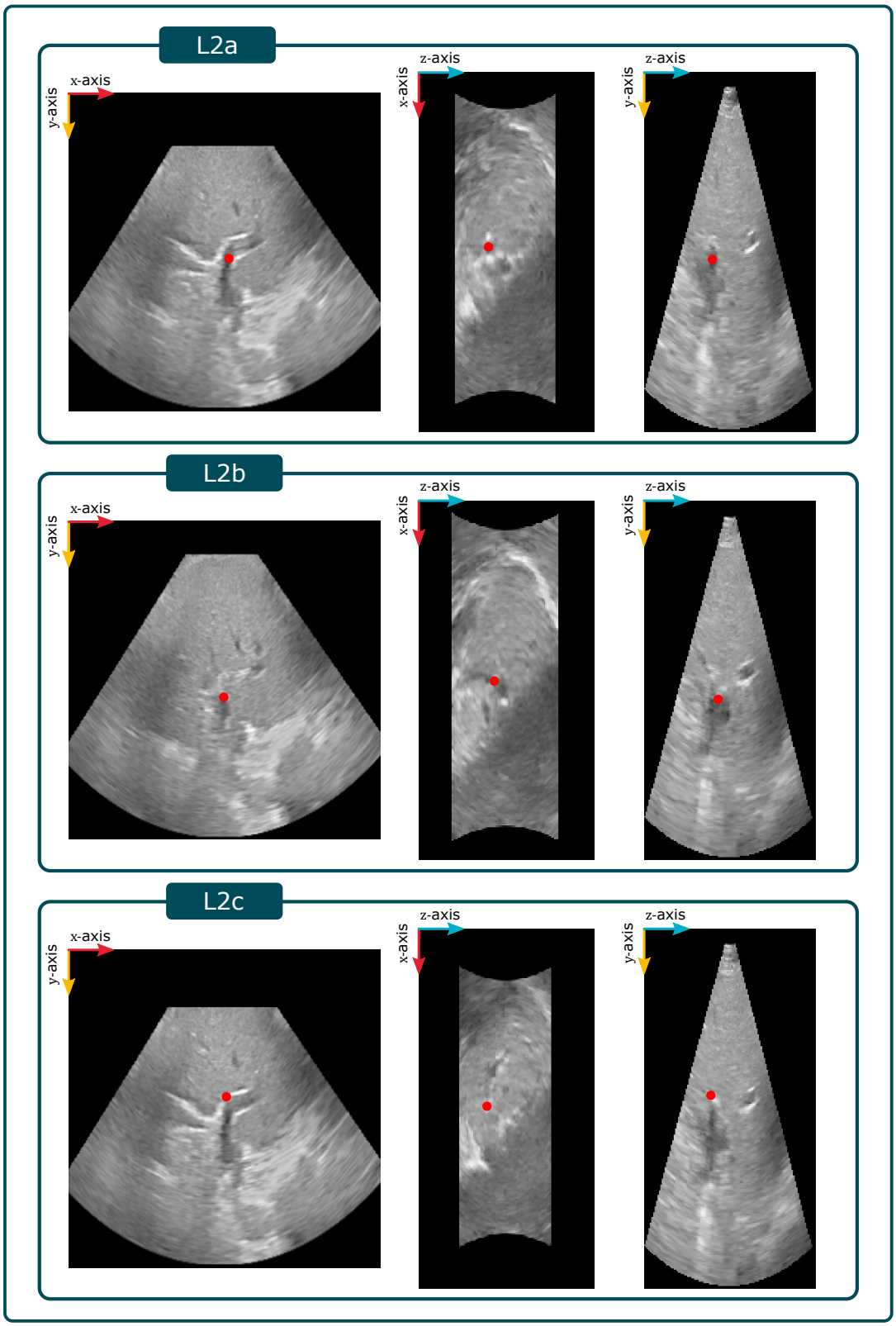

Figure A.2: Three-plane view of the landmarks L2a, L2b, and L2c set by an expert observer in the sequences L-V2-1 and L-V2-2. In the sequence L-V2-3 only the landmarks L2a and L2b were set.

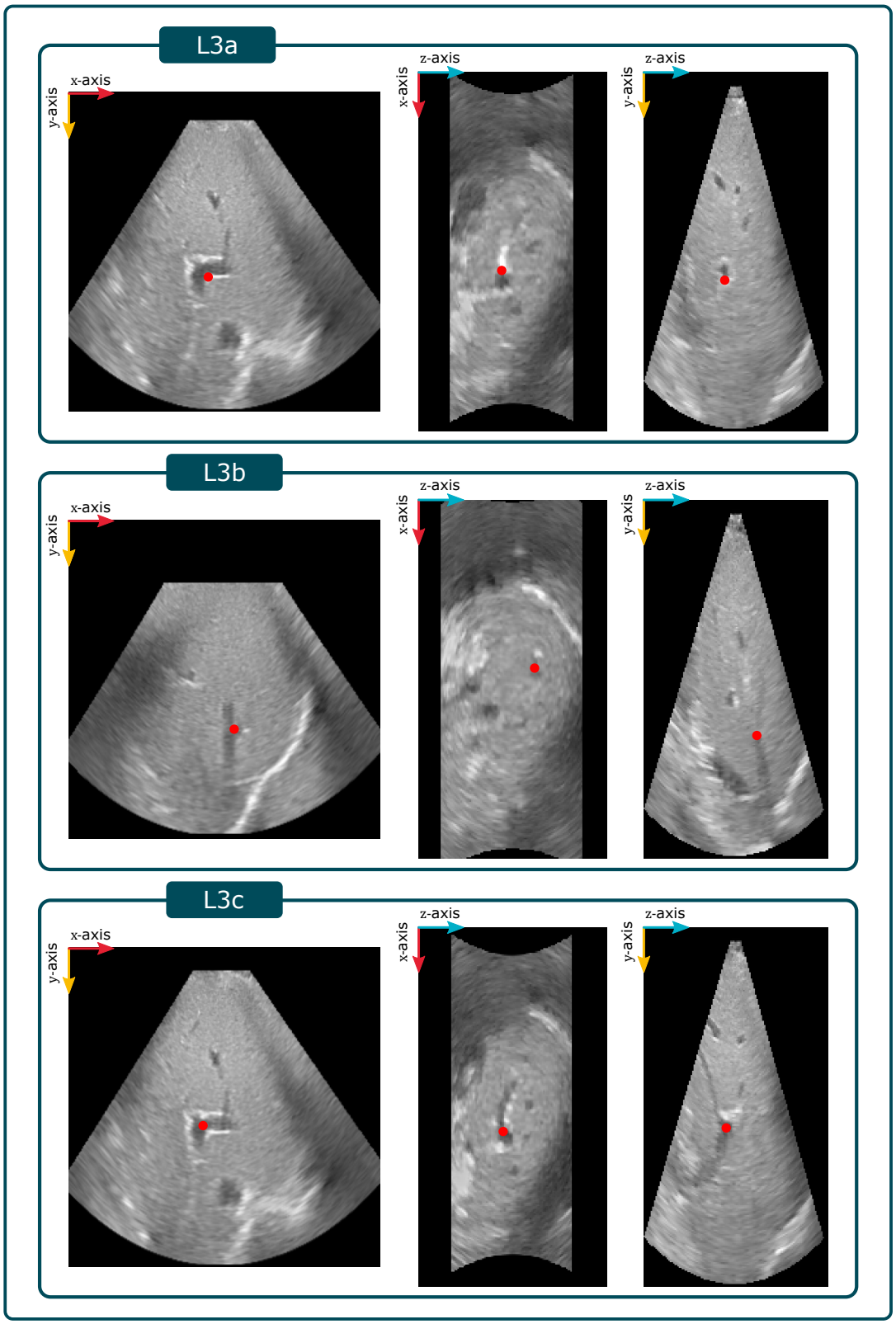

Figure A.3: Three-plane view of the landmarks L3a, L3b, and L3c set by an expert observer in the sequences L-V3-1 and L-V3-2. In the sequence L-V3-3 only the landmarks L3a and L3b were set.

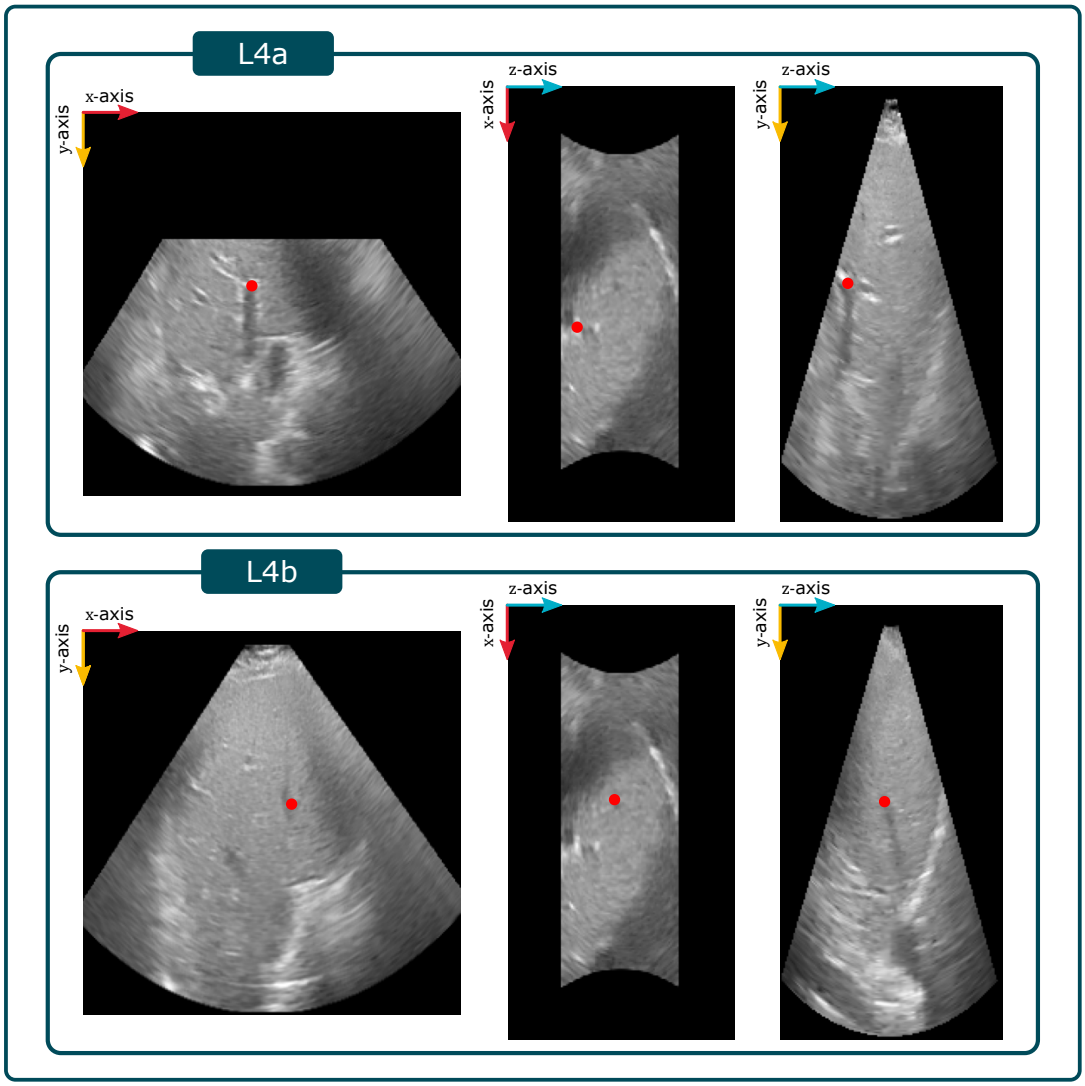

Figure A.4: Three-plane view of the landmarks L4a and L4b set by an expert observer in the sequences L-V4-1, L-V4-2 and L-V4-3.

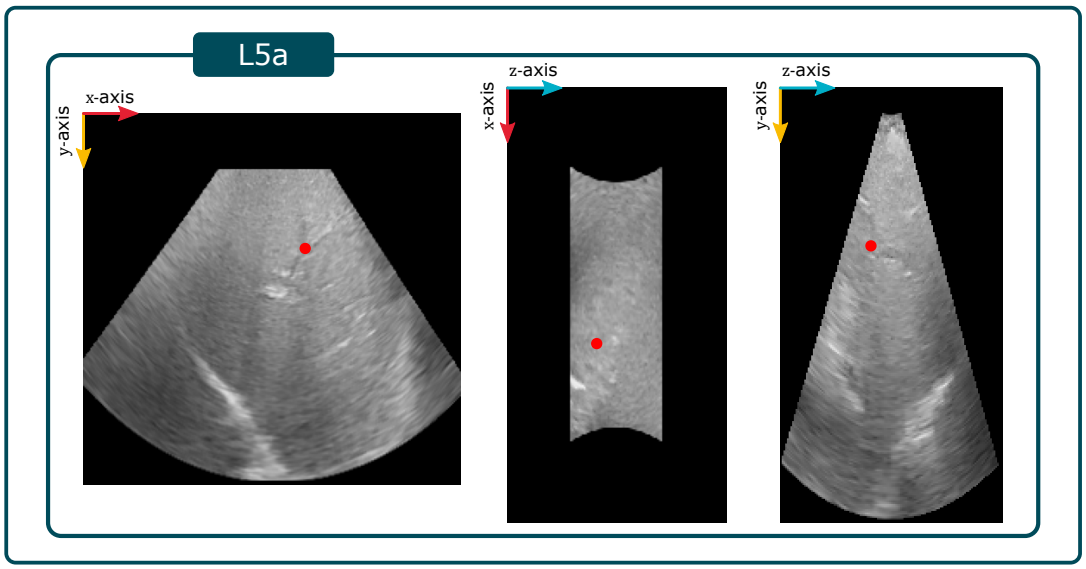

Figure A.5: Three-plane view of the landmark L5a set by an expert observer in the sequences L-V5-1 and L-V5-2.

A.2 Observer Variability

In Figure 3.5 the intra- and inter-observer variability results are presented in a box plot. Here the detailed values measured in the observer study are presented in the tables A.1, A.2 and A.3. In addition, the inter-observer variability is determined in the same way as for the CLUST data [14]. The detailed results are presented in Table A.4.

Table A.1: Results of the intra-observer variability measurements for all four observers of observer group A and every landmark L*i*a with $i \in \{1, 2, 3, 4\}$. For every landmark the best results are highlighted in bold.

Observer	**Landmark**	**Intra-observer variability(mm)**		
		Mean ± SD	**95 %tile**	**Max**
A1	L1a	1.20 ± 0.64	2.29	4.00
A1	L2a	1.83 ± 0.98	3.58	6.73
A1	L3a	**1.54 ± 0.65**	**2.78**	**3.36**
A1	L4a	2.70 ± 1.37	5.25	7.60
A1	All	1.81 ± 1.09	3.99	7.60
A2	L1a	**1.19 ± 0.53**	**2.06**	**2.81**
A2	L2a	1.52 ± 1.04	**2.84**	12.09
A2	L3a	1.86 ± 0.89	3.34	5.23
A2	L4a	**1.71 ± 0.89**	**3.39**	**5.10**
A2	All	1.58 ± 0.90	3.09	12.09
A3	L1a	1.66 ± 0.80	3.07	4.73
A3	L2a	**1.51 ± 0.72**	2.88	**4.59**
A3	L3a	1.91 ± 0.87	3.44	4.77
A3	L4a	3.27 ± 1.53	6.15	9.21
A3	All	2.05 ± 1.22	4.42	9.21
A4	L1a	1.33 ± 0.62	2.47	3.66
A4	L2a	1.86 ± 0.85	3.40	5.30
A4	L3a	2.16 ± 1.15	4.36	5.97
A4	L4a	2.44 ± 1.19	4.73	6.67
A4	All	1.95 ± 1.05	4.09	6.67

Table A.2: Results of the intra-observer variability measurements for all four observers of observer group B and every landmark Lib with $i \in \{1, 2, 3, 4\}$. For every landmark the best results are highlighted in bold.

Observer	Landmark	Intra-observer variability(mm)		
		Mean ± SD	95 %tile	Max
B1	L1b	**1.06 ± 0.54**	**1.93**	**2.90**
B1	L2b	**1.16 ± 0.65**	**2.41**	**3.98**
B1	L3b	**1.11 ± 0.67**	**2.37**	**4.41**
B1	L4b	3.23 ± 2.25	7.47	9.06
B1	All	1.60 ± 1.51	5.28	9.06
B2	L1b	2.03 ± 1.08	3.86	8.39
B2	L2b	2.10 ± 0.95	3.93	5.49
B2	L3b	2.95 ± 1.41	5.43	9.49
B2	L4b	3.52 ± 1.97	7.31	12.54
B2	All	2.63 ± 1.51	5.47	12.54
B3	L1b	1.73 ± 0.84	3.27	4.08
B3	L2b	1.84 ± 0.89	3.27	5.57
B3	L3b	2.23 ± 1.12	4.29	6.59
B3	L4b	3.53 ± 2.29	6.75	19.40
B3	All	2.31 ± 1.55	4.88	19.40
B4	L1b	1.27 ± 0.63	2.42	3.12
B4	L2b	1.51 ± 0.72	2.81	4.62
B4	L3b	2.27 ± 1.19	4.48	6.48
B4	L4b	**2.83 ± 1.45**	**5.54**	**7.37**
B4	All	1.95 ± 1.20	4.36	7.37

Table A.3: Results of the inter-observer variability measurements for both observer groups A and B and every landmark Lia and Lib with $i \in \{1, 2, 3, 4\}$.

Group	Landmark	Inter-observer variability(mm)		
		Mean ± SD	95 %tile	Max
A	L1a	1.85 ± 0.78	3.19	5.67
A	L2a	2.28 ± 1.00	3.96	10.51
A	L3a	2.19 ± 1.06	4.18	6.97
A	L4a	4.54 ± 2.16	8.06	14.67
A	All	2.68 ± 1.69	6.33	14.67
B	L1b	2.30 ± 1.00	4.10	6.42
B	L2b	2.59 ± 1.29	5.03	7.37
B	L3b	3.29 ± 1.67	6.44	9.58
B	L4b	4.15 ± 2.21	8.07	20.24
B	All	3.06 ± 1.74	6.47	20.24

Table A.4: Results of the inter-observer variability measurements calculated in the same way as done in the CLUST data for all eight observers Ai and Bi with $i \in \{1, 2, 3, 4\}$ and every landmark Lia and Lib. The best values for every observer group are highlighted in bold.

Observer	**Inter-observer variability (mm)**		
	Mean ± SD	**95 %tile**	**Max**
A1	1.68 ± 1.12	3.91	8.28
A2	**1.55 ± 0.66**	**2.68**	**4.15**
A3	1.81 ± 1.43	4.67	6.51
A4	1.60 ± 0.77	3.05	4.56
Group A	1.66 ± 1.04	3.99	8.28
B1	**1.52** ± 0.92	**3.24**	5.32
B2	2.54 ± 1.13	4.61	7.55
B3	1.91 ± 1.16	4.31	6.88
B4	1.90 ± **0.71**	**3.24**	**4.75**
Group B	1.97 ± 1.06	3.90	7.55

A.3 Neural Network Architectures

Different neural networks, especially autoencoders, were implemented in this work. For simplicity, in the corresponding sections the architectures were shown graphically without information about the exact parameters such as number of kernels and kernel size of convolutional layers. In the following, these parameters are presented in tables that may be helpful for re-implementing the networks.

Table A.5: Parameters of the autoencoder presented in section 4.2.1 and illustrated in Figure 4.4. This autoencoder was used to learn and reconstruct representations from 3D ultrasound patches.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Encoder φ:				
Input	(30, 30, 30, 1)	-	-	-
Convolution	(30, 30, 30, 16)	PReLU	16	(3, 3, 3)
Average-pooling	(15, 15, 15, 16)	-	-	(2, 2, 2)
Convolution	(15, 15, 15, 32)	PReLU	32	(3, 3, 3)
Average-pooling	(8, 8, 8, 32)	-	-	(2, 2, 2)
Convolution	(8, 8, 8, 64)	PReLU	64	(3, 3, 3)
Convolution	(8, 8, 8, 64)	PReLU	64	(3, 3, 3)
Average-pooling	(4, 4, 4, 64)	-	-	(2, 2, 2)
Flatten	(4096)	-	-	-
Fully-connected	(256)	PReLU	256	-
Representation	(50)	tanh	50	-
Decoder δ:				
Fully-connected	(256)	PReLU	256	-
Fully-connected	(4096)	PReLU	4096	-
Reshape	(4, 4, 4, 64)	-	-	-
Upsampling	(8, 8, 8, 64)	-	-	(2, 2, 2)
Convolution	(8, 8, 8, 32)	PReLU	32	(3, 3, 3)
Upsampling	(16, 16, 16, 32)	-	-	(2, 2, 2)
Convolution	(16, 16, 16, 16)	PReLU	16	(3, 3, 3)
Convolution	(16, 16, 16, 16)	PReLU	16	(3, 3, 3)
Upsampling	(32, 32, 32, 16)	-	-	(2, 2, 2)
Convolution	(32, 32, 32, 16)	PReLU	16	(3, 3, 3)
Convolution	(30, 30, 30, 1)	sigmoid	1	(3, 3, 3)

Table A.6: Parameters of the sliced-Wasserstein autoencoder presented in section 4.2.3 and illustrated in Figure 4.9. This architecture was used for investigating the optimal dimensionality of the representation space size and for performing target tracking.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Encoder φ:				
Input	(24, 24, 24, 1)	-	-	-
Convolution	(24, 24, 24, 16)	PReLU	16	(3, 3, 3)
Convolution	(24, 24, 24, 16)	PReLU	16	(3, 3, 3)
Average-pooling	(12, 12, 12, 16)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 32)	PReLU	32	(3, 3, 3)
Convolution	(12, 12, 12, 32)	PReLU	32	(3, 3, 3)
Average-pooling	(6, 6, 6, 32)	-	-	(2, 2, 2)
Flatten	(6912)	-	-	-
Fully-connected	(1024)	PReLU	1024	-
Fully-connected	(512)	PReLU	512	-
Fully-connected	(256)	PReLU	256	-
Representation	(128)	sigmoid	128	-
Decoder δ:				
Fully-connected	(256)	PReLU	256	-
Fully-connected	(6912)	PReLU	6912	-
Reshape	(6, 6, 6, 32)	-	-	-
Convolution	(6, 6, 6, 32)	PReLU	32	(3, 3, 3)
Convolution	(6, 6, 6, 32)	PReLU	32	(3, 3, 3)
Upsampling	(12, 12, 12, 32)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 16)	PReLU	16	(3, 3, 3)
Convolution	(12, 12, 12, 16)	PReLU	16	(3, 3, 3)
Upsampling	(24, 24, 24, 16)	-	-	(2, 2, 2)
Convolution	(24, 24, 24, 1)	sigmoid	1	(1, 1, 1)

Table A.7: Parameters of the variational autoencoder presented in section 5.1.2 and illustrated in Figure 5.1. This architecture was used for ultrasound image augmentation.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Encoder φ:				
Input	(128, 128, 1)	-	-	-
Convolution	(64, 64, 16)	LReLU	16	(3, 3)
Convolution	(64, 64, 16)	LReLU	16	(3, 3)
Convolution	(32, 32, 32)	LReLU	32	(3, 3)
Convolution	(32, 32, 32)	LReLU	32	(3, 3)
Convolution	(16, 16, 64)	LReLU	64	(3, 3)
Convolution	(16, 16, 64)	LReLU	64	(3, 3)
Convolution	(8, 8, 128)	LReLU	128	(3, 3)
Convolution	(1, 1, 128)	LReLU	128	(8, 8)
Representation	(64)	tanh	64	-
Decoder δ:				
Transpose convolution	(8, 8, 128)	LReLU	128	(8, 8)
Transpose convolution	(16, 16, 128)	LReLU	128	(3, 3)
Transpose convolution	(16, 16, 64)	LReLU	64	(3, 3)
Transpose convolution	(32, 32, 64)	LReLU	64	(3, 3)
Transpose convolution	(32, 32, 32)	LReLU	32	(3, 3)
Transpose convolution	(64, 64, 32)	LReLU	32	(3, 3)
Transpose convolution	(64, 64, 16)	LReLU	16	(3, 3)
Transpose convolution	(128, 128, 1)	sigmoid	1	(3, 3)

Table A.8: Parameters of the GAN-generator presented in section 5.1.2 and illustrated in Figure 5.2. This architecture was used for ultrasound image augmentation.

Layer type	Output shape	Activation function	Units/ filters	Kernel size
Ultrasound image branch:				
Input	(128, 128, 1)	-	-	-
Convolution	(128, 128, 16)	LReLU	16	(4, 4)
Convolution	(64, 64, 32)	LReLU	32	(4, 4)
Convolution	(32, 32, 64)	LReLU	64	(4, 4)
Convolution	(16, 16, 128)	LReLU	128	(4, 4)
Convolution	(8, 8, 128)	linear	128	(4, 4)
Noise vector branch:				
Input	(100)	-	-	-
Fully-connected	(16384)	linear	16384	-
Reshape	(8, 8, 128)	-	-	-
Decoder:				
Convolution	(8, 8, 128)	ReLU	128	(4, 4)
UpSampling2D	(16, 16, 128)	-	-	(2, 2)
Convolution	(16, 16, 64)	ReLU	64	(4, 4)
UpSampling2D	(32, 32, 64)	-	-	(2, 2)
Convolution	(32, 32, 32)	ReLU	32	(4, 4)
UpSampling2D	(64, 64, 32)	-	-	(2, 2)
Convolution	(64, 64, 16)	ReLU	16	(4, 4)
UpSampling2D	(128, 128, 16)	-	-	(2, 2)
Convolution	(128, 128, 1)	tanh	1	(4, 4)

Table A.9: Parameters of the GAN-discriminator presented in section 5.1.2 and illustrated in Figure 5.2. This architecture was used for ultrasound image augmentation.

Layer type	Output shape	Activation function	Units/ filters	Kernel size
First branch:				
Input	(128, 128, 1)	-	-	-
Convolution	(64, 64, 32)	LReLU	32	(4, 4)
Convolution	(32, 32, 64)	LReLU	64	(4, 4)
Convolution	(16, 16, 128)	LReLU	128	(4, 4)
Convolution	(8, 8, 256)	LReLU	256	(4, 4)
Second branch:				
Input	(128, 128, 1)	-	-	-
Convolution	(64, 64, 32)	LReLU	32	(4, 4)
Convolution	(32, 32, 64)	LReLU	64	(4, 4)
Convolution	(16, 16, 128)	LReLU	128	(4, 4)
Convolution	(8, 8, 256)	LReLU	256	(4, 4)
Output:				
Reshape	(16384)	-	-	-
Fully-connected	(1)	tanh	1	-

Table A.10: Parameters of the conditional variational autoencoder (CVAE) presented in section 5.2.2 and illustrated in Figure 5.5. This architecture was used for deformable 3D ultrasound patch augmentation.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Condition encoder ψ:				
Input	(32, 32, 32, 1)	-	-	-
Convolution	(32, 32, 32, 32)	elu	32	(3, 3, 3)
Max-pooling	(16, 16, 16, 32)	-	-	(2, 2, 2)
Convolution	(16, 16, 16, 32)	elu	32	(3, 3, 3)
Max-pooling	(8, 8, 8, 32)	-	-	(2, 2, 2)
Convolution	(8, 8, 8, 32)	elu	32	(3, 3, 3)
Max-pooling	(4, 4, 4, 32)	-	-	(2, 2, 2)
Convolution	(4, 4, 4, 32)	elu	32	(3, 3, 3)
Max-pooling	(2, 2, 2, 32)	-	-	(2, 2, 2)
Convolution	(2, 2, 2, 32)	elu	32	(3, 3, 3)
Flatten	(256)	-	-	-
Fully-connected	(2)	linear	2	-
Encoder φ:				
Input	(32, 32, 32, 1)	-	-	-
Convolution	(32, 32, 32, 32)	elu	32	(3, 3, 3)
Max-pooling	(16, 16, 16, 32)	-	-	(2, 2, 2)
Convolution	(16, 16, 16, 32)	elu	32	(3, 3, 3)
Max-pooling	(8, 8, 8, 32)	-	-	(2, 2, 2)
Convolution	(8, 8, 8, 32)	elu	32	(3, 3, 3)
Max-pooling	(4, 4, 4, 32)	-	-	(2, 2, 2)
Convolution	(4, 4, 4, 32)	elu	32	(3, 3, 3)
Flatten	(2048)	-	-	-
Fully-connected	(512)	linear	512	-
Representation	(514)	-	-	-
Decoder δ:				
Fully-connected	(2048)	linear	2048	-
Reshape	(4, 4, 4, 32)	-	-	-
Upsampling	(8, 8, 8, 32)	-	-	(2, 2, 2)
Convolution	(8, 8, 8, 32)	elu	32	(3, 3, 3)
Upsampling	(16, 16, 16, 32)	-	-	(2, 2, 2)
Convolution	(16, 16, 16, 32)	elu	32	(3, 3, 3)
Upsampling	(32, 32, 32, 32)	-	-	(2, 2, 2)
Convolution	(32, 32, 32, 32)	elu	32	(3, 3, 3)
Convolution	(32, 32, 32, 3)	linear	3	(3, 3, 3)

Table A.11: Parameters of the spatio-temporal autoencoder presented in section 6.1.2 and illustrated in Figure 6.2. This architecture was used for target appearance forecasting in a template matching-based tracking procedure.

Layer type	Output shape	Activation function	Units/ filters	Kernel size
Encoder φ:				
Input	(3, 24, 24, 24, 1)	-	-	-
Timedist. convolution	(3, 24, 24, 24, 128)	-	128	(3, 3, 3)
Timedist. convolution	(3, 24, 24, 24, 128)	-	128	(3, 3, 3)
Timedist. convolution	(3, 24, 24, 24, 128)	-	128	(3, 3, 3)
Max-pooling	(3, 12, 12, 12, 128)	-	-	(2, 2, 2)
Timedist. convolution	(3, 12, 12, 12, 64)	-	64	(3, 3, 3)
Timedist. convolution	(3, 12, 12, 12, 64)	-	64	(3, 3, 3)
Timedist. convolution	(3, 12, 12, 12, 64)	-	64	(3, 3, 3)
Max-pooling	(3, 6, 6, 6, 64)	-	-	(2, 2, 2)
Timedist. convolution	(3, 6, 6, 6, 32)	-	32	(3, 3, 3)
Timedist. convolution	(3, 6, 6, 6, 32)	-	32	(3, 3, 3)
Timedist. convolution	(3, 6, 6, 6, 32)	-	32	(3, 3, 3)
Max-pooling	(3, 3, 3, 3, 32)	-	-	(2, 2, 2)
Conv. LSTM	(3, 3, 3, 3, 32)	tanh	32	(2, 2, 2)
Convolution	(3, 3, 3, 32)	ReLU	32	(3, 3, 3)
Convolution	(3, 3, 3, 32)	-	32	(3, 3, 3)
Concatenate	(3, 3, 3, 32)	-	-	-
Convolution	(3, 3, 3, 32)	ReLU	32	(3, 3, 3)
Convolution	(3, 3, 3, 32)	-	32	(3, 3, 3)
Concatenate	(3, 3, 3, 32)	-	-	-
Flatten	(864)	-	-	-
Representation	(128)	linear	128	-
Decoder δ:				
Fully-connected	(864)	ReLU	864	-
Reshape	(3, 3, 3, 32)	-	-	-
Upsampling	(6, 6, 6, 32)	-	-	(2, 2, 2)
Convolution	(6, 6, 6, 32)	tanh	32	(3, 3, 3)
Convolution	(6, 6, 6, 32)	tanh	32	(3, 3, 3)
Convolution	(6, 6, 6, 32)	tanh	32	(3, 3, 3)
Upsampling	(12, 12, 12, 32)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 32)	tanh	32	(3, 3, 3)
Convolution	(12, 12, 12, 32)	tanh	32	(3, 3, 3)
Convolution	(12, 12, 12, 32)	tanh	32	(3, 3, 3)
Upsampling	(24, 24, 24, 32)	-	-	(2, 2, 2)
Convolution	(24, 24, 24, 32)	tanh	32	(3, 3, 3)
Convolution	(24, 24, 24, 32)	tanh	32	(3, 3, 3)
Convolution	(24, 24, 24, 1)	sigmoid	1	(3, 3, 3)

Table A.12: Parameters of the deep localization network (DLN) presented in section 6.1.3 and illustrated in Figure 6.7. This architecture was used for performing target tracking.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Input	(48, 48, 48, 1)	-	-	-
Convolution	(48, 48, 48, 16)	ReLU	16	(3, 3, 3)
Convolution	(48, 48, 48, 16)	-	16	(3, 3, 3)
Concatenate	(48, 48, 48, 16)	-	-	-
Average-pooling	(24, 24, 24, 16)	-	-	(2, 2, 2)
Convolution	(24, 24, 24, 32)	ReLU	32	(3, 3, 3)
Convolution	(24, 24, 24, 32)	-	32	(3, 3, 3)
Concatenate	(24, 24, 24, 32)	-	-	-
Average-pooling	(12, 12, 12, 32)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 64)	ReLU	64	(3, 3, 3)
Convolution	(12, 12, 12, 64)	-	64	(3, 3, 3)
Concatenate	(12, 12, 12, 64)	-	-	-
Average-pooling	(6, 6, 6, 64)	-	-	(2, 2, 2)
Flatten	(13824)	-	-	-
Fully-connected	(512)	ReLU	512	-
Fully-connected	(256)	ReLU	256	-
Fully-connected	(3)	ReLU	3	-

Table A.13: Parameters of the autoencoder presented in section 6.2.1 and illustrated in Figure 6.9. The autoencoder was used for feature description to perform target tracking.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Encoder φ:				
Input	(24, 24, 24, 1)	-	-	-
Convolution	(24, 24, 24, 32)	PReLU	32	(3, 3, 3)
Convolution	(24, 24, 24, 32)	PReLU	32	(3, 3, 3)
Average-pooling	(12, 12, 12, 32)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 64)	PReLU	64	(3, 3, 3)
Convolution	(12, 12, 12, 64)	PReLU	64	(3, 3, 3)
Convolution	(12, 12, 12, 64)	PReLU	64	(3, 3, 3)
Average-pooling	(6, 6, 6, 64)	-	-	(2, 2, 2)
Flatten	(13824)	-	-	-
Fully-connected	(512)	PReLU	512	-
Fully-connected	(256)	PReLU	256	-
Representation	(256)	tanh	256	-
Decoder δ:				
Fully-connected	(256)	PReLU	256	-
Fully-connected	(13824)	PReLU	13824	-
Reshape	(6, 6, 6, 64)	-	-	-
Upsampling	(12, 12, 12, 64)	-	-	(2, 2, 2)
Convolution	(12, 12, 12, 64)	PReLU	64	(3, 3, 3)
Convolution	(12, 12, 12, 64)	PReLU	64	(3, 3, 3)
Upsampling	(24, 24, 24, 64)	-	-	(2, 2, 2)
Convolution	(24, 24, 24, 32)	PReLU	32	(3, 3, 3)
Convolution	(24, 24, 24, 1)	PReLU	1	(3, 3, 3)

Table A.14: Parameters of the deformable convolutional autoencoder (DCAE) presented in section 6.2.4 and illustrated in Figure 6.20. The model was used to perform target tracking.

Layer type	**Output shape**	**Activation function**	**Units/ filters**	**Kernel size**
Encoder φ:				
Input	(24, 24, 9)	-	-	-
(Deform.) convolution	(24, 24, 32)	LReLU	32	(5, 5)
Convolution	(20, 20, 64)	LReLU	64	(5, 5)
Max-pooling	(10, 10, 64)	-	-	(2, 2)
(Deform.) convolution	(10, 10, 64)	LReLU	64	(3, 3)
Convolution	(8, 8, 128)	LReLU	128	(3, 3)
Max-pooling	(4, 4, 128)	-	-	(2, 2)
Flatten	(2048)	-	-	-
Fully-connected	(512)	PReLU	512	-
Fully-connected	(256)	PReLU	256	-
Representation	(128)	tanh	128	-
Decoder δ:				
Fully-connected	(256)	PReLU	256	-
Fully-connected	(512)	PReLU	512	-
Fully-connected	(2048)	PReLU	2048	-
Reshape	(4, 4, 128)	-	-	-
Upsampling	(8, 8, 128)	-	-	(2, 2)
Transpose convolution	(10, 10, 64)	LReLU	64	(3, 3)
(Deform.) convolution	(10, 10, 64)	LReLU	64	(3, 3)
Upsampling	(20, 20, 64)	-	-	(2, 2)
Transpose convolution	(24, 24, 32)	LReLU	32	(5, 5)
(Deform.) convolution	(24, 24, 32)	LReLU	32	(5, 5)
Convolution	(24, 24, 9)	-	9	(1, 1)

LIST OF ABBREVIATIONS

2D ultrasound	Two-dimensional ultrasound image with shape $m \cdot n$ px.
2D+t ultrasound	Time-resolved 2D ultrasound sequence with shape $t \cdot m \cdot n$ where t is the time.
3D ultrasound	Three-dimensional ultrasound image with shape $m \cdot n \cdot l$ vx.
4D ultrasound	Time-resolved 3D ultrasound sequence with shape $t \cdot m \cdot n \cdot l$ where t is the time. Could also be referred to as 3D+t.
AAPM	American association of physicists in medicine.
AE	Autoencoder.
AI	Artificial intelligence.
AP	Anterior-posterior.
BRIEF	Binary robust independent elementary features.
BRIEF-3D	BRIEF for 3D image data.
BRISK	Binary robust invariant scalable keypoints.
BRISK-3D	BRISK for 3D image data.
CBCT	Cone-beam computed tomography.
CH-score	Calinski-Harabasz score.
CLUST	Challenge on liver ultrasound tracking.
CNN	Convolutional neural network.
ConvLSTM	Convolutional long short-term memory layer.
CPU	Central processing unit.
CT	Computed tomography.
CVAE	Conditional variational autoencoder.
DCAE	Deformable convolutional autoencoder.
DLN	Deep localization network.
DoF	Degrees of freedom.
DoG	Difference of gaussian.
ELU	Exponential linear unit.
FAST	Features from accelerated segment test.
FAST-3D	FAST for 3D image data.
FTT	Fixed target tracking.

GAN	Generative adversarial network.
GPU	Graphics processing unit.
GUI	Graphical user interface.
ICC	Intra-class correlation coefficient.
KLD	Kullback-Leibler divergence.
LINAC	Linear accelerator.
LORA	Level of robot autonomy.
LR	Left-right.
LReLU	Leaky rectified linear unit.
LSTM	Long short-term memory layer.
MAE	Mean absolute error.
MLP	Multi-layer perceptron.
MRI	Magnetic resonance imaging.
MSE	Mean squared error.
NCC	Normalized cross-correlation.
ORB	Oriented FAST and rotated BRIEF.
PReLU	Parameterized rectified linear unit.
PTT	Predicted target tracking.
px	Pixel - Element of a 2D image.
ReLU	Rectified linear unit.
RMSE	Root mean squared error.
RNN	Recurrent neural network.
ROI	Region-of-interest.
SI	Superior-inferior.
SIFT	Scale invariant feature transform.
SNN	Siamese neural network.
SURF	Speeded-up robust features.
SUSAN	Smallest univalue segment assimilating nucleus.
SWAE	Sliced-Wasserstein autoencoder.
Tanh	Tangens hyperbolicus.
TE	Tracking error.
TRE	Target registration error.
UTT	Updated target tracking.
VAE	Variational autoencoder.
vx	Voxel - Element of a 3D image.
WPTT	Weighted predicted target tracking.

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

Während der Arbeit an dieser Dissertation habe ich Höhen und Tiefen erlebt, durch die mich viele wichtige Menschen begleitet, unterstützt und aufgebaut haben.
An erster Stelle möchte ich mich bei Prof. Dr. Floris Ernst bedanken, der mir als Doktorvater und Mentor jederzeit mit Rat und Tat zur Seite stand. Vielen Dank für die Möglichkeit, dieses interessante Thema zu bearbeiten und einen Beitrag zur Forschung am Institut zu leisten.
Des Weiteren möchte ich mich bei allen Kollegen und Ehemaligen des Instituts für Robotik und Kognitive Systeme bedanken, die mich auf meinem Weg begleitet haben. Die wertvollen Diskussionen, Tipps und Anregungen während unserer Kaffeepausen haben mich immer wieder einen Schritt weitergebracht. Ohne eure Unterstützung und euer Fachwissen wäre diese Arbeit nicht möglich gewesen. Insbesondere möchte ich mich bei Jonas Osburg, Niclas Erben, Marius Krusen und Cornelia Rieckhoff für eure wertvollen Kommentare zu dieser Arbeit bedanken. Ein großer Dank geht auch an Dr. Svenja Ipsen, die mich bereits während meiner Bachelorarbeit in dieses spannende Forschungsfeld eingeführt hat, und an Prof. Dr. Jannis Hagenah, der mich insbesondere zu Beginn meiner Promotionszeit fachlich sehr unterstützt hat. Ebenso wichtig für meine Arbeit waren die Studierenden, die ich während meiner Promotionszeit betreuen durfte. Insbesondere möchte ich mich bei Kevin Zander, Moritz Franz, Timon Dohnke, Cassandra Krause, Ricardo Sarau und Laslo Guenther für die Unterstützung meiner Arbeit bedanken.
Von ganzem Herzen möchte ich mich bei meiner Familie bedanken, auf die ich mich immer verlassen kann. Danke Mama, Papa und Tobias für eure Geduld, euren Zuspruch und eure bedingungslose Unterstützung auf meinem Weg. Ihr habt mich in jeder Phase dieser Arbeit ermutigt und mir stets den Rücken gestärkt. Ohne euch wäre diese Arbeit nie entstanden. Zuletzt möchte ich mich ganz herzlich bei Sonja und Malea bedanken. Eure Liebe und Freude haben mir immer wieder neue Kraft gegeben, mich an die Arbeit zu machen. Vielen Dank, Sonja, für deine aufopferungsvolle Unterstützung und deine Nachsicht mit mir, wenn ich mal nicht so gut drauf war. Du hast mir nicht nur in praktischen Dingen geholfen, sondern warst auch mein emotionaler Anker in stressigen Zeiten. Danke, Malea, für deine unbändige Energie und dein strahlendes Lachen, das mir jeden Tag neue Freude und Zuversicht gegeben hat. Dein Lachen war oft der Lichtblick, der mich motiviert hat, weiterzumachen.