NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY, MATHEMATICS AND ELECTRICAL ENGINEERING

MASTER'S THESIS

Student's name:	Anne Vold Torland
Area:	Intelligent systems, image processing
Title (English):	**Automatic recognition of standard views in ultrasound images of the heart**
Description:	The goal of the thesis is to develop an algorithm for automatic recognition of 5 types of standard views of the heart in ultrasound images. The candidate is expected to: - Give an overview of current object recognition approaches, and choose an algorithm suitable for the heart view recognition problem - Provide theoretical background for the chosen algorithm - Implement the algorithm in MATLAB
Department:	**Department of Computer and Information Science**
Academic supervisor:	Associate professor Ketil Bø, Department of Computer and Information Science
Supervisors:	Professor Hans Torp and Svein Arne Aase, Department of Circulation and Medical Imaging

Abstract

With medical imaging, clinicians are given new opportunities in inspection of anatomical structures, surgical planning and diagnosing. Computer vision is often used with the aim of automating these processes.

Ultrasound imaging is one of the most popular medical imaging modalities. The equipment is portable and relatively inexpensive, the procedure is non-invasive and there are few known side effects.

But the acquisition of ultrasound images, for instance of the heart, is not a trivial job for the inexperienced. Five classes of standard images, or standard views, have been developed to ensure acceptable quality of ultrasound heart images. Automatic recognition of these standard views, or classification, would be a good starting point for an "Ultrasound for dummies" project.

Recently, a new class of object recognition methods has emerged. These methods are based on matching of local features. Image content is transformed into local feature coordinates, which are ideally invariant to translation, rotation, scaling and other image parameters.

In [21], David Lowe proposes the Scale Invariant Feature Transform (SIFT), which is a method for extracting distinctive invariant features from an image. He also suggests a method for using these features to recognize different images of the same object.

In this thesis I suggest using the SIFT features to classify heart view images. The invariance requirements to a standard heart view recognition system are special. Therefore, in addition to using Lowe's algorithm for feature extraction, a new matching algorithm specialized at the heart view classification task is proposed.

Preface

This thesis is written as a part of my Master of Science degree in Computer Science at the Department of Computer and Information Science, Norwegian University of Science and Technology.

The work was carried out at the Department of Circulation and Medical Imaging.

I would like to thank the following people:

- Associate professor Ketil Bø for inspiration and competent guidance
- Svein Arne Aase and professor Hans Torp for the thesis idea, ultrasound data and helpful assistance
- Dr. med. Asbjørn Støylen for provision of ultrasound data and useful explanations
- Brage Høyem Amundsen for provision of ultrasound data

Contents

Chapter 1

Introduction

The development of medical imaging technologies has in many ways revolutionised medicine. With medical imaging, clinicians and technologists are given the opportunity to peer non-invasively into the human body. But medical imaging is more than simple visualisation and inspection of anatomic structures. It now plays an increasing role in diagnosing, and it has become a tool for among others surgical planning and simulation, intra-operative navigation and tracking the progress of disease[26].

Ultrasound imaging is for several reasons one of the most popular medical imaging modalities: it is non-invasive, the equipment is portable and relatively inexpensive, and there are few known side effects. A lot has been achieved in image processing and computer vision of ultrasound images, especially in the area of diagnosis. For some researchers the aim is more accurate measurements of organ properties. By extracting organ contours, like for example of the kidneys[23] or of the heart[3, 15, 13],volume can be estimated. Others have gone further in the automation of the diagnosis process. In Australia, the company Polartechnics has to a great extent succeeded in automatically distinguishing melanoma, a deadly form of skin cancer, from benign lesions[50]. Similar approaches have been made for liver tumours[55], and for breast cancer[7].

Acquiring ultrasound images is not such an easy task, at least not for the inexperienced. To ensure acceptable quality of ultrasound images of the heart, five classes of standard images, or views, have been developed. At the Department of Circulation and Medical Imaging they are interested in using ideas and methods from the field of computer vision to make the recording task easier. The idea is that the inexperienced user is given instructions from the ultrasound scanner as he or she attempts to obtain one of the standard images. In short: Ultrasound for dummies.

1.1 Problem definition

In order to help the inexperienced user, the standard images need to be recognised by the ultrasound scanner. This means that given an input image, the scanner should be able to classify it as either one of the established standards, or as a non-standard image.

The goal of this thesis is to develop an algorithm that automatically distinguishes between the five classes of standard images of the heart.

Although time issues will be of great importance to the future application, the implementation focus here is on result, and not on run-time. Also, the assignment is limited to the actual recognition task; quality assurance is not a requirement. In other words: the system should be able to recognise a standard image, but not necessarily be able to deduce anything about its quality. Getting the system to reject images of poor quality will be a next step.

1.2 Outline

Chapter 2 provides some background theory and a more thorough introduction to the recognition problem. Chapter 3 treats previous related work, while chapter 4 describes the well known SIFT algorithm and a new matching algorithm. Chapter 5 deals with the implementation of the recognition algorithm, and also describes two performance tests. Chapter 6 states the test results, while in chapter 7, the algorithm, results and future work are discussed. Chapter 8 concludes this thesis work.

Chapter 2

Background

This chapter provides background theory to the reader unfamiliar with ultrasound imaging, echocardiography or computer vision. Basic elements of ultrasound imaging are described in the first section. This is followed by a short introduction to the anatomy and physiology of the heart, required to understand the basics of echocardiography which is provided in the third section. The last section treats computer vision, and finally it gives a more thorough introduction to the problem of recognising standard views in ultrasound images the heart.

2.1 Medical ultrasound imaging

If nothing else is stated, the information in this section is taken from[2].

Lazzaro Spallanzani (1727-1799) is frequently referred to as the "father of ultrasound". He demonstrated that bats in are fact blind, and that they navigate by means of echo reflection using inaudible sound. The medical application of ultrasound dates back to the early 1950s, when it was demonstrated that ultrasound could be used for detection of tissue layers, tumors and heart structures. Since then ultrasound imaging has become one of the most popular medical imaging modalities. The method is non-invasive, the equipment is portable and relatively inexpensive, and there are few known side effects.

2.1.1 General principles

Ultrasound is a term for mechanical waves with frequencies above the audible range[16]. Audible sound has a frequency in the range 20-20000 Hz. Medical diagnostic usually operates in the range 2-10 MHz, but for intraarterial imaging, frequencies up to 40 MHz have been used.

Sound velocity is an important characteristic of the medium in which the ultrasound waves propagate. Sound velocity is given by:

$$c = 1/\sqrt{\rho\kappa},$$

where ρ is mass density, and κ is volume compressibility of the medium. The speed of sound is 340 m/s in air, and about 1500 m/s in water. Soft tissue consists mainly of water, with some solids added. Therefore, the speed of sound varies little between different types of soft tissue, and is about 1540 m/s[16], except for in bone.

Medical ultrasound can be used for imaging at a distance of up to 20 cm. At this scale, the properties of ultrasound resemble those of visible light[16]:

- The sound waves travel in straight lines, and can be focused and bent
- A sound wave hitting a medium with substantial differences in characteristics will be almost totally reflected

The acoustic or characteristic impedance is given by:

$$Z = \rho\ c.$$

It is variations in acoustic impedance between two materials that causes the reflection of the ultrasound wave. The reflection against air and bone is strong. This means that if some part of the heart is blocked by the lung, this part can not be imaged with ultrasound. Also, the impedance of fat is so much lower than for muscular tissue, that is causes a strong scattering of the ultrasound together with a bending of the beam. This results in artefacts in the image.

A plate of piezoelectric material with electrodes on each face can be used to generate ultrasound. A voltage source is coupled to the electrodes. This is called a transducer, or a probe, see figure 2.1[16]. According to the polarity of the voltage, the piezoelectric plate is either expanded or compressed. This vibration can be transferred to the tissue. And conversely, mechanical influence from outside will generate voltage. This way, the probe can be used both as a sender and a receiver by rapid switching of its modes.

Figure 2.1: An ultrasound probe
[16]

Ultrasound imaging is based on measuring pulse echo, the same principle as used in RADAR (RAdio wave Detection And Ranging) and SONAR (SOund Navigation And Ranging). A pulse sent out by the transducer propagates through the material. When the pulse encounters a boundary between two tissue structures, it will be partially reflected and partially transmitted. The wave now has three components: the incoming wave, the reflected, backward wave, and the transmitted, forward wave. This is illustrated in figure 2.2[2]. The time from the pulse is sent until the echo is received, is a measure of the distance from the reflecting area. Due to the reflections, the transmitted energy of the wave is reduced as it passes through the tissue interfaces.

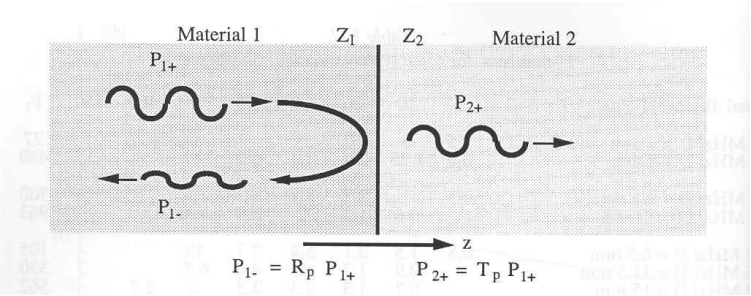


Figure 2.2: Illustration of the reflection of an incoming wave[2]

2.1.2 Image quality

The resolution of the imaging system is a measure of how small details that can be imaged. Radial resolution, Δc, in direction of the beam, and lateral resolution, Δr, are given by[16]:

$$\Delta c = \tau/2 \tag{2.1}$$

$$\Delta r = \lambda \cdot F/D \tag{2.2}$$

Figure 2.3[2] shows the object during imaging and the image. As seen from (2.1) and (2.2), radial - and lateral resolution are determined by different mechanisms. Usually the radial resolution is better than the lateral resolution. The length of the pulse determines radial resolution, while lateral resolution depends on the ratio between the wavelength and the size of the probe and increases linearly with distance from the probe. According to this, it would be best to use the highest possible frequency and the largest possible probe. But unfortunately there are other limitations as well. The higher the frequency, the more the tissue will attenuate the energy of the wave, and this limits the range. Hence there is a trade-off between resolution and penetration. In cardiology, as low as 2,5-3,5 MHz is used on adults, while 5-7,5 MHz is used on children and infants. As to probe size, the distance between the ribs places a limitation of 20 mm on the size of the probe in cardiology.

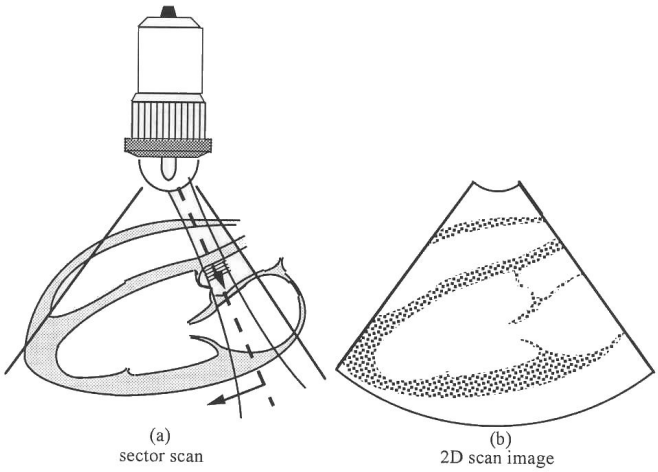


Figure 2.3: Two dimensional amplitude imaging of structures in the heart[2]

It is the mixture of the different kinds of tissue, such as muscle and fat that produce the scattering of the ultrasound which is used to generate the ultrasound images. But the mixtures of tissue types in the body wall often produce so complex inhomogeneities that they also destroy the ultrasound beam. Irregular variations in wave velocity destroy the wavefront of the beam. This blurs the beam and with this the image, and is called wave front aberration. Multiple reflections of the transmitted pulse, called reverberations, can cause false contours in the image and reduce the image quality considerably. Fatty tissue structures are the most difficult inhomogeneities, because they have a low wave velocity compared to muscular tissue.

2.1.3 Imaging modes

Several imaging modalities are used in medical ultrasound. Mainly, there is amplitude imaging, and imaging based on the Doppler effect. Only amplitude images were used when working on this thesis, so Doppler based imaging will not be described.

The first amplitude echo imaging method was A-mode. A beam was sent into the tissue, and the backscattered signal was picked up with the same transducer. The returned echo was displayed on oscilloscope.

M-mode shows the amplitude of the backscattered signal in greyscale, see figure 2.4[2]. One axis represents time, while depth in the tissue is displayed along the other axis.

B-mode gives two-dimensional greyscale images of tissue structures. Light intensity is used to show stronger echo. Figure 2.5 shows a typical B-mode image. Ordinary B-mode is based on sending and receiving on the same frequency. Receiving on twice the sending frequency, called 2nd harmonic imaging, enhances

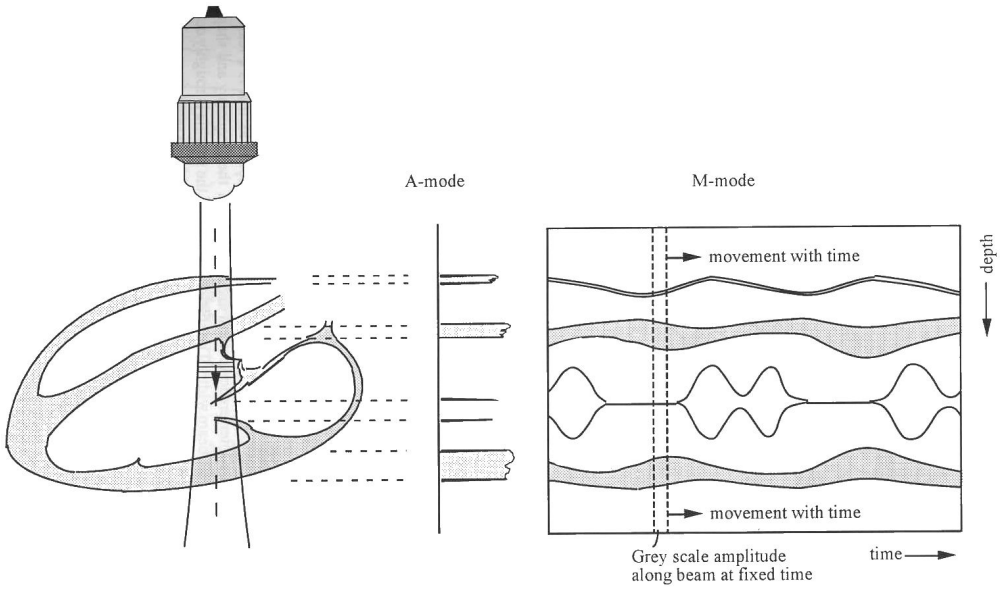


Figure 2.4: M-mode display of moving structures in the heart[2]

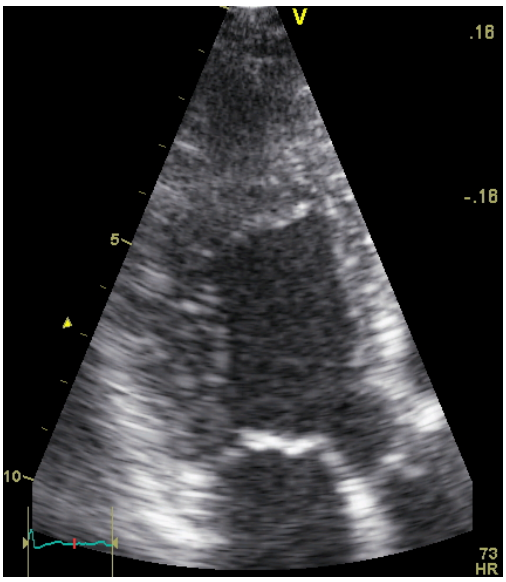


Figure 2.5: Two-dimensional B-mode image

the contrast and greatly reduces the reverberation noise[16].

2.2 The heart

This section provides an introduction to the circulatory system, and to the anatomy and physiology of the heart. If nothing else is stated, the information in this section is taken from[35] and[4].

2.2.1 The circulatory system

The circulatory system consists of the heart, which is a muscular pumping device, and a closed system of vessels. An interior wall partitions the heart in two, each part with its own pump. With each beat, the heart pumps blood into two closed circuits, see figure 2.6. The left side of the heart is the pump for the systemic circulation. Blood flows from the left side of the heart through blood vessels to all parts of the body except for the lungs, and back to the heart. The right side of the heart is the pump for the pulmonary circulation. Here blood is pumped from the right side of the heart to the gas exchange tissues of the lungs. The two circuits are arranged in series. In the same amount of time, the same amount of blood passes the systemic circulation and the pulmonary circulation.

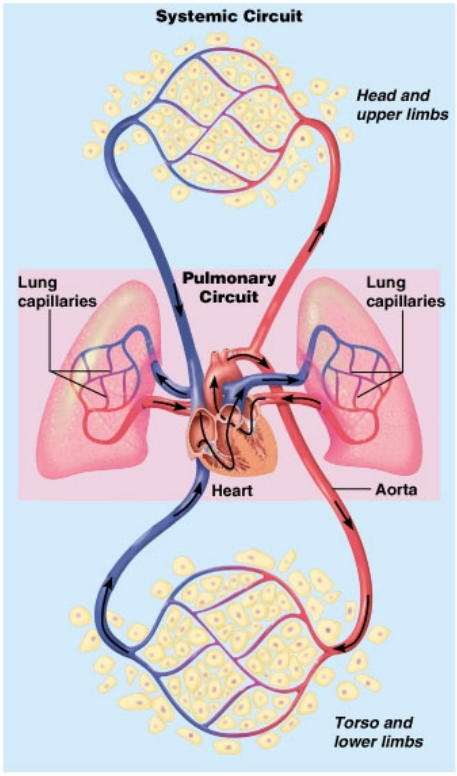
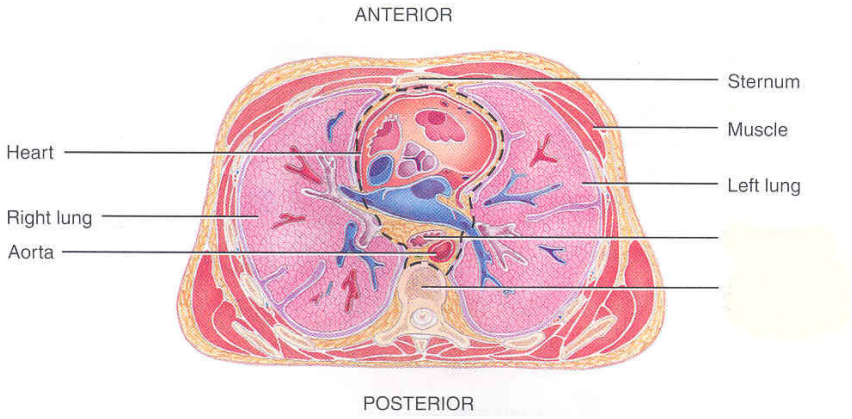


Figure 2.6: Blood flow through the circulatory system[1]

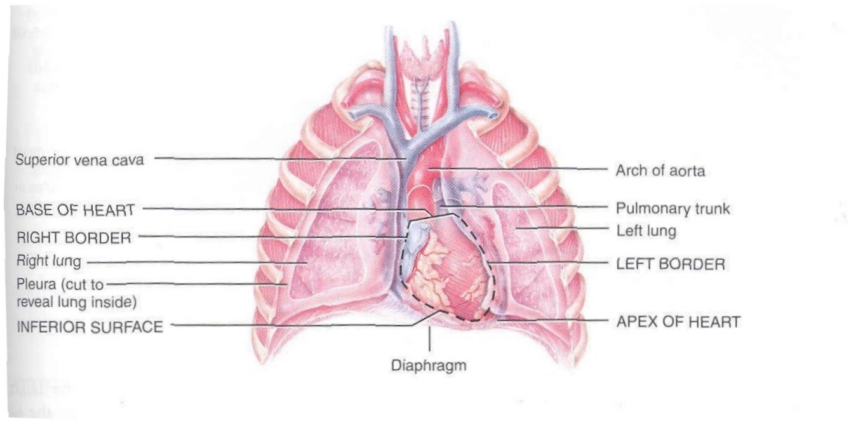
2.2.2 Anatomy of the heart

The human heart is a four-chambered muscular organ. It is shaped like a cone and sized roughly like a person's closed fist. It is normally situated slightly to the left of the middle of the thorax, underneath the sternum, and it is surrounded by the lungs. See figure 2.7. About two-thirds of the mass of the heart lies to

the left of the body's midline[11]. The pointed end of the heart is the apex, directed anteriorly, inferiorly, and to the left. The broad portion of the heart is the base, which is directed posteriorly, superiorly, and to the right.



(a)



(b)

Figure 2.7: Location of the heart[11]

The interior of the heart is divided into four chambers. The two upper chambers are the atria, and the two lower chambers are the ventricles. The left and right chambers are separated by an extension of the heart wall called the septum, see figure 2.8.

The right atrium receives blood from three veins: the superior vena cava, inferior vena cava and coronary sinus[11]. Blood flows from the right atrium and into the right ventricle through a valve called the tricuspid valve, see figure 2.9. From the right ventricle, blood flows through the pulmonary valve and into a large artery called the pulmonary trunk[11]. The left atrium receives blood from the lungs through four pulmonar veins. From the left atrium blood flows into the left ventricle through the bicuspid (mitral) valve, and from the left ventricle blood flows into the largest artery of the body, the Aorta.

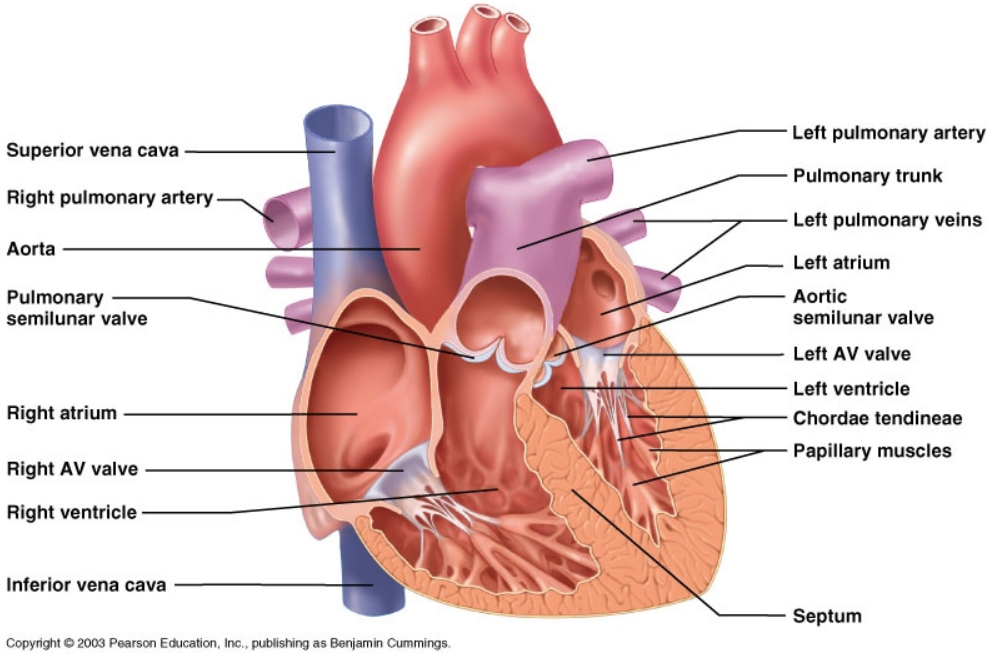


Figure 2.8: Interior of the heart

The tricuspid and bicuspid valves are also called the atrioventricular (AV) valves, because they guard the openings between the atria and the ventricles, see figure 2.8 again. The aortic and pulmonary valves are also known as the semilunar (SL) valves, because they are made up of crescent moon-shaped cusps.

The four sets of valves are of great importance to the normal function of the heart. As each chamber contracts, it pushes a volume of blood into a ventricle or into an artery. Each of the valves open and close in response to pressure changes as the heart contracts and relaxes. This way, the valves helps ensure the one way flow of blood.

The walls of the heart consists mainly of myocardium, cardiac muscle tissue[11]. The thickness of the myocardium of the four chambers varies according to each chamber's function. The walls of the ventricles are thicker than the walls of the atria. This is because the ventricles pump blood greater distances and therefore more force is needed. Likewise, the myocardium of the left ventricle is thicker than that of the right ventricle because the left ventricle pushes blood through most of the vessels of the body, whereas the right ventricle pushes blood only through the vessels that serve the gas exchange tissues of the lungs[35]. This is illustrated in figure 2.10[11].

2.2.3 The cardiac cycle

The cardiac cycle has two basic components:

1. A phase during which the ventricles are relaxed, called the diastole

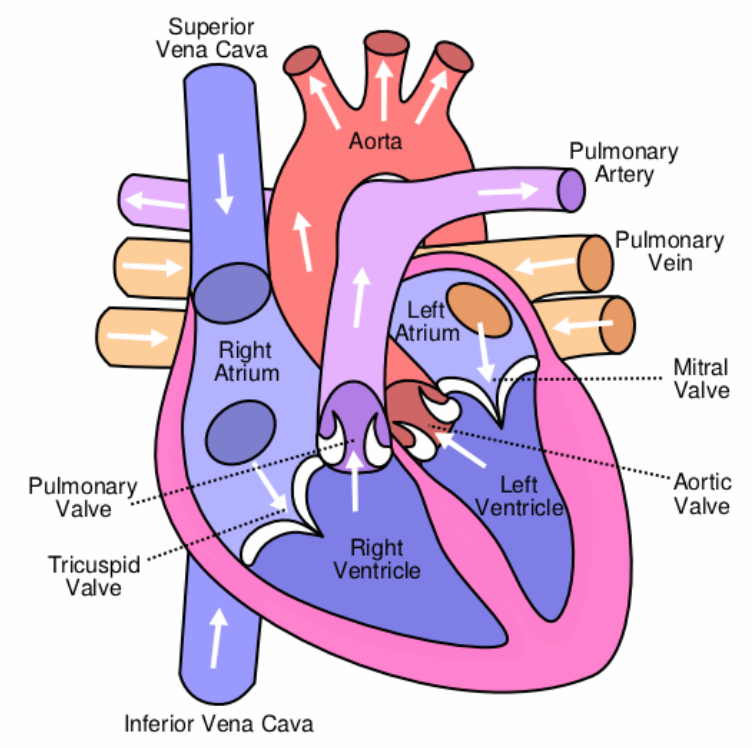


Figure 2.9: Blood flow in the heart[38]

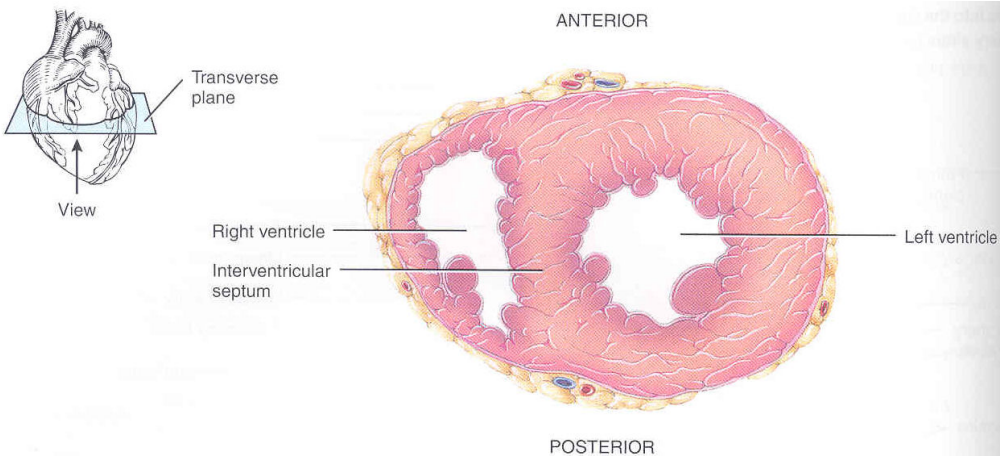


Figure 2.10: Thickness of the walls[11]

2. A phase during which the ventricles are contracted, called the systole

Figure 2.11 illustrates the cardiac cycle. When the atria fill with blood returning to the heart, the pressure opens the AV valves. Blood drains into the ventricles. The filling of the ventricles will be rapid at first, because of all the blood that has gathered in the atria during the systole, but the filling will decrease in speed as the pressure is equalized. Towards the end of the diastole the atria contract,

squeezing additional blood into the ventricles. During the diastole the pressure in the ventricles will always be less than that in the arteries, and the SL valves will therefore be closed.

The ventricles contract in the systole. Due to the contraction, the pressure in the ventricles will almost immediately exceed that in the atria, and so the AV valves close. The contraction continues and shortly after the pressure in the ventricles exceeds that in the arteries. The SL valves open, and blood is squeezed into the arteries. When the ventricles relax at the end of the systole, the pressure will again be less than that in the arteries and the SL valves close. Soon the pressure in the ventricles is also less than that in the atria. The AV valves open again, and a new cardiac cycle has begun.

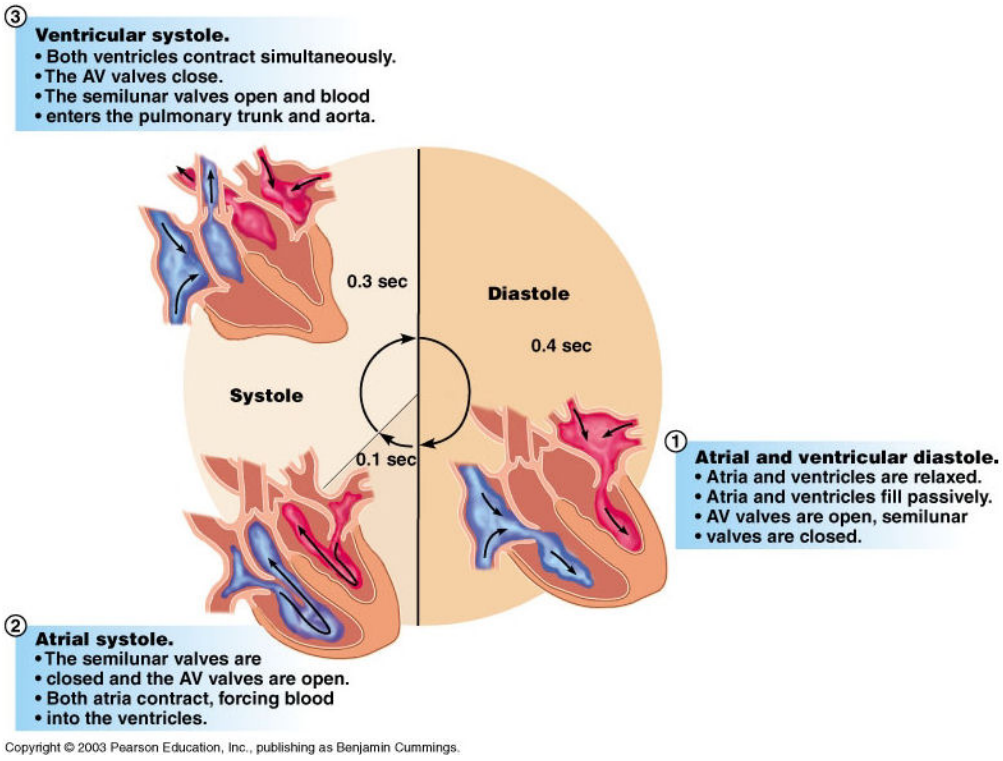
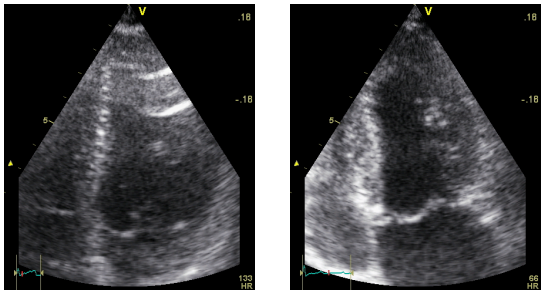


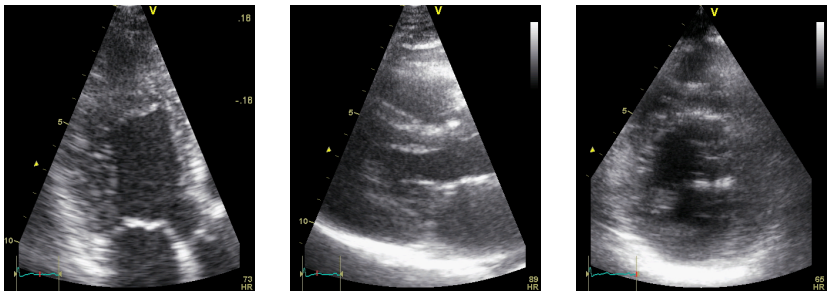
Figure 2.11: The cardiac cycle[11]

2.3 Echocardiography

The development of echocardiography, a diagnostic procedure that uses ultrasound waves to create an image of the heart muscle, started in Sweden by Inge Edler and Helmuth Hertz in the early 1950s[36]. Echocardiography is used in the diagnosis and management of heart diseases. It involves processing and display of echocardiographic information to answer specific questions regarding the size, motion, and structural and functional characteristics of the heart.



(a) Four-chamber image (b) Two-chamber image



(c) Apical long axis image (d) Parasternal long axis image (e) Parasternal short axis image

Figure 2.12: Standard ultrasound images of the heart

Echocardiography may show abnormalities such as damage to the heart tissue from a past heart attack, or poorly functioning heart valves.

Figure 2.12 shows typical ultrasound images of the heart. What can be seen in each of them is a cross-section of the heart. The walls of the heart are white because they give a stronger echo than the blood in the chambers. The images illustrate some of the challenges when performing signal processing and image processing on ultrasound images. The ultrasound images are extremely noisy, compared to for instance MRI (Magnetic Resonance Imaging) images, and it is not always easy to distinguish wall from blood.

2.3.1 standard views

Because of the heart's location between the lungs and behind the ribs, both totally reflecting the sound waves, several standard imaging methods are developed. The purpose of this standardisation is to ensure sufficient quality of the acquired images.

The apical four-chamber view is usually the easiest one to produce. This view is achieved by letting the patient lie on the back, about 45 degrees to the left side. The transducer is placed approximately at the six-jagged star in figure 2.13[10], and is directed superiorly, medially and posteriorly. The six-jagged star marks the transducer position for all the apical views. Figure 2.12(a) shows a typical four-chamber view. According to current standard, the left half of the heart is

in the right side of the image. The vertical white line a bit to the left is muscle wall. The two shorter horizontal lines in the image are the atrioventricular valves.

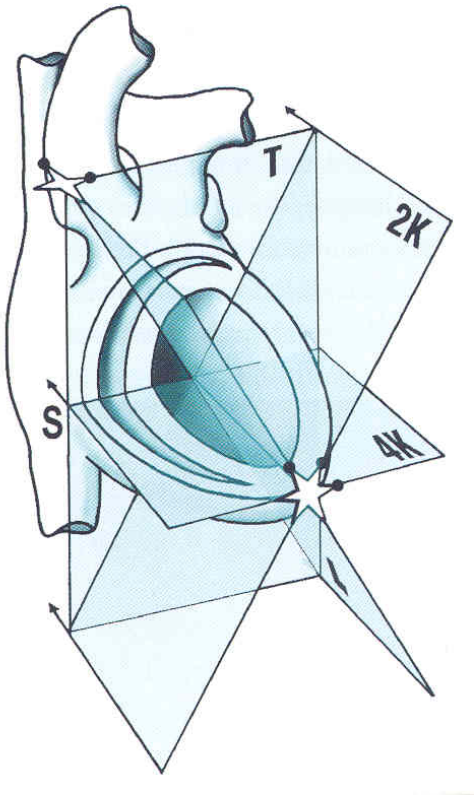


Figure 2.13: Standard imaging planes[10]

According to[10], when the four-chamber view seems satisfactory, the ultrasound operator should make sure that the transducer is positioned right on top of apex. It is also possible to achieve four-chamber views with a slightly different transducer position, but then the image plane will run aslantly through the ventricle, which is unfortunate (see section 2.3.2).

The apical long axis view can be produced with the four-chamber view as a starting point. The position and axis of the transducer remains unchanged, and the transducer is rotated clockwise 120 degrees. Figure 2.12(c) shows a typical apical long axis view. The aorta can here be seen running from the left ventricle and down to the right

The apical two-chamber view can be found halfway between the apical four-chamber view and the apical long axis view. That is, the transducer should be rotated clockwise 60 degrees from the long axis view. This is the hardest of the three apical projections, because the image plane contains no evident landmarks[10], see also section 2.3.2. Figure 2.12(b) shows a typical two-chamber view. What can be seen in the two-chamber view are the walls of the left ventricle, the mitral valve in the middle, and the left atrium at the bottom.

Letting the patient lie 90 degrees on his or her side, and placing the transducer close to sternum produces the parasternale long axis view. The scanning plane is identical to the apical long axis plane, and therefore aslant to the patient's body. But the viewpoint of the transducer is not identical to that of the apical views. Figure 2.12(d) shows a typical parasternal long axis view. The aorta and the left atrium can be seen in the right part of the image. The mitral valve is in the middle, and in the left bottom part the two ventricles are divided by the septum.

The parasternal short axis view is imaged with the transducer in the same position as that of the parasternal long axis view, but rotated clockwise 90 degrees. Figure 2.12(e) shows a typical parasternal short axis view.

2.3.2 Typical defects when recording standard views

Apart from the image quality factors presented in 2.1.2, there are one more requirement that must be fulfilled in order to image a standard view of accepted quality. This requirement is called alignment, and says that the image plane must slice both apex and the mid-atrioventricular plane that is made up by the atrioventricular valves. In other words:

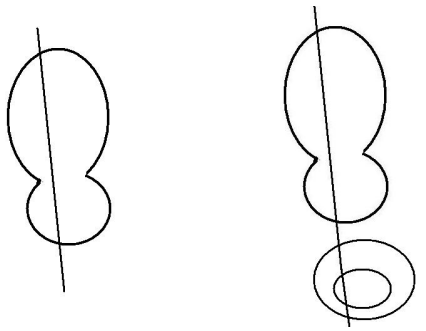
1. The slicing must be through apex
2. The angle must be right

Good alignment is hardest to achieve in two-chamber views of the heart. This is due to the non-existence of landmarks in this view; there is simply nothing to navigate after. Figure 2.14(a) roughly illustrates an example of bad alignment. The ventricle is not sliced through apex, and the slicing is not through the middle of the atrioventricular plane. This alignment would for instance be reflected in the parasternal short axis view. The bad alignment causes a reduction in diameter, illustrated in figure 2.14(b). The diameter reduction is in percentage much larger in diastole, and this can actually cause the cross-section to disappear from the images during the diastole.

Still, because lungs and ribs might get in the way, slicing through apex does not always yield the best results. The sometimes conflicting goals of alignment and image quality must be balanced.

2.4 Computer vision and object recognition

Computer vision is a branch of artificial intellligence. As we know, vision allows humans to perceive and understand their surroundings. "Computer vision aims at duplicating the effect of human vision by electronically perceiving and understanding an image"[48]. This may seem like an impossible goal to achieve. The important thing to understand is that the field of computer vision deals with methods a computer can use to obtain information about the surroundings,



(a) Example of bad alignment (b) Effect of bad alignment

Figure 2.14: Bad alignment

by analysing images of the surroundings. These methods do not necessarily resemble those used by humans to perceive the surroundings, as long as they yield correct results[42]. Computer vision often use results and methods from many other scientific disciplines, including mathematics, pattern recognition, electronics and psychophysiology[48].

There was an explosion of interest for image processing and computer vision in the 1970s. Many different and important applications were introduced, making work easier in fields such as satellite observation of Earth, navigation, industrial inspection, automatic surveillance and biomedicine[37].

But even though there have been major achievements, the monolithic goal of automatically interpreting a general digital image of an arbitrary scene still remains out of reach[17]. Giving computers the ability to see is an extremely complex task. The main problem is perhaps that we live in a three-dimensional world, while the images given to the computers by the visual sensors usually are two-dimensional. This projection of the scene onto an image plane incurs an enormous loss of information. Information like absolute scale and depth is lost. There are in fact an infinite number of scenes that can produce the exact same image[17]. In addition, the scenes in real life are usually dynamic, with a moving object or a moving camera, which makes computer vision even more complicated[48]. The difficulty of the general problem has caused the field of computer vision to focus on smaller, more constrained pieces of the problem.

When humans interpret scenes or images of scenes, they use a lot of a priori knowledge. The machine, on the other hand, starts with an array of numbers from which it is supposed to make identifications and draw conclusions. It will need both general knowledge, domain-specific knowledge and information extracted from the image when attempting to ”understand” these arrays of numbers[48].

Human object recognition allows us to understand the contents of images. From generic knowledge of an object, we can easily recognize novel instances of the object. For example, our internal model of a dog makes the recognition of a new breed of dog as a dog effortless, regardless of whether the dog is sitting or

running. This example raises some important questions: How are these internal models of the visual appearance of an object encoded? What information is extracted from an image in order to recognize an object? And how is this information compared to the internal models?

2.4.1 General recognition of 3D objects from 2D images

Recognition of general three-dimensional objects from two-dimensional images is an important part of computer vision. According to Matas and Obdrzalek, the common formulation of the problem is essentially [24]:

> *Given some knowledge of how certain objects may appear, plus an image of a scene possibly containing those objects, find which object are present in the scene and where*

Matas and Obdrzalek also define three goals in designing an object recognition system[24]:

Generality The ability to recognise any type of object

Robustness The ability to recognise the objects in arbitrary conditions

Easy learning Avoidance of special or demanding procedures to obtain the database of models

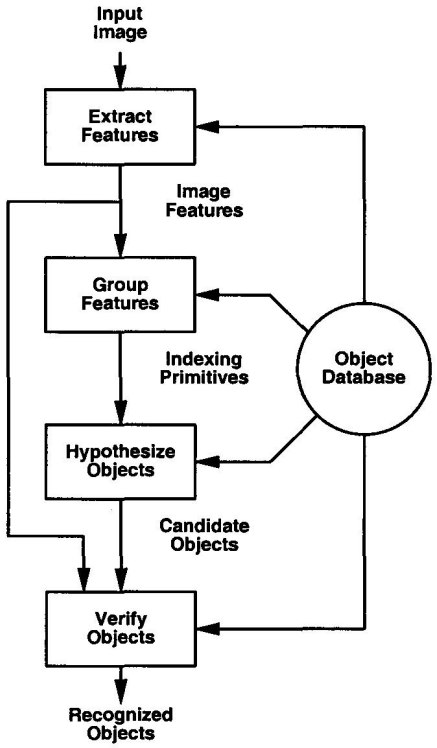


Figure 2.15: The components of an object recognition system

The components of an object recognition system is outlined in figure 2.15. Input to the system is a digital image, and output is an object label. Recognition is accomplished by matching features of an image and a model of an object, stored in an object database. The recognition has four main steps:

1. **Extract features:** First, a set of features, or descriptors, are extracted from the input image. The goal is to extract, from the large amount of image data, only the information necessary to identify or distinguish the object. Typically, the features represent some scalar properties of objects. A single feature is usually not sufficient for object representation, and therefore the features are combined into feature vectors. One simple example of a feature vector is a combination of size and compactness: $\mathbf{x} = (size, compactness)$.

2. **Group features:** After the extraction, features are grouped into meaningful collections, called indexing primitives. This can for instance be collections of extracted edges or lines

3. **Hypothesize objects:** A matching algorithm compares the indexing primitives to the database of object models. The algorithm returns a hypothesis, namely a set of candidate objects, which all contain the indexing primitive.

4. **Verify objects:** Finally, the candidate objects must be verified in terms of how well it matches the image data. Typically, a score is assigned each candidate, and the best-scoring candidate is chosen as label of the object

The backbone of any recognition system is its model object representation[8]. The model object representation governs what kind of features is extracted, how they are grouped and how they are matched to models.

Often, when there are multiple objects in an image, the object recognition system is expected not only to identify objects that are partially visible, but also to determine their exact position and orientation in the world. This is called pose estimation.

Within recognition, there is a distinction between two main tasks: identification and categorization. Computer vision techniques achieve identification, like face identification, relatively easy, but categorization with much more difficulty. For biological visual systems, on the other hand, categorization is suggested to be much simpler[40].

The main computational difficulty in object recognition is the problem of variability[40]. An object recognition system needs to generalize across huge variations in the appearance of an object. Two images of the same object can be quite different, depending on viewpoints, lightings and acquisition techniques. In addition to small variations in shape, two different objects of the same type can also differ in for example size, texture and colour. And maybe worse: often one object is partially occluded by another object. Due to this variability problem, there has been much focus in the field of object recognition on extracting features

that are invariant to the different variations. To summarize, the features extracted should optimally be invariant to rotation, translation and scaling, and they should be robust to illumination changes, noise and occlusion.

2.4.2 Recognition of standard heart views in ultrasound images

The special problem of recognising standard views of the heart in ultrasound images does in many ways resemble the general problem of recognising 3D objects from 2D images. But there are some significant differences that need to be taken into consideration when designing the heart view recognition system.

While the general object recognition problem deals with images of object surfaces, ultrasound images are cross-sections of objects, in this case hearts. We would like to classify our images into six different classes, the five standard views and one class for non-standard images. So we define the different views as different objects, even though they are all really images of the same object, the heart.

The general object recognition problem looks at static objects, which the heart is not. The heart's shape is constantly changing with chambers expanding or contracting and valves moving.

The goal of the general object recognition problem is a system invariant to rotation, translation and scaling of the object, in addition to change in illumination and background. The general system should also be able to recognise an object that is occluded by another object. The requirements to the special heart view recognition system are not all the same. Since the heart is in constant change of shape, and the fact that no two human hearts are exactly the same, some rotation and translation of the object must be accepted by the system, but not an arbitrary change. Some scaling must also be accepted. Ultrasound imaging does not involve illumination changes. Still, the ultrasound images often differ in grey levels and contrast, and the heart view recognition system have to handle this. Ultrasound images are very noisy, so the system needs to be noise tolerant. On the other hand, in a standard image of the heart there should be no occlusions. Occlusion of the heart by the lung, for instance, is unacceptable. The tolerance goals of the general object recognition problem versus those of the special heart view recognition are summarized in table 2.1.

Tolerance to	*General problem*	*Standard views*
Rotation	Any	”Some”
Translation	Any	”Some”
Scaling	Any	”Some”
Illumination change	Almost any	Not comparable
Noise/Background	Almost any	Almost any
Occlusion	Almost any	None

Table 2.1: Tolerance goals of recognition systems

The heart view recognition system should ultimately help ensure an acceptable quality of ultrasound recordings. For it to be of any real help to someone inexperienced with the ultrasound scanner, only the images that are up to standard should be classified as standard views. In section 1.1, it was stated that quality assurance is not a requirement to this thesis work. In other words, referring to table 2.1, the design of the heart view recognition system should in this thesis aim at fulfilling the tolerance goals regarding rotation, translation, scaling and noise.

Chapter 3

Related work

A considerable amount of research has been done on identifying representations that are invariant to rotation, translation, scale changes, illumination changes etc. In the following section three classes of methods for solving the general object recognition problem are presented. If nothing else is stated, the information in this section and the next is taken from [24]. Section 3.2 takes a closer look at one of the classes: recognition as a correspondence of local features, and in section 3.3 it is decided which algorithm to use as a basis for this thesis work.

3.1 Classes of object recognition methods

Methods of object recognition can be classified according to a number of characteristics. Here, focus is on knowledge representation, learning, and invariance to image formation conditions. Historically there have been two main trends: the model-based approach, and the appearance-based approach. In model-based object recognition, the knowledge of object appearance is explicitly provided as a CAD-like model. The appearance-based object recognition, on the other hand, requires no explicit object model. Object representations are rather mostly acquired through an automatic learning phase. Recently, a new class of methods has emerged which put local patches into correspondence. Objects are represented by appearance of small local elements, and the learning of models is automatic.

3.1.1 Model-based recognition

Model-based methods are also called shape- or geometry-based methods. As the names imply, information about the object, the model, is represented explicitly. Two representations are needed: one of the object model and one of the image content. Ideally, there is a simple relation between the two representations. If for example a wireframe model is used to describe the object, and linear

intensity edges are used to describe the image, each edge can be matched directly to one of the model wires.

Recognition can be interpreted as deciding whether a given image can be a projection of the known model of an object. To achieve pose and illumination invariants, the model description, or primitives, should be somewhat invariant with respect to changes in these conditions. Much effort has been directed to identify such invariant primitives[53, 39]. In [3], Baldock uses explicit models in the interpretation of echocardiogram images. Robinson, Colchester and Griffin present a high level symbolic model of the human brain in [41], together with a method for using the model to aid in the recognition of objects from medical images.

The model-based approaches have some disadvantages:

- They depend on reliable extraction of geometric primitives, like lines or circles
- The interpretation of the detected primitives can often be ambiguous, like for instance if primitives that are not modelled are present
- The modelling capabilities are restricted only to classes of objects which are composed of few easily detectable elements
- The models must be created manually

3.1.2 Appearance-based methods

Having seen all possible appearances of an object, can recognition be achieved by just efficiently remembering all of them? This is the central idea behind the appearance-based approach. The methods usually include two phases. In the first phase, a set of reference images is used to construct models of objects. The reference set includes the appearance of each object under different conditions when it comes to orientations and illumination. The images can be efficiently compressed using for instance Principal Component Analysis. In the second phase, parts of the input image are extracted and compared to the reference images.

The appearance-based approach has been successfully demonstrated for scenes with unoccluded objects on a black background[31]. Swain and Ballard proposed in [49] to represent an object by a colour histogram. Objects are identified by matching histograms of image regions to histograms of model images. This is a robust technique when it comes to object orientation, scaling and occlusion, but it does not perform well for objects that can not be identified by colour alone. The concept of histogram matching was later generalized by Schiele and Crowley[46]. They used responses of various filters, instead of pixel colours, to form the histograms.

The appearance-based methods are attractive since they do not require geometric primitives to be detected and matched. They do however require isolation of

the complete object of interest from the background. This is a major limitation, because it means that they are sensitive to occlusion and require good segmentation. A number of attempts have been made to address this problem[18, 5, 34].

To summarize, the main disadvantages of the appearance-based methods are:

- They require dense sampling of training views
- They are generally not very robust to occlusion and background clutter

3.1.3 Recognition as a correspondence of local features

Neither model-based nor appearance-based methods do well according to the requirements to an object recognition system - generality, robustness and easy learning - defined in section 2.4.1. Model-based approaches can usually handle only objects consisting of simple geometric primitives, meaning that they are not general. They also require the user to specify the object models, and so they do not support easy learning. Appearance-based methods demand an exhaustive set of learning images, taken from densely distributed views and illuminations. To obtain such a set, the object must be observed in a controlled environment, like for instance on a turntable. Both approaches are sensitive to occlusion of the objects, and to the unknown background. They are therefore not robust.

Methods based on matching of local features have been proposed as an attempt to address these issues. Objects are now represented by a set of local features, automatically computed from the training images. These features are organised into a database, and this constitutes the learning phase. Recognition of objects in an image starts by performing the exact same extraction of local features on this image. Features similar to these are then searched for in the database, and the number of local correspondences decides if an object is present or not.

It is not required that all local features match, so the approach is robust to occlusion and cluttered background. The local approach can also recognise objects from different views. Complex variations in object appearance can be modelled by simple transformations at a local scale, like for instance affine transformations. By allowing such simple transformations, significant viewpoint invariance is achieved even for objects with complicated shapes. The construction of object models is done automatically from images depicting the object, so no user intervention is required except for providing the training images.

Edges and corners are properties that can be detected and analyzed in an image. Several recognition approaches based on local features have been proposed. According to Lowe [22], the development of image matching by a set of local keypoints can be traced back to the work of Moravec[30]. He used a corner detector to select interest points. This corner detector was improved by Harris and Stephens[14] to make it more repeatable under small image variations and near edges. Since then, the Harris corner detector has been used for many other image matching tasks[22]. It was Schmid and Mohr[47] who showed that

invariant local feature matching could be extended to general image recognition problems, where features are matched against a database of images. They too used the Harris corner detector to detect interest points. However, the Harris corner detector is very sensitive to changes in image scale and does not perform well when the task is to match images of different sizes. Lowe extended the local feature approach to achieve scale invariance in [21]. He proposed a new local descriptor that provided more distinctive features and were less sensitive to local image distortions such as 3D viewpoint change.

3.1.4 Evaluation of approaches

We have seen in section 3.1.1 that the model-based approach is often used in recognition of human organs. But as mentioned in section 2.4.2, the heart is not a static object - it is constantly changing in size and shape. So making an explicit model of the heart is complicated. Another important disadvantage of the geometry-based methods is their dependency on reliable extraction of geometric primitives. As illustrated in section 2.3, the ultrasound images are quite noisy. Extracting for instance the contour of the chambers in a reliable way is a difficult task.

It seems that the appearance-based approach is more suitable for a heart view recognition system that the model-based approach. The appearance-based methods do not require explicit object models; neither do they require detection and matching of geometric primitives. The fact that they have low robustness to occlusion does in fact only make them more suitable to the application, again referring to table2.1. But, as stated in section 3.1.2, the appearance-based methods require good segmentation, which is difficult to achieve automatically in ultrasound images.

The methods of correspondence of local features are also based on appearance, meaning that there is no need to extract geometric primitives. Also, segmentation of objects from background is not required. These methods in addition have an advantage in that complex variations in object appearance are approximated by simple transformations at a local scale. The appearance-based approach seems like the best approach for recognition of standard views in ultrasound images, and this is the approach that will be followed.

3.2 More on correspondence of local features

Several approaches based on local features have been proposed, and generally they all follow this structure:

1. **Detection:** First, image elements of ”interest” are detected. An image element is of interest if it depicts a part of an object and can be repeatedly detected in images taken over a large range of conditions. The elements

of are often called keypoints, and the descriptors of local appearance are computed at these locations.

2. **Description:** The local appearance in the neighbourhood of the elements of interest is then used to make a descriptor, also called a feature vector. When designing a descriptor, there are several aspects that must be taken into account. The descriptor should be discriminative enough to distinguish between features of the different objects stored in the database. The descriptor should also be robust to variations in an object's appearance that are not reflected by the detector.

3. **Indexing:** The descriptors are stored into a database, and this is the learning of object models. In the recognition phase, descriptors are first computed on the input image. The database is then looked up for similar descriptors, or potential matches. Indexing means organising the database in a way that allows for efficient retrieval of similar descriptors. What makes up a suitable indexing structure generally depends on the properties of the descriptors.

4. **Matching:** After computing local features on the same form as for the database image, tentative correspondences are established for every feature detected in the input image. The database is searched, and typically Euclidean distance between the input feature and the stored features is evaluated. If the closest feature is close enough, it is considered a match.

5. **Verification:** The similarity of the descriptors is not necessarily a measure reliable enough to guarantee a correct match. The final step of the recognition process is a verification of presence of the model in the input image. Often a global transformation connecting the model and the input image is estimated.

3.3 Choice of method

Many different approaches to recognition as a correspondence of local features have been proposed in litterature [9, 25, 27, 32, 21, 22, 43, 45, 44, 52, 51], and with them many different descriptors. One well known descriptor is the Scale Invariant Feature Descriptor (SIFT)[22, 21], developed by Lowe in 1999. It detects points of interest with invariance to scale, rotation and translation.

Schmid and Mikolajczyk have evaluated the performance of five different interest point descriptors[28]. The criterions used were that the descriptors should be distinctive and at the same time robust to changes in viewing conditions as well as to errors of the point detector. The result of the evaluation was that the SIFT descriptor performs best.

A great advantage of Lowe's object recognition system is that it emphasizes efficiency, and achieves near real-time recognition times.

Because of the efficiency and documented performance of Lowe's SIFT algorithm, the choice is made to use it as a basis for recognition of standard views in ultrasound images of the heart. Even though focus here is not on run time, it is nice to know that it is possible to achieve near real-time performance. Due to the differences in requirements to a general recognition system and a specialized heart view recognition system, given in table 2.1, the original SIFT algorithm must be altered to fulfil the requirements of the special problem.

Chapter 4

Recognition with SIFT features

David Lowe proposes the Scale Invariant Feature Transform(SIFT) in [21]. As the name suggests, the algorithm transforms image data into scale-invariant coordinates relative to local features. An approach to using the SIFT features for object recognition is described in [22], where Lowe uses the features to perform matching between different images of the same object or scene.

The extracted features are invariant to image scale and rotation, and provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise and change in illumination. In addition, the features are highly distinctive, which means that a single feature can be correctly matched with high probability against a large database of features.

For object recognition, SIFT features are first extracted from a set of training images and stored in a database. To match a new image, each extracted feature from the new image is individually compared to the features stored in the database.

The original SIFT algorithm for matching different views of an object is described in section 4.1. The section is a summary of the theory in [21] and [22]. A new algorithm that uses the SIFT features for heart view recognition is then given in section 4.2. This algorithm uses the original features, but the matching approach is specialized at recognising standard views of the heart.

4.1 Lowe's SIFT algorithm

The description of Lows's algorithm for matching different views of an object is given in two parts: feature extraction in subsection 4.1.1, and feature matching in subsection 4.1.2. More details are given on feature extraction, since the original features are kept in the new algorithm. Lowe's feature matching is more briefly described, for reference and comparison.

4.1.1 Feature extraction

A set of local image features is extracted from each image. Each feature contains a record of its:

- Position, or pixel location (x, y), in the image
- Orientation, the dominant orientation of the image structure in the neighbourhood
- Scale, represented by standard deviation σ
- Description of the local image structure, encoded in terms of gradient histograms

The major stages of computation used to generate the set of features are:

1. **Scale-space extrema detection:** The algorithm searches over all scales and image locations to find so-called interest points that are invariant to scale and rotation. These interest points represent minima and maxima in a created difference-of-Gaussian pyramid, and are considered keypoint candidates.
2. **Keypoint localization:** At each interest point location, a detailed model is fit to determine the point's location and scale. Instable keypoint candidates are rejected.
3. **Orientation assignment:** Based on local image gradient directions, one or more orientations are assigned to each keypoint location. By performing all future operations on image data that has been transformed relative to the assigned orientation, scale, and location of each keypoint, invariance to these transformations is achieved.
4. **Keypoint description:** The keypoint descriptor vector is formed from a grid of gradient histograms constructed from a neighbourhood around the keypoint.

Scale-space extrema detection

Interest points for SIFT features correspond to local maxima and minima of difference-of-Gaussian images at different scales.

It was Witkin that showed, in [54], that detecting locations that are invariant to scale change of the image can be accomplished by searching for stable features across all possible scales, using a continuous function of scale known as scale-space. Scale-space of an image, $L(x, y, \sigma)$, is defined as:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where $*$ is the convolution operator, $I(x, y)$ is the input image, and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

is a variable-scale Gaussian.

To efficiently detect stable keypoint locations in scale-space, Lowe proposes using scale-space extrema in the difference-of-Gaussian function convolved with the image, $D(x, y, \sigma)$. This can be computed from the difference of two nearby scales separated by a constant k:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned}$$

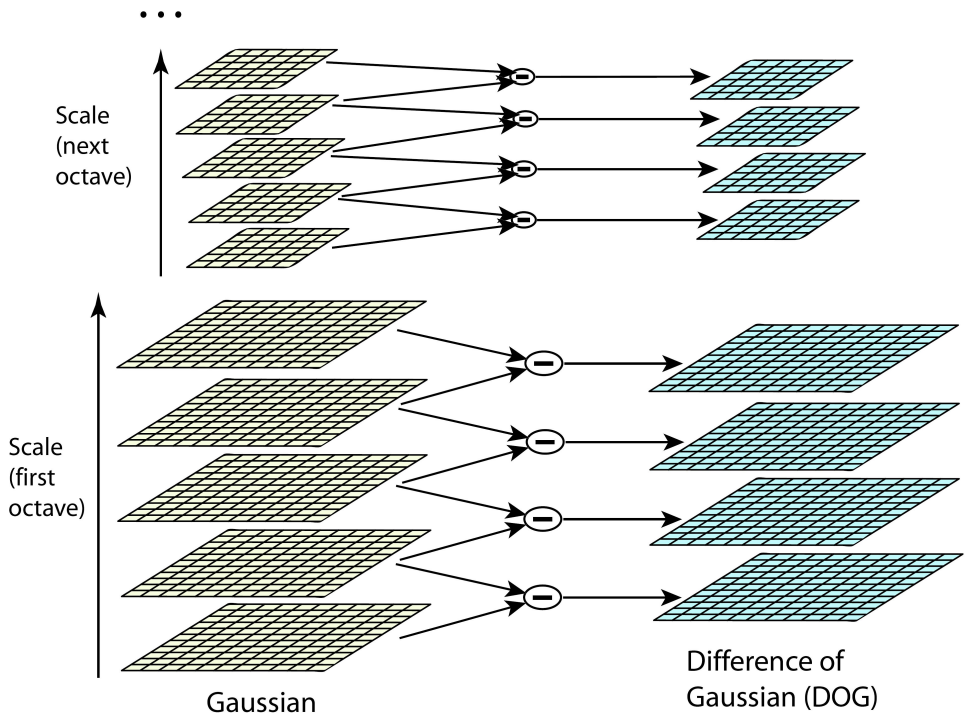


Figure 4.1: Construction of difference-of-Gaussian images[22]

The construction of $D(x, y, \sigma)$ is illustrated in figure 4.1. The input image is incrementally convolved with Gaussians. This produces images separated by a constant k in scale space, shown in the left stacks. The convolved images are grouped by octave, where an octave corresponds to doubling the value of σ. Each octave is divided into an integer number, s, of intervals, so that $k = 2^{1/s}$. For the final extremal detection to cover a complete octave, it is necessary to produce $s + 3$ Gaussian blurred images in the left stacks. The Gaussian image with twice the initial value of σ is the image two images from the top of the

stack. After processing a complete octave, a new image is made from resampling this image to half its size. The new image becomes the first image of the next octave. The right stacks show the difference-of-Gaussian images, the result of subtracting adjacent image scales in the left stack.

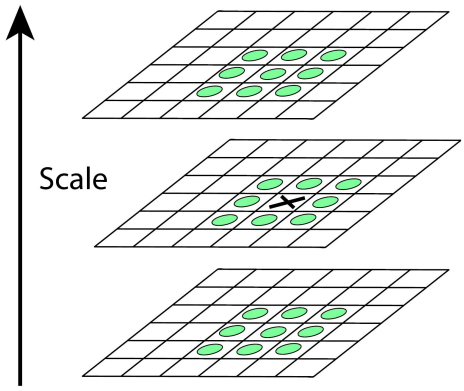


Figure 4.2: Pixel comparisons to detect maxima and minima of difference-of-Gaussian images[22]

In order to detect the local extrema of $D(x, y, \sigma)$, each sample point is compared to its eight neighbours in the current image and its nine neighbours in the scale above and below. This is illustrated in figure 4.2. A point is selected as a candidate keypoint only if it is larger than all of these neighbours or smaller than all of them.

Keypoint localization

The next step is to fit the candidate keypoints to the nearby data to accurately determine their positions. Also, keypoints with low contrast are removed, and responses along edges are eliminated.

The method used in [22] was developed by Brown[6] in 2002. The idea is to fit a 3D quadratic function to the local sample points to determine the interpolated location of the maximum or minimum. Brown's method uses the Taylor expansion of the scale-space function, $D(x, y, \sigma)$, shifted so that the origin is at the sample point:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^\mathbf{T}\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}, \tag{4.1}$$

where D and its derivatives are evaluated at the sample point and $\mathbf{x} = (x, y, \sigma)^T$ is the offset from this point. The location of the extremum, $\hat{\mathbf{x}}$, can be determined by taking the derivative of (4.1) with respect to $\mathbf{x}$ and setting it to zero. This gives:

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}}. \tag{4.2}$$

The derivative of D is approximated by using differences of neighbouring sample points. If the offset $\hat{\mathbf{x}}$ is larger than 0.5 in any dimension, it means that the extremum lies closer to a different sample point. In this case, the sample point is changed and the interpolation is performed about that point instead. The final offset $\hat{\mathbf{x}}$ is added to the location of its sample point to get the interpolated estimate for the location of the extremum.

The function value at the extremum, $D(\hat{\mathbf{x}})$, can be used to reject unstable extrema with low contrast. It can be obtained by substituting equation (4.2) into (4.1):

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\hat{\mathbf{x}}.$$

Lowe uses a threshold of 0.03 in [22], meaning that all extrema with a value of $|D(\hat{\mathbf{x}})|$ less than 0.03 are discarded.

It is not sufficient to reject keypoints with low contrast to ensure stability. The difference-of-Gaussian function will have a strong response along edges. This also counts for those edges that are poorly determined in location, and therefore unstable to small amounts of noise. A poorly defined peak in the difference-of-Gaussian function will have a large principal curvature across the edge, but a small one in the perpendicular direction. To compute the principal curvature, the 2x2 Hessian matrix, H can be used:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}.$$

$\mathbf{H}$ is computed at the location and scale of the keypoint, and the derivatives are estimated by taking differences of neighbouring sample points.

The eigenvalues of $\mathbf{H}$ are proportional to the principal curvatures of D. It is possible to avoid explicitly computing the eigenvalues, since it is only their ratios that are of interest. If α is defined to be the eigenvalue with the largest magnitude and β the smaller one, the sum of the eigenvalues can be computed from the trace of $\mathbf{H}$. Likewise, the product of the eigenvalues can be computed from the determinant:

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta.$$
$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

Let r be the ratio between the largest magnitude eigenvalue and the smaller one. Then $\alpha = r\beta$, and

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}. \tag{4.3}$$

Equation (4.3) depends only on the ratio of the eigenvalues, and not their individual values. $(r+1)^2/r$ has its minimum value when $\alpha = \beta$ and it increases with r. So, to check that the ratio of principal curvatures is below some threshold, r, it is only necessary to check

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r}.$$

Lowe uses a value of $r = 10$, meaning that keypoints that have a ratio between the principal curvatures greater than 10 are eliminated.

Orientation assignment

The next step is to assign a consistent orientation to each keypoint based on local image properties. This way, the keypoint descriptor can be represented relative to this orientation, and therefore achieve invariance to image rotation.

The Gaussian smoothed image, L, with the closest scale to that of the keypoint is selected. For each image sample, $L(x, y)$, at this scale, the gradient magnitude, $m(x, y)$, and orientation, $\theta(x, y)$, is precomputed using pixel differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

An orientation histogram is now formed from the gradient orientations of sample points within a region around the keypoint. The orientation histogram is divided into 36 bins, which cover the 360 degree range of orientations. When a sample is added to the histogram, it is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint.

Peaks in the orientation histogram correspond to dominant directions of local gradients. The highest peak in the histogram is detected and the value of the histogram bin is assigned as orientation to the keypoint. Any other local peak that is within 80% of the highest peak is used to also create a keypoint with that orientation. So, for locations with multiple peaks of similar magnitude, there will be multiple keypoints created at the same location and scale, but with different orientations. Finally, a parabola is fit to the three histogram values closest to each peak to interpolate the peak position for better accuracy.

Keypoint description

Image location, scale and orientation have been assigned to each keypoint. These parameters impose a local 2D coordinate system in which to describe the local image region, and therefore provide invariance to these parameters. The next step is to compute a descriptor for the local image region.

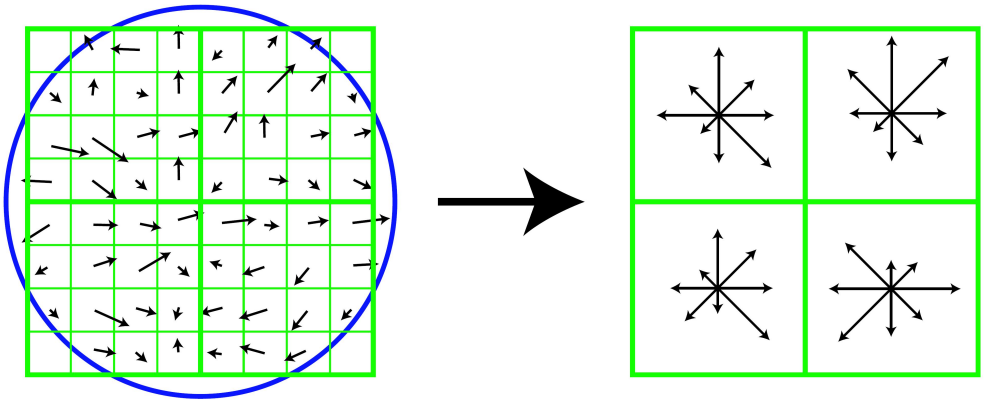


Figure 4.3: Computation of the keypoint descriptor[22]

Figure 4.3 illustrates the computation of the keypoint descriptor. The scale of the keypoint is used to select the level of Gaussian blur for the image. The image gradient magnitudes and orientations are then sampled around the keypoint location. To achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint orientation. The gradient are illustrated with small arrows at each sample location in the left of figure 4.3. Again, the contribution of each pixel is weighted by the gradient magnitude, and by a Gaussian with σ equal to 1,5 times the scale of the keypoint. This is illustrated with a circular window on the left side of the figure. With the Gaussian window, less emphasis is given to gradients that are far from the centre of the descriptor, because these are most affected by misregistration errors. The purpose of the Gaussian window also helps to avoid sudden changes in the descriptor with small changes in the posision of the window. The resulting keypoint descriptor is illustrated in the right side of figure 4.3. The figure shows eight directions for each orientation histogram. The length of each arrow corresponds to the magnitude of that histogram entry.

The descriptor is formed from a vector containing the values of all the orientation histogram entries, corresponding to the lengths of the arrows on the right side of figure 4.3. Whereas the figure shows a 2x2 array of orientation histograms, Lowes experiments showed that the best results are achieved with a 4x4 array of histograms with 8 orientation bins in each. This results in a 4x4x8 = 128 element feature vector for each keypoint.

4.1.2 Feature matching

Object recognition is performed by matching each keypoint independently to the database of keypoints extracted from the training images. Many of the initial matches will be incorrect due to ambiguous features or features that arise from background clutter. Therefore, clusters of at least 3 features are first identified that agree on an object and its pose. Then, each cluster is checked by performing a detailed geometric fit to the model, and the result is used to accept or reject the interpretation.

For each keypoint, the best candidate match is found by identifying its nearest neighbour in the database of keypoints from training images. The nearest neighbour is defined as the keypoint with minimum Euclidean distance for the invariant descriptor vector. To discard features that do not have a good enough match to the database, the distance of the closest neighbour to that of the second-closest neighbour is compared. This performs well because correct matches need to have the closest neighbour significantly closer than the closest incorrect match to achieve reliable matching. Lowe rejects all matches in which the distance ratio is greater than 0.8 in his implementation.

To increase recognition, the Hough transform is used to identify clusters of matches that vote for the same object pose. Each keypoint votes for the set of object poses that are consistent with the keypoint's location, scale and orientation. When clusters of features are found to vote for the same pose of an object, the probability of the interpretation being correct is much higher than for any single feature. Each of the keypoints specifies 4 parameters: 2D location, scale, and orientation, and each matched keypoint in the database has a record of the keypoint's parameters relative to the training image in which it was found. It is therefore possible to create a Hough transform entry predicting the model location, orientation, and scale from the match hypothesis.

After using the Hough transform to identify all clusters with at least 3 entries in a bin, a verification step matches the hypothesized object, or pose, to the image using a least-squares fit to the hypothesized location, scale, and orientation of the object. The final decision to accept or reject a model hypothesis is based on a detailed probabilistic model given in another paper by Lowe[19].

4.2 Recognition of standard heart views with SIFT

In [22], Lowe uses the SIFT features to perform matching between images of different views of one object. In this thesis, I propose an algorithm that uses the same SIFT features to perform matching between images of different objects, but of the same class, also called classification. Another important difference is that of the input data; ultrasound images are cross-sections, not surface reflections. Also, the objects of interest are dynamic, not static.

The original SIFT features are kept in the new algorithm. Just like Lowe's

algorithm, the new matching algorithm also searches for the nearest neighbours in the database. But apart from that, the matching method proposed by Lowe is not used. The new matching algorithm is fairly simple to understand and to implement. It is based on the tolerance goals of the specialized heart view recognition system, given in table 2.1.

Lowe wishes to make a system that tolerates any rotation, translation and scaling, and his matching algorithm is specialized at matching any two views of the same object. With the Hough transformation, votes are given to certain combinations of rotation, translation and scaling. The goal is to find clusters of the same combination of rotation, translation and scaling, that is clusters of the same transformation. How a point is transformed is not the issue as much as the fact that many points are transformed in the same way.

The heart view recognition system, on the other hand, should only tolerate some rotation, translation and scaling. Instead of using the Hough transform to vote for transformations, focus is on distances in position, orientation and scale for each keypoint and its corresponding nearest neighbour.

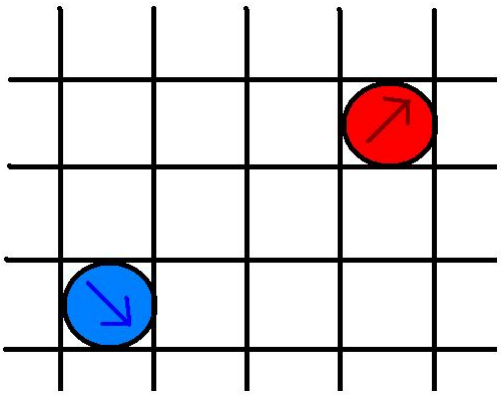


Figure 4.4: Keypoint and nearest neighbour

Figure 4.4 shows a keypoint in blue, and its corresponding nearest neighbour in red. The orientations of the points are illustrated with arrows.

Figure 4.5 illustrates what is meant by distance in position between the points. Distance in position, called *dist_pos*, is defined as

$$dist\_pos = \sqrt{(xpos\_k - xpos\_n)^2 + (ypos\_k - ypos\_n)^2},$$

where *x_posk* and *y_posk* are the coordinates of the keypoint, and *x_posn* and *y_posn* are the coordinates of the nearest neighbour.

Figure 4.6 illustrates what is meant by distance in orientation. Distance in orientation, *dist_ori*, is given by

$$dist\_ori = \sqrt{(ori\_k - ori\_n)^2},$$

where *ori_k* is the orientation of the keypoint, and *ori_n* is the orientation of the nearest neighbour.

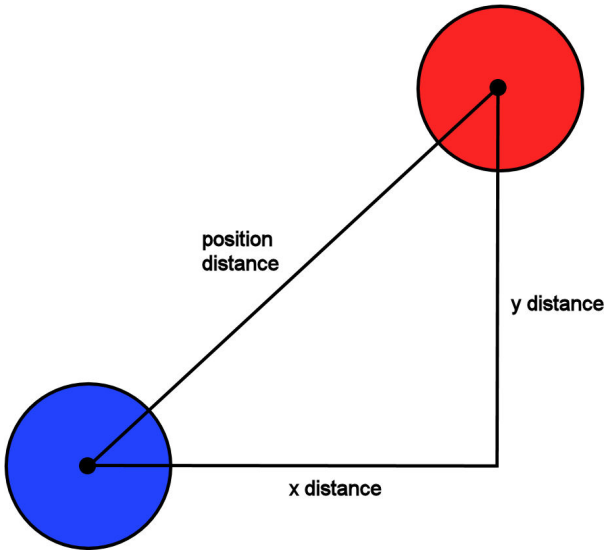


Figure 4.5: Position distance

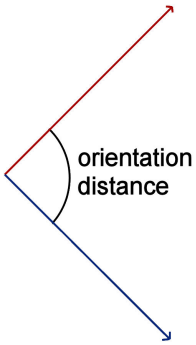


Figure 4.6: Orientation distance

Difference in scale, *dist_sca*, is defined as

$$dist\_sca = \sqrt{(sca\_k - sca\_n)^2},$$

where sca_k is the scale of the keypoint, and *sca_n* is the scale of the nearest neighbour.

Distance thresholds are defined, meaning that if the distances in position, orientation, and scale are all under their respective thresholds, a match between the keypoint and its neighbour is found. If one of the distances is greater than the corresponding threshold value, the keypoint is rejected.

The matches are then used to vote for standard image classes. When all matched keypoints have voted for their classes, the votes are counted. The class with the highest number of votes wins, but the number has to exceed a certain threshold for the winning class to be accepted.

The new matching algorithm is given here in pseudo code. The algorithm takes

as input:

- Keypoints, described by position, $(xpos\_k, ypos\_k)$, orientation, ori_k, and scale, sca_k
- Corresponding nearest neighbours, described by position, $(xpos\_n, ypos\_n)$, orientation, ori_n, and scale, sca_n
- Maximum allowed distance in position between keypoints and nearest neighbours, (pos_thresh)
- Maximum allowed distance in orientation between keypoints and nearest neighbours, (ori_thresh)
- Maximum allowed distance in scale between keypoints and nearest neighbours, (sca_thresh)
- Weighting factor for distance in position, (pos_weight)
- Weighting factor for distance in orientation, (ori_weight)
- Weighting factor for distance in scale, (sca_weight)
- Minimum number of votes for an image class to be accepted, $(votes\_thresh)$

The algorithm outputs a label that is either "two-chamber", "four-chamber", "apical long axis", "parasternal long axis", "parasternal long axis", or "no match".

Algorithm 1 MATCH-KEYPOINTS-WITH-NEAREST-NEIGHBOURS

for all keypoints k with nearest neighbour n **do**
    $dist\_pos \leftarrow \sqrt{(xpos\_k - xpos\_n)^2 + (ypos\_k - ypos\_n)^2}$
    $dist\_ori \leftarrow \sqrt{(ori\_k - ori\_n)^2}$
    $dist\_sca \leftarrow \sqrt{(sca\_k - sca\_n)^2}$
    if $dist\_pos < pos\_thresh$ **then**
        if $dist\_ori < ori\_thresh$ **then**
            if $dist\_sca < sca\_thresh$ **then**
                $pos\_vote \leftarrow pos\_weight \cdot (1 - (pos\_dist/pos\_thresh))$
                $ori\_vote \leftarrow ori\_weight \cdot (1 - (ori\_dist/ori\_thresh))$
                $sca\_vote \leftarrow sca\_weight \cdot (1 - (sca\_dist/sca\_thresh))$
                $vote = pos\_vote + ori\_vote + sca\_vote$
                accumulate the neighbour's image class with $vote$
            end if
        end if
    end if
end for
if winning image class has number of votes $> votes\_thresh$ **then**
    $label \leftarrow$ image class with max votes
else
    $label \leftarrow$ "no match"
end if
return $label$

Chapter 5

Implementation

This chapter treats the implementation of the heart view recognition system. Section 5.1 lists the tools that were used during the implementation, while section 5.2 concerns the image data used. A brief description of the MATLAB code is given in section 5.3, and section 5.4 describes some experiences with the implementation. The last section treats performance testing of the algorithm.

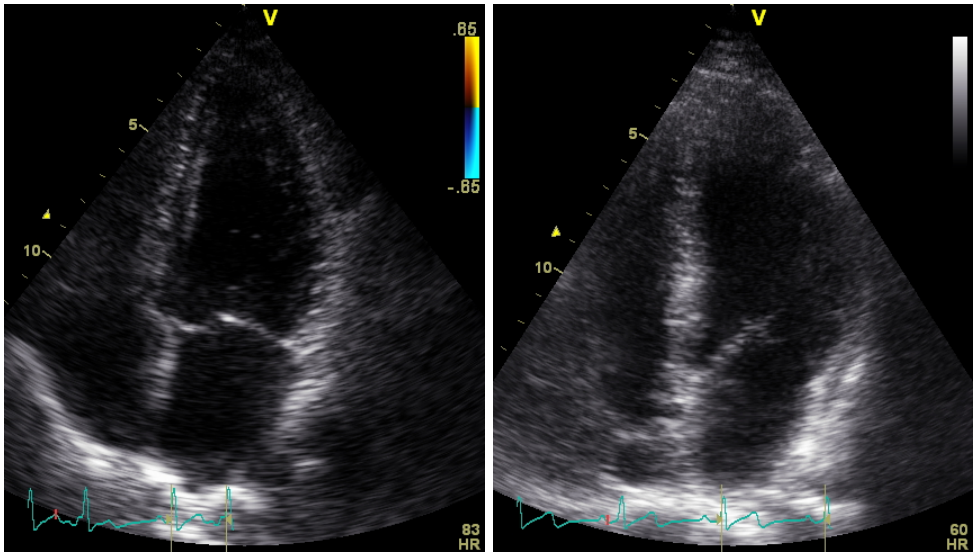
5.1 Tools

Ultrasound scanner The ultrasound images were recorded using a Vivid 7 ultrasound scanner, produced by GE Vingmed AS (Horten, Norway).

GcMat v 093.b GcMat is a program that makes it possible to show and manipulate ultrasound recordings from GE Vingmed's ultrasound scanners in MATLAB. GcMat exists for the time being as an internal product at NTNU. The program is developed by Department of Circulation and Medical Imaging in co-operation with GE Vingmed AS.

MatLab v 7.0 The ultrasound recordings were transferred to a PC and manipulated in MATLAB version 7.0, release 14 made by the MathWorks. Frame by frame were extracted from the recordings, and saved in the JPEG format.

According to the thesis' requirements, the object recognition system is currently implemented in MATLAB. This is the usual practice at the Department of Circulation and Medical Imaging. When applications are fully developed, they have to be implemented as C-code in order to run on GE Vingmed's ultrasound scanners. This is also the case for the heart view recognition system.



(a) Type 1 (b) Type 2

Figure 5.1: Examples of saved JPEG images

5.2 Image data

Most of the ultrasound recordings used were acquired by Dr. med. Asbjørn Støylen. This was a set consisting of recordings from 12 different patients. A small set of recordings were also acquired by Svein Arne Aase and Brage Høyem Amundsen. This set consisted of recordings from 2 different patients.

The recordings are of varying quality. They were viewed in MATLAB, and those of acceptable quality for the recognition task were picked out. These were saved as JPEG images, one image for each frame in the recording. It is assumed that the small potential data loss when converting to JPEG does not infect the recognition results.

Figure 5.1 gives two examples of typical JPEG image. In addition to the actual B-mode data, the images also contains some graphical objects and numbers. For one group of the obtained recordings, all these extras are coloured, see figure 5.1(a), while for the rest they are as shown in figure 5.1(b). The reason for the colour differences is not relevant to this thesis. But the extras are unwanted data for the training images, and the images had to be pre-processed before feature extraction. This was done by taking advantage of the colouring, by setting all pixels that were coloured to black. Only images of type 2 were used for training images, so that they would be free from all this unwanted data.

5.3 MATLAB code

There code were organized as three main procedures: extracting the SIFT features, adding the features to the database, and matching extracted features to features in the database.

5.3.1 Feature extraction

Before computing the features, some pre-processing had to be done on the original images. To get rid of all coloured noise, the images were transformed from RGB (Red, Green, Blue) representation to HSV (Hue, Saturation, Value) representation, and all pixels with saturation over a certain threshold were set to zero (black). After being transformed back from HSV to RGB, histogram equalization was performed on the images. As a last pre-processing step, the images were blurred a bit to prevent antialiasing. Figure 5.2 shows an example of an original image, and figure 5.3 shows an example of a pre-processed image.

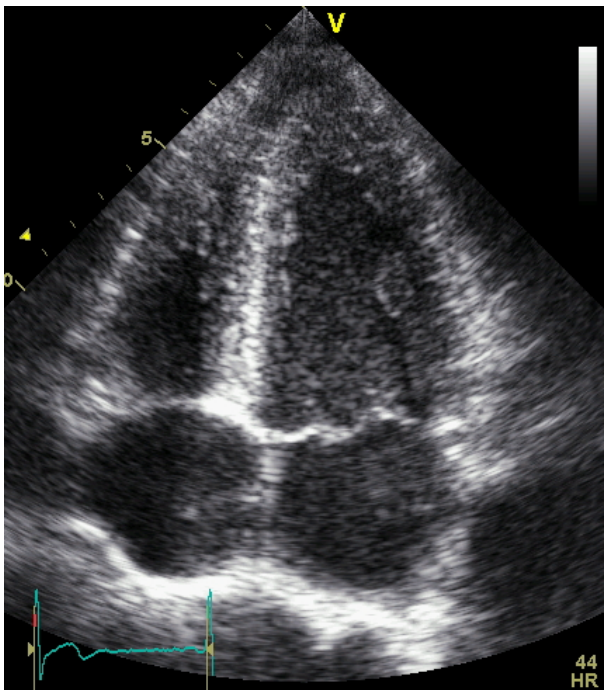


Figure 5.2: Original image

The first image of first octave of the Gaussian pyramid was blurred with a standard deviation named $first\_sigma$, given by

$$first\_sigma = \sqrt{sigma^2 - antialias\_sigma^2}.$$

Here, $sigma$ is the standard deviation used when computing the Gaussian pyramid, while $antialias\_sigma$ is the standard deviation used during the initial

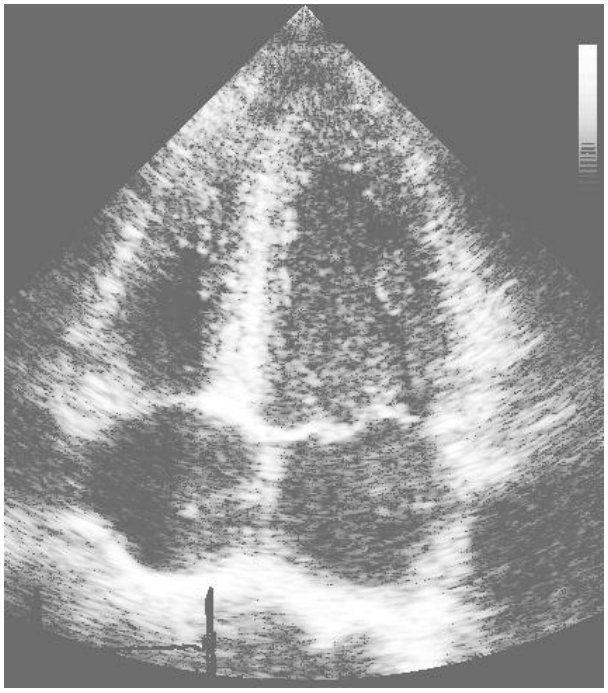


Figure 5.3: Pre-processed image

blurring. For every succeeding level of the octave, a new image was made by blurring the previous image with a Gaussian kernel with standard deviation of $\sqrt{k^2 - 1}$, where $k = 2^{1/(s)}$. After producing $s + 3$ images in the first octave, a new octave was started by downsampling the image 3 images from the top with a factor of 2. This image was used as the first image of the next octave, and the blurring procedure continued.

When all octaves of the Gaussian pyramid were made, neighbouring images were subtracted to obtain a difference-of-Gaussian pyramid.

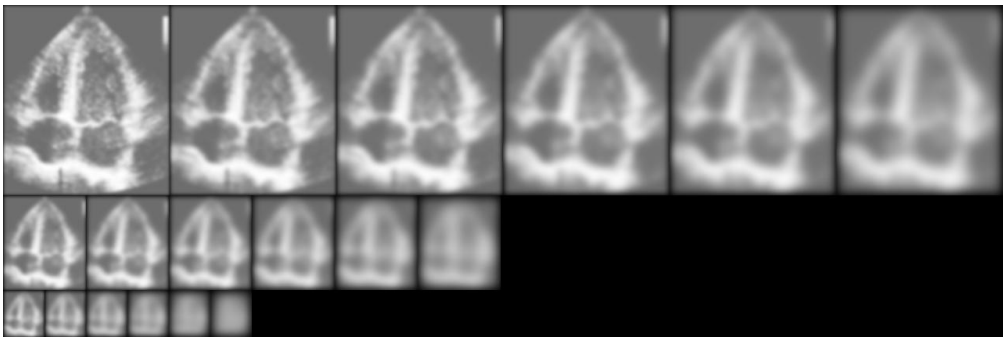


Figure 5.4: Gaussian pyramid

Figure 5.4 shows the Gaussian pyramid, and figure 5.5 shows the difference-of-Gaussian pyramid computed for the input image in figure 5.3.

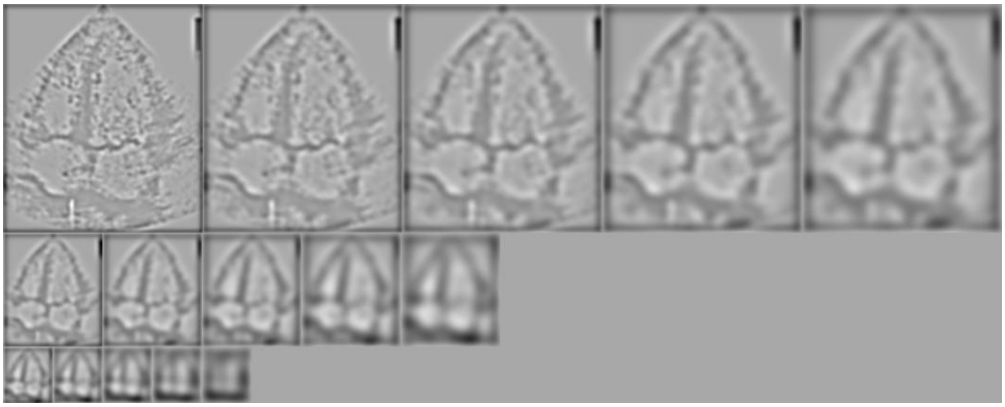


Figure 5.5: difference-of-Gaussian pyramid

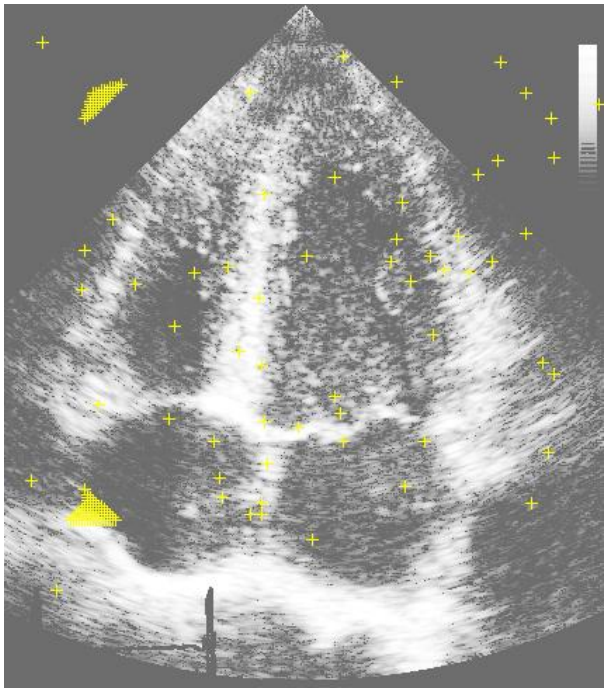


Figure 5.6: Maxima and minima in difference-of-Gaussian pyramid

To detect keypoints, the difference-of-Gaussian pyramid was searched for points that were maxima or minima compared to their neighbours in a 3x3 cube around them. All maxima and minima then had to be checked against a threshold for contrast, *constrast_thresh*, and a threshold for curvature, *curvature_thresh*. Figure 5.6 shows maxima and minima detected in the difference-of-Gaussian image. Figure 5.7 shows keypoint candidates after thresholding on contrast, and figure 5.8 shows the final keypoints after thresholding on ratio of principal curvatures.

To assign orientation to keypoints, a region defined by a Gaussian window around each keypoint was sampled for gradient orientation. Magnitude and

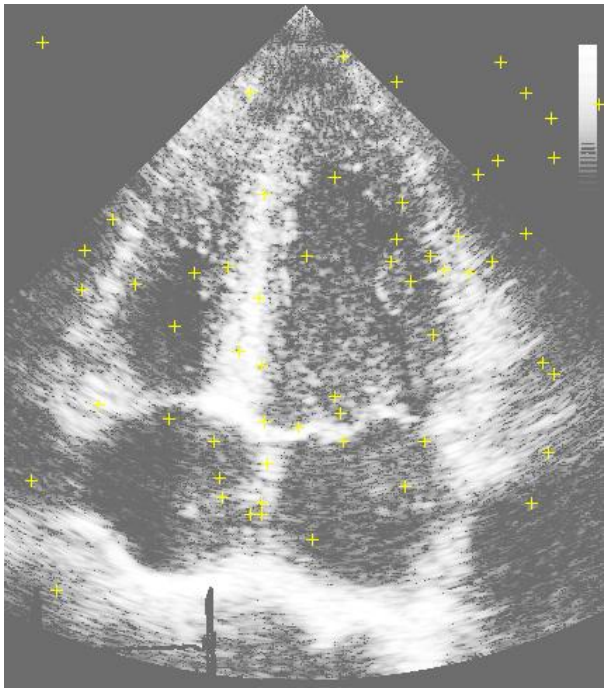


Figure 5.7: Keypoints after thresholding on contrast

orientation for the gradients were approximated using pixel differences. An orientation histogram was formed, and entries into bins were weighted with magnitude and the Gaussian window. Peaks were found in the histogram, and the largest peak and every peak within 0,8 times the size of the largest peak were picked out. A parabola was fit to interpolate each peak position for these, using the formula

$$max = -B/2A$$

to find max or min in the parabola

$$y = Ax^2 + Bx + C$$

Figure 5.9 shows keypoint orientations after defining orientations, and figure 5.10 shows the defined orientations for the working example.

Histograms were used again to compute the descriptions. The scale of the keypoints were used to select level of Gaussian blur, and image gradient orientations and magnitudes were sampled in a region around the keypoint. The gradient orientations were rotated relative to the keypoint orientations and orientation histograms with 8 bins were formed in 4x4 arrays. The entries into the histogram were weighted with gradient magnitude in addition to a Gaussian window. A 1x128 descriptor was formed from the values of all histogram entries.

Table 5.1 summarizes the parameters in use during extraction of SIFT features.

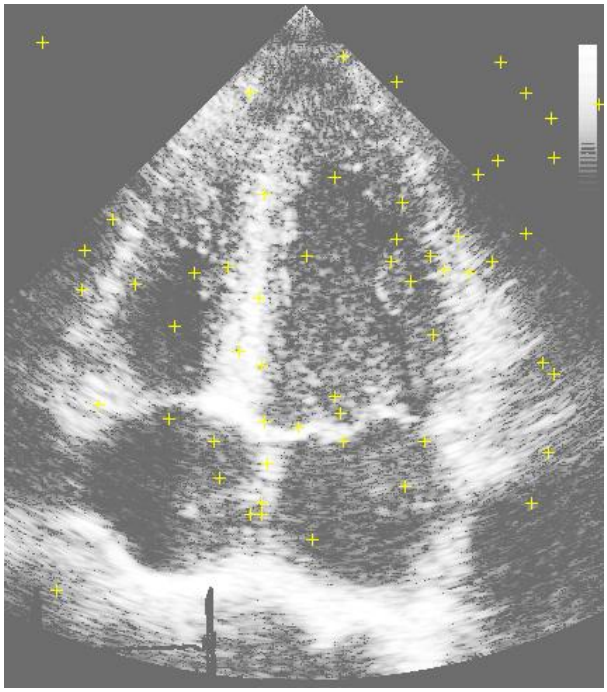


Figure 5.8: keypoints after thresholding on ratio of principal curvatures

Parameter	*Explanation*
sat_thresh	threshold value for eliminating color pixels
octaves	number of octaves in keypoint pyramid
s	number of scales per octave in keypoint pyramid
sigma	σ for Gaussian blurring, to be doubled for each octave
antialias_sigma	σ for pre-blurring of the first image of the first octave
contrast_thresh	threshold value to be used when thresholding on contrast
curvature_thresh	threshold value to be used when thresholding on curvature

Table 5.1: Parameters used during features extraction

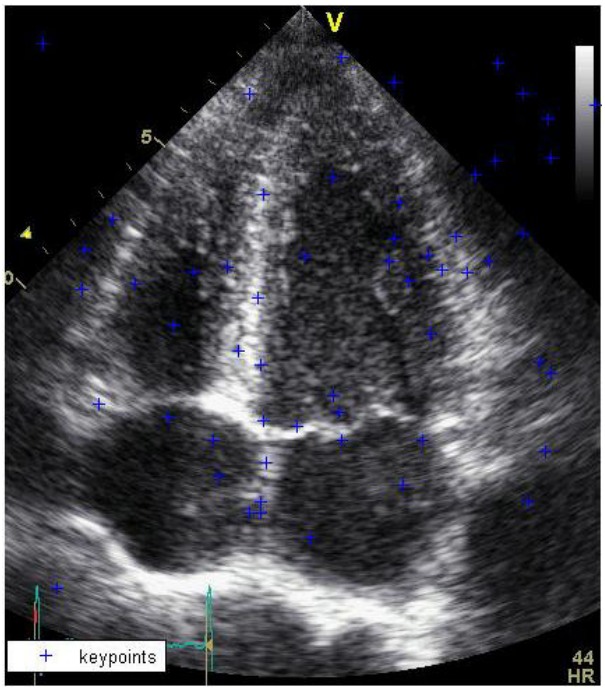


Figure 5.9: Keypoints after computing the orientation

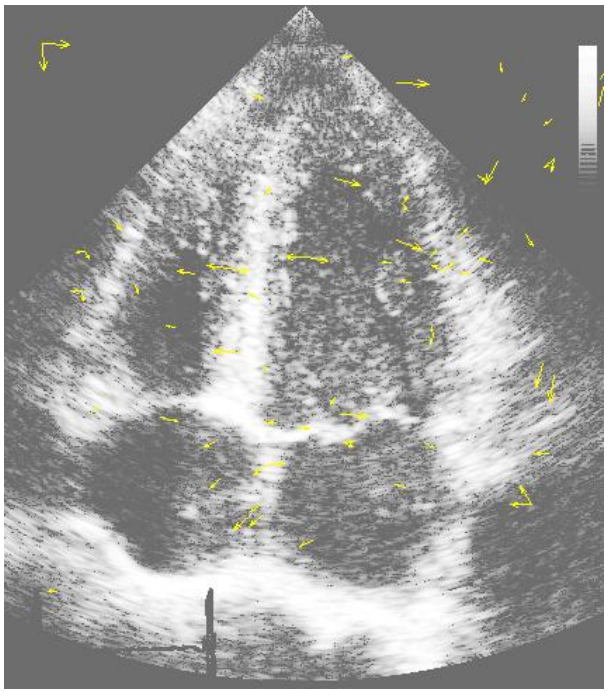


Figure 5.10: Orientations assigned to keypoints

5.3.2 Construction of database

The computed SIFT features for the training images were added to a database. The database was represented with a MATLAB Structure. The database fields used can be seen in table 5.2.

Field	*Explanation*
image	The original training image
index	Index of the image in database
position	Position of keyoints in original image
scale	Scale of keypoints
orientation	Orientation of keypoints
standard	Standard of training image
description	The 1x128 descriptor vector

Table 5.2: Database fields

5.3.3 Matching

Euclidean distance was used to decide on nearest neighbours in the database. For each keypoint, its 1x128 description vector was compared to each of the 1x128 descriptor vectors in the database. Some of the nearest neighbours were rejected as matches when checking the distance of the closest neighbour to that of the second-closest neighbour. A distance threshold, *distance_ratio* decided the allowed distance.

Figure 5.11 shows keypoints and corresponding nearest neighbours.

To find keypoints and nearest neighbours that matched well, the differences in position, orientation and scale were computed according to the algorithm given in section 4.2. During the voting on standards, a 1x5 vector was used to keep track of the votes for each standard: votes for two-chamber views were added to the first entry, votes for four-chamber views to the second entry, etc. Abbrevations used are defined in appendix A. The vector is shown in (5.1).

$$\begin{bmatrix} 2ch \\ 4ch \\ aplax \\ plax \\ sax \end{bmatrix} \tag{5.1}$$

The position of the maximum value of the vector gave a hypothesis for standard view of the input image. If the maximum value was equal to or greater than a predefined threshold, *min_votes*, the hypothesized standard was accepted. If the maximum value was lower than the threshold, the algorithm returned with ”no match”. Figure 5.12 shows the keypoints, matched neighbours and non-matched neighbours of the example, while figure 5.13 shows some of the

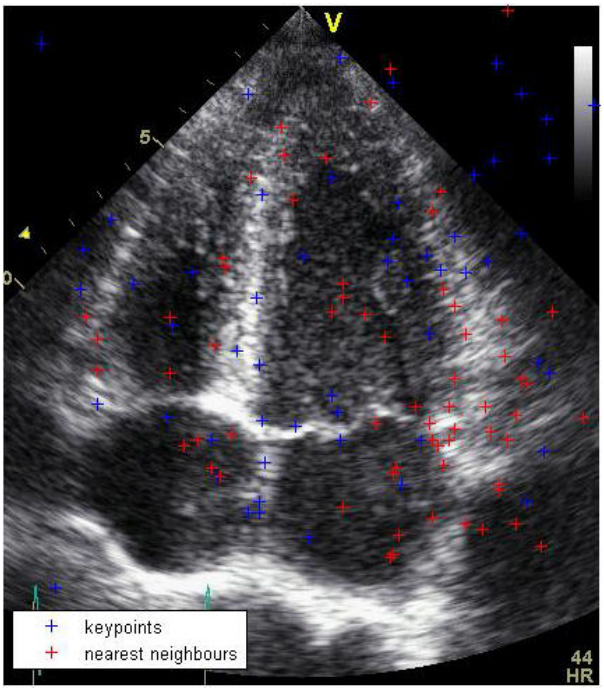


Figure 5.11: Keypoints and corresponding nearest neighbours

MATLAB print out when running the algorithm. The votes and the chosen label can be seen.

Parameter	*Explanation*
pos_thresh	max difference in position between keypoint and neighbour
ori_thresh	max difference in orientation between keypoint and neighbour
sca_thresh	max difference in scale between keypoint and neighbour
pos_weight	weight given to position difference when voting for standard
ori_weight	weight given to orientation difference when voting for standard
sca_weight	weight given to scale difference when voting for standard
min_votes	minimum number of votes for the hypotesized standard to be accepted

Table 5.3: Parameters used during feature matching

Table 5.3 summarizes the parameters used during feature matching, also explained in section 4.2.

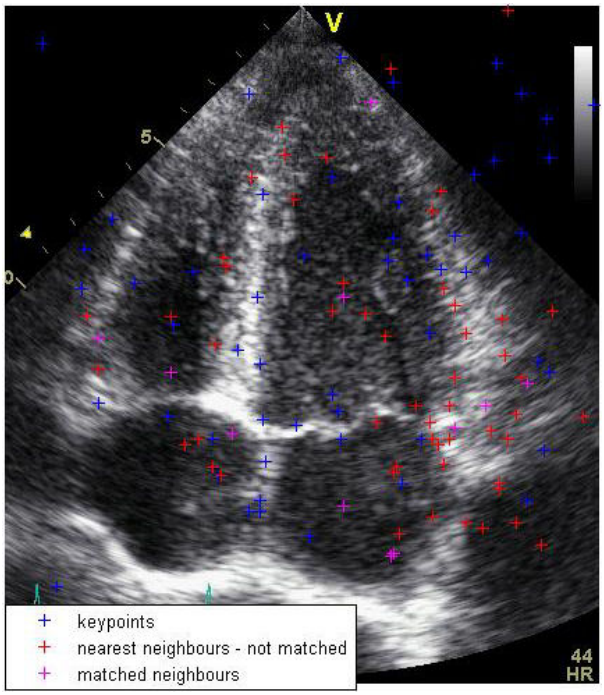


Figure 5.12: Keypoints, matched neighbours and non-matched neighbours

```
standard =

   48.3224
   26.5912
   14.8188
   22.7975
   19.9332


label =

4ch

>>
```

Figure 5.13: Print out from MATLAB: votes and chosen label

5.4 Experiences with the implementation

Lowe has released a demo version of the SIFT algorithm [20]. The source code for SIFT feature extraction is not provided. Only compiled binary programs are provided for this, in addition to a MATLAB script and C source code for performing simple matching between images. During the first implementation stages of the thesis, I considered using the binaries released by Lowe to extract SIFT features. That way I could focus all my attention at developing the specialized matching algorithm. But after some thought, I decided to implement my own SIFT code, mainly because I did not want to be fully dependent of Lowe's implementation.

During the implementation I realized that the algorithm given by Lowe in [21] and [22] was not always clear and complete, and that I had to make some choices of my own. Apart from the demo version released by Lowe, I came across several other implementations of the SIFT feature extraction, e.g. [33] and [12]. When struggling to understand the algorithm definition of the articles, it could be of great use to see how others had solved it. However, the original algorithm given in [21] in 1999 was updated in [22] in 2004, and these implementations were often of the original algorithm. And of course, none of the implementations I saw used a matching method that resembled mine.

The major difficulty with the implementation was that the SIFT algorithm, and also my own matching algorithm, have so many free parameters. Tuning the parameters to get good classification results was hard, especially since they affected each other.

5.5 Performance testing

To test the performance of the final heart view recognition system, two tests were done on two different image sets.

The first test was performed with the image data received from Dr. Med. Asbjørn Støylen. This data was divided into a training set and a test set. The training set contained 60 images from 5 different patients. These images can be viewed in A.1. Images were picked from different stages in the cardiac cycle, with the objection cover so much of the range of appearances for each image class as possible. The test set for Test 1 contained 30 images from 6 different patients. These images can also be seen in A.2.

The second test was a smaller one, performed with image data received from Svein Arne Aase and Brage Høyem Amundsen as test set, but with the same training set as used during the first test. This test set containing 8 images from 1 patient can be viewed in A.3.

Parameters used during testing are listed in table 5.4. The test results are listed in chapter 6, and discussed in 7.

Parameter	*Value*
sat_thresh	0.09
octaves	3
s	3
sigma	1.0
antialias_sigma	1.8
contrast_thresh	0.0005
curvature_thresh	30
pos_thresh	80
ori_thresh	$\pi/2$
sca_thresh	6
pos_weight	1
ori_weight	2
sca_weight	2

Table 5.4: Parameters used during testing

Chapter 6

Test results

This chapter states the results of the tests given in section 5.5. The results are discussed in chapter 7.

The abbreviations used in this chapter are defined in appendix A.

6.1 Results Test 1

Table 6.1 shows the results for Test 1. 26 of the 30 test images were given correct labels, illustrated in green in the the third column.

A summary of the results is given in table 6.2. The 26 hits correspond to a hit rate of 87%.

Image	*View*	*Label*
1	2ch	2ch
2	2ch	2ch
3	2ch	2ch
4	2ch	2ch
5	2ch	2ch
6	2ch	2ch
7	4ch	4ch
8	4ch	4ch
9	4ch	4ch
10	4ch	plax
11	4ch	4ch
12	4ch	4ch
13	aplax	4ch
14	aplax	aplax
15	aplax	aplax
16	aplax	aplax
17	aplax	aplax
18	aplax	aplax
19	plax	plax
20	plax	plax
21	plax	plax
22	plax	plax
23	plax	plax
24	plax	plax
25	sax	sax
26	sax	2ch
27	sax	sax
28	sax	sax
29	sax	2ch
30	sax	sax

Table 6.1: Results Test 1

	Number	*Percentage*
Hits	26	87%
Misses	4	13%

Table 6.2: Summarized results for Test 1

6.2 Results Test 2

Table 6.3 shows the results for Test 2. Here, 6 of the 8 test images were given correct labels, illustrated in green in the third column.

Image	*View*	*Label*
1	4ch	4ch
2	4ch	4ch
3	4ch	4ch
4	4ch	4ch
5	4ch	4ch
6	4ch	4ch
7	4ch	aplax
8	4ch	aplax

Table 6.3: Results Test 2

A summary of the results is given in table 6.4. The 6 hits correspond to a hit rate of 75%.

	Number	*Percentage*
Hits	6	75%
Misses	2	25%

Table 6.4: Summarized results for Test 2

Chapter 7

Discussion and future work

7.1 Results Test 1

For Test 1, there were 4 misses out of 30:

- One four-chamber image was classified as parasternal long axis
- One apical long axis image was classified as four-chamber
- Two parasternal short axis images were both classified as two-chamber images

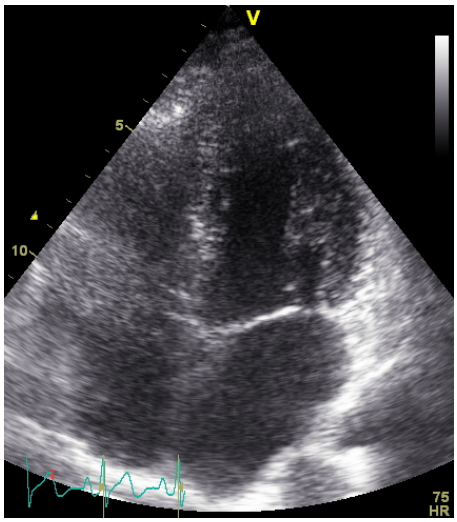
Some output images for these wrong classifications are given in figure 7.1 to 7.4.

The first miss was for the four-chamber image in figure 7.1(a), which was labelled ”plax”. Taking a closer look at the input image in figure 7.1(b), it can be noticed that the septum is barely visible. This might have something to do with the bad result for this test image. Without the septum, the image might look a bit like a parasternal long axis image. It can be seen in figure 7.1(c) that there are not many matches in the region around the septum.

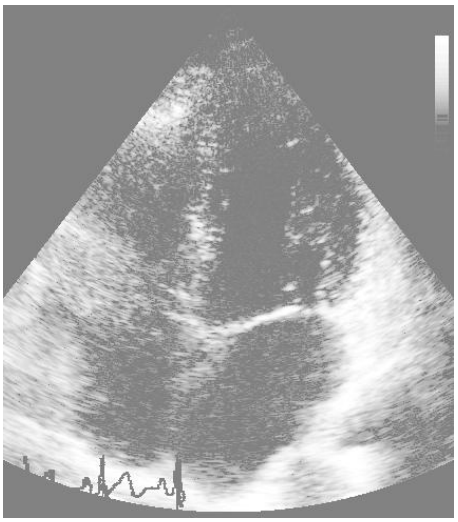
The second miss was an apical long axis image being classified as a four-chamber image. Taking a look at the original image in figure 7.2(a), it might resemble a four-chamber image that has been pulled down a bit . This might lead to the conclusion that the threshold for position distance has not been properly set, so that too much translation is allowed.

Maybe the right classification for this image is not even apical long axis view. The ventricle is really too far to the right of the image for the image to be ”standard” apical long axis, so the image could also be classified as ”non-standard” (or ”no match”). But as previously stated, quality insurance is not in focus here.

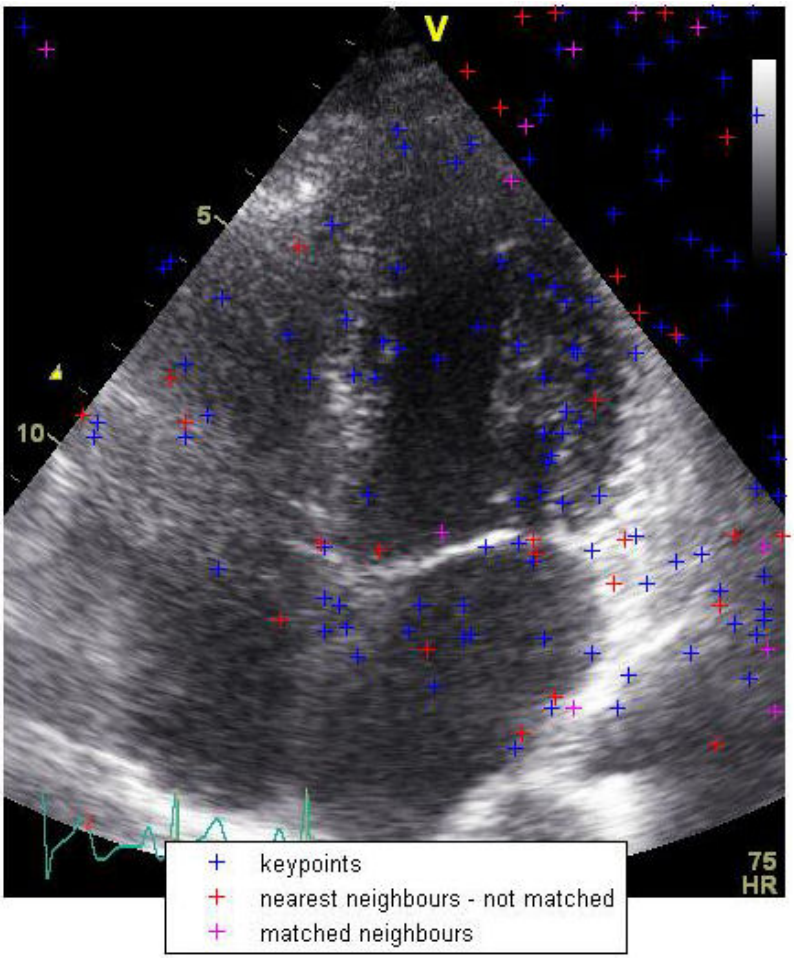
Both of the parasternal short axis views that were wrongly classified were classified as two-chamber views. This might not just be a coincidence. The images do resemble two-chamber views that have shrunk towards the middle of the image.



(a) Original image

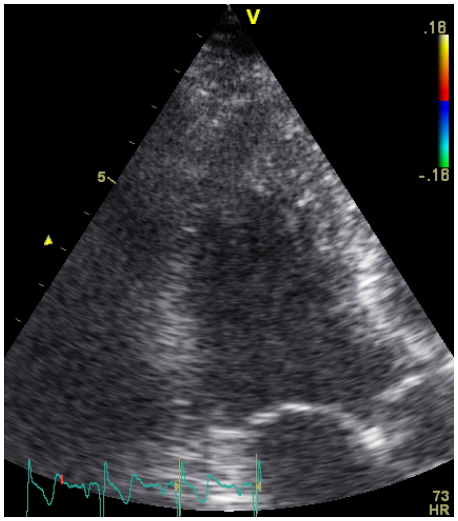


(b) Input image

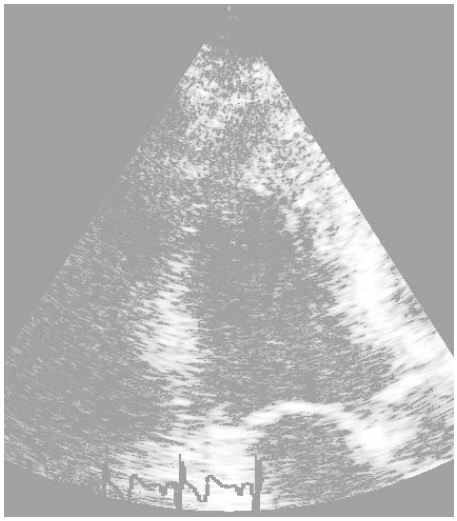


(c) keypoints, matched neigbours and non-matched neighbours

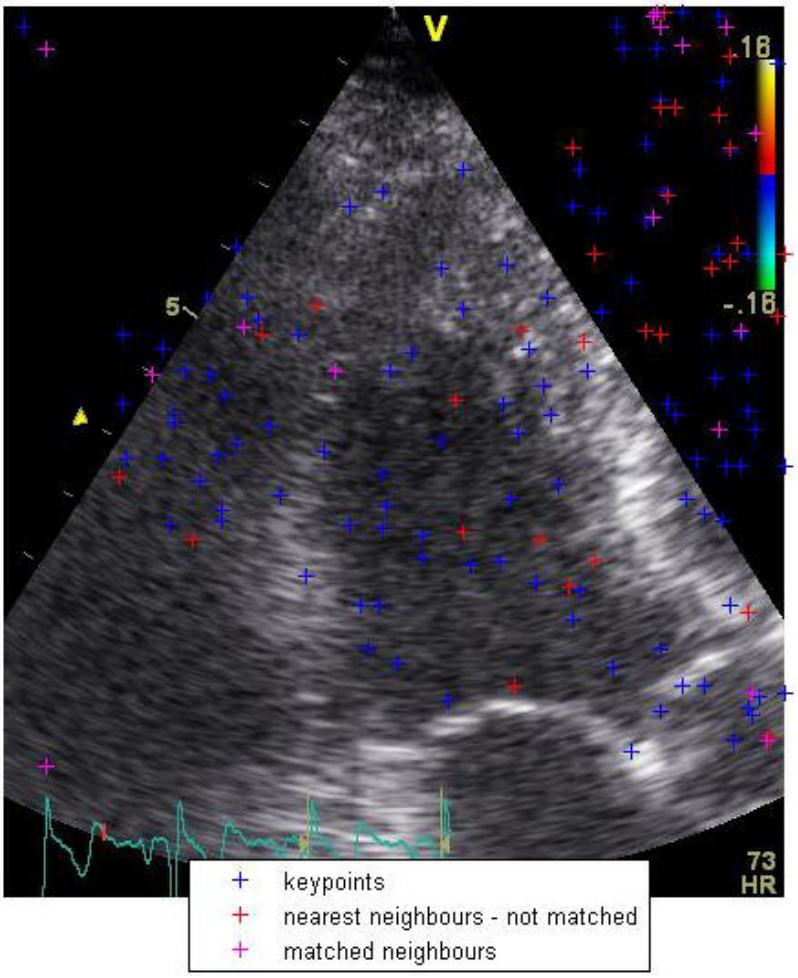
Figure 7.1: Four-chamber classified as parasternal long axis



(a) Original image

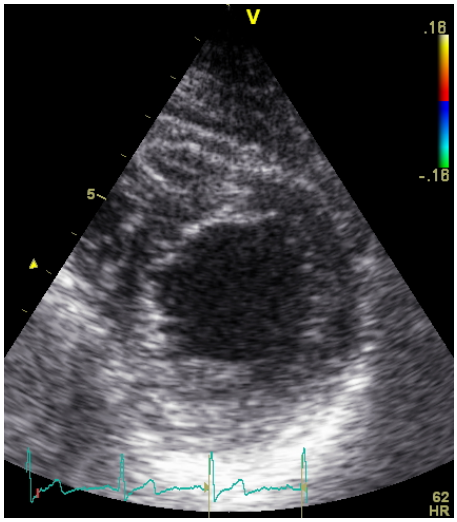


(b) Input image

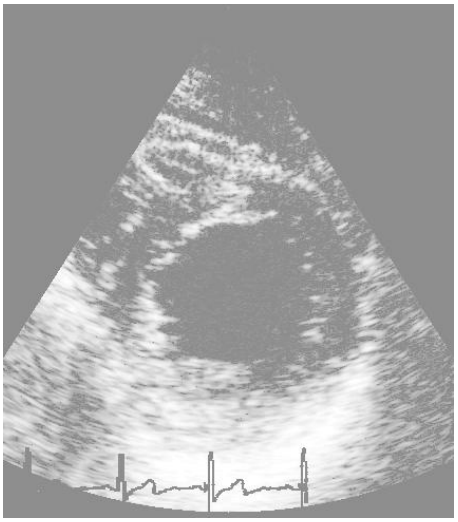


(c) keypoints

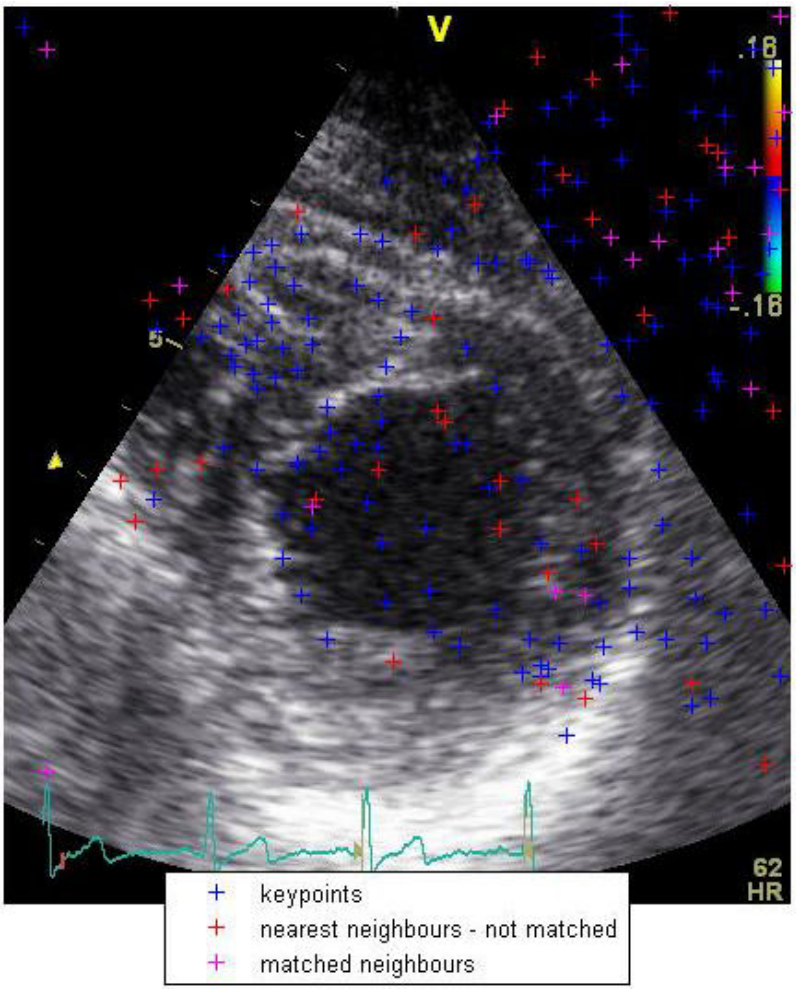
Figure 7.2: Apical long axis classified four-chamber



(a) Original image



(b) Input image



(c) keypoints, matched neigbours and non-matched neighbours

Figure 7.3: Parasternal short axis classified as two-chamber

This might also suggest that too much translation is tolerated, and therefore that the *pos_thresh* is not properly set. Looking at the original images overlaid with keypoints, matched neighbours and non-matched neighbours, in figures 7.3(c) and 7.4(c), one big difference between the two can be noticed. In the last one, there are almost no matches inside the image data sector.

Few matched features inside the image data sector is common to all four misclassified images of Test1, compare for instance figures 7.1(c) to 7.4(c) to figure 5.12. The number of matched features depend on many factors, such as the parameters *sigma*, *antialias_sigma*, *contrast_thresh*, *cuvature_thresh*, *pos_thresh*, *ori_thresh* and *sca_thresh*.

26 out of 30, or 87% were correctly classified during Test 1. This is not an extremely good result, but it does show that the method using SIFT features to classify heart view images has potential.

7.2 Results Test 2

The purpose of Test 2 was to test another aspect of the algorithm. Now the training images and the test images used were recordings achieved by two different medical experts, testing the generality of the algorithm.

For Test 2, there were 2 misses out of 8. All the test images here were four-chamber views, and both of the images that were wrongly classified were classified as apical long axis views.

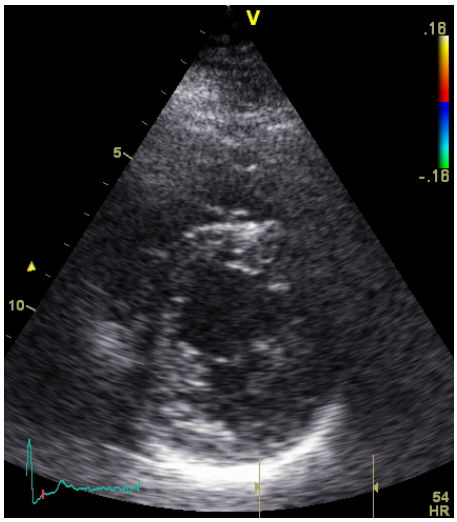
The reason that only 8 test images were used during test 2, all from one patient, was that those were the suitable images that were available at the moment of the testing. Because so few were used, it is difficult to draw conclusions from this test. But 6 hits out of 8, or a hit rate of 75%, suggests that the algorithm does not perform as well when trained with images recorded by a different medical expert.

7.3 SIFT and heart view classification

Test 1 and 2 suggest some improvements that should be done to the algorithm.

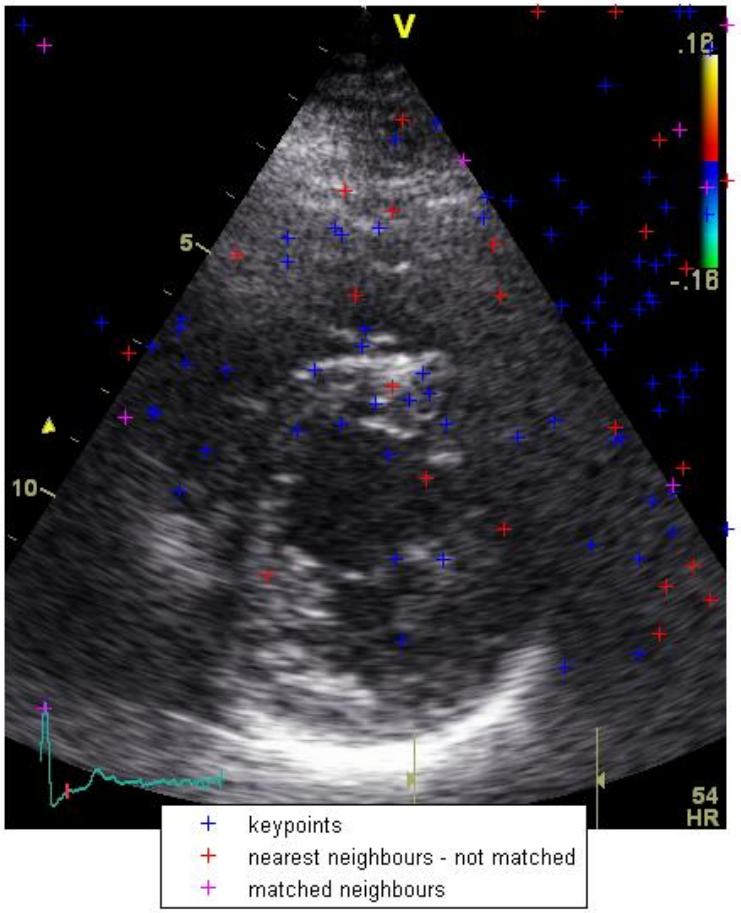
Setting proper parameters is hard. The testing of different parameters could be done in a more systematic way, or perhaps an algorithm that searches for optimal parameters given training- and test sets could be developed. This could for instance be implemented as a genetic algorithm.

Certain images of one class resemble certain images of another class, like for instance an apical long axis view resembling a two-chamber view when both valves are closed. And an apical short axis view might look a bit like a shrunken two-chamber view. To overcome this problem, it might be an idea to use several succeeding frames in the recording when performing feature extraction, instead



(a) Original image

(b) Input image



(c) keypoints, matched neigbours and non-matched neighbours

Figure 7.4: Parasternal short axis classified as two-chamber

of just using one frame. By using several frames, it is possible to take advantage of the information about movement in the image. Movement is an important source of information when human experts classify ultrasound recordings.

Another idea when it comes to views that resemble each other is to give more points to ”rare” views. For example, if other views resemble the two-chamber view, two-chamber views should be given the least amount of points when finding matches between keypoints and neighbours labelled with standard ”2ch”. More points should be given to parasternal short axis and four-chamber views when such matches are found between points.

Getting the algorithm to reject images like the one in figure 7.2(a), discussed in section 7.1, might be possible if the database was trained with bad examples. This way, the algorithm might recognize an image that are not completely up to standard. It is much work covering all bad examples. But still, these are quite controlled environments, so it should be possible.

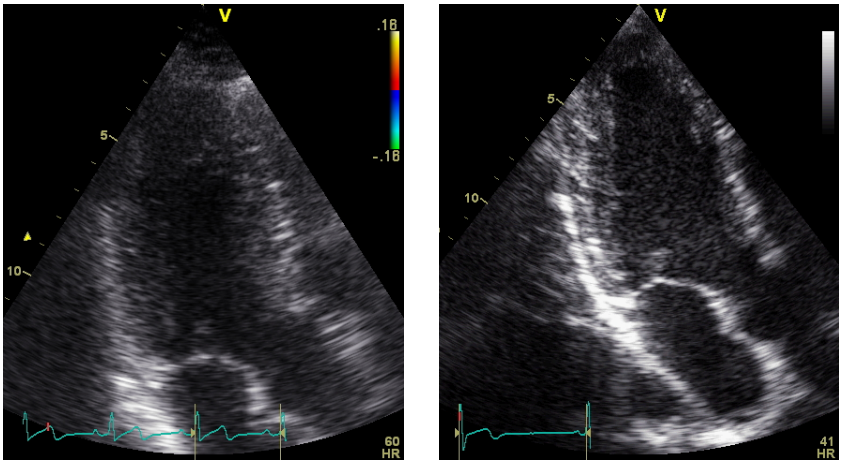
The pre-processing could have been better. Maybe another method for eliminating the unwanted data in the image should be used. Looking at the images of keypoints and matched neighbours, there are quite many keypoints and neighbours outside the image data sector. This is probably also caused by the blurring that draws the image data out. Optimally, the input images should be free from the clutter. It is possible to manually get the images without the extra objects. The problem was that I did not think of this possibility soon enough, and it is quite a demanding procedure.

One of the initial arguments for using the SIFT descriptor was that it was the one that performed best according to Schmid and Mikolajczyk[28]. Lately I discovered a new article by the same authors, where an improved SIFT descriptor outperforms the original SIFT descriptor[29]. I doubt that it is of any practical use for this thesis, but it might be worth taking a look at.

Perhaps it was not the best choice to use the exact same descriptor for classification as used for recognition of different images of the same object. There is a difference between capturing the important features of a particular object, and capturing the essence of a general class. Perhaps the feature description is too detailed, and that it should have been modified to better fit the classification problem.

Lowe says in his article [22] that ”another direction for future research will be to individually learn features that are suited to recognizing particular objects categories. This will be particular important for generic object classes that must cover a broad range of possible appearances.” Towards the end of this thesis work, I also found an article by Helmer and Lowe citeHelmer04 addressing classification using SIFT features. Their method recognizes an object class by learning a statistical model of the class. A probabilistic model decomposes the appearance of an object class into a set of local parts and models the appearance, relative location, co-occurrence, and scale of these parts. Perhaps a probabilistic model is a better approach for classification of standard views of the heart.

Some problems emerged while finishing the thesis. There were difficulties in obtaining the image data that should be used as training set and test set. I had received this data as video files early on, but had yet to extract frame by frame from the recordings. I had written and used an algorithm that could do the extraction for me. This algorithm had worked for all image data during the development phase, and I assumed that it would also work for this new image data. But for some mysterious reason, it did not. When my supervisor got back from his vacation, he found another way to extract the data, but this had already cost me many days. I should of course have checked on the image data earlier on. Because of this delay, I did not get to run as many tests as I would have hoped. In Lowe's demo version of the code, the feature extraction is implemented in C. My implementation is in MATLAB, and it is quite slow. Testing different parameters, for example, is time-consuming.



(a) Typical apical long axis image used (b) Standard apical long axis image

Figure 7.5

Another problem with the image data was that there was not enough of it. As a result of this, most of the two-chamber and apical long axis images used are not up to standard. In a real standard two-chamber or apical long axis image, the entire atrium should be included. Figure 7.5(a) shows a typical apical long axis image used during performance tesing, while figure 7.5(b) shows an apical long axis image up to standard.

It can be discussed if distinguishing between the images in figure 7.5(a) and figure 7.5(b) is within the scope of the thesis, or if it has to do with ensuring quality of ultrasound images. It is generally hard to determine what has to do with classification, and what has to do with quality assurance, and a strict division between the two seems a bit artificial.

7.4 Future work

The parameters of the heart view classification system have to be tuned, and a more thorough testing of the algorithm has to be performed. Testing should be performed with a larger set of training images, with images obtained by several medical experts, from several different patients. But before all this, I suggest implementing the SIFT feature extraction in C, which will make the testing much less time-consuming.

Perhaps also some adjustments should be made to the feature description, as discussed in section 7.3. In [22], Lowe also says: "An attractive aspect of the invariant local feature approach to matching is that there is no need to just select just one feature type, and the best results are likely to be obtained by using many different features, all of which can contribute useful matches and improve overall robustness".

If the approach to heart view classification suggested does not yield sufficient results still after thorough experimentation and testing, the probabilistic approach to classification with SIFT features seems like a natural next step.

Chapter 8

Conclusion

Despite some difficulties with image data and free parameters, the developed algorithm for recognizing standard views of the heart yields quite promising results.

During a performance test with 30 input images, 26 of these, or 87%, were correctly classified according to corresponding standard. For those images that were wrongly classified, it is possible to find some explanations for this that can be used in furter development of the algorithm. Among others, the results suggest that some of the free parameters were not properly tuned.

Results from a second test indicate that the training set should include images obtained by different medical experts. The test set contained images recorde by a different medical expert, and 6 out of 8 images were correctly classified here.

A more thorough testing and experimentation have to be done before it can be finally decided if the proposed algorithm should be the one implemented on the ultrasound scanner for recognition of standard views of the heart. Probabilistic classification with SIFT features could be another possible approach.

Bibliography

[1] `http://www.bmp.psu.edu/faculty/troyan/b4online/cardio/cardioheartutorial.htm`. Last accessed 04/07/2005.

[2] Bjørn A. J. Angelsen. *Ultrasound imaging*. Emantec, 2000.

[3] Richard A. Baldock. Trainable models for the interpretation of biomedical images. *Image and Vision Computing*, 10(6):444–450, July/August 1992.

[4] Jan G. Bjålie, Egil Haug, Olav sand, Øystein V. Sjaastad, and Kari C. Torverud. *Menneskekroppen*. Gyldendal Norsk Forlag AS, 2003.

[5] Michael J. Black and Allan D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26(1):63–84, 1998.

[6] M. Brown and D.G. Lowe. Invariant features from interest point groups. In *British Machine Vision Conference, Cardiff, Wales*, pages 656–665, 2002.

[7] Ruey-Feng Chang, Chii-Jen Chen, and Ming-Reng Ho. Breast Ultrasound Image Classification Using Fractal Analysis. In *4th IEEE International Symposium on BioInformatics and BioEngineering (BIBE 2004)*, pages 100–107. IEEE Computer Society, 2004.

[8] Sven J. Dickinson. *Rutgers University Lectures on Cognitive Science*, pages 172–207. Basil Blackwell publishers, 1999.

[9] Gyuri Dorkó and Cordelia Schmid. Selection of scale-invariant parts for object class recognition. In *International Conference on Computer Vision*, volume 1, pages 634–640, 2003.

[10] Henrik Egeblad. *Ekkokardiografi*. Lægeforeningens forlag København, 2001.

[11] Sandra Reynolds Grabowski and Gerard J. Tortora. *Prisciples of Anatomy & Physiology*. John Wiley & Sons, Inc., 2003.

[12] Christer Gustavsson, Anthony Hui, and Michael Turitzin. Sift project. `http://robots.stanford.edu/cs223b04/project9.html`. Last accessed 04/07/2005.

[13] Jøger Hansegård, Erik Steen, Stein Inge Rabben, Anders H. Torp, Hans Torp, Sigmund Frigstad, and Bjørn Olstad. Knowledge Based Extraction

of the Left Ventricular Endocardial Boundary from 2d Echocardiograms. In *2004 IEEE Ultrasonics Symposium*, 2004.

[14] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey88*, pages 147–152, 1988.

[15] Isabelle L. Herlin and Nicholas Ayache. Features extraction and analysis methods for sequences of ultrasound images. *Image and Vision Computing*, 10(10):673–682, December 1992.

[16] Sverre Holm. Medisinsk ultralydavbilding. 2000.

[17] Christopher O. Jaynes. Recent Advances in Computer Vision. *ACM Crossroads (the student magazine of the Association for Computing Machinery)*, 1996.

[18] Aleš Leonardis and Horst Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78(1):99–118, 2000.

[19] D. Lowe. Local feature view clustering for 3d object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, December 2001.

[20] David Lowe. Sift demo. `http://www.cs.ubc.ca/~lowe/keypoints/`. Last accessed 04/07/2005.

[21] David G. Lowe. Object recognition from local scale-invariant features. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, page 1150. IEEE Computer Society, 1999.

[22] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[23] Marcos Martín and Carlos Alberola. A Bayesian Approach to in vivo Kidney Ultrasound Contour Detection Using Markov Random Fields. In *Lecture Notes In Computer Science, Proceedings of the 5th International Conference on Medical Image Computing and Computer-Assisted Intervention-Part II*, pages 397–404. Springer-Verlag London, UK, 2002.

[24] Jiri Matas and Stepan Obdrzalek. Object recognition methods based on transformation covariant features. In *XII. European Signal Processing Conference EUSIPCO - 2004*, pages 1333–1336, September 2004.

[25] Jiri Matas, Stepán Obdrzálek, and Ondrej Chum. Local affine frames for wide-baseline stereo. In *ICPR (4)*, pages 363–366, 2002.

[26] Tim McInerney and Demetri Terzopoulos. Deformable Models. In Isaac Bankman, editor, *Handbook of Medical Imaging*, pages 127–145. Academic Press, 2000.

[27] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, pages 128–142. Springer-Verlag, 2002.

[28] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. In *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 257–263, June 2003.

[29] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. Accepted to PAMI, 2005.

[30] Hans Moravec. Rover visual obstacle avoidance. In *Proceedings of the seventh International Joint Conference on Artificial Intelligence*, pages 785–790, August 1981.

[31] Shree K. Nayar, Sameer A. Nene, and Hiroshi Murase. Real-Time100 Object Recognition System. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, 1996.

[32] Stepán Obdrzálek and Jiri Matas. Object recognition using local affine frames on distinguished regions. In *BMVC*. British Machine Vision Association, 2002.

[33] University of Toronto. Sift demo. `http://www.cs.toronto.edu/~jepson/csc2503/`. Last accessed 04/07/2005.

[34] Kohtaro Ohba and Katsushi Ikeuchi. Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):1043–1048, 1997.

[35] Kevin T. Patton and Gary A. Thibodeau. *Anatomy & Physiology*, pages 556–623. Mosby, 2003.

[36] Stig Persson. *Kardiologi: hjãrtsjukdomar hos vuxna*, pages 41–46. Studentlitteratur, 5 edition, 2003.

[37] Massimo Piccardi and Tony Jan. Recent Advances in Computer Vision. *The Industrial Physicist*, pages 18–21, March 2003.

[38] Eric Pierce. Heart. `http://en.wikipedia.org/wiki`. Last accessed 04/07/2005.

[39] Arthur R. Pope. Model-based object recognition - a survey of recent research. Technical report, 1994.

[40] Maximilian Riesenhuber and Tomaso Poggio. Models of object recognition. *nature neuroscience supplement*, 3:1199–1204, November 2000.

[41] Glynn P. Robinson, Alan C. F. Colchester, and Lewis D. Griffin. Model-based recognition of anatomical objects from medical images. *Image and Vision Computing*, 12(8):499–507, 1994.

[42] Arziel Rosenfeld. *Hal's Legacy: "2001's" Computer as Dream and Reality*, chapter 10. The MIT Press, 1998.

[43] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 3d object modeling and recognition using affine-invariant patches and multi-

view spatial constraints. In *CVPR (2)*, pages 272–280. IEEE Computer Society, 2003.

[44] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 636–643, july 2001.

[45] Frederik Schaffalitzky and Andrew Zisserman. Geometric grouping of repeated elements within images. In *Shape, Contour and Grouping in Computer Vision*, pages 165–181, London, UK, 1999. Springer-Verlag.

[46] Bernt Schiele and James L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36(1):31–50, 2000.

[47] Cordelia Schmid and Roger Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intellence*, 19(5):530–535, 1997.

[48] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, 1999.

[49] Michael J. Swain and Dana H. Ballard. Indexing Via Color Histograms. In *Proceedings, Third International Conference on Computer Vision, 1990*, pages 390–393, December 1990.

[50] Hugues Talbot and Leanne Bischof. An overview of the Polartechnics SolarScan melanoma diagnosis algorithms. In *Proceedings of the 2003 aprs workshop on digital image computing*, pages 33–38. The Australian Pattern Recognition Society, 2003.

[51] Tinne Tuytelaars and Luc J. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *BMVC*. British Machine Vision Association, 2000.

[52] Tinne Tuytelaars, Luc J. Van Gool, L. D'haene, and Reinhard Koch. Matching of affinely invariant regions for visual servoing. In *ICRA*, pages 1601–1606, 1999.

[53] Isaac Weiss and Manjit Ray. Model-based recognition of 3d objects from single images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):116–128, 2001.

[54] Andrew Witkin. Scale-space filtering. In *International Joint Conference on Artificial Intelligence*, pages 1019–1022, 1983.

[55] Hiroyuki Yoshida, David D. Casalino, Bilgin Keserci, Abdulhakim Coskun, Omer Ozturk, and Ahmet Savranlar. Wavelet-packet-based texture analysis for differentiation between benign and malignant liver tumours in ultrasound images. *Physics in Medicine and Biology*, 48(22):3735–3753, November 2003.

Appendix A

Image data

Abbrevations used are defined as follows:

2ch Two-chamber view

4ch Four-chamber view

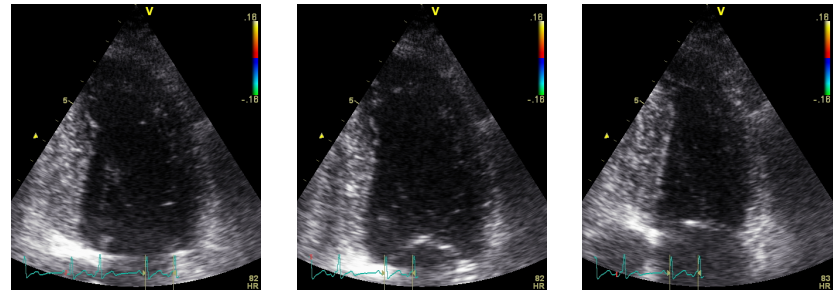
aplax Apical long axis view

plax Parasternal long axis view

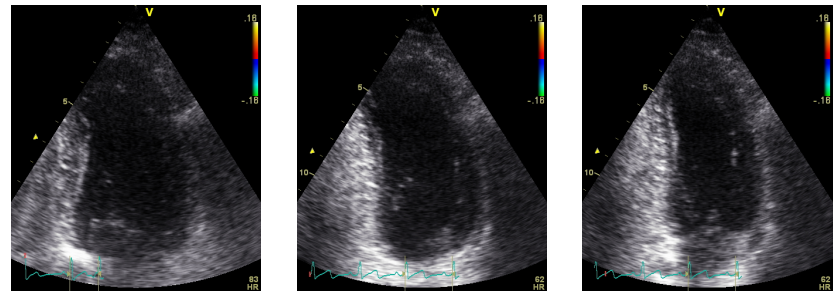
sax Parasternal long axis view

A.1 Training set

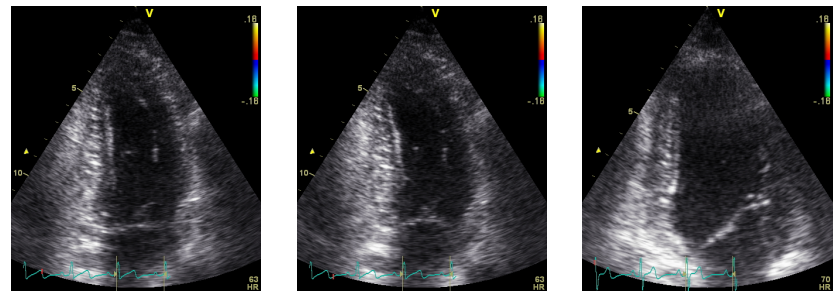
Figures A.1 to A.5 show the images that make up the training set.



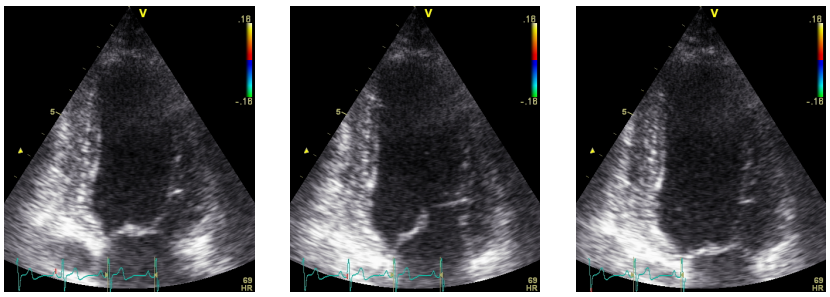
(a) Patient 12 (b) Patient 12 (c) Patient 12



(d) Patient 12 (e) Patient 11 (f) Patient 11

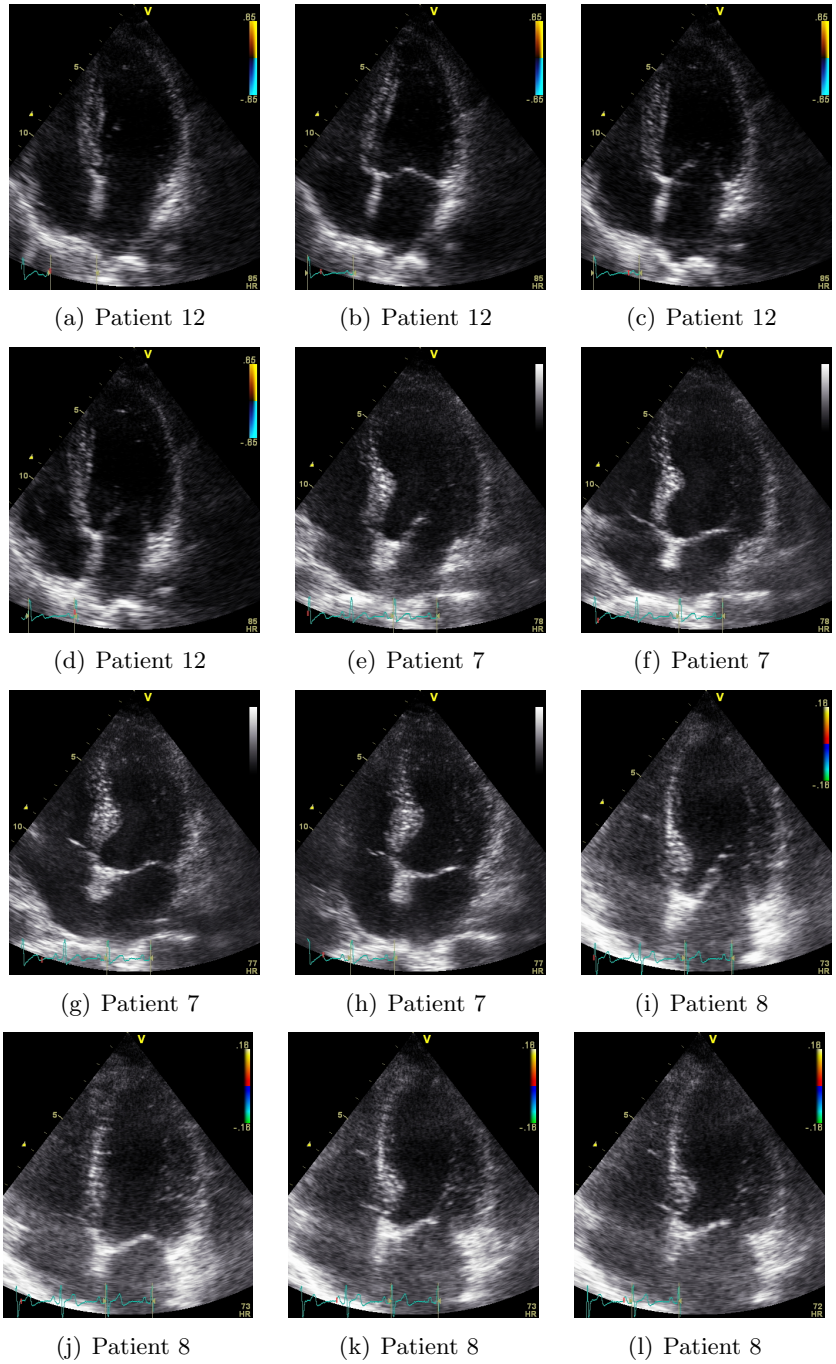


(g) Patient 11 (h) Patient 11 (i) Patient 8



(j) Patient 8 (k) Patient 8 (l) Patient 8

Figure A.1: Training images, two-chamber view



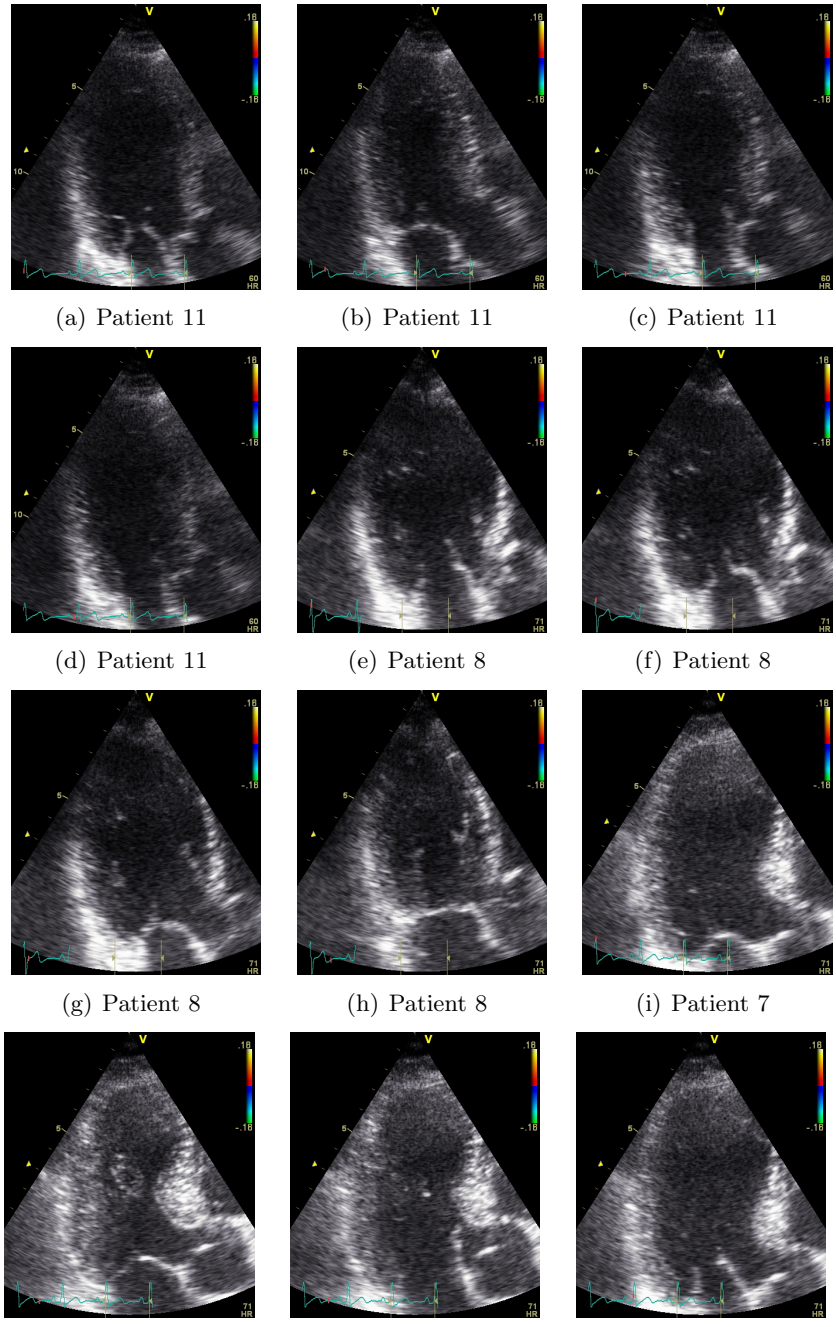
(a) Patient 12 (b) Patient 12 (c) Patient 12

(d) Patient 12 (e) Patient 7 (f) Patient 7

(g) Patient 7 (h) Patient 7 (i) Patient 8

(j) Patient 8 (k) Patient 8 (l) Patient 8

Figure A.2: Training images, four-chamber view



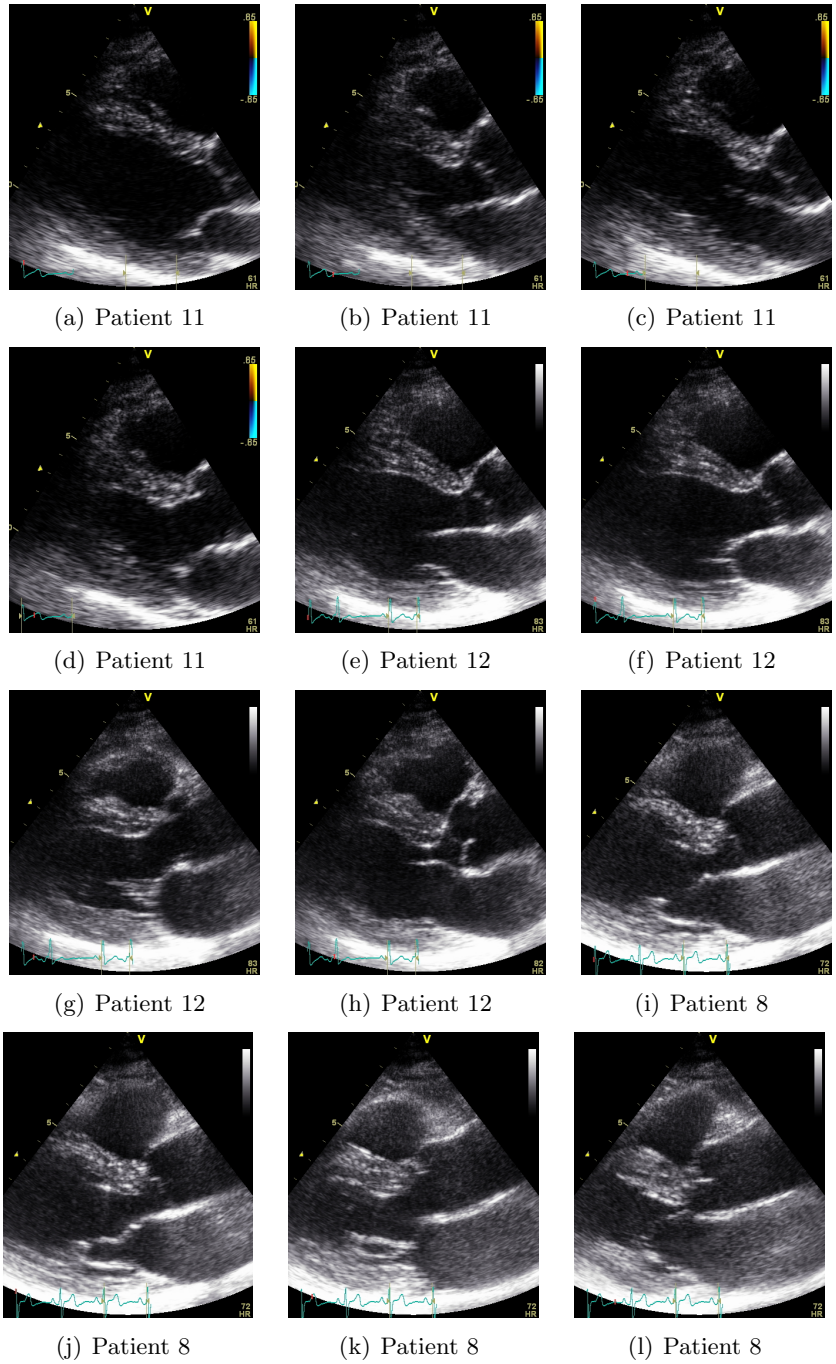
(a) Patient 11 (b) Patient 11 (c) Patient 11

(d) Patient 11 (e) Patient 8 (f) Patient 8

(g) Patient 8 (h) Patient 8 (i) Patient 7

(j) Patient 7 (k) Patient 7 (l) Patient 7

Figure A.3: Training images, apical long axis view



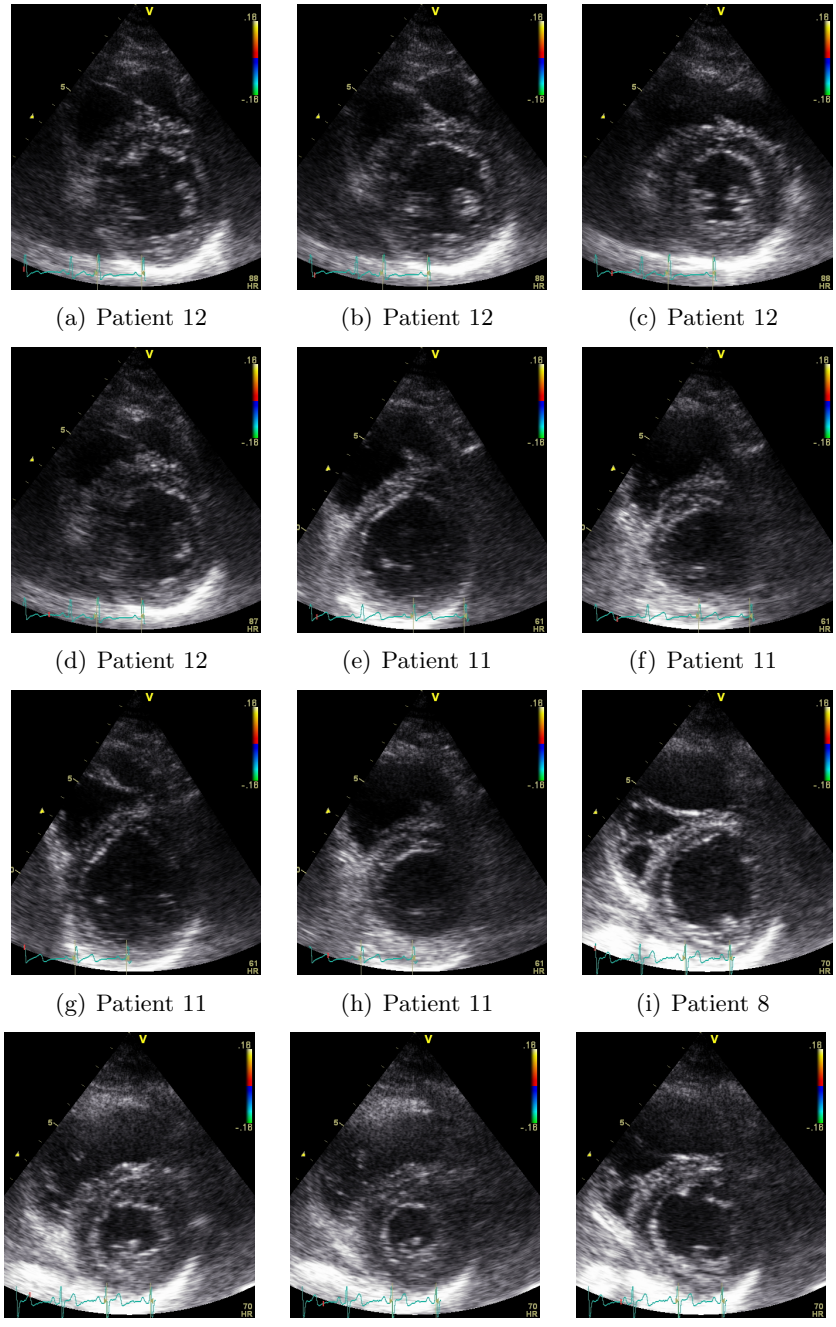
(a) Patient 11 (b) Patient 11 (c) Patient 11

(d) Patient 11 (e) Patient 12 (f) Patient 12

(g) Patient 12 (h) Patient 12 (i) Patient 8

(j) Patient 8 (k) Patient 8 (l) Patient 8

Figure A.4: Training images, parasternal long axis view



(a) Patient 12 (b) Patient 12 (c) Patient 12

(d) Patient 12 (e) Patient 11 (f) Patient 11

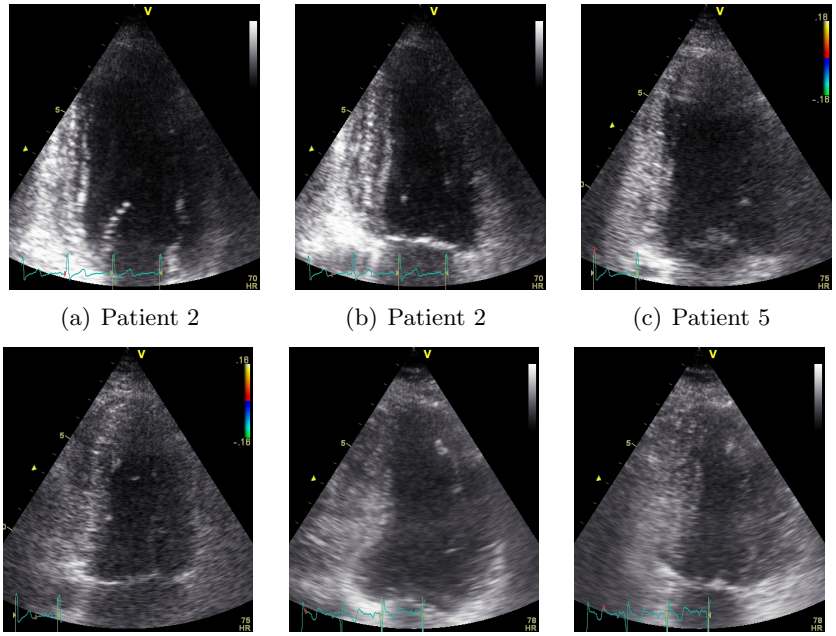
(g) Patient 11 (h) Patient 11 (i) Patient 8

(j) Patient 8 (k) Patient 8 (l) Patient 8

Figure A.5: Training images, parasternal short axis view

A.2 Test set 1

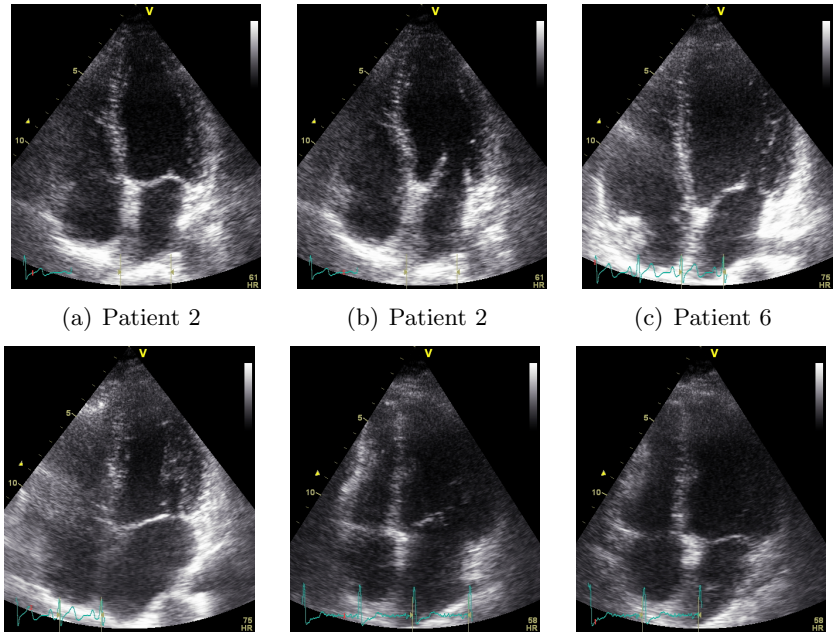
Figures A.6 to A.10 show the images that make up test set 1.



(a) Patient 2 (b) Patient 2 (c) Patient 5

(d) Patient 5 (e) Patient 4 (f) Patient 4

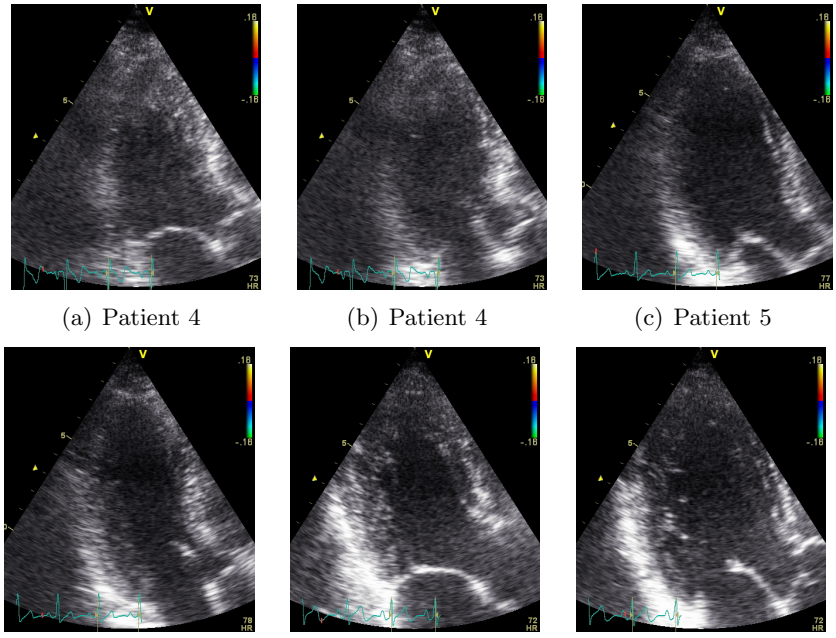
Figure A.6: Test set 1, two-chamber view



(a) Patient 2 (b) Patient 2 (c) Patient 6

(d) Patient 6 (e) Patient 3 (f) Patient 3

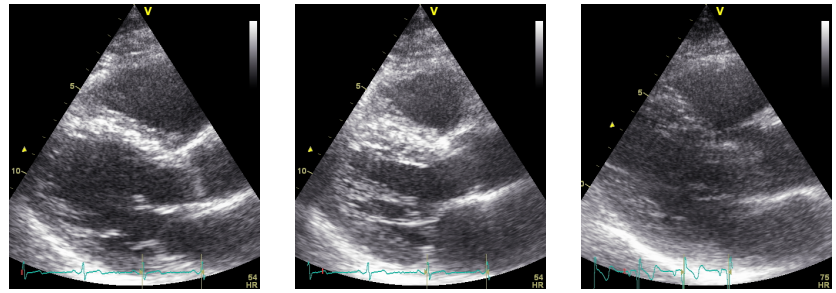
Figure A.7: Test set 1, four-chamber view



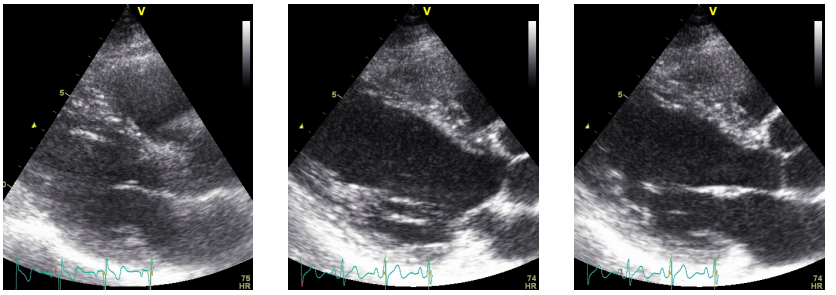
(a) Patient 4 (b) Patient 4 (c) Patient 5

(d) Patient 5 (e) Patient 6 (f) Patient 6

Figure A.8: Test set 1, apical long axis view

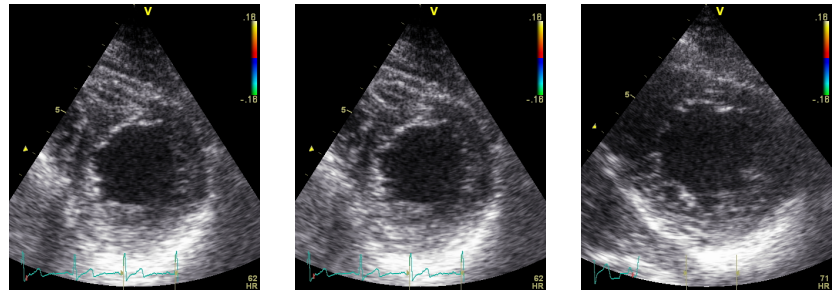


(a) Patient 11 (b) Patient 11 (c) Patient 4

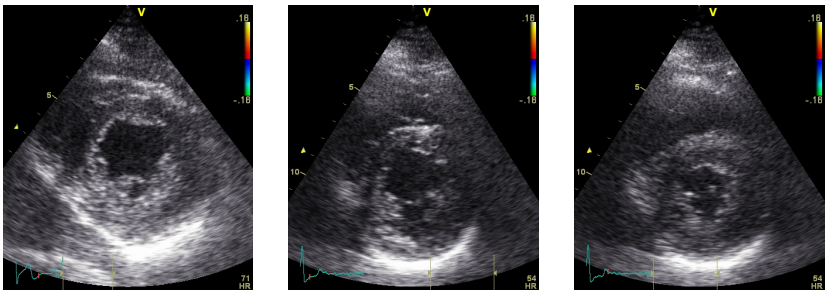


(d) Patient 4 (e) Patient 6 (f) Patient 6

Figure A.9: Test 1, parasternal long axis view



(a) Patient 2 (b) Patient 2 (c) Patient 6

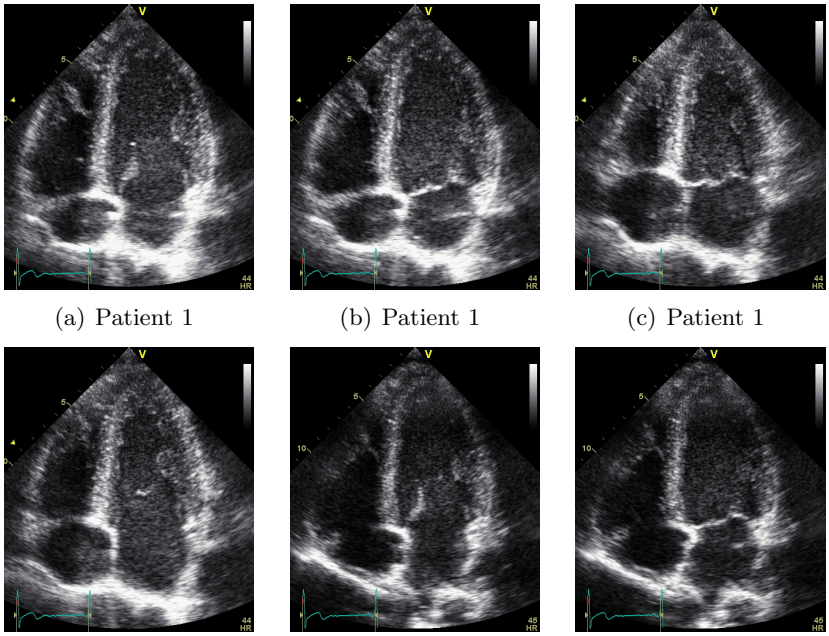


(d) Patient 6 (e) Patient 3 (f) Patient 3

Figure A.10: Test set 1, parasternal short axis view

A.3 Test set 2

Figure A.11 shows the images that make up test set 2.



(a) Patient 1 (b) Patient 1 (c) Patient 1

(d) Patient 1 (e) Patient 1 (f) Patient 1

(g) Patient 1 (h) Patient 1

Figure A.11: Test set 2, four-chamber view