Master's thesis

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

Sven Goffin

Improved Strain Computation for Transesophageal Echocardiography Acquisitions

Master's thesis in Electronic Systems Design
Supervisor: Gabriel Hanssen Kiss and Ilangko Balasingham
June 2021

NTNU
Norwegian University of Science and Technology

Sven Goffin

Improved Strain Computation for Transesophageal Echocardiography Acquisitions

Master's thesis in Electronic Systems Design
Supervisor: Gabriel Hanssen Kiss and Ilangko Balasingham
June 2021

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

NTNU
Norwegian University of
Science and Technology

Abstract

Cardiac surgeries are major interventions prone to serious complications that can occur during and after the operation. Monitoring the heart allows to anticipate such complications and has thus become a standard practice to perform through the perioperative period. Echocardiographic assessment of the myocardial contractility is a common monitoring procedure that is generally performed by visual inspection. The qualitative nature of this technique makes it highly vulnerable to the operator's subjectivity and thus drove cardiologists to develop standardized quantitative measures of cardiac function, such as strain. Strain estimation still requires manual annotation of images and still suffers from inter- and intra-observer variability.

This thesis is presented as a contribution towards the complete automatization of the strain estimation task in transesophageal echocardiographic (TEE) images. A novel strain estimation pipeline is proposed. It focuses on the estimation of the longitudinal strain in the basal segments of the 4-chamber, 2-chamber and apical long-axis views of the heart. This pipeline uses the segmentation model U-Net and a custom thinning algorithm to automatically extract myocardial points from the first frame of a B-mode sequence and estimates their motion with optical flow methods. Strain is then computed based on the estimated displacement of these points through cardiac cycles. Four optical flow models are experimented, among which two have a convolutional neural network-based architecture. Integration of tissue velocity imaging (TVI) data and a novel tracking method based on Kalman filtering are developed in order to improve the motion estimation and tracking processes.

U-Net and the two CNN-based models are trained on B-mode recordings from 70 patients. A test set of 18 patients is used to evaluate the tracking and strain estimation performances of the different models. The myocardial point extraction algorithm gives usable results in 50% and 57% of the cases when applied to high and low frame rate B-mode sequences respectively. Three optical flow algorithms present outstanding tracking performances in five of the six basal segments. It is shown that exploiting TVI data improves tracking performances. The same observation is made when the Kalman filtering-based tracking method is applied to high frame rate sequences. The proposed techniques achieve state-of-the-art strain

estimation performances. A mean absolute error of $(2.74 + 2.38)\%$ is achieved in the inferoseptal segment. The inferior and anterior segments are the segments in which the correlation between strain estimates and ground truth values is the highest: the Pearson correlation coefficient reaches the value 0.77 in the inferior segment and 0.79 in the anterior segment.

Preface

This thesis brings my electrical engineering curriculum to an end. These years at university were probably the most enriching of my life, both on a personal and professional level. They have left me with plenty of knowledge, and even more importantly, with a lot of new friends that I am sure will keep an indispensable role to play in my future life. Two years ago, I challenged myself and left Belgium to come study at NTNU in Trondheim, Norway. I would be lying if I said it was easy but I am proud of my journey and completing this thesis proved I fulfilled the goals I set to myself.

I chose this project mainly because of its biomedical aspect. Indeed, putting the skills and knowledge I have acquired throughout my studies at the service of a science as useful as medicine is meaningful to me. Moreover, I was sure this project would confront me to new challenges that would require advanced programming skills as well as a deep understanding of the topic. Now that I delivered this thesis, I can say it was true.

This project gave me the exceptional opportunity to collaborate with experts in another field than my own. I learned a lot from this experience: discussing my point of view with skillful people brought me a new insight on my work and allowed me to strengthen the weaknesses of the scientific approach I follow. Diving deep into one project motivated me to pursue my university curriculum and to apply for a Ph.D position in my native country.

Acknowledgments

I would like to thank Håvard Dalen (Ph.D., M.D.), Espen Holte (Ph.D., M.D.), and Bjørnar Grenne (Ph.D., M.D.) who acquired the ultrasound recordings used in this thesis, and without who this work would have been impossible to accomplish. I also thank Erik Andreas Rye Berg (M.D.), who provided ground truth values of strain for a subset of the acquired data. This time-consuming task greatly helped to draw conclusions about the different methods implemented in this thesis.

My supervisor, Dr. Gabriel Hanssen Kiss, deserves much more than my simple acknowledgments. His advice was instrumental. He granted a tremendous amount

of his time by following closely the progress of the project. He made it a point of honor to always be available and answered every email I sent with pertinent comments and suggestions. He also provided valued feedback on my writings. For his precious help and the numerous meetings we had through the whole duration of the project, he has my most sincere gratitude.

For their support and their help for proof-reading, I owe a big thank to my friends and fellow students Chloé Bolle and Cyril Georgtay. For the moral support they gave me during the entire semester, I also thank my girlfriend Ann Iren Fossøy, as well as my mother and sister, Fabienne Marion and Oriane Goffin.

Contents

Figures

Tables

Chapter 1

Introduction

Preliminary work of this thesis was conducted at NTNU during the Autumn semester 2020 in the framework of the course *TFE4590 - Specialization Project*. Although significantly extended, some parts of the introduction and theoretical background chapters have thus been adapted from the specialization project report.

1.1 Background

Monitoring the autonomic function of the heart has become a standard practice in various surgical interventions. In high-risk surgeries, such monitoring allows for a quick and reliable assessment of patients' health. Indeed, the characteristics exhibited by the autonomic function of the heart are affected by the seriousness of the undergoing surgical procedure. [1] As for a dysfunction of the automatic nervous system, cardiovascular function imbalance causes the risk of sudden death in operated patients to rise. [2, 3] Some interventions like cardiac valve replacements and coronary artery bypass grafting can destabilize the automatic cardiovascular function, which could potentially decrease myocardial contractility, and in some cases, lead to atrial fibrillation and myocardial infraction. [4, 5] These are as many reasons for surgeons to show a growing interest for heart function monitoring during operations.

In most cases, heart function monitoring is performed visually by analyzing echocardiographic images. [6] Echocardiography consists in building two- or three-dimensional images of the heart by analyzing the attenuation and reflections undergone by ultrasound waves in body tissues. Contrary to MRI and computed tomography scanners, ultrasound imaging techniques are cheap and non-invasive. This is also a portable technology, which makes it particularly well suited for real-time monitoring applications and to be used in operating rooms. [7, 8] Transthoracic echocardiography (TTE) is the most popular technique and uses an external ultrasound probe placed on the thorax to retrieve images from the heart. Its popularity is mainly due to its ease of use. [9] Besides TTE, transesophageal

echocardiography (TEE) has gained in popularity among physicians since its introduction in the 80's. In this procedure, the ultrasound probe is placed in the patient's esophagus. The ultrasound waves can thus reach the heart with a clearer and more direct path. This results in good-quality ultrasound images of cardiac structures. [10] Complications that could occur due to a poor probe placement or manipulation are rare. [11] Moreover, TEE proved itself to be a tool of choice in the diagnosis of cardiovascular disease and failure such as prosthetic-valve failure and endocarditis. When used in operating rooms, the TEE probe stays in place for the entire duration of the surgery. Acquisition of new images is therefore possible without the need for an operator to be present, unlike TTE. [12]

Reichert *et al.* [13] proved that the myocardial strain has a prognostic value in patients undergoing cardiac surgery. In 2010, Dalen *et al.* [14] proposed some reference values for myocardial strain in healthy patients. The myocardial strain is a measure of the myocardial contractility, one of the three factors governing the systolic volumes of the left ventricle. [15] Consequently, the systolic global longitudinal strain is highly correlated to the left ventricular ejection fraction (LVEF), which is the volumetric difference in percent of the left ventricle between the end-systole and the end-diastole. [16] The latter constitutes a quantitative assessment of the systolic function of the left ventricle. The systolic function is the most commonly monitored cardiovascular function: its dysfunction can generally be interpreted as an early sign of several cardiac complications, such as the congestive heart failure (CHF). [17, 18]

Practically, the strain is computed by assessing the myocardial deformation. This is only possible if the motion of the myocardium is tracked properly through cardiac cycles. The literature references three main approaches to solve this task. The simplest one consists in estimating the displacement of the myocardium by integrating its velocity, obtained using tissue velocity imaging (TVI). TVI is an ultrasound imaging technique that uses the Doppler effect in order to measure 1-D tissue velocities along the ultrasound beam direction. Due to the uni-dimensional nature of TVI data, this technique provides meaningful results only if a certain degree of parallelism between the myocardium and the ultrasound beam is achieved. [19, 20] Another approach, called speckle tracking, identifies noticeable speckles in a B-mode image sequence and tracks them from one frame to the next. A similarity measure is optimized between consecutive frames to do so, which makes this procedure computationally heavier than the previous one. [19, 21] The last approach is called deformable image registration. This technique attempts to warp two consecutive frames on each other. The inverse transformation can then be used to perform landmark tracking on the myocardium from one frame to the next. [22] The study led by Heyde *et al.* [23] showed that deformable image registration and speckle tracking exhibit similar performances when compared to a gold standard reference measurement. Even though newly applied in the ultrasound imaging field, image registration is not a new concept. In 1972, Barnea and Silverman

[24] presented a class of sequential similarity detection algorithms (SSDAs) to solve the image registration problem more efficiently. In 1981, Lucas, Kanade *et al.* [25] developed an iterative image registration algorithm and applied it to a stereo vision problem with success.

The image registration problem can be solved by using an optical flow algorithm. Such an algorithm takes two frames of a video sequence, generally consecutive, and computes a sparse or dense vector field, called optical flow, describing the motion between those two frames. Optical flow can then be used to fold one frame on the other. In 1993, Black and Anandan [26] introduced a new framework for a robust estimation of optical flow. The presented method was able to identify image regions where brightness constancy and spatial smoothness assumptions are violated. Ten years later, Farnebäck [27] developed a robust optical flow estimation algorithm based on polynomials expansion. More recently, Dosovitskiy *et al.* [28] proved that optical flow estimation could be seen as a machine learning problem. They developed a convolutional neural network (CNN)-based architecture for optical flow estimation. Although presenting similar performances to state-of-the-art techniques, this new method performed generally faster. Two years later, an improved version of FlowNet, called FlowNet 2.0, was presented. A stacked architecture and a sub-network specialized in small motions estimation drastically improved the quality of the estimated flow. The scheduled manner the training data was presented also greatly enhanced the performances. [29] The same optical flow model was used along with the image inpainting model *EdgeConnect* [30] to develop an efficient flow-edge guided video completion algorithm. [31] In 2020, Teed and Deng [32] introduced a new deep architecture for iterative optical flow estimation. This model called Recurrent All-Pairs Field Transforms (RAFT) stacks a feature encoder, a correlation layer, and a GRU-based update operator. It achieves state-of-the-art performances while presenting a strong cross data set generalization. It also requires less iterations than other methods for a same optical flow quality. The authors won the best paper award at the European Conference on Computer Vision (ECCV) 2020. While being modern and well-adapted to the computer era, deep learning-based models for optical flow estimation suffer from the lack of realistic training data. Indeed, extracting ground-truth optical flow from a real scene with natural motion is a tedious work. For this reason, most data sets are synthetic. Among the most popular, the literature references the *MPI-Sintel* data set [33], derived for a 3D animated film, the *FlyingChairs* data set, introduced by Dosovitskiy *et al.* [28], and the *FlyingThings3D* data set [34], relatively similar to the previous. In 2013, a team from the Karlsruhe Institute of Technology (Germany) equipped a car with a set of sensors and generated ground-truth optical flows for different traffic scenarios. The so-called *KITTY* data set is, to the author's knowledge, the only reasonably large data set derived from real scenes. [35]

Performing strain estimation requires the expertise of a well-trained cardiologist.

This time consuming process is therefore poorly suited for real-time applications, like numerous of other medical imaging tasks. Nevertheless, the recent breakthrough of deep learning, or machine learning more generally, has disrupted the field of medical image processing. CNNs are particularly convenient to process images. In 2018, a CNN model outperformed most of the participating dermatologists on dermoscopic melanoma recognition task. [36] Kooi *et al.* [37] analyzed the performances of another CNN model on a mammographic lesions detection task. They showed that it outperformed a state-of-the-art system in computer aided detection (CAD), and performed just as good as a panel of three certified screening radiologists. In 2015, Knackstedt *et al.* [38] developed a fully automated software called AutoLV which was able to perform accurate and reproducible LVEF measurements and average biplane longitudinal strain. Another convolutional model, developed by Østvik *et al.* [39], estimated accurately global longitudinal strain from echocardiographic images. In 2019, Haukom *et al.* [22] presented a deep learning-based model for automatic regional strain estimation in TEE. For high-quality ultrasound images, this model exhibited satisfying results for basal strain estimation in 4- and 2-chamber views of the heart, in a time that allows for real-time applications. The model architecture was based an the work of Vos *et al.* [40, 41], who developed a deep learning framework for unsupervised affine and deformable image registration. Within this framework, models are trained to perform coarse-to-fine image registration without the need of labeled data. CNNs also showed promising performances in the segmentation task. In 2015, Ronneberger *et al.* [42] presented the now famous *U-Net*, a CNN architecture performing particularly well in biomedical image segmentation. A year later, Wiehman *et al.* [43] proposed an unsupervised pre-training procedure for CNNs that decreased the output variance of *U-Net* without affecting its mean performances. *U-Net* was later used by Smistad, Østvik *et al.* [44] to perform 2D left ventricle segmentation. Their goal was to analyze *U-Net* performances when trained by a Kalman filter automatic segmentation tool. Even if the model slightly outperformed the tool it was trained with, it still required training with annotated ultrasound images to achieve state-of-the-art performances.

Despite the recent progress made in the field, ultrasound image processing stays a challenging discipline due to the inherent noise of ultrasound acquisition techniques. Indeed, the speckle noise of commercial echographs alters the ultrasound image quality in regions of interest. Ultrasound image denoising is therefore a search field in its own right. In 2001, Zhang *et al.* [45] developed a novel method for Doppler ultrasound signal denoising using wavelet frame analysis. They showed that using wavelet frame analysis instead of wavelet transform analysis leads to better denoising performances. Several years later, Andria *et al.* [46] analyzed the performances of seven different wavelet coefficients linear filtering methods on ultrasound medical images. In 2011, De Fontes *et al.* [47] presented a modified version of the NL-means algorithm adapted for speckle noise reduction in ultrasound images. The proposed denoising method had very good performances but

was computationally heavy. Six years later, Singh *et al.* [48] presented a hybrid algorithm for speckle noise reduction. The proposed filter was composed of three simpler filters connected in cascade: a guided filter, a speckle reducing bilateral filter, and a rotation-invariant bilateral non-local means filter. This new hybrid algorithm outperformed all existing speckle denoising algorithms while keeping a reasonable complexity.

1.2 Aim of current work

Strain estimation from echocardiographic images is a time-consuming process affected by the inherent subjectivity of cardiologists. With the emergence of deep learning-based models, a full automation of this task can be considered. This thesis is an attempt to get closer to this objective. More particularly, it focuses on the automation of myocardial points tracking, the strain being simply deduced from the distance between two myocardial points at the end of systole and diastole.

Four optical flow methods are used to estimate the frame-to-frame motion of cardiac tissues:

1. the Daisy-chaining model, a coarse-to-fine image registration model strongly inspired from the work of Haukom *et al.* [22] and Vos *et al.* [40, 41],
2. the Recurrent All-Pairs Field Transforms (RAFT) method [32],
3. the Lucas-Kanade algorithm [25],
4. and the Gunnar-Farnebäck algorithm [27].

The feasibility of improving optical flow estimates by taking into account tissue velocity imaging (TVI) data is studied. Two tracking methods are developed in order to follow points of interest from one frame to the next. They both rely on the estimation of optical flow between successive frames. One of them implements a Kalman filter and is proposed as a solution to the point drifting issue. Myocardial segmentation is also experimented in order to semi-automatically extract interesting points to track. Finally, the ability to estimate accurately the strain in the basal segments of the myocardium is assessed for every combination of optical flow and tracking methods.

To sum up, the current work tries to answer the following questions:

- Can frame to frame tracking on TEE images be performed using optical flow-based motion estimation methods?
- Can TVI data be integrated with the tracking method to improve the quality of optical flow estimation?
- Can a Kalman filter-based tracking method reduce the drift problem?
- Can a myocardium segmentation tool be developed to extract interesting points to be used in the strain estimation process?
- Can point tracking on TEE images lead to an accurate estimation of the strain in basal segments of the myocardium when compared to expert measure-

ments?

1.3 Outline

A non-exhaustive literature review was made in this chapter. The goal pursued in this work was established. The second chapter provides all the theory needed for a deep understanding of the subject: the cardiac anatomy is reminded and the strain estimation process is detailed. Basics of B-mode and TVI are presented, as well as fundamentals of deep learning. The different optical flow algorithms are also presented. The third chapter is dedicated to the description of the data sets used and the methodology followed in this thesis. The fourth chapter presents the obtained results which are discussed in the fifth chapter. Finally, a conclusion is drawn in the sixth chapter.

Chapter 2

Theoretical background

2.1 Cardiac Anatomy

The heart is the organ responsible for blood circulation and acts as a pump. Its structure is depicted in Figure 2.1.

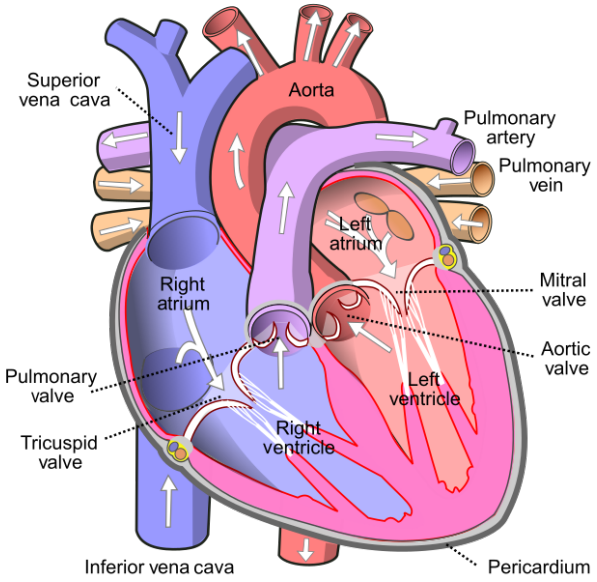


Figure 2.1: Schematic of the cardiac structure. White arrows indicate blood flow. Illustration by Wapcaplet - Own work, CC BY-SA 3.0, `https://commons.wikimedia.org/w/index.php?curid=830253`.

The heart has four chambers, two atria and two ventricles, and four valves to delimit them. The two atrioventricular valves, called mitral and tricupsid valves, separate the atrium from the ventricle in the left and right part of the heart respectively. The aortic valve defines the boundary between the left ventricle and the aorta, while the pulmonary valve is located at the junction between the right

ventricle and the pulmonary artery. The cardiac valves only allow blood flow in one direction: from an atrium to a ventricle or from a ventricle to an artery. The heart wall is made up of three layers: an inner layer called endocardium, a middle layer called myocardium, and an outer layer called epicardium. The myocardium is the thickest one (colored in pink in Figure 2.1) and consists of muscular fibers responsible for the cardiac contraction. A protective layer called the pericardium encompasses the entire organ. [15]

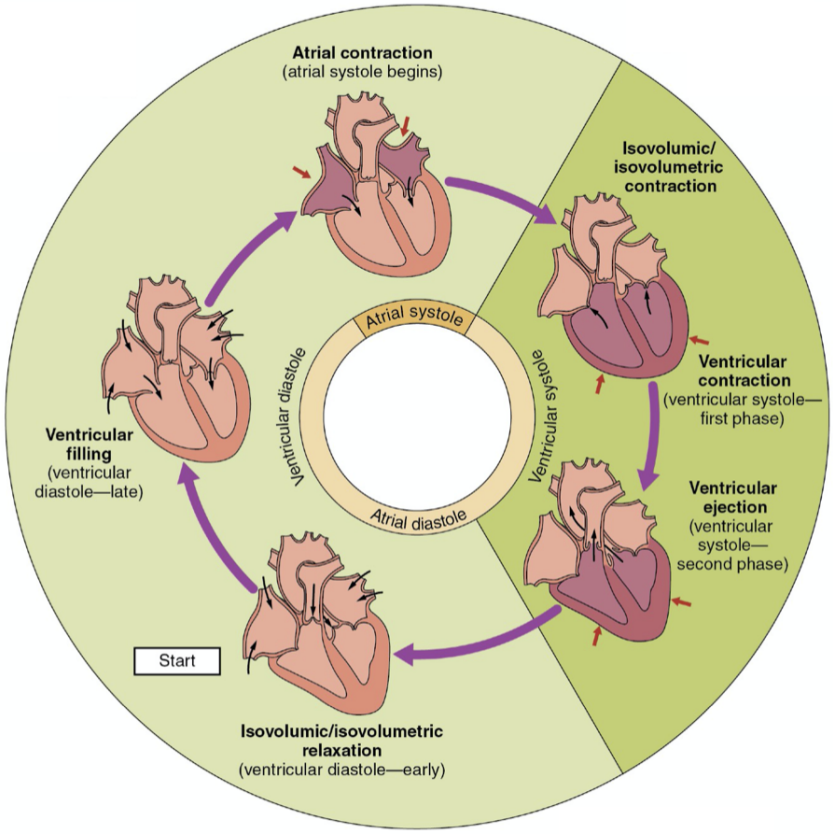


Figure 2.2: Stages of the cardiac cycle. Black arrows indicate blood flow. Illustration by OpenStax College - Anatomy & Physiology, Connexions Web site. `http://cnx.org/content/col11496/1.6/`, Jun 19, 2013., CC BY 3.0, `https://commons.wikimedia.org/w/index.php?curid=30148227`.

The time laps separating two heart beats corresponds to a repetition of the cardiac cycle. This cycle can be detailed stage by stage as followed:

1. **Isovolumic/isovolumetric relaxation** - The ventricles and atria are in their relaxation phase. The left atrium is filled with oxygenated blood coming from the pulmonary vein. The right atrium is filled with de-oxygenated blood coming from the inferior and superior vena cava.
2. **Ventricular filling** - The rise of pressure in both atria causes the mitral and

tricupsid valves to open. Both ventricles start to fill with blood.
3. **Atrial contraction** - Both atria contract and push the blood in the corresponding ventricle.
4. **Isovolumic/isovolumetric contraction** - Both ventricles start to contract, causing the mitral and tricupsid valves to close. During this phase, the ventricular pressure is not sufficient to cause the aortic and pulmonary valves to open.
5. **Ventricular contraction** - The ventricular pressure becomes sufficient to force the aortic and pulmonary valves to open.
6. **Ventricular ejection** - The ventricular contraction pushes the blood away from both ventricles to the aortic and pulmonary arteries. After this final stage, another cycle starts and stage 1 takes place again.

An illustration of the complete cycle is shown in Figure 2.2. The cycle can be sub-divided into two main phases: the systole and the diastole. The term systole refers to a contraction phase, while the term diastole refers to a relaxation phase. Note that those terms can either indicate the atrial or ventricular contraction and relaxation. In the following, the words systole and diastole designate the ventricular systole and diastole. [49]

Visualizing a Wiggers diagram can help improve the understanding of the cardiovascular physiology. It shows the temporal relationship that exists between the cardiac cycle and the electrocardiogram (ECG), the ventricular volume, and the blood pressure in the ventricles, atria and aorta. A typical Wiggers diagram is shown in Figure 2.3. The end-of-diastole (ED) and the end-of-systole (ES) are important cycle time instants that are generally used in the strain estimation process. In this thesis, the ECG and the position of the mitral valve are used to locate precisely those time instants. Figure 2.3 shows that the ES coincides with the end of the T-wave in the ECG signal and with the moment the mitral valve opens. As for the ED, it corresponds to the highest peak of the QRS-complex in the ECG signal and to the moment the mitral valve closes. [49, 50]

2.2 Echocardiography: B-mode and TVI images

Ultrasound imaging is the imaging modality that uses ultrasound waves to build images of tissue structures. More precisely, an ultrasound probe transmits focused beams of sound waves through body tissues. The level of sound reflected or scattered back to the transducer depends on the acoustic impedance along the path of the ultrasound beam. The amplitude and phase of echo waves can thus be analyzed to determine the structure of the tissues through which the beam traveled. Array of sound beams with different incident angles are used to build a 2- or 3-D image. Echocardiography is the ultrasound imaging branch that focuses on depicting cardiac structures. [51] Depending on the probe orientation, the heart is observed under different views. In this work, the 4-chambers, the 2-chambers and

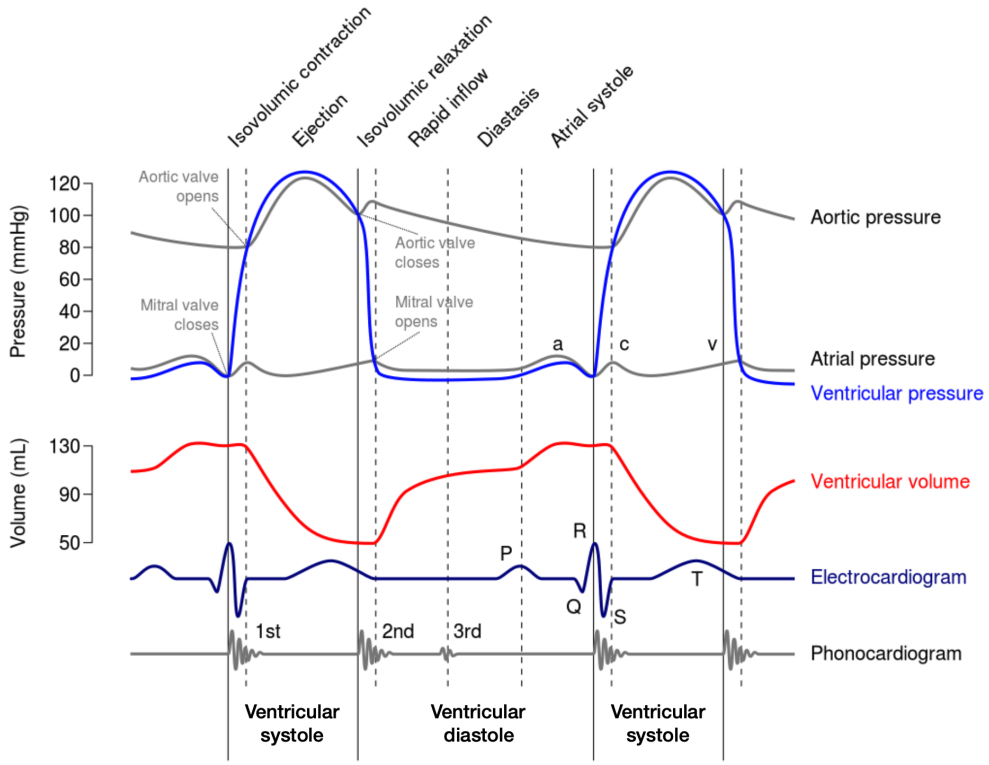


Figure 2.3: Typical Wiggers diagram. Illustration by DanielChangMD revised original work of DestinyQx; Redrawn as SVG by xavax - Wikimedia Commons, File:Wiggers Diagram.png., CC BY-SA 2.5, `https://commons.wikimedia.org/w/index.php?curid=18764854`.

the apical long-axis views of the heart are used. The two atria and two ventricles are visible in the 4-chambers view. The 2-chambers view only shows the left atrium and ventricle. The apical long-axis view shows the left atrium, the left ventricle and the aorta. A simplified diagram of these views is shown in Figure 2.6.

Two-dimensional brightness-mode (B-mode) is the most widespread ultrasound imaging mode. B-mode images are cross-sectional scans of tissue and organ boundaries. In B-mode imaging, vertical and horizontal dimensions of the image correspond to real spatial dimensions of the scanned tissue area. The brightness of a point in the image is directly proportional to the amplitude of the ultrasound echo coming from the corresponding point in the scanned tissue area. The location of tissue boundaries relative to the ultrasound probe are deduced from the time of arrival of echoes. Modern B-mode imaging systems still implement this simple principle, although they may use it within more complex arrangements for the sake of performance. [51, 52] A B-mode image representing the apical long-axis view of a human heart is shown in Figure 2.4a. Tissue velocity imaging (TVI), also called tissue Dopple imaging (TDI), is another well-known ultrasound imaging mode used

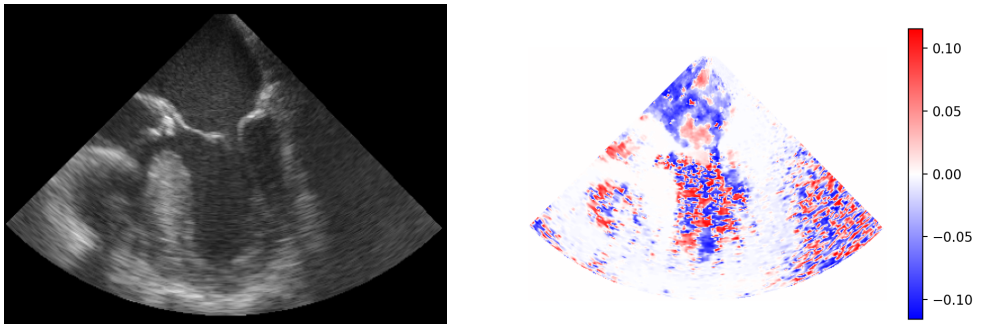
to measure blood and tissue velocities. In cardiology, it is commonly used to assess the performances of cardiac valves. This imaging mode makes use of the Doppler effect: the wave reflected on a surface in motion relatively to the the stationary ultrasound probe incurs a frequency shift Δf which is directly proportional to the surface velocity. The so-called Doppler shift is given by

$$\Delta f = \frac{-2 f_0 v_d \cos(\theta)}{v + v_d \cos(\theta)}, \tag{2.1}$$

where f_0 is the transmitted ultrasound frequency, v_d is the velocity of the moving surface, θ is the angle formed by the ultrasound beam and the surface velocity vector, and v is the speed of sound in the considered medium. Since v_d is generally much smaller than v, Equation 2.1 can be approximated by

$$\Delta f \approx \frac{-2 f_0 v_d \cos(\theta)}{v}. \tag{2.2}$$

The term $v_d \cos(\theta)$ is computed for each point of the TVI image. Note that this term represents the projection of vector $\vec{v_d}$ on the axis going from the ultrasound probe to the reflection point. Therefore, the Doppler mode computes the tissue velocity in the direction of the ultrasound beam only, the component of $\vec{v_d}$ perpendicular to this direction remaining unknown. [51–53] A TVI image of the apical long-axis view of a human heart is shown in Figure 2.4b.



(a) B-mode image **(b)** TVI image

Figure 2.4: B-mode and TVI images of the apical long-axis view of a human heart. Pixel color in the TVI image indicates the velocity of the considered pixel in the direction of the ultrasound beam.

When it comes to imaging the heart, the ultrasound probe can be placed in several ways. In transthoracic echocardiography (TTE), the probe is placed on the skin in an area close to the third and fourth left intercostal space. The ease of setup makes it the most popular probe placement. In transesophageal echocardiography (TEE), the ultrasound probe is placed in the patient's esophagus, as shown in Figure 2.5. Setting up a TEE probe is more complicated and can lead to complications in

rare cases [11]. Nevertheless, TEE benefits from an improved visualization power compared to TTE: the quality of TTE images can sometimes be affected by excess body tissues, scarring, ribs or collapsed lungs. Moreover, TEE probes do not require a continual support and adjustment, unlike TTE probes. All B-mode and TVI image sequences processed in this work were acquired using a TEE probe.

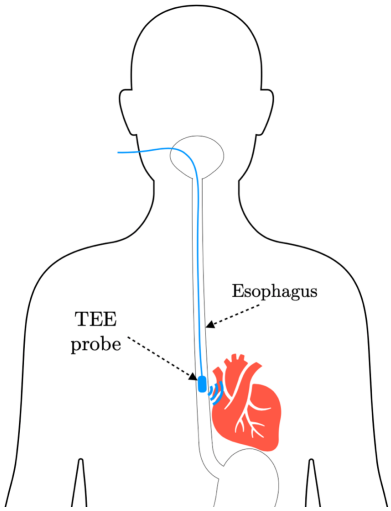


Figure 2.5: Simplified diagram showing the TEE probe placement in the human body.

2.3 Strain estimation

The ventricular cardiac function can be assessed by measuring the strain resulting from the local contraction of the myocardium. A local shortening, thickening or elongation of the myocardium is globally defined as myocardial strain. A measure of the shortening along the x-, y- and z-axis, and of the shear in the xy, xz and yz planes gives a full description of the myocardial strain. When strain is estimated from B-mode images, assessing those six components becomes a tedious task. A simplified metric called the Lagrange strain is thus preferred. It consists in measuring the shortening incurred by a specific myocardial region. [54] Given two material points located at both ends of the considered region, the 1D strain $\epsilon(t)$ is given by

$$\epsilon(t) = \frac{L(t) - L_0}{L_0}, \tag{2.3}$$

where $L(t)$ is the distance between the two material points at time t, and $L_0 = L(t_0)$, t_0 being the reference time. In this thesis, the end-systolic strain is used to assess the myocardial contractility. The end-systolic strain is computed by setting $t_0 = \text{ED}$ and $t = \text{ES}$ in Equation 2.3. [22, 54]

In 2002, the American Heart Association published a standardized segmentation model dividing the myocardium into seventeen distinct segments. [55] This model is illustrated in Figure 2.6 for the 4-chamber, 2-chamber and apical long-axis views of the heart. For each myocardial segment, the strain can be assessed in three different directions: longitudinal, radial and circumferential. The longitudinal strain assesses the myocardial deformation along the atrioventricular axis[1] direction while the radial strain assesses the myocardial deformation along the perpendicular direction. The circumferential strain assesses the circumferential shortening of the myocardium around the atrioventricular axis. The quality of TEE is generally higher close to the cardiac valves and decreases as getting closer to the apex segment. For this reason, this work focuses on basal longitudinal strain estimation only (i.e. longitudinal strain estimation in the basal segments visible in Figure 2.6).

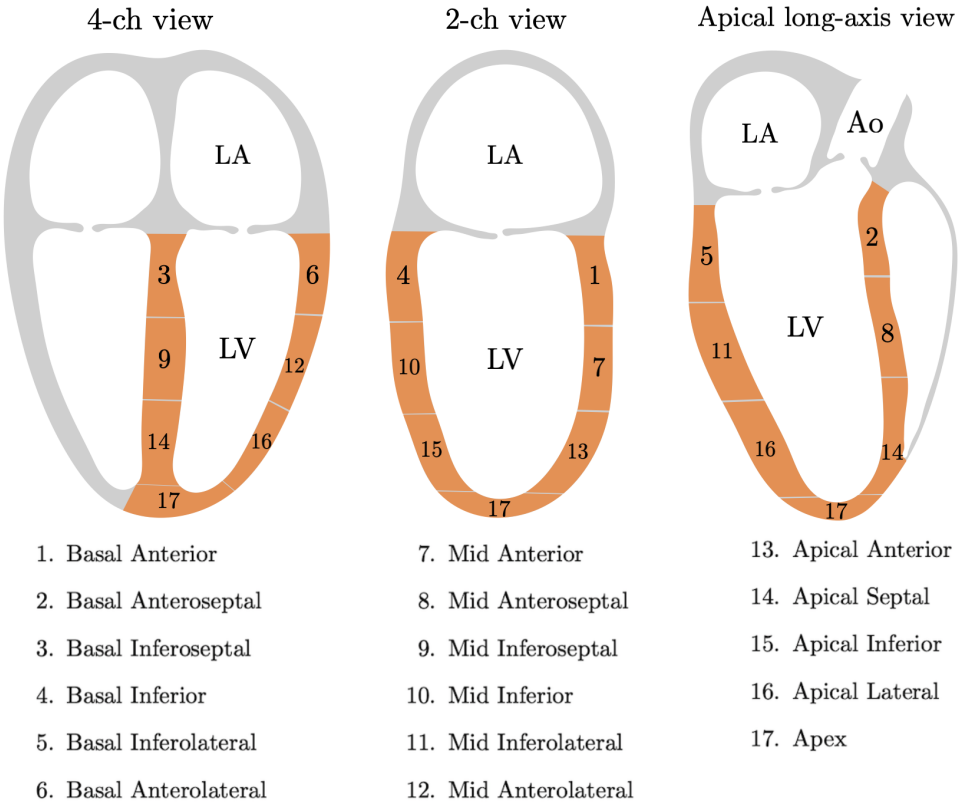


Figure 2.6: 17-segments model of the myocardium proposed by the American Heart Association. [55] LA, LV and Ao respectively stand for Left Atrium, Left Ventricle and Aorta. The heart is observed under the 4-chambers, 2-chambers and long axis views. In this thesis, strain is estimated in basal segments only.

[1] i.e. the axis passing by the mitral annulus and perpendicular to the plane separating the left atrium from the left ventricle.

2.4 Deep learning fundamentals

Machine learning is a relatively new area of research that studies the ability of computers to learn how to execute a specific task without being explicitly programmed for it. The branch of machine learning that focuses on the development of complex models composed of several simpler non-linear estimators is called deep learning. Deep learning models are so called due to the depth of their forward path[2]. The architecture of deep learning models was originally inspired from neuronal interconnections and was designed to mimic the functioning of the human brain. Nowadays, there exist several task-specific architectures such as the convolutional neural networks (CNNs), specific to vision, detection and segmentation tasks, the recurrent neural networks (RNNs), specific to temporal sequences analysis, and many others. [56] Architectures relevant to this work will be presented in this section. In the following, all vectors and tensors are displayed in bold.

2.4.1 Feed-forward neural networks

Feed-forward neural networks are the simplest type of deep learning networks. Their building block is called neuron or logistic unit. Mathematically, a neuron is the generally non-linear function $h : \mathbb{R}^p \mapsto \mathbb{R} : \boldsymbol{x} \to h(\boldsymbol{x})$ described as

$$h(\boldsymbol{x}) = \phi(\boldsymbol{w}^T \boldsymbol{x} + b), \tag{2.4}$$

where $\boldsymbol{w} \in \mathbb{R}^p$ is a weight vector, $\boldsymbol{b} \in \mathbb{R}$ is a bias, and $\phi(y) : \mathbb{R} \to \mathbb{R} : y \to \phi(y)$ is a non-linear function called activation function. The sigmoid, rectified linear unit (ReLU), threshold function and hyperbolic tangent (tanh) are commonly used activation functions. [56, 57]

Several neurons can be stacked and used in parallel to form a layer. Formally, with the input $\boldsymbol{x} \in \mathbb{R}^p$, a layer $\boldsymbol{h}(\boldsymbol{x}) : \mathbb{R}^p \to \mathbb{R}^q : \boldsymbol{x} \to \boldsymbol{h}(\boldsymbol{x})$ is defined as

$$\boldsymbol{h}(\boldsymbol{x}) = \phi(\boldsymbol{W}^T \boldsymbol{x} + \boldsymbol{b}), \tag{2.5}$$

where $\boldsymbol{W} \in \mathbb{R}^{p \times q}$ is the weight matrix of the layer, $\boldsymbol{b} \in \mathbb{R}^q$ is the vector of bias of the layer, and where the activation function ϕ is applied element-wise. Such a model is called single-layer perceptron and is the simplest example of artificial neural network (ANN). [56, 57]

The representation capacity of the single-layer perceptron, i.e. its capability to model multi-dimensional mappings, is very limited. On a classification task, it is only able to learn linearly separable patterns. A more complex model with a better representation capacity can be built by stacking several layers of logistic units. Indeed, deeper models theoretically lead to improved representation abilities. The same statement stands for models whose layers are composed of more neurons.

[2]The forward path is the process path that links the output of the model to its input.

[56–58] This stacked model is called multi-layer perceptron (MLP) and is described as followed:

$$\begin{aligned} &\boldsymbol{h_0} = \boldsymbol{x}, \\ &\boldsymbol{h_1} = \phi(\boldsymbol{W_1}^T \boldsymbol{h_0} + \boldsymbol{b_1}), \\ &\cdots \\ &\boldsymbol{h_L} = \phi(\boldsymbol{W_L}^T \boldsymbol{h_{L-1}} + \boldsymbol{b_L}), \end{aligned} \tag{2.6}$$

where $\boldsymbol{W_i} \in \mathbb{R}^{p \times q}$ and $\boldsymbol{b_i} \in \mathbb{R}^q$ are respectively the weight matrix and bias vector of layer $\boldsymbol{h_i}$. The first and last layers (i.e. $\boldsymbol{h_1}$ and $\boldsymbol{h_L}$) of the MLP are respectively called input and output layers. Inner layers $\boldsymbol{h_l}$ with $l \in \{2, 3, \cdots, L-1\}$ are called hidden layers. A diagram of a simple MLP is shown in Figure 2.7. The MLP model is called fully-connected owing to the complete interconnection of neurons of two adjacent layers.

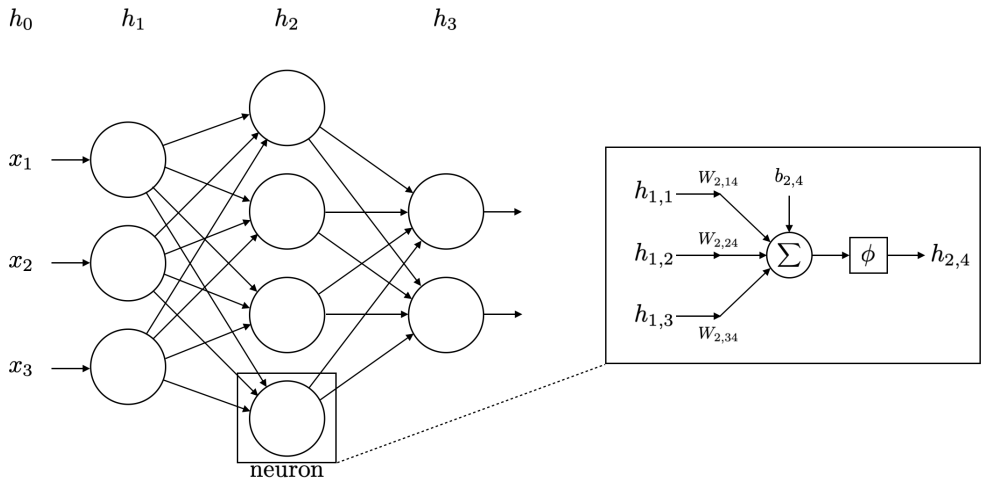


Figure 2.7: Multi-layer perceptron architecture with three input features, a single hidden layer with four neurons, and two outputs. Note that the subscription used here is such that $h_{i,j} = [\boldsymbol{h_i}]_j$, $b_{i,j} = [\boldsymbol{b_i}]_j$ and $W_{i,jk} = [\boldsymbol{W_i}]_{jk}$.

2.4.2 Convolutional neural networks

Convolutional neural networks (CNNs) were introduced to overcome shortcomings of fully-connected models on vision tasks. Indeed, the number of parameters of fully-connected models increases drastically with the number of input features to process. This quickly leads to models of intractable size when inputs with large dimensions like images need to be handled. Moreover, treating vision signals requires models to have specific properties, like invariance to translation, locality and hierarchical compositionality, which fully-connected models do not have. [57] In a convolutional layer, each neuron is only connected to a subset of neurons from the previous layer, called the receptive field. The same non-linear transformation is

applied to the receptive field of each neuron in the layer. More rigorously, consider an input feature map of dimension three, i.e. a 3-D tensor $\boldsymbol{x} \in \mathbb{R}^{C\times H\times W}$. This could be a RGB image, in which case $C = 3$, H and W would be respectively the number of channels, the height and the width of the image. The 3-D convolution operation is defined by the kernel $\boldsymbol{u} \in \mathbb{R}^{C\times h\times w}$, generally with $h \ll H$ and $w \ll W$. It produces a 2-D output feature map $\boldsymbol{o} \in \mathbb{R}^{(H-h+1)\times(W-w+1)}$ whose element $\boldsymbol{O}_{ji}$ is given by

$$\boldsymbol{o}_{ji} = \boldsymbol{b}_{ji} + \sum_{c=0}^{C-1}\sum_{n=0}^{h-1}\sum_{m=0}^{w-1} \boldsymbol{x}_{c,n+j,m+i} \cdot \boldsymbol{u}_{c,n,m}, \tag{2.7}$$

where $\boldsymbol{b}_{ji}$ is the bias. Since a same kernel is slided along the entire input feature map, the convolution operation is equivariant[3] to translation. D convolutions can be applied to the same input feature map (with different convolution kernels) in order to create a 3-D output feature map of dimension $D\times(H-h+1)\times(W-w+1)$. [57] Note that the convolution operation has three additional parameters:

- **Stride** - The step size used to translate the kernel across the input feature map.
- **Padding** - Specifies how many columns and rows of zero need to be added around the input feature map before the operation.
- **Dilation** - Modulates the expansion of the kernel support by adding rows and columns of zeros between the kernel coefficients. Dilating the kernel increase the size of the receptive field of a neuron without increasing the number of its parameters.

An illustration of the convolution operation is shown in Figure 2.8.

Another layer commonly used in CNNs is the pooling layer. Its role is to decrease the input tensor dimensions while preserving its global structure and the important features it contains as much as possible. Consider a 3-D input tensor $\boldsymbol{x} \in \mathbb{R}^{C\times rh\times sw}$ and a pooling area of size $h \times w$. There exist two pooling operations:

- **Average pooling** - Produces the output tensor $\boldsymbol{o} \in \mathbb{R}^{C\times r\times s}$ such that

$$\boldsymbol{o}_{c,j,i} = \frac{1}{hw}\sum_{n=0}^{h-1}\sum_{m=0}^{w-1} \boldsymbol{x}_{c,rj+n,si+m}. \tag{2.8}$$

- **Max pooling** - Produces the output tensor $\boldsymbol{o} \in \mathbb{R}^{C\times r\times s}$ such that

$$\boldsymbol{o}_{c,j,i} = \max_{n<h,m<w} \boldsymbol{x}_{c,rj+n,si+m}. \tag{2.9}$$

An illustration of the max pooling operation is shown in Figure 2.9. Note that this figure illustrates the max pooling operation on a 2-D input tensor. The generalization to 3-D input tensors is trivial and let to the reader. Pooling operations are invariant[4] to any permutation occurring within a pooling cell. This is particularly

[3] A function f is equivariant to the function g if $f(g(x)) = g(f(x))$.
[4] A function f is invariant to a function g if $f(g(x)) = f(x)$.

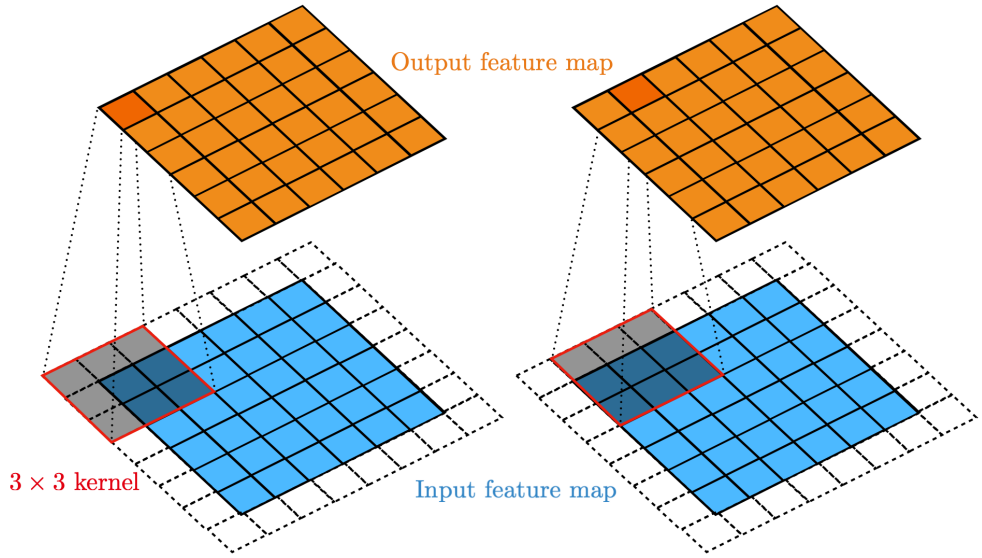


Figure 2.8: Illustration of a convolution operation with a 3 × 3 kernel. Padding and stride parameters are both set to 1. No kernel dilation is applied.

useful if the detection of a pattern matters more than finding its exact location. [57]

CNNs are generally built as an arbitrary composition of convolutional layers,

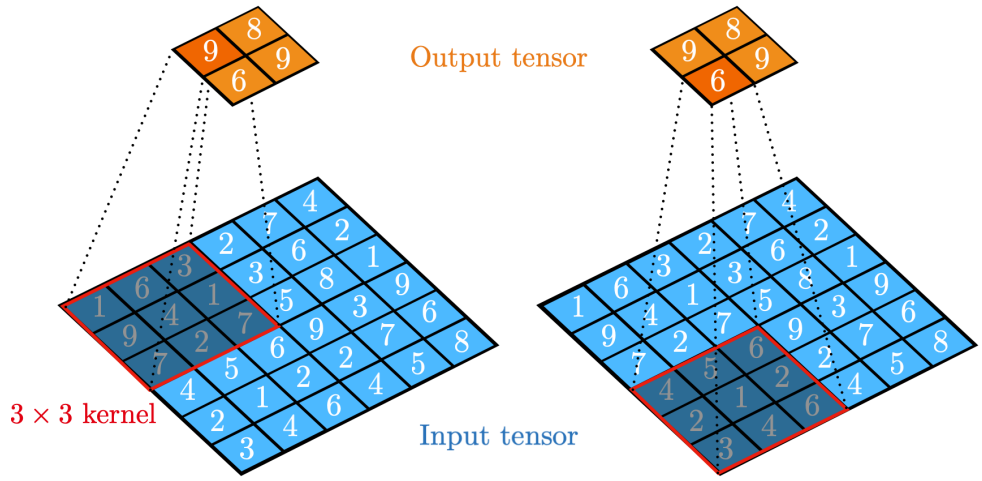


Figure 2.9: Illustration of the max pooling operation with a 3 × 3 kernel. The red frame defines a pooling cell in the input feature map. Each element of the output tensor is the element with the highest value in the corresponding pooling cell.

pooling layers, linear rectifiers with ReLU activation functions, and fully-connected layers. Such networks are able to learn a hierarchical composition of complex patterns, a particularly convenient property for vision signals processing. [56, 57]

2.4.3 Training neural networks

Deep learning models, and even more generally machine learning models, need to be trained before being able to perform a given task. Mathematically, every task underlies an implicit mapping $f : \mathcal{X} \to \mathcal{Y} : \boldsymbol{x} \to \boldsymbol{f(x)}$ where $\mathcal{X}$ and $\mathcal{Y}$ are respectively the input and output spaces. Training consists in finding the best estimate $\hat{f}$ of f. Three learning paradigms can be used to this end. They are called supervised, unsupervised and reinforcement learning. [56] In the following, Θ is the space of the model parameters, and the function $\hat{f}(\cdot\,;\boldsymbol{\theta}) : \mathcal{X} \mapsto \mathcal{Y} : \boldsymbol{x} \mapsto \boldsymbol{\hat{f}(x;\theta)}$ refers to the model with parameters value $\boldsymbol{\theta} \in \Theta$.

Learning paradigms

Supervised learning consists in learning from examples. In this setting, a training set $\mathcal{X}_{tr} \subset \mathcal{X}$ and the set of corresponding labels (also called ground truths) $\mathcal{Y}_{tr} = \{\boldsymbol{f(x)} \mid \boldsymbol{x} \in \mathcal{X}_{tr}\}$ are accessible to the model. Training consists in adjusting the model parameters $\boldsymbol{\theta}$ so that $\hat{f}(\cdot\,;\boldsymbol{\theta})$ estimates at best the actual mapping f. Practically, a loss metric between the predictions $\boldsymbol{\hat{y}} = \boldsymbol{\hat{f}(x)}$ ($\boldsymbol{x} \in \mathcal{X}_{tr}$) and the true labels $\boldsymbol{y} = \boldsymbol{f(x)}$ is defined. Then, an iterative optimization algorithm is used to update $\boldsymbol{\theta}$ and minimize this loss metric. This paradigm is intuitive. However, it often requires a relatively large training set for the model's predictions to apply to previously unseen data. [56, 57, 59] Annotating a large amount of data is often extremely time consuming, even sometimes not feasible, and thus constitutes a challenge in itself.

Unsupervised learning covers any learning algorithm that draws inferences from a training set $\mathcal{X}_{tr} \subset \mathcal{X}$ without any knowledge about the corresponding labels. This framework is particularly convenient since it does not require any annotated data. However, it is not suitable for all tasks. Clustering algorithms are the best example of unsupervised learning: they find hidden patterns or groups in the input data with any other knowledge than the data itself. [56, 60] The k-means algorithm is the most famous clustering algorithm. [61] The deformable image registration framework developed by Vos *et al.* [40, 41] also uses this paradigm.

The last learning paradigm called reinforcement learning consists in learning by experiencing. In this framework, the model is an intelligent agent evolving in an environment. The agent encounters various situations (i.e. the model receives inputs) and makes decisions (i.e. produces outputs) based on the situations and its knowledge of the environment. A feedback is provided for every decision: the agent is rewarded in case of smart decision, and punished otherwise. The agent's goal is to maximize its total reward. It uses the feedback to update its decision policy and improve its knowledge of the environment. In this framework, the agent makes decisions while continuously adapting to its environment. For this reason, reinforcement learning is widely used for artificial intelligence training. [62] Omron Global developed a robot called Forpheus using this paradigm: this

tennis-table robot is designed to continuously improve its technique while playing against real people. [63]

Model parameters optimization

In most cases, reinforcement learning problems are solved using the Q-learning algorithm developed by Watkins and Dayan [64]. Reinforcement learning techniques are out of the scope of this work and will not be detailed further.

In both supervised and unsupervised settings, the model parameters are updated by minimizing a loss function over the training set $\mathcal{X}_{tr}$. Formally, let define the loss function $J : \Theta \mapsto \mathbb{R} : \boldsymbol{\theta} \mapsto J(\boldsymbol{\theta})$ as

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{X}_{tr}} \left\{ L\left(\boldsymbol{f}(\boldsymbol{x}), \hat{\boldsymbol{f}}(\boldsymbol{x}; \boldsymbol{\theta})\right) \right\}, \tag{2.10}$$

where $\boldsymbol{x} \in \mathcal{X}_{tr}$, $L(\cdot, \cdot)$ is an arbitrary loss metric[5], and $\mathbb{E}_{\mathcal{X}_{tr}}\{\cdot\}$ is the expectation operator over the training set. Model parameters are generally updated according to a gradient descent method. If subscript $t = 0, 1, 2, \ldots$ denotes the current iteration of the algorithm, then

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \gamma \cdot \boldsymbol{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_t) \\ &= \boldsymbol{\theta}_t - \gamma \cdot \frac{1}{N} \sum_{n=0}^{N-1} \boldsymbol{\nabla}_{\boldsymbol{\theta}} L(\boldsymbol{f}(\boldsymbol{x_n}), \hat{\boldsymbol{f}}(\boldsymbol{x_n}; \boldsymbol{\theta}_t)), \;\; \boldsymbol{x_n} \in \mathcal{X}_{tr}, \end{aligned} \tag{2.11}$$

where $\boldsymbol{\nabla}_{\boldsymbol{\theta}}$ is the gradient operator with respect to vector $\boldsymbol{\theta}$, $N = |\mathcal{X}_{tr}|$, and γ is a training hyper-parameter called learning rate. Computing the gradient over the entire training set is a costly operation. For this reason, the mini-batch stochastic gradient descent (SGD) algorithm is generally preferred. This algorithm uses only a subset $\mathcal{B}_t \subset \mathcal{X}_{tr}$ to update $\boldsymbol{\theta}_t$. The batch $\mathcal{B}_t$ is built by randomly picking B elements from the set $\mathcal{X}_{\text{batches}} = \left\{ \boldsymbol{x} \in \mathcal{X}_{tr} \mid \boldsymbol{x} \notin \mathcal{B}_i, \left\lceil \frac{N}{B} \right\rceil \cdot \left\lfloor \frac{B}{N} t \right\rfloor \leq i < t \right\}$. Equation 2.11 thus becomes

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \gamma \cdot \frac{1}{B} \sum_{n=0}^{B-1} \boldsymbol{\nabla}_{\boldsymbol{\theta}} L(\boldsymbol{f}(\boldsymbol{x_n}), \hat{\boldsymbol{f}}(\boldsymbol{x_n}; \boldsymbol{\theta}_t)), \;\; \boldsymbol{x_n} \in \mathcal{B}_t. \tag{2.12}$$

The time period needed by the algorithm to go through the whole training set is called an epoch. Notice that $\mathcal{X}_{\text{batches}} = \mathcal{X}_{tr}$ at the beginning of each epoch. [57]

The direction of the resulting gradient computed in Equation 2.12 has a high variance, especially if the batch size is small. In order to dampen large changes in

[5] Note that the loss metric differs if the supervised or unsupervised training framework is used. In the supervised case, the loss metric $L : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R} : (\boldsymbol{y}, \hat{\boldsymbol{y}}) \mapsto L(\boldsymbol{y}, \hat{\boldsymbol{y}})$ measures the error between a prediction $\hat{\boldsymbol{y}} = \hat{\boldsymbol{f}}(\boldsymbol{x}; \boldsymbol{\theta})$ and the ground truth $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{X}_{tr}$. In the unsupervised case, the loss metric $L : \Xi \times \mathcal{Y} \mapsto \mathbb{R} : (\boldsymbol{\xi}, \hat{\boldsymbol{y}}) \mapsto L(\boldsymbol{\xi}, \hat{\boldsymbol{y}})$ assigns a score to a model prediction $\hat{\boldsymbol{y}} = \hat{\boldsymbol{f}}(\boldsymbol{x}; \boldsymbol{\theta})$, $\boldsymbol{x} \in \mathcal{X}_{tr}$, based on another variable $\boldsymbol{\xi} \in \Xi$. In the framework developed by Vos *et al.* [40, 41], the metric is a dissimilarity measure between the original image and its warped version. Equation 2.10 refers to the supervised case.

gradient direction from one training iteration to the next, some optimizers add inertia to the updating process. This mechanism is called momentum. Another common mechanism called adaptive learning rate consists in adapting γ to each individual component of $\nabla_{\theta} J(\theta_t)$. Indeed, the curvature of the parameter space Θ is anisotropic in a lot of cases, and a smaller learning rate should be used in more curved directions. The Adam optimizer implements those two mechanisms. Its update rule is given by

$$\begin{aligned} s_t &= \rho_1 s_{t-1} + (1-\rho_1) g_t \\ \hat{s}_t &= \frac{s_t}{1-\rho_1^t} \\ r_t &= \rho_2 r_{t-1} + (1-\rho_2) g_t \cdot g_t \\ \hat{r}_t &= \frac{r_t}{1-\rho_2^t} \\ \theta_{t+1} &= \theta_t - \gamma \cdot \frac{\hat{s}_t}{\epsilon + \sqrt{\hat{r}_t}}, \end{aligned} \tag{2.13}$$

where all operations involving vectors are performed element-wise, ρ_1 and ρ_2 are hyper-parameters usually set to 0.9 and 0.999 respectively, ϵ is a small constant to avoid division by zero, and where $g_t = \frac{1}{B} \sum_{n=0}^{B-1} \nabla_{\theta} L(f(x_n), \hat{f}(x_n; \theta_t))$. Adam is nowadays the default optimizer of most training algorithms. [57, 65]

The value the learning rate γ is crucial: a too high learning rate could prevent the optimization algorithm from converging while a too small learning rate would decrease the convergence rate and increase the chances to find a sub-optimal solution to the problem. The value of the learning rate is generally decreased as the training progresses. [56, 57]

Each iteration of the updating process of θ requires an important number of gradients to be computed. The back-propagation algorithm is generally used for this purpose. In a very time-efficient way, it back-propagates the gradients computation from the last to the first layer of the model architecture. This dynamic programming algorithm uses the chain rule to compute the gradient of the loss metric with respect to each parameter of a layer, based on the gradients of the loss metric with respect to the parameters of the next layer. [66]

Since the parameter space Θ is generally non-convex, the convergence of the training procedure is not guaranteed. However, there are some rules of thumb to prevent the optimization algorithm from diverging. One of the most important ones concerns the model weights initialization. Very little is known about how to properly initialize the weights, except that it should break symmetry and that the scale of the weights matters. A good strategy is to control the variance in the forward and backward paths. Indeed, keeping the variance of the activations (i.e.

the outputs of the activation functions) constant in the forward pass ensures that information keeps flowing through layers without having its magnitude reduced or magnified. In the same idea, maintaining the variance of the gradients with respect to the activations through the backward pass prevents the gradients magnitude from vanishing or exploding. Note that maintaining the variances in the forward and backward paths leads to two contradicting constraints. The best compromise consists in initializing randomly the model weights $\boldsymbol{W}_l$ from a distribution with variance

$$\mathbb{V}\{\boldsymbol{W}_l\} = \frac{2}{q_{l-1} + q_l}, \tag{2.14}$$

where $\mathbb{V}\{\cdot\}$ is the variance operator, and q_{l-1} and q_l are respectively the number of neurons in layers $l-1$ and l. This initialization strategy is called Xavier or Glorot initialization. [67] It is nowadays the default initialization strategy in most of deep learning programming libraries.

Splitting training data

The loss function $J(\boldsymbol{\theta})$ is called empirical risk and indicates how poorly the estimated mapping $\hat{f}(\cdot\,;\boldsymbol{\theta})$ performs on the training set $\mathcal{X}_{tr}$. The training process minimizes the empirical risk. However, the model should have good generalization properties and be able to predict accurately the output for all possible inputs $\boldsymbol{x} \in \mathcal{X}$, and not only for $\boldsymbol{x} \in \mathcal{X}_{tr}$. Training should then minimize the expected risk

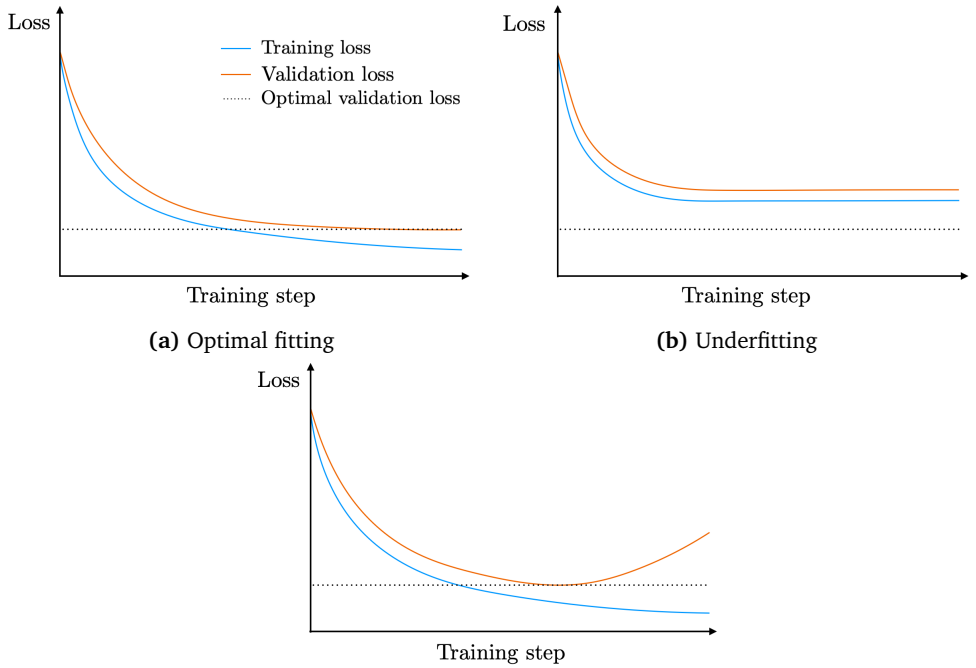
$$R(\boldsymbol{\theta}) = \mathbb{E}_{\mathcal{X}}\left\{L\left(\boldsymbol{f}(\boldsymbol{x}), \hat{\boldsymbol{f}}(\boldsymbol{x};\boldsymbol{\theta})\right)\right\} \tag{2.15}$$

instead of the empirical risk $J(\boldsymbol{\theta})$. This is not feasible since only $\mathcal{X}_{tr} \subset \mathcal{X}$ is accessible during training. Accordingly, in order to estimate properly the expected risk during training, the available training data is split into three disjointed sets:

- The actual **training set** - Gathers all the data the model is actually trained on.
- The **validation set** - Acts as an unseen data set, used to periodically assess the prediction abilities of $\hat{f}(\cdot\,;\boldsymbol{\theta})$ during training. The validation data set does not take part in the model parameters updating process.
- The **test set** - Used to assess the prediction performances of the final model on unseen data, when the training phase is over.

When the training phase starts, the training and validation losses[6] both decrease. The training phase should end when the validation loss reaches its minimal value. Beyond this point, the validation loss rises even if the training loss still decreases. The model starts overfitting the training data: it becomes too specific to the training set and its generalization abilities drop. In some other cases, the architecture of the model is too simple and $\hat{f}(\cdot\,;\boldsymbol{\theta})$ is a poor estimate of f, $\forall \boldsymbol{\theta} \in \Theta$. The model fails to reach the optimal validation loss and is said to underfit. Those different scenarios are illustrated in Figure 2.10.

[6]The training and validation losses respectively refer to the average loss on the training and validation sets.



(a) Optimal fitting

(b) Underfitting

(c) Overfitting

Figure 2.10: Illustration of the optimal fitting, underfitting and overfitting cases.

2.5 Optical flow models

In this section, an unsupervised training framework for optical flow models is described and several optical flow models are presented. These optical flow algorithms are at the heart of the models developed in this thesis. They take two consecutive video frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1} \in \mathbb{R}^{H\times W\times C}$ as inputs and output a dense displacement field $\boldsymbol{D} \in \mathbb{R}^{H\times W\times 2}$ with one motion vector per pixel. H and W denote the height and width of the input images and C is their number of channels (RGB images have $C = 3$ channels, while $C = 1$ for grayscale images). The third dimension of $\boldsymbol{D}$ refers to the horizontal and vertical components of the motion vector of image pixels. Mathematically, it comes

$$\boldsymbol{I}_i(x, y, c) \approx \boldsymbol{I}_{i+1}\left(x + \boldsymbol{D}(x, y, 1),\ y + \boldsymbol{D}(x, y, 0),\ c\right), \tag{2.16}$$

where the input images and the corresponding displacement field are represented as 3-D tensors, and where the notation $\boldsymbol{A}(x, y, z)$ is equivalent to $[\boldsymbol{A}]_{xyz}$, $\boldsymbol{A} \in \mathbb{R}^{l\times m\times n}$.

2.5.1 Unsupervised framework for optical flow methods

As mentioned in chapter 1, Vos *et al.* [40, 41] developed an unsupervised training framework for deformable image registration, later adapted by Haukom *et al.* [22]. The adapted version can be used to train any CNN-based optical flow model in an

unsupervised way. The training procedure is simple. Firstly, two consecutive video frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$ are given as inputs to the considered optical flow model. Secondly, $\boldsymbol{I}_{i+1}$ is warped accordingly to the computed flow[7]. If this output displacement field is accurate enough, the warped image $\boldsymbol{I}^w_{i+1}$ is such that

$$\boldsymbol{I}^w_{i+1} \approx \boldsymbol{I}_i. \tag{2.17}$$

Finally, a dissimilarity measure between $\boldsymbol{I}_i$ and $\boldsymbol{I}^w_{i+1}$ is minimized and the parameters of the optical flow model are updated according to a gradient-descent approach. A bending penalty is introduced in the optimization process. This tunable penalty ensures a certain spatial smoothness of the optical flow. A diagram describing the unsupervised training process is shown in Figure 2.11.

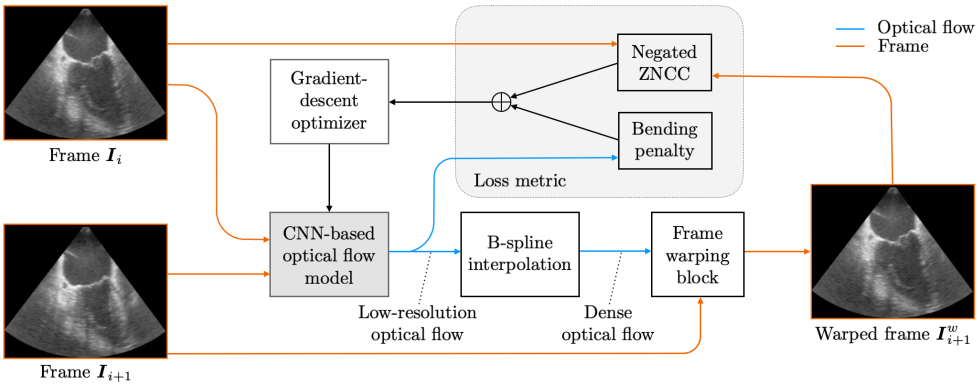


Figure 2.11: Diagram of the unsupervised learning framework developed by Vos *et al.* [40, 41] and adapted by Haukom *et al.* [22]. The dissimilarity measure used is the negated zero-mean normalized cross-correlation (negated ZNCC).

A good dissimilarity measure is the negated zero-mean normalized cross-correlation (negated ZNCC). This metric is computed as

$$-\frac{1}{HW}\cdot\frac{\sum_{x=0}^{H-1}\sum_{y=0}^{W-1}(\boldsymbol{I}_i(x,y)-\mathbb{E}\{\boldsymbol{I}_i\})(\boldsymbol{I}^w_{i+1}(x,y)-\mathbb{E}\{\boldsymbol{I}^w_{i+1}\})}{\sqrt{\mathbb{V}\{\boldsymbol{I}_i\}\cdot\mathbb{V}\{\boldsymbol{I}^w_{i+1}\}}}, \tag{2.18}$$

where $\mathbb{E}\{\boldsymbol{I}_i\}$ and $\mathbb{V}\{\boldsymbol{I}_i\}$ are respectively the mean and variance of the pixel values of image $\boldsymbol{I}_i$. Note that $\boldsymbol{I}_i$ and $\boldsymbol{I}^w_{i+1}$ are turned into grayscale images beforehand. A negated ZNCC value of -1 indicates a perfect similarity between the two images, while a value of 0 indicates no correlation. The bending penalty P ensures that the second order spatial derivatives of the computed optical flow $\boldsymbol{D}$ are minimized.

[7]Note that cubic B-spline interpolation is used beforehand to match the input frames and output flow spatial resolutions if the model computes a low-resolution displacement field.

It forces the warping transformation to be globally smooth. [68] Its expression is given by

$$P(\boldsymbol{D}) = \beta \sum_{x=0}^{H-1} \sum_{y=0}^{W-1} \left(\frac{\partial^2 \boldsymbol{D}}{\partial x^2} \right)^2 + \left(\frac{\partial^2 \boldsymbol{D}}{\partial y^2} \right)^2 + 2 \left(\frac{\partial^2 \boldsymbol{D}}{\partial x y} \right)^2, \tag{2.19}$$

where β is the scaling factor that controls the amount of regularization.

2.5.2 Daisy-chaining model

In 2019, Haukom *et al.* [22] developed a technique to automatically estimate regional strain from TEE images. This method relies on deep learning optical flow models. The Daisy-chaining model exhibited the best performances among the three presented models. It is made of a succession of three simpler optical flow CNN units separated by frame warping blocks. The Daisy-chaining model estimates the displacement field between frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$ in a coarse-to-fine manner: a coarse field is first estimated at low-resolution, and then refined with estimations of higher resolution. Concretely, the first optical flow unit produces a coarse estimation $\boldsymbol{D}_{\text{coarse}}$ of the motion between the input frames. The warping block that follows warps input frame $\boldsymbol{I}_{i+1}$ onto $\boldsymbol{I}_i$ according to $\boldsymbol{D}_{\text{coarse}}$. Then, the following units apply the same procedure to frames $\boldsymbol{I}_i$ and $\boldsymbol{I}^w_{i+1}$, the latter being the warped version of $\boldsymbol{I}_{i+1}$. In this way, the residual displacement field existing between $\boldsymbol{I}_i$ and $\boldsymbol{I}^w_{i+1}$ is estimated in the next stages. The estimations produced by the different optical flow units are combined to form the resulting displacement field $\boldsymbol{D}$. Note that the deeper an optical flow unit located, the higher the resolution of the flow estimation. In order to match the horizontal and vertical dimensions of a frame, every flow estimate is upsampled using B-spline interpolation before feeding the next frame warping block. A schematic describing the estimation process of the Daisy-chaining model is shown in Figure 2.12. An optical flow unit is composed of a succession of n 3×3 convolution layers, interleaved with n 2×2 average pooling layers. Two additional 3×3 convolution layers followed by two 1×1 convolution layers and a B-spline interpolation layer end the unit. Interpolation is necessary to obtain a dense optical flow with one motion vector per pixel. A diagram showing the structure of the optical flow unit is given in Figure 2.13.

A Tensorflow implementation of the Daisy-chaining model is available at `https://github.com/torjush/Strain_estimation`. Haukom *et al.* [22] trained the model using the unsupervised framework presented in subsection 2.5.1.

2.5.3 Recurrent All-Pairs Field Transforms (RAFT) model

The RAFT model is not a coarse-to-fine optical flow model like Daisy-chaining: RAFT maintains and refines a single high-resolution flow field. This procedure mitigates several limitations that coarse-to-fine models generally encounter: the complexity of correcting large errors at coarse resolutions, the proclivity to overlook small fast-moving objects, and the large number of training iterations usually required

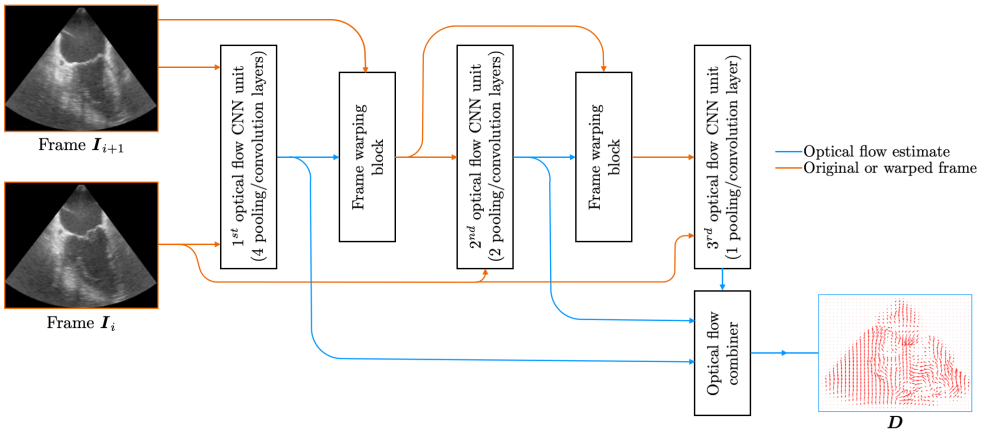


Figure 2.12: Simplified architecture of the Daisy-chaining model developed by Haukom *et al.* [22]. Arrows indicate the flow of inputs and outputs. The output of a warping block is the warped version of its input frame.

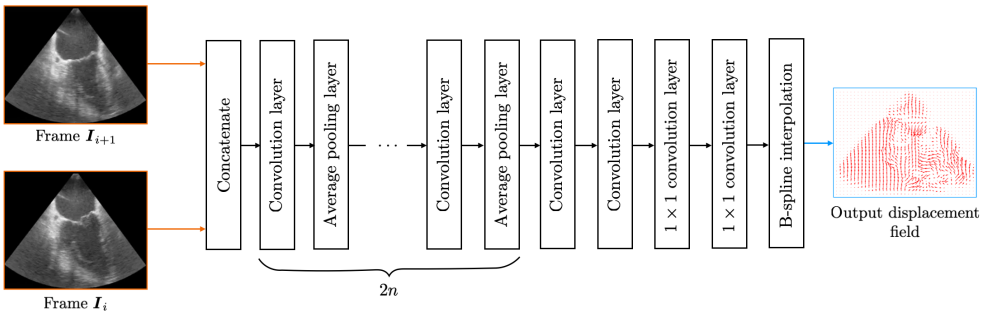


Figure 2.13: Architecture of an optical flow unit of the Daisy-chaining model. The estimated flow has a resolution 2^n times lower than the input frames. B-spline interpolation is used to match the dimensions of the estimated flow and the input frames.

by multi-stage cascades to present decent performances. The RAFT model can be broken down into three stages:

1. **Feature and context encoders** - The feature encoder is a CNN that extracts feature maps from both input frames. The extracted feature maps have a resolution 8 times lower than the input frames. The context encoder has a similar architecture to the feature encoder and extracts features from the first frame only. Mathematically, both encoders are mappings from $\mathbb{R}^{H\times W\times 3}$ to $\mathbb{R}^{\frac{H}{8}\times\frac{W}{8}\times 256}$, with H and W the height and width of the input frames.
2. **Correlation layer** - This layer computes the inner product of all pairs of feature vectors, producing a 4-D correlation volume of dimensions $H \times W \times H \times W$. Then, the two last dimensions of this volume are pooled with 1×1, 2×2, 4×4 and 8×8 kernels in order to build a 4-layer pyramid $\{\boldsymbol{C}^1, \boldsymbol{C}^2, \boldsymbol{C}^3, \boldsymbol{C}^4\}$, where $\boldsymbol{C}^k$ has dimensions $H \times W \times H/2^k \times W/2^k$.

3. **Update operator** - Produces a sequence of flow estimates $\{\boldsymbol{D}_1, \cdots, \boldsymbol{D}_N\}$ according to the update equation $\boldsymbol{D}_{k+1} = \boldsymbol{D}_k + \boldsymbol{\Delta D}_k$, with $\boldsymbol{D}_0 = \boldsymbol{0}$. The update $\boldsymbol{\Delta D}_k$ depends on the previous flow estimates, the correlation pyramid, the context feature map and a latent hidden state specific to the update operator. The current flow estimate is used to look up values from the set of correlation volumes. The architecture core of the operator is a sequence of gated recurrent units (GRUs) where fully connected layers are replaced by convolutions. The diagram of the GRU is detailed in Appendix A.

A simple diagram representing the architecture of RAFT is shown in Figure 2.14. A fully-functional PyTorch implementation of RAFT is available at `https://github.com/princeton-vl/RAFT`.

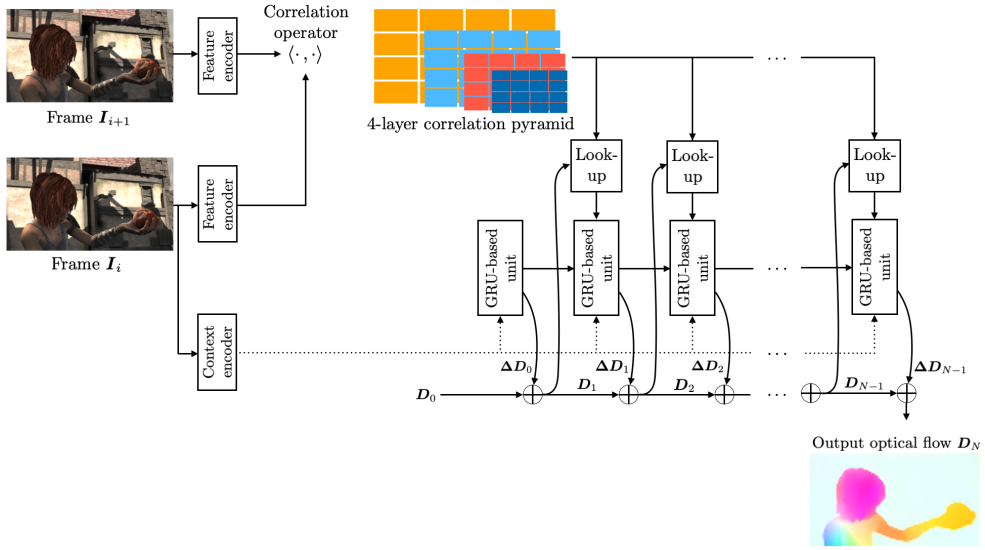


Figure 2.14: Simplified architecture of the RAFT model developed by Teed and Deng [32].

2.5.4 Lucas-Kanade method

The Lucas-Kanade method [25] is an optical flow algorithm that strongly relies on spatial coherence: it assumes the flow is essentially constant in a local neighborhood around a considered pixel. This thesis uses the pyramidal implementation proposed by Bouguet *et al.* [69]. This method builds two L-level pyramidal representations $\{\boldsymbol{I}_i^l\}$ and $\{\boldsymbol{I}_{i+1}^l\}$ of both images $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$ such that $\boldsymbol{I}_j^0 = \boldsymbol{I}_j$ and $\boldsymbol{I}_j^l \in \mathbb{R}^{\frac{H}{2^l} \times \frac{W}{2^l}}$, $0 \leq l \leq L$. Consider a window of dimension $(2\omega_x + 1) \times (2\omega_y + 1)$ centered on a considered pixel $\mathbf{u}$, the method estimates the flow in this window in a coarse-to-fine manner, pulling up from layer L to layer 0. It starts with guesses $\boldsymbol{G}^{\boldsymbol{L}} \in \mathbb{R}^{2\times 2}$ and $\boldsymbol{g}^L \in \mathbb{R}^2$ and finds the residual affine matrix $\boldsymbol{A}^L$ and residual displacement vector

$\boldsymbol{d}^L$ such that the error

$$\epsilon\left(\boldsymbol{d}^L, \boldsymbol{A}^L\right) = \sum_{x=-\omega_x}^{\omega_x} \sum_{y=-\omega_y}^{\omega_y} \left(\boldsymbol{J}_i^L(\mathbf{x}) - \boldsymbol{J}_{i+1}^L(\boldsymbol{A}^L\mathbf{x} + \boldsymbol{d}^L)\right)^2 \tag{2.20}$$

is minimized. In this equation, $\mathbf{x} = [x, y]^T$ and

$$\boldsymbol{J}_i^L(\mathbf{x}) = \boldsymbol{I}_i^L\left(\mathbf{x} + \mathbf{u}^L\right), \tag{2.21}$$

$$\boldsymbol{J}_{i+1}^L(\mathbf{x}) = \boldsymbol{I}_{i+1}^L\left(\boldsymbol{G}^L\mathbf{x} + \boldsymbol{g}^L + \mathbf{u}^L\right), \tag{2.22}$$

where the superscript indicates the current layer of the pyramidal representation. Computations are then propagated to the above layer by setting

$$\boldsymbol{g}^{L-1} = 2(\boldsymbol{g}^L + \boldsymbol{G}^L\boldsymbol{d}^L) \tag{2.23}$$

$$\boldsymbol{G}^{L-1} = \boldsymbol{G}^L\boldsymbol{A}^L. \tag{2.24}$$

Knowing the affine transformation undergone by pixels in the considered window, the flow is easily computed. Note that the minimization of Equation 2.20 is performed iteratively at every pyramidal level. Further details about pyramidal representation, the optimization of Equation 2.20 and the flow computation can be found in [69]. Note that this optical flow technique is not a deep learning model and does not require training. The python library OpenCV provides a practical implementation of the described method (see `OpenCV/doc`).

2.5.5 Gunnar Farnebäck method

Farnebäck [27] developed an optical flow method based on polynomial expansion. In concrete terms, a local neighborhood $\mathcal{N}$ of pixels in images $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$ is approximated by a quadratic polynomial such that

$$\boldsymbol{I}_i(\mathbf{x}) \approx \mathbf{x}^T\boldsymbol{A}_1(\mathbf{x})\mathbf{x} + \mathbf{b}_1(\mathbf{x})\mathbf{x} + c_1(\mathbf{x}), \tag{2.25}$$

$$\boldsymbol{I}_{i+1}(\mathbf{x}) \approx \mathbf{x}^T\boldsymbol{A}_2(\mathbf{x})\mathbf{x} + \mathbf{b}_2(\mathbf{x})\mathbf{x} + c_2(\mathbf{x}), \tag{2.26}$$

for all $\mathbf{x}$ in the neighborhood. Let define

$$\boldsymbol{A}(\mathbf{x}) = \frac{\boldsymbol{A}_1(\mathbf{x}) + \boldsymbol{A}_2(\mathbf{x})}{2} \tag{2.27}$$

and

$$\boldsymbol{\Delta b}(\mathbf{x}) = \frac{\boldsymbol{b}_2(\mathbf{x}) - \boldsymbol{b}_1(\mathbf{x})}{2}. \tag{2.28}$$

Properties of polynomials show that

$$\boldsymbol{A}(\mathbf{x})\boldsymbol{d}(\mathbf{x}) = \boldsymbol{\Delta b}(\mathbf{x}) \tag{2.29}$$

if the polynomial from Equation 2.26 is obtained by translating the polynomial from Equation 2.25 with the vector $\boldsymbol{d}(\mathbf{x})$. The Gunnar Farnebäk method exploits this result and finds the motion vector $\boldsymbol{d}(\mathbf{x})$ that minimizes the expression

$$\sum_{\Delta\mathbf{x}\in\mathcal{N}} w(\Delta\mathbf{x}) \left\| \boldsymbol{A}(\mathbf{x}+\Delta\mathbf{x})\boldsymbol{d}(\mathbf{x}) - \Delta\mathbf{b}(\mathbf{x}+\Delta\mathbf{x}) \right\|^2, \tag{2.30}$$

where $w(\cdot)$ is an arbitrary weight function for pixels in $\mathcal{N}$. As the Lucas-Kanade method presented previously, the algorithm also uses a pyramidal representation of its input images. The optimization of Equation 2.30 is then performed iteratively at every pyramidal level. The proof of Equation 2.29 and further explanations about the method can be found in [27]. The python library OpenCV provides a practical implementation of the described method (see `OpenCV/doc`).

2.6 U-Net: biomedical images segmentation model

Ronneberger *et al.* [42] developed an efficient convolutional model for biomedical images segmentation called U-Net. This neural network allows for a strong use of data augmentation[8], which is particularly convenient when only a small amount of annotated images is available for training. The model architecture is made of a 4-stages contracting path, followed by a 4-stages expansive path. Short-cut connections concatenate the output of each stage of the contracting path to the input of expansion stage located at the same depth. A stage of the contracting part consists of two consecutive 3×3 unpadded convolutional layers (with ReLU activation functions), followed by a 2×2 max pooling layer with stride 2. The number of feature channels is doubled after each stage, starting with 64 feature channels on the first stage. Stages of the expansion path have the same structure, except that the max pooling layer is replaced by a 2×2 up-convolution layer, i.e. a 2×2 up-sampling operator followed by a 2×2 unpadded convolution. The number of feature channels is divided by 2 at each stage of the expansion path. A detailed diagram of the architecture is available in Figure 2.15. A python implementation of U-Net is available at `http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net`.

2.7 Point tracking in 2-D video sequences using a Kalman filter

Consider a point moving along a video sequence. Let assume that this point is represented on a frame by a single pixel. Its motion can be described by the

[8] Data augmentation consists in building a larger training set from a small set of annotated inputs. New data is created by voluntarily altering or/and transforming the available annotated data. In image processing, image cropping, rotation, translation, flipping,... are common data augmentation operations.

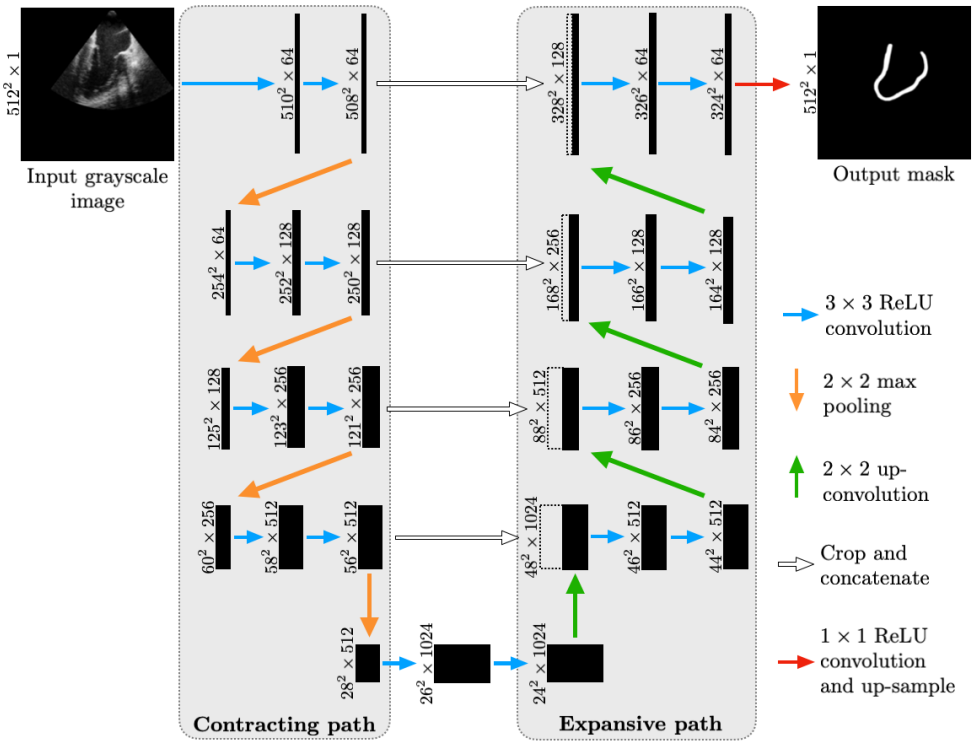


Figure 2.15: Diagram of U-Net architecture. [42] The dimensions of the images and features maps are written along their left side. The squared number refers to the xy-size (e.g. $512^2 = 512 \times 512$), the last number refers to the depth.

following discrete-time linear time-invariant model:

$$\mathbf{x}_{i+1} = \boldsymbol{F}\mathbf{x}_i + \boldsymbol{G}\boldsymbol{u}_i + \boldsymbol{n}_i, \tag{2.31}$$

where $\mathbf{x}_i \in \mathbb{R}^2$ is the coordinate vector of the pixel in frame i, $\boldsymbol{n}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{C_n})$ is the noise inherent to the process modeled as a Gaussian random vector with covariance $\boldsymbol{C_n}$, and $\boldsymbol{u}_i \in \mathbb{R}^2$ is the velocity vector of the moving pixel in frame i. The velocity is considered as an input to the system. The matrices $\boldsymbol{F} \in \mathbb{R}^{2\times2}$ and $\boldsymbol{G} \in \mathbb{R}^{2\times2}$ are the transition and control matrices. They respectively describe the dynamics of the system and how the inputs (i.e. the velocity of the point) are coupled to the system states (i.e. the coordinates of the point). In the case presented here, they are given by

$$\boldsymbol{F} = \boldsymbol{G} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{2.32}$$

Kalman filtering is a two-step procedure. Firstly, an estimate $\hat{\mathbf{x}}^+_{i+1}$ of the system state is predicted using

$$\hat{\mathbf{x}}^+_{i+1} = \boldsymbol{F}\hat{\mathbf{x}}_i + \boldsymbol{G}\boldsymbol{u}_i. \tag{2.33}$$

The uncertainty in $\hat{\mathbf{x}}_i$ is given by the estimated covariance

$$\hat{\boldsymbol{P}}_{i+1}^{+} = \boldsymbol{F}\hat{\boldsymbol{P}}_i\boldsymbol{F}^T + \hat{\boldsymbol{V}}, \tag{2.34}$$

where $\hat{\boldsymbol{V}}$ is an estimate of the process noise covariance $\boldsymbol{C_n}$. Secondly, the state and uncertainty estimates are updated with available measurements. The relation between the measurement $\boldsymbol{z}_i$ and the corresponding state can be written as

$$\boldsymbol{z}_i = \boldsymbol{H}\mathbf{x}_i + \boldsymbol{w}_i, \tag{2.35}$$

where $\boldsymbol{H} \in \mathbb{R}^{2\times 2}$ is called the measurement matrix, and $\boldsymbol{w}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{C_w})$ is a Gaussian random variable with covariance $\boldsymbol{C_w}$ modeling the measurement noise. The updated state and covariance are given by

$$\hat{\mathbf{x}}_{i+1} = \hat{\mathbf{x}}_{i+1}^{+} + \boldsymbol{K}(\boldsymbol{z}_{i+1} - \boldsymbol{H}\hat{\mathbf{x}}_{i+1}^{+}), \tag{2.36}$$

$$\hat{\mathbf{P}}_{i+1} = \hat{\mathbf{P}}_{i+1}^{+} - \boldsymbol{K}\boldsymbol{H}\hat{\mathbf{P}}_{i+1}^{+}, \tag{2.37}$$

where $\boldsymbol{K}$ is called Kalman gain and is computed as

$$\boldsymbol{K} = \hat{\mathbf{P}}_{i+1}^{+}\boldsymbol{H}^T\left(\boldsymbol{H}\hat{\mathbf{P}}_{i+1}^{+}\boldsymbol{H}^T + \hat{\boldsymbol{W}}\right)^{-1}. \tag{2.38}$$

$\hat{\boldsymbol{W}}$ is an estimation of the measurement noise covariance $\boldsymbol{C}_w$. If the process noise and the measurement noise actually follow a Gaussian distribution with zero mean, the Kalman filter constitutes an optimal estimator and $\hat{\mathbf{x}}$ is the best estimate that can be computed in view of available information. [70] The python library OpenCV provides an implementation of the Kalman filter (see `OpenCV/doc`).

Chapter 3

Materials and Method

3.1 Data

Two data sets were built using two different ultrasound data acquisition techniques. The first set of data consists of high-rate B-mode sequences of the 4-chamber, 2-chamber and apical long-axis views of the heart. The second set consists of standard TVI sequences, acquired on the same patient cohort as the first, recorded simultaneously with B-mode sequences of lower frame-rate. The two sets are respectively referred as the HR and TVI sets.

The set of data used to perform point tracking and strain estimation can have a big impact on the results. Indeed, high-rate B-mode sequences exhibit smaller frame-to-frame tissue displacements compared to lower frame-rate sequences, which could be easier to predict. On the other hand, TVI data gives partial information about tissue displacements that can be used to improve frame-to-frame motion predictions in low-rate sequences. Point tracking and strain estimation is performed on both sets. Optical flow methods are adapted to take advantage of TVI data when applied on the TVI set.

3.1.1 Data acquisition

B-mode image sequences were recorded for 88 different patients at the Clinic of Cardiology at St. Olavs University Hospital located in Trondheim, Norway. Recordings were made during routine exams by cardiologists from the Echocardiography Unit. To do so, they used GE Vivid E9, E95 and S70 scanners and a TEE 6VT-D probe (GE Vingmed, Horten, Norway). No patient selection was performed. The collection and use of ultrasound data were consented and approved by the patients involved and the ethics committee of St. Olavs University Hospital. The recordings of 18 of the 88 patients are used as test set and are not used in the training procedures described later in this chapter.

The raw DICOM data was scan-converted to an isotropic pixel size of 0.5 mm. The

pixel brightness of B-mode sequences ranges from 0 to 255. The frame rate of HR B-mode sequences is included in [21.1, 61.2] fps. B-mode sequences from the TVI set have a frame rate included in [12.4, 33.5] fps, while TVI sequences frame rate is in [49.5, 111.5] fps. The time resolution is in average four times greater for a TVI sequence compared to the corresponding low-rate B-mode sequence.

3.1.2 Data pre-processing

A data pre-processing pipeline is applied to the B-mode and TVI sequences before any further processing. The first stage of the pipeline consists in zero-padding and/or cropping[1] the video frames so that their dimension matches 512×512. Then, a proprietary contrast enhancement algorithm is applied to B-mode sequences. The algorithm performs a non-linear mapping of the gray values to enhance the tissue boundary. B-mode pixel values are then scaled down[2] to $[0, 1]$. Simultaneously, TVI sequences undergo a polar-to-Cartesian transformation in order to express velocity in the Cartesian framework of a frame. Then, temporal and spatial alignments of the TVI and low-rate B-mode sequences are performed. Indeed, the sector size and the orientation of the ultrasound beam in TVI and B-mode acquisitions differ. The scan conversion algorithm used to make the TVI and B-mode beam sectors match can displace of few pixels the origin of the probe in the TVI frames. TVI and B-mode acquisition starting instants can also differ slightly. Zero-padding is used to generate TVI images that match the size of the B-mode frames. Temporal alignment is done by erasing the parts of the TVI and corresponding B-mode sequences that do not overlap in time. The last stage of the pre-processing pipeline consists in integrating TVI sequences over several short periods of time. This results in a sequence of coarse displacement fields, where the field located at index i in the sequence describes the motion between frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$ of the corresponding B-mode sequence. An illustration of the pre-processing pipeline is given in Figure 3.1. Additional information concerning the polar-to-Cartesian transformation and the TVI integration process is available in Appendix B.

3.1.3 Data annotation

Before being able to track points and compute strain values, some data annotations are required. First, the ES and ED instants are spotted and tagged for every single B-mode sequence. A graphical user interface is developed for this purpose. It is also necessary to determine the points at the extremities of the basal segments that will be used in the strain computation process. This operation needs to be performed on the first frame of all B-mode sequences, an optical flow model combined with

[1] Note that cropping a frame does not affect areas that are of interest in the basal strain estimation process.

[2] Scaling down brightness values to the range $[0, 1]$ is a common practice when preparing data to feed CNN-based models.

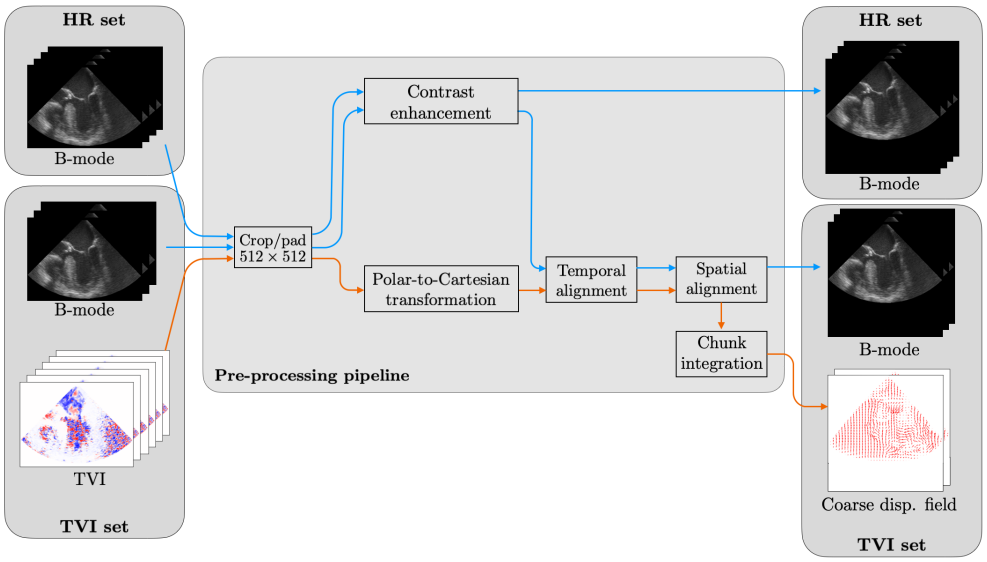


Figure 3.1: Diagram of the data pre-processing pipeline. HR and TVI sets are the two sets of data on which point tracking and strain estimation are performed.

a tracking method taking care of tracking the points the rest of the sequence. A semi-automatic tool is created to do so. More details about this tool are given in section 3.2.

3.2 Semi-automatic tool for myocardial point extraction using segmentation

Basal strain computation requires to measure the distance between the extremities of basal segments at different time instants. A semi-automatic tool is developed to place suitable points at those locations on the first frame of every B-mode sequence. These points are then tracked along the sequence and the distance between them is used in the strain computation process (see Equation 2.3).

The semi-automatic tool uses U-Net to segment the myocardium surrounding the left ventricle (LV). A thinning algorithm is then applied to extract the centerline of the segmented area. The points constituting the centerline are decimated to keep a user-defined number of equidistant points spread around the LV. The user's intervention is finally required to choose four points among the remaining ones (one for each extremity of both basal segments present on a frame) to be used for basal strain computation. Even though they do not intervene directly in the strain estimation process, the points that are not chosen by the user are used by a tracking method based on Kalman filtering.

3.2.1 Myocardial segmentation using U-Net

Segmentation data sets

To train U-Net, one-hundred B-mode frames are manually annotated by the author himself. Each one of them are picked randomly from different B-mode sequences, themselves chosen randomly among all B-mode sequences of both TVI and HR sets. The annotation consists in creating a mask for the part of the myocardium surrounding the LV. Data augmentation is used to increase the size of the training set from 100 to 5000 images. To do so, all annotated frames are sent 50 times through a data augmentation pipeline that comprises the following stages:

1. **Affine transformation** - The same random affine transformation is applied to the frame and the corresponding mask. The transformation is a composition of a rotation of angle $|\alpha| \leq 180°$, a translation of vector $\boldsymbol{d}_{\text{tr}} = [d_x, d_y]$ with d_x and d_y respectively smaller than 20% of the height and width of the frame, a scaling operation with a scaling factor $k \in [0.8, 1.2]$, and a shearing operation with horizontal and vertical shear angles $\in [-10°, 10°]$.
2. **Horizontal flip** - The frame and its corresponding mask are horizontally flipped with a probability of 0.5.
3. **Vertical flip** - The frame and its corresponding mask are vertically flipped with a probability of 0.5.
4. **Gamma augmentation** - The frame $\boldsymbol{I}_{in}$ is modified according to the pixel-wise operation $\boldsymbol{I}_{out} = \boldsymbol{I}_{in}^{\gamma}$, with $\gamma \in [0.5, 1.25]$. This operation modifies the contrast of the frame: $\gamma > 1$ enhances the brightness while $\gamma < 1$ makes the image darker.
5. **Denoising filter** - A denoising filter with a kernel size of 5×5 is applied to 80% of the annotated frames. Its role is to blur the image in order to smooth out the speckle noise inherent to B-mode images. The type of filter is either mean, median, bilateral or Gaussian. Each type is applied to a same number of frames. This procedure should make U-Net less sensitive to the type of filter that is used.

Figure 3.2 shows a B-mode frame and its manually annotated mask, along with the results of two realizations of the data augmentation pipeline. 20% of the training set (randomly chosen) is used for validation. Therefore, U-Net is trained on 4000 B-mode frames, and validation is run after every epoch over 1000 B-mode frames.

U-Net training

U-Net is trained with the supervised learning paradigm. The binary cross-entropy is minimized during training. This loss function is defined as

$$BCE(\boldsymbol{P}, \boldsymbol{M}) = -\frac{1}{HW} \sum_{i=0}^{H-1} \sum_{i=0}^{W-1} \boldsymbol{M}_{ij} \cdot \log_2(\boldsymbol{P}_{ij}) + (1 - \boldsymbol{M}_{ij}) \cdot \log_2(1 - \boldsymbol{P}_{ij}), \quad (3.1)$$

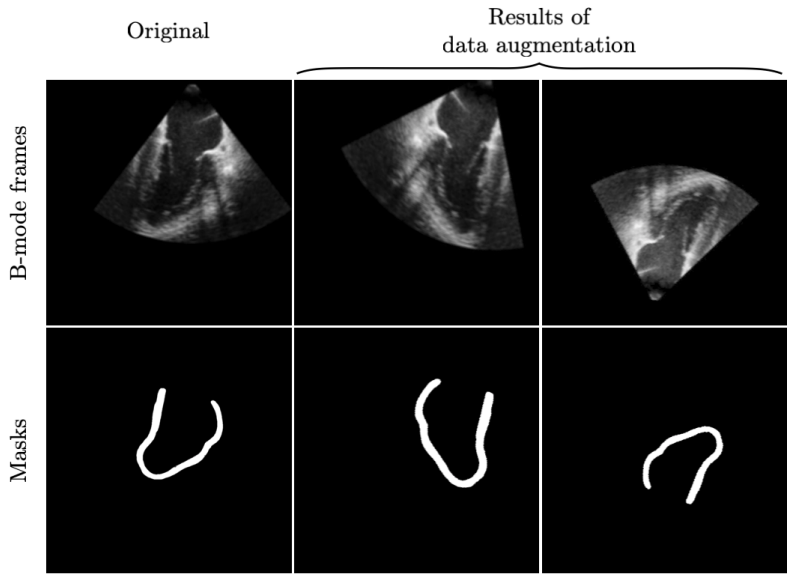


Figure 3.2: Examples of data used to train U-Net for the LV myocardium segmentation task. B-mode frames are given as inputs to U-Net, the masks are the corresponding labels. The original frame was manually annotated. Augmented data were obtained by applying a data augmentation pipeline on the original B-mode frame and its corresponding mask.

where $\boldsymbol{P}$ and $\boldsymbol{M}$ are respectively the predicted and targeted masks, H and W being their height and width. Note that the pixel values of $\boldsymbol{P}$ and $\boldsymbol{M}$ are in the range $[0,1]$. The binary accuracy is used as a validation metric. It is computed as

$$BA(\mathcal{V}) = \frac{1}{|\mathcal{V}|} \sum_{\boldsymbol{I} \in \mathcal{V}} \frac{\text{number of pixels in } \boldsymbol{I} \text{ correctly predicted}}{\text{number of pixels in } \boldsymbol{I}}, \tag{3.2}$$

where $\mathcal{V}$ is the validation set and $|\mathcal{V}|$ is its cardinality. A pixel is correctly predicted if its predicted category (1 if it belongs to the LV myocardium, 0 otherwise) matches the category specified by the mask. The validation metric is computed after every epoch.

U-Net is trained with the Adam optimizer. The batch size is set to 4 and the model weights are initialized with a truncated normal distribution. The learning rate is set to 0.001 and is reduced by a factor of 0.8 each time the validation metric does not improve for more than two epochs. The learning rate is not reduced further when its value reaches 10^{-6}. The training is stopped when no significant improvement of the validation metric is observed.

3.2.2 Thinning algorithm and point extraction

Beforehand, the masks produced by U-Net are processed in such a way that only the largest segmented area is retained. Indeed, U-Net can sometimes produce

masks that are composed of several pieces. Most of the time, the largest piece is the desired area of the myocardium while other small pieces are wrongly detected segment that need to be discarded.

A thinning algorithm is developed in order to extract the centerline of LV myocardium masks. The algorithm can be divided into three stages. The first stage applies the thinning algorithm developed by Zhang and Suen [71] in 1984. This algorithm is detailed in Appendix C. It extracts the skeleton of the mask. Since predicted masks are not perfect, the Zhang-Suen algorithm generally outputs a skeleton that has several ramifications and/or inner loops rather than a single centerline. The second stage consists thus in refining this skeleton in order to find the line that approximates at best the true centerline of the myocardium surrounding the LV. The refinement process begins by finding the two ends of the skeleton that are most likely to be the two ends of the desired centerline. Then, the longest path in the skeleton that joins those two ends is kept as centerline. More details about the skeleton refining algorithm can be found in Appendix C.

Once the centerline of the myocardium around the LV is found, its pixels are decimated until only N_p equidistant pixels remain. Note that the pixels located at the extremities of the centerline are among the remaining pixels. The number N_p is defined by the user. Then, the user is asked to select four pixels among the N_p left to be used in the strain estimation process. The entire point extraction procedure is illustrated in Figure 3.3. Results obtained at the different stages of the pipeline for three different B-mode frames are shown in Figure 3.4.

3.3 Optical flow methods

Four different optical flow algorithms are tested to predict the frame-to-frame motion of the points to be tracked: the Daisy-chaining model, the RAFT model, the Lucas-Kanade method and the Gunnar Farnebäck method. The two first models are deep learning-based and require some training before being able to produce meaningful results. On the other hand, the Lucas-Kanade and Gunnar Farnebäck methods do not have any trainable parameters and do not require any training before being applied. When applied to the TVI set, the Daisy-chaining model and the Lucas-Kanade and Gunnar Farnebäck methods exploit the coarse displacement field computed by integrating TVI data. Modifying the architecture of RAFT to use TVI data required too much time and is left as future work. The RAFT model is therefore only used on the HR set.

3.3.1 Daisy-chaining model training

To take advantage of the information provided by the TVI data, a slightly modified version of the Daisy-chaining model is made: before feeding the model, the second input frame is folded according to the coarse displacement field obtained by in-

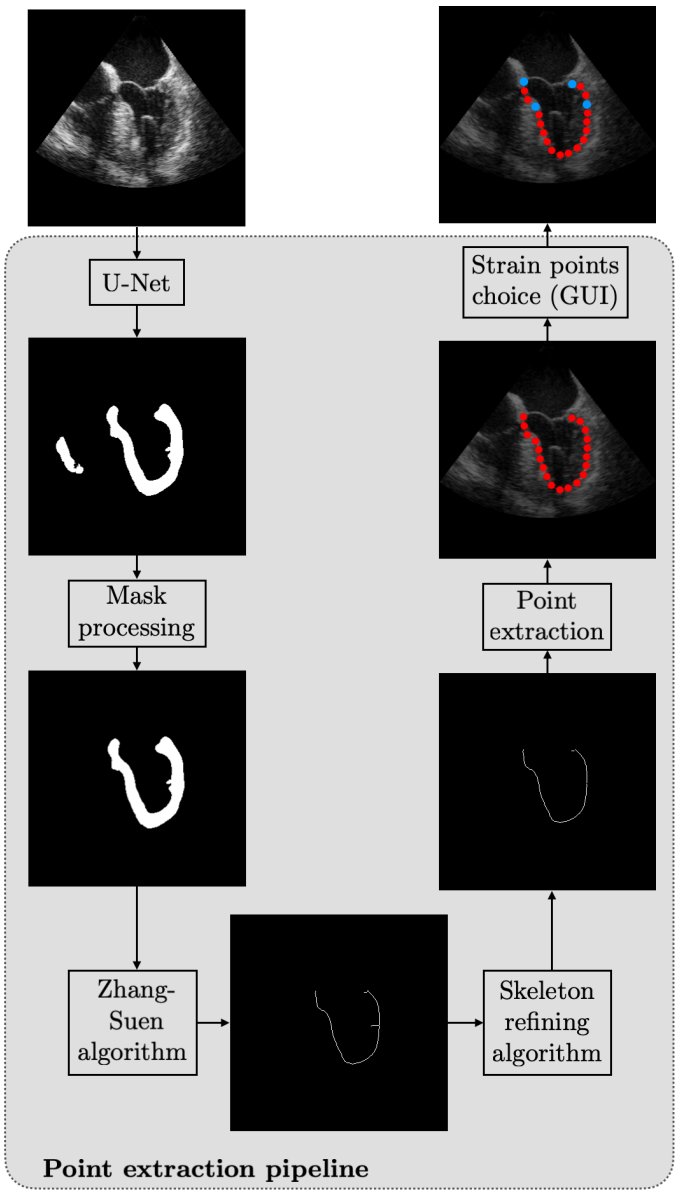


Figure 3.3: Diagram of the myocardial point extraction pipeline. In this illustration, the number of points N_p extracted from the centerline is set to 20. Blue points are the points chosen by the user to be used in the strain estimation process. Red points are still useful to the Kalman filter-based tracking method (see section 3.4).

tegrating the TVI data. This new model version is shown in Figure 3.5. Note that setting the coarse displacement field to zero everywhere gives the same behavior as the classical Daisy-chaining model.

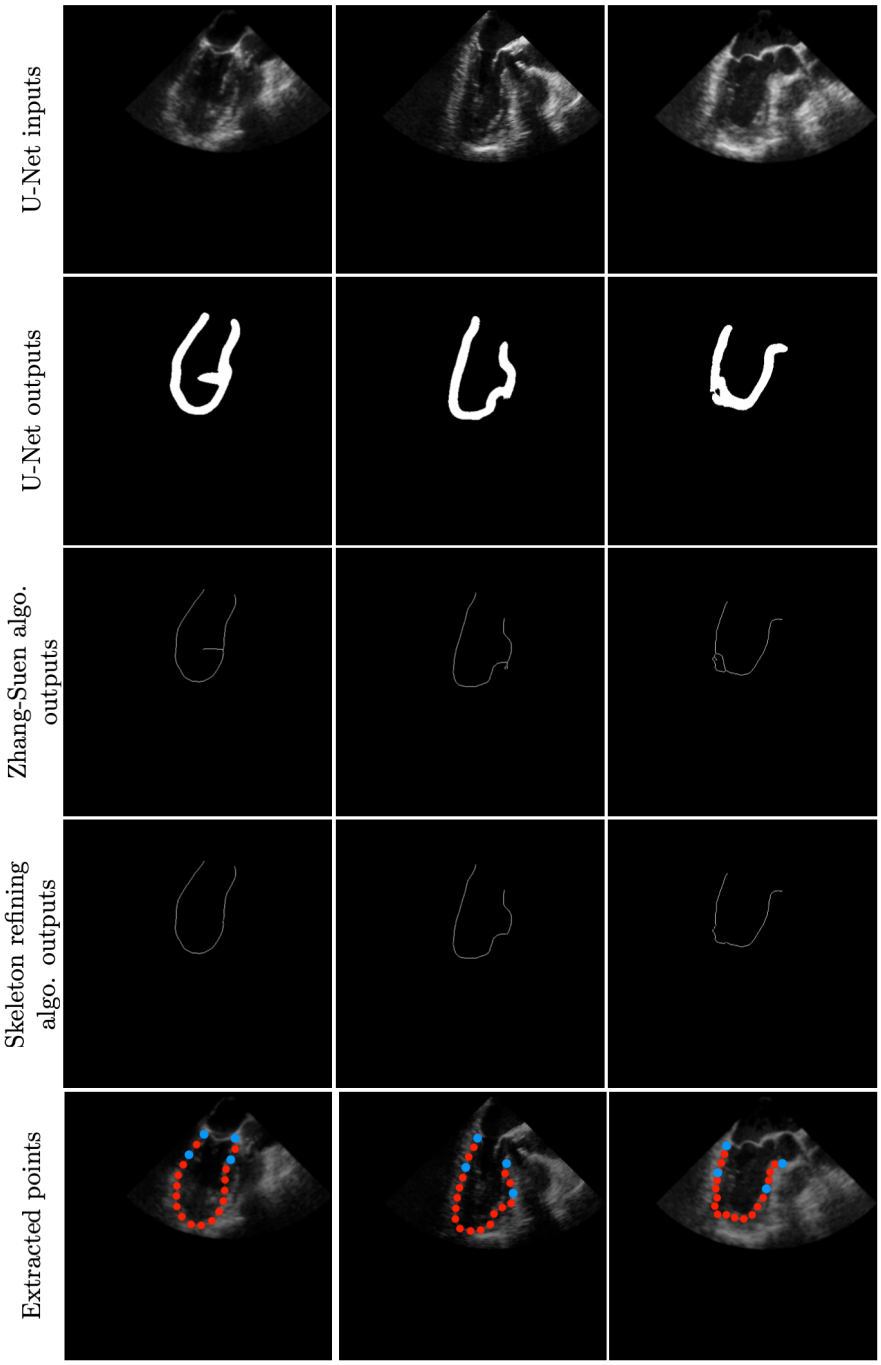


Figure 3.4: Examples of results obtained at the different stages of the myocardial point extraction pipeline. In this illustration, the number of points N_p extracted from the centerline is set to 20. Blue points are the points chosen by the user to be used in the strain estimation process. Red points are still useful to the Kalman filter-based tracking method (see section 3.4).

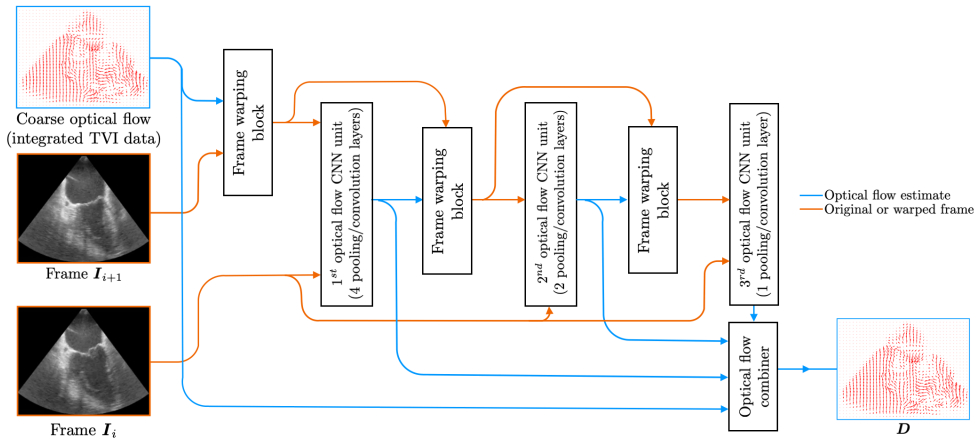


Figure 3.5: Adapted version of the Daisy-chaining model for TVI information exploitation. Arrows indicate the flow of inputs and outputs. The output of a warping block is the warped version of its input frame.

The Daisy-chaining model is trained using the unsupervised framework described in subsection 2.5.1. The training set is formed by pairing consecutive B-mode frames together. A B-mode sequence of n frames produces $n-1$ pairs. The B-mode sequences of the TVI set only are used: the HR set was made available to the author once the training phase of the Daisy-chaining model was done. Training a second time the Daisy-chaining model on the HR set would have caused too much delay. However, TVI data is used only half of the time: the model from Figure 3.5 is used for all pairs, but the displacement field estimate obtained from TVI data integration is replaced by a zero-field in half of the cases. This practice should improve the abilities of the model to predict large tissue displacements. Data augmentation is used to double the number of training pairs. The data augmentation procedure is the following:

- The first third of the pairs are horizontally flipped. The tensors that represent the corresponding displacement fields are horizontally flipped and the sign of the horizontal component of the displacement is inverted.
- The second third of the pairs are vertically flipped. The tensors that represent the corresponding displacement fields are vertically flipped and the sign of the vertical component of the displacement is inverted.
- A random zoom is performed on the last third of the pairs. The zoom factor is chosen randomly in the range $[1.3, 3]$. The portion of the B-mode frame on which the zoom is performed is chosen randomly and always contains moving tissues. Bilinear interpolation is used to resize the zoomed frame. The same transformation is applied to the corresponding displacement field.

10% of the pairs are used as validation set.

The Daisy-chaining model is trained step-by-step. The first optical flow unit of the model is trained alone until the validation loss (negated ZNCC) reaches a

plateau. Then, the second optical flow unit is added. The trainable parameters of first unit are frozen and the model now composed of two optical flow units connected in cascade is trained. Once the validation loss reaches a new plateau, the last optical flow unit is added to the model. The complete model can thus start to be trained, the trainable parameters of the two previous layers being frozen. The Adam optimizer is used with a batch size of 16. Xavier initialization is used to initialize the model weights. The learning rate and the bending penalty scaling factor β (see Equation 2.19) are initially set to 10^{-4} and $5 \cdot 10^{-6}$ respectively. Once the validation loss reaches a plateau, fine-tuning is performed. The best results are obtained by decreasing the learning rate to $5 \cdot 10^{-5}$ and resetting β to $5 \cdot 10^{-7}$.

3.3.2 RAFT model training

The RAFT model is trained using the unsupervised framework (see subsection 2.5.1) with a different loss function than the negated ZNCC. The model architecture is not modified to take advantage of the TVI data and the model is therefore trained using the HR set only. As for the Daisy-chaining model, the training set is formed by pairing consecutive B-mode frames. Transfer learning is used: the model is initialized with weights that were pre-trained on the *FlyingThings3D* data set. This practice makes training converge faster. Since the pre-trained model already learned how to compute optical flow and just needs to be more specific to TEE images, it is believed that the size of the training set is sufficient and that no data augmentation is needed. Moreover, the diversity generally introduced by augmenting data is also not absolutely necessary in this case. Indeed, the B-mode sequences that will be used as test data are really similar to those the model is trained on. Consequently, the RAFT model should generalize easily to the test set data, as long as overfitting is avoided. A blurring filter with a 5×5 kernel is applied to 80% of the B-mode frame pairs: a Gaussian, a bilateral, a mean and a median filters are used, each of them filtering a same number of frame pairs.

Minimizing the negated ZNCC between frame $\boldsymbol{I}_i$ and warped frame $\boldsymbol{I}_{i+1}^w$ did not give any satisfactory results. Therefore, other loss functions are experimented to train the model. Consider two input frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$. Let distinguish the forward optical flow $\boldsymbol{D}^f$, describing the motion from $\boldsymbol{I}_i$ to $\boldsymbol{I}_{i+1}$, from the backward optical flow $\boldsymbol{D}^b$ that describes the motion from $\boldsymbol{I}_{i+1}$ to $\boldsymbol{I}_i$. The forward and backward flows are such that

$$\boldsymbol{D}^f(x, y) + \boldsymbol{D}^b\left(x + \boldsymbol{D}^f(x, y, 1),\, y + \boldsymbol{D}^f(x, y, 0)\right) = \boldsymbol{0}, \tag{3.3}$$

where $\boldsymbol{D}^f(x, y, 0)$ and $\boldsymbol{D}^f(x, y, 1)$ are the horizontal and vertical components of the displacement vector $\boldsymbol{D}^f(x, y)$ located at coordinates (x, y) in the field $\boldsymbol{D}^f$. RAFT is used to compute the forward and backward flows. It thus produces the sequences $\{\boldsymbol{D}_n^f\}_{n=1}^N$ and $\{\boldsymbol{D}_n^b\}_{n=1}^N$ of forward and backward flow estimates, where N is the number of iterations of the update operator of the model. Let $\mathcal{W}$ be the warping operator that warps an image according to a displacement field. If the

forward and backward flow estimates $\boldsymbol{D}_n^f$ and $\boldsymbol{D}_n^b$ are good, it comes

$$\begin{cases} \boldsymbol{I}_{i+1}^w = \mathcal{W}(\boldsymbol{I}_{i+1}, \boldsymbol{D}_n^f) & \approx \boldsymbol{I}_i, \\ \boldsymbol{I}_i^w = \mathcal{W}(\boldsymbol{I}_i, \boldsymbol{D}_n^b) & \approx \boldsymbol{I}_{i+1}, \\ \boldsymbol{I}_i^{ww} = \mathcal{W}\Big(\mathcal{W}(\boldsymbol{I}_i, \boldsymbol{D}_n^b), \boldsymbol{D}_n^f\Big) & \approx \boldsymbol{I}_i, \\ \boldsymbol{I}_{i+1}^{ww} = \mathcal{W}\Big(\mathcal{W}(\boldsymbol{I}_{i+1}, \boldsymbol{D}_n^f), \boldsymbol{D}_n^b\Big) & \approx \boldsymbol{I}_{i+1}. \end{cases} \tag{3.4}$$

The first loss function to be experimented is defined as

$$\begin{aligned} L_{RAFT,1} = \frac{1-\eta}{1-\eta^N} \sum_{n=1}^{N} \frac{\eta^{N-n}}{2} \Big[& P(\boldsymbol{D}_n^f) + P(\boldsymbol{D}_n^b) \\ & - 0.2 \Big(\mathrm{ZNCC}(\boldsymbol{I}_{i+1}^w, \boldsymbol{I}_i) + \mathrm{ZNCC}(\boldsymbol{I}_i^w, \boldsymbol{I}_{i+1}) \Big) \\ & - 0.8 \Big(\mathrm{ZNCC}(\boldsymbol{I}_i^{ww}, \boldsymbol{I}_i) + \mathrm{ZNCC}(\boldsymbol{I}_{i+1}^{ww}, \boldsymbol{I}_{i+1}) \Big) \Big], \end{aligned} \tag{3.5}$$

where η is a constant $\in [0, 1[$ set to 0.8, $P(\cdot)$ is the bending penalty function, and $\mathrm{ZNCC}(\boldsymbol{I}_i, \boldsymbol{I}_{i+1})$ is the zero-mean normalized cross-correlation between images $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$. The weights 0.2 and 0.8 were chosen empirically. The terms $\mathrm{ZNCC}(\boldsymbol{I}_i^{ww}, \boldsymbol{I}_i)$ and $\mathrm{ZNCC}(\boldsymbol{I}_{i+1}^{ww}, \boldsymbol{I}_{i+1})$ try to enforce the relation described in Equation 3.3. Both terms reach their maximal value if $\boldsymbol{D}_n^f = \boldsymbol{D}_n^b$. As for the terms $\mathrm{ZNCC}(\boldsymbol{I}_{i+1}^w, \boldsymbol{I}_i)$ and $\mathrm{ZNCC}(\boldsymbol{I}_i^w, \boldsymbol{I}_{i+1})$, they prevent the training procedure from converging towards the degenerate case $\boldsymbol{D}_n^f = \boldsymbol{D}_n^b = \boldsymbol{0}$. Note that the loss is computed over the full prediction sequences $\{\boldsymbol{D}_n^f\}_{n=1}^N$ and $\{\boldsymbol{D}_n^b\}_{n=1}^N$, with exponentially increasing weights. The second loss function to be experimented is defined as

$$\begin{aligned} L_{RAFT,2} = \frac{1-\eta}{1-\eta^N} \sum_{n=1}^{N} \frac{\eta^{N-n}}{2} \Big[& P(\boldsymbol{D}_n^f) + P(\boldsymbol{D}_n^b) \\ & - \mathrm{ZNCC}(\boldsymbol{I}_i^{ww}, \boldsymbol{I}_i) - \mathrm{ZNCC}(\boldsymbol{I}_{i+1}^{ww}, \boldsymbol{I}_{i+1}) \\ & + \frac{1}{H \cdot W} \Big(\|\boldsymbol{D}_n^f - \boldsymbol{D}_{GF}^f\|_2^2 + \|\boldsymbol{D}_n^b - \boldsymbol{D}_{GF}^b\|_2^2 \Big) \Big], \end{aligned} \tag{3.6}$$

where $\|\cdot\|_2$ is the L2-norm operator, H and W are the height and width of the B-mode frames, and where $\boldsymbol{D}_{GF}^f$ and $\boldsymbol{D}_{GF}^b$ are forward and backward flow estimates computed with the Gunnar Farnebäck method. It was observed that the flows computed with the Gunnar Farnebäck method are usually quite accurate. The loss in Equation 3.6 considers $\boldsymbol{D}_{GF}^f$ and $\boldsymbol{D}_{GF}^b$ as ground truth and thus penalizes big discrepancies between them and the estimates $\boldsymbol{D}_n^f$ and $\boldsymbol{D}_n^b$. This also prevents the training procedure from converging towards the degenerate case $\boldsymbol{D}_n^f = \boldsymbol{D}_n^b = \boldsymbol{0}$. Other terms appearing in $L_{RAFT,2}$ play the same role as in Equation 3.5.

The Adam optimizer is used with a batch size of 4. The learning rate is initially set to 10^{-4} and is modified after every batch using the one-cycle learning rate policy

described in [72]. Fine-tuning showed that $5 \cdot 10^{-5}$ is the best value to use for the bending penalty factor β. The number of iterations N of the update operator of the model is set to 15.

3.3.3 Lucas-Kanade and Gunnar Farnebäck methods

Both the Lucas-Kanade and the Gunnar Farnebäck methods are used as described in section 2.5. When they are applied on the TVI set, the optical flow is initialized with the coarse displacement field obtained by integrating the TVI data. The value of their parameters is found empirically.

The window size used by the lucas-Kanade method is set to 51×51. The number of layers of the pyramidal representation of images is set to 2. At each pyramid level l, the algorithm performs 10 iterations to incrementally find the best residual affine matrix $\boldsymbol{A}^l$ and displacement vector $\boldsymbol{d}^l$.

The number of layers of the pyramidal representation of images is set to 3 for the Gunnar Farnebäck method. At each pyramidal level, the algorithm performs 15 iterations.

3.4 Point tracking methods

Two methods are developed to track points from one B-mode frame to the next. They both rely on the estimation of optical flow between successive frames. They are referred as the classic and the Kalman auto-correction methods. Consider $M+1$ points in frame $\boldsymbol{I}_i$ represented by their vector of estimated coordinates $\hat{\mathbf{x}}_i^m = [x_i^m, y_i^m]^T$, $m = 0, \ldots, M$. The goal of the tracking methods is to estimate at best the locations $\hat{\mathbf{x}}_{i+1}^m$ of the points in frame $\boldsymbol{I}_{i+1}$ based on $\hat{\mathbf{x}}_i^m$ and the optical flow $\boldsymbol{D}$ describing the motion from $\boldsymbol{I}_i$ to $\boldsymbol{I}_{i+1}$. Flow $\boldsymbol{D}$ is computed with one of the four optical flow methods described in the previous section.

3.4.1 Classic method

The classic method is really simple. It computes the new positions using

$$\hat{\mathbf{x}}_{i+1}^m = \hat{\mathbf{x}}_i^m + \boldsymbol{D}(x_i^m, y_i^m). \tag{3.7}$$

Note that the optical flow estimate $\boldsymbol{D}$ is a discrete displacement field presenting a displacement vector for each pixel. Bilinear interpolation is used in the case where coordinates x_i^m and/or y_i^m are not integers.

3.4.2 Kalman auto-correction method

This method uses the Kalman filter described in section 2.7. As a reminder, the Kalman filter updates the position estimates $\hat{\mathbf{x}}_{i+1}^{m,+} = \hat{\mathbf{x}}_i^m + \boldsymbol{D}(x_i^m, y_i^m)$ with some

noisy measurements $\boldsymbol{z}_{i+1}^{m}$. However, such measurements are not available here. This issue is circumvented by a little legerdemain. Let assume that the tracked points are the points found by the semi-automatic point extraction tool, as shown in Figure 3.4. The sequence of $M+1$ points is ordered such that points $\mathbf{x}_{i+1}^{0}$ and $\mathbf{x}_{i+1}^{M}$ are the points closest to the mitral valve. Let also assume that M is large enough so that three adjacent points are almost located on a straight line. $\hat{\mathbf{x}}_{i+1}^{m,+}$ is updated with

$$\boldsymbol{z}_{i+1}^{m} = \begin{cases} \hat{\mathbf{x}}_{i+1}^{m,+} & \text{if } m \in \{0, M\} \text{ or if } m \text{ and } i \text{ are not} \\ & \text{both even or odd numbers,} \\ \frac{\hat{\mathbf{x}}_{i+1}^{m-1} + \hat{\mathbf{x}}_{i+1}^{m+1}}{2} & \text{if } m \notin \{0, M\} \text{ and if } m \text{ and } i \text{ are} \\ & \text{both even or odd numbers.} \end{cases} \tag{3.8}$$

The trick consists thus in correcting the position of a point with the location of the middle of the segment joining its two adjacent neighbor points. Note that the correction is made every two frames, alternating between points of even and odd index m. The points of index $m = 0$ or M have a single neighbor and are generally placed in a location that is easy to track (right next to the mitral valve). Consequently, it is decided that their position is not corrected. The Kalman auto-correction method is conceived to prevent the points from drifting[3]. By Imposing a constraint that links the location of a point to the location of its neighbors, the set of points acts more as a single deformable entity rather than a set of individual points. There is therefore less chances a point starts drifting if its neighbors are not affected by the drift.

The Kalman filter requires to estimate the process and measurement noise covariances $\boldsymbol{C}_n$ and $\boldsymbol{C}_w$. Since the measurements used by this tracking method are not real measurements but estimates obtained under strong assumptions, the measurement covariance is set so that more trust is put in the estimate $\hat{\mathbf{x}}_{i+1}^{m,+}$ than in $\boldsymbol{z}_{i+1}^{m}$. Fine-tuning led to

$$\hat{\boldsymbol{V}} = 0.0005\, \boldsymbol{I}_{2\times 2} \approx \boldsymbol{C}_n, \tag{3.9}$$

$$\hat{\boldsymbol{W}} = 0.5\, \boldsymbol{I}_{2\times 2} \approx \boldsymbol{C}_w, \tag{3.10}$$

where $\boldsymbol{I}_{2\times 2}$ is the 2×2 identity matrix.

3.5 Tracking and strain estimation assessment

The expert Erik Andreas Rye Berg provided ground truth strain values for 18 patients among the 88 who agreed to have their cardiac data used in this work. Those strain values were computed using the EchoPac package (GE Vingmed Ultrasound,

[3]It was noticed that one of the difficulties of tracking points through an ultrasound sequence is to prevent tracked points from slowly and constantly moving away from their initial position. This is called drifting.

Norway), a clinically approved tool. The B-mode and TVI sequences of those 18 patients constitute a test set on which the different optical flow models and tracking methods are tested and compared. It is important to note that the data of those patients were not used for training any of the deep learning-based models mentioned in this thesis.

Basal strain is estimated for the entire test set. All possible combinations of data set types (HR or TVI), optical flow algorithms and tracking methods are tried, except the combinations pairing the RAFT model and the TVI data set together. The points to be tracked are obtained using the semi-automatic point extraction tool. If U-Net fails to segment correctly the myocardium part surrounding the LV, the user has the possibility to perform the segmentation manually. Strain is computed according to Equation 2.3 using the four points chosen by the user. Note that for every B-mode sequence, the same points are tracked whatever the optical flow algorithm and tracking method that are used. This facilitates the comparison.

The tracking performances are analyzed subjectively by visual inspection: for every B-mode sequence and every optical flow and tracking methods, the tracking of the four points used in the strain estimation process is assessed. The percentage of successfully tracked points is reported for all tried combinations of data set type, optical flow algorithm and tracking method.

The strain in every basal segment of the three views of the heart is computed for every cardiac cycle in the considered B-mode sequence. The strain averaged across all cardiac cycles is then compared to ground truth values. The correlation ρ between the ground truth and computed strain values is calculated. The mean and standard deviation, noted MD and σ, of the absolute error between computed and ground truth strain values are also reported. A variability index is computed in every view for all combinations of methods used. It is computed as

$$\text{Variability index } = \frac{1}{P}\sum_{p=1}^{P}\frac{\sqrt{\text{Var}\{\boldsymbol{s}_p\}}}{\text{Mean}\{\boldsymbol{s}_p\}}, \tag{3.11}$$

$\boldsymbol{s}_p$ is the vector of strain values of patient p, computed for the considered combination of methods and data set. In other words,

$$\boldsymbol{s}_p = \begin{bmatrix} \text{strain value of } 1^{st} \text{ cardiac cycle of patient } p \\ \text{strain value of } 2^{nd} \text{ cardiac cycle of patient } p \\ \vdots \\ \text{strain value of last cardiac cycle of patient } p \end{bmatrix}. \tag{3.12}$$

P is the number of patients among the 18 constituting the test set for which $\text{Var}\{\boldsymbol{s}_p\}$ is not an outlier. The condition to be an outlier is

$$\text{Var}\{\boldsymbol{s}_p\} > Q_3 + 3 * (Q_3 - Q_1), \tag{3.13}$$

where Q_1 and Q_3 are the first and third quartiles of the set $\left\{\text{Var}\{\boldsymbol{s}_p\}\right\}_{p=1}^{P}$.

The four computed statistics play different roles:

1. The mean absolute error MD between the computed and ground truth averaged strain values measures how close to ground truth the strain estimates are in average on a considered set of patients (MD stands for Mean Distance).
2. The standard deviation σ of the absolute error gives an indication of the dispersion of the absolute estimation errors around MD.
3. The Pearson correlation coefficient ρ measures to which extent computed and ground truth strain values are related to each other. A perfect correlation between the two gives $\rho = 1$ while the total absence of correlation gives $\rho = 0$.
4. The variability index is proposed as a relative measure of the confidence a model has when estimating strain: a model that outputs strain values really different from one cardiac cycle to another will have a large variability index, while a model that predicts fairly similar strain values for all cardiac cycles of patient will have a low variability index. Equation 3.13 is necessary to move aside patients for who unsuccessful tracking leads to an abnormally large variance term $\text{Var}\{\boldsymbol{s}_p\}$.

Chapter 4

Results

This chapter gathers the results obtained by following the methodology described in chapter 3. First, the evolution of the metrics monitored during the training of U-Net, Daisy-chaining and RAFT models is shown. Then, performances of the point extraction tool are presented, followed by a qualitative analysis of the tracking. Finally, results of a quantitative strain comparison against expert reference values are shown.

4.1 Training curves

The evolution of the training and validation loss metrics as a function of the training step or epoch is shown for the three deep learning-based models used in this work. Figure 4.1 shows the training and validation curves of U-Net, trained to segment the myocardial area that surrounds the left ventricle. Curves obtained during the training phase of the Daisy-chaining model are displayed in Figure 4.2. The training and validation curves of RAFT are presented in Figure 4.3. Figure 4.3a shows the evolution of $L_{RAFT,1}$, defined in Equation 3.5, while Figure 4.3b shows the evolution of $L_{RAFT,2}$, defined in Equation 3.6.

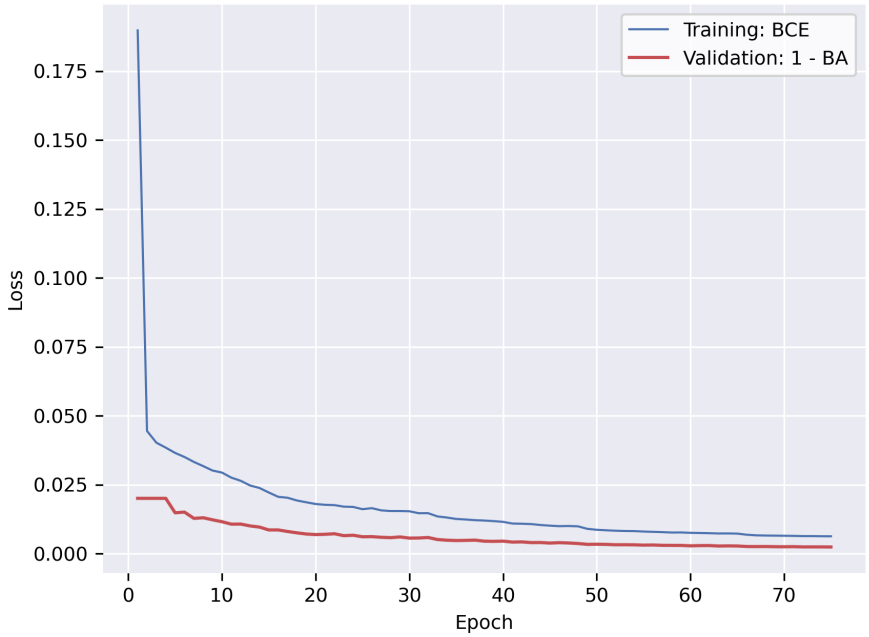


Figure 4.1: Training and validation curves of U-Net obtained during its training phase for the LV myocardial area segmentation. The training loss is the binary cross-entropy (BCE). The validation loss is 1−BA, where BA is the binary accuracy.

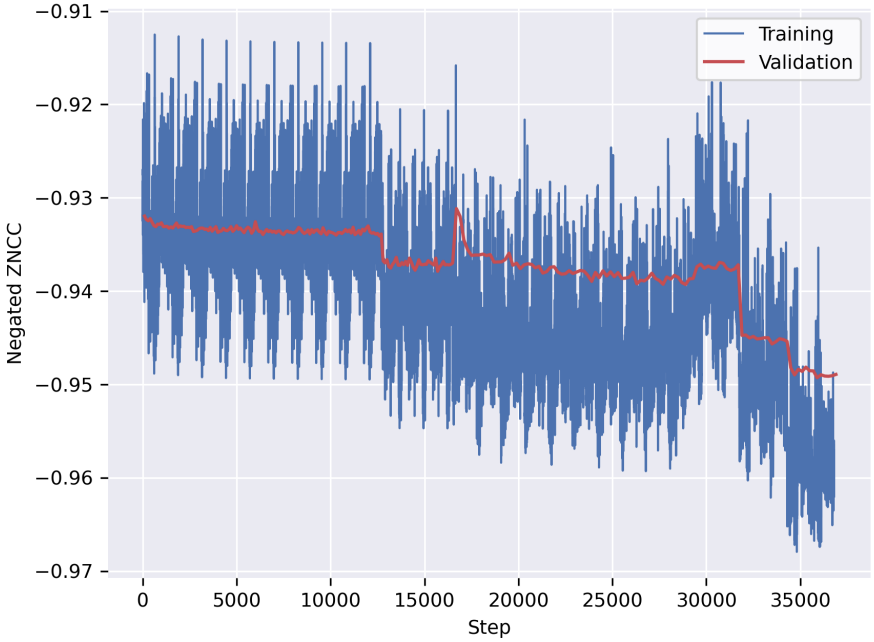
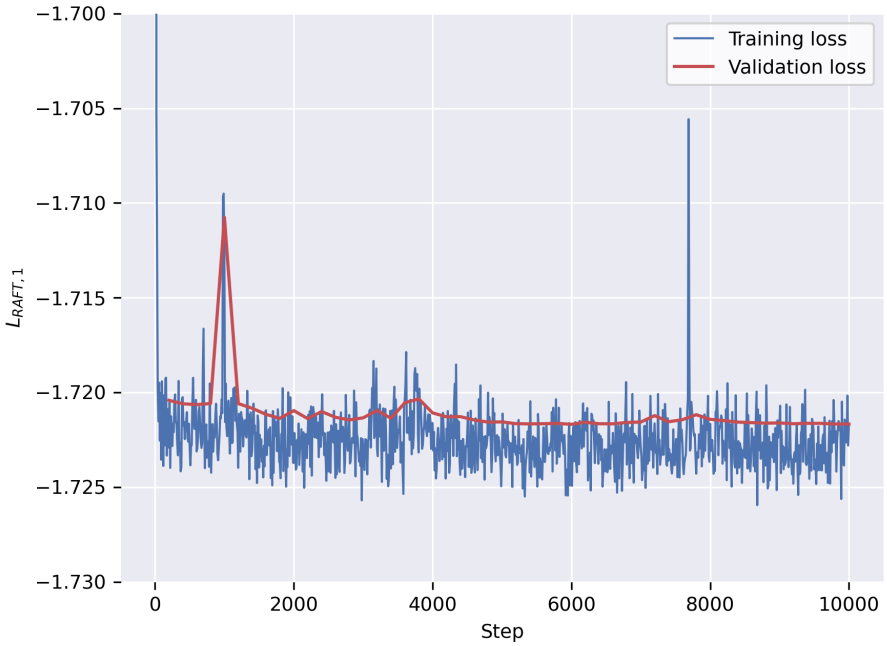
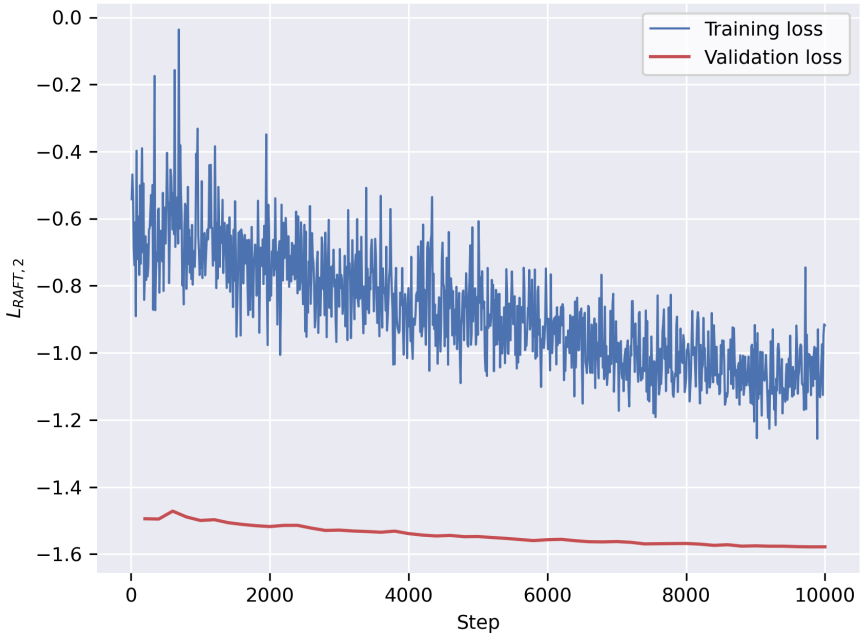


Figure 4.2: Training and validation losses of the Daisy-chaining model as a function of the training step. The negated zero-mean normalized cross-correlation is used as training and validation losses.



(a) Evolution of the RAFT training and validation loss $L_{RAFT,1}$ as a function of the training step.



(b) Evolution of the RAFT training and validation loss $L_{RAFT,2}$ as a function of the training step.

Figure 4.3: Training and validation curves of RAFT model. The curves are shown for both loss functions $L_{RAFT,1}$ and $L_{RAFT,2}$, defined in Equation 3.5 and Equation 3.6 respectively.

4.2 Point extraction tool

The point extraction tool is applied on all first frames of the B-mode sequences composing the test set. For the HR and TVI sets, the tool extracts points that are usable for tracking in respectively 50% and 57.4% of the cases. Table 4.1 details the number of successes and failures of the tool for both sets. Figure 4.4 and Figure 4.5 show 3 examples of successful and failed point extractions for the two different sets, along with the corresponding masks output by U-Net.

Table 4.1: Performances of the point extraction tool on the test set.

	HR set	TVI set
# of point extraction successes	27 (50%)	31 (57%)
# of point extraction failures	27 (50%)	23 (43%)

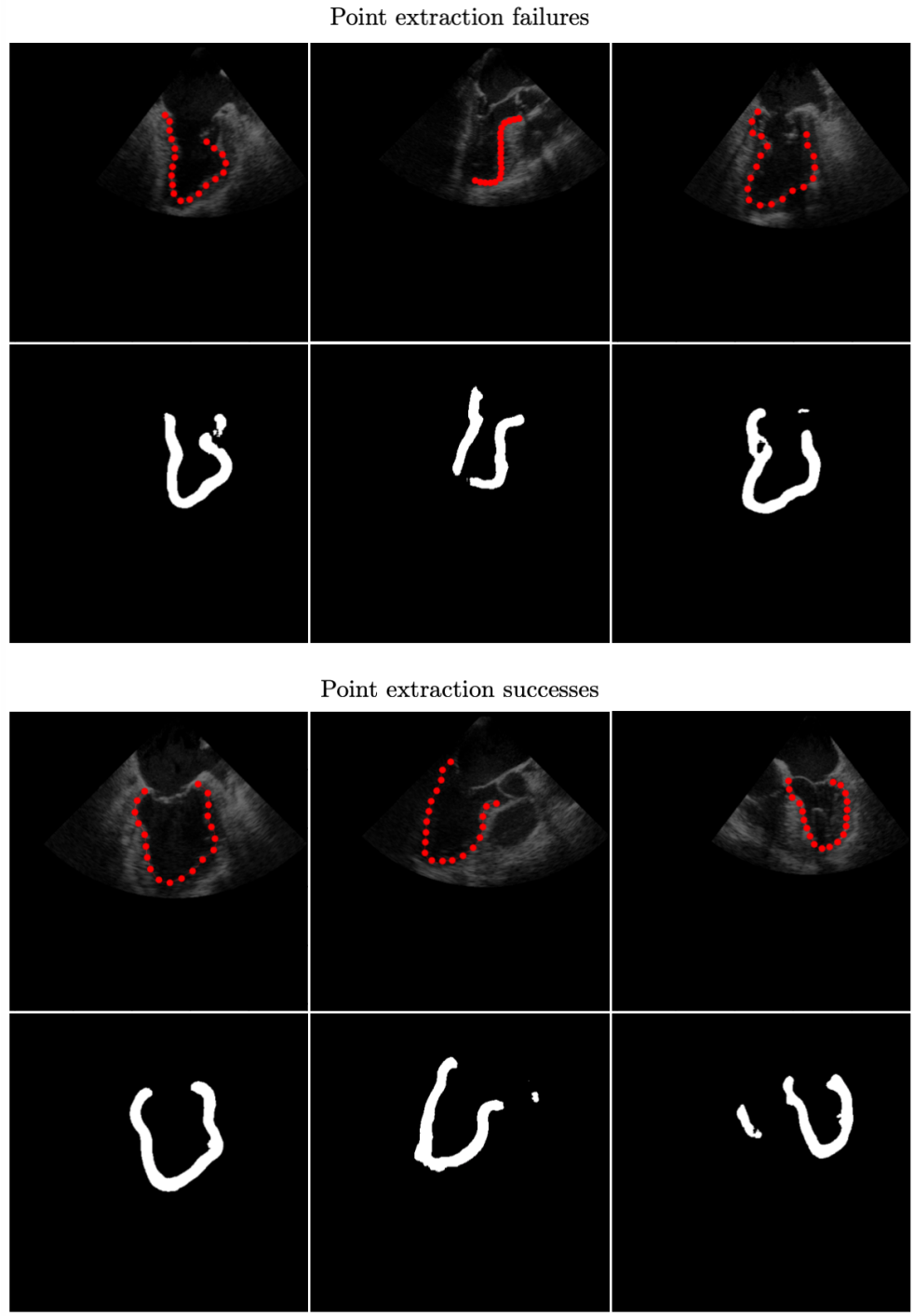


Figure 4.4: Examples of failed and successful point extractions performed by the semi-automatic tool on frames taken from the HR set. The corresponding masks output by U-Net are also shown. Extracted points are depicted in red.

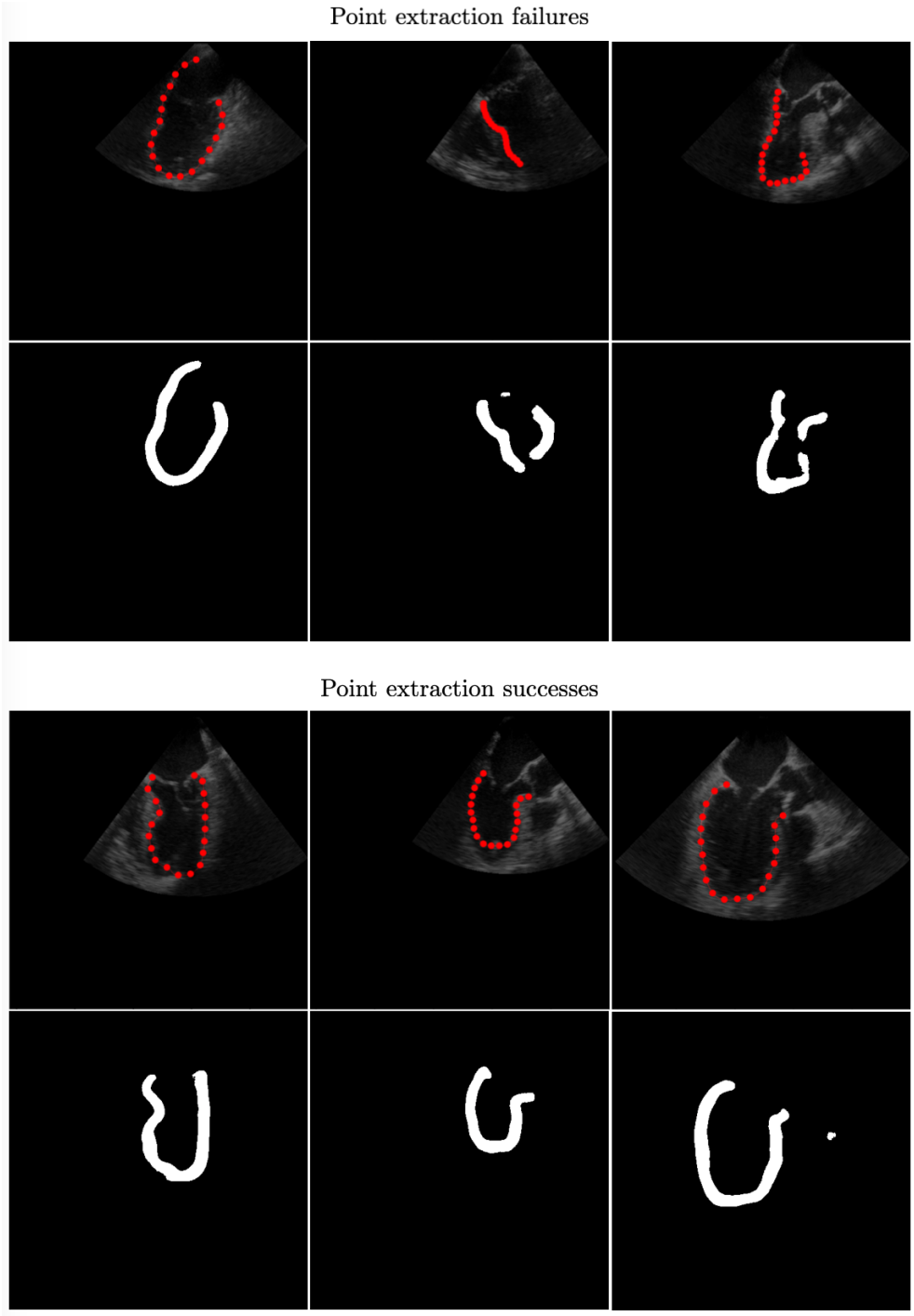


Figure 4.5: Examples of failed and successful point extractions performed by the semi-automatic tool on frames taken from the TVI set. The corresponding masks output by U-Net are also shown. Extracted points are depicted in red.

4.3 Point tracking visual inspection

The point tracking quality is assessed subjectively by visual inspection[1]. The percentage of successfully tracked points is reported for every one of the six basal segments defined in Figure 2.6, for all tried combinations of data set type, optical flow and tracking methods. The inspection focuses on the points that are chosen to be used in the strain estimation process. Results are shown in Figures 4.6 to 4.11, each figure corresponding to a basal segment. The diminutives Daisy, LK, GF, classic and Kalman respectively refer to the Daisy-chaining model, the Lucas-Kanade algorithm, the Gunnar Farnebäck algorithm, the classic tracking method and the Kalman auto-correction tracking method. Few examples of tracking videos obtained by following the procedure described in chapter 3 are available at `https://folk.ntnu.no/kiss/sgoffin/`.

[1]This inspection is performed by the author himself.

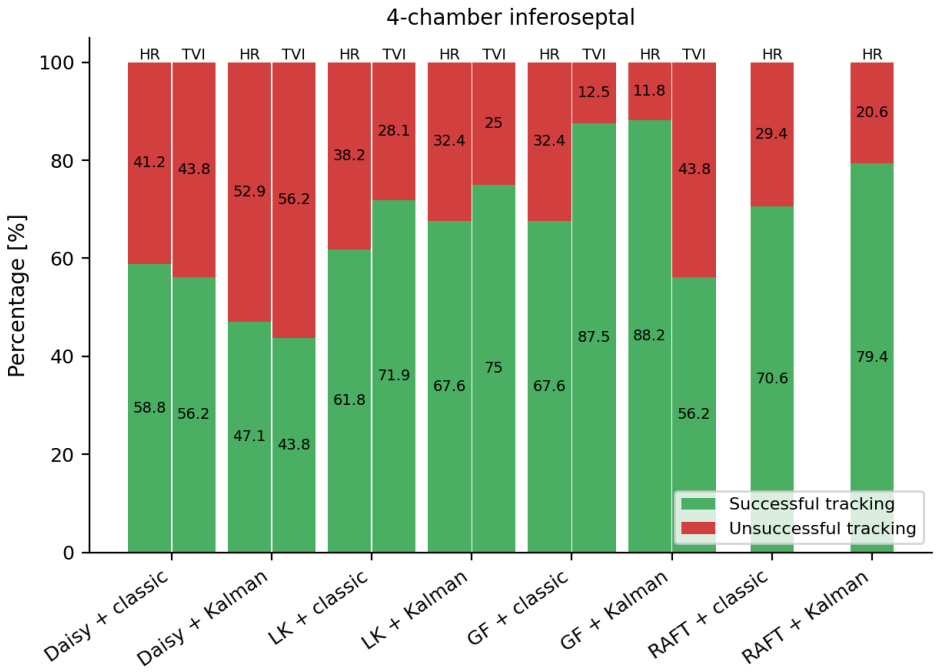


Figure 4.6: Percentage of successfully tracked points in the basal inferoseptal segment of the myocardium (4-chamber view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

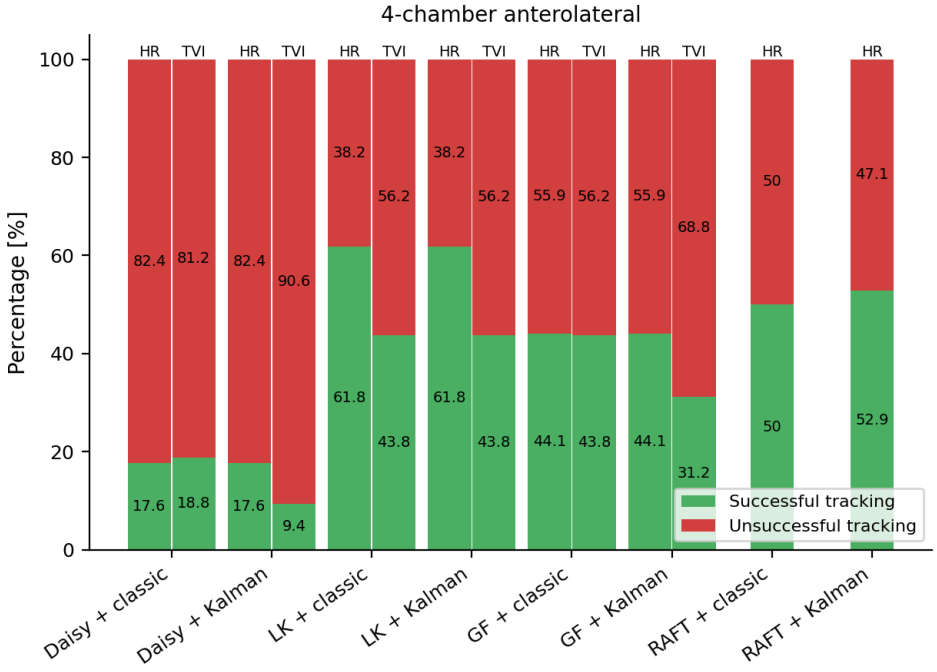


Figure 4.7: Percentage of successfully tracked points in the basal anterolateral segment of the myocardium (4-chamber view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

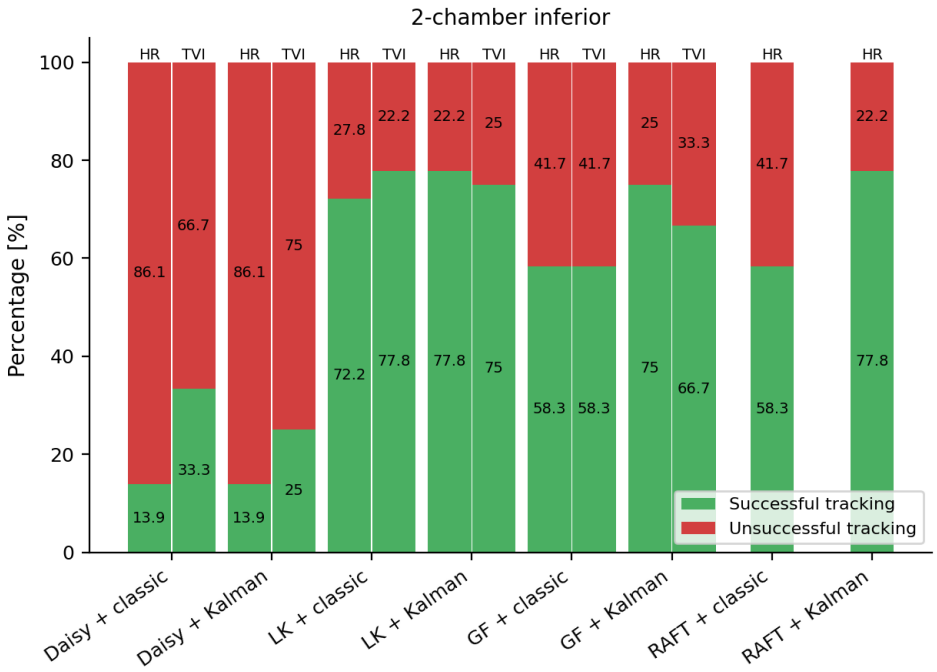


Figure 4.8: Percentage of successfully tracked points in the basal inferior segment of the myocardium (2-chamber view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

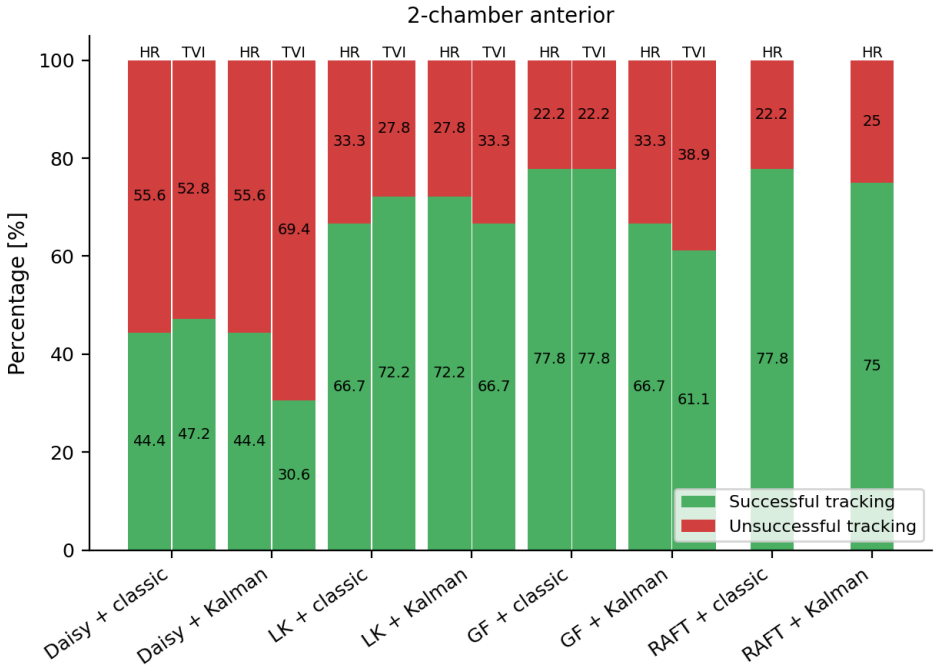


Figure 4.9: Percentage of successfully tracked points in the basal anterior segment of the myocardium (2-chamber view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

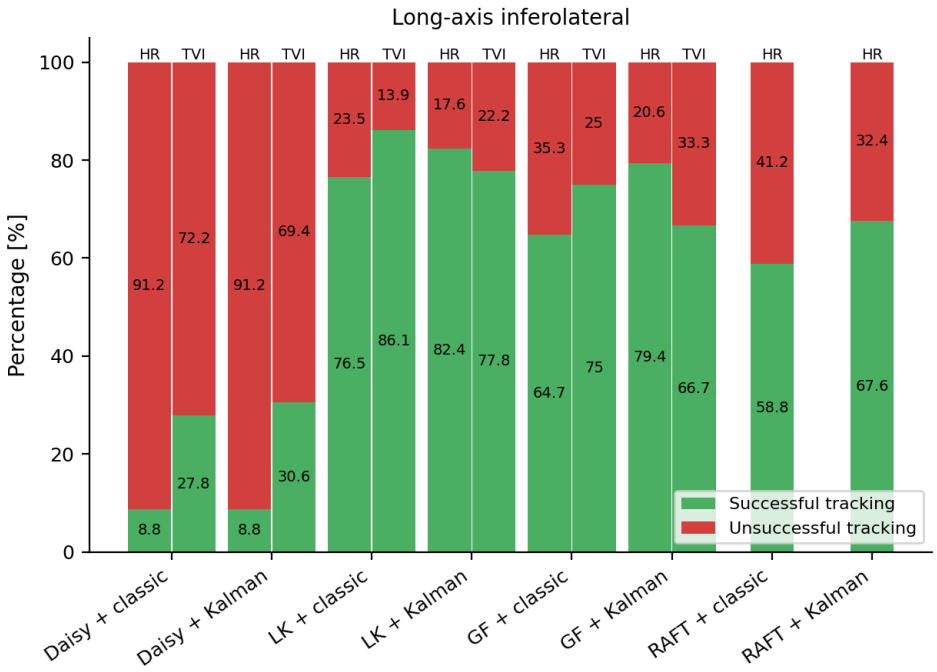


Figure 4.10: Percentage of successfully tracked points in the basal inferolateral segment of the myocardium (Apical long-axis view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

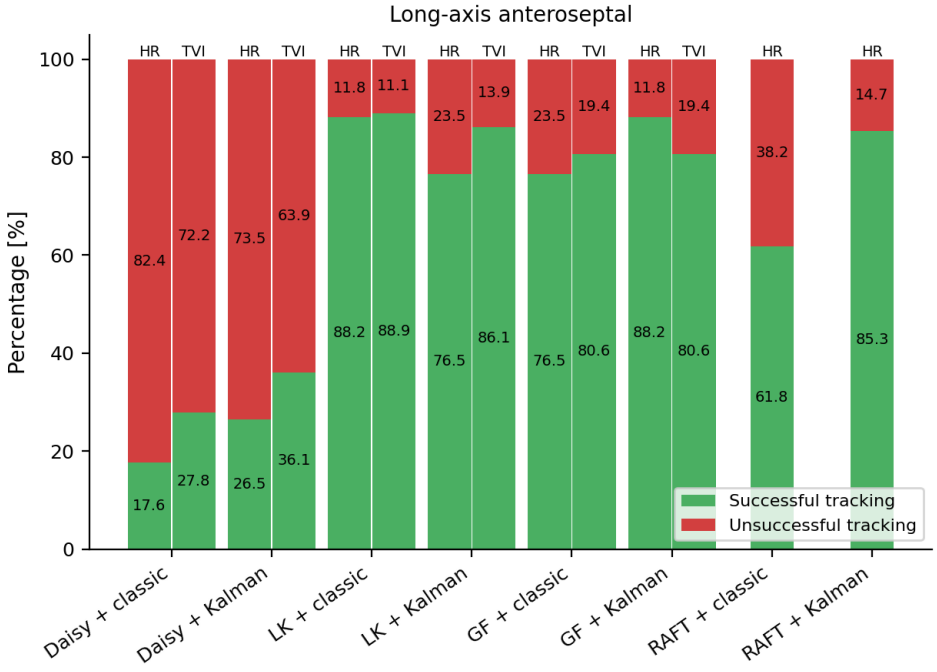


Figure 4.11: Percentage of successfully tracked points in the basal anteroseptal segment of the myocardium (Apical long-axis view). The percentage is reported for every combination of data set type (HR or TVI), optical flow model, and tracking method.

4.4 Strain estimation

The average strain in all six basal segments of each patient of the test set are computed for all tried combinations of data set type (HR or TVI), optical flow and tracking methods. Figures 4.12 to 4.25 plot the computed strain values against their corresponding ground truth. The mean and standard deviation of the absolute error between computed and ground truth strain values are reported in Table 4.2, Table 4.3 and Table 4.4. They are respectively referred as MD and σ. The Pearson correlation coefficient ρ and the variability index are also provided. Best values are displayed in green for each basal segments.

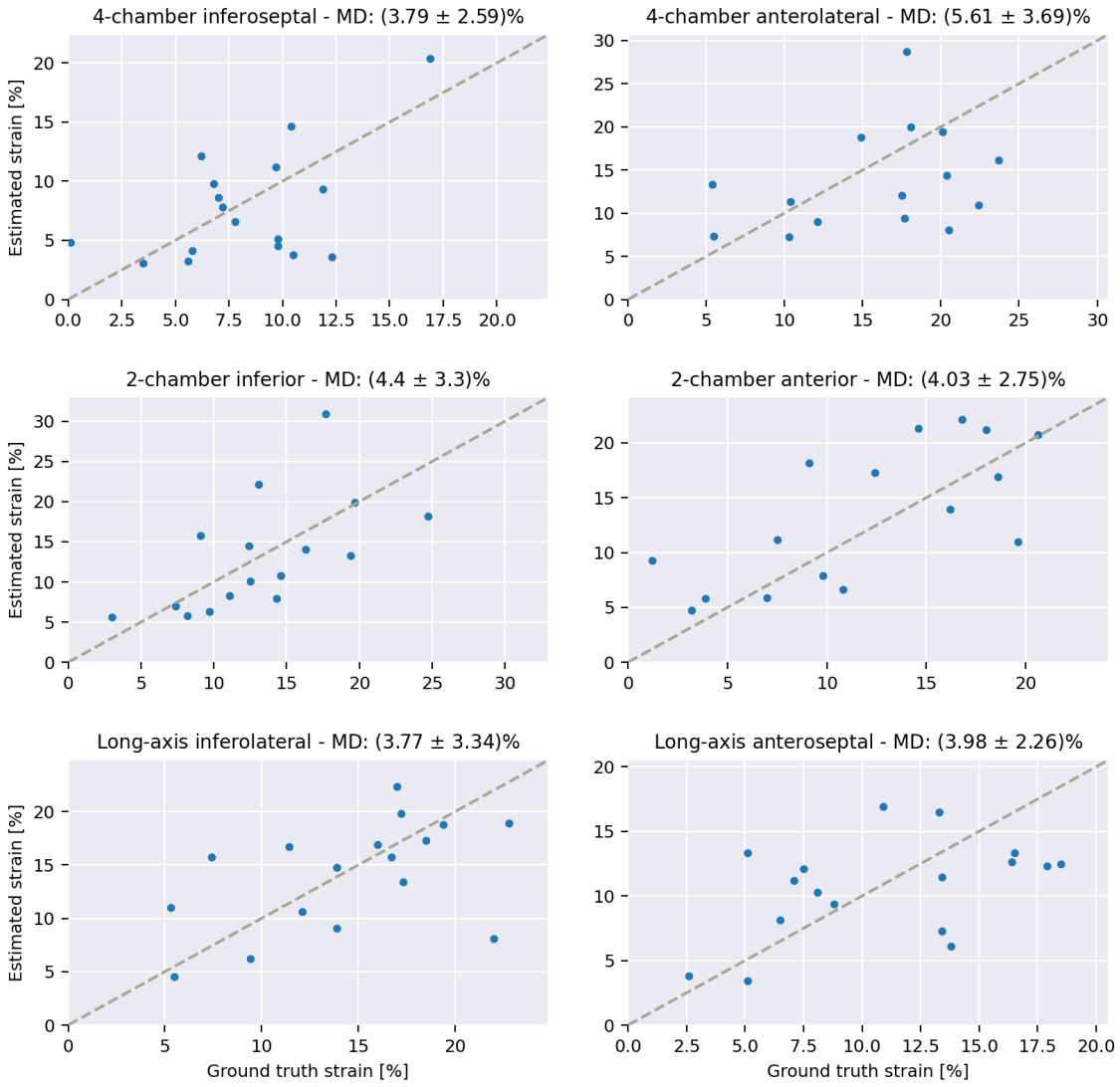


Figure 4.12: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Daisy-chaining** model with the **classic tracking** method.

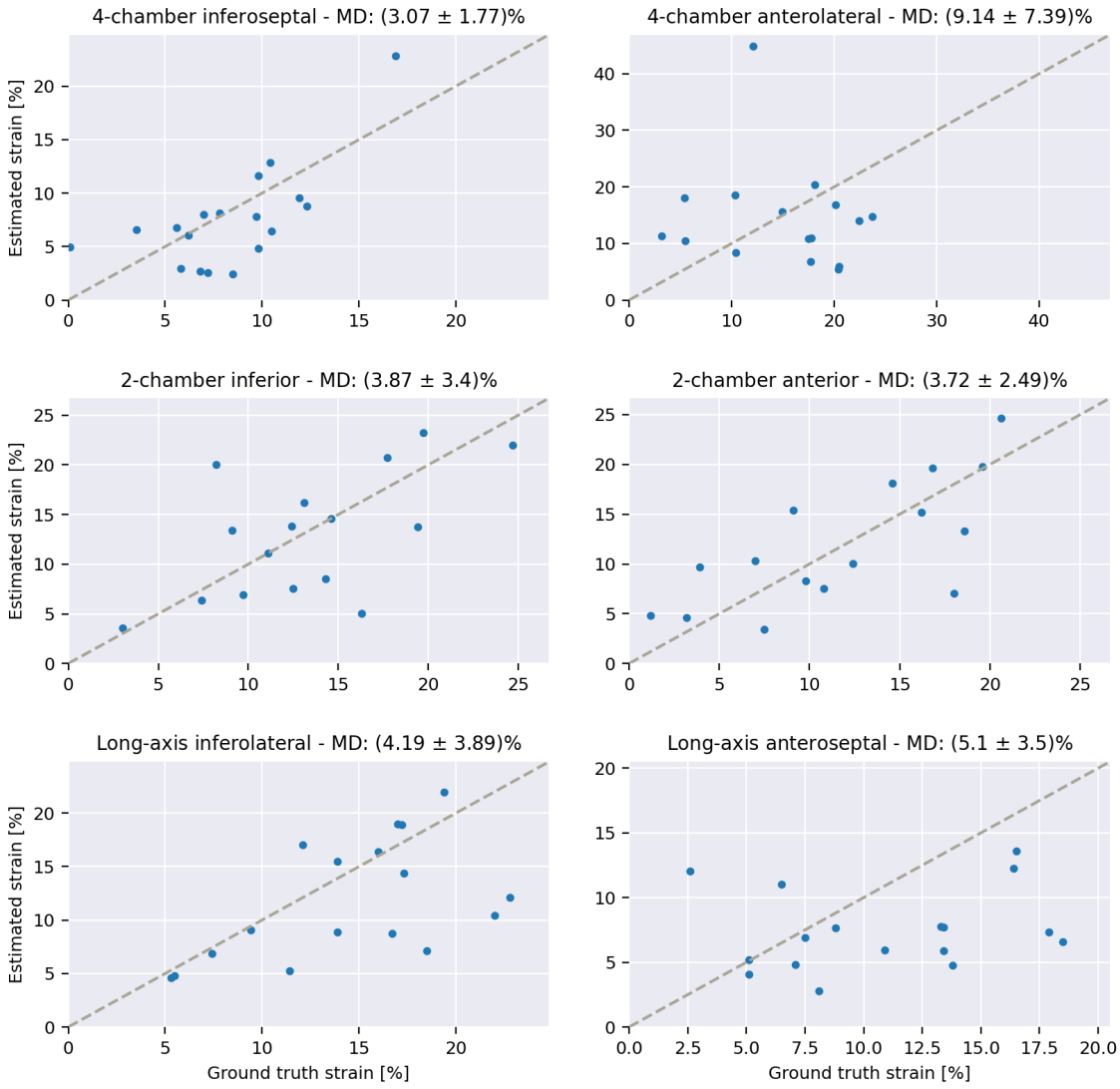


Figure 4.13: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Daisy-chaining** model with the **classic tracking** method.

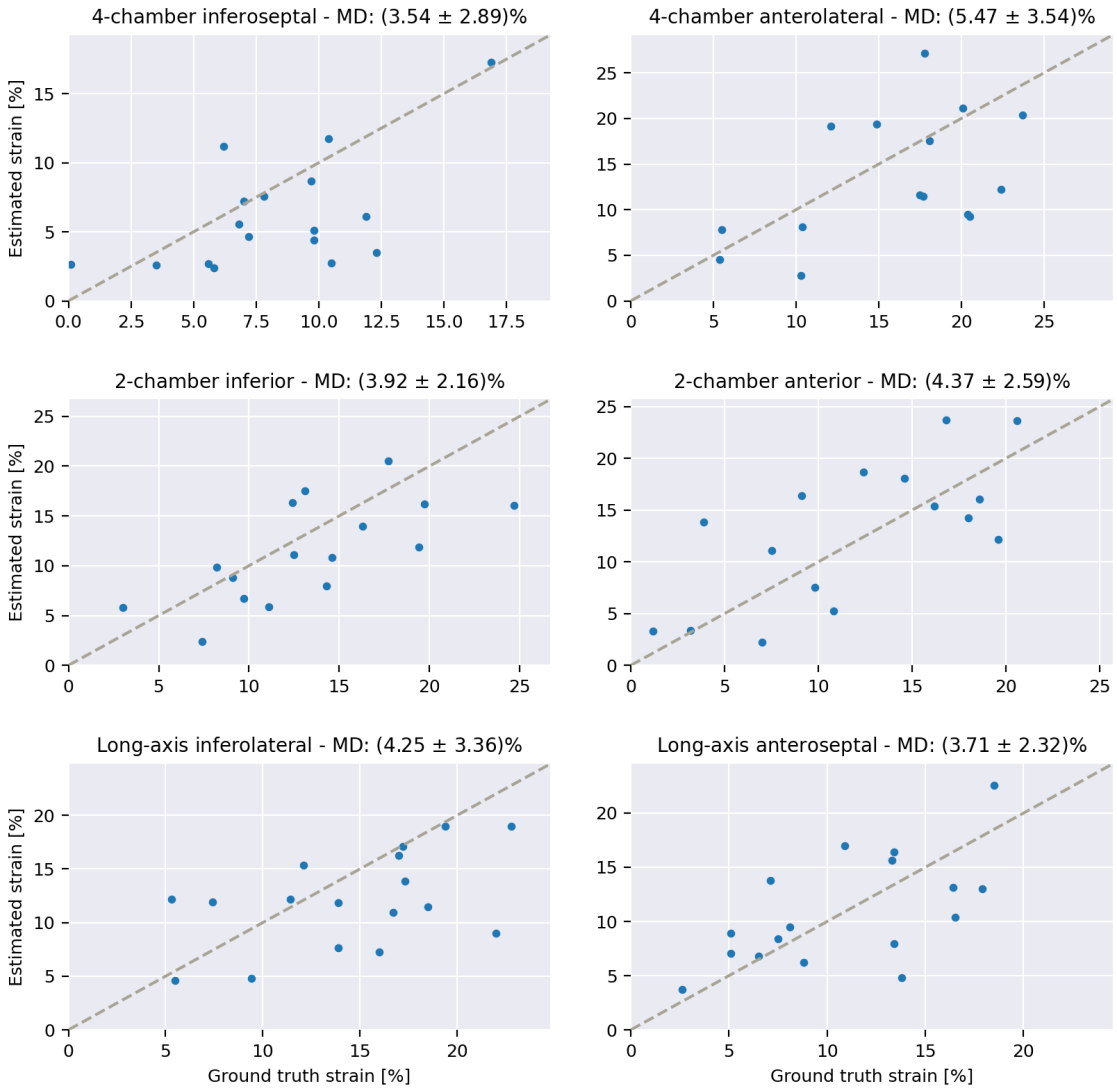


Figure 4.14: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Daisy-chaining** model with the **Kalman auto-correction tracking** method.

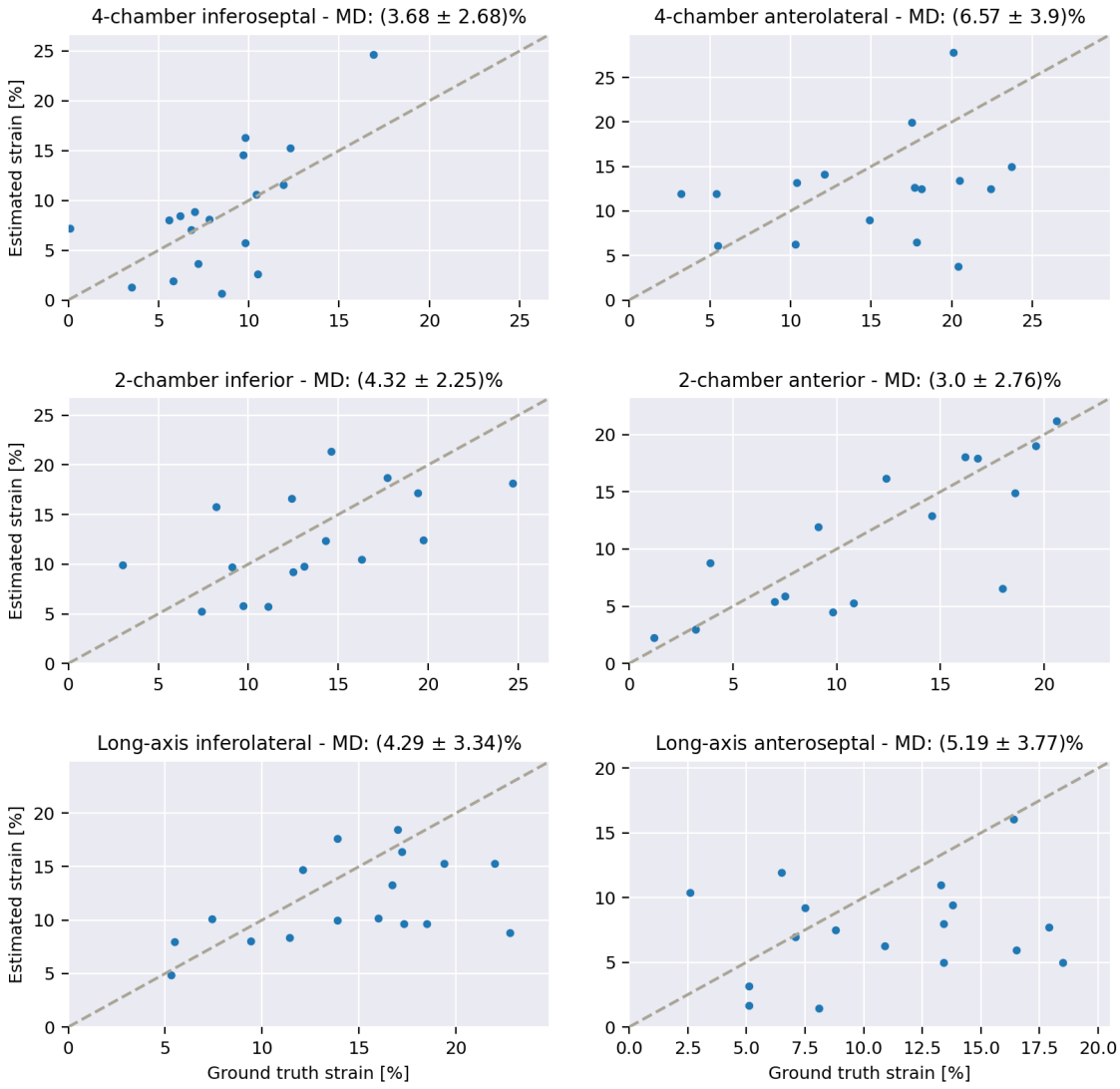


Figure 4.15: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Daisy-chaining** model with the **Kalman auto-correction tracking** method.

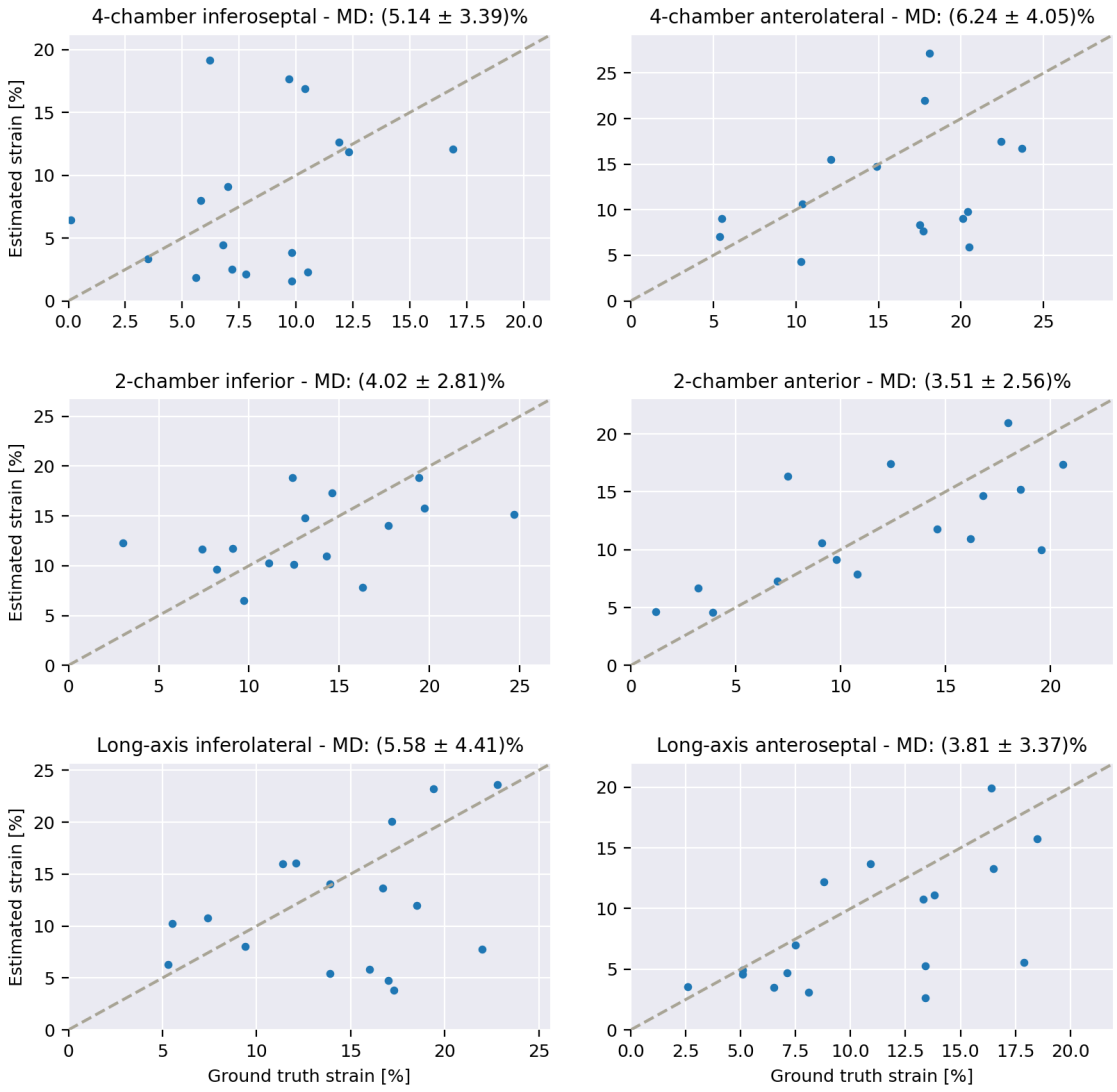


Figure 4.16: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Lucas-Kanade** algorithm with the **classic tracking** method.

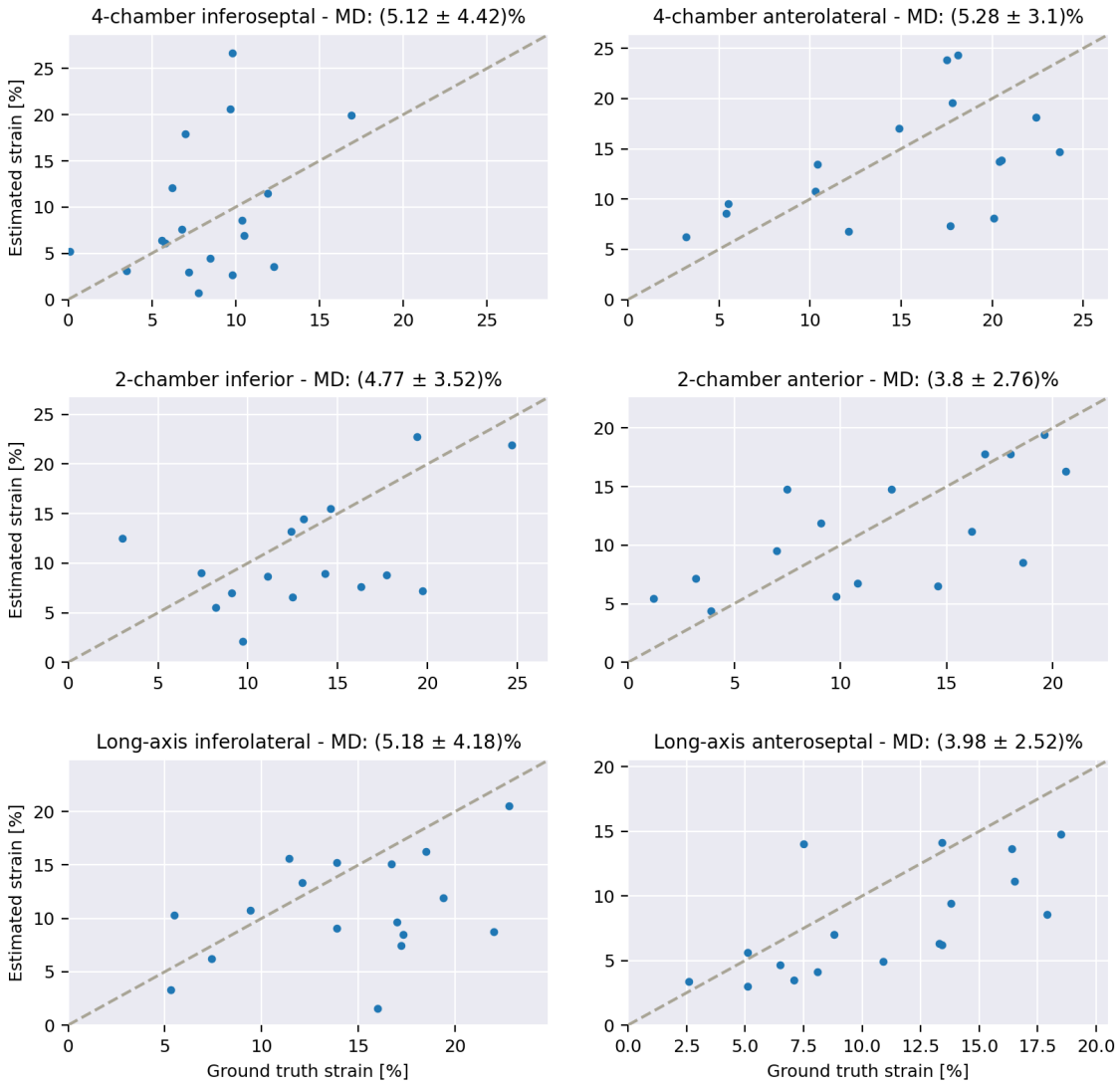


Figure 4.17: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Lucas-Kanade** algorithm with the **classic tracking** method.

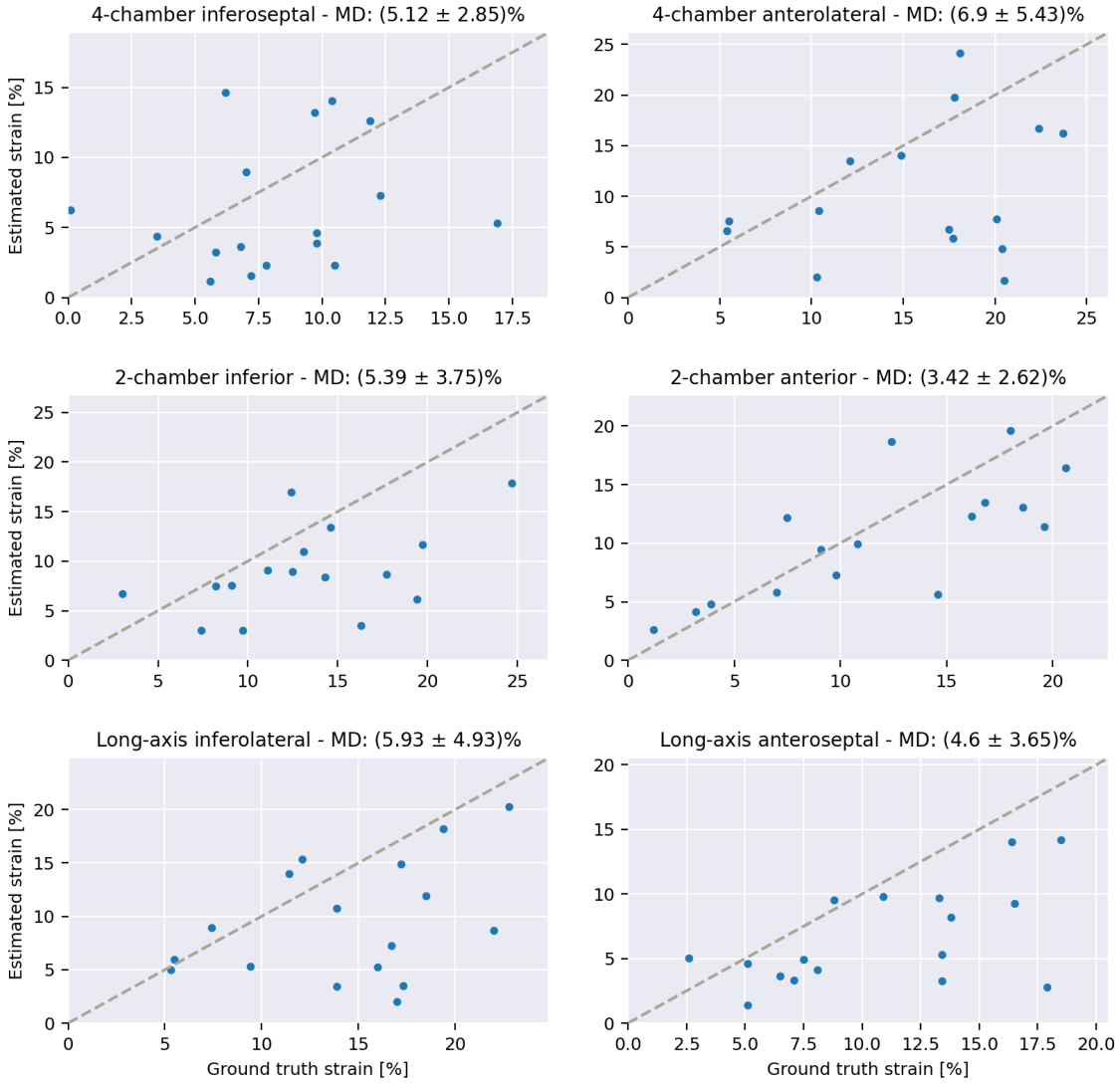


Figure 4.18: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Lucas-Kanade** algorithm with the **Kalman auto-correction tracking** method.

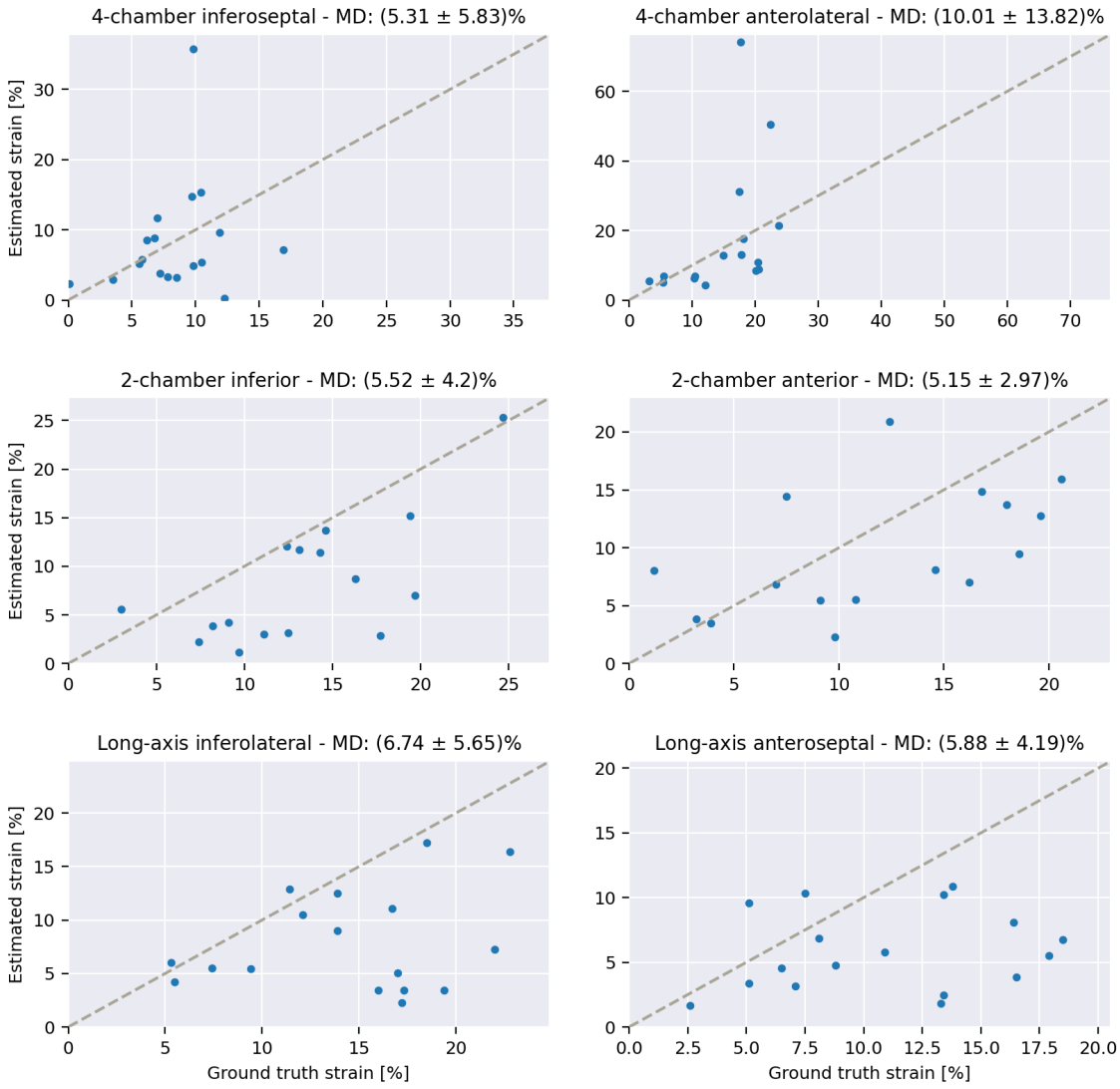


Figure 4.19: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Lucas-Kanade** algorithm with the **Kalman auto-correction tracking** method.

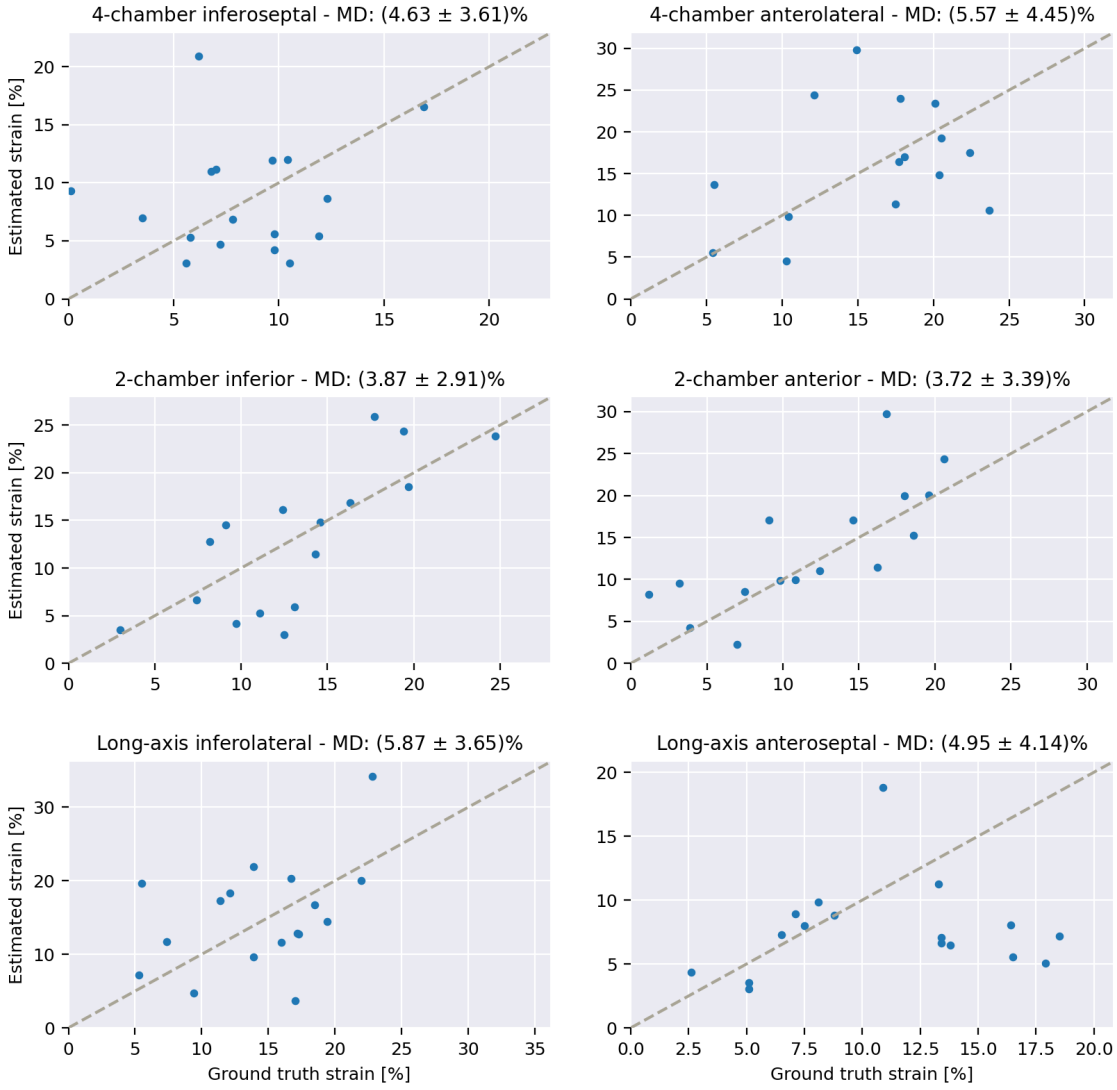


Figure 4.20: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Gunnar Farnebäck** algorithm with the **classic tracking** method.

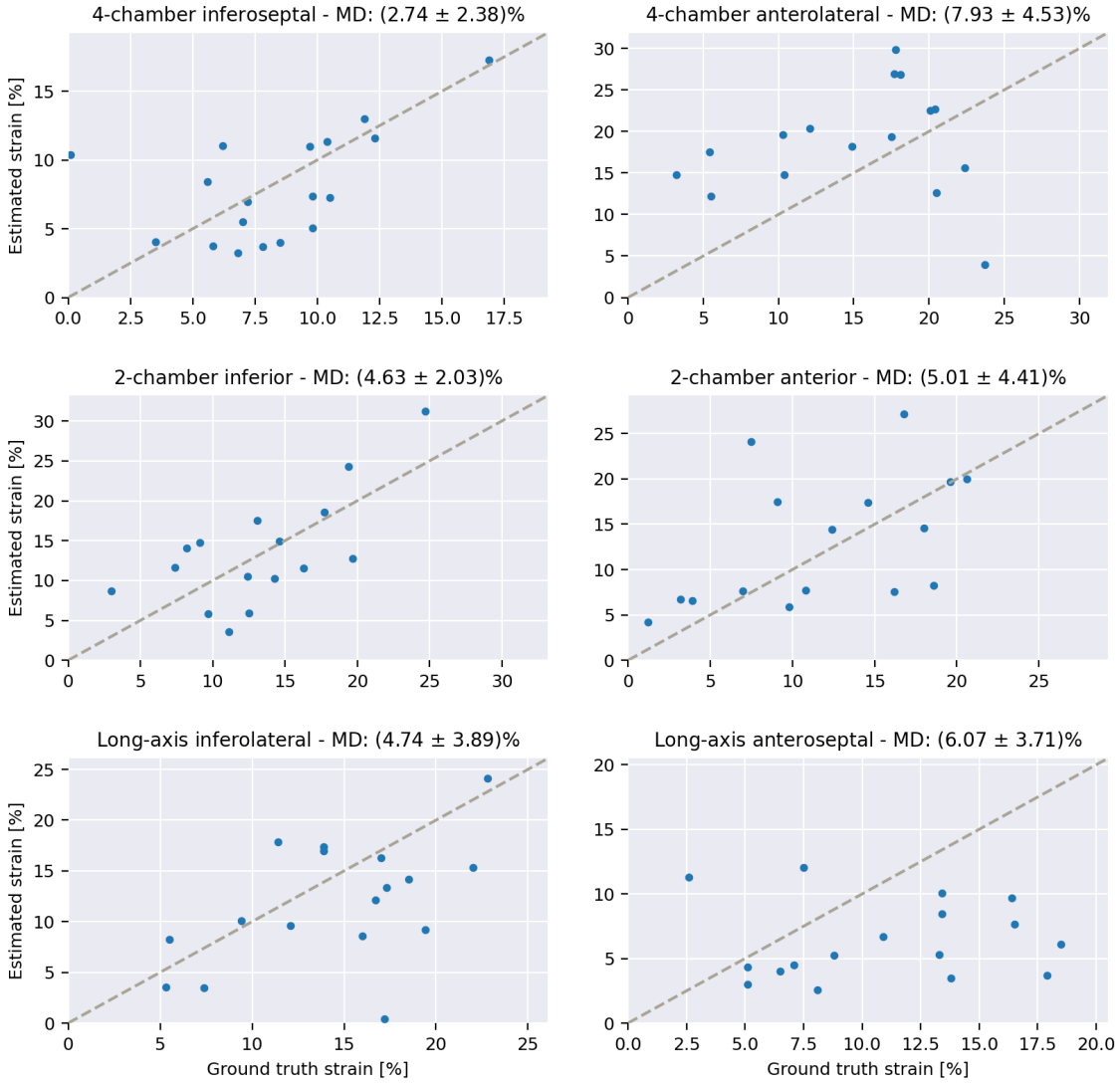


Figure 4.21: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Gunnar Farnebäck** algorithm with the **classic tracking** method.

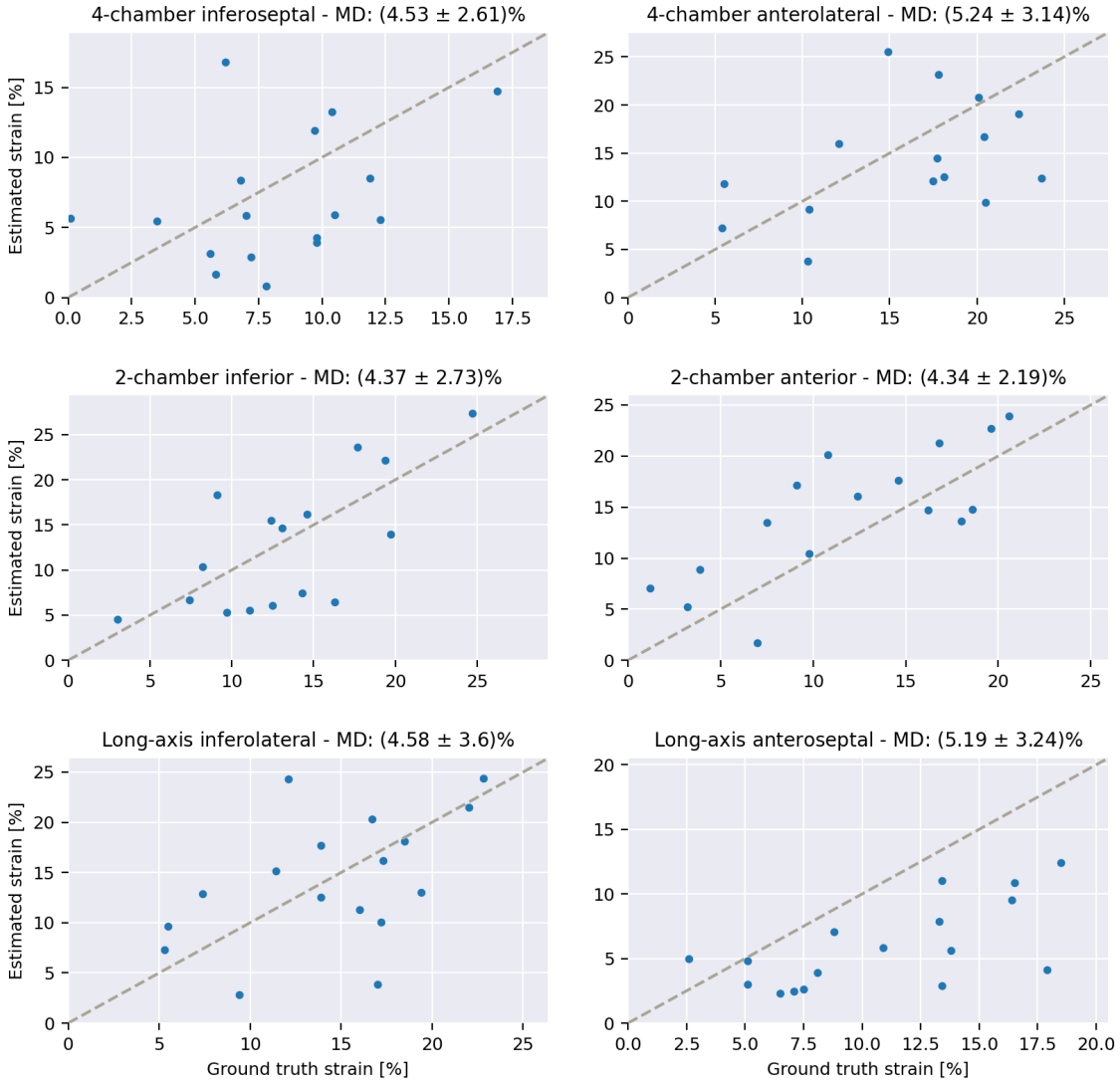


Figure 4.22: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using the **Gunnar Farnebäck** algorithm with the **Kalman auto-correction tracking** method.

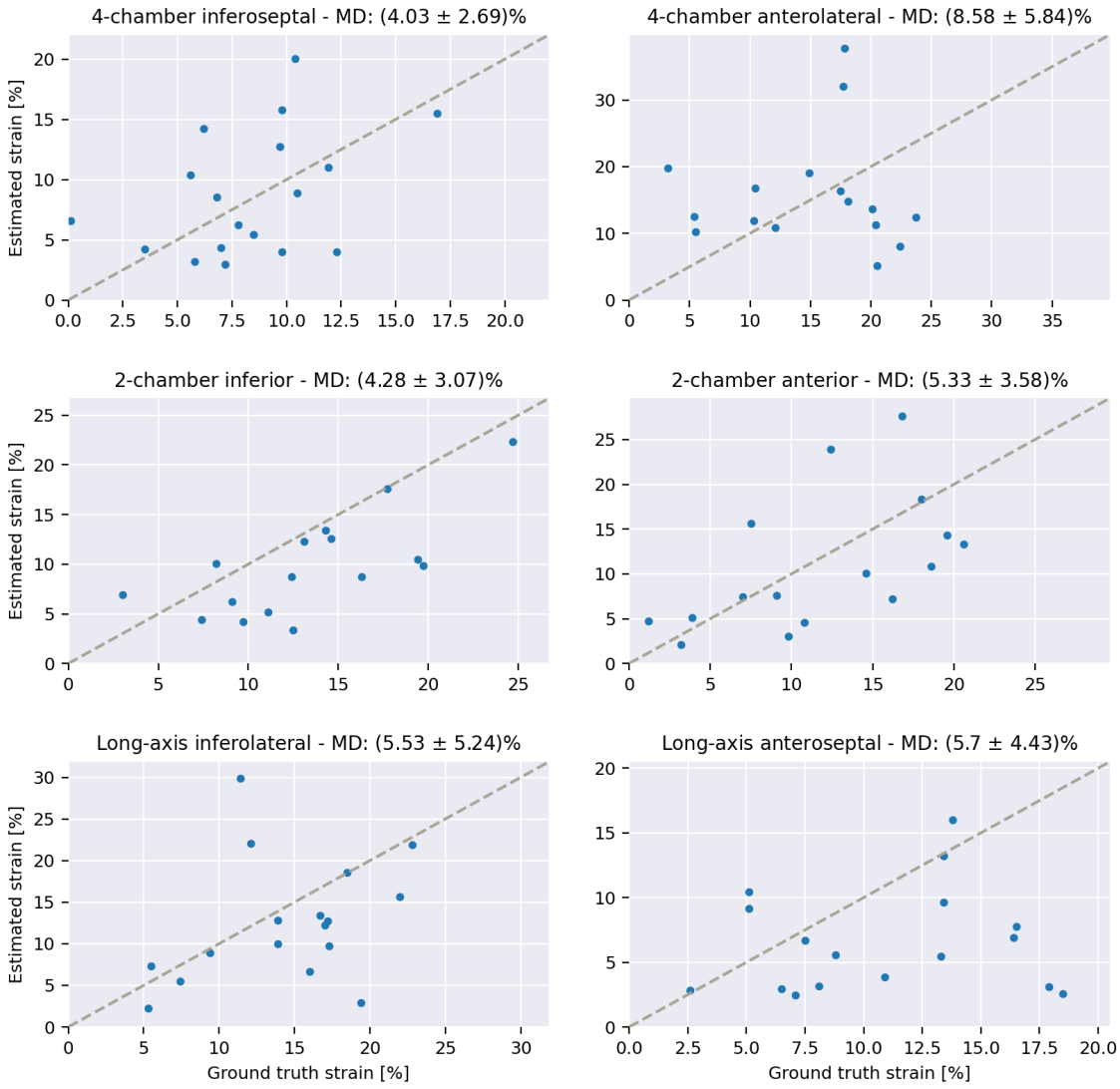


Figure 4.23: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **TVI set**, using the **Gunnar Farnebäck** algorithm with the **Kalman auto-correction tracking** method.

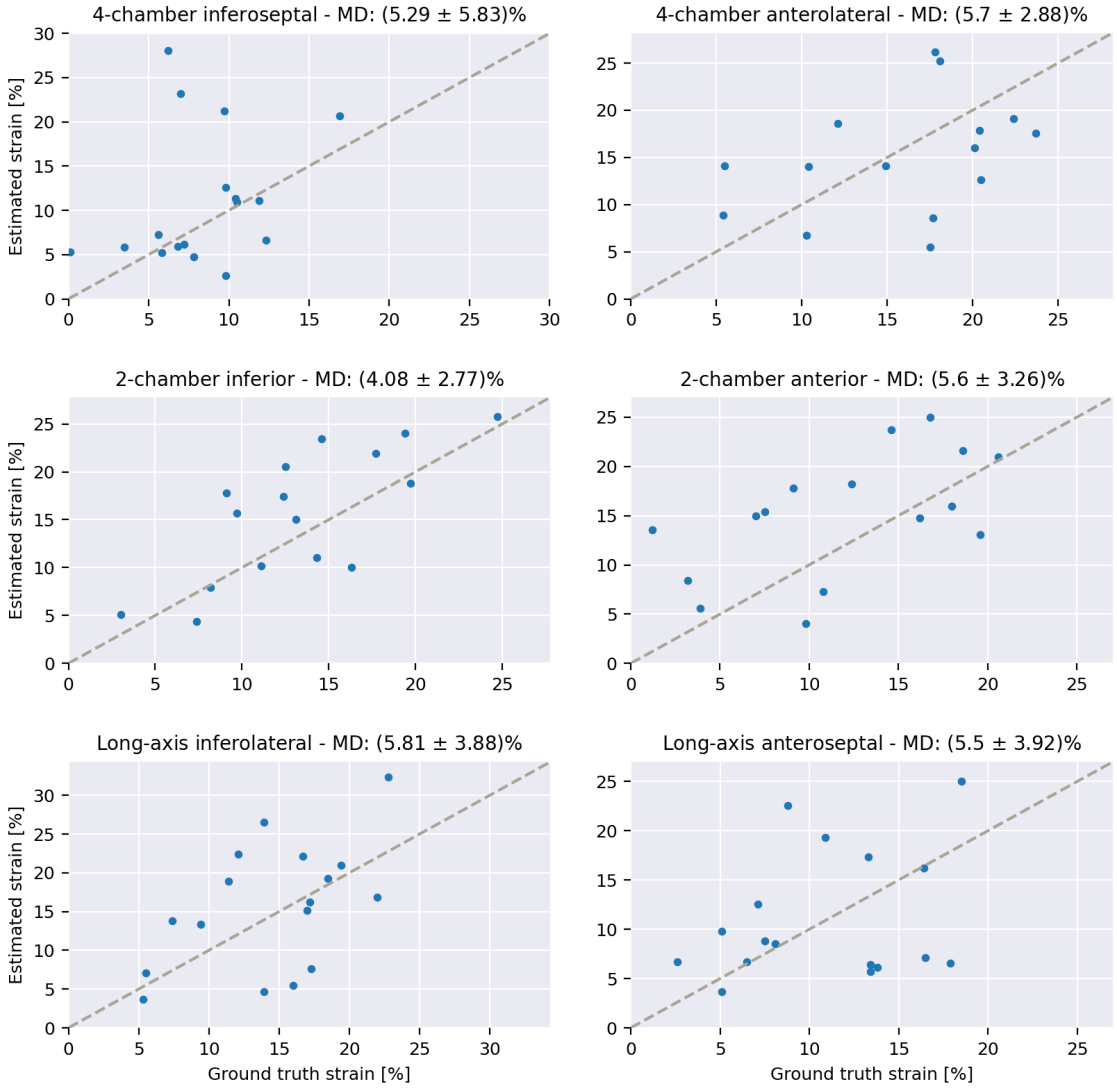


Figure 4.24: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using **RAFT** model with the **classic tracking** method.

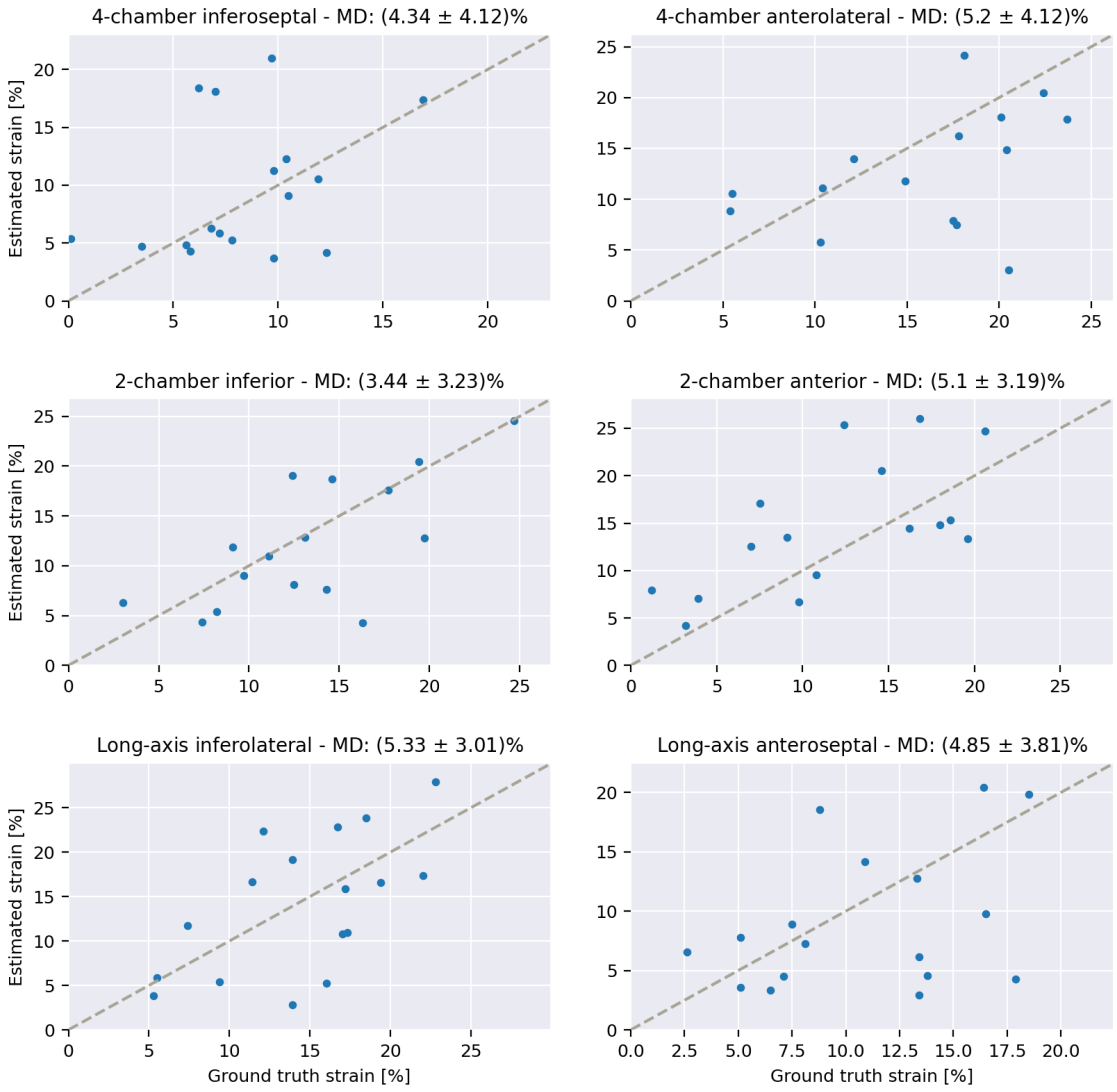


Figure 4.25: Basal strain estimates plotted against the corresponding ground truth values. Strain estimation is performed on the **HR set**, using **RAFT** model with the **Kalman auto-correction tracking** method.

Table 4.2: Comparison between ground truth and estimated strain values in percent in the 4-chamber view. MD, σ and ρ are respectively the mean absolute difference, the standard deviation and the Pearson correlation coefficient. Best results are displayed in green.

4-chamber view

Basal inferoseptal segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	3.79	2.59	0.48	0.44
Daisy + classic + TVI set	3.07	1.77	**0.68**	0.52
Daisy + Kalman + HR set	3.54	2.89	0.52	0.39
Daisy + Kalman + TVI set	3.68	2.68	0.65	0.43
LK + classic + HR set	5.14	3.39	0.28	0.38
LK + classic + TVI set	5.12	4.42	0.38	0.34
LK + Kalman + HR set	5.12	2.85	0.16	0.38
LK + Kalman + TVI set	5.31	5.83	0.21	0.52
GF + classic + HR set	4.63	3.61	0.13	0.34
GF + classic + TVI set	**2.74**	2.38	0.53	0.37
GF + Kalman + HR set	4.53	2.61	0.33	0.45
GF + Kalman + TVI set	4.03	2.69	0.41	0.33
RAFT + classic + HR set	5.29	5.83	0.26	0.41
RAFT + Kalman + HR set	4.34	4.12	0.32	**0.26**

Basal anterolateral segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	5.61	3.69	0.51	0.39
Daisy + classic + TVI set	9.14	7.39	-0.16	0.37
Daisy + Kalman + HR set	5.47	3.54	**0.62**	0.42
Daisy + Kalman + TVI set	6.57	3.9	0.28	0.54
LK + classic + HR set	6.24	4.05	0.49	**0.24**
LK + classic + TVI set	5.28	3.1	0.5	0.36
LK + Kalman + HR set	6.9	5.43	0.43	0.3
LK + Kalman + TVI set	10.01	13.82	0.44	0.38
GF + classic + HR set	5.57	4.45	0.52	0.39
GF + classic + TVI set	7.93	4.53	0.1	0.36
GF + Kalman + HR set	5.24	3.14	0.59	0.44
GF + Kalman + TVI set	8.58	5.84	-0.02	0.44
RAFT + classic + HR set	5.7	2.88	0.54	0.32
RAFT + Kalman + HR set	**5.2**	4.12	0.57	0.26

Table 4.3: Comparison between ground truth and estimated strain values in percent in the 2-chamber view. MD, σ and ρ are respectively the mean absolute difference, the standard deviation and the Pearson correlation coefficient. Best results are displayed in green.

2-chamber view

Basal inferior segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	4.4	3.3	0.61	0.35
Daisy + classic + TVI set	3.87	3.4	0.6	0.39
Daisy + Kalman + HR set	3.92	2.16	0.69	0.47
Daisy + Kalman + TVI set	4.32	2.25	0.56	0.4
LK + classic + HR set	4.02	2.81	0.44	**0.19**
LK + classic + TVI set	4.77	3.52	0.51	0.29
LK + Kalman + HR set	5.39	3.75	0.48	0.34
LK + Kalman + TVI set	5.52	4.2	0.68	0.39
GF + classic + HR set	3.87	2.91	**0.77**	0.29
GF + classic + TVI set	4.63	2.03	0.68	0.38
GF + Kalman + HR set	4.37	2.73	0.69	0.37
GF + Kalman + TVI set	4.28	3.07	0.71	0.37
RAFT + classic + HR set	4.08	2.77	0.75	0.24
RAFT + Kalman + HR set	**3.44**	3.23	0.69	0.3

Basal anterior segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	4.03	2.75	0.71	0.32
Daisy + classic + TVI set	3.72	2.49	0.73	0.43
Daisy + Kalman + HR set	4.37	2.59	0.69	0.27
Daisy + Kalman + TVI set	**3.0**	2.76	**0.79**	0.35
LK + classic + HR set	3.51	2.56	0.7	**0.2**
LK + classic + TVI set	3.8	2.76	0.66	0.26
LK + Kalman + HR set	3.42	2.62	0.74	0.23
LK + Kalman + TVI set	5.15	2.97	0.52	0.34
GF + classic + HR set	3.72	3.39	0.76	0.3
GF + classic + TVI set	5.01	4.41	0.5	0.35
GF + Kalman + HR set	4.34	2.19	0.76	0.26
GF + Kalman + TVI set	5.33	3.58	0.55	0.43
RAFT + classic + HR set	5.6	3.26	0.56	0.23
RAFT + Kalman + HR set	5.1	3.19	0.64	**0.2**

Table 4.4: Comparison between ground truth and estimated strain values in percent for the apical long-axis view. MD, σ and ρ are respectively the mean absolute difference, the standard deviation and the Pearson correlation coefficient. Best results are displayed in green.

Apical long-axis view

Basal inferolateral segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	**3.77**	3.34	0.5	0.43
Daisy + classic + TVI set	4.19	3.89	0.54	0.43
Daisy + Kalman + HR set	4.25	3.36	0.49	0.42
Daisy + Kalman + TVI set	4.29	3.34	0.49	0.38
LK + classic + HR set	5.58	4.41	0.32	**0.25**
LK + classic + TVI set	5.18	4.18	0.37	0.36
LK + Kalman + HR set	5.93	4.93	0.38	0.31
LK + Kalman + TVI set	6.74	5.65	0.25	0.43
GF + classic + HR set	5.87	3.65	0.41	0.26
GF + classic + TVI set	4.74	3.89	0.51	0.46
GF + Kalman + HR set	4.58	3.6	0.49	0.34
GF + Kalman + TVI set	5.53	5.24	0.32	0.37
RAFT + classic + HR set	5.81	3.88	0.51	0.32
RAFT + Kalman + HR set	5.33	3.01	**0.59**	0.35

Basal anteroseptal segment

	MD	σ	ρ	Variability index
Daisy + classic + HR set	3.98	2.26	0.45	0.46
Daisy + classic + TVI set	5.1	3.5	0.19	0.6
Daisy + Kalman + HR set	**3.71**	2.32	0.59	0.47
Daisy + Kalman + TVI set	5.19	3.77	0.2	0.56
LK + classic + HR set	3.81	3.37	0.6	0.36
LK + classic + TVI set	3.98	2.52	**0.67**	**0.34**
LK + Kalman + HR set	4.6	3.65	0.57	0.39
LK + Kalman + TVI set	5.88	4.19	0.17	0.5
GF + classic + HR set	4.95	4.14	0.12	0.5
GF + classic + TVI set	6.07	3.71	0.04	0.47
GF + Kalman + HR set	5.19	3.24	0.65	0.51
GF + Kalman + TVI set	5.7	4.43	0.12	0.55
RAFT + classic + HR set	5.5	3.92	0.29	0.46
RAFT + Kalman + HR set	4.85	3.81	0.38	0.38

Chapter 5

Discussion

5.1 Training curves

5.1.1 U-Net model

U-Net is trained over more than 70 epochs. The training and validation curves are shown in Figure 4.1. The training loss is the binary cross-entropy (BCE) and the validation loss is 1-BA, where BA stands for binary accuracy. The curves exhibit the expected shape: both the training and validation losses decrease as training progresses. The rate at which they evolve decreases as the training epoch grows. This is a typical behavior. Indeed, at the beginning of training, U-Net produces masks of very poor quality and learns a lot from every training batch. As training continues, the model improves and output masks become better and better. The model has to learn finer and finer details to continue improving and the amount of useful information extracted from batches drops.

Unexpectedly, the validation loss still marginally decreases after more than 70 training epochs and does not present any sign of overfitting. This is probably due to a high correlation between the validation and training sets. Indeed, data augmentation was used excessively to increase the number of annotated examples from 100 to 5000. Although different, images output by the data augmentation pipeline of U-Net still have numerous features in common with the image they are originated from. Since the training-validation split was performed after the data augmentation, images from the validation set present a relatively great similarity with some images of the training set. This prevents the binary accuracy from dropping when the model starts to overfit. To tackle this issue, the segmentation performances of U-Net are regularly assessed during training by visually inspecting the masks output for non-annotated images: the training procedure is stopped before a significant drop in the output masks quality is observed. Another solution would consists in performing the training-validation split on the manually annotated data, and then performing data augmentation independently on both sets. However, this alternative requires more annotated examples to ensure that

both training and validation sets present enough diversity to be representative populations of the U-Net input space.

5.1.2 Daisy-chaining model

The evolution of the training and validation negated ZNCC is shown in Figure 4.2. The training and validation curves exhibit an atypical shape, with sudden jumps due to the step-by-step training procedure. The sharp transitions in the validation loss occurring at steps 12700 and 16500 mark the beginning of the training of the second and third optical flow units of the model. The first, second and third optical flow units were respectively trained for 12700, 3800 and 12700 training steps. Note that each unit was actually trained until no improvement is observed. The weights giving the best validation performances were stored and used for the rest of the training procedure. Fine-tuning starts at step 29200. It greatly improves the model performances, causing the negated ZNCC to drop from -0.937 to -0.949. The periodical nature of the training curve is due to the fact that training data was not shuffled between epochs. This should not have any impact on performances.

5.1.3 Raft model

The evolution of $L_{RAFT,1}$ and $L_{RAFT,2}$ as a function of the training step is shown in Figure 4.3. $L_{RAFT,1}$ decreases really slowly compared to $L_{RAFT,2}$. This could be expected. Indeed, $L_{RAFT,1}$ mainly depends on the results of warping operations. The link between the loss and the computed flows is thus very complex from an optimization point of view. As for $L_{RAFT,2}$, it takes into account the discrepancy between the output flow and the flow computed with the Gunnar Farnebäck algorithm. This term is faster to minimize since it depends on the output flows in a more direct way.

The values of $L_{RAFT,1}$ and $L_{RAFT,2}$ computed over the training and validation sets cannot be compared since they measure the model performances in a different way. Instead, the performances of the models trained with $L_{RAFT,1}$ and $L_{RAFT,2}$ are assessed by visually inspecting the tracking achieved using the output flows and the classic tracking method on few B-mode sequences chosen randomly. The model trained with $L_{RAFT,1}$ presents slightly better results than the other. A plausible explanation is that the Gunnar Farnebäck method produces sometimes optical flows that are not accurate enough to be considered as ground truth. The model trained with $L_{RAFT,2}$ is then misled in its learning process. The model trained with $L_{RAFT,1}$ is used for the following tracking and strain estimation assessment.

5.2 Semi-automatic point extraction tool

The performances of the automatic tool developed to find suitable points to track around the left ventricle are not optimal but promising (50% and 57% of success for the HR and TVI sets). The point extraction algorithm fails when the interesting area

of the myocardium is segmented in several pieces. This phenomenon is responsible for most of the failure cases. It is the case for five of the six failure examples shown in Figure 4.4 and Figure 4.5. Too dark images seems to be another source of failure. As shown in Figure 4.5, the frame located in the top-left corner of the figure is too dark to allow U-Net to segment accurately the myocardium. Note that too dark images can also cause the interesting area to be segmented into several pieces.

A good avenue of improvement would consist in providing the tool with some information about the locations of the mitral valves and the apex. This information could be exploited to help U-Net generating masks of better quality, and more importantly, it could greatly help the thinning algorithm to extract the best centerline from a mask, even though the mask is in several pieces. Increasing the size of the training set by manually annotating more frames would also be beneficial.

5.3 Tracking visual inspection

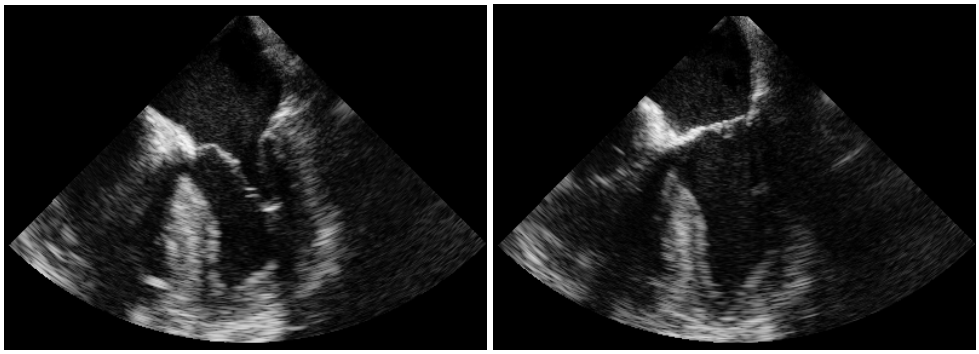
The visual inspection of tracking videos led to the results displayed in Figures 4.6 to 4.11. The influence of the optical flow algorithm is first analyzed. Then, the impact of the tracking method and test set (HR or TVI) on tracking performances is discussed. Finally, important observations about tracking performances are summed up. It is strongly recommended that the reader keeps Figures 4.6 to 4.11 at hand while reading this part of the discussion.

Influence of optical flow method

Looking at Figures 4.6 to 4.11, it can be seen that the choice of optical flow method has a big impact on performance. Consider the data set type and the tracking method as fixed. A first obvious observation is that the Daisy-chaining model leads to the worst tracking performances, whatever the basal segment that is considered. The tracking video analysis reveals that this model suffers from heavy point drifting. This phenomenon is accentuated when the tracked point is located in darker areas. Performing gamma augmentation on the B-mode frame pairs the model is trained on could help to mitigate this problem: the Daisy-chaining model would be trained to perform better in areas with a lower contrast. It is also noticed that the Daisy-chaining model fails at estimating large and fast displacements. This is a well-known limitation of coarse-to-fine optical flow models.

The tracking performances obtained using Lucas-Kanade, Gunnar Farnebäck and RAFT algorithms are good in most basal segments. Only the basal anterolateral segment (4-chamber view) seems to pause a problem: an overall drop in performances is observed, whatever the optical flow method that is used. Looking at the tracking videos, it can be noticed that the anterolateral segment often enters an obstructed area of the scanning sector[1] during cardiac cycles. This shadowing

[1] i.e. a zone where the ultrasound beam is obstructed and where tissues cannot be imaged.



(a) Anterolateral segment visible **(b)** Anterolateral segment not visible

Figure 5.1: Illustration of the shadowing problem. The basal anterolateral segment (4-chamber view) disappears from the frame during a cardiac cycle.

poblem is illustrated in Figure 5.1. This issue causes any optical flow method to fail since their functioning is based on a brightness constancy assumption underlying Equation 2.16: the brightness of a tracked pixel is assumed to be almost constant along the entire video sequence. In the other segments, these three methods estimate the points displacement pretty accurately. The few cases of failure are either caused by drift or shadowing issues. It is difficult to determine at first sight which one of the Lucas-Kanade, Gunnar Farnebäck and RAFT algorithms leads to the best tracking performances overall.

Impact of test set and tracking method

The choice of data set (HR od TVI) affects the tracking performances. Looking at Figures 4.6 to 4.11, three observations can be made:

1. When the classic tracking method is used, exploiting TVI data (i.e. using the TVI set) improves or maintains the performances for almost all optical flow methods, and all basal segments. The only three exceptions occur when the Daisy-chaining model is used in the basal inferoseptal segment (4-chamber view), and when the Lucas-Kanade and Gunnar Farnebäck methods are used in the basal anterolateral segment (4-chamber view). Using TVI data with the classic tracking method can cause an increase in the percentage of points successfully tracked of up to 20%.

2. When the Kalman auto-correction tracking method is combined to the Gunnar Farnebäck algorithm, using the TVI set degrades the tracking performances in all basal segments. The drop in the percentage of successfully tracked points can go up to 32%.

3. When the Kalman auto-correction tracking method is combined with the Daisy-chaining model or the Lucas-Kanade algorithm, using the TVI set gives mixed results. Performances are degraded in half of the segments

and improved in the other half for the Daisy-chaining model. As for the Lucas-Kanade method, performances are improved in two segments only, the performances being degraded in the others.

The degradation of the tracking performances occurring when the Kalman auto-correction method is applied on the TVI set is probably not caused by the use of TVI data in itself. It is rather due to the lower frame rate of the B-mode sequences of this set. As a reminder, the Kalman auto-correction tracking method updates the position estimate $\hat{\mathbf{x}}_{i+1}^{m}$ of point m with the fake measurement $\boldsymbol{z}_{i+1}^{m}$ defined in Equation 3.8. Since the period of time separating consecutive B-mode frames is in average two times longer in the TVI set than in the HR set, the time between two updates of $\hat{\mathbf{x}}_{i+1}^{m}$ is doubled. $\boldsymbol{z}_{i+1}^{m}$ is therefore more likely to be far from the true position of point m if points $m-1$ and/or $m+1$ start to drift. If it is the case, the estimate $\hat{\mathbf{x}}_{i+1}^{m}$ is corrected with a completely erroneous value and the Kalman auto-correction method amplifies the drift problem rather than solving it. The fact that the Lucas-Kanade method and the Daisy-chaining model present sometimes better tracking performances when the Kalman-based tracking method is applied to the TVI set could hypothetically be due to the beneficial use of TVI information.

The last observations do not mean that the Kalman auto-correction tracking method is inefficient. Indeed, Figures 4.6 to 4.11 reveal that, on the HR set, applying the Kalman auto-correction method rather than the classic method maintains or improves the tracking performances for all optical flow algorithms in almost all basal segments. The inspection of tracking videos recorded on the HR set shows that the Kalman-based tracking method helps reducing the drift problem. In few cases, it can even limit the harmful influence of shadowing. The only exceptions occur for the Daisy-chaining model in the inferoseptal segment (4-chamber view), the Gunnar Farnebäck method and RAFT model in the anterior segment (2-chamber view), and the Lucas-Kanade model in the anteroseptal segment (apical long-axis view). This observation corroborates the hypothesis that a lower frame rate causes the Kalman auto-correction tracking technique to be less efficient, or even harmful.

Visual inspection summary

The visual analysis of tracking videos revealed that:

- The Daisy-chaining model provides optical flows of poor quality overall, and achieves low tracking performances. It heavily suffers from drift and shadowing issues.
- The RAFT model and the Lucas-Kanade and Gunnar Farnebäck methods provide optical flows of good quality overall, and can be used to perform an efficient tracking. They mitigate the drift problem quite well, but are still heavily affected by shadowing.
- Integrating TVI data and exploiting the resulting displacement information (i.e. using the TVI set) is beneficial in almost all cases involving the classic

tracking method.
- Applying the Kalman auto-correction tracking method on the HR set generally mitigates the drift issue and leads to better tracking performances.
- Applying the Kalman auto-correction tracking method on the TVI set most often degrades the tracking performances. The most probable cause to this is the low frame rate of B-mode sequences composing the TVI set. Nevertheless, further analyses are still required to assure that this is not caused by the use of TVI information in itself.

The combinations of data set type, optical flow algorithm and tracking method that give the best tracking performances for each basal segment are gathered in Table 5.1.

Table 5.1: Combinations of data set type, optical flow algorithm and tracking method giving the best tracking performances for each basal segment.

Basal segment	**Best combination**	**Successful tracking**
Inferoseptal (4-CH)	HR + GF + Kalman	88.2%
Anterolateral (4-CH)	HR + LK + classic HR + LK + Kalman	61.8%
Inferior (2-CH)	TVI + LK + classic HR + LK + Kalman HR + RAFT + Kalman	77.8%
Anterior (2-CH)	HR + GF + classic TVI + GF + classic HR + RAFT + classic	77.8%
Inferolateral (LAX)	TVI + LK + classic	86.1%
Anteroseptal (LAX)	TVI + LK + classic	88.9%

5.3.1 Strain estimation

First, Figures 4.12 to 4.25 that show the estimated strain values against the ground truth values are discussed. General observations concerning the relation between the combination of data set type, optical flow method and tracking algorithm, and strain estimation performances are made. Then, the statistics gathered in Tables 4.2 to 4.4 are analyzed. It is strongly recommended that the reader keeps the appropriate figures and tables at hand while reading this part of the discussion.

Computed against ground truth strain values

Looking at Figures 4.12 to 4.25, a large spread around the diagonal is observed for all possible combinations of data set type, optical flow model and tracking method. This was expected as the strain estimation procedure developed in this thesis is tarnished by several sources of inaccuracy. Firstly, the placement of the points used in the strain estimation process defined in Equation 2.3 and the annotation of the ES and ED were not performed by a cardiologist. Secondly, the accuracy with which optical flow methods estimate displacements of cardiac tissues can be degraded by various factors. Unsuccessful tracking, mainly caused by drifting and shadowing, may introduce large variations in the estimated strain values. Moreover, ground truth values were computed using a commercial speckle tracking method. Even though speckle tracking methods are commonly utilized and commercially accessible, their output should not be considered as an absolute truth. These techniques suffer from considerable inter- and intra-observer variability, as well as inter-vendor variability. [73]

The average strain estimates obtained using the Daisy-chaining model are relatively accurate for the tracking of poor quality this optical flow method offers. It is possible that averaging strain estimates across multiple cardiac cycles mitigates the harmful effect drifting can have on the estimation process.

Comparing Figures 4.12 to 4.25, it is difficult to spot a combination of data set type, optical flow algorithm and tracking method that really outperforms the others in the strain estimation task. Strain estimation performances of a given combination really depend on the basal segment that is considered. Drawing a parallel between the combinations gathered in Table 5.1 and their corresponding strain estimation abilities, it seems that better tracking performances do not necessary lead to a better strain estimation.

Even if they are not as accurate as desired, the strain estimates produced by some combinations of data set type, optical flow algorithm and tracking method are somehow strongly correlated to the corresponding ground truth values. This is for example the case in the inferoseptal (4-chamber) and inferior (2-chamber) segments for the Gunnar Farnebäck method when it is combined to the classic tracking algorithm and applied on the TVI set (see Figure 4.21). The Lucas-Kanade and Gunnar Farnebäck methods seem to lead to an underestimation of the strain in the basal segments of the 2-chamber and apical long-axis views when they are combined to the Kalman auto-correction method and applied on the TVI set. This observation shows once again that the Kalman auto-correction method should not be applied on the TVI set.

Strain estimation statistics

The best values of MD, ρ and variability index for every basal segment are depicted in green in Tables 4.2 to 4.4. The statistics confirm that tracking and strain estimation performances are weakly correlated in the case of the methods implemented in this thesis. Indeed, the lowest mean absolute error is achieved by the Daisy-chaining model in three of the six basal segments (anterior, inferolateral and anteroseptal) despite the poorest tracking performances. The same observation is made for the Pearson correlation coefficient: the Daisy-chaining model achieves the highest correlation in three of the six basal segments (inferoseptal, anterolateral and anterior). However, the combinations that involve the Daisy-chaining model have a variability index among the highest in almost all basal segments. This indicates a lack of consistency in the Daisy-chaining strain estimates.

The correlation coefficient achieved by a combination depends quite a lot on the considered segment. It can be noticed that all combinations achieve a relatively high correlation in the inferior and anterior segments in the 2-chamber view.

5.4 Limitations of study and future work

The point extraction tool developed in this thesis is semi-automatic in the sense that it requires the user's intervention in order to select suitable points to use in the strain estimation process. This is clearly a major shortcoming of the proposed estimation pipeline that limits its use for real-time applications. Designing a model able to find automatically the points required to estimate the myocardial strain is a complex but necessary task in order to reach the end-goal of full automatization of the basal longitudinal strain estimation process in TEE. At present, the methods developed in this thesis cannot be used in real-time applications due to their long inference time. Their implementation should be optimized towards this goal while taking into account the actual hardware available in the operating theater.

The performances of the point extraction tool could be improved by training the segmentation model U-Net on a larger data set. This would require manually annotating a considerable amount of segmentation masks. Training a separate instance of U-Net for each view (4-chamber, 2-chamber and apical long-axis) would probably also improve the quality of output segmentation masks: the variation of the patterns to detect in input images would be reduced, thus decreasing the complexity of learning and allowing U-Net to be more specialized. The skeleton refining algorithm could also be made more robust against masks divided into several pieces. An interesting lead consists in performing morphological closing at both ends of oblong segmented areas in order to unify potential pieces of the desired area to segment.

Tracking performances can be improved by fine-tuning more carefully the train-

ing hyper-parameters of the Daisy-chaining and RAFT models and parameters of the Lucas-Kanade and Gunnar Farnebäck methods. The same training procedure should also be applied to the Daisy-chaining and RAFT models in order to compare their performances on the same basis. As U-Net, a separate optical flow model could be trained for each view. This would only apply to the Daisy-chaining and RAFT models since Lucas-Kanade and Gunnar Farnebäck are not trainable methods.

The unsupervised framework for optical flow methods described in subsection 2.5.1 could potentially be improved by designing a training loss that progressively goes from $L_{RAFT,2}$ to $L_{RAFT,1}$: using the Gunnar Farnebäck method to supervise the model at the beginning of its training phase could speed the convergence up. More weight would then be progressively accorded to $L_{RAFT,1}$ to smoothly switch to a completely unsupervised training. This modification still does not require optical flow ground truth.

It would be interesting to establish a benchmark procedure for a more accurate strain estimation assessment. The developed methods should be compared to recent state-of-the-art approaches, such as the FlowNet-based technique introduced by Østvik *et al.* [39]. Carrying this comparison on a same and larger test set would provide a more detailed view of the strengths and limitations of the suggested methods.

Chapter 6

Conclusion

This thesis aimed at automatizing the regional basal strain estimation procedure in transesophageal echocardiographic (TEE) images. Novel approaches were suggested, consisting of myocardial point extraction and tracking, and strain computation. A semi-automatic tool was developed to extract interesting myocardial points to track. Myocardial tissues displacement was estimated using optical flow methods. Four models were experimented, including two deep learning-based models trained using an adapted version of the Deep Learning Framework for Unsupervised Affine and Deformable Image Registration initially suggested by Vos *et al.* [40, 41]. The integration of tissue velocity imaging (TVI) data and a novel tracking method based on Kalman filtering were proposed as attempts of improvement of the motion estimation and tracking processes.

Training and testing data consisted in unselected transesophageal echocardiographic images of the 4-chamber, 2-chamber, and apical long-axis views of the heart. The remarkably good tracking performances obtained in five of the six basal segments using the RAFT, Lucas-Kanade and Gunnar Farnebäck optical flow algorithms proved that optical flow-based motion estimation is highly suitable for frame to frame tracking on TEE images. Experiments showed that the integration of TVI data with the classic tracking method improves in most cases the quality of optical flow estimation. The Kalman filtering-based tracking method proposed in this work also helped to reduce the drift problem on high frame rate B-mode sequences.

The segmentation model U-Net was successfully combined to a custom thinning algorithm in order to create a myocardial point extraction tool. Major improvements can still be made but decent performances were achieved, showing that myocardium segmentation can be used to extract the points necessary to the strain estimation process.

Strain estimation performances did not reflect the good tracking quality achieved with three of the optical flow methods experimented in this thesis. However, strain estimates were highly correlated to ground truth values, especially in the inferior

and anterior basal segments. Despite these promising results, the suggested methods are still affected by multiple sources of error that will need to be tackled in order to reach a strain estimation accuracy comparable to expert measurements.

Strain estimation from TEE images is a challenging task. However, this thesis proved that optical flow-based methods can be used to perform accurate tracking of myocardial tissues. This constitutes a step closer to the full automatization of the regional basal strain estimation procedure.

Bibliography

[1] S. Zaunseder, M. Riedl, J. Kurths, H. Malberg, R. Bauernschmitt and N. Wessel, 'Impact of cardiac surgery on the autonomic cardiovascular function,' *Journal of Computational Surgery*, vol. 1, no. 1, p. 9, 2014.

[2] P. P. Soares, A. M. Moreno, S. L. Cravo and A. C. Nóbrega, 'Coronary artery bypass surgery and longitudinal evaluation of the autonomic cardiovascular function,' *Critical Care*, vol. 9, no. 2, R124, 2005.

[3] B. Medić, 'The role of autonomic control in cardiovascular system: Summary of basic principles,' *Medicinski podmladak*, vol. 67, no. 1, pp. 14–18, 2016.

[4] D. C. Warltier, J. G. Laffey, J. F. Boylan and D. C. Cheng, 'The systemic inflammatory response to cardiac surgeryimplications for the anesthesiologist,' *Anesthesiology: The Journal of the American Society of Anesthesiologists*, vol. 97, no. 1, pp. 215–252, 2002.

[5] L. L. Creswell, J. C. Alexander Jr, T. B. Ferguson Jr, A. Lisbon and L. A. Fleisher, 'Intraoperative interventions: American college of chest physicians guidelines for the prevention and management of postoperative atrial fibrillation after cardiac surgery,' *Chest*, vol. 128, no. 2, 28S–35S, 2005.

[6] F. Jardin and A. Vieillard-Baron, 'Monitoring of right-sided heart function,' *Current Opinion in Critical Care*, vol. 11, no. 3, pp. 271–279, 2005.

[7] I. Balasingham, F. Helri and R. Noormohammadi, *Signaling and communication in biological systems: A compendium for the course ttt23*, 2020.

[8] C. M. Otto, *Textbook of clinical echocardiography*. Elsevier Health Sciences, 2013.

[9] M. Jensen, E. Sloth, K. Larsen and M. Schmidt, 'Transthoracic echocardiography for cardiopulmonary monitoring in intensive care,' *European Journal of Anaesthesiology| EJA*, vol. 21, no. 9, pp. 700–707, 2004.

[10] J. B. Seward, B. K. Khanderia, J. K. OH, M. D. Abel, R. W. Hughes Jr, W. D. Edwards, B. A. Nichols, W. K. Freeman and A. J. Tajik, 'Transesophageal echocardiography: Technique, anatomic correlations, implementation, and clinical applications,' in *Mayo Clinic Proceedings*, Elsevier, vol. 63, 1988, pp. 649–680.

[11] J. N. Hilberath, D. A. Oakes, S. K. Shernan, B. E. Bulwer, M. N. D'Ambra and H. K. Eltzschig, 'Safety of transesophageal echocardiography,' *Journal of the American Society of Echocardiography*, vol. 23, no. 11, pp. 1115–1127, 2010, ISSN: 0894-7317. DOI: https://doi.org/10.1016/j.echo.2010.08.013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0894731710006930.

[12] W. G. Daniel and A. Mügge, 'Transesophageal echocardiography,' *New England Journal of Medicine*, vol. 332, no. 19, pp. 1268–1280, 1995.

[13] C. L. Reichert, C. A. Visser, R. B. van den Brink, J. J. Koolen, H. B. van Wezel, A. C. Moulijn and A. J. Dunning, 'Prognostic value of biventricular function in hypotensive patients after cardiac surgery as assessed by transesophageal echocardiography,' *Journal of cardiothoracic and vascular anesthesia*, vol. 6, no. 4, pp. 429–432, 1992.

[14] H. Dalen, A. Thorstensen, S. A. Aase, C. B. Ingul, H. Torp, L. J. Vatten and A. Stoylen, 'Segmental and global longitudinal strain and strain rate based on echocardiography of 1266 healthy individuals: The hunt study in norway,' *European Journal of Echocardiography*, vol. 11, no. 2, pp. 176–183, 2010.

[15] G. J. Tortora and S. R. Grabowski, *Principes d'anatomie et de physiologie*. De Boeck Supérieur, 2001.

[16] V. Delgado, S. A. Mollema, C. Ypenburg, L. F. Tops, E. E. van der Wall, M. J. Schalij and J. J. Bax, 'Relation between global left ventricular longitudinal strain assessed with novel automated function imaging and biplane left ventricular ejection fraction in patients with coronary artery disease,' *Journal of the American Society of Echocardiography*, vol. 21, no. 11, pp. 1244–1250, 2008.

[17] J. P. Curtis, S. I. Sokol, Y. Wang, S. S. Rathore, D. T. Ko, F. Jadbabaie, E. L. Portnay, S. J. Marshalko, M. J. Radford and H. M. Krumholz, 'The association of left ventricular ejection fraction, mortality, and cause of death in stable outpatients with heart failure,' *Journal of the American College of Cardiology*, vol. 42, no. 4, pp. 736–742, 2003.

[18] P. R. Marantz, J. N. Tobin, S. Wassertheil-Smoller, R. M. Steingart, J. P. Wexler, N. Budner, L. Lense and J. Wachspress, 'The relationship between left ventricular systolic function and congestive heart failure diagnosed by clinical criteria.,' *Circulation*, vol. 77, no. 3, pp. 607–612, 1988.

[19] J. Gorcsan and H. Tanaka, 'Echocardiographic assessment of myocardial strain,' *Journal of the American College of Cardiology*, vol. 58, no. 14, pp. 1401–1413, 2011.

[20] T. P. Abraham, V. L. Dimaano and H.-Y. Liang, 'Role of tissue doppler and strain echocardiography in current clinical practice,' *Circulation*, vol. 116, no. 22, pp. 2597–2609, 2007.

[21] D. Rappaport, D. Adam, P. Lysyansky and S. Riesner, 'Assessment of myocardial regional strain and strain rate by tissue tracking in b-mode echocardiograms,' *Ultrasound in medicine & biology*, vol. 32, no. 8, pp. 1181–1192, 2006.

[22] T. Haukom, E. A. R. Berg, S. Aakhus and G. H. Kiss, 'Basal strain estimation in transesophageal echocardiography (tee) using deep learning based unsupervised deformable image registration,' in *2019 IEEE International Ultrasonics Symposium (IUS)*, IEEE, 2019, pp. 1421–1424.

[23] B. Heyde, R. Jasaityte, D. Barbosa, V. Robesyn, S. Bouchez, P. Wouters, F. Maes, P. Claus and J. D'hooge, 'Elastic image registration versus speckle tracking for 2-d myocardial motion estimation: A direct comparison in vivo,' *IEEE transactions on medical imaging*, vol. 32, no. 2, pp. 449–459, 2012.

[24] D. I. Barnea and H. F. Silverman, 'A class of algorithms for fast digital image registration,' *IEEE transactions on Computers*, vol. 100, no. 2, pp. 179–186, 1972.

[25] B. D. Lucas, T. Kanade *et al.*, 'An iterative image registration technique with an application to stereo vision,' Vancouver, British Columbia, 1981.

[26] M. J. Black and P. Anandan, 'A framework for the robust estimation of optical flow,' in *1993 (4th) International Conference on Computer Vision*, IEEE, 1993, pp. 231–236.

[27] G. Farnebäck, 'Two-frame motion estimation based on polynomial expansion,' in *Scandinavian conference on Image analysis*, Springer, 2003, pp. 363–370.

[28] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers and T. Brox, 'Flownet: Learning optical flow with convolutional networks,' in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[29] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy and T. Brox, 'Flownet 2.0: Evolution of optical flow estimation with deep networks,' in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.

[30] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi and M. Ebrahimi, 'Edgeconnect: Generative image inpainting with adversarial edge learning,' *arXiv preprint arXiv:1901.00212*, 2019.

[31] C. Gao, A. Saraf, J.-B. Huang and J. Kopf, 'Flow-edge guided video completion,' in *European Conference on Computer Vision*, Springer, 2020, pp. 713–729.

[32] Z. Teed and J. Deng, 'Raft: Recurrent all-pairs field transforms for optical flow,' in *European Conference on Computer Vision*, Springer, 2020, pp. 402–419.

[33] D. J. Butler, J. Wulff, G. B. Stanley and M. J. Black, 'A naturalistic open source movie for optical flow evaluation,' in *European Conf. on Computer Vision (ECCV)*, A. Fitzgibbon et al. (Eds.), Ed., ser. Part IV, LNCS 7577, Springer-Verlag, Oct. 2012, pp. 611–625.

[34] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy and T. Brox, 'A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,' in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048.

[35] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, 'Vision meets robotics: The kitti dataset,' *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[36] H. A. Haenssle, C. Fink, R. Schneiderbauer, F. Toberer, T. Buhl, A. Blum, A. Kalloo, A. B. H. Hassen, L. Thomas, A. Enk *et al.*, 'Man against machine: Diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists,' *Annals of Oncology*, vol. 29, no. 8, pp. 1836–1842, 2018.

[37] T. Kooi, G. Litjens, B. Van Ginneken, A. Gubern-Mérida, C. I. Sánchez, R. Mann, A. den Heeten and N. Karssemeijer, 'Large scale deep learning for computer aided detection of mammographic lesions,' *Medical image analysis*, vol. 35, pp. 303–312, 2017.

[38] C. Knackstedt, S. C. Bekkers, G. Schummers, M. Schreckenberg, D. Muraru, L. P. Badano, A. Franke, C. Bavishi, A. M. S. Omar and P. P. Sengupta, 'Fully automated versus standard tracking of left ventricular ejection fraction and longitudinal strain: The fast-efs multicenter study,' *Journal of the American College of Cardiology*, vol. 66, no. 13, pp. 1456–1466, 2015.

[39] A. Østvik, E. Smistad, T. Espeland, E. A. R. Berg and L. Lovstakken, 'Automatic myocardial strain imaging in echocardiography using deep learning,' in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, Springer, 2018, pp. 309–316.

[40] B. D. de Vos, F. F. Berendsen, M. A. Viergever, M. Staring and I. Išgum, 'End-to-end unsupervised deformable image registration with a convolutional neural network,' in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, Springer, 2017, pp. 204–212.

[41] B. D. de Vos, F. F. Berendsen, M. A. Viergever, H. Sokooti, M. Staring and I. Išgum, 'A deep learning framework for unsupervised affine and deformable image registration,' *Medical image analysis*, vol. 52, pp. 128–143, 2019.

[42] O. Ronneberger, P. Fischer and T. Brox, 'U-net: Convolutional networks for biomedical image segmentation,' in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[43] S. Wiehman, S. Kroon and H. De Villiers, 'Unsupervised pre-training for fully convolutional neural networks,' in *2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech)*, IEEE, 2016, pp. 1–6.

[44] E. Smistad, A. Østvik *et al.*, '2d left ventricle segmentation using deep learning,' in *2017 IEEE international ultrasonics symposium (IUS)*, IEEE, 2017, pp. 1–4.

[45] Y. Zhang, Y. Wang, W. Wang and B. Liu, 'Doppler ultrasound signal denoising based on wavelet frames,' *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 48, no. 3, pp. 709–716, 2001.

[46] G. Andria, F. Attivissimo, G. Cavone, N. Giaquinto and A. Lanzolla, 'Linear filtering of 2-d wavelet coefficients for denoising ultrasound medical images,' *Measurement*, vol. 45, no. 7, pp. 1792–1800, 2012.

[47] F. P. X. De Fontes, G. A. Barroso, P. Coupé and P. Hellier, 'Real time ultrasound image denoising,' *Journal of real-time image processing*, vol. 6, no. 1, pp. 15–22, 2011.

[48] K. Singh, S. K. Ranade and C. Singh, 'A hybrid algorithm for speckle noise reduction of ultrasound images,' *Computer methods and programs in biomedicine*, vol. 148, pp. 55–69, 2017.

[49] G. J. Tortora and B. Derrickson, *Manuel d'anatomie et de physiologie humaines*. De Boeck supérieur, 2017.

[50] J. R. Mitchell and J.-J. Wang, 'Expanding application of the wiggers diagram to teach cardiovascular physiology,' *Advances in physiology education*, vol. 38, no. 2, pp. 170–175, 2014.

[51] K. Iniewski, *Medical Imaging*. Wiley Online Library, 2009.

[52] P. R. Hoskins, K. Martin and A. Thrush, *Diagnostic ultrasound: physics and equipment*. CRC Press, 2019.

[53] C. Y. Ho and S. D. Solomon, 'A clinician's guide to tissue doppler imaging,' *Circulation*, vol. 113, no. 10, e396–e398, 2006.

[54] O. A. Smiseth, H. Torp, A. Opdahl, K. H. Haugaa and S. Urheim, 'Myocardial strain imaging: How useful is it in clinical decision making?' *European heart journal*, vol. 37, no. 15, pp. 1196–1207, 2016.

[55] A. H. A. W. G. on Myocardial Segmentation, R. for Cardiac Imaging: M. D. Cerqueira, N. J. Weissman, V. Dilsizian, A. K. Jacobs, S. Kaul, W. K. Laskey, D. J. Pennell, J. A. Rumberger, T. Ryan *et al.*, 'Standardized myocardial segmentation and nomenclature for tomographic imaging of the heart: A statement for healthcare professionals from the cardiac imaging committee of the council on clinical cardiology of the american heart association,' *Circulation*, vol. 105, no. 4, pp. 539–542, 2002.

[56] P. Geurts and L. Wehenkel, *Lecture notes in introduction to machine learning*, Sep. 2020. [Online]. Available: https://people.montefiore.uliege.be/lwh/AIA/.

[57] G. Louppe, *Lecture notes in deep learning*, Sep. 2020. [Online]. Available: https://github.com/glouppe/info8010-deep-learning.

[58] P. Baldi and R. Vershynin, 'The capacity of feedforward neural networks,' *Neural networks*, vol. 116, pp. 288–311, 2019.

[59] P. Cunningham, M. Cord and S. J. Delany, 'Supervised learning,' in *Machine learning techniques for multimedia*, Springer, 2008, pp. 21–49.

[60] *Machine learning technique for finding hidden patterns or intrinsic structures in data*, https://www.mathworks.com/discovery/unsupervised-learning.html, Accessed: 2020-12-11.

[61] E. W. Forgy, 'Cluster analysis of multivariate data: Efficiency versus interpretability of classifications,' *biometrics*, vol. 21, pp. 768–769, 1965.

[62] D. Ernst, *Lecture notes in optimal decision making for complex problems*, Jun. 2020. [Online]. Available: http://blogs.ulg.ac.be/damien-ernst/info8003-1-optimal-decision-making-for-complex-problems/.

[63] *Forpheus*, https://www.omron.com/global/en/technology/information/forpheus/, Accessed: 2020-12-11.

[64] C. J. Watkins and P. Dayan, 'Q-learning,' *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[65] D. P. Kingma and J. Ba, 'Adam: A method for stochastic optimization,' *arXiv preprint arXiv:1412.6980*, 2014.

[66] R. Rojas, 'The backpropagation algorithm,' in *Neural networks*, Springer, 1996, pp. 149–182.

[67] X. Glorot and Y. Bengio, 'Understanding the difficulty of training deep feedforward neural networks,' in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[68] M. Staring, S. Klein and J. P. Pluim, 'A rigidity penalty term for nonrigid registration,' *Medical physics*, vol. 34, no. 11, pp. 4098–4108, 2007.

[69] J.-Y. Bouguet *et al.*, 'Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm,' *Intel corporation*, vol. 5, no. 1-10, p. 4, 2001.

[70] P. Corke, *Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised*. Springer, 2017, vol. 118.

[71] T. Zhang and C. Y. Suen, 'A fast parallel algorithm for thinning digital patterns,' *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984.

[72] L. N. Smith and N. Topin, 'Super-convergence: Very fast training of neural networks using large learning rates,' in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, International Society for Optics and Photonics, vol. 11006, 2019, p. 1 100 612.

[73] S. Ünlü, O. Mirea, S. Bézy, J. Duchenne, E. D. Pagourelias, J. Bogaert, J. D. Thomas, L. P. Badano and J.-U. Voigt, 'Inter-vendor variability in strain measurements depends on software rather than image characteristics,' *The International Journal of Cardiovascular Imaging*, vol. 37, no. 5, pp. 1689–1697, 2021.

Appendix A

RAFT: update operator GRU

The update operator of the RAFT model is composed of a sequence of gated recurrent units (GRUs) where fully connected layers are replaced by 3×3 convolutions. A diagram of the structure of such unit is given in Figure A.1. The input $\mathbf{x}_t$ is the result of the concatenation of flow, correlation, and context features. Mathematically, the update gate $\boldsymbol{z}_t$, the reset gate $\boldsymbol{r}_t$ and the output state $\boldsymbol{h}_t$ are such that

$$\boldsymbol{z}_t = \sigma\left(\text{conv}_{3\times3}([\boldsymbol{h}_{t-1}, \mathbf{x}_t],\, \boldsymbol{W}_z)\right), \tag{A.1}$$

$$\boldsymbol{r}_t = \sigma\left(\text{conv}_{3\times3}([\boldsymbol{h}_{t-1}, \mathbf{x}_t],\, \boldsymbol{W}_r)\right), \tag{A.2}$$

$$\widetilde{\boldsymbol{h}}_t = \tanh\left(\text{conv}_{3\times3}([\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}, \mathbf{x}_t],\, \boldsymbol{W}_h)\right), \tag{A.3}$$

$$\boldsymbol{h}_t = (\mathbf{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \widetilde{\boldsymbol{h}}_t, \tag{A.4}$$

where σ is the sigmoid function, $\odot$ is the element-wise multiplication operator, and $\boldsymbol{W}_z$, $\boldsymbol{W}_r$ and $\boldsymbol{W}_h$ are the parameters of the convolution layers.

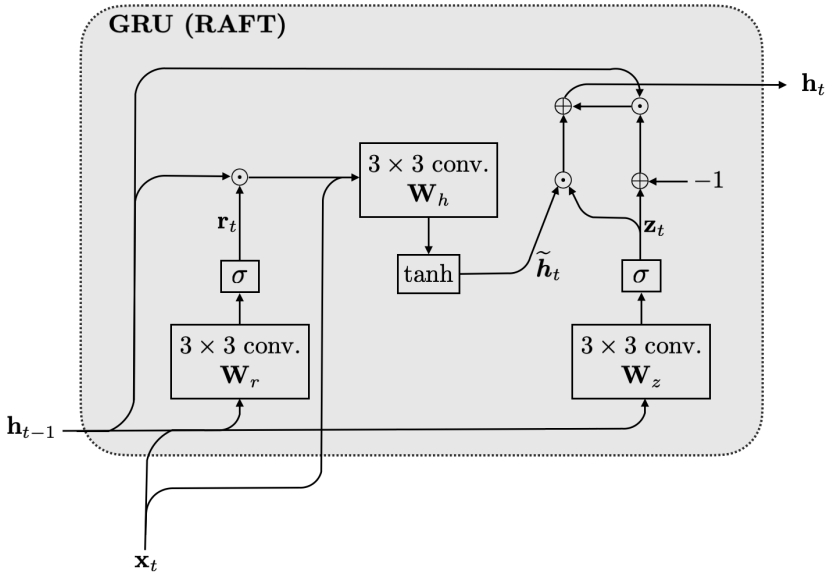


Figure A.1: Structure of the gated recurrent unit composing the update operator of RAFT model. σ is the sigmoid function. The input $\mathbf{x}_t$ is the result of the concatenation of flow, correlation, and context features. $\boldsymbol{h}_{t-1}$ and $\boldsymbol{h}_t$ are the input and output states of the unit.

Appendix B

Pre-processing pipeline

B.1 Polar-to-Cartesian transformation

The polar-to-Cartesian transformation undergone by the TVI sequences in the pre-processing pipeline is formally defined by the system

$$\begin{cases} \boldsymbol{v_x} &= -v_p \sin(\theta)\hat{\boldsymbol{x}}, \\ \boldsymbol{v_y} &= -v_p \cos(\theta)\hat{\boldsymbol{y}}. \end{cases} \tag{B.1}$$

v_p is the value of the considered pixel, $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{y}}$ are the unit vectors defining the Cartesian orthonormal framework of the frame, and $\boldsymbol{v_x}$ and $\boldsymbol{v_y}$ are the components of the pixel velocity vector $\boldsymbol{v_p}$ (in the ultrasound beam direction) in this Cartesian framework. The situation is depicted in Figure B.1.

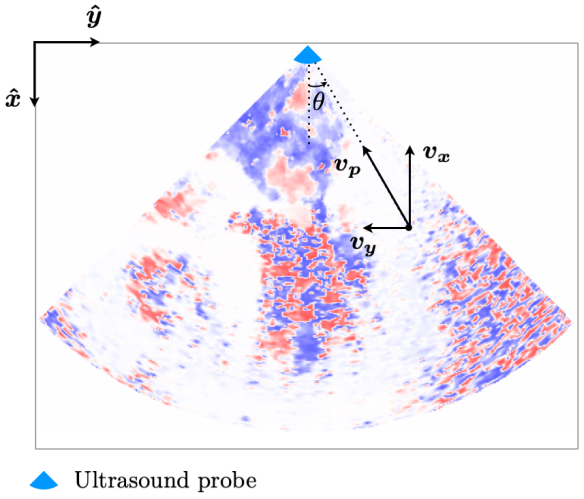


Figure B.1: Illustration of the polar-to-Cartesian transformation applied to TVI images.

B.2 TVI data integration

A coarse displacement field between two B-mode frames is computed by integrating the corresponding TVI sequence over the time period separating those frames. Let consider a B-mode sequence taken from the TVI set and its corresponding TVI sequence. The TVI sequence is chopped into chunks of m frames, with

$$m = \frac{\text{TVI frame rate}}{\text{B-mode frame rate}}. \tag{B.2}$$

Dividing the velocity vector (or its components v_x and v_y) of a pixel by the TVI frame rate gives its displacement vector from the current TVI frame to the next. The resulting displacement vector of all pixels is computed over the m frames of a chunk. This way, an optical flow describing the motion of each pixel between the two considered B-mode frames is created. Note that this displacement field is a poor estimate of the actual pixel displacement since TVI only captures speed in the ultrasound beam direction. Nevertheless, it constitutes a good first estimate for the optical flow methods developed in this thesis. TVI integration is illustrated in Figure B.2 for a single pixel.

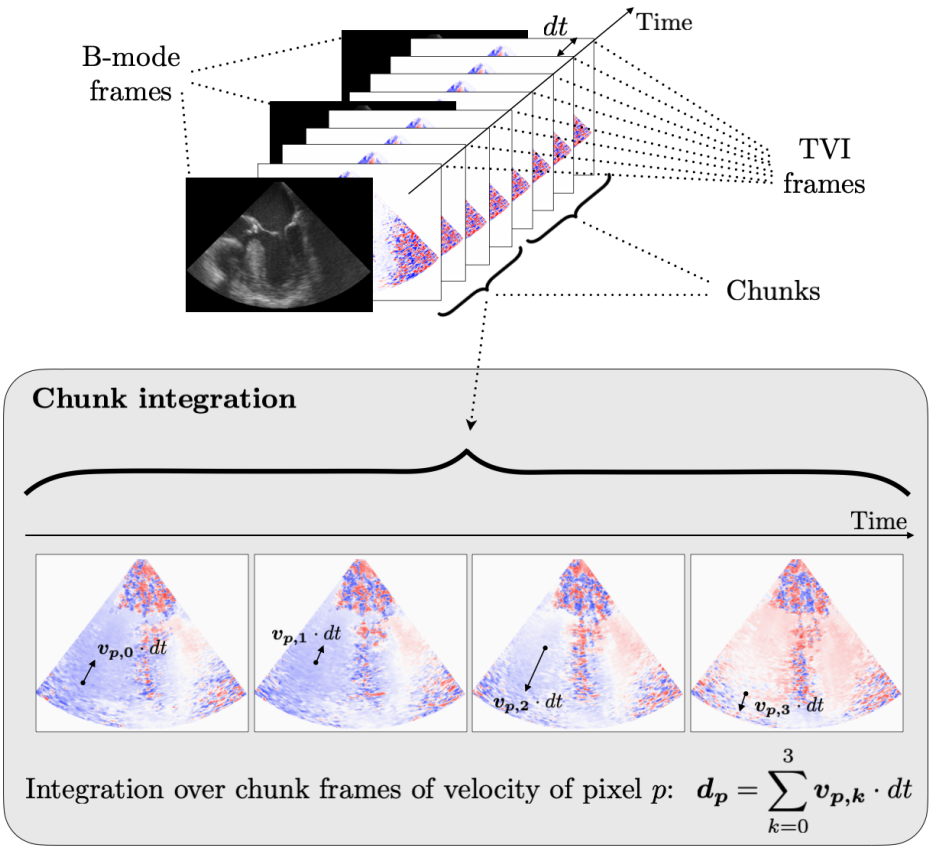


Figure B.2: Illustration of the TVI data integration process for a single pixel. The considered pixel is represented by a black dot. $v_{p,k}$ is the velocity vector of this pixel in TVI frame k. d_p is the resulting displacement vector of the considered pixel over a chunk of TVI frames.

Appendix C

Thinning algorithm

C.1 Zhang-Suen thinning algorithm

The algorithm developed by Zhang and Suen [71] is a thinning algorithm used for extracting the centerline of any shape in a binary image. Let consider a binary mask of the LV, where the pixel value is 1 if the pixel is part of the LV, 0 otherwise. Let consider a given pixel P_1 that is not located on the border of the image. P_1 has 8 neighbors, ordered as shown in Figure C.1. Let define $A(P_1)$ as the number

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

Figure C.1: Convention taken by Zhang and Suen [71] to represent the 8 pixel neighbors of a pixel P_1.

of transitions from 0 to 1 in the sequence $P_2P_3P_4P_5P_6P_7P_8P_9P_2$, and $B(P_1)$ as the number of pixel of value 1 among the 8 neighbors of P_1. The algorithm is described by the following steps:

- **Step 1** - Iterate over all pixels P_1 of value 1 that have 8 neighbors. Keep in memory (without performing any modification) the pixels that satisfy the following conditions:
  1. $2 \leq B(P_1) \leq 6$,
  2. $A(P_1) = 1$,
  3. At least one of P_2, P_4 and P_6 has the value 0,
  4. At least one of P_4, P_6 and P_8 has the value 0.

Once all pixels P_1 of value 1 that have 8 neighbors are tested, set the value of the pixels kept in memory to 0.

- **Step 2** - Iterate a second time over all pixels P_1 of value 1 that have 8 neighbors. Keep in memory (without performing any modification) the pixels that satisfy the following conditions:

  1. $2 \leq B(P_1) \leq 6$,
  2. $A(P_1) = 1$,
  3. At least one of P_2, P_4 and P_8 has the value 0,
  4. At least one of P_2, P_6 and P_8 has the value 0.

  Once all pixels P_1 of value 1 that have 8 neighbors are tested, set the value of the pixels kept in memory to 0.
- **Step 2** - Repeat **Step 1** and **Step 2** until no image pixels are changed.

The output of the algorithm is illustrated for three different LV masks in Figure C.2.

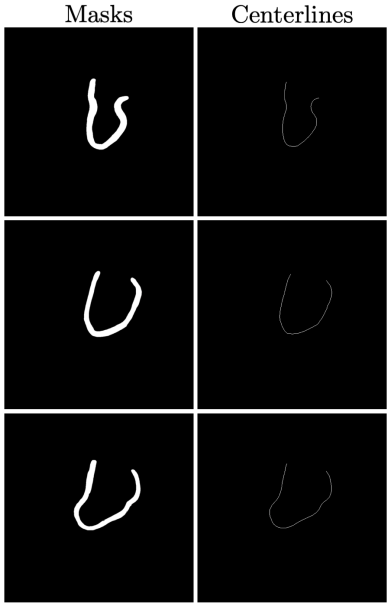


Figure C.2: Illustration showing three LV masks and their corresponding centerlines computed with the Zhang-Suen algorithm. [71]

C.2 Skeleton refining algorithm

The LV myocardium masks generated using U-Net are not perfect, and the skeleton of the segmented area can present ramifications and/or inner loops. The skeleton refining algorithm is charged to prune the unwanted ramifications and break the inner loops of the skeleton in order to keep a single centerline for the mask. First, the algorithm tries to find the two skeleton ends that are the most likely to be the

the extremities of the desired centerline. To do so, the skeleton image is split into two parts. The cut is made in the height direction at the location of the ultrasound probe. The skeleton end in each image part that is the closest to the location of the ultrasound probe is chosen as extremity of the desired centerline. If one of the part of the image does not contain any skeleton end, then the algorithm chooses the ends that are the closest to the location of the ultrasound probe, regardless of the part of the image they are located in. The longest path in the skeleton joining those two ends is then kept as centerline. It can happen that a skeleton has only one or no end. This is the case if the skeleton is made of one or several loops interconnected together. In this case, a small portion of the skeleton is erased. This creates two new ends and the algorithm can be applied as before. The erased portion is 10 pixels long and is the closest to the ultrasound probe location in the image. The skeleton refining algorithm is illustrated in Figure C.3. Note that this algorithm fails to find a suitable centerline when the LV myocardium mask is too far from the one expected. In this case, the user has the possibility to manually segment the myocardium around the LV.

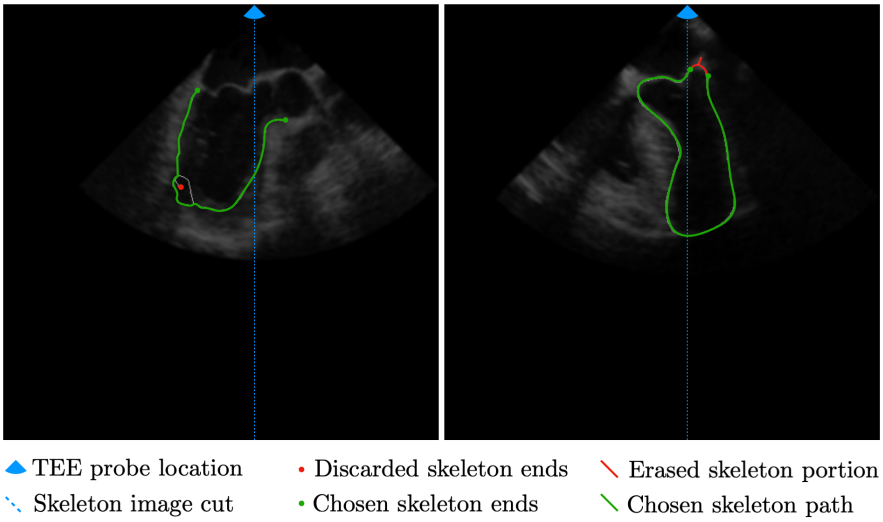


Figure C.3: Illustration of the skeleton refining algorithm applied to two cases. The skeleton on the left is a classic case. The skeleton on the right is a special case in the fact that it has a single end. The red portion of the skeleton is thus erased and two new ends are created. It also illustrates the case where one side of the cut has no end. The chosen skeleton path is used as centerline. The B-mode frames for which the centerline is to be found are displayed in transparency for the purpose of illustration.

Sven Goffin

Improved Strain Computation for Transesophageal Echocardiography Acquisitions

NTNU
Norwegian University of
Science and Technology