Universitetet i Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study program/specialization: Information Technology – Automation and Signal Processing	Spring semester, 2017 Open / ~~Confidential~~
Author: Elisabeth Noss Karlsen	..... Elisabeth N. Karlsen ..... (signature author)

Instructor: Kjersti Engan

Supervisor: Kjersti Engan

Title of Master's Thesis: Automatic Detection of the Intersection Point Between Left and Right Ventricle in LG-CMR images.
Norwegian title: Automatisk Deteksjon av Skjæringspunkt Mellom Venstre og Høyre Hjerteventrikkel i LG-CMR bilder.

ECTS: 30

Subject headings: Intersection point, image processing, point detection,	Pages: 59 + attachments/other: 4 Stavanger, June 15th / 2017 Date/year

Abstract

Each year Myocardial Infarction affects millions of people worldwide. Approximately 66% of those experiencing a Myocardial Infarction survives, but are at a higher risk of heart related complications and another Infarction. Approximately 30% of post Myocardial Infarction Patients will experience two- or more Infarctions.

A Myocardial Infarction causes damage to the heart muscles, which is later healed by forming a scar at the damaged area. Scarring on the heart impacts the hearts ability to pump blood around the body, how big of an impact depends on the scars size and localization.

When cardiac magnetic resonance images are used to study the scar, general anatomical reference points can be used to define the scars size and/or localization in terms of the angular position. Two meaningful reference points, similar in all patients are the left ventricles center point and the point where the left- and right ventricle intersect.

The reference points are often manually marked by a cardiologist. Since this is a time-consuming work, it is desirable to automate this process. Detection of the left- and right ventricles center point has not yet been automated. The objective of this thesis is therefore to determine whether the intersection point can be automatically found with use of image processing, in terms of image segmentation and point features, in terms of texture and location.

The system presented in this study showed promising results, with room for improvement. 109 images, from 15 different patients, with various contrast resolutions, placement of the scar and eventual enlargement of the heart were tested.

Results showed that 70% of the images where an intersection point was detected were classified as a satisfying result, while the average Euclidean distance between the automatically- and real intersection point was 10,46 pixels. After an interpolation method was used to find the missing intersection points, the results were somewhat worsened to 67% and 11,13 pixels.

Preface

This thesis states the end of my master studies in the field of Automation and Signal Processing at the University of Stavanger. I would like to thank my supervisor Professor Kjersti Engan for her valuable guidance and advise. I would also like to thank her for her inspiring eager to always engage her students.

I would also use the opportunity to thank my co-students for making these two years memorable.

Finally, I would like to thank Even Wick Kvalvaag for his patience and support during these final weeks.

Thank you!

Elisabeth Noss Karlsen
University of Stavanger, June 15th / 2017

Contents

List of abbreviations

ADABoost	Adaptive Boosting
AMI	Acute Myocardial Infarction
CMR	Cardiac Magnetic Resonance
DICOM	Digital Imaging and Communications in Medicine
LG	Late Gadolinium
MI	Myocardial Infarction
NORMI	Norwegian Myocardial Infarction Register
ROS	Random Oversampling
RUS	Random Undersampling
SE	Structuring Element
SMOTE	Synthetic Minority Oversampling Technique
WHO	World Health Organization

1 Introduction

Each year Myocardial Infarction (MI) affects millions of people worldwide [1]. It occurs when the supply of oxygen to the heart is severely- or completely cut off causing damage to the myocardium, also known as the heart muscle. When the myocardium is damaged, it heals by forming a scar tissue. Since scarred tissue does not contract as well as healthy tissue, this will have an impact on the hearts ability to pump blood around the body [2].

A study conducted in Norway [3] in 2015, show that approximately 2/3 of those who experienced MI was still alive after 30 days. With proper treatment, many post-MI patients can live a full life without any further damage, but this group has a higher death rate and an increased risk of another infarction. Approximately 30% will experience two or more infarctions [1] [3]. Because of the severity and extent connected to MIs and their complications there have been conducted several studies on this field [4] [5] [6]. At the University of Stavanger, in cooperation with the Department of Cardiology at Stavanger University Hospital there has i.e. been conducted studies where late gadolinium enhanced cardiac magnetic resonance (LG-CMR) images of post-MI patients are used [7] [8].

To conduct studies where information about the scars localization and/or size is needed, the scar and healthy myocardium has to be marked. Also, to have meaningful reference points similar in all patients, the left ventricles heart center and the intersection point between the left and right ventricle need to be marked. Using the heart center and the intersection point makes comparison with the American Heart Association 17 segments model easy [9]. The segmentation and markings can be done manually by a cardiologist. Since manually mapping is a time-consuming task, it is desirable to automate this process. There has been proposed several successful methods for automatically segmentation of scar and healthy myocardium [10] [11] [12] and methods which automatically finds the center of the left ventricle [13].

As of the authors knowledge there has not yet been presented a method which automatically detects the intersection point between the two ventricles. The objective of this thesis will be to check if it is possible to automatically detect the intersection point with use of image processing, features describing texture and location and classification.

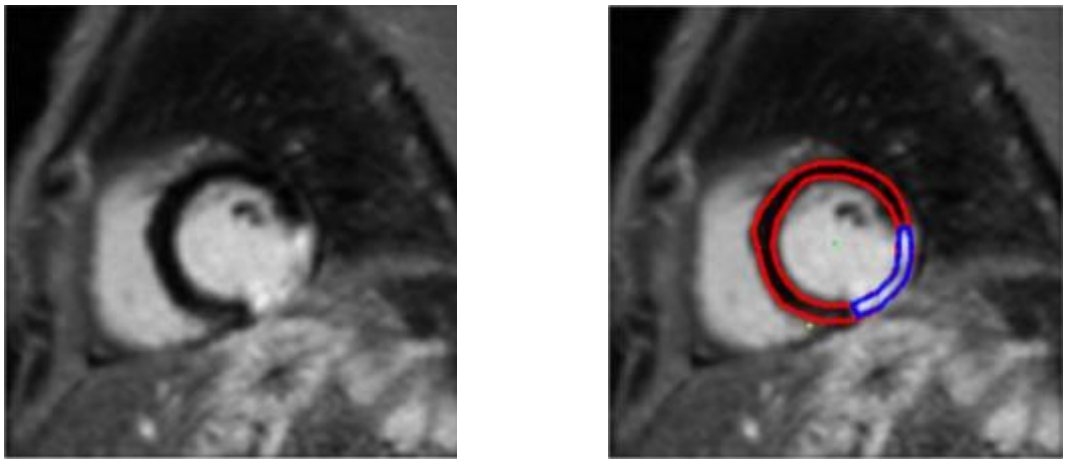


Figure 1.1 Healthy and scarred myocardium, with and without markings. Damaged myocardium marked blue and healthy myocardium marked red.

2 Background

This chapter contains a presentation of the medical background information this thesis bases upon. First, a brief presentation of the heart, myocardial infarction (MI) and epidemiology of MIs in Norway is presented. Thereafter, the data used in this study is presented.

2.1 Heart

A heart consists of a total of four chambers – two superior atria and two inferior ventricles. The atria are separated by the interatrial septum, while the ventricles are separated by the interventricular septum. The atrium and ventricle are separated by internal valves, which opens and closes in response to differences in blood pressure and prevents blood flowing back from the ventricle to the atrium [15].

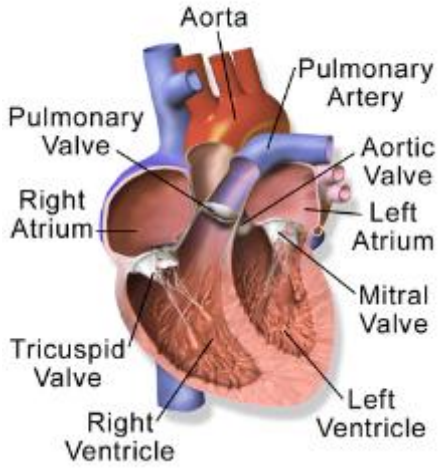


Figure 2.1 Illustration of a human heart with its four chambers and internal valves.
Illustration by Blausen.com [16], Distributed under a CC-BY 3.0 license.
Modifications are done by the author of this thesis.

The heart is the pump in the body's circulation system, pumping oxygen-poor blood to the lungs, and oxygen-rich blood to the rest of the body [15].

2.2 Myocardium Infarction

The heart wall consists of three layers, the outer epicardium, the myocardium, which is the contracting layer, and the inner endocardium. The myocardium is supplied with oxygen-rich blood from vessels known as the coronary arteries. If the supply of oxygen to parts of the

myocardium is interrupted, severely or completely cut off, and heart cells gets damaged or dies it is called myocardial infarction, commonly known as heart attack [15].

An interruption is often caused by blockage of one, or more, of the coronary arteries. Over time the coronary arteries can become narrower from buildups of plaques, such as fat and cholesterol. These buildups can cause an interruption of the flow of oxygenated blood to the heart. A plaque can also break loose in one of the arteries. When a plaque breaks loose a blood clot can form itself around it, causing a total blockage of blood flow to the heart [15] [2].

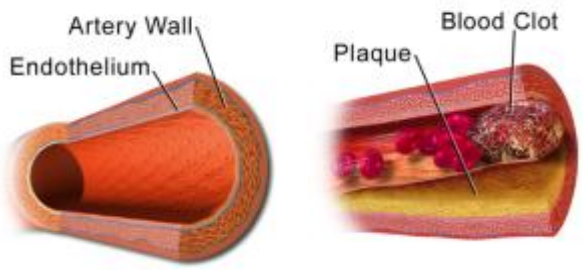


Figure 2.2 Illustration showing a healthy artery (left) and a clogged artery (right).
Illustration by Blausen.com [17]. Distributed under a CC-BY 3.0 license.
Modifications are done by the author of this thesis.

If the myocardium cells get damaged or dies, it will heal by forming a scar tissue in the area where the damage has been done. A scar on the myocardium will impact the hearts ability to pump blood around the body. How big of an impact the scar has, depends on the scars size and location, respectively the time between injury and treatment and which area that were affected by the blockage [2].

2.3 Epidemiology in Norway

Since 2012 there has been published annual reports presenting data from the Norwegian Myocardial Infarction Register (NORMI). NORMI is a web-based medical quality register containing information about medical history, diagnostics, treatments and complications regarding patients hospitalized with MI. It is organized within the framework of the Norwegian Cardiovascular Disease Registry controlled by the Norwegian Institute of Public Health [18].

In 2015 88% of all patients, with a Norwegian identification number, hospitalized and treated for MI was registered in NORMI[1]. This percentage is based on patients registered in NORMI against patients registered in the Norwegian Patient Registry [18].

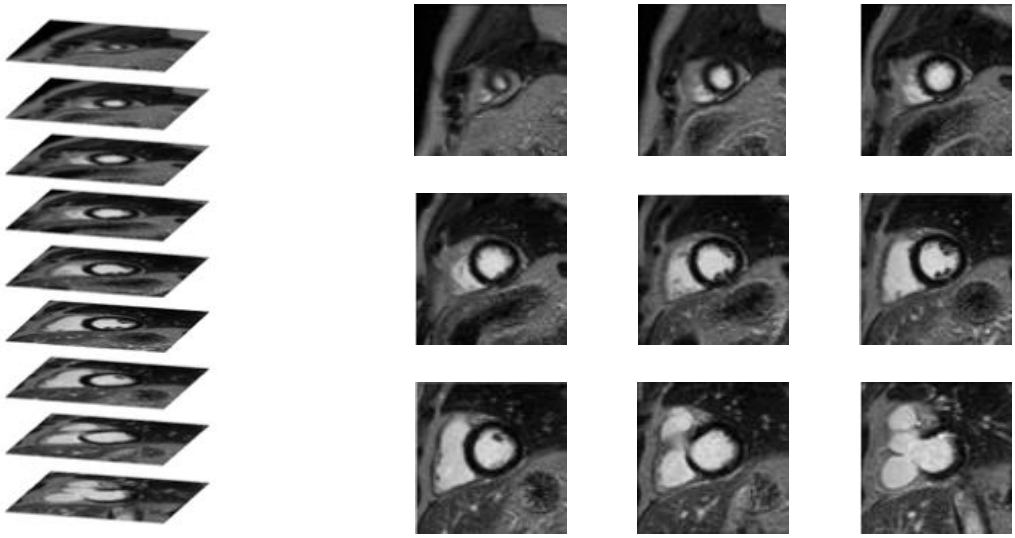
[1] All patients in NORMI are registered with the ICD101-codes I21 or I22 as main- or bi diagnosis [18]. Code I21 and I22 respectively defines diagnosis connected to AMI and subsequent MI [21].

In the 2015 annual report from NORMI, there were 13397 registered acute myocardial infarctions (AMI) for 12621 individual persons, meaning some persons experienced two or more AMIs. Almost 1/3 (32%) of the patients also had a medical history of previous MIs.

In a study from 2015 [3], conducted on participants with an incident coronary heart disease event, the mortality rate due to MI was presented. Out of the 1845 cases considered in the study, the mortality rate within 28 days was approximately 1/3. Out of these deaths, 58% occurred as sudden deaths outside the hospital.

2.4 Data Material

The data material used in this study are collected from a data set of LG-CMR images provided by the Department of Cardiology at Stavanger University Hospital. The data set consist of images from 91 post-MI patients, from groups of both high- and low risk for cardiac arrhythmia. All images were obtained with a pixel size of 0.82 x 0.82 mm^2, covering the whole ventricle with short-axis slices with thickness of 10 mm. The images were stored per the Digital Imaging and Communications in Medicine (DICOM) format with a 512 x 512 pixel resolution. More information about the image capturing process can be found in [18].



(a) Stacked slices *(b) All short-axis slices from one patient*

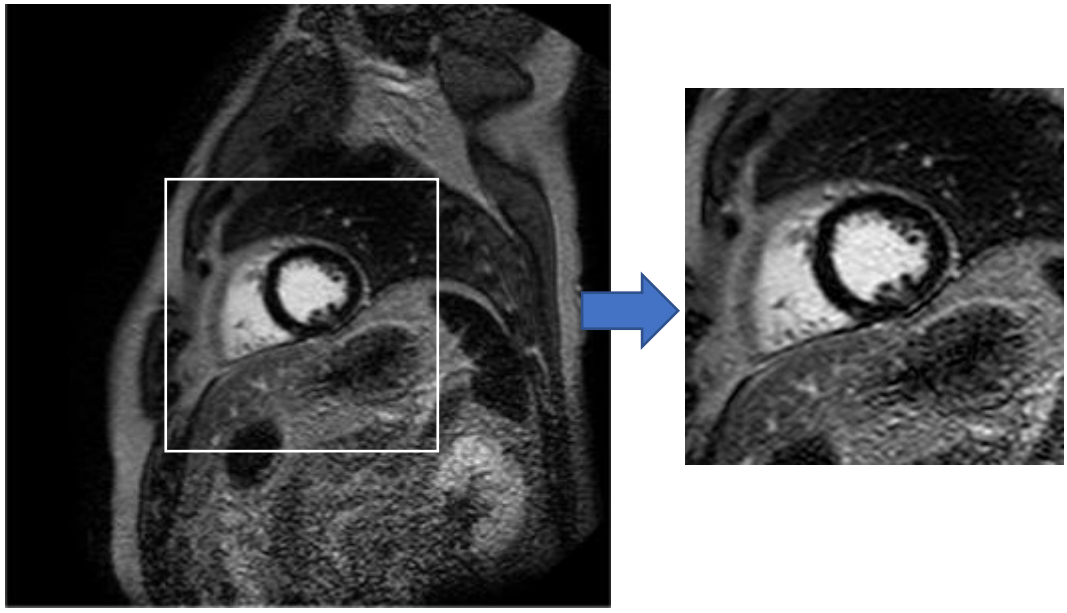
Figure 2.3 Cropped short-axis slices

In this study the focus will be on the images taken minimum one year after the infarction, by then the swelling and scar have settled. Out of 91 patients, 6 patients lacked images taken one year after the infarction and were therefore discarded. To tell how well the proposed system works, it is necessary to have data to compare the automatic detected points with. Images without an intersection point marked by a cardiologist were therefore discarded from the data set as well. Out of the remaining 85 patients, 20 were removed due to lack of markings set by the cardiologist, leaving a data set of images from 60 patients.

2.4.1 Previous work on this data set

The original data set contains LG-CMR images of the whole torso from the patients. There are several studies at the University of Stavanger that have taken this data set into use [7] [10]. The focus area for these studies, as well as the presented study, have been on the heart.

At UiS, there have been developed functions which are used in this study. The first function used, organizes the sub-images from one patient in order from bottom to top, as is illustrated in Figure 2.3 The second function crops the original image, narrowing in the focus area, as illustrated in Figure 2.4. The last function automatically finds the left ventricles center point.



(a) Original image *(b) Cropped image*

Figure 2.4 Illustration of have the original image is cropped

3 Theory

This chapter contains a presentation of the theoretical information used in the study. Bilateral filtering and Otsu's method are first presented, morphological operations and contrast adjustment are then presented. Finally, performance measures and the used classifier are presented.

3.1 Bilateral Filtering

Bilateral filtering is a method proposed by C.Tomasi and R.Manduchi in 1998 [19]. The basic idea of bilateral filtering was to create a filter which averaged away noise within the smooth regions in an image without averaging across the edges. This way creating a filter which smooth the image, while still preserving the edges.

The proposed filter is a filter which combines low-pass domain filtering with range filtering. Combining these filters enforces both geometric- and photometric locality. The filter applied to an image, I(x,y) produces an output image as follows:

$$I_{BF}(x,y) = k^{-1}(x,y)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x_n,y_n)\, c((x_n,y_n),(x,y))\, s\big(I(x_n,y_n),I(x,y)\big)\, d(x_n,y_n) \tag{3.1}$$

where (x,y) is pixel coordinates, (x_n, y_n) a nearby point to (x,y), $c((x_n,y_n),(x,y))$ the geometric closeness between (x,y) and (x_n, y_n), $s\big(I(x_n,y_n),I(x,y)\big)$ the photometric similarity between the pixel at (x,y) and the pixel at (x_n, y_n) and k(x,y) the normalization for the filter given by

$$k(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} c((x_n,y_n),(x,y))\, s\big(I(x_n,y_n),I(x,y)\big)\, d\,(x_n,y_n) \tag{3.2}$$

The low-pass domain filter, $h_d(x,y)$, the range filter, $h_r(x,y)$, and their respectively normalizations $k_d(x,y)$ and $k_r(x,y)$, can be defined as followed

$$h_d(x,y) = k_d^{-1}(x,y)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x_n,y_n)\, c((x_n,y_n),(x,y))\, d(x_n,y_n) \tag{3.3}$$

$$k_d(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} c((x_n,y_n),(x,y))\, d(x_n,y_n) \tag{3.4}$$

$$h_r(x,y) = k_r^{-1}(x,y)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x_n,y_n)s\big(I(x_n,y_n),I(x,y)\big)\,d(x_n,y_n) \tag{3.5}$$

$$k_r(\boldsymbol{x},\boldsymbol{y}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} s\big(I(x_n,y_n),I(x,y)\big)\,d(x_n,y_n) \tag{3.6}$$

The filter takes the pixel value at x and replaces it with an average of nearby and similar pixel values. If the filter is centered in a smooth region, where the pixel values in the neighborhood are similar to each other, the normalized similarity function will be close to one and the filter will essentially act as a pure domain filter. In smooth regions, the filter averages away the small correlated differences between the pixel values which is caused by noise.

If the filter is centered on either side of a sharp boundary, the similarity function will assume that values close to one, for center on bright side, as pixel on the same side and values close to zero as pixels on the different side and vice versa for center on dark side. The filter then ignores the pixels on the wrong side, and replaces the pixel at the center by an average of close pixels on the same side.

The closeness function, $c((x_n,y_n),(x,y))$, and the similarity function, $s\big(I(x_n,y_n),I(x,y)\big)$, are Gaussian functions of the Euclidean distance between their arguments and can be defined

$$c((x_n,y_n),(x,y)) = e^{-\frac{1}{2}\left(\frac{d((x_n,y_n)(x,y))}{\sigma_d}\right)^2} \tag{3.7}$$

$$d((x_n,y_n)(x,y)) = d((x_n,y_n)-(x,y)) = \|(x_n,y_n)-(x,y)\| \tag{3.8}$$

$$s\big(I(x_n,y_n),I(x,y)\big) = e^{-\frac{1}{2}\left(\frac{\delta(I(x_n,y_n),I(x,y))}{\sigma_r}\right)^2} \tag{3.9}$$

$$\delta\big(I(x_n,y_n),I(x,y)\big) = \delta\big(I(x_n,y_n)-I(x,y)\big) = \|I(x_n,y_n)-I(x,y)\| \tag{3.10}$$

The amount of low-pass filtering is chosen based on the geometric spread, σ_d. Larger σ_d combines pixel values from more remote image locations and blurs the image more.
The photometric spread, σ_r, loosely speaking, chooses the amount of pixels which is mixed together. If the difference in pixel values are smaller than σ_r the pixels are mixed together, if not, the pixels are not mixed together. The higher σ_r, the more pixels are mixed together.

Figure 3.1 Bilateral filtering on grayscale image, $\sigma_R = 0{,}1$, $\sigma_D = 1$
Image source: MATLAB test images

3.2 Otsu's Method

A simple approach to image segmentation is to choose suitable thresholds which can be used to divide the image into distinct regions. I.e. separating objects from the background. A renowned method for threshold selection is Otsu's method [20]. Otsu's method is a simple procedure which automatically selects one or several optimal thresholds.

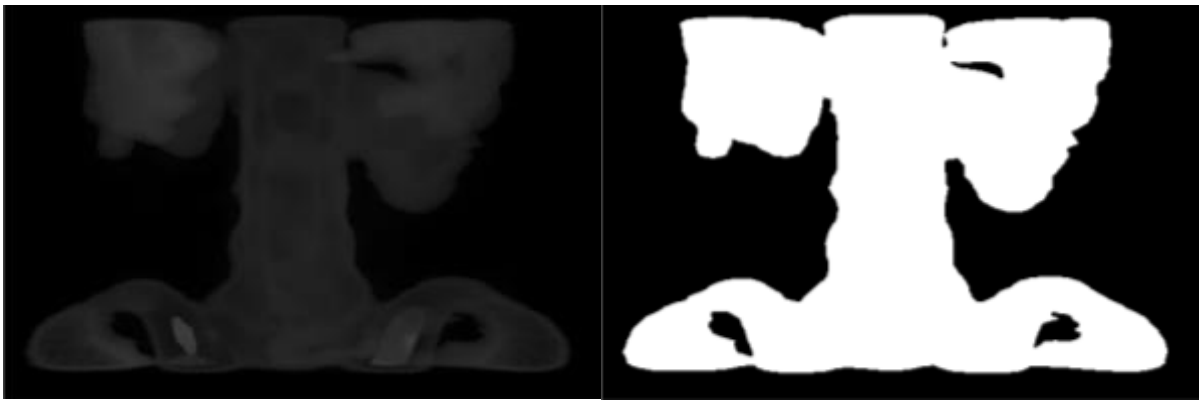


Figure 3.2 Otsu's method used to segment a grayscale image.
Image source: MATLAB test images

Given an image where the pixels are represented by L gray levels with a normalized gray-level histogram p. The pixels are divided into two classes by a threshold at level k, where the probability of the two classes are given by

$$w_0 = \sum_{i=1}^{k} p_i \,, \qquad w_1 = \sum_{i=k+1}^{L} p_i \tag{3.11}$$

where p_i denotes bin i in the normalized histogram p. The two class mean levels and the total mean level are respectively given by

$$\mu_0 = \sum_{i=1}^{k} i \frac{p_i}{w_0}, \qquad \mu_1 = \sum_{i=k+1}^{L} i \frac{p_i}{w_1}, \qquad \mu_T = \sum_{i=1}^{L} i p_i \tag{3.12}$$

and the variance for the two class levels are given by

$$\sigma_0^2 = \sum_{i=1}^{k} (i - \mu_0)^2 \frac{p_i}{w_0}, \qquad \sigma_1^2 = \sum_{i=k+1}^{L} (i - \mu_1)^2 \frac{p_i}{w_1} \tag{3.13}$$

When the class means, class variances and total mean and variance are given, the method calculates the within-class variance level

$$\begin{aligned} \sigma_B^2 &= w_o(\mu_0 - \mu_T)^2 + w_1(\mu_1 - \mu_T)^2 \\ &= w_0 w_1 (\mu_1 - \mu_0)^2 \end{aligned} \tag{3.14}$$

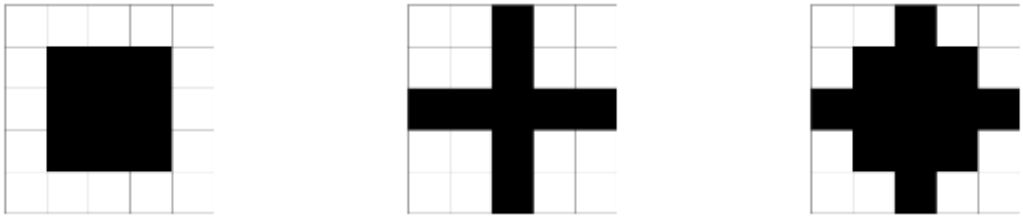
To find the optimal threshold value k*, the method calculates σ_B^2 for each threshold k finding which threshold giving the maximum σ_B^2. The threshold k giving the maximum σ_B^2 is chosen as the optimal threshold value.

3.3 Morphological operations

In image processing, working with binary or grayscale images, mathematical morphology has proven to provide successful- and useful tools. Morphological operations provide algorithms that can be used to identify and extract image components of interest. The basic principle is to extract topological and geometrical information from an image, using transformations inputting the image and a well-defined structuring element (SE) [21].

Structuring Element

The basic idea is to probe an image with a SE checking how well the SE fits within an image. A SE is often represented by a matrix, containing binary valued pixels, and can have any shape and size, depending on its use. The SE has a great impact when applying the morphological operator to the image, so defining the size and shape is a crucial step. Defining the wrong shape or size may result in loss of structural information regarding the image. [21] [22] [23].



a) Square *b) Cross* *c) Diamond*

Figure 3.3 Example shapes for structuring elements

Erosion and dilation

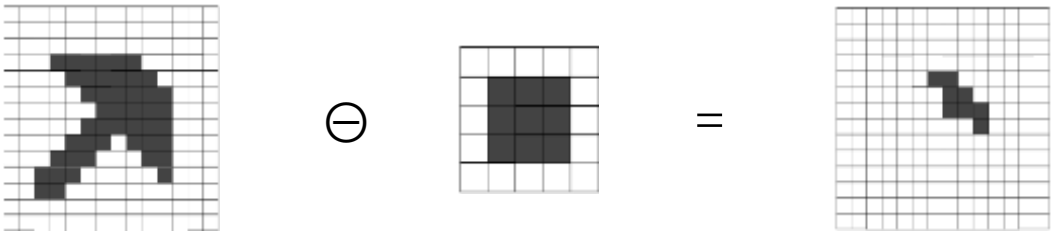
Every morphological operation can be defined in terms of two morphological operators, erosion and dilation. The erosion- and dilation operation on image A by a SE B can respectively be defined as

$$I_{eroded} = I \ominus B \tag{3.15}$$

$$I_{dilated} = I \oplus B \tag{3.16}$$

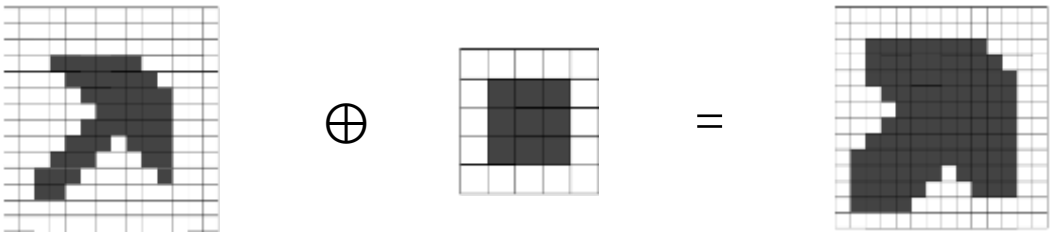
Erosion is an operation designed to thin-, while dilation is an operation designed to thicken the objects in the image. The SEs size and shape controls the extent and direction of the thinning and thickening. For both erosion and dilation, the SE slides over the image placing its center pixel over each for- or background pixel whichever is desired. The size of the SE defines which neighborhood pixels to be looked at.

For erosion, only pixels where the SEs origin can be placed and the shape of the SE is fully contained within the corresponding neighborhood pixels is kept. These kept pixels make out the eroded image. For dilation, the SEs origin is also placed over all the chosen object pixels, but here all neighborhood pixels that are covered by the shape of the SE is set as an object pixel making out the dilated image [24].



(a) Original image *(b) SE* *(c) Eroded image*

Figure 3.4 Morphological erosion



(a) Original image *(b) SE* *(c) Dilated image*

Figure 3.5 Morphological dilation

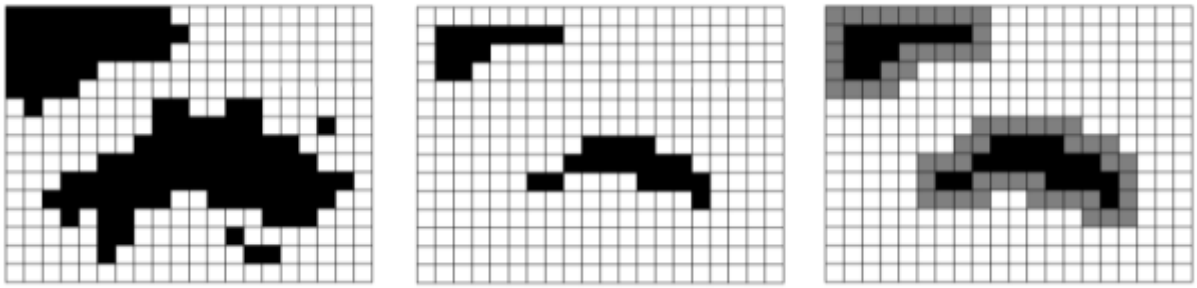
Opening and closing

Two other morphological operations that plays a key role in the processing of an image are opening and closing. Opening and closing are dual to each other and are defined with use of erosion and dilation. The equations for opening and closing can respectively be expressed as followed

$$I_{open} = I \circ B = (I \ominus B) \oplus B \tag{3.17}$$

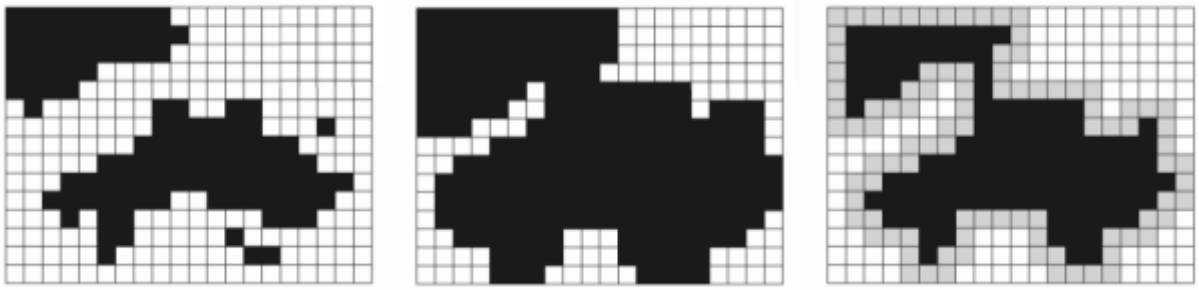
$$I_{close} = I \bullet B = (I \oplus B) \ominus B \tag{3.18}$$

where I is the image, and B the SE.



(a) Original image *(b) Erosion of original image* *(c) Dilation of eroded image*

Figure 3.6 Morphological opening.
Illustration by Robotyczna Owca, Distributed under CC0



(a) Original image *(b) Dilation of original image* *(c) Erosion of dilated image*

Figure 3.7 Morphological closing.
Illustration by Robotyczna Owca, Distributed under CC0

The use of opening and closing, results in images replicating the original images in a finer manner than using only erosion and dilation. Opening smooths the boundary and removes objects which are smaller than the SE from the foreground, setting them as background. Closing sets small regions of background into foreground, closing small holes which may occur in the object [23] [24].

Top-Hat Transformation

There are two types of top-hat transforms, open top-hat transform and, its complementary, closed top-hat transform. The focus for this chapter will be the open top-hat transform, since this is the operator used in the proposed system.

Open top-hat transform is defined as the difference between the original image, I, and the morphologically opened image, I_{open}

$$I_{TopHat} = I - I_{open} = I - (I \circ B)$$

The transformation can efficiently be used to find small clusters of dark pixels surrounded by a bright background or white pixels surrounded by a dark background. It can also be a useful tool to correct uneven illumination in an image, and in images with little noise it can be used to find edges [23].

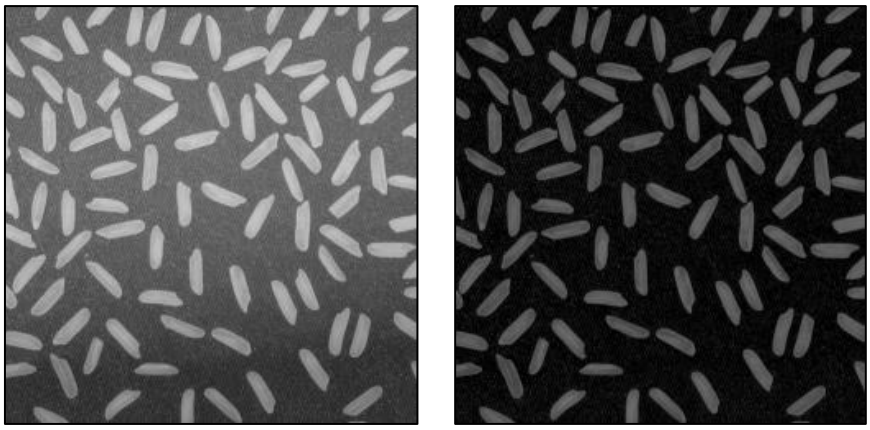


Figure 3.8 Image illustration Top-Hat transformation with use of a disk-shaped SE.
Image Source: MATLAB test images

3.4 Classifier

Using data-sets where the classes are imbalanced tend to be challenging when the data is used to train a classifier. Traditional classification algorithms tend to create models where the overrepresented class is favored. Since the class of interest often is the underrepresented class, such a model would be ineffective for this class [25].
Several methods have been proposed to ease the problem of class imbalance. In this section two of these techniques are presented, data sampling and boosting [26].

Data Sampling

Data sampling is one method which can be used to overcome the problem of unbalanced classes. Data sampling oversamples the minor class by adding examples or undersamples the major class by removing examples.

Random Undersampling
A simple method for undersampling is random undersampling (RUS). RUS reduces the examples available in the major class by randomly removing examples in the training-set [27] The benefits of using RUS is that the required time needed to train the classifier is reduced. However, removing examples also results in loss of information.

Random Oversampling
A simple method for oversampling is random oversampling (ROS). ROS increases the class ratio of the minor class by duplicating the class' examples. The benefits of using ROS is that no information is lost due to the duplication of already existing examples. However, this can also lead to overfitting [28]

Synthetic Minority Oversampling Technique
Synthetic Minority Oversampling Technique (SMOTE) is another oversampling method which adds minority examples by extrapolation instead of duplication of pre-existing examples [29].

Boosting

Boosting is technique which also can be used to overcome the problem of unbalanced classes. Boosting is a technique which tries to boost the performance of any weak classifier. A well-known boosting algorithm is adaptive boosting.

Adaptive Boosting
Adaptive boosting (ADABoost) iteratively builds an ensemble of models. For each iterations ADABoost updates weights related to the examples. The weights are updated with the intention of classifying examples which were classified incorrectly in the current iteration correctly. When all iterations are completed, a weighted model are constructed and used to classify new unlabeled examples [30]

3.4.1 RUSBoost classifier

RUSBoost was presented by Seiffert et al. in 2008 [25] It is a hybrid approach which uses ADABoost and RUS to alleviate class imbalance. The algorithm is based on SMOTEBoost's

algorithm [31] which uses ADABoost in combination with SMOTE to improve the classification performance by balancing the class distribution. In difference from SMOTEBoost, RUSBoost improves the classification performance with use of RUS to balance the class distribution, making the algorithm both simpler and faster due to its reduce amount of data. The main drawback of using RUS is loss of information. By combining RUS with boosting, this drawback is greatly overcome as examples absent in one iteration of boosting is likely to be included in other iterations [25]. An overview of the RUSBoost algorithm can be seen in Algorithm 1.

Algorithm 1 RUSBoost classifier [25]

Given:

Set S of examples $(x_1, y_1), \ldots, (x_m, y_m)$ with minority class $y^r \in Y$, $|Y| = 2$

Weak Learner, *WeakLearn*

Number of iterations, T

Desired percentage of total instances to be represented by the minority class, N

1 Initialize $D_1(i) = \frac{1}{m}$ for all i

2 Do for $t = 1,2, \ldots, T$

a Create temporary training dataset S_t' with distribution D_t' using random undersampling

b Call *WeakLearn*, providing it with examples S_t' and their weights D_t'.

c Get back a hypothesis $h_t : X \times Y \rightarrow [0, 1]$.

d Calculate the pseudo-loss (for S and D_t) :

$$\epsilon_t = \sum_{(i,y):y_i \neq y} D_t(i)(1 - h_t(x_i, y_i) + h_t(x_i, y))$$

e Calculate the weight update parameter:

$$\alpha_t = \frac{\epsilon_t}{1-\epsilon_t}$$

f Update D_t :

$$D_{t+1}(i) = D_t(i)\alpha_t^{\frac{1}{2}(1+h_t(x_i, y_i)-h_t(x_i, y: y \neq y_i))}$$

g Normalize D_{t+1} : Let $Z_t = \sum_i D_{t+1}(i)$

$$D_{t+1}(i) = \frac{D_{t+1}(i)}{Z_t}$$

3 Output the final hypothesis:

$$H(x) = \underset{y \in Y}{\operatorname{argmax}} \sum_{t=1}^{T} h_t(x, y) log \frac{1}{\alpha_t}.$$

4 Methods

In this chapter, each implemented method is presented in detail. First, the pre-processing and quantization methods are presented. Thereafter, the point detection methods, followed by the feature extraction method are presented. Last, the classification and detection of intersection point methods are presented.

An overview of the devised system is shown in Figure 4.1.

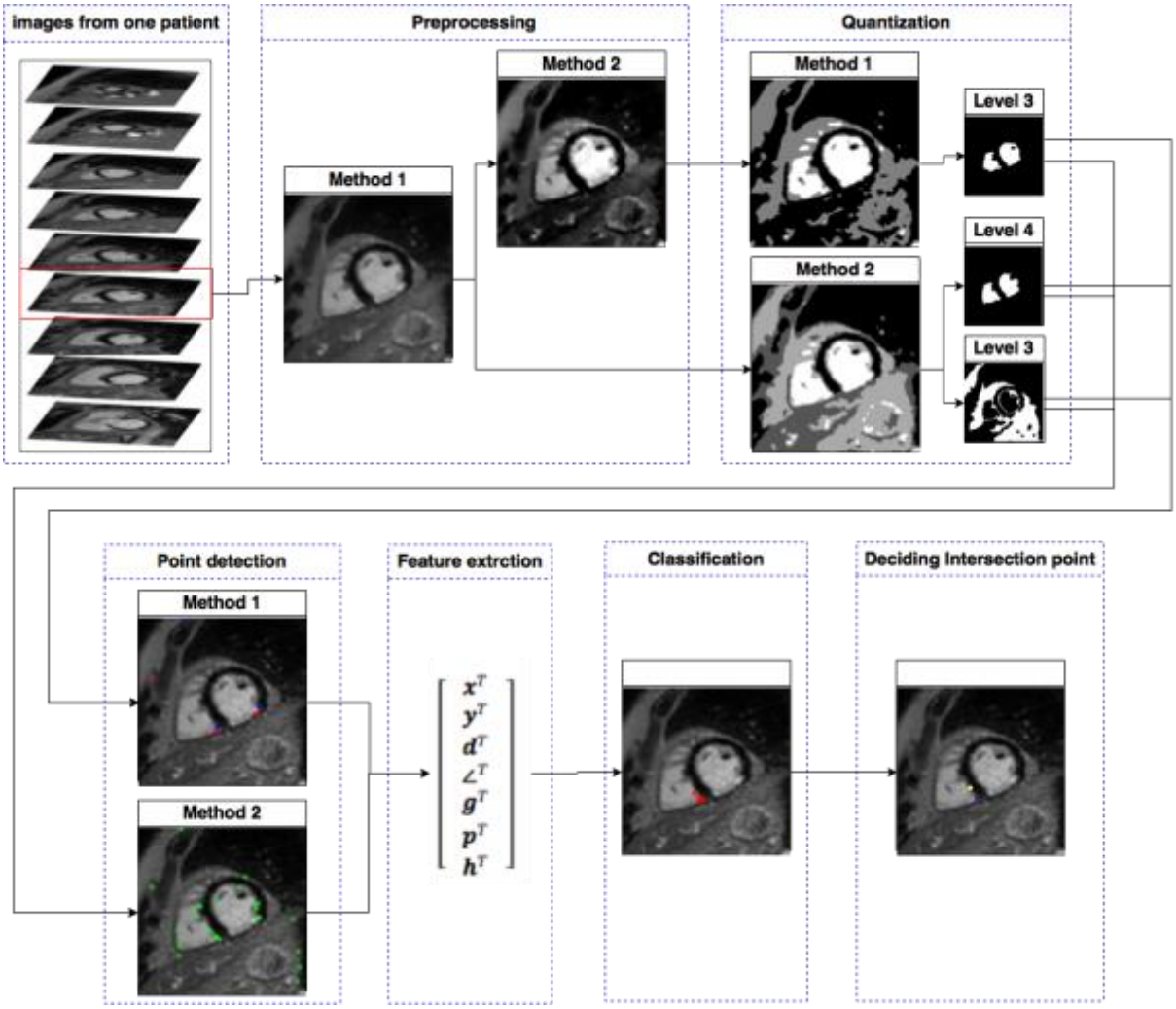


Figure 4.1 Overview of devised system

4.1 Normalization

Data-material used in the proposed system consist of grayscale images from several patients. Due to variations in grayscale-range for each patient, a range normalization is done. A normalization gives consistency in the images. First, the images are normalized to a [0 255] range before they are cropped using the following equation

$$I_{N(i)} = \frac{I_{O(i)}}{\max(I_O)} * 255 \quad (4.1)$$

$I_{N(i)}$ and $I_{O(i)}$ represents the normalized and original sub-image *i*, while I_O represent all original sub-images from one patient. Each pixel in the sub-image are divided by the maximum pixel value for all sub-images before it is multiplied with 255.

After the images are cropped each sub-image goes through one more normalization that changes the grayscale range from [0 255] to [0 1] by dividing each pixel by 255. This normalization makes it possible to work with the images as double floating-point matrices.

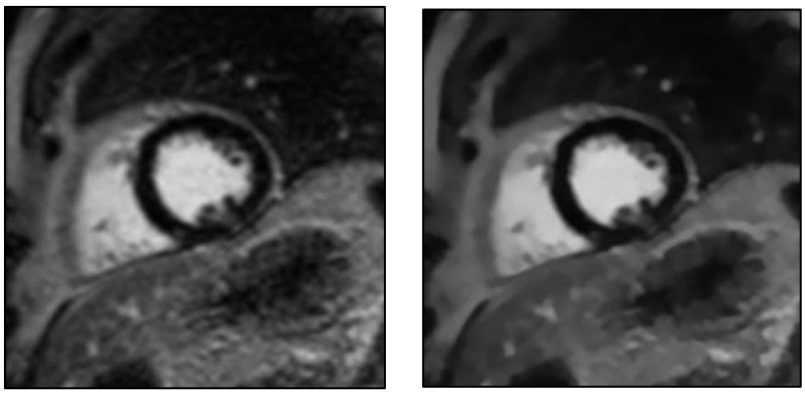
4.2 Preprocessing

This section describes the pre-processing block in Figure 4.1 Two pre-processing methods are presented.

4.2.1 P.M 1: Bilateral filtering

The images provided by the data-set contains noise. To smooth the image and remove noise, a Bilateral filter is applied. This filter smooths the image, with minimal loss of the images' shapes and contrasts.

The filter is implemented based on the theory proposed by C.Tomasi and R.Manduchi [19] in 1998, section 3.1.



(a) Original image *(b) Bilateral filtered image*

Figure 4.2 Images illustration the effect of bilateral filtering.

4.2.2 P.M 2: Top Hat filtering and contrast adjustment

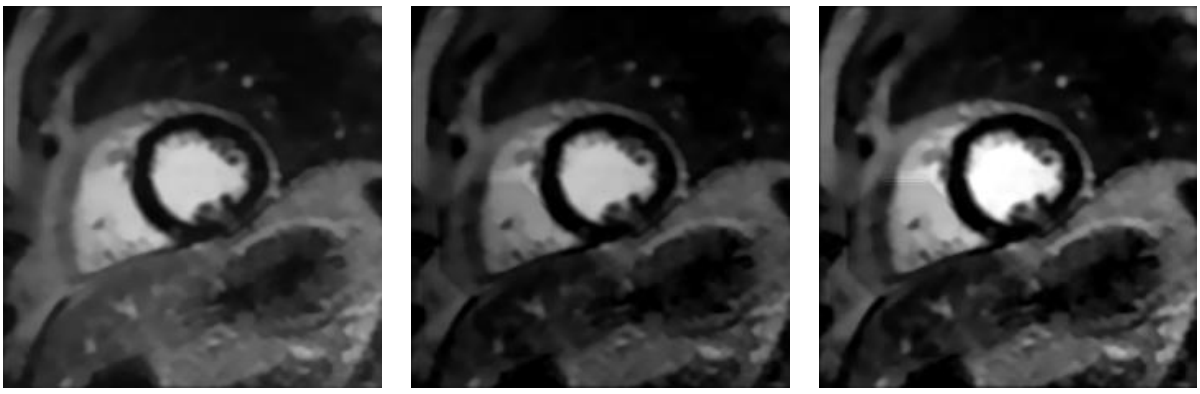
In some of the sub-images, the contrast between the ventricles (further referred to as objects) and the background are not that distinct. Low contrast differences might cause the background to be interpreted as objects or the objects to be interpreted as background. When the images are further processed in the upcoming methods, this might cause the objects to be merged into the background, or vice versa, resulting in poor placement- or no detection of the intersection point.

To overcome this problem, a second preprocessing is done on the image. First, the bilateral filtered image is Top-Hat filtered with a suitable disk-shaped[2] SE to remove uneven background illumination, as described in section 3.3. Thereafter, the Top-Hat filtered image is contrast adjusted to highlight the objects.

Contrast adjustment is done by setting the bottom and top 1% of all the images' pixel values to 0 and 1, before the intensities are stretched to fill the [0, 1] range. Stretching of the images' intensities is done using equation (4.2).

$$\boldsymbol{I}_{CA} = (\boldsymbol{I}_{\boldsymbol{TH}} - \min(\boldsymbol{I}_{\boldsymbol{TH}})) = \frac{\max(\boldsymbol{I}_{CA}) - \min(\boldsymbol{I}_{CA})}{\max(\boldsymbol{I}_{\boldsymbol{TH}}) - \min(\boldsymbol{I}_{\boldsymbol{TH}})} + \min(\boldsymbol{I}_{CA}) \qquad (4.2)$$

$\mathbf{I}_{CA}$ represents the contrast adjusted image and $\mathbf{I}_{TH}$ represents the Top-Hat filtered image.



(a) Bilateral filtered image *(b) Top-Hat filtered image* *(c) Contrast adjusted image*

Figure 4.3 Top-Hat filtering and contrast adjustment of the image

[2] A disk-shaped SE is chosen because the objects of interest are mainly of circular and/or elliptical shapes.

4.3 Quantization

This section presents the different steps used in the *quantizing* block in Figure 4.1.

To extract information using structure and contrast, without loss of valuable information, the filtered image is quantized two times. Method Q.M1 is used for the image pre-processed by both P.M1 and P.M2. Method Q.M2 is used for the image only pre-processed by P.M1.

4.3.1 Q.M1: Quantization after pre-processing by P.M1 and P.M2

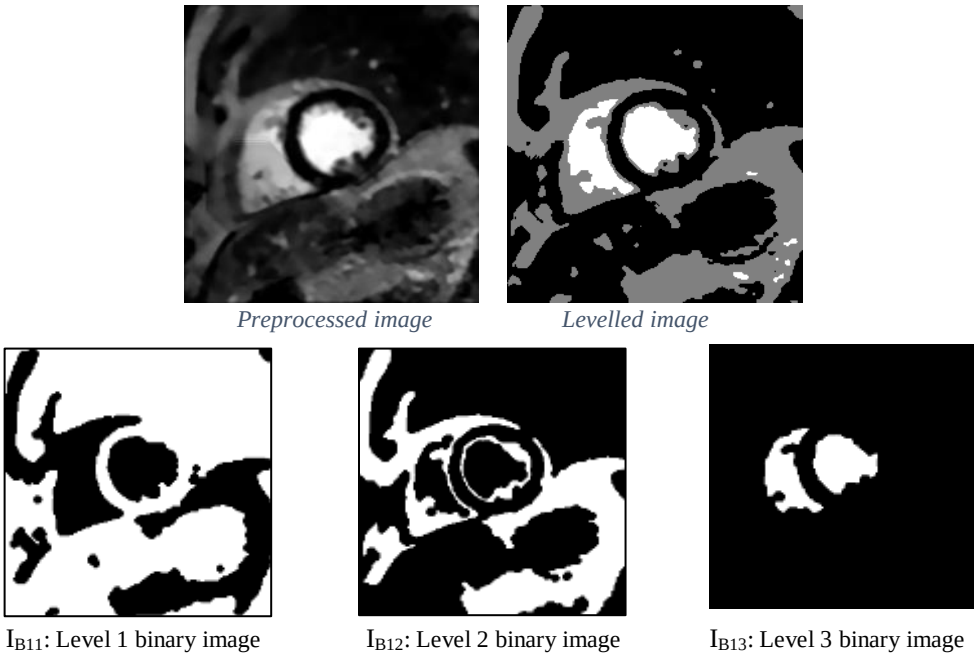
After the image has been preprocessed, two thresholds are automatically chosen using Otsu's method, as described in section 3.2. The image is thereafter quantized into an image with three discrete levels using the thresholds and the following criteria

$$\begin{array}{lllll} If: & & I_F(x,y) & \leq \boldsymbol{t}(1) & then: \;\; I_L(x,y) = 1 \\ If: & \boldsymbol{t}(n-1) < & I_F(x,y) & \leq \boldsymbol{t}(n) & then: \;\; I_L(x,y) = n \\ If: & & I_F(x,y) & > \boldsymbol{t}(n) & then: \;\; I_L(x,y) = T+1 \end{array} \qquad (4.3)$$

where t is the threshold-vector, T the total number of thresholds, I_F the filtered image and I_L the levelled image.

The pixels in the levelled image are now indexed 1, 2 or 3, where pixels indexed 1 makes up the background layer while pixels indexed 3 makes up the object layer. Using the index values, the layers are split into three binary images. Splitting is done by setting all pixels with chosen index value to 1, the rest to 0. The process is repeated for all indexes.

Each of the three binary images goes through a process which morphologically opens the image, setting small objects as background, then closes it, filling in small dark holes. The quantization- and splitting process for method 1 can be seen in Figure 4.4.



Preprocessed image *Levelled image*

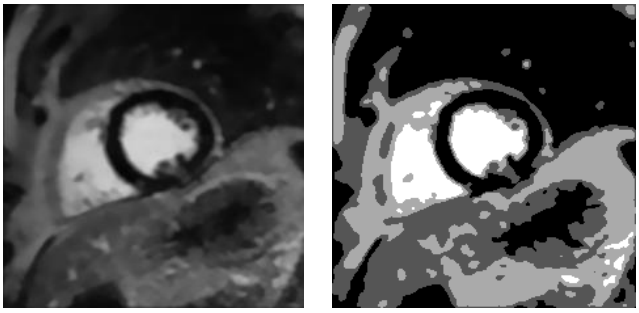
I_{B11}: Level 1 binary image I_{B12}: Level 2 binary image I_{B13}: Level 3 binary image

Figure 4.4 Levelled image and its associating binary images.

4.3.2 Q.M2: Quantization after pre-processing by only P.M1

Unlike Q.M1, Q.M2 uses the bilateral filtered image without any further preprocessing. It also chooses three thresholds, instead of two, giving an image with four discrete levels. The quantization is done according to the criteria presented in equation (4.3).

The pixels in the levelled image are now indexed 1, 2, 3 and 4, where pixels indexed 1 makes up the background layer and pixels indexed 4 makes up the object layer. The different levels can be seen by the different gray-tones in Figure 4.4 and Figure 4.5. Like the first method, the index values are used to split the image into binary images by setting all pixels with a chosen index value to 1 and the rest to 0. Repeating this process for all index values results in four binary images which is then morphologically opened and closed. The quantization - and splitting process for method 2 can be seen in Figure 4.5.



Preprocessed image *4-levelled image*

I_{B21}*: Level 1 binary image* I_{B22}*: Level 2 binary image* I_{B23}*: Level 3 binary image* I_{B24}*: Level 4 binary image*

Figure 4.5 Levelled image and its associating binary images.

4.4 Point detection

This section presents the *Point Detection* block in Figure 4.1

In this section $\mathbf{I_{B\_j}}$ represents the binary images produces by Q.M1 and Q.M2, where *j* indicates the different binary images and (x,y) represents the pixel position. $\mathbf{I_{B\_j}}$ contains one- or several objects. An object is defined as a neighborhood of pixels with value 1 (white pixels) surrounded by pixels with value 0 (black pixels).

Figure 4.6 Binary image, I_{B_jk}, with two objects.

In total, the quantization processes produce seven binary images. Binary images containing little- to none objective information are of no further interest. The two first images from Q.M1 and Q.M2 (**I_{B11}, I_{B12}, I_{B21}, I_{B22}**) are therefore removed since these contains mostly background information. Each of the remaining binary images (**I_{B13}, I_{B23}, I_{B24}**) goes through two separate point-detector methods, PD.M1 and PD.M2.

PD.M1 uses simple addition and finding operations, and PD.M2 looks for equalities between the image and an adaptive binary mask.

4.4.1 PD.M1: Point detection using simple addition and find operations

PD.M1 uses simple operations, as addition and find, to detect points. It looks separately at each object in **I_{B_j}**. For each object the method performs two operations, OP1 and OP2, resulting in two potential intersection points.

For every object in **I_{B_j}**, OP1 searches through the objects pixels, finding the pixel where the added x- and y-coordinate value is the highest. The process is described in equation (4.4).

$$(\boldsymbol{x}_{max}, \boldsymbol{y}_{max}) = \ argmax\{x + y, object(x, y)\} \tag{4.4}$$

If several pixels are found, the potential intersection point, (x_{pIS}, y_{pIS}), is chosen based on the pixel with the highest y-value. If only one pixel is found (x_{pIS}, y_{pIS}) = (x_{max}, y_{max}).

An overview of OP1 can be seen in Algorithm **4.1**.

Algorithm 4.1: Point detection using PD.M1, OP1

for I_{B_j} **do**
  for *all objects* **do**
    $(x_{max}, y_{max}) = argmax\{x + y, object(x, y)\}$
    if $(\boldsymbol{x}_{max}, \boldsymbol{y}_{max})$ contains more than one point **then**
      $(x_{pIS}, y_{pIS}) = max\{y_{max}, (\boldsymbol{x}_{max}, \boldsymbol{y}_{max})\}$
    else
      $(x_{pIS}, y_{pIS}) = (x_{max}, y_{max})$
    end
  end
end

The second operation ignores all object pixels that lays above the left ventricles center point, creating an array containing only object pixels laying below the center point. The operation then searches the new array for the highest x-coordinate. The pixel with the highest x-coordinate is then added and stored as a potential intersection point. If there are several coordinates with the same x-value, the operation chooses the coordinate with the lowest y-value. An overview of the method can be seen in Algorithm **4.2**.

Algorithm 4.2: Point detection using highest x-coordinate

for I_{B_j} **do**
  for *all objects* **do**
    for *all pixels in object* **do**
      if $y_{pixel} < y_{center}$
        Add point to array
      end
    end
    Find pixel with largest x-value in array.
    if *pixel with largest x-value > 1* **then**
      Mark pixel with lowest y-value as potential intersection point
    else
      Mark pixel as potential intersection point.
    end
  end
end

4.4.2 PD.M2: Adaptive masking

Method 2 looks separately at each binary image using an adaptive binary mask which run through the image. For each step in the run-through the method checks if there are equalities between the adaptive mask and the sectional area. If the mask fits the sectional area perfectly, the left middle pixel in the sectional area are added and stored as a potential intersection point. An overview of the method can be seen in Algorithm 4.3.

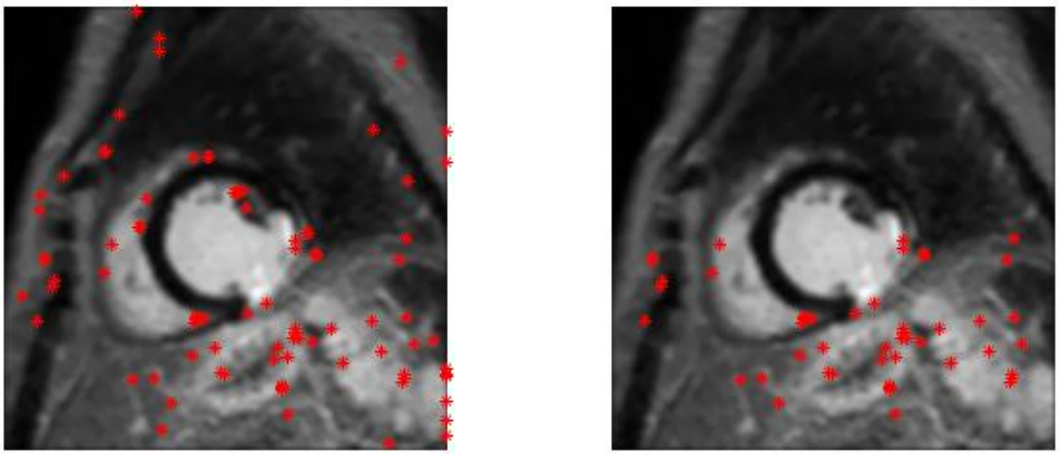
Algorithm 4.3: Point detection using an adaptive mask

```
for chosen binary images do
    create mask
    for whole image do
        run mask through each section of image
        if mask fits section do
            Mark middle left pixel in section as potential intersection point
        end
    end
    expand mask
end
```

4.4.3 Point reduction and manual classification of training-data

As a consequence of detecting points in the whole image, a majority of the detected points are not suitable as an intersection point. To reduce the inequality between suitable- and not suitable points, an algorithm is devised. The algorithm excludes points laying above the left ventricles center point, or within a radius, *r*, from the image wall.

This solution can be used since all the images are taken on equal premises where the left ventricle is located to the right for the right ventricle and the intersection point is in the lower intersection between the left and right ventricle.



(a) All detected points (b) Points left after reduction

Figure 4.7 Detected points before and after point reduction

4.5 Feature extraction

This section presents the different steps used in the *Feature Extraction* block in Figure 4.1.

For each potential intersection point, features describing the points intensity, its position and the neighboring area are extracted.

4.5.1 Intensity and position

Features describing the intensity and position are introduced because of previous knowledge of the intersection points placement. The features extracted regarding the points position and intensity are:

- Position
  - The points' pixel coordinates: (x_p, y_p).
  - The Euclidean distance, d, between the point, (x_p, y_p) and the associated center-coordinate (x_c, y_c), found using the following equation.

$$d = \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2} \tag{4.5}$$

  - The angle, $\angle$, between the point, (x_p, y_p), the center-coordinate, (x_c, y_c), and the point located at the left image wall, (x_w, y_w), positioned at the same y-axis as the center-coordinate. The angle is found using the following equation

$$\angle = \cos^{-1}\left(\frac{a^2 + d^2 - b^2}{2ad}\right) \tag{4.6}$$

where

$$a = \sqrt{(x_w - x_c)^2 + (y_w - y_c)^2}$$

and

$$b = \sqrt{(x_p - x_w)^2 + (y_p - y_w)^2}$$

- Intensity
  - The pixels' intensity value, g, in $I_{BF}(x, y)$, where $g = I_{BF}(x_p, y_p)$.

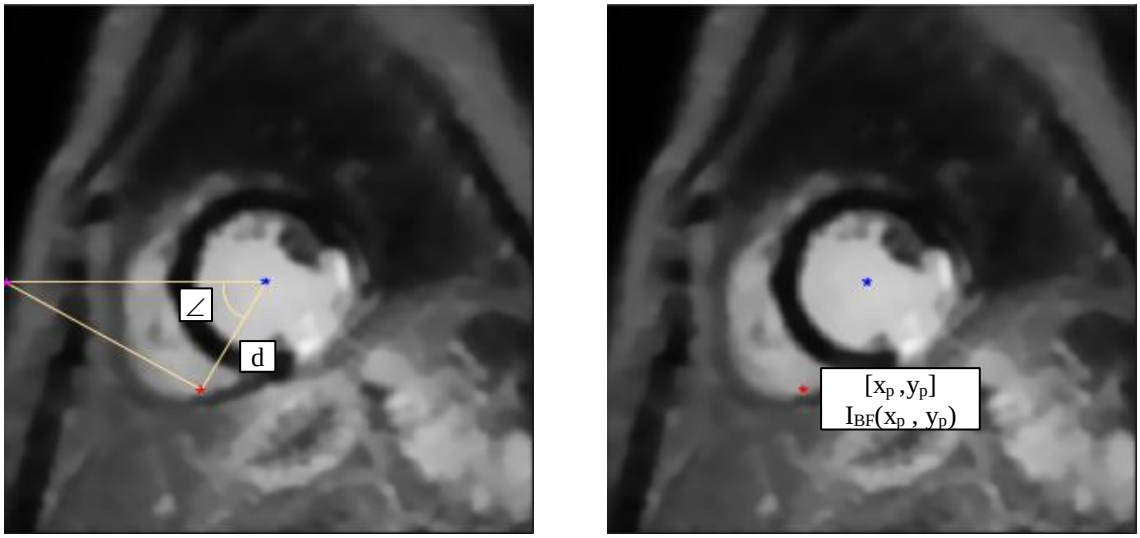


Figure 4.8 Images illustrating the extraction of position- and intensity features

4.5.2 Neighboring area

Features extracted using the neighboring area is found using the percentage of dark pixels and how the pixel intensities are distributed row-wise.

- The percentage of dark pixels, *p*, is found by looking at the number of pixels, n_P, with an intensity lower than g_L compared to the overall number of pixels, N_T, in the neighborhood. *p* is found using the following equation

$$p = \frac{n_P}{N_T} \tag{4.7}$$

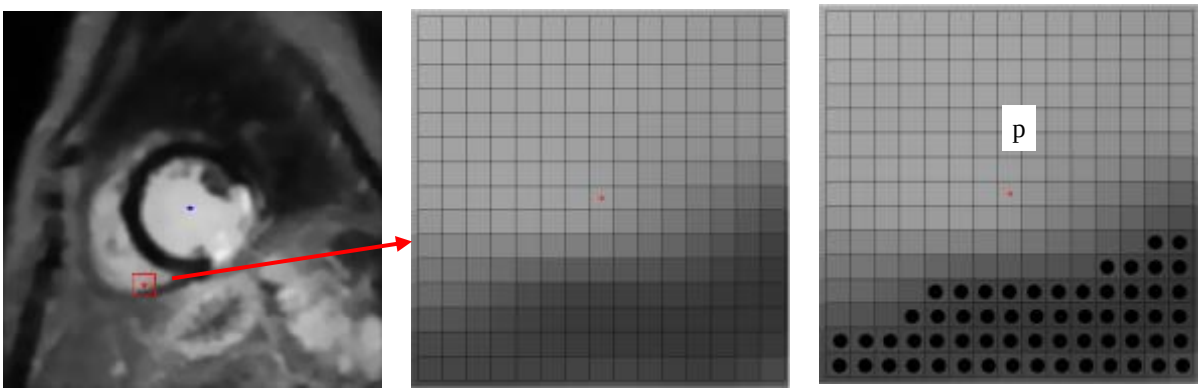


Figure 4.9 Percentage of dark pixels (marked with black circles)

- Afterwards, all pixel intensities in each row are summarized using equation (4.8).

$$\boldsymbol{V}(ry) = \sum_{rx=1}^{rL} I_{BF}(rx, ry) \tag{4.8}$$

The maximum value, v_{max}, in **V** is then found before the local minima, v_{min1} and v_{min2}, on both side of the maximum are found, as shown in equation (4.9). The absolute difference, *h*, between the two local minima is calculated and used as a neighboring feature.

$$\begin{aligned} v_{max} &= \max(\boldsymbol{V}) \\ v_{min1} &= \min(\boldsymbol{V}(1 : v_{max} + 1)) \\ v_{min2} &= \min(\boldsymbol{V}(v_{max} + 1 : end)) \end{aligned} \tag{4.9}$$

$$h = |v_{min1} - v_{min2}| \tag{4.10}$$

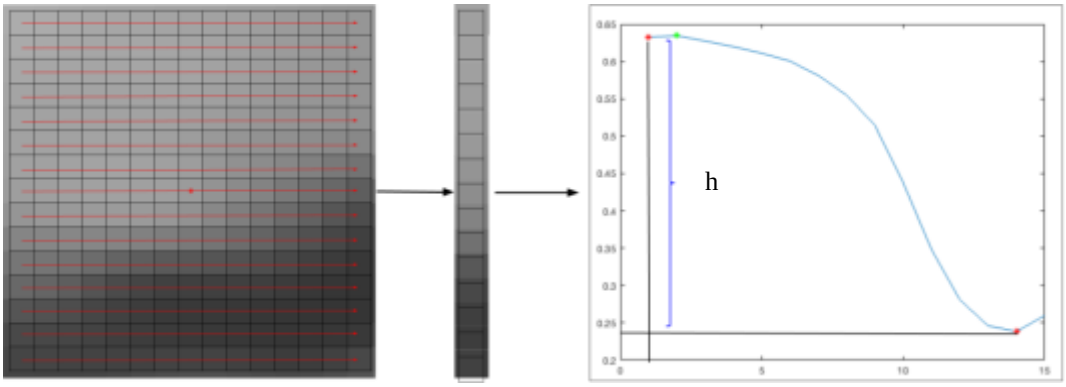


Figure 4.10 Difference between the two local minima (red points)

Combining the position-, intensity- and neighboring features results in a seven-feature vector, shown in equation (4.11).

$$\boldsymbol{f} = \begin{bmatrix} \boldsymbol{x}_p^T \\ \boldsymbol{y}_p^T \\ \boldsymbol{d}^T \\ \angle^T \\ \boldsymbol{g}^T \\ \boldsymbol{p}^T \\ \boldsymbol{h}^T \end{bmatrix} \tag{4.11}$$

4.6 Classification

This section presents the *Classification* block in Figure 4.1.

The classifier implemented in the proposed system is a RUSBoosted classifier. The classifier is trained using features extracted from points found from a dedicated training-set, as well as the points desired output response.

The desired output response is one of three classes:

1. Point laying within a Euclidean distance of *r* pixels from the actual intersection point,
2. Point laying on or within the left ventricles myocardium[3],
3. Point belonging to either of the previous classes.

Points labelled as class 1 are of further interest since these points are close to the actual intersection point. The class boundaries are shown in Figure 4.11. In the figure, the actual intersection point is marked red, the left ventricles center point is marked blue, class boundary for class 2 are marked cyan while the magenta circle shows the boundary for class 1.

[3] If a point lays in both class 1 and class 2, the point is classified as class 1.

Afterwards, the trained classifier is used to predict the class for each point found from a dedicated testing-set.

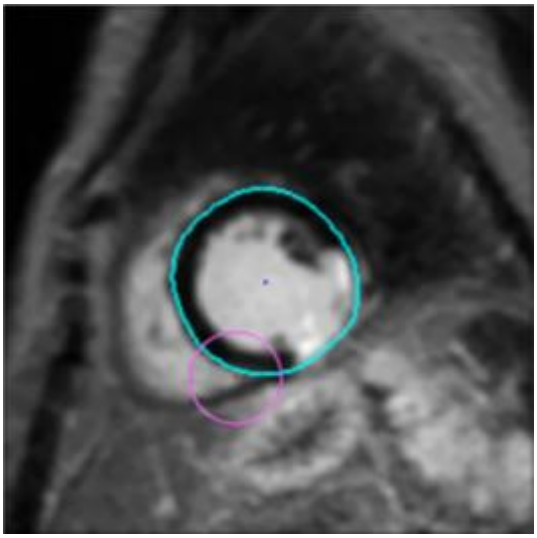


Figure 4.11 Image showing the boundaries for class 1 (magenta) and class 2 (cyan).

4.6.1 Feature standardization

Since the features extracted from each point uses different range of values, standardization of features is implemented in the classifier. Before training the classifier, each feature is standardized to have a normal distribution with the parameters $\mu_N = 0$ and $\sigma_N = 1$, where μ_N is the standardized feature mean and σ_N is the standardized standard deviation. The standardization is done using the following equation

$$\boldsymbol{f}_{s,i} = \frac{\boldsymbol{f}_{t,i} - \mu(\boldsymbol{f}_{tr,i})}{\sigma(\boldsymbol{f}_{tr,i})} \tag{4.12}$$

where $\mathbf{f}_{s,i}$ is the standardized feature *i*, $\mathbf{f}_{t,i}$ is the feature *i* from the dataset to be standardized, $\mathbf{f}_{tr,i}$ is the feature *i* from the training-set and μ and σ is the associating mean and standard deviation.

Features extracted from the test-set are also standardized, before testing, using the same equation, but instead of calculating new mean and standard deviations for the features, the same μ and σ parameters found for the training-set are used.

4.7 Decision of intersection point

This section describes the deciding block in Figure 4.1

The classifier might find several- or no points that might be potential intersection points. To decide the exact intersection point in cases where several points have been found, two decision methods, D.M1 and D.M2 is implemented in the system.

D.M1 detects an intersection points, while D.M2 alters the position of the detected intersection point. For cases where no points have been detected, an interpolation method, D.M3, finding the missing intersection points have been implemented.

4.7.1 D.M1: Find intersection point

D.M1 uses the detected points placement in regards to the center point to narrow all points in to one specific intersection point. The features taken into account are the points angle, $\angle$, and Euclidean distance, *d*, as extracted in section 4.5

An overview of the method can be seen in Algorithm **4.4**, where:
$(\boldsymbol{x}_A, \boldsymbol{y}_A)$ are the points left after removal of points due to angle size
$(\boldsymbol{x}_D, \boldsymbol{y}_D)$ are the points left after removal of points due to distance size

d_A are the mean distance found by the following equation, where *n* is the number of points

$$d_A = \frac{\sum_{ni=1}^{n} d_{ni}}{n} \tag{4.13}$$

(x_{M1}, y_{M1}) is the median point found using the following equation for either odd or even number of points.

Odd: $$x_{M1} = \left(\frac{n+1}{2}\right)^{th} x_D \qquad y_{M1} = \left(\frac{n+1}{2}\right)^{th} y_D$$

Even: $$x_{M1} = \frac{\left(\frac{n}{2}\right)^{th} x_D + \left(\frac{n}{2}+1\right)^{th} x_D}{2} \qquad y_{M1} = \frac{\left(\frac{n}{2}\right)^{th} y_D + \left(\frac{n}{2}+1\right)^{th} y_D}{2} \tag{4.14}$$

Algorithm 4.4 Detection of intersection point

Part 1

```
for all (xp, yp) do
    if ∠ < 90° do
        (xA, yA) = (xp, yp)
    end
end
if (xA, yA) is empty do
    (xA, yA) = (xp, yp)
end
```

Part 2

d_A, = *mean*((x_A, y_A))

for all (x_A, y_A) **do**
  if $d_A - r_d < d < d_A + r_d$ **do**
    $(x_D, y_D) = (x_A, y_A)$
  end
end
if (x_D, y_D) is empty **do**
  $(\mathbf{x}_D, \mathbf{y}_D) = (\mathbf{x}_A, \mathbf{y}_A)$
end

(x_{M1}, y_{M1}) = *median*($(\mathbf{x}_D, \mathbf{y}_D)$)

4.7.2 D.M2 Alteration of detected intersection point

When all possible intersection points are calculated, the method alters the intersection points placement using Algorithm **4.5**. In the algorithm (x_{IS}, y_{IS}) is the intersection point after alteration, r_x number of pixel to move and g_L a specific grayscale intensity.

Algorithm 4.5 Alteration of intersection point

$(x_{IS}, y_{IS}) = (x_{M1}, y_{M1})$
for $i := 1$ *to* r_x **do**
  if $I_{BF}(x_{M1} + r_x, y_{M1}) < g_L$ **do**
    $(x_{IS}, y_{IS}) = (x_{M1} + r_x - 1, y_{M1})$
    return (x_{IS}, y_{IS})
  end
end

4.7.3 D.M3 Find missing intersection point

In some cases, no points are detected as potential intersection points. D.3 is implemented to find these missing intersection points. The missing intersection points are found using the detected intersection points in the nearby sub-images.

An overview of the method can be seen in Algorithm **4.6**, where $(x_{IS}, y_{IS})(i)$ represents the missing intersection point in sub-image i and $(i+n)$ represents the next sub-image after sub-image $(i \pm 1)$ containing an intersection point.

Algorithm 4.6 Detection of missing intersection points

for all missing $(x_{IS}, y_{IS})(i)$ **do**

if $(x_{IS}, y_{IS})(i-1)$ and $(x_{IS}, y_{IS})(i+1) \neq (0, 0)$ **do**

$$(x_{IS}, y_{IS})(i) = \frac{(x_{IS}, y_{IS})(i-1) + (x_{IS}, y_{IS})(i+1)}{2}$$

elseif $(x_{IS}, y_{IS})(i-1)$ or $(x_{IS}, y_{IS})(i+1) = (0, 0)$ **do**

$$(x_{IS}, y_{IS})(i) = (x_{IS}, y_{IS})(i \pm 1) \pm \frac{(x_{IS}, y_{IS})(i \pm 1) + (x_{IS}, y_{IS})(i \pm n)}{2}$$

end

end

4.8 Implementation

The methods in the proposed system are implemented using the matrix-based programming language, MATLAB® [25]. MATLAB® provides built-in graphics and a vast library of prebuilt toolboxes, making this program suited for this thesis. Among these toolboxes, the Image Processing Toolbox™ [26], Statistics and Machine Learning Toolbox™ [27] and their associated pre-built functions were widely used.

4.8.1 External- internal or function provided by UiS

This section lists all the functions used in the proposed system. It gives an overview of which functions that are internal (self-written), which is collected from external sources and the functions provided by UiS. All internal functions use one- or several functions provided by MatLabs toolboxes. Below the presented table are some of the MATLAB® functions used, presented.

In appendix A all internal functions used in the study, and its associating codes are described and attached.

Main.m	External	UiS	Internal
loadDataSet.m			X
dbread.m		X	
genFeatures.m			X
organizeimage_KE.m		X	
normalize_inD.m			X
crop_heart_v2016.m		X	
bilateralFilter.m [35]	X		
Quantization.m			X
simplePointDet.m			X

maskPointDet.m			X
findLengthAndAngle.m			X
characteristics.m			X
rusBoostedClassifier.m			
Standardization.m			X
confMatrix.m			X
findInterSection.m			X
findAngDistIS.m			X
findNewIS.m			X

Table 4.1 Table which shows whether the methods are internal- external or provided by UiS.

4.8.2 MATLAB® functions

This section presents some of the MATLAB® functions used in the methods.

Quantization.m

- *imtophat.m* is used for Top-Hat filtering.
- *imadjust.m* is used for contrast adjustment.
- *multithresh.m* uses Otsu's method to find suitable threshold values.
- *imquantize.m* is used to quantize the images.

findLengthAndAngle.m

- *norm.m* is used to calculate the Euclidean distance between two coordinates
- *acosd.m* is used to find the inverse cosine in degrees

characteristics.m

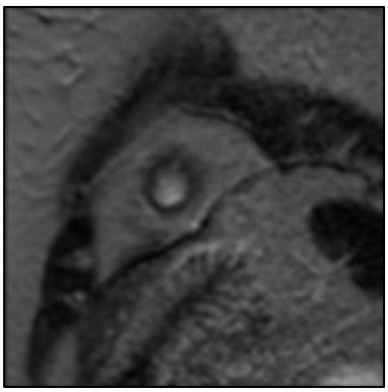
- *imcrop.m* is used to crop out the neighbourhood, creating a separate image of the neighbourhood.

5 Experiments and Results

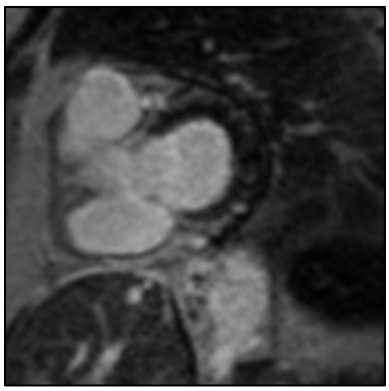
The proposed system is designed to automatically find the lower intersection point between the left- and right ventricles in LG-CMR images. Images from one or several patients, without any prior knowledge of the intersection points, are inputted in the system.

All images, except the first- and last sub-image from each patient, are sent through the system one-by-one. Each image is first processed before potential intersection points are found. Features extracted from each potential intersection point are then inputted in a trained classifier that classifies each point in one of three classes. All points classified as class 1 are then processed and narrowed into one specific intersection point. An overview of all methods can be seen in Figure 4.1.

In most patients, the sub-image of the lower part of the heart contains little- to none information about the right ventricle. In the sub-image of the upper part of the heart, the ventricles are covered by the aorta, the pulmonary artery and/or the atriums, as seen in Figure 2.1. Since the ventricles are covered or not present, these images are not processed and no intersection points are found. Images illustrating the first- and last sub-image can be seen in Figure 5.1.



(a) Image illustrating the lower part of the heart



(b) Image illustrating the upper part of the heart

Figure 5.1 Illustrations of the first- and last sub-image from one patient

To optimize the systems performance, a series of experiments are conducted. This chapter presents the experiments and the results achieved for each experiment.

First, validation of possible intersection points detected and validation of classification parameters is described. Results achieved when intersection point is decided are then presented, before results comparing the detected intersection point using manually and automatic set center point of the left ventricle is presented.

The data is split into sub-sets before it is used in the experiments. To get variety of patients with images of good and poor contrast differences and resolution, as well as variety of patients having normal- and enlarged heart, the patients were first manually split into associated groups. Thereafter, each group were split into testing- and training sets before the training- and testing-sets were merged into one dedicated training-set, T1, and one dedicated testing-set, T2.

Patients, where images were used in the creation of methods and to adjust parameters during the process were put directly in the dedicated training-set.

In total, T1 contained 3/4 of the data, while T2 contained 1/4 of the data. T1 were thereafter split into two data-sets, T1-1 and T1-2, where T1-1 contained 2/3 of the data.

Because some of the results are further used in the upcoming experiments, each experiment and its results are presented before the next experiment with its results are presented.

5.1 Detection of possible intersection points

Two quantization methods, Q.M1 and Q.M2, and two point detection methods, PD.M1 and PD.M2, are implemented in the devised system. In total, Q.M1 and Q.M2 produces seven binary images, as described in 4.3. Four of the binary images are discarded due to little- to no objective information, while the remaining three are further used and sent through PD.M1 and PD.M2.

To find the optimal combination of Q.M1 and Q.M2 and which resulting binary images to use in PD.M1 and PD.M2, an experiment was devised. The experiment was conducted on 109 images from 30 patients from dataset T1-1. The resulting amounts of points detected, and their distribution between the three classes were used to find the optimal combination.

In total, the experiment tested 12 different combinations. Three combinations of Q.M1 and Q.M2 were used and can be seen in Table 5.1. The table describes the combinations and the resulting binary images produced in each combination.

	Combination	**Resulting binary images[4]**
E.Q1	Only binary images as a result of Q.M1 is used	$\mathbf{I}_{B\_11}$, $\mathbf{I}_{B\_12}$, $\mathbf{I}_{B\_13}$
E.Q2	Only binary images as a result of Q.M2 is used	$\mathbf{I}_{B\_21}$, $\mathbf{I}_{B\_22}$, $\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$
E.Q3	Binary images from both Q.M1 and Q.M2 is used	$\mathbf{I}_{B\_11}$, $\mathbf{I}_{B\_12}$, $\mathbf{I}_{B\_13}$, $\mathbf{I}_{B\_21}$, $\mathbf{I}_{B\_22}$, $\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$

Table 5.1 Illustration of the combination of E.Q1 and E.Q2 and the resulting binary images

The resulting binary images were categorized in two groups, E.G1 and E.G2, which can be seen in Table 5.2. In E.G1 the image, from both Q.M1 and Q.M2, containing most background information was removed. In E.G2 the two images, from both Q.M1 and Q.M2, which contained most background information were removed.

	E.Q1	**E.Q2**	**E.Q3**
E1.G1	$\mathbf{I}_{B\_12}$, $\mathbf{I}_{B\_13}$	$\mathbf{I}_{B\_22}$, $\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$	$\mathbf{I}_{B\_12}$, $\mathbf{I}_{B\_13}$, $\mathbf{I}_{B\_22}$, $\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$
E1.G2	$\mathbf{I}_{B\_13}$	$\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$	$\mathbf{I}_{B\_13}$, $\mathbf{I}_{IB\_23}$, $\mathbf{I}_{B\_24}$

Table 5.2 Illustration of the groups, E1.G1 and E1.G2, the images are split into

[4] The binary images can be seen in Figure 4.4 and Figure 4.5.

The groups of images were tested in four different combinations, E.PD1, E.PD2, E.PD3 and E.PD4, with PD.M1 and PD.M2. These combinations are shown in Table 5.3

	COMBINATION
E.PD1	PD.M1 find points in E1.G1 images PD.M2 find points in E1.G1 images
E.PD2	PD.M1 find points in E1.G2 images PD.M2 find points in E1.G2 images
E.PD3	PD.M1 find points in E1.G1 images PD.M2 find points in E1.G2 images
E.PD4	PD.M1 find points in E1.G2 images PD.M2 find points in E1.G1 images

Table 5.3 Illustration of the combination of binary images used with PD.M1 and PD.M2

Results

Table 5.4 shows the total amount of points detected and their distribution in each class for all 12 combinations. In Table 5.5 the results are shown in percentage.

	E.Q1			**E.Q2**			**E.Q3**		
E.PD1	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	371	428	1540	456	582	2502	827	1010	4042
E.PD2	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	274	289	547	404	482	1351	678	771	1898
E.PD3	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	301	334	724	406	500	1514	707	834	2238
E.PD4	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	344	383	1363	454	564	2339	798	947	3702

Table 5.4 Total amounts of points detected in the 12 combinations

	E.Q1			**E.Q2**			**E.Q3**		
E.PD1	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	16 %	18 %	66 %	13 %	16 %	71 %	14 %	17 %	69 %
E.PD2	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	25 %	26 %	49 %	18 %	22 %	60 %	20 %	23 %	57 %
E.PD3	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	22 %	25 %	53 %	17 %	21 %	63 %	19 %	22 %	59 %
E.PD4	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**	**Class1**	**Class2**	**Class3**
	16 %	18 %	65 %	14 %	17 %	70 %	15 %	17 %	68 %

Table 5.5 Total amount of pixels detected in the 12 combinations shown in percentage

Due to a smaller search area for classes 1 and 2 compared to class 3, it is expected that the distribution of points will be uneven. Even though, it is of interest to find a combination that minimizes the irregularity to get minimum unevenness of data in the three classes.

Overall, the distribution of points detected in each class were, as expected, unevenly distributed. In all 12 combinations, class 3 contains the majority of points, while class 1 and 2 distributes the remaining points evenly between themselves.

Combining E.PD1 and E.PD4 with E.Q1, E.Q2 or E.Q3 results in a poor distribution:

- Class 3 contains minimum 65% of the points.
- Class 1 contains maximum 16% of the points.

Combining E.PD2 and E.PD3 with E.Q1 results in the best overall distribution:

- Class 1 contains respectively 25% and 22% of the points.

To minimize the risk of encountering images where no points are detected, it is also desirable to find the combination that produces a sufficient amount of points. By looking at the amount of points detected using E.PD2 or E.PD3 with E.Q1, there were only 274 and 301 points detected in the 109 images.

Overall, all combinations using E.Q1 or E.Q2 results in fewer detected points in class 1 than desired. The maximum average of potential intersection points were 3,7 and were found using E.PD3 in combination with E.Q2 which only had a distribution of class1 points of 17%.

The combination that satisfies the criterion of minimizing the distribution, as well as satisfies the criterion of amount of points the most, are combination E.PD2 or E.PD3 with E.Q3. Due to a slightly better distribution combining E.PD2 with E.Q3, this combination is chosen for further use.

5.2 Features used for classification

Several features were chosen as potential parameters to be used for training of the classifier and testing using the trained classifier. The chosen features can be seen in (4.11). A problem that might occur if there are too many parameters is that the classification model gets overfitted, giving a poor predictive performance.

Too many parameters, or several parameters representing the same, can also result in an unnecessary slow model. To reduce the chance of an overfitted- or a slow model, an experiment was devised to decide if the features found were the optimal combination of features, or if this combination gave poorer predictive performance than combinations with fewer features.

For this experiment, the features were split into four groups. Each of the four groups respectively represents the points coordinates, the points position relative to the left ventricles center point, the points grayscale intensity and the points neighboring area. For this experiment the groups of features are labelled 1-4 and can be seen in (5.1)

$$\begin{array}{ll} 1: & \{ x_p , y_p \} \\ 2: & \{ d, \angle \} \\ 3: & g \\ 4: & \{ p, h \} \end{array} \qquad (5.1)$$

In the first three groups, there are no system parameters directly affecting the features. In group four, the features are directly affected by the size of the surrounding neighborhood, NB. Due to the direct effect of the NB size, the experiment was conducted by testing both single, and multiple combinations of features in combination with different sizes of the NB.

Each of the combinations were trained and tested using dataset T1_1 on the described classifier with a 10-fold cross validation. Optimal combination was evaluated based on system performance, in terms of the overall accuracy for all classes as well as the precision rate for class 1.

Results

The results are presented for both single features in combinations with various sizes of the NB, and for multiple features in combination with various NB sizes. Since a k-fold cross validation is used, the classifier finds the optimal model for each trial. There will therefore be some small differences in the results for combination of features not affected by the NB size in combination with different NB sizes.

Single feature combination with various NB size

Using single feature combinations (FC) in combination with different NB sizes resulted in overall good system performance, for FC=1 and FC=2 with an overall correct classification of 89%-90% and 92% and a precision rate of 84% and 87%. For FC = 3 only 38% of points were classified correctly, and a maximum of 38% of the points classified as class 1 were actual class 1. FC = 4 had a slightly better overall accuracy of 58% with NB size of 17x17, but the same poor precision rate as FC = 3.

NB: / FC:	9x9	13x13	17x17
1	89%	90%	90%
2	92%	92%	92%
3	38%	38%	38%
4	53%	55%	58%
Accuracy			

NB: / FC:	9x9	13x13	17x17
1	84%	84%	84%
2	87%	87%	87%
3	36%	36%	37%
4	38%	38%	38%
Precision			

Multiple feature combinations with various NB size

When multiple FCs was combined, the overall results improved slightly. FC = {3,4} gave, by far, the poorest results of multiple combinations independent of the NB size. The best overall accuracy, 63%, for this FC were in combination with a NB size of 17x17 and 21x21.

The remaining FCs had overall similar results in all NB size combinations. System performance were overall best for combinations were FC = 1 and FC = 2 were included.

FC = {1,2} in combination with NB = {21x21} and FC = {1,2,3,4} in combination with NB = 17x17 gave the best accuracy rate with 94% correct classifications. FC = {1,2,3} in combination with NB = 9x9, FC = {1,2,4} in combination with FC = 13x13, 17x17 and 21x21 and FC = {1,2,3,4} in combination with FC = 17x17 gave the best precision rate. Results were 89% of the points classified as class 1, actually were class 1.

NB: / FC:	9x9	13x13	17x17
{1,2}	93%	93%	93%
{1,3}	90%	90%	90%
{1,4}	90%	90%	90%
{2,3}	92%	92%	92%
{2,4}	92%	92%	93%
{3,4}	59%	62%	63%
{1,2,3}	93%	93%	93%
{1,2,4}	93%	93%	93%
{1,3,4}	90%	90%	90%
{2,3,4}	93%	93%	93%
{1,2,3,4}	93%	93%	94%
Accuracy			

NB: / FC:	9x9	13x13	17x17
{1,2}	88%	88%	88%
{1,3}	85%	84%	84%
{1,4}	84%	84%	85%
{2,3}	87%	87%	87%
{2,4}	87%	87%	87%
{3,4}	37%	39%	43%
{1,2,3}	89%	88%	88%
{1,2,4}	88%	89%	89%
{1,3,4}	84%	85%	85%
{2,3,4}	87%	87%	87%
{1,2,3,4}	88%	88%	89%
Precision			

5.3 Detection of intersection point

Detection and classification of possible intersection points leaves everything from none to several points that might be suitable as an intersection point. To narrow these points into one specific intersection point, several methods were considered interesting.

Due to unavoidable misclassifications, all points might not be as suitable as others. Considering this, the experiment tested methods with different angles, finding the method that pin-pointed the intersection points most accurately.

The experiment tested both pure mathematical methods as median and mean, found using equation (4.14) and (4.13) and methods where the points placement were considered, to decide the intersection point.

Three methods, E3.M1, E3.M2 and E3.M3, where the points placement were considered were tested.

- E3.M1 considered each points angle-feature, $\angle$, found in section 4.5 The possible intersection point was narrowed in using Algorithm **4.4**, *part 1*, before the intersection point were found by finding the median of the narrowed points, using equation (4.14).

- E3.M2 used each points Euclidean distance, *d*, found in section 4.5, and used this distance with Algorithm **4.4**, *part 2*, to narrow in the detected intersection points. When the points were narrowed in, the median of the points were found, using equation (4.14), and used as the intersection point.

- E3.M3 used a combination of both E3.M1 and E3.M2 and found the intersection point by both *part 1* and *part 2* of Algorithm **4.4**

During pre-processing and quantizing of the images, the ventricles size and shape might have been altered and narrowed in. Because of this, the experiment was split into two separate sub-experiments, SE1 and SE2. SE1 found the intersection point by testing the proposed methods on all points classified as class 1, while SE2 altered the placement of the intersection points found in SE1.

For both SE1 and SE2, only images, where points of class 1 were detected, were validated. Images where no points were detected, were ignored and not implemented in the results. The results were validated using the resulting Euclidean distance between the detected intersection points, and the intersection point set by a cardiologist. The Euclidean distance was found using (4.5). Average-, minimum- and maximum Euclidean distance as well as the percentage of points laying within a Euclidean distance of *r* pixels, were found and used in the determination of the most optimal method.

A Euclidean distance of *r=10* pixels were chosen as the limitation of whether the detected intersection point was satisfying or not. If the detected intersection point laid within a Euclidean distance of *r=10* from the intersection point set by the cardiologist, the intersection point was classified as a satisfying result.

5.3.1 SE1: Detection of intersection point

In SE1, the classifier was trained using dataset T1, before dataset T2 was tested on the trained classifier. In total, T2 consisted of 102 images. In 10 of these images, the classifier did not classify any points as class 1.

Results

	Median	**Mean**	**E3.M1**	**E3.M2**	**E3.M3**
Minimum	0,83	0,90	0,83	0,83	0,83
Maximum	46,39	46,39	48,33	46,39	48,33
Average	11,74	12,04	11,72	12,15	11,84
Percentage	59 %	57 %	60 %	54 %	60 %

Overall, using E3.M2 to find the intersection point gave the poorest result. Only 54% of the intersection points laid within a Euclidean distance of 10 pixels from the manually set intersection point. The average Euclidean distance were the highest with 12,11 pixels.

The remaining methods generally had better results without major differences in the minimum-and average Euclidean distance. For the maximum Euclidean distance, both median and mean had the best overall results with 44,18 and 42,73 pixels.

E3.M1 and E3.M3 gave the best results of intersection points laying within a Euclidean distance of 10 pixels, from the manually set intersection point, with 60%. E3.M1 also had the overall lowest average Euclidean distance.

5.3.2 SE2: Alteration of detected intersection point

Since the shape and size of the objects might have been narrowed in during pre-processing and quantization, the points found by PD.M1 and PD.M2 might lay further into the right ventricle than necessary.

The experiment was devised to see if the results were improved by altering the detected intersection points placement. Since Mean and E3.M2 presented the poorest result in S1, these were left out from this experiment. The alteration was done using Algorithm **4.5** with three values of r_x, $r_x = 10$, $r_x = 15$ and $r_x = 20$.

Results

	Median	E3.M1	E3.M3		Median	E3.M1	E3.M3		Median	E3.M1	E3.M3
Minimum	0,18	0,18	0,18		0,50	0,50	0,50		0,50	0,50	0,50
Maximum	47,42	48,95	48,95		51,74	51,55	51,74		56,19	55,96	56,19
Average	10,42	10,10	10,18		10,69	10,24	10,19		11,18	10,57	10,46
Percentage	67 %	68 %	67 %		65 %	67 %	68 %		65 %	68 %	70 %
	$r_x = 10$				$r_x = 15$				$r_x = 20$		

For all combinations of r_x, the minimum and average distance, as well as the percentage of points laying within a distance of 10 pixels from the real intersection point, were improved compared to the points found in SE1. The maximum Euclidean distance were worse for all combinations of r_x.

Method E3.M3 gave the best result in all combination of r_x, while median in combination with $r_x = 20$ gave the poorest results for all combinations. E3.M3 in combination with $r_x = 20$ found most points laying within a distance of 10 pixels from the real intersection point, while E3.M3 in combination with $r_x = 10$ had the best average distance.

Since E3.M3 in combination with $r_x = 20$ found most points within a distance of 10 pixels from the real intersection point, and the difference between its average distance and the best average distance were only 0,28 pixels, this combination was used for further processing.

5.4 Interpolation of missing intersection points

If the classifier does not classify any points as class 1, no intersection point is either found. Since it is desirable to find an intersection point for all the inputted images, an interpolation method is implemented in the system. The interpolation method uses information about intersection points to the nearby images to decide the missing intersection point.

The missing intersection points were found using Algorithm **4.6**

To validate the results, the resulting Euclidean distances were used in terms of average-minimum and maximum distance as well as the percentage of points laying within a Euclidean distance of 10 pixels.

Results

Number of not detected points	0
Minimum	0,50
Maximum	56,19
Average	11,13
Percentage	67%

The presented results were poor in comparison to the results found in section 5.3.2. Number of detected intersection points which were satisfying was reduced from 70% to 67%. There was no change in minimum- and maximum distance, but the average distance was worsened with 0,67 pixels.

5.5 Manually set heart center compared to automatically found heart center

It was desired to propose a fully automatic system. Due to this, the proposed system uses the left ventricles center point, found automatically by a function previously presented at the UiS, instead of the manually set intersection point.

Since the proposed system bases itself on the automatically found center point of the left ventricle, it was of interest to see if there were any differences in the outputted intersection points found using manually- or automatic set heart centers.

An experiment was devised to compare the differences in the outputted results from both the manually- and the automatic set heart center. The experiment only focused on the intersection points found before any interpolation of missing intersection point. The eventual differences would be in the detection and classification of points of class 1.

The experiment was conducted by finding intersection points using method E3.M3, as described in Algorithm **4.4**. Results were validated using the resulting Euclidean distances, in

terms of the average- minimum and maximum distance, percentage of points laying within a distance of 10 pixels from the real intersection point and number of missing points.

Results

	Automatic	Manually
Number of not detected points	10	5
Minimum	0,50	0,50
Maximum	56,19	56,19
Average	10,46	10,98
Percentage	70%	68%

In total 102 images were tested in the experiment. Both with use of manually- and automatic found center point, there were some images where an intersection point was not found. Using automatic set heart center the intersection point was not found in 10 of the images. With use of manually set heart center this number was halved.

For both the average distance and the percentage, the result was poorer with 2% and 0,52 pixels when the manually set heart center was used instead of the automatic set heart center.

6 Discussion

This chapter discusses the results found in the previous chapter.

6.1 Data material

The data material used in this study is a collection of LG-CMR images provided to UiS by the Department of Cardiology at Stavanger University Hospital. In addition to the images, associating mat-files where the left ventricles center point, the intersection point, the scarred- and the healthy myocardium plotted by a cardiologist where provided.

In total, the data set consists of images taken minimum one year after the infarction from 91 post-MI patients. Due to lack of intersection point or center point in some of the mat-files, only images from 60 patients were further used in the study. It would have been desirable with a larger set of data, to test the system on several patients. Images taken earlier than one year after the infarction were available, but because of swelling of the heart and scar, it was decided to only use images taken after the swellings had settled. The provided data were therefore considered as a good starting point.

6.2 Image processing

The proposed system uses the left- and right ventricles edges to find points that might be potential intersection points. It was therefore important to have minimum damage to the edges when smoothing was applied to remove the noise. During the early stages of the study, two noise removal methods were considered interesting: a Gaussian smoothing filter and a Bilateral filter. Both methods showed visually good smoothing results, but the Bilateral filter showed better visual results when it came to preservation of the edges and was therefore used without any further experimenting.

Quantization

Even though the images were normalized and contrast adjusted, the intensity values for the ventricles varied from patient to patient. It was therefore decided to quantize the images into several layers and look at the different layers separately. This resulted in several binary images.

Using quantization can be both positive and negative. If chosen thresholds are optimal, quantization can successfully extract the desired objects from the background, making it easy to find what is desired. In other cases, quantization can cause problems. Information about the object's structure and size can be altered by being levelled at different levels, or that background and object are levelled at the same level.

To reduce, and prevent big loss of information, precautions were taken. It was decided to implement two quantization processes, Q.M1 and Q.M2. With use of two quantization processes, the chance of information loss, was reduced.

In addition, Q.M1 and Q.M2 were quantized with different number of thresholds in case information about the ventricles got lost using one- or the other set of thresholds. Three and four levels of quantization were considered preferable. Using fewer, or more, levels were considered unprofitable since this could increase the chance that background and ventricles were merged, or that the ventricles was split between several layers.

An experiment was devised to see whether it was useful to use both Q.M1 and Q.M2 or not. In addition, the experiment tested which of the binary images that was most profitable to use.

The distribution of points detected in the three classes was overall best using only Q.M1 in combination with the three most objective related binary images. This combination resulted in a distribution of points of class 1 of 25%. Even though the distribution was good, only an average of 2,5 potential intersection points was detected in each image. Using both methods in combination with the same binary images resulted in a poorer distribution, but several potential intersection points. Use of both methods were considered preferable to reduce the risk of not finding any relevant points.

6.3 Point detection

Initially, only PD.M1 were initialized in the system to detect possible intersection points. To reduce the chance of encounter images, where no points were detected or none of the detected points found by PD.M1 were suitable as an intersection point, PD.M2 was also implemented.

Using both PD.M1 and PD.M2 increased the number of potential intersection points and thereby the chance that the classifier would find suitable points in all images. Using both methods also increased the detection of points not suitable as intersection points. It was however considered preferable to use both methods since this increased the chance of detecting the intersection point.

6.4 Feature extraction

Each potential intersection point was described by seven features representing the points location in the image, location in relative to the left ventricles' center point, grayscale intensity and its surrounding neighborhood. The features were chosen due to previous knowledge about the position of the intersection points.

Having too many, too few, or no suitable features would increase the risk of a poor predictive classification model. An experiment was conducted to find the optimal combination of features, in combination with different sizes of the surrounding neighborhood. The results showed prominent results where the pixels coordinate and position in relation to the center coordinate were used. Use of only the intensity feature, the features describing the points'

neighborhood or these in combination gave poor results with an accuracy of maximum 59% and a precision rate of maximum 43%.

The best overall combination was found using all features and a neighborhood size of 17x17 pixels. This resulted in an accuracy of 94% and a precision rate of 89%. In most of the results where multiple combinations of features were used in combination with different NB sizes, minimal difference was detected. It is therefore uncertain if other combinations could improve results in further use.

Because the intensity and neighborhood features showed poor results, as shown in section 5.2, when tested alone, it was considered to remove these from further use. It was nevertheless decided to bring them along since the overall result were best when all features were used.

In addition, it should be noted, that the validation was done using a 10-fold cross validation. Use of several folds or leave-one-out cross validation might provide different results, where other feature combinations comes out with better results.

6.5 Decision of intersection point

A Euclidean distance limit of 10 pixels from the real intersection point was set to determine if the points' position was satisfying or not. It was also used to determine if the points used to train the classifier were class1 or not. This limit was set based on visual observations, by the writer of this thesis, and was not discussed with an expert cardiologist. Narrowing in- or expand this limit will change the outcome of the results.

To reduce the potential intersection points down to one single intersection point, several methods were considered interesting. Median and mean were chosen as two potential methods. This way, if there were any points completely misplaced, these would be removed and not considered. Mean was chosen as another method, based on the same basis as the median, giving misplaced points little effect in the decision of the intersection point.

In some images, there were other bright objects, in addition to the ventricles, in the image.This resulted in more misplaced potential points. Methods that looked at the points' distance, d, and/or the points' angle, $\angle$ were also tested. The main idea for choosing these methods was to remove points that was placed far away- or too far to the right of the center point before intersection point was chosen.

None of the proposed methods showed satisfying results. Only a maximum of 60% of the points laid within a distance of 10 pixels from the true intersection point, and the minimum average distance were 11,72 pixels.

Since the shapes of the ventricles risk getting altered during the quantization process, the associated potential points were also placed into the right ventricle, further away from the edge than wanted. It was therefore implemented a method which altered the position.

Alteration of the points improved the overall results. Most points were placed closer to the real intersection point, giving a lower average distance, as well as a higher percentage of

satisfying points. Even though, the maximum distance where worsened, meaning that some points were moved away from the real intersection point.

6.6 Use of manual and automatic center point

Since it was desirable to create a fully automatic system, use of automatic chosen center point was used throughout the study. To see if there were any differences in the use of automatic- and manually set center point, the system was tested with both points. Before conduction of the experiments, manually set heart center was expected to provide the best results.

The results were somewhat improved using manually set heart center. The number of images where no intersection point was found were halved from ten to five images, with use of manually set heart center, compared to automatic set heart center. Even though the numbers of missing intersection points were reduced, the overall average and percentage of satisfying intersection point were somewhat poorer.

Before drawing any exact conclusion about the system performance, the test-data must be taken into account. Only images from 15 patients were tested in experiment 5.3 to 5.5. Data from the 15 patients were also a variety of patients with enlarged- and normal hearts and patients where the images had both poor and good contrast.

On a general basis, the systems' performance was overall good for images with good resolution and clear edges. The poorest results were found in images were the scar is placed right at the intersection point.

7 Conclusion

In this thesis, an approach for automatic detection of the lower intersection point between the left- and the right ventricle in LG-CMR images has been proposed. Due to little- to no information about the ventricles in the upper- and lower sub-images from each patient, these were not considered in this study.
With use of image quantization, the ventricles are extracted from the background. After extraction, points that might be of interest are detected, before features describing the points are computed and used to predict whether the points are of further interest or not. Points found to be of further interest are then processed and narrowed into one specific point, the intersection point.
Due to some misclassifications, or no classification of relevant points, the intersection point was not detected in all images. In these images, an interpolation method was used to fill in the missing intersection points.
Results show that in 70% of the images, where points were detected, the detected intersection point was satisfying. The results also show that the average Euclidean distance for each point was 10,46 pixels, which was slightly higher than the desired limit of 10 pixels.
After interpolation of missing points, the result of satisfying intersection points was reduced to 67%, and the average distance was increased to 11,13 pixels.
Visual results show that the poorest results are found in images with poor contrast, no visible right ventricle or where the scar is placed at the intersection. Best results are found in images with good contrasts.
Due to poor contrast and/or scar placed at the intersection, the extraction of ventricles worked poorly, or did not work at all for some of the images. In these images, either the background and ventricles were merged or both ventricles were merged together.
Overall, there are several weaknesses in the various methods which can be further improved. Finding a method which detects possible points with use of the grayscale image, and not only a quantized version of it might be one improvement. Another improvement is to create a method that better narrows the potential points. Using methods that look at the points in relation to the segmented myocardium is one potential solution.

7.1 Future work

Improvement of interpolation method
The interpolation method implemented in the system is not optimal and has room for improvements. Using the distance, d, and angle, $\angle$, or autoregressive modelling for interpolation might be a better solution than to only rely on the already detected intersection points.

Extrapolation of points in top- and bottom sub-image
The top- and bottom sub-images were ignored in this study due to little- to none information about the ventricles. To automatically detect intersection points in these images, an extrapolation method needs to be implemented. The extrapolation method can be implemented using information about already detected intersection points, such as position or closeness to the myocardium.

Detection of points

In some of the images, the quantization methods were not optimal, due to the scars position or poor contrast resolution. Proposing methods where points are detected with use of the grayscale image might provide better results for these images.

Bibliography

[1] World Health Organization, "Cardiovascular disease - Strategic priorities," [Online]. Available: http://www.who.int/cardiovascular_diseases/priorities/secondary_prevention/country/en/index1.html. [Accessed 04 May 2017].

[2] American Heart Association Inc., "heart.org," 27 January 2017. [Online]. Available: http://www.heart.org/HEARTORG/Conditions/HeartAttack/AboutHeartAttacks/About-Heart-Attacks_UCM_002038_Article.jsp#.WRCvb7HJJsN. [Accessed March 2017].

[3] J. Mannsverk, T. Wilsgaard, E. B. Mathiesen, M.-L. Løchen, K. Rasmussen, D. S. Thelle, I. Njølstad, L. A. Hopstock and K. H. Bønaa, "Trends in modifiable risk factors are associated with declining incidence of hospitalized and non-hospitalized acute coronary heart disease in a population," *Circulation,* vol. 133, no. 1, pp. 74-81, 5 January 2015.

[4] P. K. Stein, D. Sanghavi, P. P. Domitrovich, R. A. Mackey and P. Deedwania, "Ambulatory ECG-Based T-Wave Alternans Predicts Sudden Cardiac Death in High-Risk Post-MI Patients with Left Ventricular Dysfunction in the EPHESUS Study," *Journal of cardiovascular electrophysiology,* vol. 19, no. 10, pp. 1037-1042, 2008.

[5] R. E. Kleiger, J. P. Miller, J. T. Bigger and A. J. Moss, "Decreased heart rate variability and its association with increased mortality after acute myocardial infarction," *The American journal of cardiology,* vol. 59, no. 4, pp. 256-262, 1987.

[6] S. Yusuf, S. Hawken, S. Ôunpuu, T. Dans, A. Avezum, F. Lanas, M. McQueen, A. Budaj, P. Pais and J. Varigos, "Effect of potentially modifiable risk factors associated with myocardial infarction in 52 countries (the INTERHEART study): case-control study," *The lancet,* vol. 364, no. 9438, pp. 937-952, 2004.

[7] L. P. Kotu, K. Engan, T. Eftestøl, L. Woie, S. Ørn and A. K. Katsaggelos, "Local binary patterns used on cardiac MRI to classify high and low risk patient groups," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, Bucharest, Romania, 2012.

[8] L. Woie, T. Eftestøl, K. Engan, J. T. Kvaløy, D. W. Nilsen and S. Ørn, "The heart rate of ventricular tachycardia following an old myocardial infarction is inversely related to the size of scarring," *Europace,* vol. 13, no. 6, pp. 864-868, 2011.

[9] K. Engan, L. Woie and T. Eftestøl, "Defining angular and radial positions and parameters for myocardial pixels in cardiac MR images," in *Computing in Cardiology Conference (CinC), 2014*, Cambridge, MA, USA, 2014.

[10] L. P. Kotu, K. Engan, T. Eftestøl, S. Ørn and L. Woie, "Segmentation of scarred and non-scarred myocardium in LG enhanced CMR images using intensity-based textural analysis," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, Boston, MA, USA, 2011.

[11] L. P. Kotu, K. Engan, K. Skretting, S. Ørn, L. Woie and T. Eftestøl, "Segmentation of scarred myocardium in cardiac magnetic resonance images," *ISRN Biomedical Imaging,* vol. 2013, 2013.

[12] L. P. Kotu, K. Engan, K. Skretting, F. Måløy, S. Ørn, L. Woie and T. Eftestøl, "Probability mapping of scarred myocardium using texture and intensity features in CMR images," *Biomedical engineering online,* vol. 12, no. 1, p. 91, 2013.

[13] K. Engan, V. Naranjo, T. Eftestøl, S. Ørn and L. Woie, "Automatic Segmentation of the Epicardium in Late Gadolinium Enhanced," in *Computing in Cardiology Conference (CinC),*

2013, Zaragoza, Spain, 2013.

[14] E. N. Marieb and K. Hoehn, Human Anatomy and Physiology, vol. seventh edition, Pearson Benjamin Cumming, 2007, pp. 677-712.

[15] Blausen.com staff (2014), ""Medical gallery of Blausen Medical 2014"," [Online]. Available: DOI:10.15347/wjm/2014.010.

[16] Blausen.com staff (2014), ""Medical gallery of Blausen Medical 2014"," [Online]. Available: DOI: 10.15347/wjm/2014.010.

[17] H. Karlsaune, T. Digre, S. Sneeggen, R. E. S. Govatsmark and K. H. Bønaa, "Årsrapport 2015 - Med plan for forbedringstiltak," Norsk Hjerteinfarktregister, Trondheim, 2016.

[18] L. Woie, K. Engan, T. Eftestøl, A. I. Larsen and S. Ørn, "The Localization and Characterization of Ischemic Scars in relation to the Infarct Related Coronary Artery Assessed by Cardiac Magnetic Resonance and a Novel Automatic Postprocessing Method," *Cardiology research and practice,* vol. 2015, 2015.

[19] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images," *Sixth International Conference on Computer Vision,* pp. 839-846, January 1998.

[20] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE transactions on systems, man, and cybernetics,* no. 9.1, pp. 62-66, 1979.

[21] O. Marques, Practical image and video processing using MATLAB, John Wiley & Sons, 2011, pp. 299-334.

[22] P. Soille, Morphological Image Analysis: Principles and Applications, Springer Science & Buisness Media, 2013.

[23] E. R. Dougherty and R. A. Lotufo, Hands-on Morphological Image Processing, SPIE - The International Society for Optical Engineering, 2003.

[24] C. Solomon and T. Breckon, Fundamentals of Digital Image Processing: A practical approach with examples in Matlab, John Wiley & Sons, 2011, pp. 197-234.

[25] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans,* vol. 40, no. 1, pp. 185-197, 2010.

[26] G. M. Weiss, "Mining with rarity: a unifying framework," *ACM Sigkdd Explorations Newsletter,* vol. 6, no. 1, pp. 7-19, 2004.

[27] G. E. Batista, C. Ronaldo and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM Sigkdd Explorations Newsletter,* vol. 6, no. 1, pp. 20-29, 2004.

[28] C. Drummond and R. C. Holte, "C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling," vol. 11, Citeseer Washington DC, 2003.

[29] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of artificial intelligence research,* vol. 16, pp. 321-357, 2002.

[30] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *icml,* vol. 96, 1996, pp. 148-156.

[31] N. V. Chawla, A. Lazarevic, L. O. Hall and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *European Conference on Principles of Data Mining and Knowledge Discovery,* 2003, pp. 107-119.

[32] The MathWorks Inc., "The Language of Technical Computing," [Online]. Available: https://se.mathworks.com/products/matlab.html. [Accessed 3 June 2017].

[33] The MathWorks Inc., "Perform image processing, analysis, and algorithm development," [Online]. Available: https://se.mathworks.com/products/image.html. [Accessed 3 June 2017].

[34] The MathWorks Inc., "Analyze and model data using statistics and machine learning," [Online]. Available: https://se.mathworks.com/products/statistics.html. [Accessed 3 June 2017].

[35] D. Lanman, "Bilateral Filtering," 2006.

[36] J. Cheng, S. W. Foo and S. Krishnan, "Automatic detection of region of interest and center point of left ventricle using watershed segmentation," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, Kobe, Japan, 2005.

[37] World Health Organization, "ICD-10 Version:2016," 2016. [Online]. Available: http://apps.who.int/classifications/icd10/browse/2016/en#/I20-I25. [Accessed April 2017].

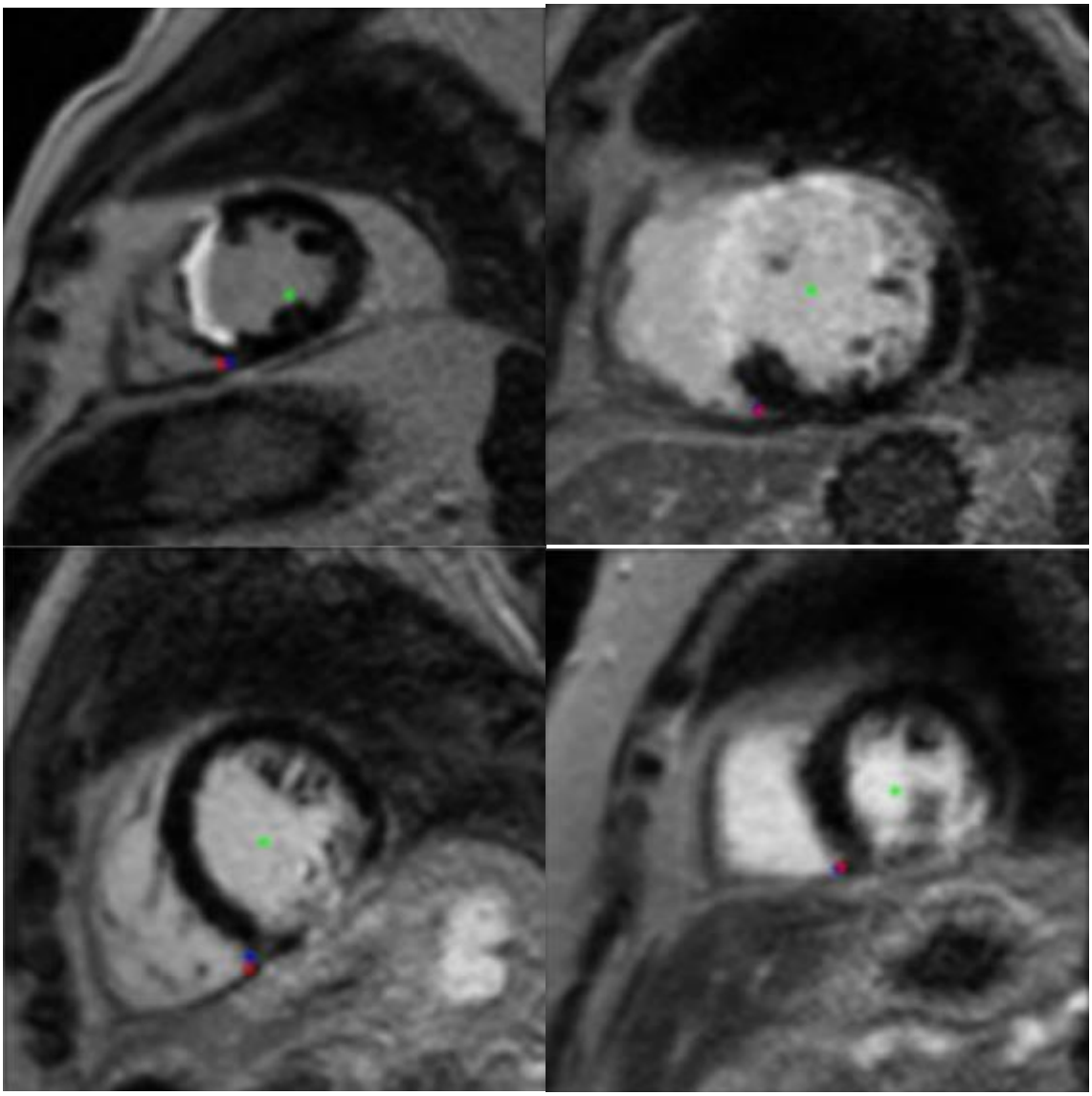
Appendix A Matlab code

Main.m	
	Main program where parameters are set and data read in.
loadDataSet.m	
	Splits the dataset used in this project into a training- and testing-set.
genFeatures.m	
	Inputs: Original image and parameters Run through the pre-processing, quantizing, point detection and feature extraction functions, before outputting the extracted features.
normalize_inD.m	
	Inputs: Original images Normalizes the inputted images.
bilateralFilter.m	
	Source: Douglas R. Lanman, Brown University Inputs: Normalized image Pre-processes the images using bilateral filter
Quantization.m	
	Inputs: Bilateral filtered image Quantizes the images using Q.M1 and Q.M2. Outputs binary images.
simplePointDet.m	
	Inputs: Binary images Detect points in inputted images using PD.M1.
maskPointDet.m	
	Inputs: Binary Images. Detect points in inputted images using PD.M2
findLengthAndAngle.m	
	Inputs: Detected points Calculates the Euclidean distance, d, and angle $\angle$

characteristics.m	
	Inputs: Detected points Calculate *g, p, h*
rusBoostedClassifier.m	
	Source: Matlab classification learner app. Inputs: training-set and testing-set Train classifier with training-set. Predict classes for testing-set
standardization.m	
	Inputs: training- and/or testing-data. Standardize inputted data.
confMatrix.m	
	Inputs: Predicted and real classes for testing-set Calculates Accuracy and Precision rate.
findInterSection.m	
	Inputs: Detected points Separate patients, images and detected points.
findAngDistIS	
	Inputs: Detected points Find intersection point using D.M1
findNewIS.m	
	Inputes detected intersection points Alter the points position by using D.M2

Appendix B Detection of intersection point

Images where detection of intersection point gave good results.
Blue mark is original point, while red mark is automatic detected point.



Images where detection of intersection point gave poor results.