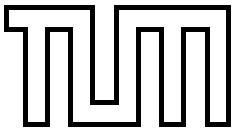
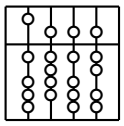


Technische Universität München
Fakultät für Informatik



IDP

Extraction of Vessels from X-Ray Angiograms

Titus Rosu

Advisor:	Prof. Dr. R. Lasser
Supervisor:	Andreas Keil
Date:	May 26, 2008

Contents

Chapter 1

Introduction

Many applications from computer based surgery involve an automatic and exact localization of vessels in contrasted radiographs. However the segmentation of 2D images is more difficult than segmenting 3D images because of less information received from the data. In addition the radiographs are often blended because of the nature how information is projected on the images (e.g. blended with bones and different vessel parts). The objective of this IDP in mathematics is the implementation and testing of different methods of vessel segmentation based on eigen-value analysis.

Chapter 2

Data

The implemented methods are tested on different angiography sequences (esp. rotational angiography with stationary C-arms). The images have no radial distortion since they are acquired with a flat panel detector. The pixel spacing is $0.3mm$ or $0.6mm$ which is accurate for segmenting the coronary arteries, our main target vessels.

The vessels in the DICOM-Sequences are dark on light background (the gray value of the contrasted coronary arteries represents the measured radiation). We extracted some slices on which we tested the implemented methods.

Figure 2.1: Coronary Arteries

Chapter 3

Methods

In this chapter the different implemented vessel detection algorithms are described as well the different approaches of pre- and post-processing image algorithms to reach significant final results in the segmentation. The programming language for all implemented methods is C++ together with the ITK-Library[1] v3.x [ITK].

3.1 Preprocessing

Before segmenting vessels we improve the images by adjusting the signal-to-noise ratio and subtracting the background in the preliminary steps. The influence of the image processing [GW02]-[Jäh02] steps on the final segmentation results will be also analyzed.

3.1.1 Cropping

The images used from the data sets have on their borders very high or low frequencies (dark or light areas). These regions with no interesting information, unlike the sought-after coronary arteries, affects the used improving image algorithms e.g. thresholding with the variance of the mean value of the pixel-intensities. Therefore areas with a width of approximately 50 pixels are cut off on each border.

[1] Insight Segmentation and Registration Toolkit (http://www.itk.org/)

3.1.2 Adjusting Intensities

After the cropping step the image intensities are adjusted. Normally the intensities of the angiography images are defined by this equation:

$$I = I_0 \cdot e^{-\int \mu(x)dx} \tag{3.1}$$

For getting e.g. the attenuation coefficient μ of the integral (3.1) the logarithm has to be applied. So we get our adjusted images intensities for the further imaging processing by applying the logarithm and multiplying -1 to the equation (3.1) :

$$-lnI = \int \mu(x)dx - lnI_0 \tag{3.2}$$

Before adjusting the image intensities the coronary arteries were dark on light background, now they are light on a dark background with values in the range of $[0, 1]$.

3.1.3 Thresholding

The vessels in the images are now light on dark backgrounds (as described above). Nevertheless in the dark areas of the images are local high frequencies where the segmentation algorithms ([KGSD95],[FNVV98]) would find edges for tubular structures (e.g. the vessels of the lungs or midriff - Figure 3.1). To get rid of these areas thresholding is applied. The thresh-

Figure 3.1: Vessels of the lung would be segmented without thresholding

old value is automatically gathered of the variance and the mean value over all intensities (3.3):

$$threshold = meanvalue + percent * variance \tag{3.3}$$

The $percentage$ value is 0.8 and the outside value[2] is the $threshold$ itself, not 0 or 1, because the segmentation algorithms would segment these high frequencies as edges around the borders of the areas.

3.2 Vessel Detection Algorithm

Most vessel segmentation techniques [KQ04]-[HEM+99] are based on their linear structure and employ an eigenvalue analysis of the image intensities. Furthermore, the data will be analyzed at different scales because vessels have a varying diameter and get smaller as the degree of branching increases. Therefore, all methods have the computation of the Hessian with several derivatives of Gauss kernels on several scale steps and the subsequent eigenvalue analysis on every pixel in common. This chapter describes three different vessel detection algorithm which have been implemented and tested. The level set algorithm of Nain [NYT04] and the 2 Gaussian derivative algorithms of Frangi [FNVV98] and Koller [KGSD95]. However the Nain algorithm was discontinued in implementing and segmenting because of unresolved questions.

3.2.1 Level Set Algorithm

The level set segmentation tracks the evolution of a contour (embedded as zero level set of the higher dimensional function $\Psi(X,t)$ under the control of the differential equation:

$$\frac{d}{dt}\Psi = -\alpha A(x) \cdot \nabla\Psi - \beta P(x) \mid \nabla\Psi \mid + \gamma Z(x)\kappa \mid \nabla\Psi \mid \tag{3.4}$$

where A is the advection term, P is the propagation (expansion) term, and Z is the spatial modifier term for the mean curvature κ. The scalar constants α, β, and γ weight the relative influence of each of the terms on the movement of the interface [**?**]. At any time t the zero level set can be extracted $\Gamma(X,t) = \{\Psi(X,t) = 0\}$. Levels sets are useful functions for segmentation by using image-based features such as mean intensity, gradient and edges [Set96].

[2] the value assigned for the areas with pixel intensity $> threshold$

Nain

The Nain segmentation algorithm uses the propagation and mean curvature term of the the level set equation. The propagation term consists of the measures ϵ_1 (Fig. 3.5), ϵ_2 (Fig. 3.6) and parameter a, which influences the deviation (Fig. 3.7). ϵ_1 measures the local ball filter of the percentage of points in the disk $B(x,r)$ inside and on the contour , ϵ_2 is the output of a local ball filter, where the contour points close to the widening of the contour have a higher ϵ_2 measure than contour points on the tube. ϵ_2 looks if the contour point lies near a leak region because points inside the widening region have a higher ϵ_1 measure but most points on the contour have the same measure. [NYT04]

$$\epsilon_1(x) = \int_{B(x,r)} X(y)dy \quad where \quad X(y) = \begin{cases} 1 & \text{if } y \in R \text{ (inside and on the contour)} \\ 0 & \text{if } y \notin R \text{ (outside of the vessel)} \end{cases} \tag{3.5}$$

$$\epsilon_2(x,p) = \epsilon_1^p(x) + p \int_{B(x,r)} \epsilon_1^{p-1}(y)X(y)dy \geq 0 \tag{3.6}$$

The level set equation (3.4) of Nain is derived (from the first derivation of the Energy Function $E(C) = \int_R \phi dx + \int_C ds$) by setting the constant $\alpha = 0$ in the advection term, $\beta = 1$ in the propagation term and $(Z(x) \cdot \gamma) = 1$ in the mean curvature term of the level set equation 3.4. The propagation term $P(x) = -\Phi + a \cdot \epsilon_2$ consists of the erosion term ($a \cdot \epsilon_2$, flow along the inward normal) and the image term Φ:

$$\begin{aligned} \frac{d}{dt}\Psi &= -0 \cdot A(x)\nabla\Psi - \beta P(x) \mid \nabla\Psi \mid +1 \cdot \kappa \mid \nabla\Psi \mid \\ &= (-P(x) + \kappa) \cdot \mid \nabla\Psi \mid \\ &= (\Phi - a \cdot \epsilon_2 + \kappa) \cdot \mid \nabla\Psi \mid \\ &= \frac{\partial C(x)}{dt} \end{aligned} \tag{3.7}$$

The values of the constants are a = 0.65, p = 2 and Φ = 1 for the Nain equation. Due to unresolved issues of evolving the curve for every iteration step (the propagation didn't stop) and in the implementation with [ITK] (to reach the term $P(x)$ some sources of the library had to be changed) the Nain paper was not further analyzed.

3.2.2 Gaussian Derivative Algorithm - 'Frangi'

The Frangi vessel filter [FNVV98] is a eigenvalue analysis vessel detection filter of the image intensities which uses the Gaussian derivatives and computes for every image point x the Hessian matrix (Fig. 3.8) for different scales. The eigenvalues $(\lambda_1 \leq \lambda_2 \leq \lambda_3)$ of the Hessian matrix are sorted ascending and the relations are analyzed for detection of different structures by comparing them, because this gives picture of the gray value variation along the eigenvectors.

$$H(x) = \begin{pmatrix} g_{xx}(x) & g_{xy}(x) & g_{xz}(x) \\ g_{yx}(x) & g_{yy}(x) & g_{yz}(x) \\ g_{zx}(x) & g_{zy}(x) & g_{zz}(x) \end{pmatrix} \tag{3.8}$$

The vesselness function 3.9 detects tubular like structures in regions with high contrast on dark background. The first two terms of the second case detect tube-like structures whereas the third term includes the gray value variation.

$$V(x) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \text{ or } \lambda_3 > 0 \\ \left(1 - e^{-\frac{\lambda_2^2}{2\lambda_3^2\alpha^2}}\right)\left(e^{-\frac{\lambda_1^2}{2|\lambda_2\lambda_3|\beta^2}}\right)\left(1 - e^{-\frac{\lambda_1^2+\lambda_2^2+\lambda_3^2}{2c^2}}\right) & \text{else} \end{cases} \tag{3.9}$$

The constants α, β are fixed to 0.5 and c depends on the grey-scale range of the image and is set to half of the value of the maximum Hessian norm [FNVV98].

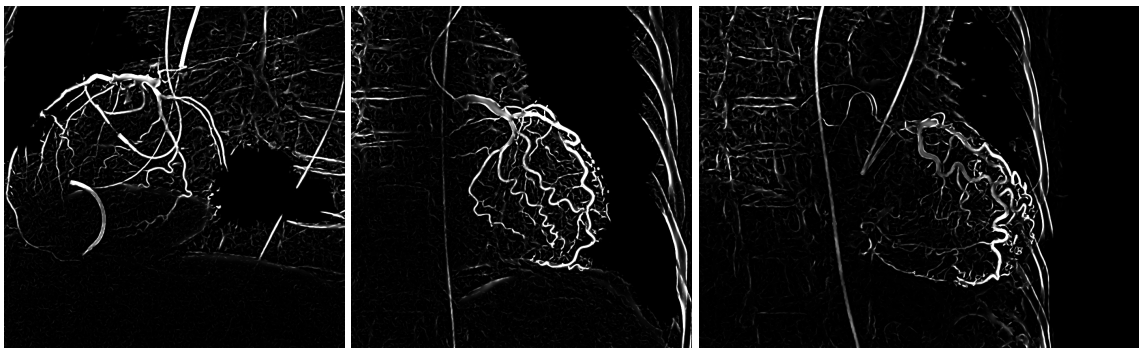


Figure 3.2: Computed Frangi Images (From left to right: Data sets Coburg 070822E, 071011C, 070822B)

Code

The code is implemented according to Frangi [FNVV98] and produces 3 output images (Frangi, best scales and perpendicular directions/eigenvector image) which will be needed

for the segmentation (chapter 3.3)

3.2.3 Gaussian Derivative Algorithm - Multiscale detection of curvilinear structures in 2-D - 'Koller'

The 'Multiscale detection of curvilinear structures in 2-D'-Algorithm [KGSD95] is a nonlinear gaussian derivative algorithm which uses different scales to detect the best possible vessel edge on the left or right depending on the local center pixel, saves the minimum of these two and selects the maximum over all scales in the end.

Introduction

According to Canny [Can83] the second derivative of the Gaussian function ($G_\sigma(x) = e^{\frac{-x^2}{2\sigma^2}}$)

$$F_L = -c_\sigma G''_\sigma = -c_\sigma(\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2})e^{\frac{-x^2}{2\sigma^2}} \tag{3.10}$$

locates the maximum of the convolution of the line profile (c_σ a normalization function depending on σ). According to Koller [KGSD95] the convolution with a bar of width w gives the sum of two edge responses

$$R_{L_\sigma} = c_\sigma(G'_\sigma(x + \frac{w}{2}) - G'_\sigma(x - \frac{w}{2})) \tag{3.11}$$

and has the maximum at $x = 0$. So when we rewrite the second derivative

$$F_L = -a\sigma\frac{G'_\sigma(x + h) - G'_\sigma(x - h)}{2h} + O(h^2) \tag{3.12}$$

to a discrete function with $h = \sigma = \frac{w}{2} = s$ (Scale)

$$F_L \approx \frac{a}{2}(-G'_\sigma(x + s) + G'_\sigma(x - s)) \tag{3.13}$$

($G'_\sigma(x)$ is a known edge detector after Canny [Can86]). It gives the edge detectors $E_l(x) = -G'_\sigma(x + s)$, $E_r(x) = G'_\sigma(x - s)$ where only the positive parts of the detectors are taken $Pos(x) = x \cdot \Theta(x)$ $(= x, x > 0)$. To overcome the multiple line response and the sensitivity to edges, the convolution must be combined in a nonlinear way [KGSD95] where the minimum function $F(R_l, R_r) = min(R_l, R_r)$ was taken as a possible solution. At a location x for a profile $f(x)$ the line filter in 1D is defined as follows:

$$R_s(x) = min(Pos((E_l \otimes f)(x)), Pos((E_r \otimes f)(x))) \tag{3.14}$$

and the 2D multiscale line filter is defined with the function $f(\vec{x})$ ($\vec{x} = (x_1, x_2)$), a direction α with $\vec{d_0} = (\cos\alpha, \sin\alpha)$ and a 2D rotation matrix R_α as:

$$R_s(\vec{x}) = min(Pos((E_l \otimes f)(\vec{x})), Pos((E_r \otimes f)(\vec{x}))) \tag{3.15}$$

$$E_l(x) = R_\alpha(G_s^{'}(x_1 + s) \cdot G_s(x_2))R_\alpha^T \tag{3.16}$$

$$E_r(x) = R_\alpha(G_s^{'}(x_1 - s) \cdot G_s(x_2))R_\alpha^T \tag{3.17}$$

Direction

To find the direction α with $\vec{d_0} = (\cos\alpha, \sin\alpha)$ perpendicular to the vessel $\vec{d_l}$ the method (3.18) given by Koller [KGSD95] is preferred to find where the direction of the second derivative of the function $f_s(\vec{x}) = (f \otimes G_s)(\vec{x})$ is maximal.

$$\arctan(2\alpha) = 2\frac{\partial^2 f}{\partial x_1 \partial x_2} / (\frac{\partial^2 f}{\partial x_1^2} - \frac{\partial^2 f}{\partial x_2^2}) \tag{3.18}$$

Two possible direction solutions in $\{0, \pi\}$ are computed and compared (Figure A.1 in the appendix) to get the maximum $\vec{d_0}$ which is the max. eigenvector perpendicular to the vessel direction. The values computed with this method were compared to the max. eigenvector values computed of the 'itkEigenAnalysisImageFilter' from the ITK-Library [ITK] and delivered the same results.

Properties

- Suppression of step edge response

  One of the edge detectors (left, right) is negative over all scales s and x depending on the slope of the edge. Combined they are 0.

- Multi scale response

  The Filter selects the maximum over all scales (at $x = 0$).

- Optimal scale

  The optimal scale at the maximum $x = 0$ on a line with width w is $s_{opt} = 0.83356\frac{w}{2}$.

- Minimum of scale at $x = O$

  The local minimum is at the center $x = 0$.

- Different line profiles

  Detecting different line profiles (e.g. roof) but always with sharp peak.

- Detection of noisy profiles

  The multiscale response delivers a maximum near the center of the bar.

Code

The code is exact implemented according to Koller [KGSD95] and produces 3 output images (Koller, best scales and perpendicular directions/eigenvector image) which will be needed for the segmenting (chapter 3.3):

- For all scales s
  - Calculate $f_s = f \otimes G_s$
  - Calculate the gradient ∇f_s
  - Calculate the gradient $\vec{x}$
    - Calculate the direction $\vec{d_0}$
    - Calculate the edge responses as:
      $R_l = D_\alpha f_s(\vec{x} + s\vec{d}) = \nabla f_s(\vec{x} + s\vec{d}) \cdot \vec{d}$
      $R_r = -D_\alpha f_s(\vec{x} - s\vec{d}) = -\nabla f_s(\vec{x} + s\vec{d}) \cdot \vec{d}$
    - Calculate the response as:
      $min(Pos(R_l), Pos(R_r))$
    - take only the maximum (≥ 0) of the min-values for every point

See the Code in the appendix A.1.

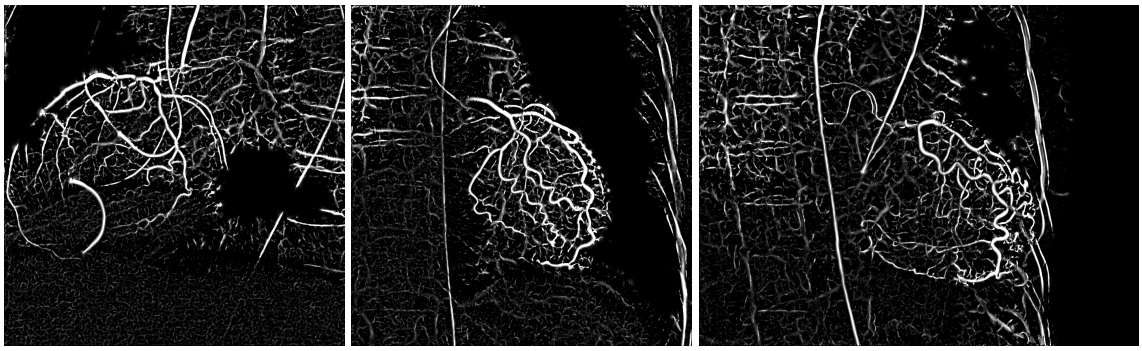


Figure 3.3: Computed Koller Images

3.3 Segmentation

Several well-known vessel segmentation/enhancement methods from the literature have been implemented and tested. In this chapter the extraction of the desired regions (coronary arteries) is done after the computed vessel detection algorithms (with the 3 produced output images). The segmentation of the Frangi, Koller images is performed with simple thresholding or for the Koller images only by extracting the centerlines (non maximum suppression) and using the information for segmentation.

3.3.1 Centerline Detection

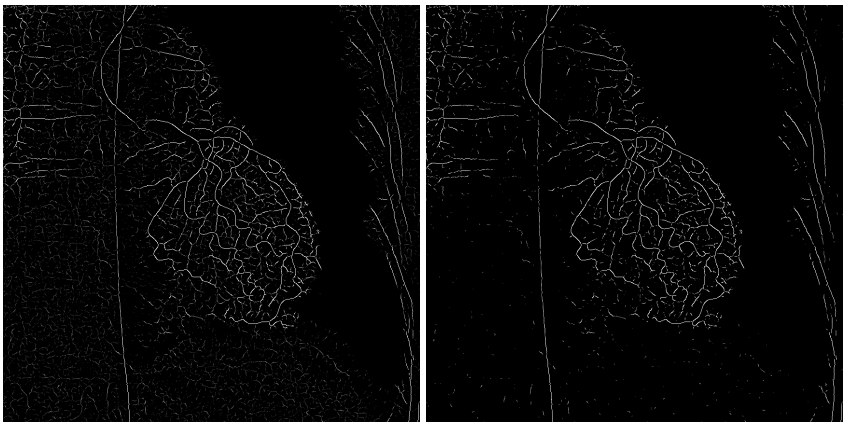
Before extracting the centerline image a thresholding with a low threshold value (a value of 0.18 percent of the maximum pixel value of the Koller image turned out to be useful for all tested slices) is performed to cut the noise, which would be in the next step extracted as centerlines, then the centerline detection algorithm and after this a double thresholding for reducing the rest of all noisy elements. The last operation will be the connected line detection [BMV+06].

Non Maximum Suppression

The non maximum suppression is the centerline detection algorithm for the Koller images. The idea is based on the simple fact that the produced Koller-Images have the highest value in the middle of the tube like structures (e.g. vessels).

The algortihm uses the best scale and the eigenvector $\vec{d_0}$ image (produced in the Koller algortihm process) which have saved on the same pixel position of every pixel of the Koller

image the corresponding needed data. For every Pixel which is the actual local center pixel the local maximum is looked for by traveling along the $\vec{d_0}$ vector left and right. The radius is the best scale of every pixel. If the actual local center pixel has the maximum value it will be saved in the centerline image (Figure 3.4). As described above a thresholding before (Figure 3.4) is processed to cut the noisy tube like structures which can't be vessels which are a side effect through the Koller algorithm principles but other tube like structure with high values (e.g. bones) can't be cropped.



(a) without thresholding (b) with thresholding

Figure 3.4: 'Non Maximum Suppression'

Double Threshold

The 'Non Maximum Suppression' step produces centerlines which are probably not vessels or very small ones. Therefore centerline image post processing with double thresholding and connected line detection (below) is processed to get rid of them (after the way described in [BMV+06]) and because only the thicker coronary arteries are the main interest. For the double thresholding the narrow value 0.1 and wide value 0.9 is used (Figure 3.5).

Connected Line Detection

The last step before segmenting and after the double thresholding is the connected line detection. It extracts the tube like structures with a minimum length to assure to get 'real' vessels because vessels wouldn't be dots ([BMV+06]). The pixels connected through corners are also considered. The minimum connected line length of 5 pixels has been chosen (Figure 3.6).

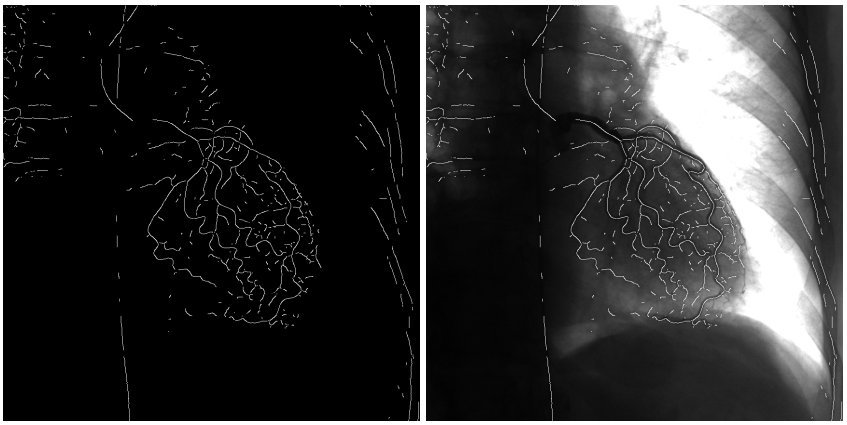


Figure 3.5: 'Double Thresholding' after 'Non Maximum Suppression'

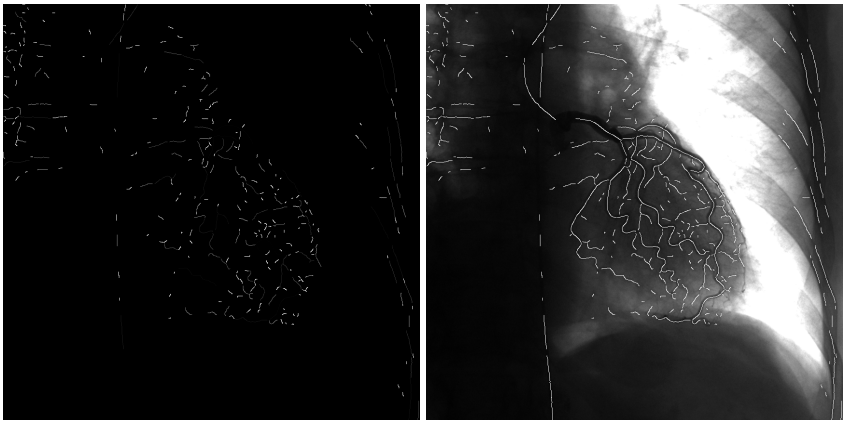
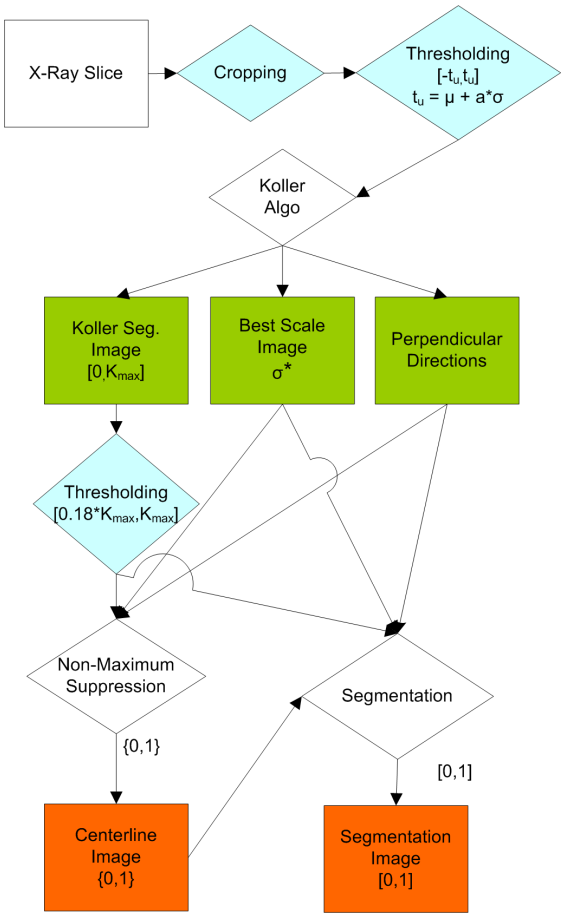


Figure 3.6: 'Connected Line Detection' after 'Double Thresholding'

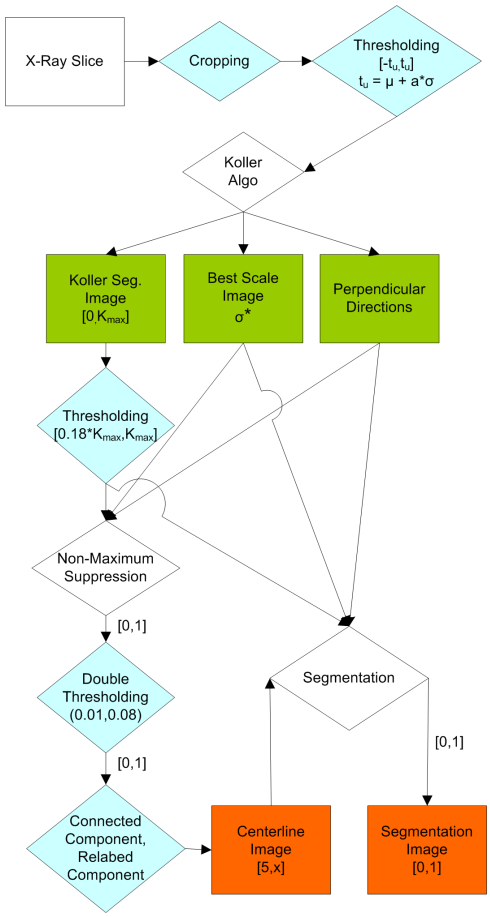
3.3.2 Segmentation

The segmenting is done with the help of the previous generated images (centerline or post processed centerline, best scale, perpendicular directions, Koller or Frangi image) or only with thresholding (single/double) to see the useful or useless differences against the segmenting algorithm A.2.

The Figure 3.7 shows the whole segmenting process for the Koller images with the connections to the previous needed algorithms and needed input data to generate the desired results. There are two different approaches, the first is done with only the help of the 'Non Maximum Suppression' image (centerline image) after the Koller algorithm. The second uses the post processed 'Non Maximum Suppression' image as described above.



(a) Segmenting with only the 'Non Max. Suppression' image



(b) Segmenting with the post processed centeline image

Figure 3.7: The Koller segmenting process with different centerline images

Single/Double Threshold Segmentation

The single/double thresholding images are generated direct from the Koller/Frangi images. The single threshold value is a manually selected minimum value of one edge and the double threshold values are about 0.90 percent (wide) and 0.99 (narrow) of the maximum image values (Figure 3.8).

Segmenting with centerline image

The Figure A.2 (appendix) shows the C++ 'Pseudo-Code' how the segmenting is done with the help of the other previously generated images. The Frangi-Image segmentation doesn't use the centerline image like the Koller-Image (the commented text in the inner

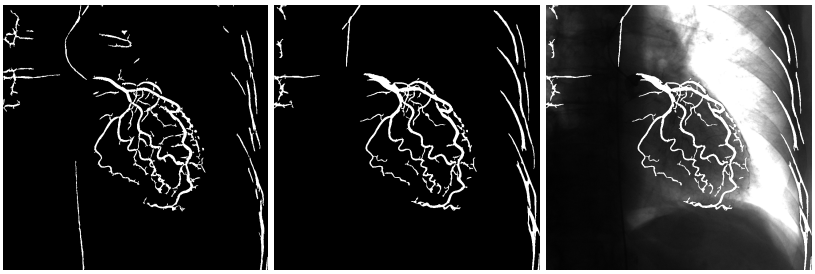
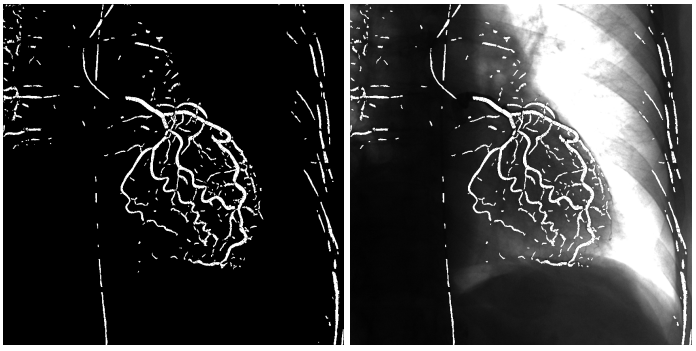


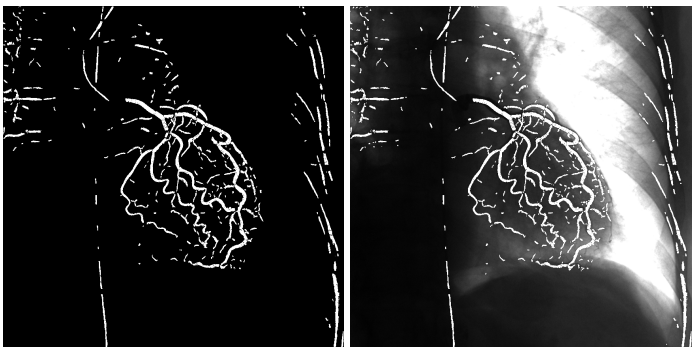
Figure 3.8: Double Threshold Koller (left), Frangi (middle) and Frangi added on the original image (right) images

loop). For every pixel, which is a centerline pixel, the corresponding best scale is traveled along 'left' and 'right' according to the perpendicular directions and the actual pixel gets the value divided with the local maximum value of the perpendicular line (Figure 3.9). The final thresholding yields the pixels with a high probability which are inside and on the vessels.



(a) Segmenting with the 'Non max. Suppr.' centerline image

(b) Segmenting with the 'Non max. Suppr.' centerline image added on the original image



(c) Segmenting with the connected component centerline image

(d) Segmenting with the connected component centerline image added on the original image

Figure 3.9: Center line segmentation

Chapter 4

Evaluation

The evaluation of the results are quantitatively compared with each other and mainly with manually segmented data (numerical comparison). Furthermore, the accuracy improvement which is possible by specializing these algorithms to the problem at hand have been investigated.

4.1 Results Post processed: Koller vs. Frangi

4.1.1 Double Threshold Comparison

The double threshold Frangi segmentation image compared against the double threshold Koller segmentation image, each created after their computation algorithm (Figure 4.1).

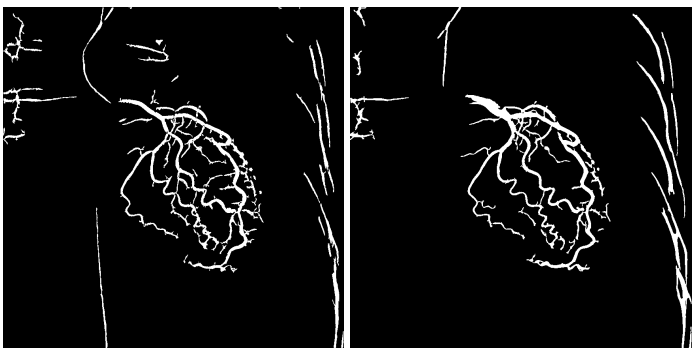


Figure 4.1: The double threshold Koller (left) and Frangi segmentation images (right)

4.1.2 Post Processed Centerline Koller vs. Double Threshold Comparison Frangi

The double threshold Frangi segmentation image compared against the post processed Koller segmentation image with the the centerline image and the steps described in Blondel (Figure 4.2).

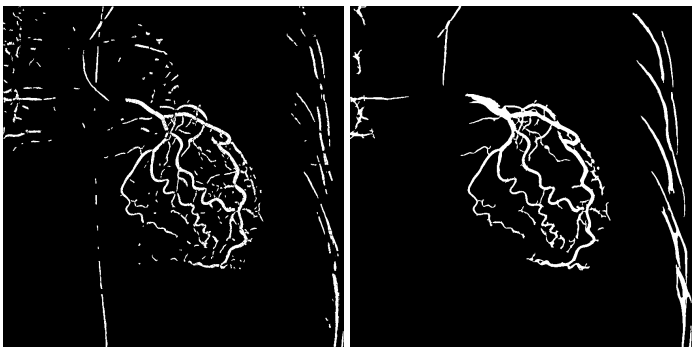


Figure 4.2: The post processed Koller segmentation image (left) and double threshold Frangi segmentation (right) image

4.2 Manually Segmented

The manually segmented images (Figure 4.3) were made by hand in an image editor as good as possible. They will be the reference images against the computed ones, the 'ground truth'. Before comparing the manually segmented images with the computed Koller and Frangi images, we used post processing algorithms on all comparing images to get objective results in the end. The two filters which have been used to get the elements of interest (looked for vessels length $(2, 20)$) are the 'WhiteTopHatImageFilter' with the radius greater than our selected one > 20 (max. scale) and for the smaller elements the binary morphological filter ('BinaryMorphologicalImageFilter') with the radius < 2 (min. scale) both from the ITK-Library ([ITK]).

The numerical comparison is done by comparing each pixel of the Koller/Frangi segmented images with the manually segmented images if there is a pixel on the same index locations or not, by counting them all and divide the results through the number of ground truth pixels $\mid \widetilde{O}_{Ground} \mid$ (Figure 4.4). So if the Koller or Frangi image has set the pixels where the manually segmented image has its, then it would be true otherwise false. For false are two possibilities, 'false negativ' (FN), if the Koller/Frangi segmented image has set no pixels

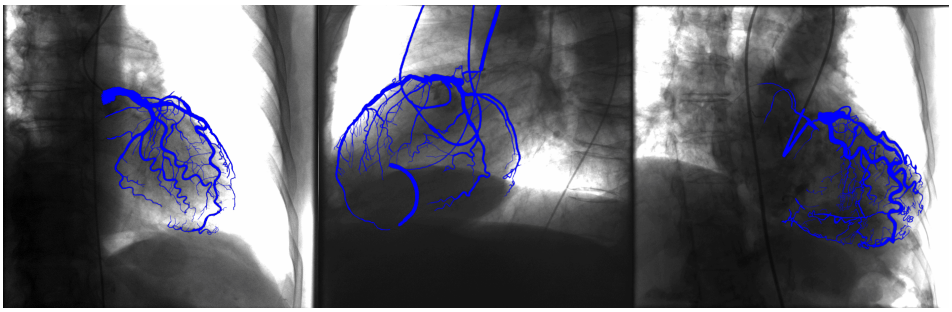


Figure 4.3: Coronary Arteries Hand Segmented

on positions as the manually one has:

$$\mid FN \mid = \mid \widetilde{O}_{Ground} \setminus \widetilde{O}_{Koller} \mid \tag{4.1}$$

or 'false positive' (FP), if the Koller/Frangi segmented image has set pixels on positions which have not been set on the manually image:

$$\mid FP \mid = \mid \widetilde{O}_{Koller} \setminus \widetilde{O}_{Ground} \mid \tag{4.2}$$

The 'true positive' result are the pixels shared from both images ($\mid TP \mid = \mid \widetilde{O}_{Ground} \mid - \mid FN \mid$). At last the equations must be divided to get the final probability results, the main interest:

$$R_{FP} = \frac{\mid FP \mid}{\mid \widetilde{O}_{Ground} \mid} \quad , \quad R_{FN} = \frac{\mid FN \mid}{\mid \widetilde{O}_{Ground} \mid} \tag{4.3}$$

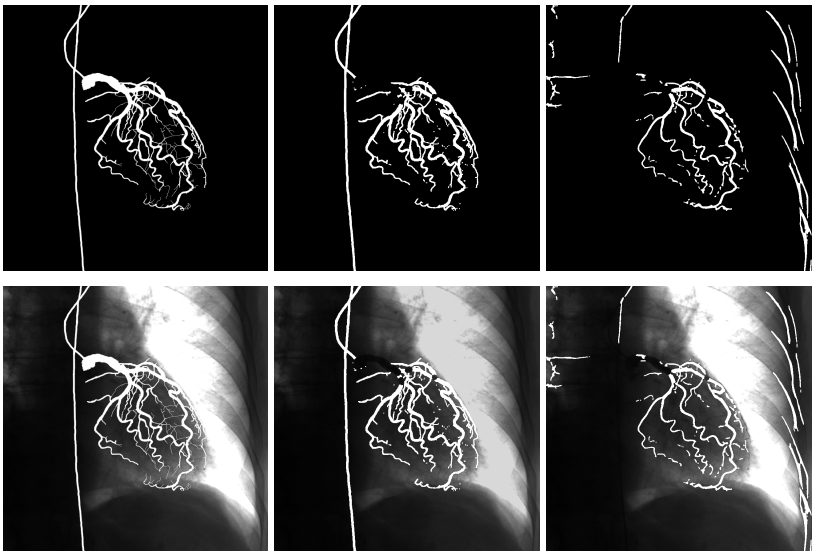
$$R_{TP} = 1 - R_{FN} \tag{4.4}$$

4.2.1 Results Frangi

The numerical values for the 3 images from the different datasets of the Frangi-L8-Segementation (double threshold segmentation after generating the Frangi image) and the Manual-L11-Segmentation comparison images are shown in the table below (Fig. 4.4). The average of same pixels is between $45 - 50\%$, the under segmentation of $50 - 55\%$ average and the over segmentation differs from $17 - 40\%$ (Fig. 4.5]).

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_F \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	34282	40062	16255	22035	18027	0.55	0.41	0.45
COBURG-070822E	33202	50634	8701	26133	24501	0.52	0.17	0.48
COBURG-070822B	43469	57323	17019	30873	26450	0.54	0.30	0.46

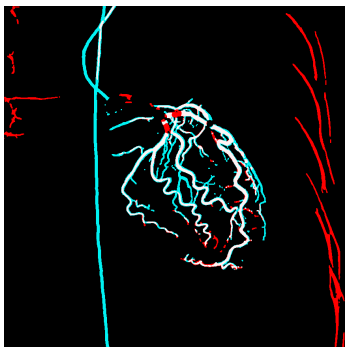
Figure 4.4: Numerical outputs for the different tested slices



(a) original manually segmented images

(b) adjusted manually segmented images

(c) computed Frangi images



(d) Compared Frangi (red) and manually seg. images (blue), overlay pixels are white segmented

Figure 4.5: Data set Coburg 071011C

4.2.2 Results Koller

The numerical values for the 3 images from the different data sets of the Koller-L6-Segmentation (only double threshold, after the generation of the Koller image), Koller-L11-

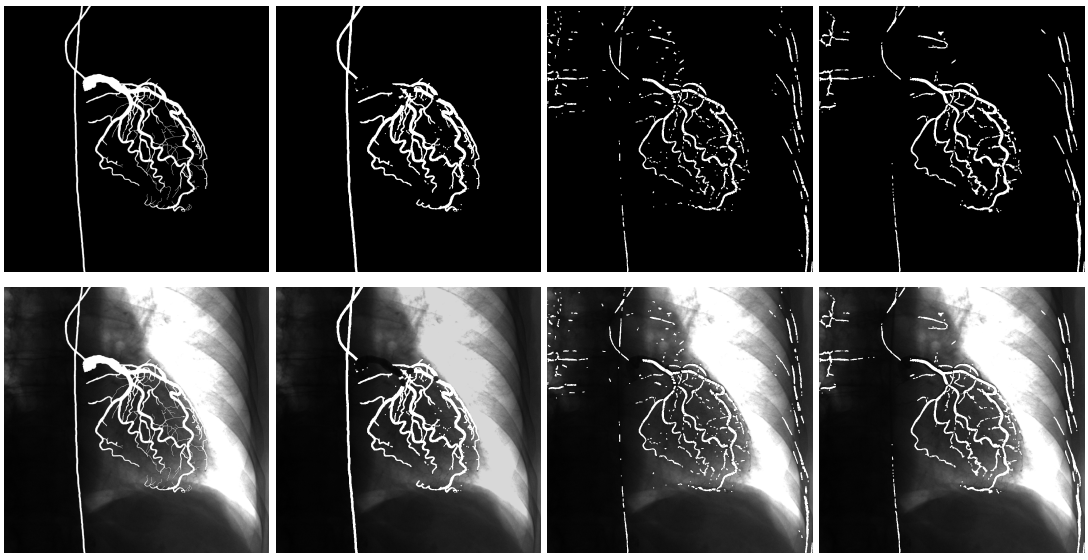
Segmentation (post processed image with min. connected lines of 5 pixels, see Blondel) compared against the Manual-L11-Segmentation comparison images are shown in the tables below (Fig. 4.6- 4.7). Clearly visible is an average of 40% of identical pixels in the Koller-L11-Segmentation image comparision but there is a high average percentage of 60% of under segmentation. The over segmentation average is under 40%. The reference Koller-L6-Segmentation image has an average hit of 45% in the comparision but the over segmentation varies more and the under segmentation is nearly the same on both Koller comparision images (Fig. 4.8-**??**).

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	31416	40062	14274	22920	17142	0.57	0.36	0.43
COBURG-070822E	32023	50634	11034	29645	20989	0.59	0.22	0.41
COBURG-070822B	44585	57323	21092	33830	23493	0.60	0.37	0.41

Figure 4.6: Numerical outputs of the 3 data sets of the Koller-L11-Segmentation images compared with the Manual-L11-Segmentation comparison images

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	35588	40062	16937	21411	18651	0.53	0.42	0.47
COBURG-070822E	32696	50634	9910	27848	22786	0.55	0.20	0.45
COBURG-070822B	44585	57323	19180	32190	25133	0.56	0.33	0.44

Figure 4.7: Numerical outputs of the 3 data sets of the Koller-L6-Segmentation images compared with the Manual-L11-Segmentation comparison images



(a) original manually segmented images (b) adjusted manually segmented images (c) computed (min. connected lines of 5 pixels) Koller images (d) computed double threshold Koller images

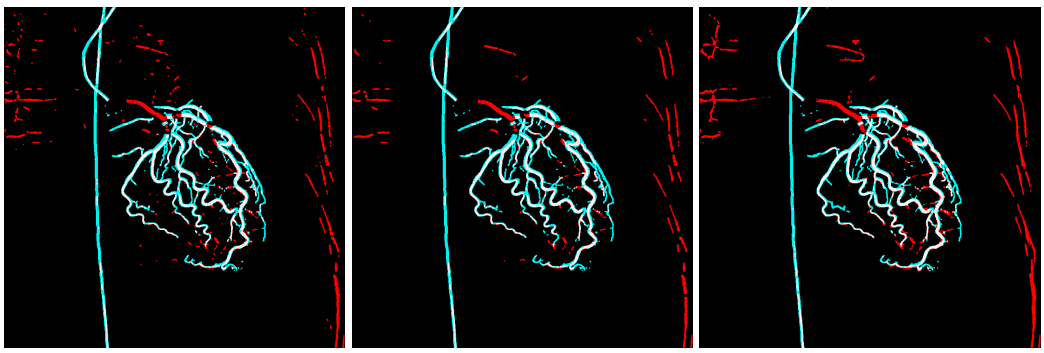
Figure 4.8: Data set 'COBURG-071011C' images

4.2.3 Enhancements

To reduce the over segmentation in the comparison Koller segmentation images ones (Fig. 4.6) the value of the minimum connected line was increased to 20 pixels. The result were false positive percentages under 30% during nearly constant averages of 40% of identical setted pixels but an high under segmentation of nearly 60%(Fig. 4.9-4.10).

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	27006	40062	10892	23948	16114	0.60	0.28	0.40
COBURG-070822E	27469	50634	7971	31136	19498	0.62	0.16	0.39
COBURG-070822B	36290	57323	14796	35829	21494	0.63	0.26	0.37

Figure 4.9: Numerical outputs for the different tested slices



(a) Compared Koller (min. connected lines of 5 pixels) (red) and manually seg. images (blue), overlay pixels are white segmented

(b) Compared Koller (min. connected lines of 20 pixels) (red) and manually seg. images (blue), overlay pixels are white segmented

(c) Compared Koller (only double threshold) (red) and manually seg. images (blue), overlay pixels are white segmented

Figure 4.10: Data set 'COBURG-071011C' images

Chapter 5

Conclusions

The results of the Koller double threshold vs. Koller post processed segmentation image against the man. segmented image has only a better average hit of 4% but the false positives value varies to much. So the enhenced min. connected line of 20 pixels image should be a better choice together with a morphological image filter to fill the black insular pixels inside of the vessels to increase the correct hits.

The different interesting results of the Koller vs. Frangi segmentation images compared against the man. seg. images led to the idea to merge each Koller, Frangi image to one image $x = max(Koller(x), Frangi(x))$ and test them against the manually segmentation images (Fig. 5.1, 5.2). The identical pixels increased positively to an average of over 50% but unfortunately as well the over segmented percentage to max. over 60% (the min. connected line of 20 pixels has a max. 10% better result). On the other hand the under segmentation decreased positively to max. 46%. For reference the last table (Fig. 5.3) shows the added Frangi image with the only double threshold segmentation Koller image. The hits and the under segmentation are here slightly better, too, but the over segmentation varies again too much (Fig. 5.4).

The presented algorithm have both pros and cons. Frangi for example segments lesser noisy connected pixel areas but more intense the bones. Both didn't fully detect the catheter in this slice (please see the appendix for the other slices). Clearly visible is on the generated images right after the algorithm that Frangi detects better the vessel width (unfortunately the 'Tophat' filter erased some of them during generation of the comparison images) but Koller the smaller vessels (which could be noisy pixels as well). The merged images produced therefore better TP/FN but increased the FPs, so optimizing the constants and the post processing algorithms would certainly get better final results. Anyway the examples should give only an good overview for an objective result more manually images would be

necessary.

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	46503	40062	24150	17709	22353	0.44	0.60	0.56
COBURG-070822E	42386	50634	14477	22725	27909	0.45	0.29	0.55
COBURG-070822B	61019	57323	29057	25361	31962	0.44	0.51	0.56

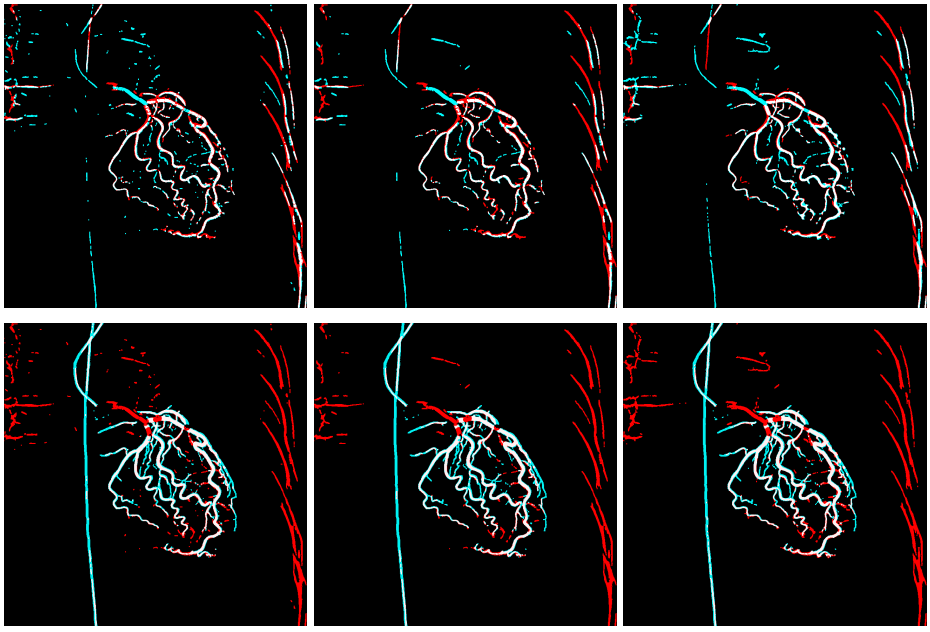
Figure 5.1: The merged Koller (min. 5 connected component pixels), Frangi seg. images compared against the man. seg. images

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	43327	40062	21525	18260	21802	0.46	0.54	0.54
COBURG-070822E	39176	50634	11917	23375	27259	0.46	0.24	0.54
COBURG-070822B	54786	57323	23883	26420	30903	0.46	0.42	0.54

Figure 5.2: The merged Koller (min. 20 connected component pixels), Frangi seg. images compared against the man. seg. images

$\frac{Pixels/Propabilities}{Datasets}$	$\mid \widetilde{O}_K \mid$	$\mid \widetilde{O}_G \mid$	$\mid FP \mid$	$\mid FN \mid$	$\mid TP \mid$	R_{FN}	R_{FP}	R_{TP}
COBURG-071011C	48784	40062	25756	17034	23028	0.43	0.64	0.57
COBURG-070822E	41138	50634	12936	22432	28202	0.44	0.26	0.56
COBURG-070822B	58497	57323	26459	25285	32038	0.44	0.46	0.56

Figure 5.3: The merged Koller (only double threshold), Frangi seg. images compared against the man. seg. images



(a) Min. 5 pixels connected component pixels Koller + Frangi image

(b) Min. 20 pixels connected component pixels Koller + Frangi image

(c) Double treshold Koller + Frangi image

Figure 5.4: Coburg data set 071011C: On the Top the compared merged Frangi (red) and Koller (blue) images (the minimum is white $x = min(Koller(x), Frangi(x))$) and merged Frangi, Koller (red) and manually segmentation images (blue) (below) - overlapped pixels are segmented white

Appendix A

Code

A.1 Koller

The following code shows the C++ Pseudo-Code of the 'Multiscale detection of curvilinear structures in 2-D'-Algorithm implemented in this Project:

```
% IDfxx --- dxx Gy
% IDfxy --- dx Gx * dy Gy
% IDfyy --- dyy Gx
% Ifx --- dx Gy
% Ify --- dy Gx
% IOutput --- Output Image
% IVector --- Image which contains the d0_vector for
%             every pix orthogonal to the vessel
% ISigmaBest --- Image which contains the best sigmas
%                of the max pixel over all scales
for(sigma = sigmas.Begin();sigma <=sigmas.End(); sigma++)
{
    IDfxx.setSigma(sigma);
    IDfxy.setSigma(sigma);
    IDfyy.setSigma(sigma);
    Ifx.setSigma(sigma);
    Ify.setSigma(sigma);
    for(IOutput++,ISigmaBest++,IVector++)
    {
        //tan_1(2* alpha) = 2 * IDfxy / (IDfxx - IDfyy)
        alpha = atan( 2 *IDfxy / (IDfxx-IDfyy)/2);
        alpha90 = atan( 2 *IDfxy / (IDfxx-IDfyy)/2 + PI/2);

        dx0 = cos(alpha);
        dy0 = sin(alpha);
        dx1 = cos(alpha90);
        dy1 = sin(alpha90);

        dHd0=IDfxx*dx0*dx0 +2*IDfxy*dx0*dy0 +IDfyy*dy0*dy0;
        dHd1=IDfxx*dx1*dx1 +2*IDfxy*dx1*dy1 +IDfyy*dy1*dy1;

        if(dHd0 > dHd1)
        {
```

```
            offset[0] = pos + sigma * dx0;
            offset[1] = pos + sigma * dy0;
            Dlf = Ifx(offset) * dx0 + Ify(offset) * dy0;

            offset[0] = pos - sigma * dx0;
            offset[1] = pos - sigma * dy0;
            Drf = -Ifx(offset) * dx0 - Ify(offset) * dy0;
            vector = {dx0 , dy0};
        }
        else
        {
            offset[0] = pos + sigma * dx1;
            offset[1] = pos + sigma * dy1;
            Dlf = Ifx(offset) * dx1 + Ify(offset) * dy1;

            offset[0] = pos - sigma * dx1;
            offset[1] = pos - sigma * dy1;
            Drf = -Ifx(offset) * dx1 - Ify(offset) * dy1;
            vector = {dx1 , dy1};
        }

        vesselness = 0
        if(Dlf > 0 && Drf > 0)
        {
            vesselness = min(Dlf.Get(),Dlf.Get());
        }
        IOutput.Set(max(IOutput.Get(), vesselness));
        ISigmaBest.Set(max(ISigmaBest.Get(), sigma));
        IVector.Set(max(IVector.Get(), vector));
    }
}
```

Figure A.1: Multiscale detection of curvilinear structures in 2-D

A.2 Segmenting with centerline image

The Figure A.2 shows the C++ 'Pseudo-Code' how the segmenting is done with the help of the other previously generated images.

```
% IKoller ---- Image computed with the multiscale
%              detection algo (Koller)
% IVector ---- Image which contains the d0_vector for
%              every pix orthogonal to the vessel
% ISigmaBest ---- Image which contains the best sigmas
%                 of the max pixel over all scales
% ICenterLine ---- computed centerlines of the IKoller
%                  Image with non-maximum supression
% IOutput ---- Output Image
% threshold ---- minimum threshold (> 0) to get the
%                "real" vessels, no noise
for(float pos = 0; pos = IKoller; IKoller++)
{
   p = IKoller.Get(pos);
   d0_vec = IVector.Get(pos);
   sigma = ISigmaBest.Get(pos);
   if( p > threshold )
   {
      m = threshold;
      for(float dist= -sigma;dist<=sigma; dist=dist+0.5)
      {
         c = ICenterLine.Get(pos + dist * d0_vec);
         pNow = IKoller.Get(pos + dist * d0_vec);
         sNow = ISigmaBest.Get(pos + dist * d0_vec);
         %Koller segmenting
         if( c > 0 && pNow >= m)
         {
            m = pNow;
         }
         %Frangi segmenting
         %if(sNow >= dist && pNow >= m)
         %{
         %   m = pNow;
         %}
      }
      IOutput.Set(pos, p / m );
   }
   else
   {
      IOutput.Set(pos, 0);
   }
}
```

Figure A.2: Multiscale detection of curvilinear structures in 2-D

Appendix B

Images

B.1 Hierarchy of the generated Images

B.1.1 Koller

Koller-, scales-, vector-Image	L1-CROPPED L2-THRESHOLD-INVERTED L3-ALGO L3-SCALES L3-VECTOR
Segmentation without centerline image (for all following images, L3 images needed)	L4-SEGMENTATION-DOUBLETHRESHOLD L5-SEGMENTATION-DOUBLETHRESHOLD-TOPHAT L6-SEGMENTATION-DOUBLETHRESHOLD-BINMORPH L7-FINAL-ORIGINAL-SEGMENTATION-DOUBLETHRESHOLD (L5-L7 for comparison with the manually segmented image)
Centerline Images	L4-CENTERLINE-TITUS L5-ORIGINAL-CENTERLINE
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-ORIGINAL-CENTERLINE
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-CENTERLINE-CONNECTEDCOMPONENT L7-CENTERLINE-RELABELEDCOMPONENT L8-CENTERLINE-ALL L9-ORIGINAL-CENTERLINE

Figure B.1: Generated Images and Their Hierarchy

Segmentation images with centerline image	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-SEGMENTATION L7-ORIGINAL-SEGMENTED
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-SEGMENTATION L7-SEGMENTATION-TOPHAT L8-SEGMENTATION-BINMORPH L9-FINAL-ORIGINAL-SEGMENTATION (L7-L9 for comparison with the manually segmented image)
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-CENTERLINE-CONNECTEDCOMPONENT L7-CENTERLINE-RELABELEDCOMPONENT L8-CENTERLINE-ALL L9-SEGMENTATION-ALL
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-CENTERLINE-CONNECTEDCOMPONENT L7-CENTERLINE-RELABELEDCOMPONENT L8-CENTERLINE-ALL L9-SEGMENTATION-ALL L10-ORIGINAL-SEGMENTED
	L4-CENTERLINE-TITUS L5-CENTERLINE-DOUBLETHRESHOLD L6-CENTERLINE-CONNECTEDCOMPONENT L7-CENTERLINE-RELABELEDCOMPONENT L8-CENTERLINE-ALL L9-SEGMENTATION-ALL L10-SEGMENTATION-ALL-TOPHAT L11-SEGMENTATION-ALL-BINMORPH L12-FINAL-ORIGINAL-SEGMENTATION-ALL (L10-L12 for comparison with the manually segmented image)

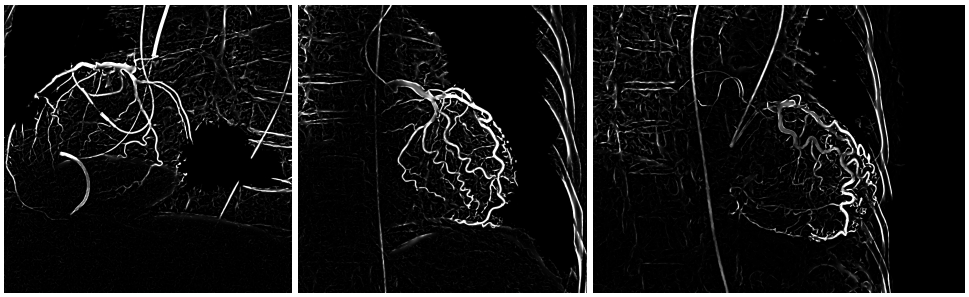
Figure B.2: Generated Images and Their Hierarchy - Part 2

B.1.2 Frangi

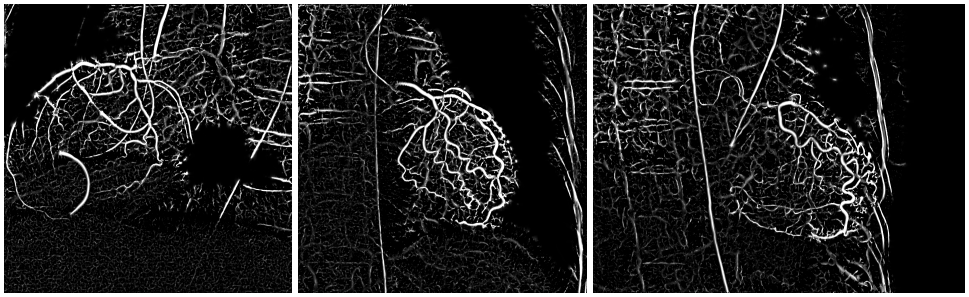
Frangi, scales-, vector-Image	L1-CROPPED L2-THRESHOLD-INVERTED L3-ALGO L3-SCALES L3-VECTOR
Segmentation without centerline image (for all following images, L3 images needed)	L4-SEGMENTATION-DOUBLETHRESHOLD L5-SEGMENTATION-DOUBLETHRESHOLD-TOPHAT L6-SEGMENTATION-DOUBLETHRESHOLD-BINMORPH L7-FINAL-ORIGINAL-SEGMENTATION-DOUBLETHRESHOLD (L5-L7 for comparison with the manually segmented image)
	L4-THRESHOLD L6-SEGMENTATION L7-SEGMENTATION-TOPHAT L8-SEGMENTATION-BINMORPH L9-FINAL-ORIGINAL-SEGMENTATION (L7-L9 for comparison with the manually segmented image)

Figure B.3: Generated Images and Their Hierarchy

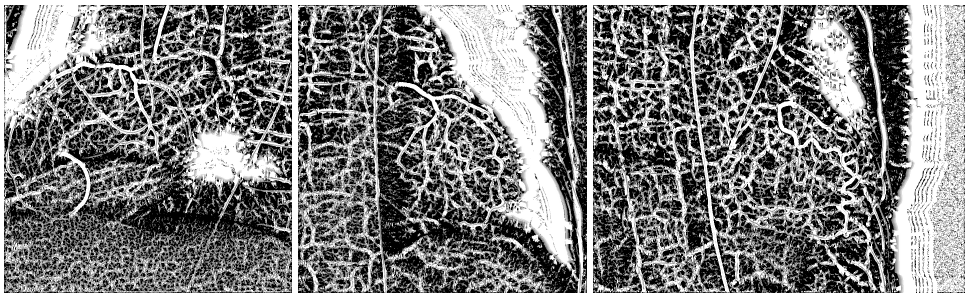
B.2 Computed Frangi and Koller Algorithm Images



(a) Computed Frangi algo images



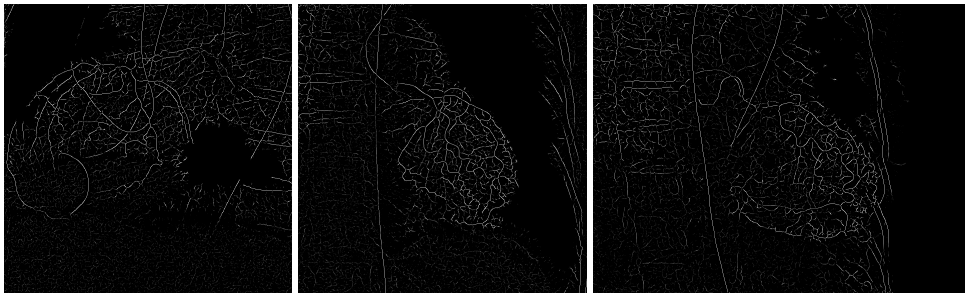
(b) Computed Koller algo images



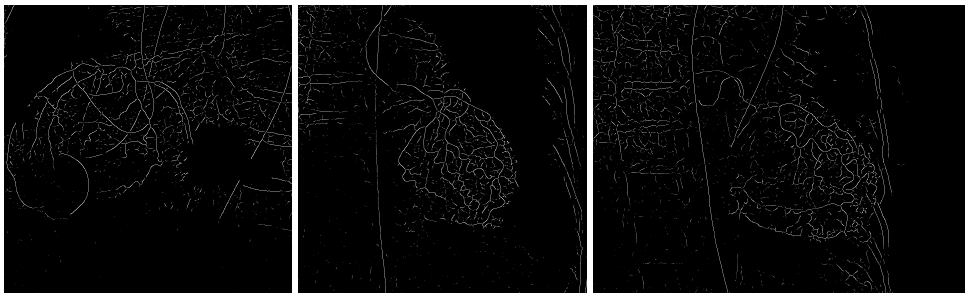
(c) Computed Koller scales images

Figure B.4: Computed Frangi and Koller algorithm images (from left to right: Data set Coburg 070822E, 071011C, 070822B)

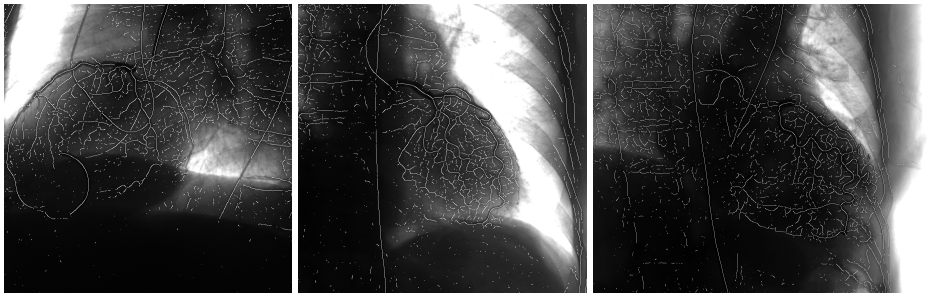
B.3 Centerline Images



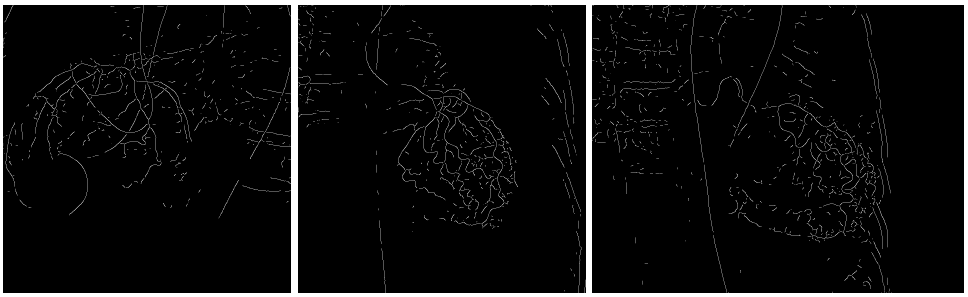
(a) 'Non Maximum Suppression' without thresholding



(b) 'Non Maximum Suppression' with thresholding

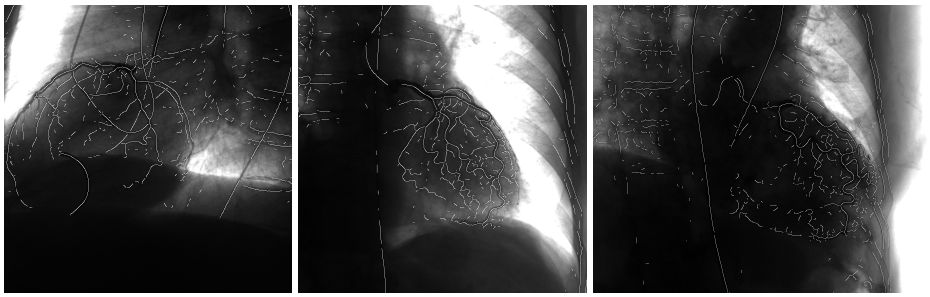


(c) 'Non Maximum Suppression' with thresholding on the original image

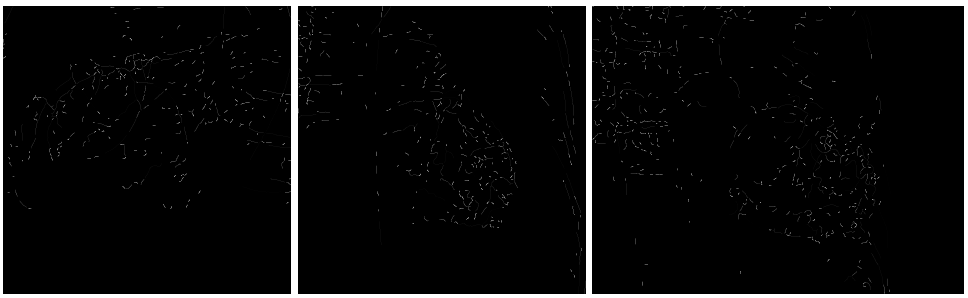


(d) 'Double Thresholding' after 'Non Maximum Suppression'

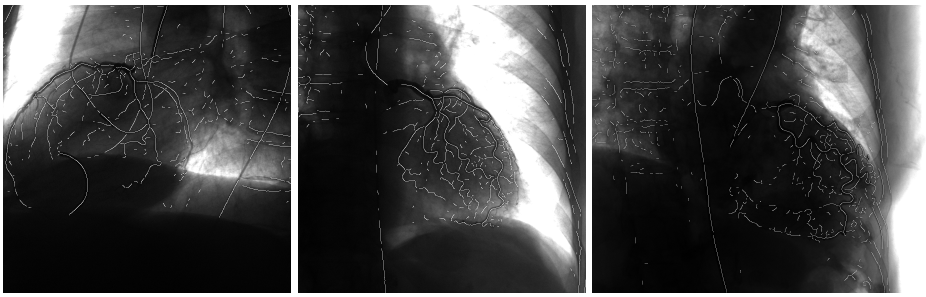
Figure B.5: Centerline Images



(a) 'Double Thresholding' after 'Non Maximum Suppression' added on original images



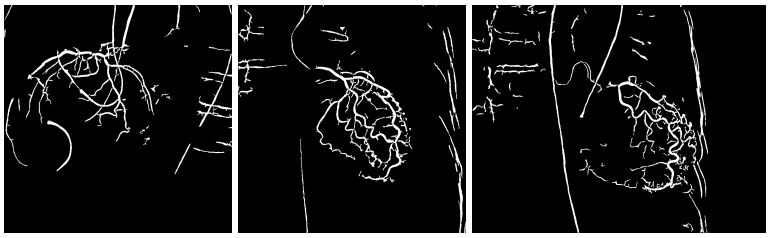
(b) 'Connected Line Detection' after 'Double Thresholding'



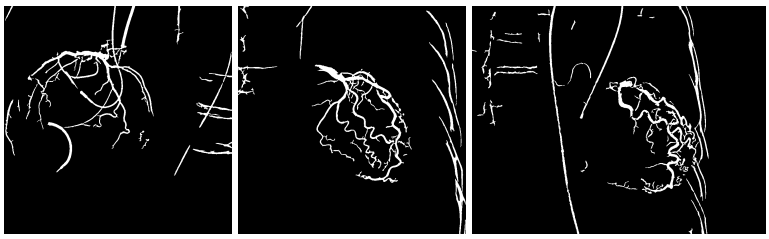
(c) 'Connected Line Detection' after 'Double Thresholding' added on original images

Figure B.6: Centerline Images (from left to right: Data set Coburg 070822E, 071011C, 070822B)

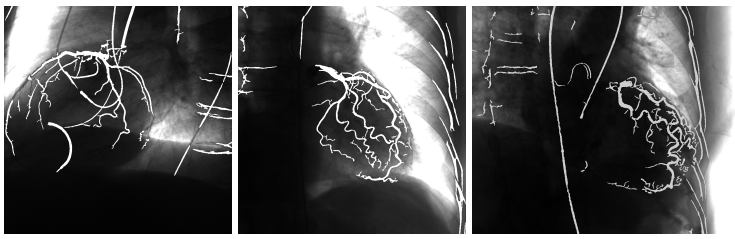
B.4 Segmentation Images



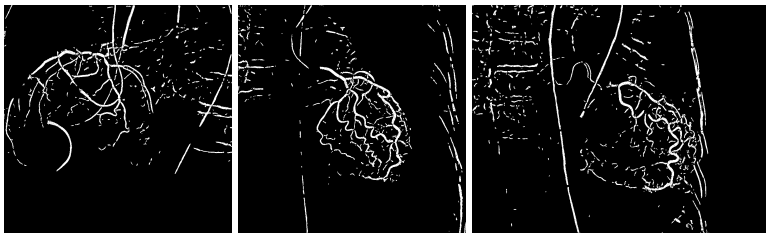
(a) Double Threshold Koller Images



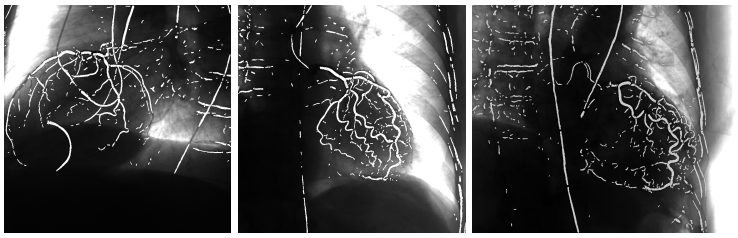
(b) Double Threshold Frangi Images



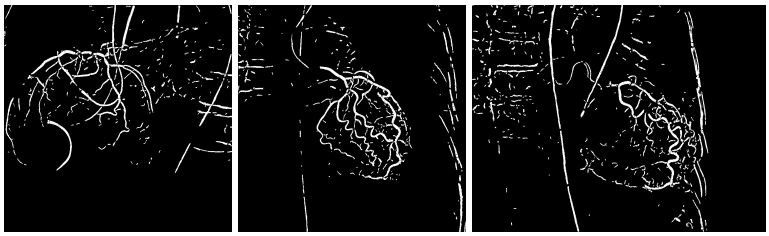
(c) Double threshold segmentation Frangi images added on the original image



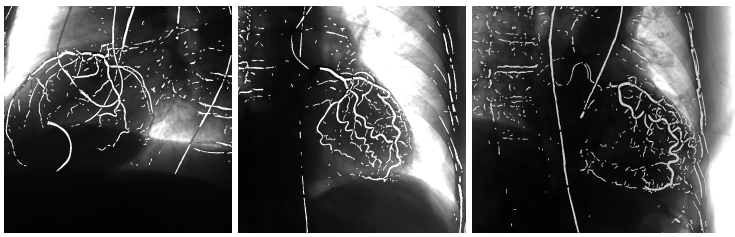
(d) Segmenting with the 'Non max. Suppr.' centerline image



(e) Segmenting with the 'Non max. Suppr.' centerline image added on the original image



(f) Segmenting with the connected component centerline image



(g) Segmenting with the connected component centerline image added on the original image

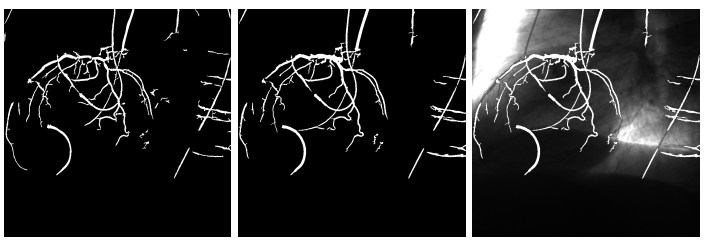
Figure B.7: Double threshold and center line segmentation images (from left to right: Data set Coburg 070822E, 071011C, 070822B)

B.5 Evaluation

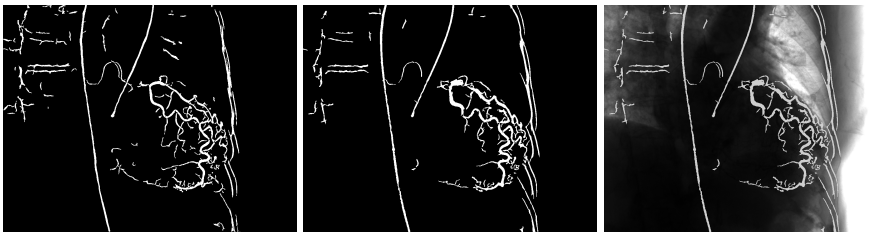
B.5.1 Segmentation results: Frangi vs. Koller

Double threshold comparison

(a) Data set Coburg-071011C



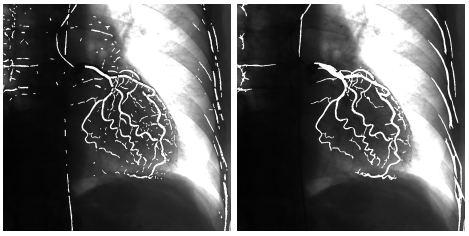
(b) Data set Coburg-070822E



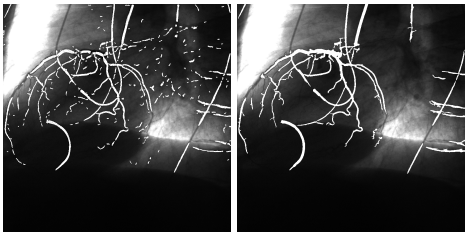
(c) Data set Coburg-070822B

Figure B.8: Double threshold Koller (left), Frangi segmented images (middle) and Frangi segmentation image added on the original image (right)

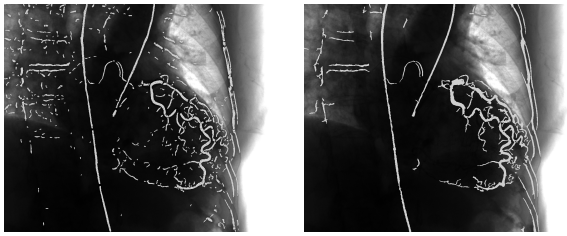
Double threshold Frangi vs. center line Koller segmentation



(a) Data set Coburg-071011C: Koller (left) vs. Frangi (right)



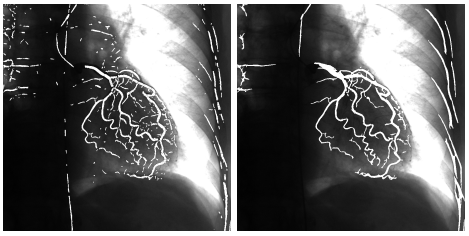
(b) Data set Coburg-070822E: Koller (left) vs. Frangi (right)



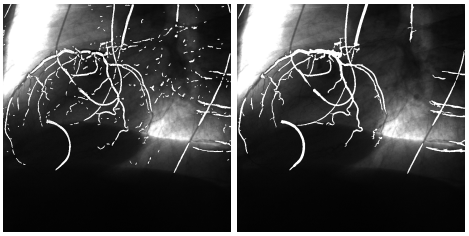
(c) Data set Coburg-070822B: Koller (left) vs. Frangi (right)

Figure B.9: The 'Non max. Suppr.' center line Koller vs. double threshold Frangi segmentation images added on the original image

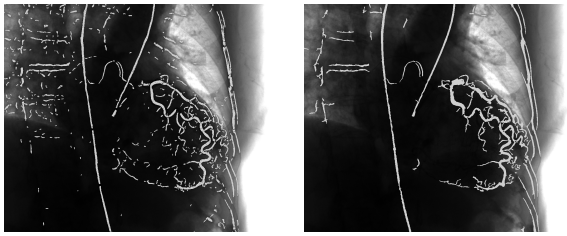
Double threshold Frangi vs. center line post processed Koller segmentation



(a) Data set Coburg-071011C: Koller (left) vs. Frangi (right)



(b) Data set Coburg-070822E: Koller (left) vs. Frangi (right)

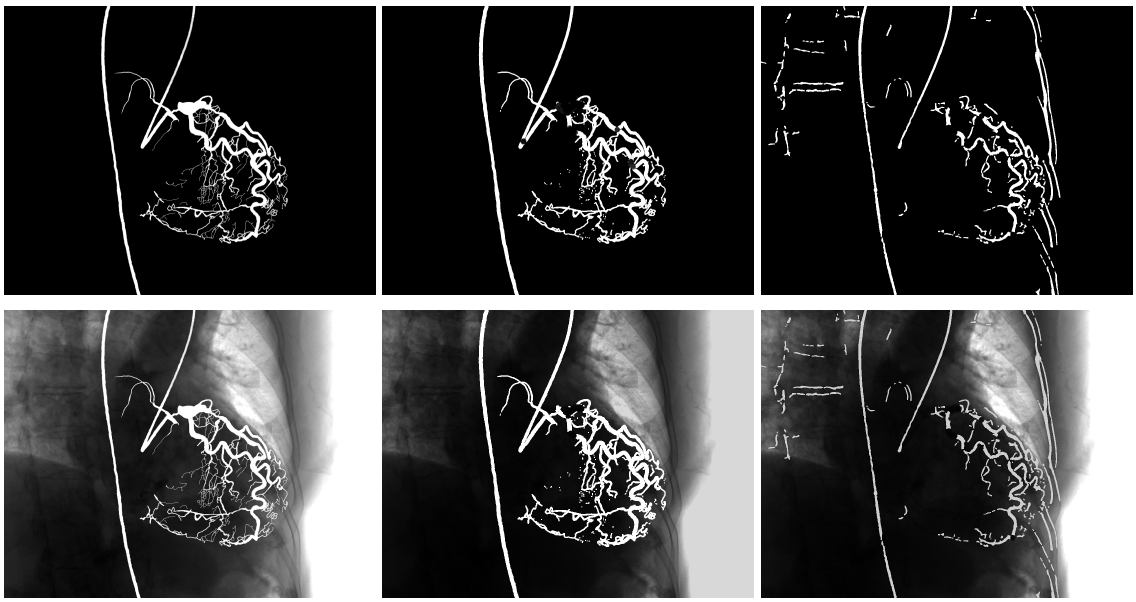


(c) Data set Coburg-070822B: Koller (left) vs. Frangi (right)

Figure B.10: The connected component center line (min. connected lines of 5 pixels) Koller vs. double threshold Frangi segmentation images added on the original image

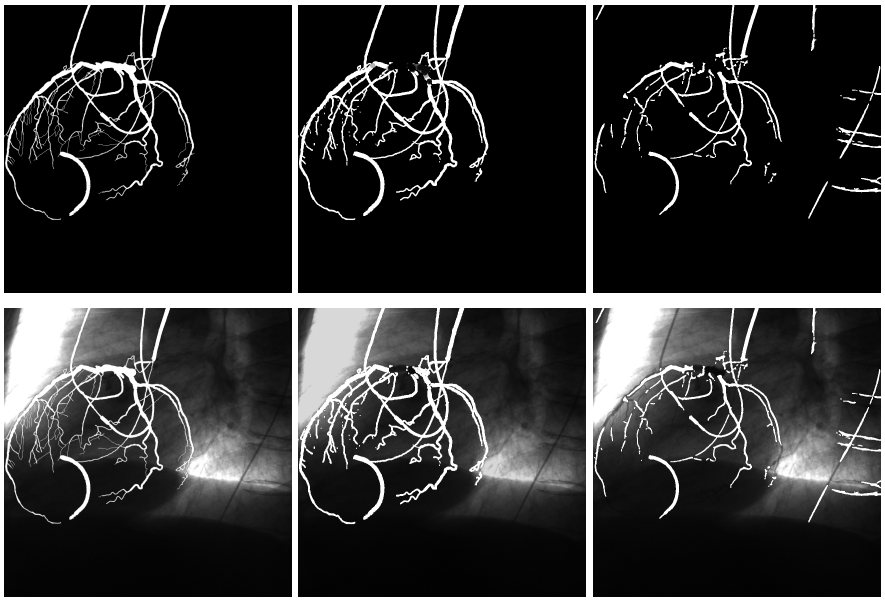
B.5.2 Manually Segmented

Results Frangi



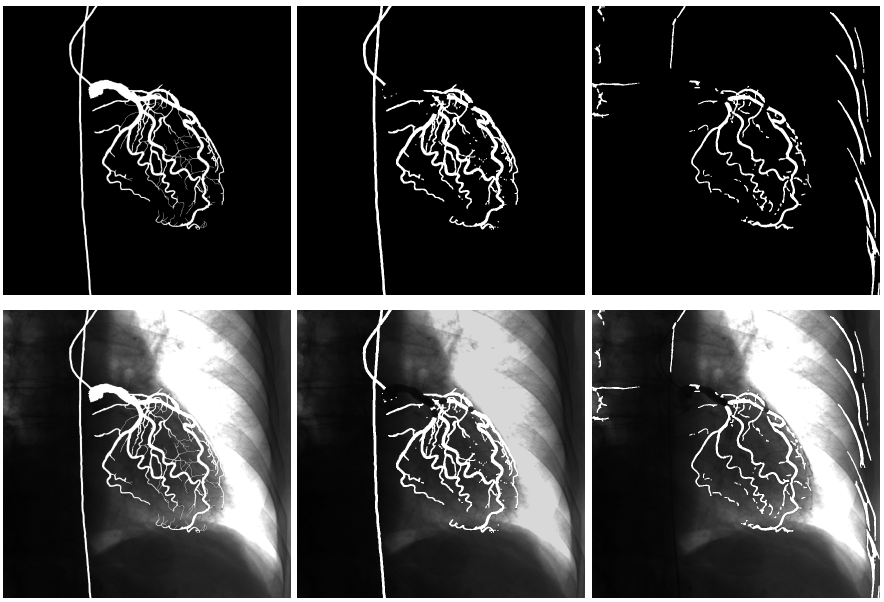
(a) original manually segmented images (b) adjusted manually segmented images (c) computed Frangi images

Figure B.11: Frangi compared to man. seg. images (Data set Coburg-070822B)



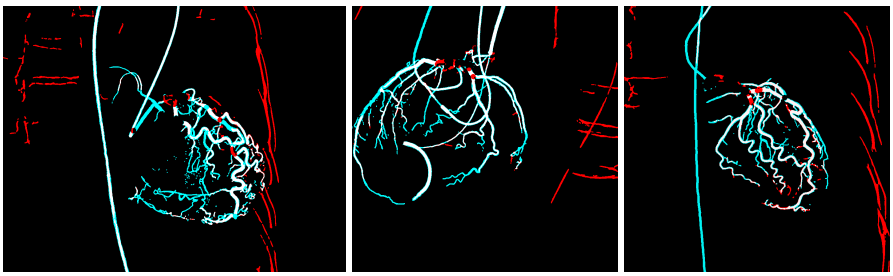
(a) original manually segmented images (b) adjusted manually segmented images (c) computed Frangi images

Figure B.12: Frangi compared to man. seg. images (Data set Coburg-070822E)



(a) original manually segmented images (b) adjusted manually segmented images (c) computed Frangi images

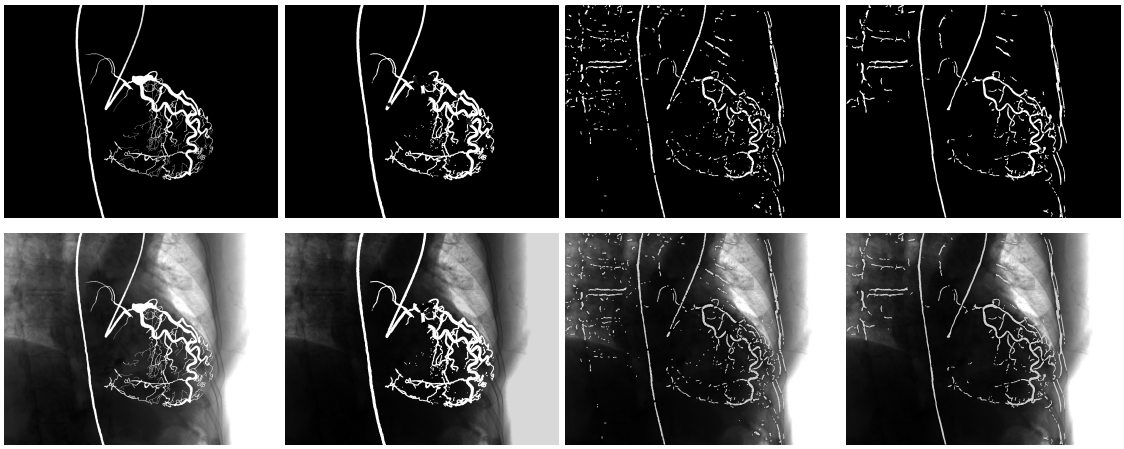
Figure B.13: Frangi compared to man. seg. images (Data set Coburg-071011C)



(a) Compared Frangi (red) and manually seg. images (blue), overlay pixels are white segmented (from left to right: data set Coburg 070822B, 070822E, 071011C)

Figure B.14: Compared data sets with Frangi and man. seg. images

Results Koller



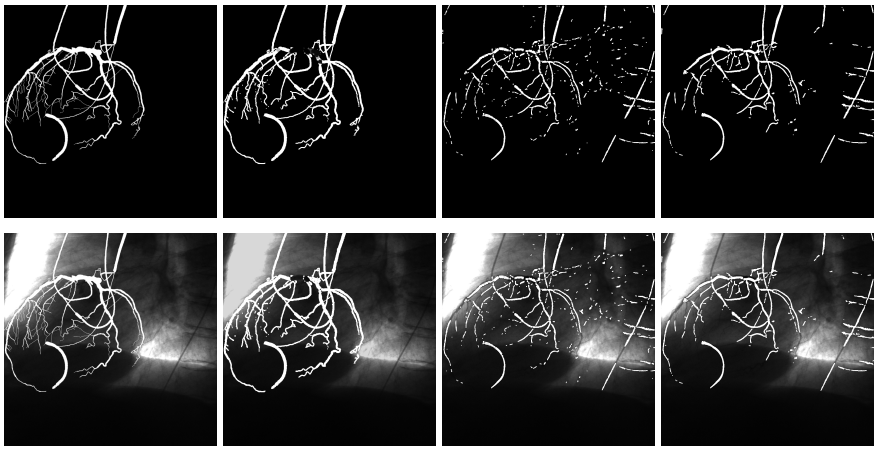
(a) original manually segmented images

(b) adjusted manually segmented images

(c) computed (min. connected lines of 5 pixels) Koller images

(d) computed double threshold Koller images

Figure B.15: Koller compared to man. seg. images (Data set Coburg-070822E)



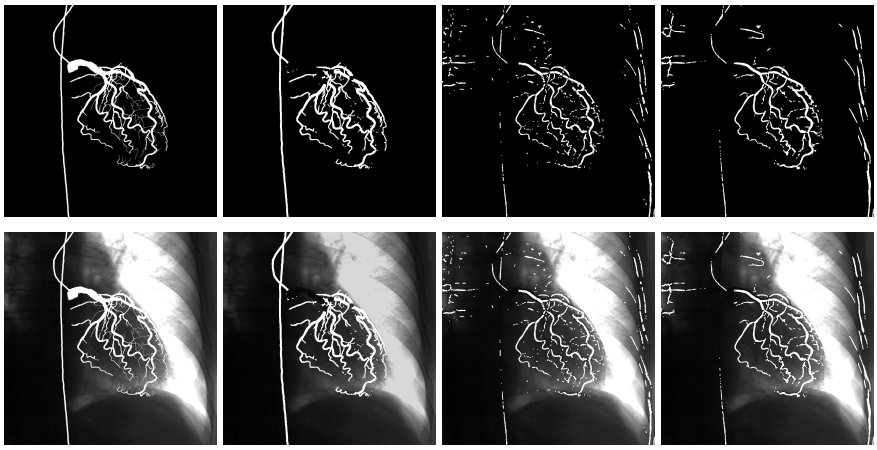
(a) original manually segmented images

(b) adjusted manually segmented images

(c) computed (min. connected lines of 5 pixels) Koller images

(d) computed double threshold Koller images

Figure B.16: Koller compared to man. seg. images (Data set Coburg-070822E)



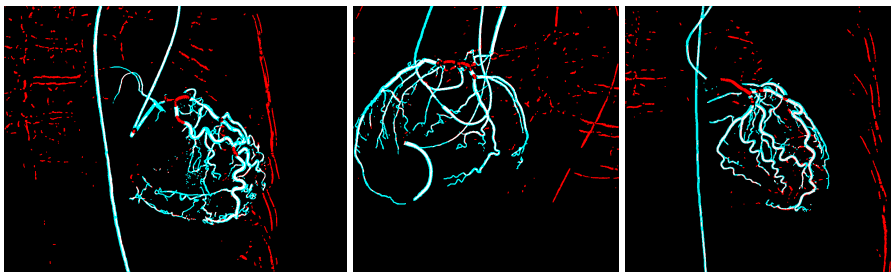
(a) original manually segmented images

(b) adjusted manually segmented images

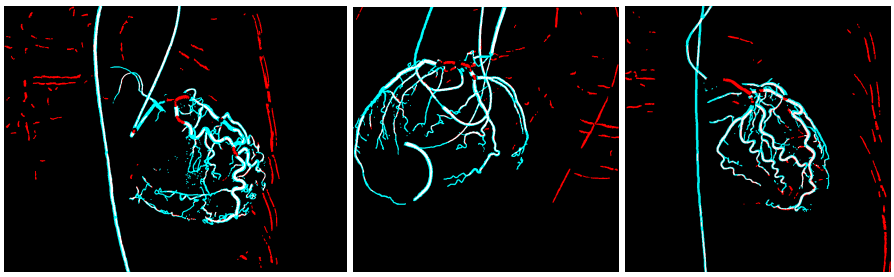
(c) computed (min. connected lines of 5 pixels) Koller images

(d) computed double threshold Koller images

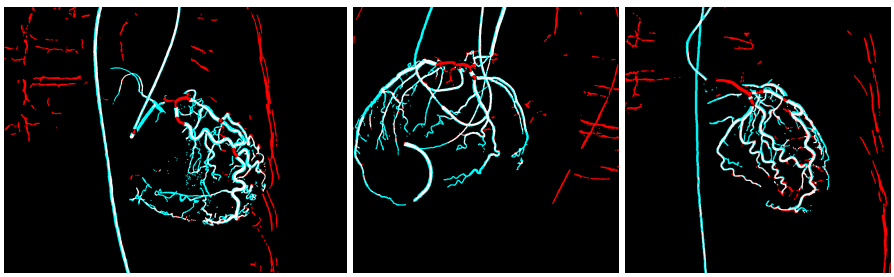
Figure B.17: Koller compared to man. seg. images (Data set Coburg-071011C)



(a) Compared Koller (min. connected lines of 5 pixels) (red) and manually seg. images (blue), overlay pixels are white segmented (from left to right: data set Coburg 070822B, 070822E, 071011C)



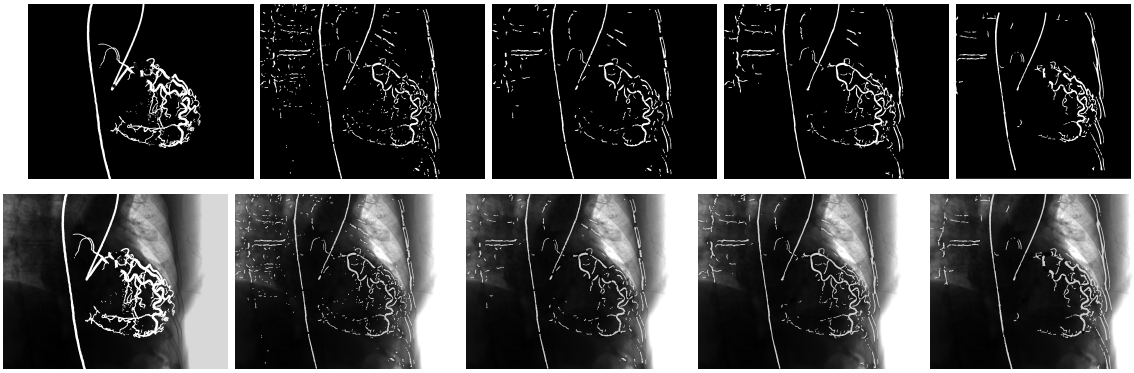
(b) Compared Koller (min. connected lines of 20 pixels) (red) and manually seg. images (blue), overlay pixels are white segmented (from left to right: data set Coburg 070822B, 070822E, 071011C)



(c) Compared Koller (only double threshold) (red) and manually seg. images (blue), overlay pixels are white segmented (from left to right: data set Coburg 070822B, 070822E, 071011C)

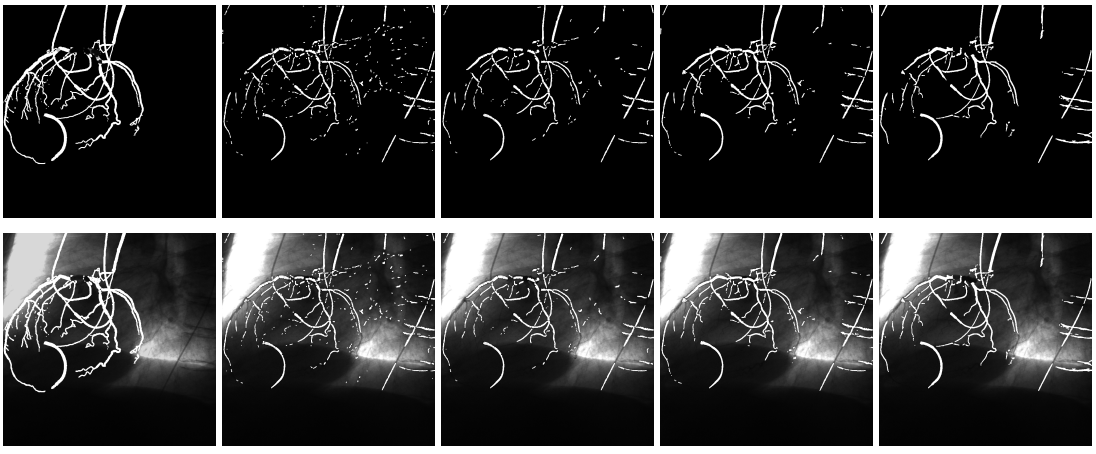
Figure B.18: Compared data sets with Koller and man. seg. images

All results compared



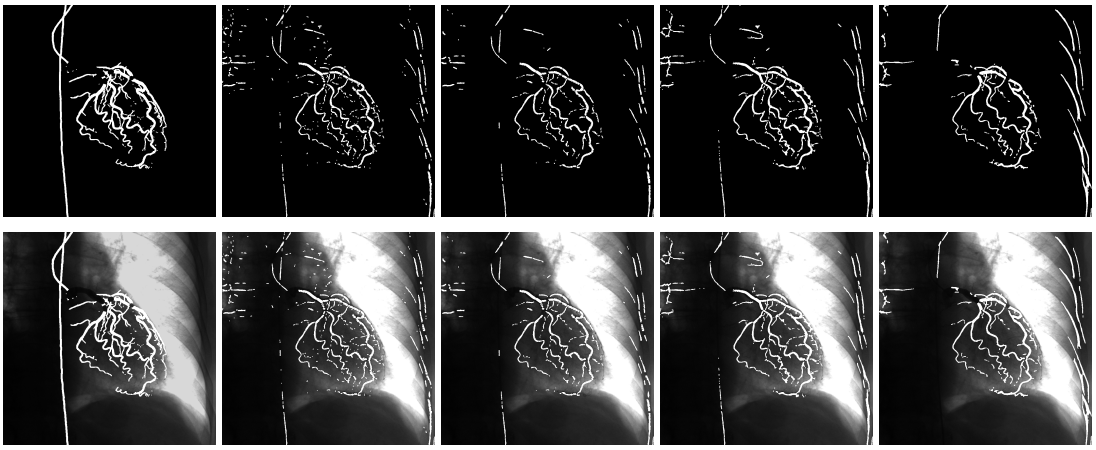
(a) adjusted manually segmented images | (b) computed (min. connected lines of 5 pixels) Koller images | (c) computed (min. connected lines of 20 pixels) Koller images | (d) computed double threshold Koller images | (e) computed Frangi images

Figure B.19: Koller, Frangi and man. seg. images compared (Data set Coburg-070822B)



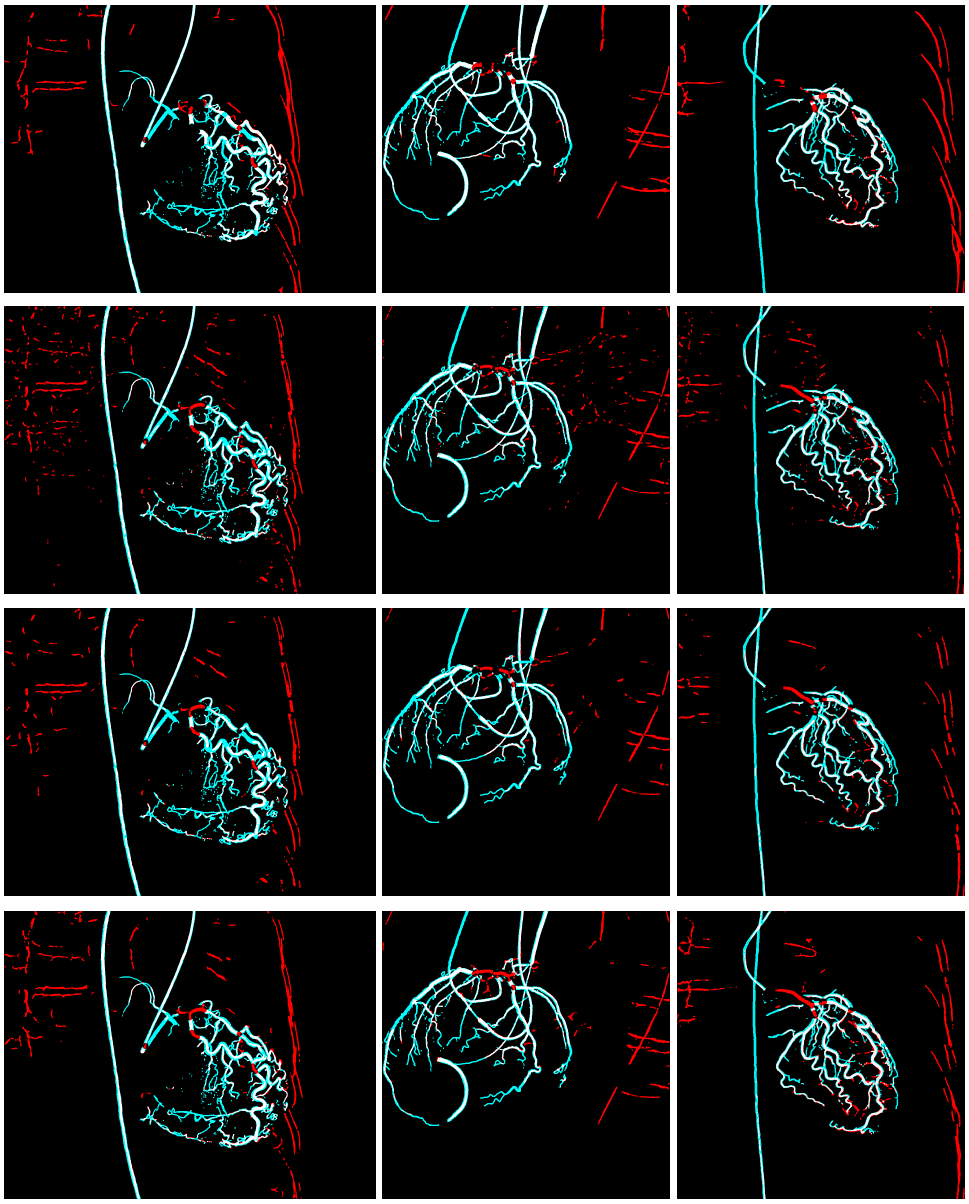
(a) adjusted manually segmented images | (b) computed (min. connected lines of 5 pixels) Koller images | (c) computed (min. connected lines of 20 pixels) Koller images | (d) computed double threshold Koller images | (e) computed Frangi images

Figure B.20: Koller, Frangi and man. seg. images compared (Data set Coburg-070822E)



(a) adjusted manually segmented images | (b) computed (min. connected lines of 5 pixels) Koller images | (c) computed (min. connected lines of 20 pixels) Koller images | (d) computed double threshold Koller images | (e) computed Frangi images

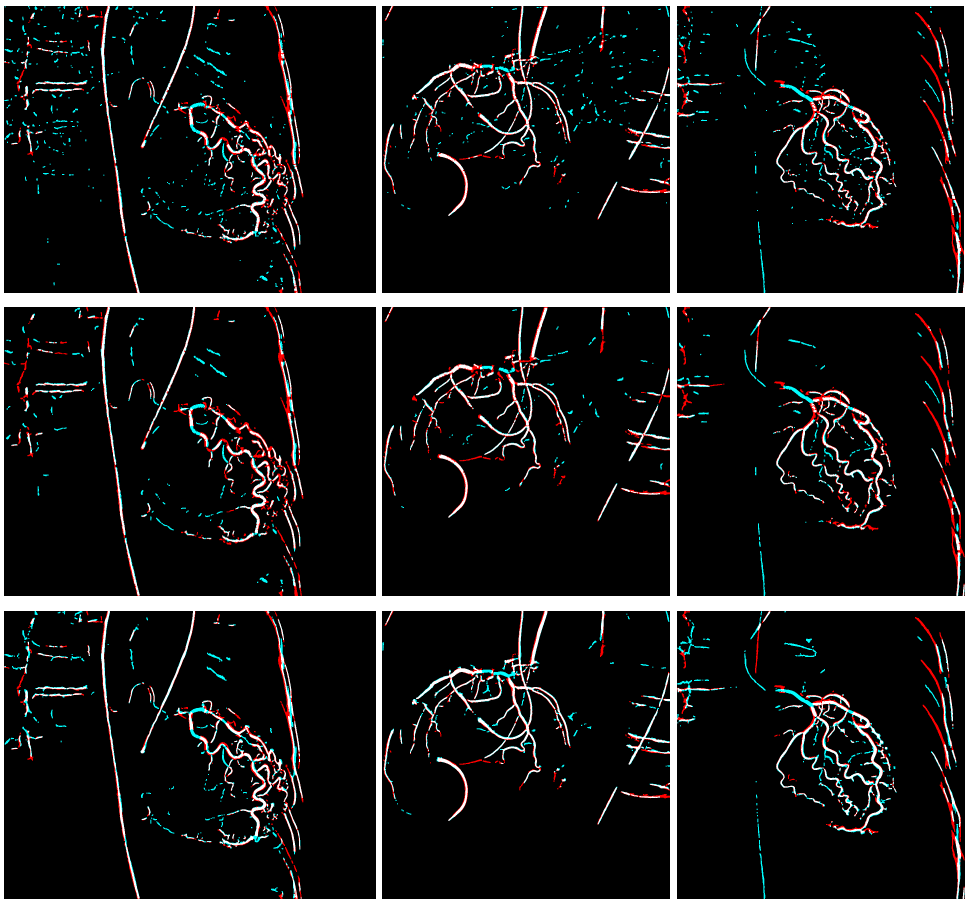
Figure B.21: Koller, Frangi and man. seg. images compared (Data set Coburg-071011C)



(a) Data set Coburg 070822B (b) Data set Coburg 070822E (c) Data set Coburg 071011C

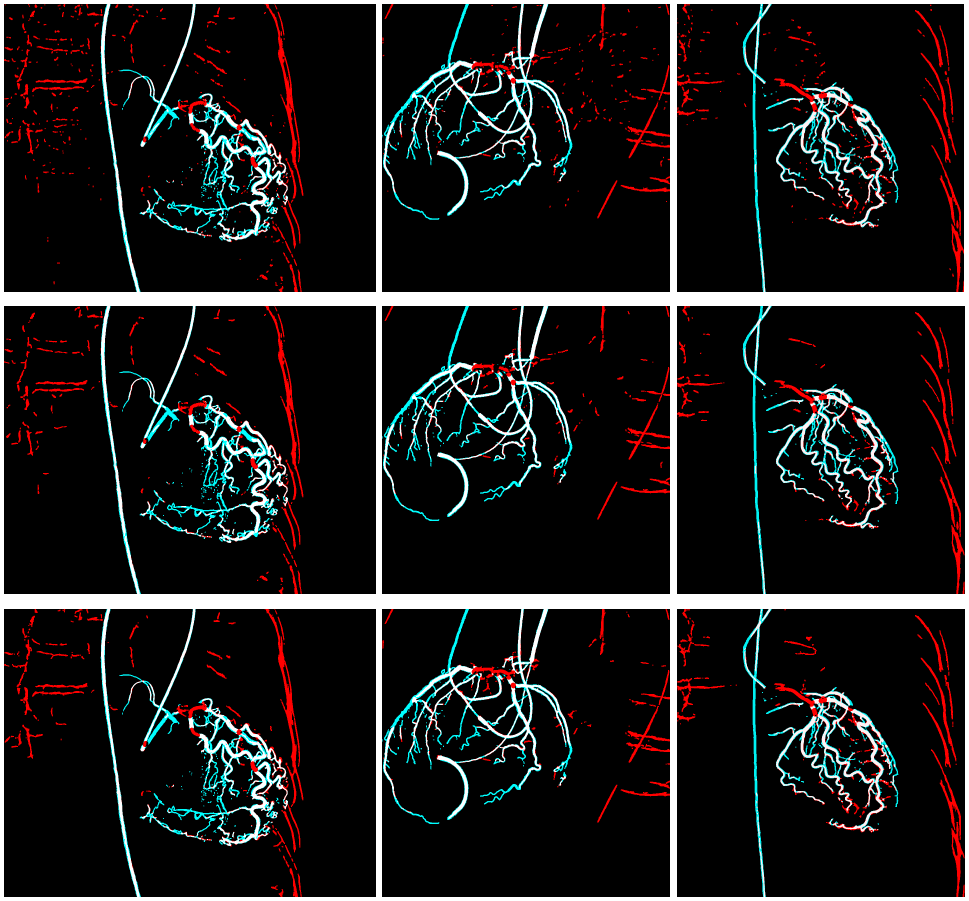
Figure B.22: Compared Frangi, Koller (red) with the manually segmentation images (blue), overlay pixels are white segmented (from top to bottom: Frangi, min. 5, 20 pixels connected component pixels and double treshold Koller images)

B.6 Conclusion



(a) Data set Coburg 070822B (b) Data set Coburg 070822E (c) Data set Coburg 071011C

Figure B.23: Compared merged Frangi (red) and Koller (blue) segmentation images, overlapped pixels are segmented white (from top to bottom: min. 5, 20 pixels connected component pixels and double treshold Koller with Frangi added images)



(a) Data set Coburg 070822B (b) Data set Coburg 070822E (c) Data set Coburg 071011C

Figure B.24: Compared merged Frangi, Koller (red) and manually segmentation images (blue), overlapped pixels are segmented white (from top to bottom: min. 5, 20 pixels connected component pixels and double treshold Koller with Frangi added images)

Bibliography

[BMV+06] C. BLONDEL, G. MALANDAIN, R. VAILLANT, , and N. AYACHE, *Reconstruction of Coronary Arteries From a Single Rotational X-Ray Projection Sequence*, IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 25 (2006), pp. 653–663.

[Can83] J. F. CANNY, *Finding edges and lines in images*, Technical Report 720, MIT Artificial Intelligence Laboratory, (1983).

[Can86] J. F. CANNY, *A computational approach to edge detection*, IEEE PAMI, 8 (1986), pp. 679–698.

[FNVV98] A. F. FRANGI, W. J. NIESSEN, K. L. VINCKEN, and M. A. VIERGEVER, *Multiscale vessel enhancement filtering*, In William M. Wells, III, Alan C. F. Colchester, and Scott L. Delp, editors, Proc. Int'l Conf. Med. Image Computing and Computer Assisted Intervention (MICCAI), volume 1496 of Lecture Notes in Computer Science. Springer (1998), pp. 130–137.

[GW02] R. C. GONZALEZ and R. E. WOODS, *Digital Image Processing*, Prentice Hall, 2nd edition (2002).

[HEM+99] K. HARIS, S. N. EFSTRTIADIS, N. MAGLAVERAS, C. PAPPAS, J. GOURASSAS, and G. LOURIDAS, *Model-based morphological segmentation and labeling of coronary angiograms*, IEEE Trans. Med. Imag., 18(10) (1999), pp. 1003–1015.

[ITK] ITK, *National Library of Medicine Insight Segmentation and Registration Toolkit*, http://www.itk.org/.

[Jäh02] B. JÄHNE, *Image Processing*, Springer, 5th revised and extended edition, (2002).

[KGSD95] T. M. KOLLER, G. GERIG, G. SZÉKELY, and D. DETTWILER, *Multiscale detection of curvilinear structures in 2-D and 3-D image data*, In Proc. Int'l Conf. Comp. Vis. (ICCV), (1995), pp. 864–869.

[KQ04] C. KIRBAS and F. QUEK, *A review of vessel extraction techniques and algorithms*, ACM Computing Surveys, 36(2) (2004), pp. 81–121.

[NYT04] D. NAIN, A. YEZZI, , and G. TURK, *Vessel segmentation using a shape driven flow*, In Christian Barillot, David R. Haynor, and Pierre Hellier, editors, Proc. Int'l Conf. Med. Image Computing and Computer Assisted Intervention (MICCAI), volume 3216 of Lecture Notes in Computer Science. Springer (2004), pp. 51–59.

[Set96] J. SETHIAN, *Level Set Methods and Fast Marching Methods*, Cambridge University Press, (1996).

List of Figures