Modelling of Cutting of Pig Carcasses

Allan Lyckegaard

Kongens Lyngby 2007

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Abstract

This thesis describes the methods used to analyse and model the cutting of pig carcasses into the products: back, belly, spine and ribs.

Automatic pre-processing methods to segment out the individual products from raw DICOM images are developed. A bit resolution reduction method is presented for producing images that are readable for other software.

Image registration method are studied and used for finding the cuttings, namely Thirion's demons, nonrigid B-spline based free-form deformations and thin plate splines. Image registration gave no success. Insted, a set of ad hoc method were delveloped for finding the cuttings, which turned out to give reasonable results.

Statistical shape models are presented and used on the found cuttings. The statistical shape model is based on a 13 eigenvector description spanning 95% of the data space. The outline for a virtual cutting is given.

No ground truth is available for the data. Instead, the results are evaluated by visual inspection, since no other possibility exist.

Resumé

Denne rapport beskriver metoderne benyttet til at analysere og modellere opskæring af svineslagekroppe til produkterne: kam, brystflæsk, ryggrad og ribben.

Automatiske præ-processerings metoder til at udskære produkterne fra de rå DICOM billeder er udviklet. En bitreduktionsmetode er præsenteret for at gøre billeddata læseligt for andet software.

Billedregistreringsmetoder er studeret og benyttet til at finde opskæringen, specifikt Thirion's dæmoner, ikke-rigid B-spline baseret free-form deformations og thin plate splines. Billedregistering var ikke succesfuld. I stedet er der udviklet et sæt ad hoc metoder for at finde opskæringerne, hvilket gav rimelige resultater.

Statistiske shape modeller er præsenteret og benyttet på de fundne opskæringer. Den endelige statistiske model er baseret på 13 egenvektorer som beskriver 95% af datarummet. Udkastet til en virtuel opskæring er givet.

Ingen ground truth er til rådighed. I stedet evalueres resultaterne ved visual inspektion, siden ingen mulighed findes.

Preface

This thesis has been prepared at the section of Image Analysis, Informatics and Mathematical Modelling (IMM), Technical University of Denmark (DTU). The project started November 6th, 2006, and finished May 14th, 2007. The amount of work correspond to 30 ECTS academic points.

The project is part of the requirements for obtaining the degree Master of Science Engineering (M.Sc.Eng).

Lyngby, May 2007

Allan Lyckegaard

Acknowledgements

First of all, I would like to thank my supervisor Rasmus Larsen for all his good comments, ideas and suggestions throughout the project.
Furthermore I thank the three ph.d-students Martin Vester-Christensen, Søren Erbou and Mads Fogtmann Hansen for taking their time to discuss and explain the things I did not understand.
Eli V. Olsen and Lars Bager Christensen from DMRI are thanked for providing data. Without you, there would be no project.
Finally, I thank my family and friends for being there for me. My most deepfelt thanks goes to my mother and father, without you I would not be where I am today.

Til sidst vil jeg takke min familie og venner for at være der for mig. Min mest dybtfølte tak går til min mor og min far, uden jer ville jeg ikke være hvor jeg er i dag.

Contents

Abstract **i**

Resumé **iii**

Preface **v**

Acknowledgements **vii**

1 Introduction **1**

1.1 Motivation 1

1.2 Description of Dataset 3

1.3 Computed Tomography 5

1.4 Objectives 5

1.5 Overview 5

1.6 Mathematical Notation 6

2 Pre-processing of Dataset 7

2.1 Shelf removal 7

2.2 Reduction of bit resolution 11

2.3 Summary and discussion 12

3 Finding Cuttings using Image Registration 15

3.1 Image Registration Theory 15

3.2 Four Methods for Image Registration 24

3.3 Results 33

4 Finding Cuttings using Ad hoc Methods 51

4.1 Description of the Ad Hoc Methods 51

4.2 Results 62

5 Statistical Shape Model of Cuttings 69

5.1 Preliminaries 69

5.2 Inter-middle registration 72

5.3 Statistical Shape Model 73

5.4 Virtual Cutting 75

5.5 Discussion 77

6 Conclusion 79

6.1 Conclusion 79

6.2 Future work 80

A Pig middle scanning numbers **81**

B Software **83**

Chapter 1

Introduction

1.1 Motivation

Since the middle of the 19th century, when the Danish farmers started to join in cooperative movements, the Danish agrocultural industry has been steadily growing. Today the Danish farmers and slaughterhouses produce and process about 25 million pigs per year [2].
At the slaughterhouse the pig is separated into three parts: Fore-end, middle and the hind leg. The fore-end contains a lot of bone structures, but can be processed into roast-type of cuts. The hind leg is what you buy, when you buy ham. The middle is basically the part of the pig from the shoulder and down to the hips and is the most valuable part of the pig. The middle is cut into four parts (some times refered to as products or cuts): spine, ribs, loin and belly. The spine is usually not used for human consumer products and the ribs can be sold as spareribs. The loin can be used for the probably most famous product of them all, back bacon, or for the danish speciality *flæskesteg*. The belly can be processed into *streaky bacon* or the Italian speciality *pancetta*.
The slaughtering process of a carcass include heavy manual labour and can give rise to health problems, e.g. back problems, for the persons working in the slaughterhouses. Particularly, the first steps in the slaugthering process, splitting the whole pig carcass into parts of more manageable size, is a wearing job. For the same reason automatic slaughterhouse robots have been developed

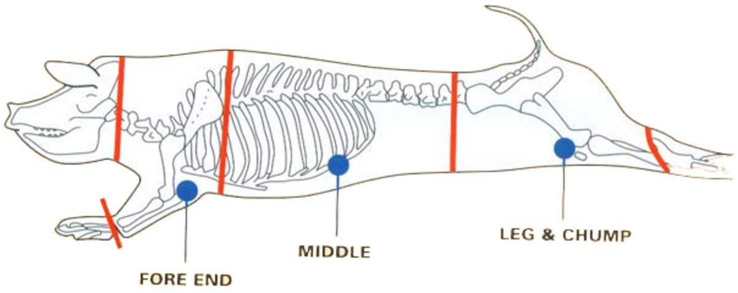

Figure 1.1: The pig is separated into three parts: Fore-end, middle and hind leg [1].

to reduce this amount of heavy manual labour. One type of slaughterhouse robot is a *middle cutting machine*, see figure 1.2.

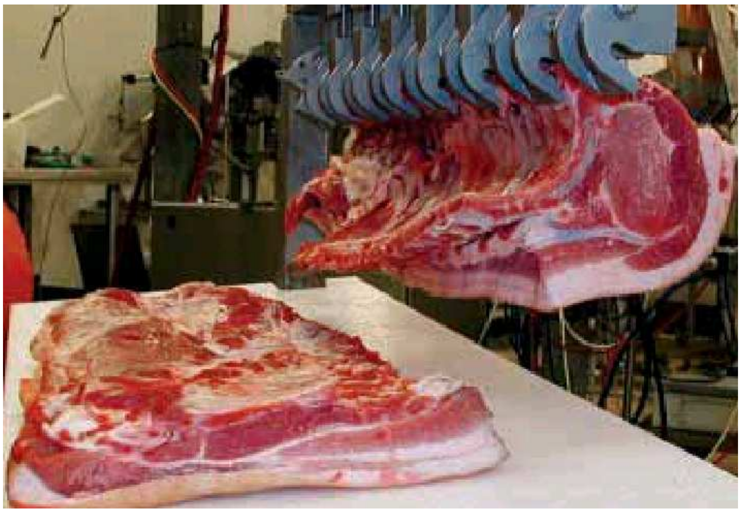

Figure 1.2: The middle cutting machine in action. The belly is on the table. The back with the ribs is hanging from the hooks.

The middle cutting machine cuts the middle of the pig into spine, ribs, back and belly, see figure 1.3. The position of the cuts can be adjusted so the machine e.g. will cut the back with a specific size.

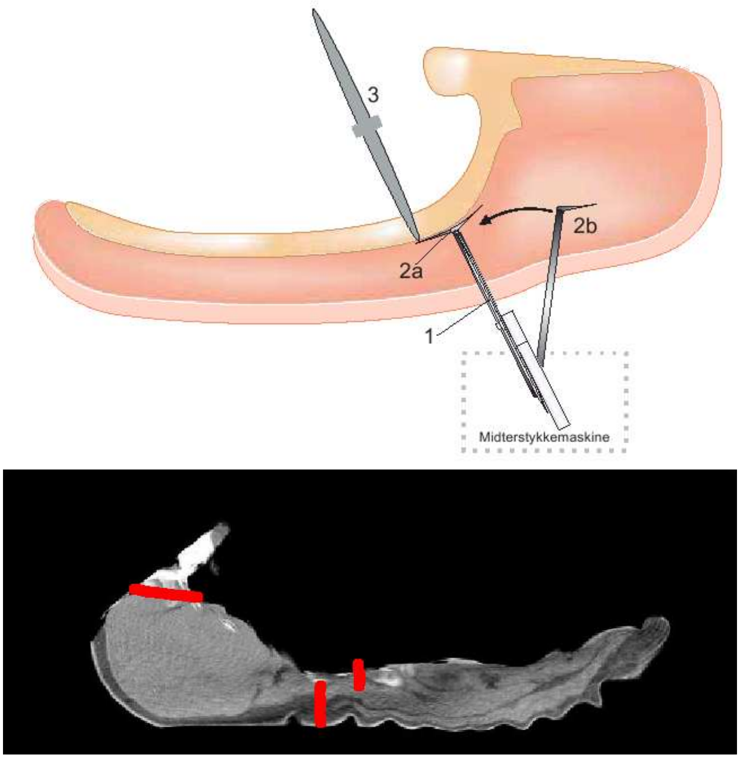

Figure 1.3: Models of the cut made by the middle cutting machine.

1.2 Description of Dataset

The data set available is 39 CT scans of pig middles. The middles are physically quite similar except for the number of ribs. The number of ribs varies between 10 and 12. Some of the lowest ribs may also be more or less cartilagelike. All 39 middles have been scanned before processed by the middle cutting machine. 37 of the middles were scanned after they were processed into products. 2 pig middles were not completely inside the field of view of the scanner so they left out from the analysis, giving a dataset of 35 pigs.
Each scan has the size 512 pixels by 512 pixels in xy-direction with a variable size in the z-direction, usually between 55 and 65 pixels. The pixel size is 1mm by 1mm by 10mm in the xyz-direction, respectvely. In the low resolution z-direction the feed is 10mm and the slice thickness is 10mm meaning the

slices are averages of 10mm. Due to this fact, the spine is aligned with the z-direction during scanning, since the anatomical structures of the pig changes the least along this. The avereging in the z-direction destroys the correspondence between the whole pig middle and the products. If same product is scanned twice, but the second time with a slight displacement wrt. the scanners coordinate systme, the averaging volume will change, meaning the measures data will change, see figure 1.4.
The middle itself is placed on a shelf in the center of the scanner. For the scanning of the four products, the ribs and the spine a laid in a semi-flexible tray in the bottom of the scanner with the two other products placed on shelves above, the loin in the center and the belly on the top.
Each pixel value is recorded in monochrome 12-bit resolution, i.e. 4096 gray tone levels. Since no 12 bit IEEE type exist the values are stored as unsigned 16 bit on the disk. The fileformat used is DICOM (Digital Imaging and Communications in Medicine,[14]) which is often seen in medical applications and equipment. Besides image data the DICOM fileformat also contains a image header which describes all possible extra information, such as patient (pig middle) identification number, weight, etc., which may be valuable in the further analysis. CT data can have negative values (e.g. air, -1000 HU), why the DICOM header also contains an offset and scaling to do a linear transformation of the values.

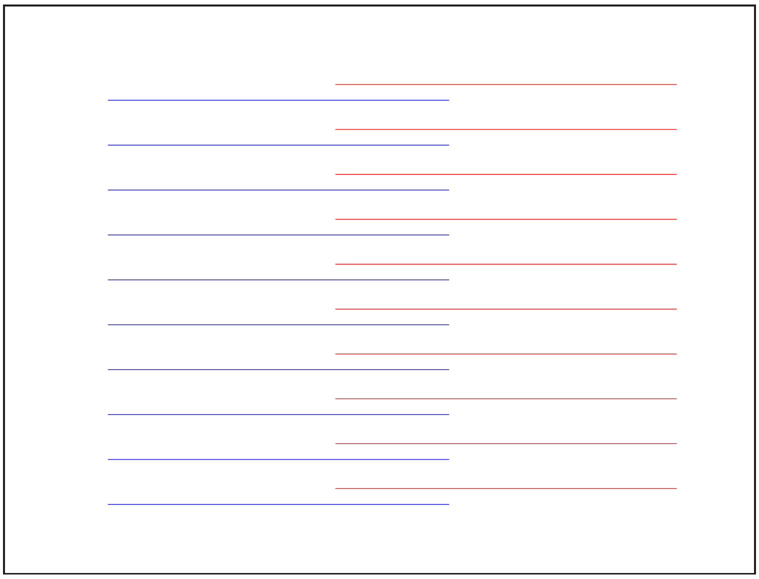

Figure 1.4: Red and blue are the same product scanned twice, but with a slight displacement, meaning the averaging volume is not the same.

1.3 Computed Tomography

The data used in this analysis are 3D computed tomography(CT) images. The creation of 3D CT images in the CT scanner is basically a two step procedure: Data acquisition and data processing. The data acquisition is done by exposing the object in the scanner to X-rays from one side of the object and detect non-absorbed rays on the opposite side of the object. Fixing X-ray tube and detector on a ring, and rotating this, a number of scans from different angles are obtained. The data processing is done using a reconstruction algorithm on a computer to get human readible images.
The 3D CT images are monochrome (though dual energy scanner are available) and each voxel specify the absorbtion of the material in Hounsfield Units (HU). The absorbtion for tissues interisting in this context are shown in table 1.1 A

Tissue	**Absorbtion**
Air	-1000
Water	0
Fat	$-60 \rightarrow -100$
Soft tissue, muscle	$40 \rightarrow 80$
Bone	$400 \rightarrow 1000$

Table 1.1: Absorbtion in Hounsfield Units [HU] for different tissues([5]).

thorough description of the principle behind CT is out of the scope of this report and refer to specific books on this topic like [5].

1.4 Objectives

The main objective of this thesis is to analyse the cuttings of pig carcasses and build a mathematical model of these. More specific, only the cuttings of the middle of the pig is analysed. The use of image registration for analysing the cuttings is investigated. If not succesful, a set of ad hoc methods should be applied. Eventually, a statistical shape model of the cuttings should be found and used for making virtual cuttings of pig middles.

1.5 Overview

Chapter 2 describes pre-processing of the dataset.

Chapter 3 covers image registration and applies it

Chapter 4 introduces a set of ad hoc methods and applies them

Chapter 5 study statistical shape models and virtual cuttings

Chapter 6 gives the concluding remarks

1.6 Mathematical Notation

For clarity, the following mathematical notation is used throughout the thesis.

f	Functions values and scalars are in lowercase, normal font.
$\mathbf{v}$	Vectors are in lowercase, bold font. Vectors are column vectors.
$\mathbf{x}$	Point sets are in lowercase, bold font.
$\mathbf{M}$	Matrices are in uppercase, bold font.
$\mathcal{I}$	Images are in calligraphic font.
$I(\cdot)$	Intensities are in uppercase, normal font.

Chapter 2

Pre-processing of Dataset

The pre-processing of the dataset falls in two parts: One where the shelves are cut away and the individual products are isolated, and another where the intensity information is is reduced to a 8-bit resolution.

2.1 Shelf removal

In the scanner the products are placed on a shelf construction, see figure 2.1. These shelves must be removed before any further analysis can be done. The products are placed on three different shelves, two solid wooden ones in the top and a flexible one of unknown material at the bottom. The whole pig middle is placed on a single solid wooden shelf.

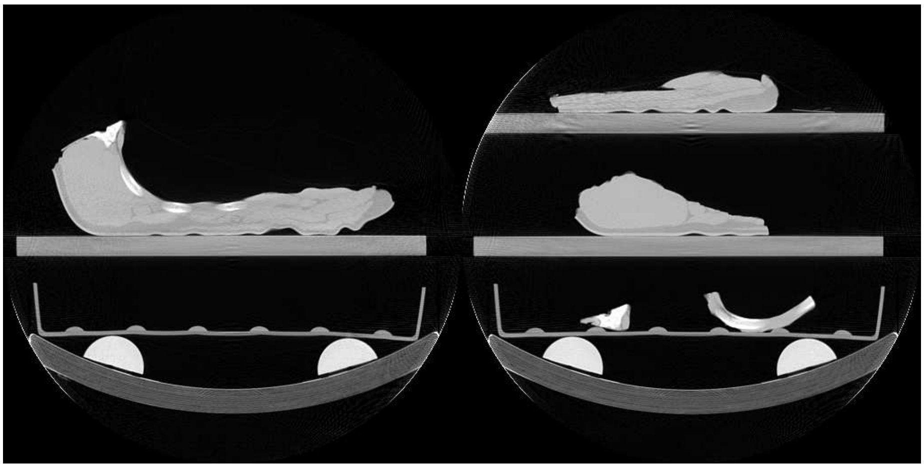
Figure 2.1: Shelf construction of both the whole middle and the processed.

2.1.1 Removal of wooden shelves

The shelves are in this case parallel with the horizontal axis, which makes things a lot easier. The procedure for removal is:

1. take the derivative of the image in the vertical direction (filter with $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$)
2. sum along the horizontal axis
3. the two most negative peaks are the top of the wooden shelves
4. set pixel intensity from the negative peaks and 40 pixels down to 0

Figure 2.2 shows the derivative of the image and the summation along the horizontal axis.

2.1.2 Removal of flexible shelf

The problem with the bottom shelf is that it is flexible and therefore cannot be removed as simple as the wooden shelves. Additionally, the intensity of the

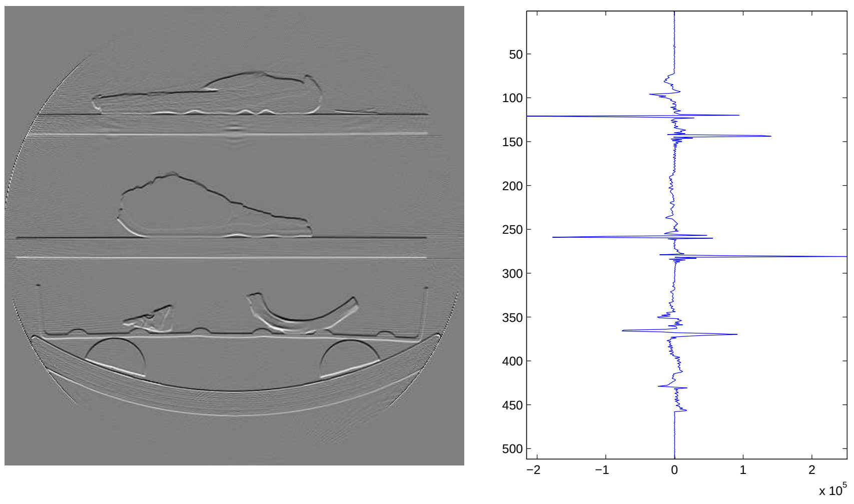

Figure 2.2: Left: Derivate of image. Right: Summation along the horizontal axis.

shelf is very close to that of bones, so thresholding is not a possibility. But there are some things we know from manually inspecting the image: The shelf is 7 pixels thick and it has 6 bumps placed with a regular distance. The solution is to cut out a single bump and correlate this with the image. Points with a high correlation almost on a straight line along the horizontal axis will be bumps, see figure 2.3. Knowing the bumps, we can cut them away and likewise cut away 7 pixel thick line between the bumps. We approximately know the distance from the outermost placed bumps and to the sides of the shelf, so we just remove everything further away than this distance.

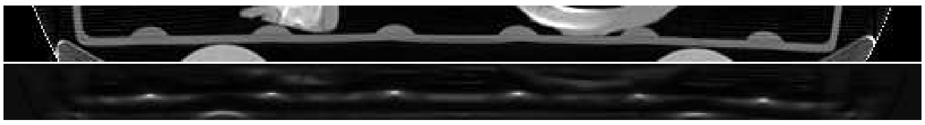

Figure 2.3: Top: Flexible shelf. Bottom: Image correlated with a bump template.

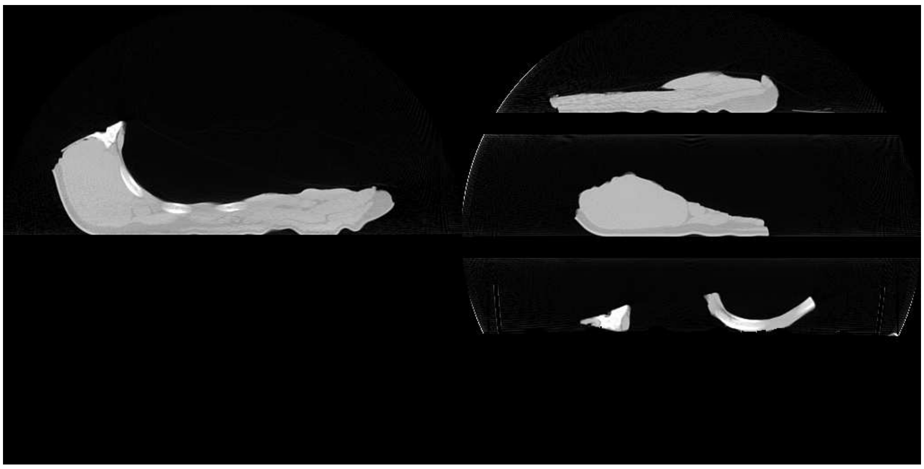

Figure 2.4: Images after removal of shelves.

2.2 Reduction of bit resolution

Some of the image registration implementations that we will use (Thirion's demons and nonrigid B-spline based free-form deformations), can only handle integers from 0-255 (8 bit). Since the original data has values below 0 and above 255, we need to do some kind of scaling. The original data are stored in 16 bit, so the easiest thing to do, would be to divide by 2^8 and round off, but this has the side effect that we lose precision, since our data does only fills up a part of the range from 0 to 2^16. Looking at a histogram of intensities of a typical pig middle might guide us to a solution, see figure 2.2.

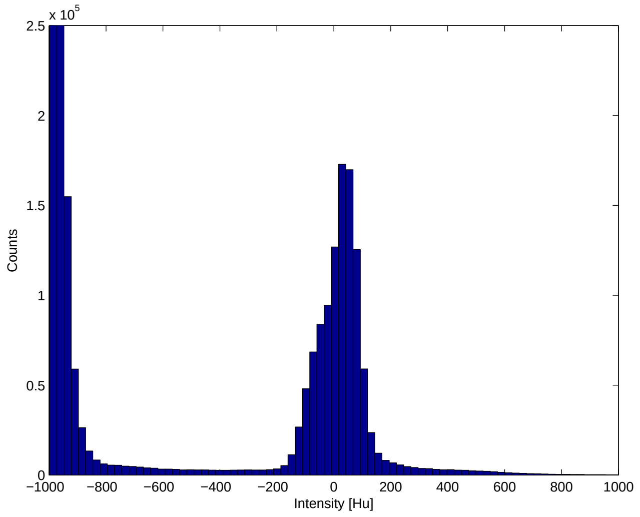

Figure 2.5: Histogram of the original pig middle.

The histrogram shows us a couple of things. First, there is a clear seperation of background and the pig middle, and furthermore the intensity range of the pig middle is only around 500 Hu. We can therefore do a truncation of the intensity spectrum in the following way:

1. set all values above 300 Hu to 300 Hu.
2. set all values below -200 Hu to -200 Hu.
3. divide by 500 and multiply by 255.
4. round off and store as `uint8`.

We cut off at -200 Hu to ensure that there still is an intensity gap between the background and the pig middle. Likewise, we cut off at 300 Hu because above this intensity there is only bone. Figure 2.2 shows the histogram of the scaled pig middle, there is still an intensity gap between background and the pig middle. Figure 2.2 shows the the origninal, the scaled and difference image between. The reverse procedure has been applied to the scaled to match the orginal again. It is clear that there is very little visual difference between the original and the scaled, the difference image shows us that the only place where there is an difference is at the bones.

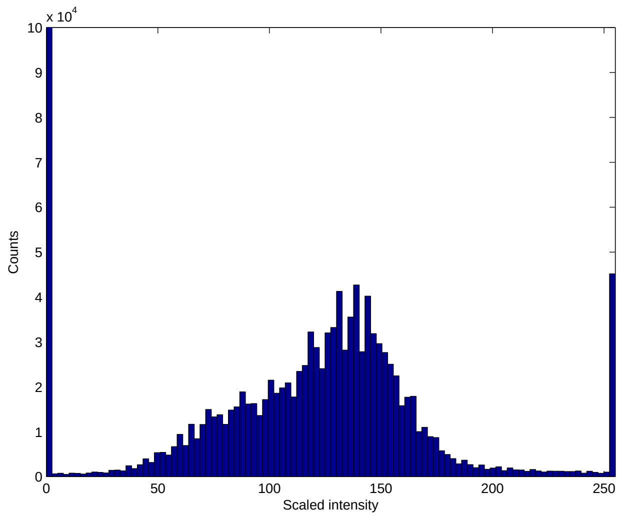

Figure 2.6: Histogram of the scaled pig middle.

2.3 Summary and discussion

A method for removing the shelves in the raw image data was described. Likewise, a method for reducing bit resulution. It is always a problem to find the best way or least worse way to reduce data resolution. Here we cut away some of the intensities that holds very little information. How much you should cut adn where can be discussed, some kind of truncation is needed to avoid too low dynamic range.

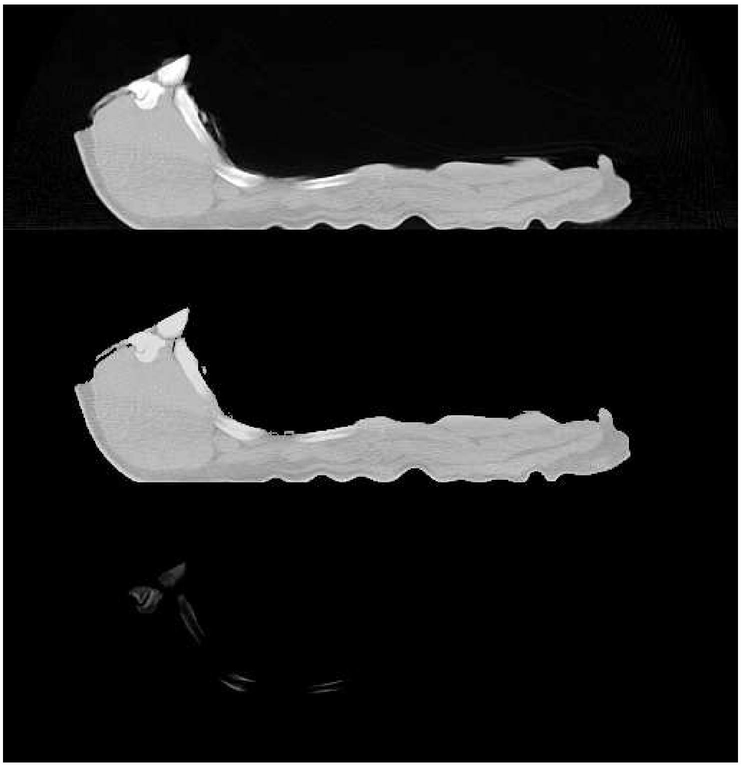

Figure 2.7: Top: Original image. Middle: Scaled image, shifted back to match the original. Bottom: Difference image.

Chapter 3

Finding Cuttings using Image Registration

This chapter will study the use of image registration for finding cuttings. First the concept of image registration is introduced, next the methods studied in this project is described in detail and finally the results are shown.

3.1 Image Registration Theory

This section will describe the image registration in general terms and introduce some mathematical notation for later use. First let us define what an image is:

Definition An image $\mathcal{I}$ is a finite set of points, $\boldsymbol{x} \subset \mathbb{R}^d$, on a regular grid with a corresponding intensity, $I(\boldsymbol{x}) \subset \mathbb{R}$.

One could simply see an image as a mapping from $\mathbb{R}^d$ to $\mathbb{R}$. With this at hand we define image registration as:

Definition Image registration is the process of geometrically transforming collected images so they become similar to one another.

Or more mathematical: Given a reference image $\mathcal{R}$ and a template image $\mathcal{T}$, we seek a geometric transformation $W(\cdot)$ such that

$$R(\boldsymbol{x}_{\mathcal{R}}) = T(W(\boldsymbol{x}_{\mathcal{T}}, \boldsymbol{p})). \tag{3.1}$$

The vector $\boldsymbol{p}$ contains parameters for the transformation. Equation (3.1) is of course an ideal situation, where the transformed template image is completely similar to the reference image. Often, the images we are dealing with come from different sources and they are most certainly noisy, so it would be more realistic to exchange the $=$ with $\simeq$. So how do we find the transformation $W(\cdot)$ that performs the correct mapping and how do we know that the mapping is correct? Well, to find an answer to this, there are some things to consider:

- What kind of transformation $W(\cdot)$ should be used?
- How is similarity between images measured?

The following will discuss these topics in detail.

3.1.1 Choice of Transformation

The are many aspects to consider when choosing the transformation. Transformations vary from very simple general ones to advanced specialised ones. When choosing transformation, one must consider some important aspects. The most important ones will be described below.

3.1.1.1 Domain

The domain can either be global or local. A global transformation is a transformation where any point is controlled by the same transformation. A global translation, $W(\boldsymbol{x}, \boldsymbol{p}) = \boldsymbol{x} + \boldsymbol{p}$, is a good example; all points in the image are translated by the same amount.
A local transformation means that the transformation changes according to the local neighbourhood. A local neighbourhood can be anything from a single pixel to larger areas (volumes), making it capable of modelling local characteristics or deformations. A local deformation is often preceded by a global to ensure fast and correct convergence.

3.1.1.2 Rigidity

A transformation can be rigid or nonrigid. A rigid transformation resembles the way a rigid body is moved, allowing only for rotation and translation. In the 2-dimensional case, one could think of a picture painted on a sheet of paper in the middle of a table. The only way to "transform" the image (wrt. the middle of the table) is to rotate and translate the paper sheet.
A nonrigid transformation can contain bends, curvature and stretch. Exemplifying again, think of a picture painted on a sheet of rubber. Now the picture can be translated, rotated, stretched and shrunk both locally and globally.

3.1.1.3 Parametrisation

Parametrisation tells whether the transformation is controlled by parameters or not. A translation, $W(\boldsymbol{x}, \boldsymbol{p}) = \boldsymbol{x} + \boldsymbol{p}$, is perfect example of a parametrised transformation: By adjusting the translation parameter $\boldsymbol{p}$ the transformation is controlled.
Nonparametric transformations are not controlled by parameters, only the template and reference images, and derivatives of these are controlling the transformation. For the nonparametric transform it is convenient to introduce a displacement field, u, and put the transformation on the form $W(\boldsymbol{x}) = \boldsymbol{x} + u(\boldsymbol{x})$. The whole idea is then to find the displacement field u that makes the two images most similar. But by introducing a displacement vector for every pixel there is no guarantee that the displacement field u will be smooth, hence the problem is ill-posed. By adding a regulariser when solving the problem, the smoothness of u can be controlled.[12, chap. 8]

3.1.1.4 Other Issues

Besides the things mentioned in the previous, there are some minor things one can consider when choosing the transformation. One thing is diffeomorphism. A diffeomorphic function is smooth and the inverse exists. In image registration terms this means that folding cannot occur. It is possible to get a diffeomorphic registration by constraining the determinant of the Jacobian of the transformation to be greater than zero, $\det(J(W)) > 0$.
An even stronger constraint is volume preservation, where the determinant of the Jacobian of the transformation must be exactly unity. Volume preservation can be used when working with e.g. physical incompressible tissue. It will ensure that the solution is feasible from a physical point of view.

3.1.1.5 Forward versus inverse transformation

In our definition of an image, we state that image points $\boldsymbol{x}$ is exist on a regular grid. A problem occurs when the template image points are transformed onto the domain of the reference image, since there is no mechanism to ensure that $W(\boldsymbol{x}_{\mathcal{T}}, \boldsymbol{p})$ exist on the same regular grid as $\boldsymbol{x}_{\mathcal{R}}$. The two images must exist on the same regular grid or else it is not possible to compare the two. An example could be a translation by 1.5 where the regular grid spacing is 1, see figure 3.1.

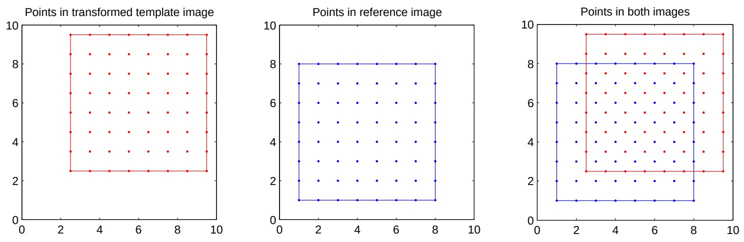

Figure 3.1: Left: Translated template image points. Middle: Reference image points. Right: Both point sets plotted. The two point sets do not exist on the same regular grid.

A way to overcome this problem is interpolation. The interpolation can either be performed in the untransformed or transformed template. We first consider the untransformed case and next the transformed.
If we do an inverse transformation of the regular grid in the reference image, $W^{-1}(\boldsymbol{x}_{\mathcal{R}})$, these points now exist in the frame of the template image. Interpolation in this frame is simple since the values to interpolate between are on a regular grid. With a bit of book keeping it is easy to go back again to the reference frame. The biggest problem is the existence of the inverse transformation W^{-1}. For translations and rotations it is simple to find the inverse transformation, but if the transformation is nonrigid and local, things start to get a bit more complicated. Fortunately, it is not necessary to know both forward and inverse transformations, in fact, you just decide what kind of transform the inverse should be and ignore the forward. This way, you get the advantage of interpolation on a regular grid without caring about the forward/inverse existence of the transform. The interpolation method can be chosen from a large variety. Among the most common are nearest neighbour, (bi-/tri-)linear, cubic or spline.
If you for some reason must work with the forward transformation W it is still

possible. The problem occurs because the template image points transformed onto the reference image, $W(\boldsymbol{x}_{\mathcal{T}})$, no longer are on a regular grid. In this case the interpolation is performed on the regular grid of the reference image and the values to interpolate between located in an irregular grid. In this case the interpolation could be done by Delaunay triangulation, kriging or thin plate spline. The drawback of interpolation on a irregular grid is that it requires a lot more computations than interpolation on a regular grid [8]
When nothing else is stated, it is assumed that the inverse transformation is used.

3.1.2 Choice of Similarity Measure

Before mentioning similarity measures we need to introduce registration bases. When data have been collected, one has to consider what features can be extracted. The extracted features are the registration bases. Three different types of registration bases will be studied here: Landmarks, surfaces and intensities. Similarity measures are dependent on the type of registration basis, so each one will be studied separately.

3.1.2.1 Landmarks

Landmarks, sometimes called fiducial points, are a set of distinct points in an image which have corresponding points the another image, see figure 3.2. By matching corresponding points in the two images the transformation can be found, or if the transformation is decided beforehand, landmarks can be used to evaluate how good the transformation is. Landmarks can either be set by an operator or extracted automatically using e.g scale-invariant feature transform (SIFT)[9].
Regardless of the method of extraction, there will be displacement errors on the landmarks, e.g. due to finite resolution of the image. These kind of errors are known as fiducial localisation errors ($\boldsymbol{FLE}$) and considered random with mean $\mu_{|\boldsymbol{FLE}|} = 0$ and variance $\sigma^2_{|\boldsymbol{FLE}|}$. Notice that $\boldsymbol{FLE}$ is a set of vectors, one for each landmark. It is not possible to determine $\boldsymbol{FLE}$ without ground truth, but assumptions can be made on basis of how well defined the structure in the image is.
Having decided which kind of transform W to use, the N_l landmarks in the template image, $\boldsymbol{l}_{\mathcal{T}}$, are now mapped into the domain of the reference image. The misalignment between corresponding landmarks is called fiducial registration

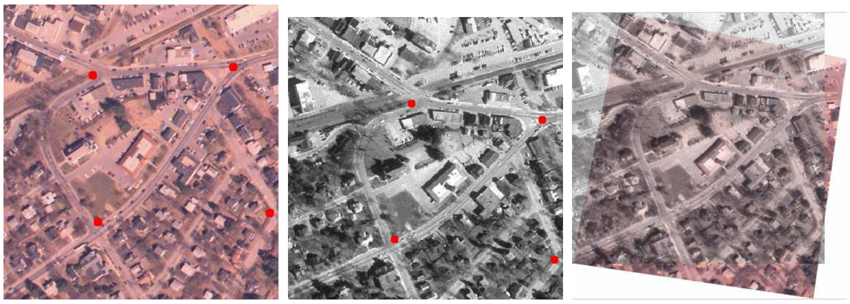
Figure 3.2: Example of image registration using landmarks as basis. Four distinct landmarks (marked by ●) are extracted from both target and reference image. Matching corresponding points in the two images makes it possible to merge the two. From [10]

error ($\boldsymbol{FRE}$). Mathematically this is expressed

$$\boldsymbol{FRE}_i = W(\boldsymbol{l}_{\mathcal{T},i}, \boldsymbol{p}) - \boldsymbol{l}_{\mathcal{R},i} \quad , \quad i = 1..N_L \tag{3.2}$$

From equation (3.2) we can compute the overall misalignment between the landmarks, FRE:

$$\mathrm{FRE}^2 = \frac{1}{N_l}\sum_{i}^{N_l} w_i^2 |\boldsymbol{FRE}_i|^2 = \frac{1}{N_l}\sum_{i}^{N_l} w_i^2 |W(\boldsymbol{l}_{\mathcal{T},i}, \boldsymbol{p}) - \boldsymbol{l}_{\mathcal{R},i}|^2 \tag{3.3}$$

where w_i is a weighting factor. If there is knowledge about the precision ($\sigma^2_{|\boldsymbol{FLE}|}$) of the individual landmarks, the weighting w_i can be scaled according to the this, else it is set to unity. The FRE can directly be used as similarity measure; the smaller the FRE, the closer the transformed template landmarks are to the reference landmarks. If working with parametrised transforms, the FRE can be used to estimate the parameter vector $\boldsymbol{p}$ by using equation (3.3) as cost function and $\boldsymbol{p}$ as unknown in an optimization scheme. [20, chap. 8.3]

3.1.2.2 Surfaces

Surface registration basis can be seen as an extension of the landmark registration basis. Surface registration operates with data points and model surface. The N_d data points are measurements (in template image) from e.g. a 3D laser scanner, and the model surface (in reference image) is what we register onto. The surface model is not necessarily an analytic surface, but could e.g. be a spline fitted to point measurements. The similarity measure used is the disparity function δ

$$\delta(\boldsymbol{d}, \boldsymbol{s}) = \sqrt{\sum_{i}^{N_D} w_j^2 ||W(\boldsymbol{d}_i, \boldsymbol{p}) - C(\boldsymbol{d}_i, \boldsymbol{s})||^2} \tag{3.4}$$

where $\boldsymbol{d}$ denotes data points and $\boldsymbol{s}$ model surface. Unlike in the landmark registration case, no correspondence is given here, so instead we have to use the nearest neighbour operator C. $C(\boldsymbol{d}_i, \boldsymbol{s})$ returns the point on the model surface $\boldsymbol{s}$ closest to a given data point $\boldsymbol{d}_i$. Like in the case with landmarks and FRE, the disparity function δ can be used together with an optimization scheme to estimate parameters $\boldsymbol{p}$ of a parametrised transform, this method is known as the *iterative closest point algorithm* (ICP). A good starting guess for the algorithm may be estimated using a principal component analysis.[3][20, chap. 8.4].

3.1.2.3 Intensities

The last registration basis is based on intensities. Here the transformed template and reference image's intensities $T(W(\boldsymbol{x}_{\mathcal{T}}, \boldsymbol{p}))$ and $R(\boldsymbol{x}_{\mathcal{R}})$ are used to quantify similarity. Now that both our images exist on the same regular grid, we can start to measure similarity. The similarity is measured point by point on the regular grid. Considering two images $\mathcal{I}_1$ and $\mathcal{I}_2$ on the same regular grid we can measure similarity in a set of N points $\boldsymbol{x}$ (e.g. a predefined mask) by comparing the intensities, $I_1(\boldsymbol{x})$ and $I_2(\boldsymbol{x})$, in these points. Among the similarity measures are sum of squared differences (SSD, eq. 3.5), correlation coefficient (CC, eq. 3.6) and normalized mutual information (NMI, eq. 3.8).

$$SSD = \frac{1}{N} \sum_{1}^{N} |I_1(\boldsymbol{x}) - I_2(\boldsymbol{x})|^2 \tag{3.5}$$

Sum of squared differences is the most simple and most intuitive of the similarity measure. It is fast to compute and it is easy to take differentiate. Furthermore, if leaving out the summation you are left with a error image which can show spatially where the largest errors may be.

$$CC = \frac{\sum_i (I_1(\boldsymbol{x}) - \mu(I_1(\boldsymbol{x})))(I_2(\boldsymbol{x}) - \mu(I_2(\boldsymbol{x})))}{\sqrt{\sum_i (I_1(\boldsymbol{x}) - \mu(I_1(\boldsymbol{x})))^2 \sum_i (I_2(\boldsymbol{x}) - \mu(I_2(\boldsymbol{x}))^2)}}, \quad \mu = \text{mean} \tag{3.6}$$

The correlation coefficient is a well known tool from statistics which apparently also has it's use as similarity measure. The summation makes it impossible to see an error image.

$$NMI = \frac{H(I_1(\boldsymbol{x})) + H(I_2(\boldsymbol{x}))}{H(I_1(\boldsymbol{x}), I_2(\boldsymbol{x}))} \tag{3.7}$$
$$H(I) = -\sum_{j,k} PDF[j,k] \log PDF[j,k]$$
$$PDF = \text{probability distribution function}$$

Normalised mutual information has it's roots in information theory. It is more complicated than two previous two, but has the advantage that it can handle images of different modality, i.e. compare a CT to a MR images. This is due to the fact that it is based on probability distribution functions. On the other hand, the probability distribution functions makes it impossible to see an error image.[20, chap. 8.5][4]

3.1.3 Minimisation Problems in Image Registration

Until know we have only talked about how to choose a transformation and how to measure similarity. The next step is to minimise the similarity measure given some transformation. The minimisation problem is dependent on the type of transformation.

3.1.3.1 Parametric

If the we are working with parametric transformation the minimisation is performed by constructing a optimization problem. This optimization problem will have similarity measure as cost function $\Delta(\cdot)$ and the transformation parameter $\boldsymbol{p}$ as optimization parameter. Mathematically this can be written:

$$\text{argmin}_{\boldsymbol{p}} \Delta(\boldsymbol{p}) \tag{3.8}$$

If the transformation is also nonrigid it is possible to add a term $\Gamma(\cdot)$ that will penalize non-smooth transformation

$$\text{argmin}_{\boldsymbol{p}} \Delta(\boldsymbol{p}) + \alpha\Gamma(\boldsymbol{p}) \tag{3.9}$$

In this case one most also find a suitable value for α, usually this can be done by the trial and error method. In the literature, we find a large variety of methods for solving optimization problems like (3.8) and (3.9). Among the most popular ones for unconstrained problems are gradient descent, Newton's method, Quasi-Newton methods, conjugate gradient method. Some of the method may also gain performance by adding a line-search technique. Constrained problem can be turned into unconstrained problems by using Lagrange multipliers [7, 15, 6].

3.1.3.2 Nonparametric

If the transformation is nonparametric the type of problem is a bit different. Since there are no parameters, the minimisation is dependent on the displacement field u.

$$\text{argmin}_{\boldsymbol{u}} \Delta(\boldsymbol{u}) + \alpha \Gamma(\boldsymbol{u}) \tag{3.10}$$

This problem can be solved by applying gradient descent based methods. A more sophisticated way of solving this problem is by using the Euler-Lagrange equations to build an iterative update rule. It is out of the scope of this thesis to go into details, so it is for the reader to consult [12, chap. 8] for thorough explanation.

3.1.3.3 Multiresolution schemes

No matter what method is chosen for the optimization problem, it will be computational heavy due to the large amount of data an image consist of. A way to speed up things and also avoid local minima in the minimization process is to use a multiresolution scheme. The idea is the following:

1. Set scaling factor between successive levels, α .
2. Set number of levels to use, N.
3. Reduce resolution to $\frac{1}{\alpha^N}$ for both template and reference image.
4. Find solution to minimization problem.
5. If $N = 0$ then stop, else $N = N - 1$ and go to 3.

Figure 3.3 shows an example using the Matlabs cameraman image. The scaling factor is $\alpha = 4$ and the number of levels are $N = 2$.

Since image registration has it's roots from many places: physics, mathematics, image analysis, optimization etc., it is easy to find specialized image registration methods that has not been covered in the previous. The goal was to give a brief overview of the most common methods.

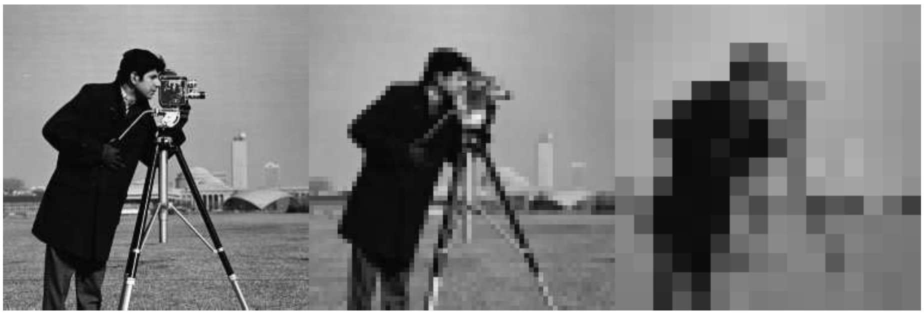
Figure 3.3: The Matlab cameraman in three different resolutions. Left: Original 256×256. Middle: Scaled to 64×64. Right: Scaled to 16×16.

3.2 Four Methods for Image Registration

Where the previous was a more general introduction to image registration, the following will describe four specific image registration methods in detail: Rigid, Thirions demons, free-form deformation and thin plate spline.

3.2.1 Rigid Transformation

The word *rigid* here refers to the possible transformation of a physical rigid body. A physical rigid body, such as a wooden cube, can only transformation or move it's mass by either a rotation or a translation. Some authors claim that the rigid transformation also should include a scaling in each dimension, but following [20], inclusion of such is considered a nonrigid transformation. Hence, the transformation is of the following type:

$$W(\boldsymbol{x}, \boldsymbol{p}) = \boldsymbol{R}\boldsymbol{x}^T + \boldsymbol{t}. \tag{3.11}$$

Two parameters are needed: A rotation matrix, $\boldsymbol{R}$, and a translation vector, $\boldsymbol{t}$. The rotation matrix is a $d \times d$-matrix with elements $R_{i,j} \in [-1; 1]$. To allow only physically possible rotations and avoid reflections, $\det(\boldsymbol{R}) = 1$. The individual elements can be traced back to d rotations around the d individual axes of the coordinate system (Euler angles). If the usual 3-dimensional Cartesian coordinate system is considered, we get:

$$\boldsymbol{R} = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix}$$
$$= \begin{bmatrix} \cos\theta_y\cos\theta_z & -\cos\theta_x\sin\theta_z+\sin\theta_x\sin\theta_y\cos\theta_z & \sin\theta_x\sin\theta_z+\cos\theta_x\sin\theta_y\cos\theta_z \\ \cos\theta_y\sin\theta_z & \cos\theta_x\cos\theta_z+\sin\theta_x\sin\theta_y\sin\theta_z & -\sin\theta_x\cos\theta_z+\cos\theta_x\sin\theta_y\sin\theta_z \\ -\sin\theta_y & \sin\theta_x\cos\theta_y & \cos\theta_x\cos\theta_y \end{bmatrix} \tag{3.12}$$

The translation vector $\boldsymbol{t}$ is a $d \times 1$ vector. It is added after the rotation (the rotation is always around origo). The number of parameters in the rigid transformation sums to $2d$.

3.2.1.1 Summary of Rigid Tranformation

The rigid transformation is characterised by the following properties:

- global transformation
- preserves distance
- preserves straight lines
- preserves angles between straight lines

Even though the choice has fell upon a nonrigid transformation, the first step is often to do a rigid transformation. This way each transformation take care of each their task: The rigid will globally align the two images, and the nonrigid will align the images on a local scale [20]. A cousine of the rigid transformation is the affine transformation. It also allows for scaling and skewing.

3.2.2 Thirion's demons

Thirions demons is a nonparametric transformation based on Maxwell's demons. Here the term *demon* refers to an instance that controls movement. In Maxwell's theory the demon would control the movement from one chamber to another, and only allow molecules with an above-average speed to pass. In the world of image registration, the demons controls the force of movement of points in the template image based on gradient information and a difference image. Consider a 1-dimensional example, see figure 3.4. Given a demon in position d, compute the gradient of the reference image $\nabla R(d)$ and the difference image $R(d)-T(d)$. When the difference image $R(d) - T(d) < 0$, the template is moved in the direction of $\nabla R(d)$, otherwise it is moved in the direction of $-\nabla R(d)$.

$$f = \nabla R(d)(R(d) - T(d)) \tag{3.13}$$

is called the force and summarizes the previous statements about the movement.

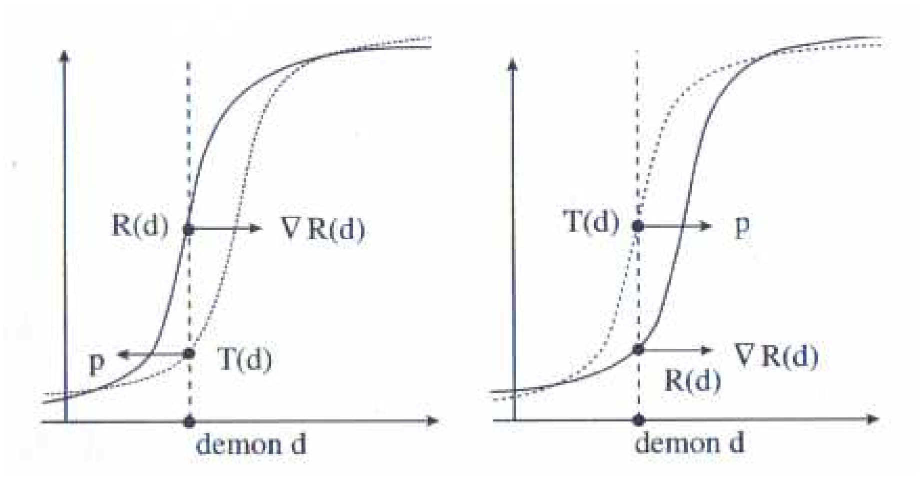

Figure 3.4: 1-dimensional example of Thirions demons. From [12]

The 1-dimensional example was a forward transformation type. If we instead want to use the inverse, the two images must be interchanged. In practice, the method works in the following iterative fashion:

1. Set scaling coefficient β
2. Set displacement field $\boldsymbol{u} = 0$.
3. Compute forces of movement, $f = \nabla T(\boldsymbol{x}_{\mathcal{T}} + \boldsymbol{u})^T (T(\boldsymbol{x}_{\mathcal{T}} + \boldsymbol{u}) - R(\boldsymbol{x}_{\mathcal{R}}))$.
4. Smooth forces with a Gaussian kernel and update the displacement field, $\boldsymbol{u} = \boldsymbol{u} - \beta(G_\sigma \star f)$.
5. If converged then stop, else go to 2.

The gradient of an image is very sensitive to noise in the image, so to ensure a regular displacement field, the forces must be smoothed. Furthermore, stable convergence is preferred so the update term is multiplied with a coefficient $0 < \beta < 1$. For faster convergence, a multiresolution scheme can be used. The approach described in the list is based on the Euler-Lagrange equations, however it is also possible to see the problem from a gradient descent perspective, see e.g. [23].

3.2.2.1 Summary of Thirion's demons

Some of the notable characteristics of Thirion's demons are:

- possibility for local transformation
- intiutively and easy to understand from a 1-dimensional example
- a basic implementation of method only includes filtering and derivatives
- multiresolution implementation is a possibility

3.2.3 Nonrigid B-spline Based Free-form Deformation

A nonrigid B-spline based free-form deformation is a parametrised, local, intensity-based image registration method. The idea is to put a mesh of control points on top of the image, see figure 3.5, and by manipulating the individual control points in the mesh, it is possible achieve a transformation that is local. Assuming three dimensions, let the domain of the image be

$$\Omega = \{(x, y, z) | 0 \leq x \leq x_{max}, 0 \leq y \leq y_{max}, 0 \leq z \leq z_{max}\} \tag{3.14}$$

where $(x_{max}, y_{max}, z_{max})$ are the dimensions of the image. Furthermore we introduce the mesh of control points Φ of size $n_x \times n_y \times n_z$. Each mesh control point $\phi_{i,j,k}$ is a position in image domain. The transformation of a given point in the image $\boldsymbol{x} = (x, y, z)$ is a tensor product of 1-dimensional basis functions.

$$W(\boldsymbol{x}, \phi) = \sum_{l=0}^{3} \sum_{m=0}^{3} \sum_{n=0}^{3} B_l(u) B_m(v) B_n(w) \phi_{i+l, j+m, k+n} \tag{3.15}$$

The indices i, j, k determine which neighbourhood of control points the given point belongs to

$$i = \lfloor \frac{x}{n_x} \rfloor - 1, \quad j = \lfloor \frac{y}{n_y} \rfloor - 1, \quad k = \lfloor \frac{z}{n_z} \rfloor - 1. \tag{3.16}$$

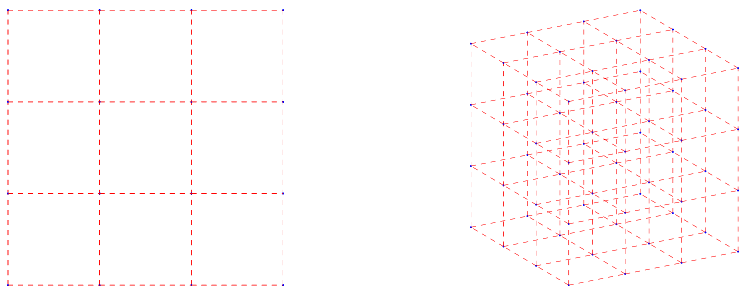

Figure 3.5: Left: Projection of a $4 \times 4 \times 4$ control point mesh onto the z-plane. Right: 3D view of a $4 \times 4 \times 4$ control point mesh.

The scalars u, v, w are spatial distances along the coordinate axes from the point $\boldsymbol{x}$ to the closest control point.

$$u = \frac{x}{n_x} - \lfloor \frac{x}{n_x} \rfloor, \quad v = \frac{y}{n_y} - \lfloor \frac{y}{n_y} \rfloor, \quad w = \frac{z}{n_z} - \lfloor \frac{z}{n_z} \rfloor. \tag{3.17}$$

The functions B are cubic B-spline basis functions. They are describe the mutual weighting between the control points given a set of distances u, v, w.

$$\begin{aligned} B_0 &= (1-u)^3/6 \\ B_1 &= (3u^3 - 6u^2 + 4)/6 \\ B_2 &= (-3u^3 + 3u^2 + 3u + 1)^3/6 \\ B_3 &= u^3/6 \end{aligned} \tag{3.18}$$

The B-splines are symmetric around the center of the mesh cell that the point belongs to, and are constructed in a way such that a transition from e.g. index i to $i + 1$ is smooth. Furthermore, the transformation W is constructed in a way such that a change in a single control point only affects the neighbourhood around it. For the same reason this method is not efficient for globally aligning and should be preceded by an rigid (or affine) transformation.
The parameters enter into the transformation as the spatial position of the control points Φ, that is, d parameters for each control point in d dimensions. If there are N_p control points, then there will be $d \cdot N_p$ parameters that control the transformation. Even with the simplest similarity measure of them all, sum of squared differences, as cost function $C(\Phi)$, it not possible to make a closed form solution for the control points that solves the minimization problem. Therefore, it necessary to utilize an iterative gradient descent technique to minimize $C(\Phi)$. This can be summarized to:

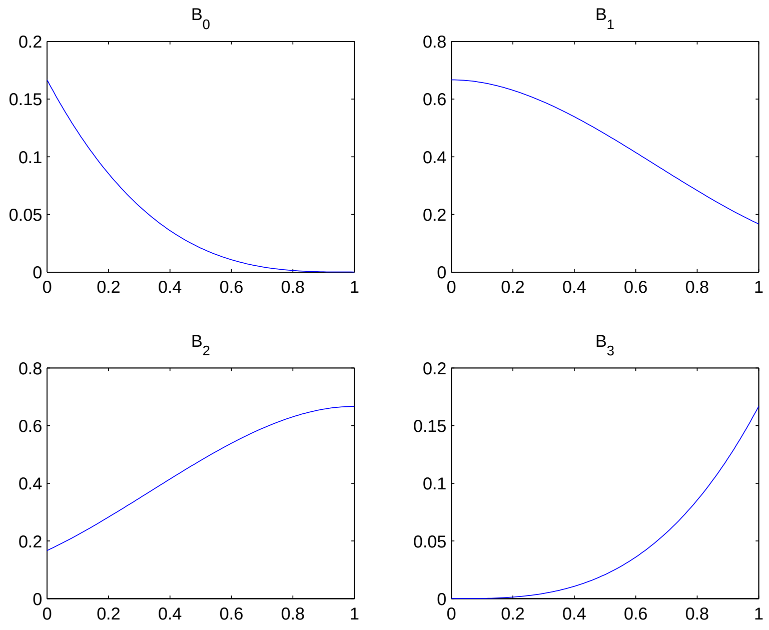

Figure 3.6: The cubic B-spline basis functions.

1. perform a rigid or affine registration.
2. initialize control points Φ
3. calculate the gradient of the cost function wrt. Φ, $\nabla C(\Phi) = \frac{\partial C(\Phi)}{\partial \Phi}$
4. update control points $\Phi = \Phi + \mu \frac{\nabla C(\Phi)}{||\nabla C(\Phi)||}$
5. if converged then stop, else go to 3

It is also possible to incorporate a multiresolution scheme, and it is also possible to use multilevel scheme for the control point mesh. The idea is to first catch large variations in the deformation using a large control point spacing, and then gradually subdivide the mesh, until the desired control point spacing is reached. This will speed up convergence of the algorithm. A smoothness term can also be added to the cost function to ensure a deformation field, but it is not always necessary since the smoothness of the B-spline basis functions help a lot[18, 19].

3.2.3.1 Summary of Nonrigid B-spline Based Free-form Deformation

Some of the notable characteristics of nonrigid B-spline based free-form deformations are:

- possibility for local transformation
- the possibility of multilevel control point mesh and multiresolution makes it very flexible
- computational heavy, even with the added speed improvements

3.2.4 Thin Plate Spline

Thin plate splines registration is a parametric, nonrigid, landmark-based registration method. Thin plate spline were originally long flexible strips of metal used in drawing offices for drawing curved lines. They were placed on top of the sheet of paper and adjusted using a set of fixpoints along the spline. Producing the splines in types of metal or alloys with different stiffness (bending energy) it was possible to control the flexibility. In the same way we want to adjust the transformation using a set landmarks.
Considering a 2-dimensional case with two sets of landmarks, template $\boldsymbol{\tau} = (\tau_x, \tau_y)$ landmarks and reference landmarks $\boldsymbol{\rho} = (\rho_x, \rho_y)$ of length N, we seek the transformation $W(\boldsymbol{\tau})$ that solves

$$\underset{W}{min} \frac{1}{N} \sum_{i=1}^{N} \{\boldsymbol{\rho} - W(\boldsymbol{\tau})\}^2 + \lambda J(W) \tag{3.19}$$

where $J(W)$ is a function that expresses the curvature (bending energy) of W

$$J(W) = \int \int_{\mathbb{R}^{\not\in}} \Big(\frac{\partial^2 W}{\partial \tau_x^2}\Big) + 2\Big(\frac{\partial^2 W}{\partial \tau_x \tau_y}\Big) + \Big(\frac{\partial^2 W}{\partial \tau_y^2}\Big) d\tau_x d\tau_y \tag{3.20}$$

and λ is a parameter for controlling the curvature penality. The higher value of λ, the smoother the solution. In the limit $\lambda = \inf$, the solution will be a straight line with no curvature. Setting $\lambda = 0$ gives a solution that passes exactly through each reference landmark. Notice that equation (3.21) in some sense is a expansion of equation (3.2) for FRE^2 to include a smoothness term.

The mathematical nature of $W(\boldsymbol{\tau})$ is twofold. It has both a linear term and a radial basis function term

$$W(\boldsymbol{\tau}) = a_1 + a_2\tau_x + a_3\tau_y + \sum_{i=0}^{N} w_i U(|\boldsymbol{\tau} - \boldsymbol{\rho}_i|) \tag{3.21}$$

where w_i are weights and

$$U(r) = \begin{cases} r^2 \log r^2 & r > 0 \\ 0 & r = 0 \end{cases} \tag{3.22}$$

This is an underdetermined system since there are $3 + N$ unknown and N equations, but since the bending energy $J(W)$ must be zero for a straight line as input, we must add three constraints

$$\sum_{i=0}^{N} w_i = 0, \quad \sum_{i=0}^{N} w_i \rho_x = \sum_{i=0}^{N} w_i \rho_y = 0. \tag{3.23}$$

Instead of setting up a two seperate systems, one for the x-coordinates and one for the y-coordinates, we set up one system for both coordinates based on equation (3.19) and the constraints (3.23):

$$\begin{bmatrix} \boldsymbol{A} + \lambda \boldsymbol{I} & \boldsymbol{P} \\ \boldsymbol{P}^T & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_x & \boldsymbol{w}_y \\ \boldsymbol{a}_x & \boldsymbol{a}_y \end{bmatrix} = \begin{bmatrix} \rho_x & \rho_x \\ \boldsymbol{z} & \boldsymbol{z} \end{bmatrix} \tag{3.24}$$

where $\boldsymbol{A}_{i,j} = U(|\boldsymbol{\tau}_i - \boldsymbol{\rho}_j|)$, $\boldsymbol{P}_{i,:} = (1, \boldsymbol{\rho}_i)$ and $\boldsymbol{Z}$ and $\boldsymbol{z}$ are 3×3 and 3×1 zero matrices respectively. $\boldsymbol{I}$ is the $N \times N$ identity matrix, λ is a scalar, $\boldsymbol{w}_x$ and $\boldsymbol{w}_y$ are $N \times 1$ and $\boldsymbol{a}_x$ and $\boldsymbol{a}_y$ are 3×1. Solving this system linear for the unknown weights and affine paramters, we get:

$$\begin{bmatrix} \boldsymbol{w}_x & \boldsymbol{w}_y \\ \boldsymbol{a}_x & \boldsymbol{a}_y \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} + \lambda \boldsymbol{I} & \boldsymbol{P} \\ \boldsymbol{P}^T & \boldsymbol{Z} \end{bmatrix}^{-1} \begin{bmatrix} \rho_x & \rho_x \\ \boldsymbol{z} & \boldsymbol{z} \end{bmatrix} = K^{-1} \begin{bmatrix} \rho_x & \rho_x \\ \boldsymbol{z} & \boldsymbol{z} \end{bmatrix} \tag{3.25}$$

Knowing $\boldsymbol{w}_x, \boldsymbol{w}_y, \boldsymbol{a}_x, \boldsymbol{a}_y$ for a given set of template and reference landmarks, we can easily compute the transformation for any point, since we are dealing with a linear system. A point in the coordinate system of the template (t_x, t_y) can be transformed to the coordinate system of the reference image by

$$\begin{bmatrix} r_x & r_y \end{bmatrix} = \begin{bmatrix} B & 1 & t_x & t_y \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_x & \boldsymbol{w}_y \\ \boldsymbol{a}_x & \boldsymbol{a}_y \end{bmatrix} \tag{3.26}$$

where $B_j = U(||(t_x, t_y) - \boldsymbol{\rho}_j||)$. The fact, that it is a linear system makes fast and easy to handle in Matlab [24].

3.2.4.1 Landmark extraction

In order to apply the thin plate spline transformation you need to have some landmarks in both the template and the reference image. The template image will be a slice of the belly product and the reference image will be the corresponding slice of the unprocessed middle. The landmarks will be extracted from

the interface between air and skin, and the interface between the subcutaneous fat layer and the first meat layer, see figure 3.7.
In some sense, this is just a surface extraction. An automatic method for finding the interfaces could probably easily be developed since both interfaces have a very high intensity gradient. Edge filtering and dynamic programming would maybe solve it. But for testing, we annotate the interfaces manually. Assuming these two interfaces to be flexible but unstretchable, we can place landmarks with a predefined distance along the two interfaces in both images, see figure 3.7. The procedure is:

1. annotate the interfaces manually
2. interpolate the annotated interfaces with a cubic spline
3. start in the far right side of the interfaces and place landmarks with a regular distance d
4. do it for both template and reference

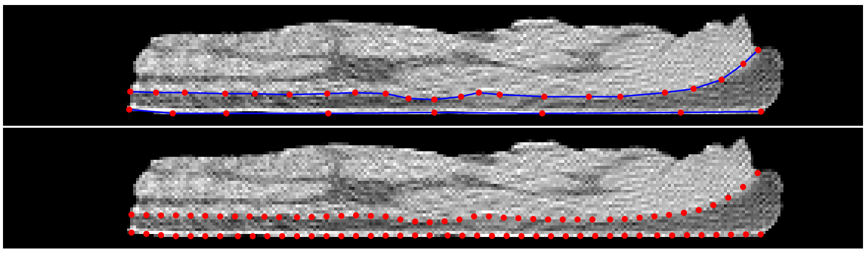

Figure 3.7: Top: Annotated interfaces of the belly and the whole middle. Bottom: Extracted landmarks of the belly and the whole middle for a landmark distance $d = 5px$.

The landmark extraction depends, in the manual case, of course on how well the interfaces been annotated. The more precise and the more close the annotations are the better.

3.3 Results

The three of the four image registration studied in the previous section are applied to the available data. 2-dimensional examples are studied, to keep it as simple as possible. The methods are applied to the belly product only, since this piece from manual inspection seems to be the simplest to do registration on. In case of succes, it can always be applied to the harder back product.
For Thirion's demons and the free-form deformation, the rigid transform is used as a first step.

3.3.1 Thirion's Demons

An implementation of Thirion's demons was made on basis of a collection of image registration Matlab methods implemented by Jan Moderzitski, University of Lubeck [11]. Among these were good methods for interpolation and taking derivatives of images. The implementation includes a multiresolution scheme. This is needed to "bend" the product enough to match the whole middle. The first step is always a rigid registrastion. Below are shown registration of two different slices (middle 2557, slice 55 and 35).

3.3.1.1 Slice 55

To see how the multiresolution scheme works with Thirion's demons, a serie of images have been produced to show the registration result just before a step up in resolution is taken, see figures 3.8 and 3.9. The used parameters are:

Resolution	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$	1
σ of the Gaussian	50	30	15	15
update coefficient β	0.0020	0.0010	0.0005	0.0003
No. iterations	50	200	200	400

The parameters were found by the trial-and-error method. It is difficult to say anything about the parameters apriori, since they vary with problem type and image. No automatic stopping criteria was used, instead the number of iterations was set high.
As the images show, the template to downscaled $\frac{1}{8}$ resolution, does not change much, probably because the resolution redction has blurred out even the biggest structures in the image. It is not until $\frac{1}{4}$ resolution that registration really begins

to take place. Especially at $\frac{1}{4}$ and $\frac{1}{2}$ resolution a change can be seen. Figure 3.10 shows a color image with both reference and template image. Here it is clear that the registration has found matches between the subcutaneuos fat layers, but alse stretched the image to much in the top. The same thing is seen from the difference image, see figure 3.11. This could be due to the large intensity difference between the air background and the product. The standard devitaion of the difference image inside the ROI is 101.10 Hu.

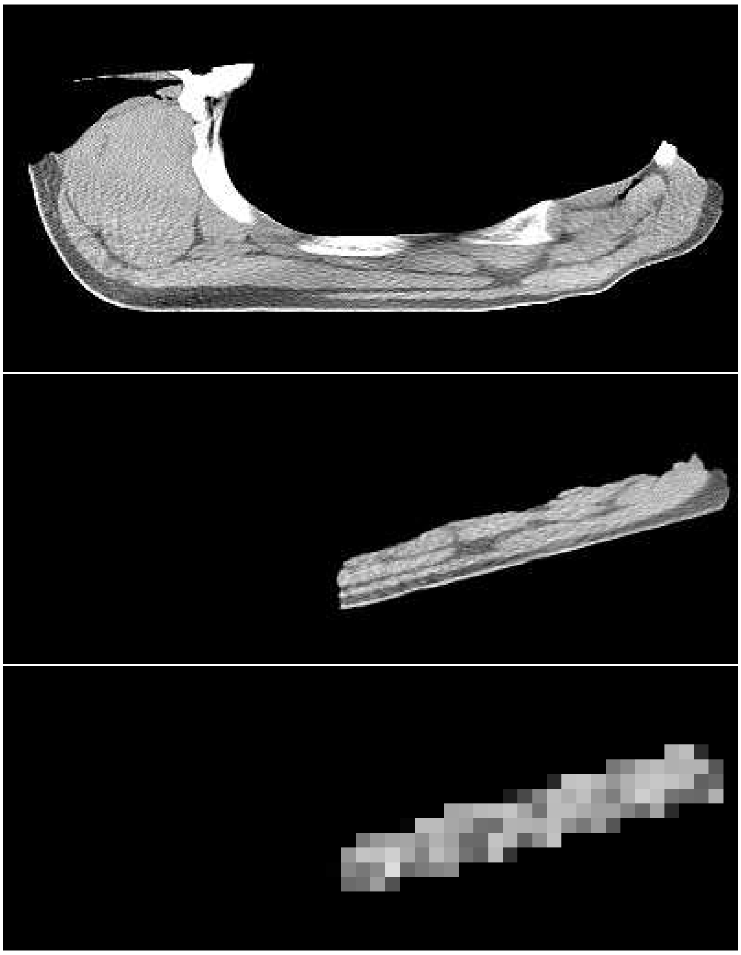

Figure 3.8: Top: Reference image. Middle: Template, rigid registered. Bottom: Template downscaled $\frac{1}{8}$ resolution.

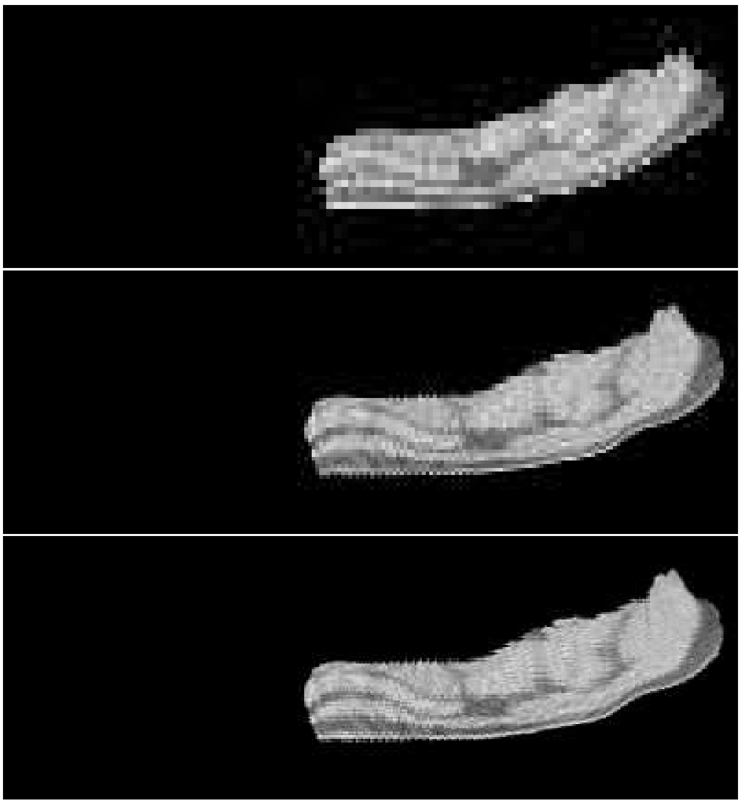

Figure 3.9: Top: Template downscaled $\frac{1}{4}$ resolution. Middle: Template downscaled $\frac{1}{2}$ resolution. Bottom: Template in full resolution, fully registered.

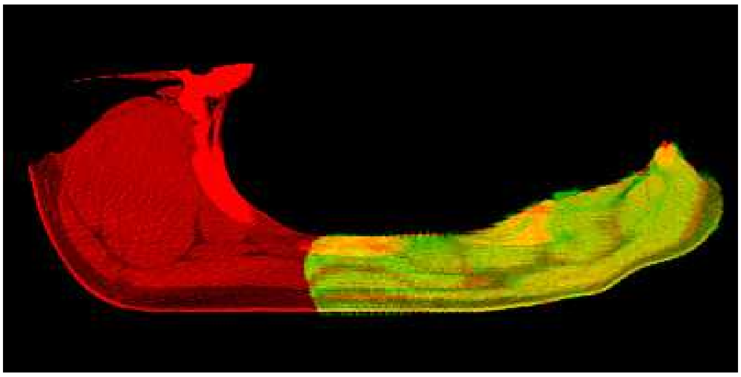

Figure 3.10: Red color is refence, green is template. The template has been stretched to much by the registration method, it is not volume preserving.

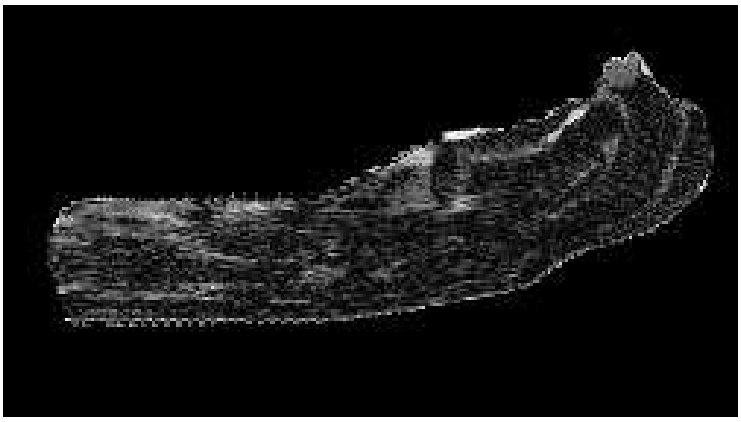

Figure 3.11: Difference image inside the ROI of the template. Large areas of high errors occur in the top of the image where it has been stretched to much.

3.3.1.2 Slice 35

Another slice is testet with the same set of parameters as before. The SSD was monitored to study how it evolves, see figure 3.12. The first thing one notices, is the jumps in the SSD. These jumps occur when a step up in resolution is taken. The reduction in resolution acts as a mean filter so, this is what causes the jumps. Also we can see that the convergence is fastest in the begining, just like we aimed for with the multiresolution scheme.

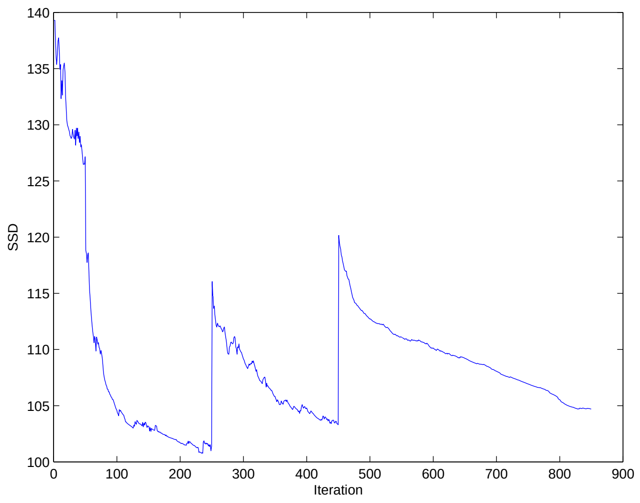

Figure 3.12: SSD versus iteration for Thirion's demons.

Template, reference and result is seen in figure, 3.13. Figure 3.14 shows a color image with both reference and template image. The registration has not found as good a match as before. It has some problem with the bumps in the skin and again it has stretched to much so that the template covers some of the bones. The difference image also shows the problems with bumps in the skin, see figure 3.15. The standard devitaion of the difference image inside the ROI is 104.69 Hu.

3.3.1.3 Summary

The results of the image registration using Thirion's demons shows us that we do not get a satisfactory result. The is no control of the volume, so it grows

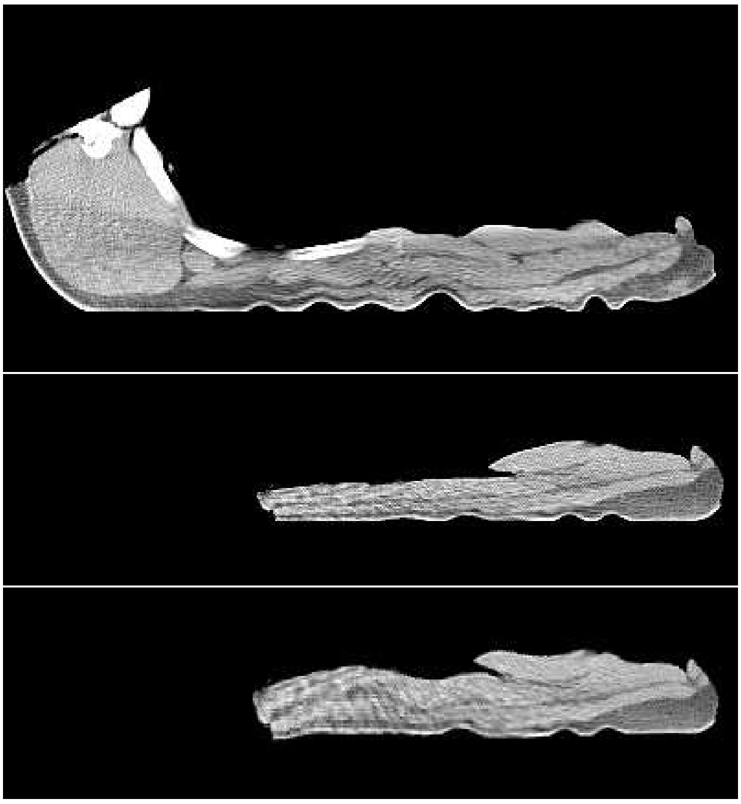

Figure 3.13: Top: Reference image. Middle: Template, rigid registered. Bottom: Template downscaled $\frac{1}{8}$ resolution.

beyond what is reasonable. The multiresolution scheme speeds things up and ensures the transformation catches the "bend" in the reference image, but it is probably also cause of the uncontrolable volume.

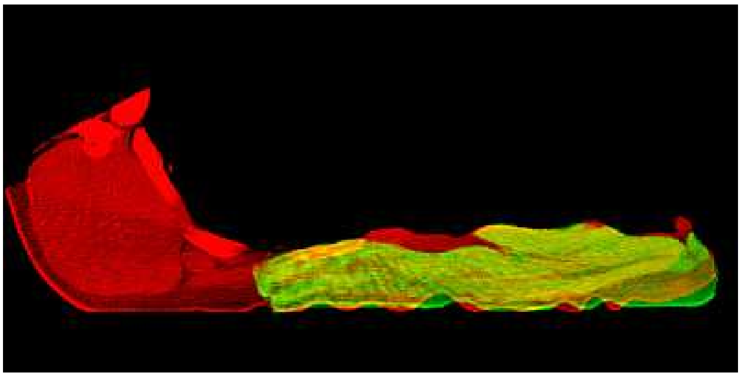

Figure 3.14: Red color is refence, green is template. Again, the template has been stretched to much by the registration method.

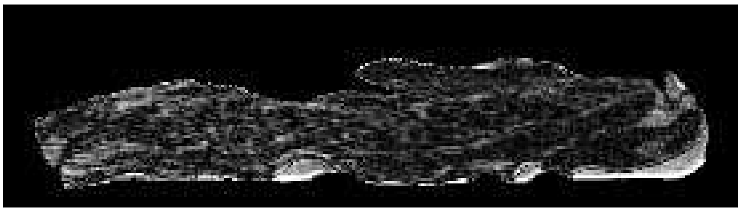

Figure 3.15: Difference image inside the ROI of the template. Again, large areas of high errors occur in the top of the image where it has been stretched to much.

3.3.2 Nonrigid B-Spline Based Free-Form Deformation

For the free-form deformation method, an image registration toolkit called ITK (Image Registration Toolkit, The Image Registration Toolkit was used under Licence from Ixico Ltd. Additionally) written in C++ was used. Again, we will use slice 35 and 55 from middle 2557. A multilevel scheme is used for the control point mesh. The coarsest level will be 40×40, followed by a 20×20 and eventually a 10×10. An image was produced for each converged multilevel registration, so it was possible to see intermediate results. The step size was set to $\mu = 5$.

3.3.2.1 Slice 55

Figure 3.16 shows results of the registration. It is clear, that even after the 40×40 registration, the image has been stretched too much. It gets even worse with the 20×20 and 10×10. From these, it is impossible to see that it should be a piece of meat. Figure 3.17 shows a plot of the SSD versus the mesh size. We can see that the SSD decrease like it should, but apparently the decrease is created by enlarging the volume (that is, incresing N in eq. 3.5). This shows that the SSD alone might not be the best way to measure similarity in this case. The final value of SSD is 88.41 Hu.

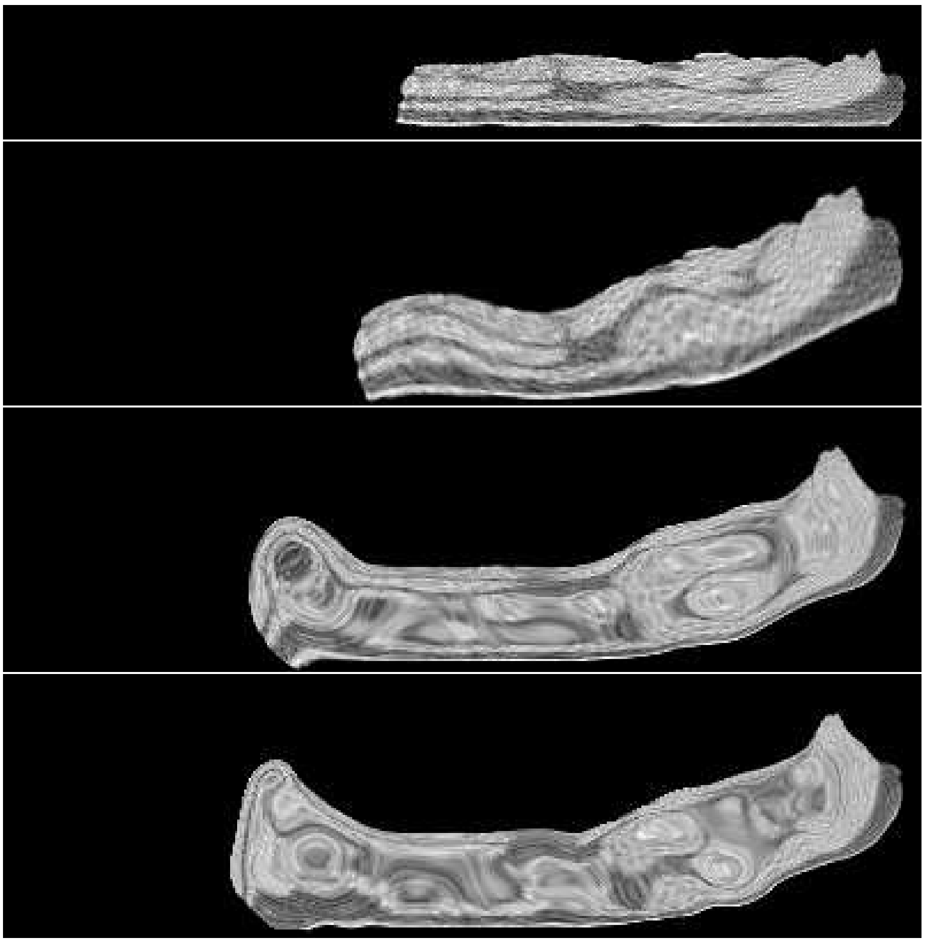

Figure 3.16: From the top: Rigid registered. 40×40 mesh. 20×20 mesh. 10×10 mesh.

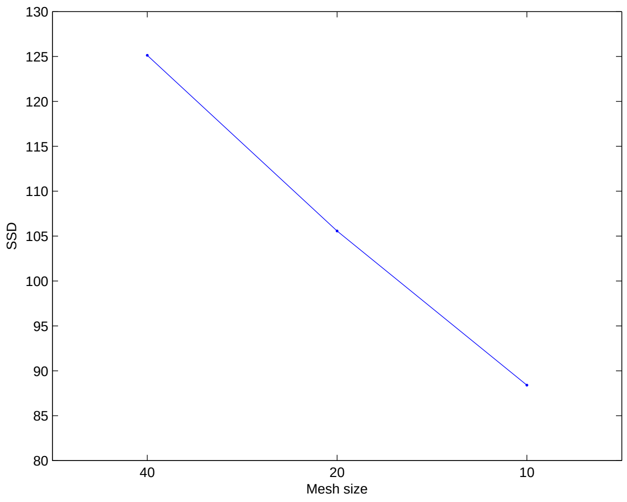

Figure 3.17: SSD versus mesh size for the registration.

3.3.2.2 Slice 35

As with slice 55, we see an slightly stretched 40×40 image followed by two bad images, 20×20 and 10×10 in figure 3.18. Again the SSD plot in figure 3.19 shows that the SSD is decreased. The final value of SSD is 90.86 Hu.

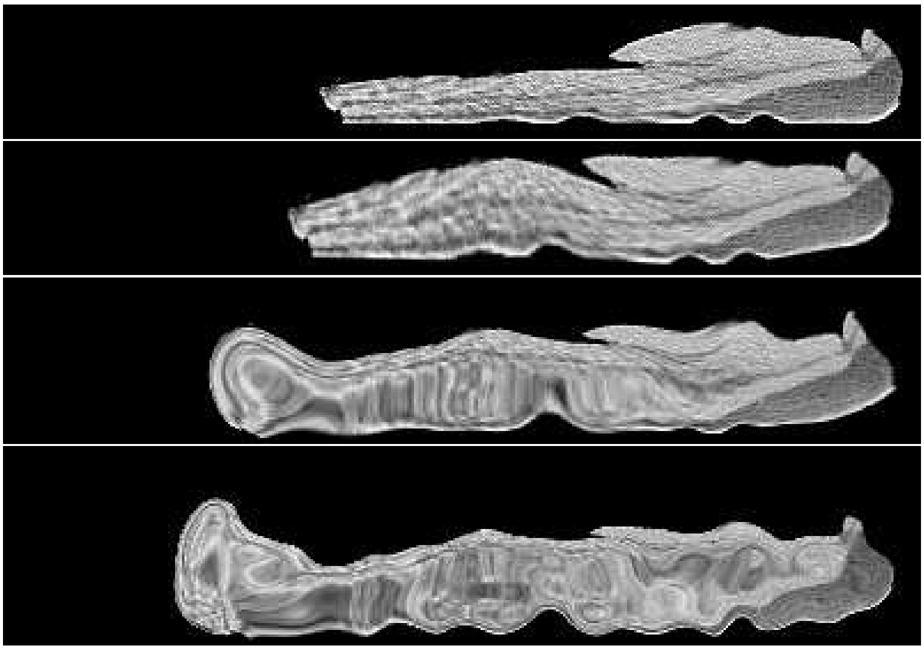

Figure 3.18: From the top: Rigid registered. 40×40 mesh. 20×20 mesh. 10×10 mesh.

3.3.2.3 Summary

These results show that the nonrigid B-Spline based free-form deformation is not good for registration of products to a whole middle. The missing of control with the volume makes it possible for the registration method to deform the image so much that it is physical impossible. This is partly due to the way the SSD similarity measure is computed.

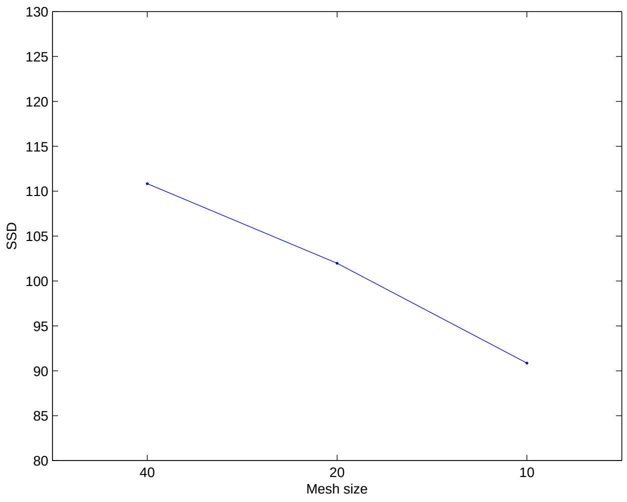

Figure 3.19: SSD versus mesh size for the registration.

3.3.3 Thin Plate Spline

A thin plate spline method as described in the theory section was implemented in Matlab. The interface annotation was done by hand to avoid misannotation. Again slice 35 and 55 from middle 2557 was used.

3.3.3.1 Slice 55

The image were annotated and the interfaces found, see figure 3.20. The distance between landmarks were set to $d = 5$, see figure 3.21. The regularization parameter was set to $\lambda = 0$, i.e. completely flexible. The result of the transformation in figure 3.22 shows promising, it is not stretched in any way at least. The difference image in figure 3.23, does not show much, but at least it tells us that there are no large areas with high error, i.e. bone or air. The SSD is 51.01 Hu. Altogether, a realistic transformation for this particular slice.

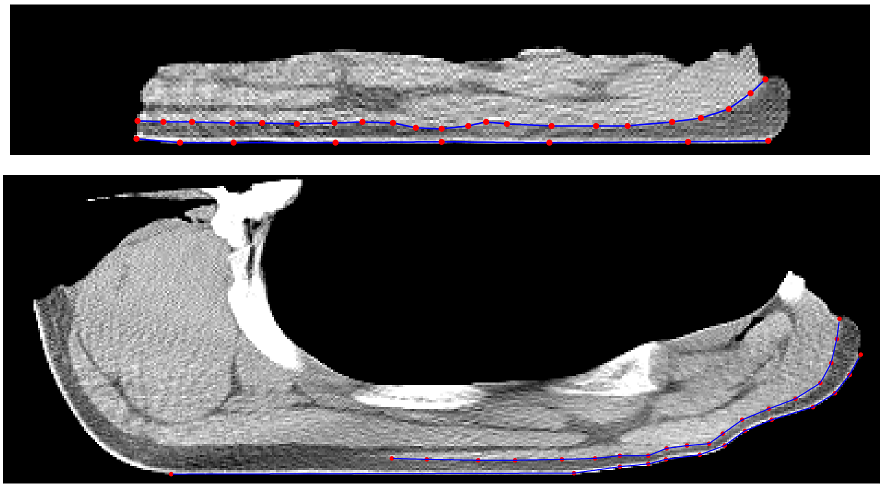

Figure 3.20: Annotated interfaces of the belly and the whole middle. ● are manually chosen points. — are the interfaces.

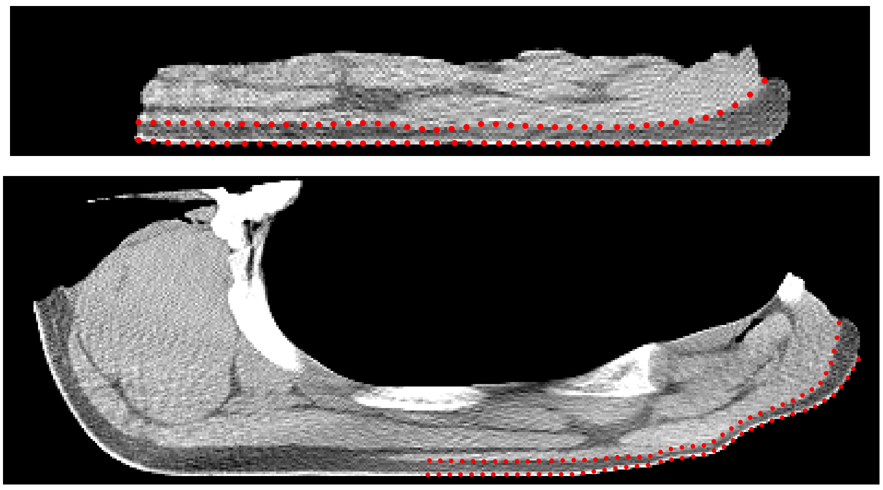

Figure 3.21: Extracted landmarks of the belly and the whole middle.

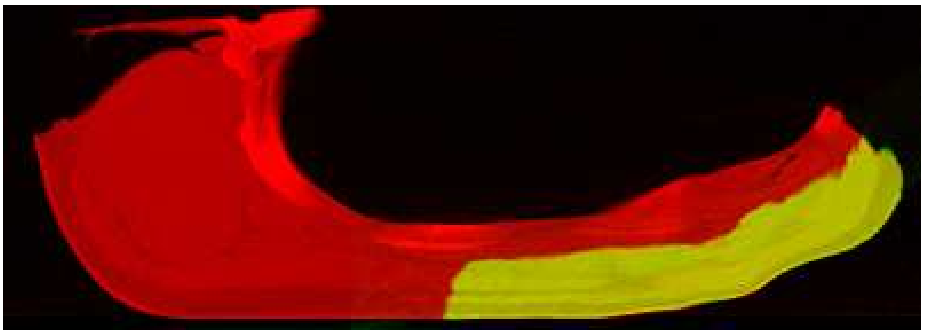

Figure 3.22: Color image showing whole middle with the registered slice 55 on top.

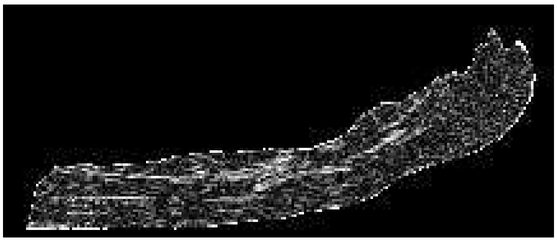

Figure 3.23: Difference image of thin plate spline transfomation of slice 55.

3.3.3.2 Slice 35

Again annotated were made, see figure 3.24. The distance between landmarks were set to $d = 5$, see figure 3.25. The regularization parameter was again also set to $\lambda = 0$. The result in figure 3.26 is not as good as that of slice 55. There are some error at the right end and at the left end of the belly. The left end of the belly is simple skewed to much towards the left, the knife in the machine would never cut this way. The difference image in figure 3.27, also shows that the errors are higher in the left end, indicating a bad match. The SSD is 65.40 Hu.
The reason for the skewed left end should probably be found in the distance between the two interfaces. The further away the two interfaces are from each other, the less impact noise will have. In fact, what is done here is comparable to extrapolation. So the farther away a point is from the data and the more noise on the data, the worse an extrapolation.

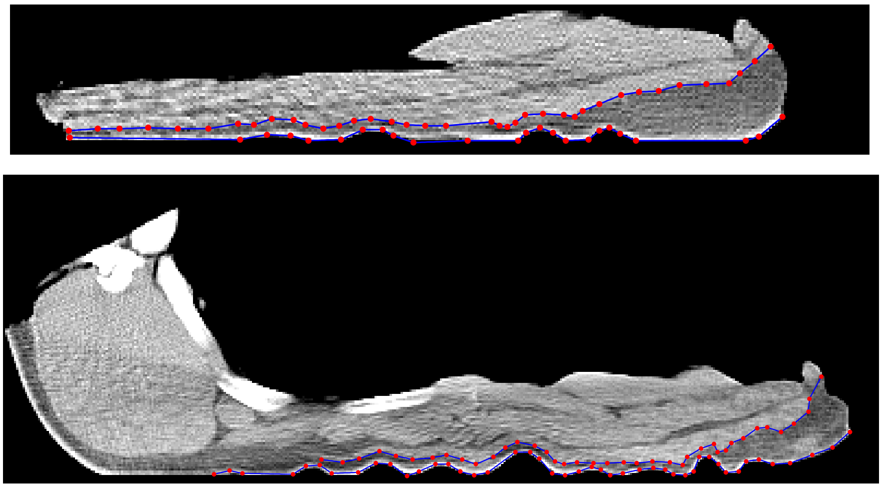

Figure 3.24: Annotated interfaces of the belly and the whole middle. ● are manually chosen points. — are the interfaces.

3.3.3.3 Summary

The two results shown here with the thin plate spline have shown to have the lowest SSD of the three methods. Slice 55 gave a really good result and slice 35

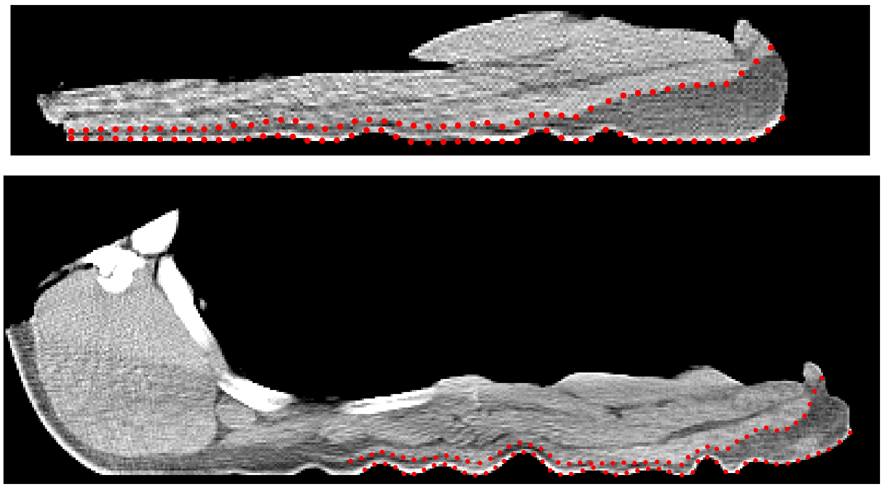
Figure 3.25: Extracted landmarks of the belly and the whole middle.

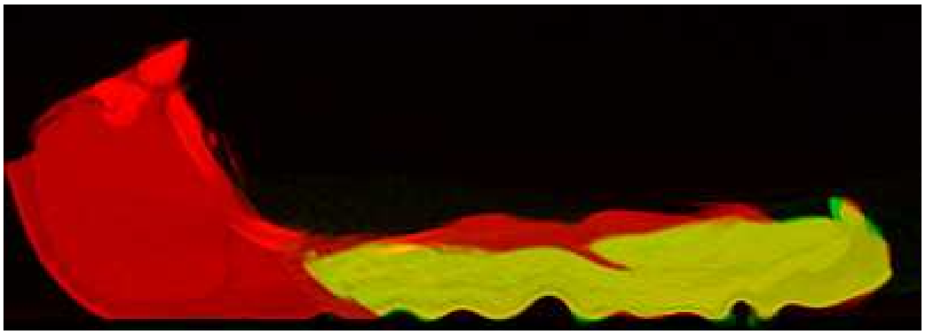
Figure 3.26: Color image showing whole middle with the registered slice 35 on top.

a less good result. Studies of the slices showed that the two interfaces must be far away from one another, to get a resonable result, but unfortunately only few of the slices in the whole pig middle has this property, so using the thin plate spline is not a possibility.

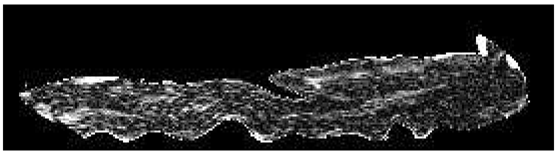

Figure 3.27: Difference image of thin plate spline transfomation of slice 35.

Chapter 4

Finding Cuttings using Ad hoc Methods

The previous section showed that neither of the three studied image registration methods could find the cuttings in the pig middle. To analyse the data further some ad hoc methods must be developed to find the cuttings.

4.1 Description of the Ad Hoc Methods

This section describes the ad hoc methods used to find the cuttings.

4.1.1 Spine

The cut off spine is a long triangular shaped object, see figure 4.2 top and 4.1. Figure 4.1 shows a projection of the spine onto the yz-plane, the resolution in the z-direction is clearly lower. The figure also shows the slight curvedness that the spine possesses. The curvedness and the low resolution in the z-direction rules out the possibility of using image registration. Instead, we can represent the spine by model and transfer this to the pig middle.

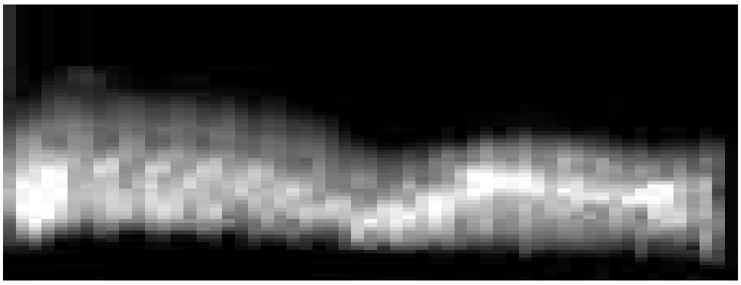

Figure 4.1: Projection of the spine.

4.1.1.1 Triangular Spine Model

As figure 4.2 shows, the spine is almost triangular but also that there is a large variation in the size of the spine segments along the spine. Since the shape of the spine is almost a triangle, we can represent the spine with a model of a triangle. The model has six parameters, two for each corner of the model triangle. The fit is found using a Nelder-Mead Simplex Search [13] in shape of Matlabs `fminsearch`. Given a set of model triangle corners, the cost to minimize is constructed in the following way:

1. find the mean value of the spine in the image $\mathcal{I}_{Data}$ by averaging all pixel intensities above -500
2. initialize the model image $\mathcal{I}_{Model}$ to have the same size as the spine image and fill with -1000 (air background)
3. in $\mathcal{I}_{Model}$, set pixels inside the model triangle to the mean value of the spine
4. compute the cost as the SSD of $\mathcal{I}_{Model}$ and $\mathcal{I}_{Data}$

We have no ground truth to use for evaluation of the fit, but visual inspection shows that the results in general are reasonable and can be used for further analysis.

4.1.1.2 Spine Matching

Now that the cut off spine has been found, it should be transfered to the whole pig middle. The slice correspondance is wasy to find since both the cut off spine

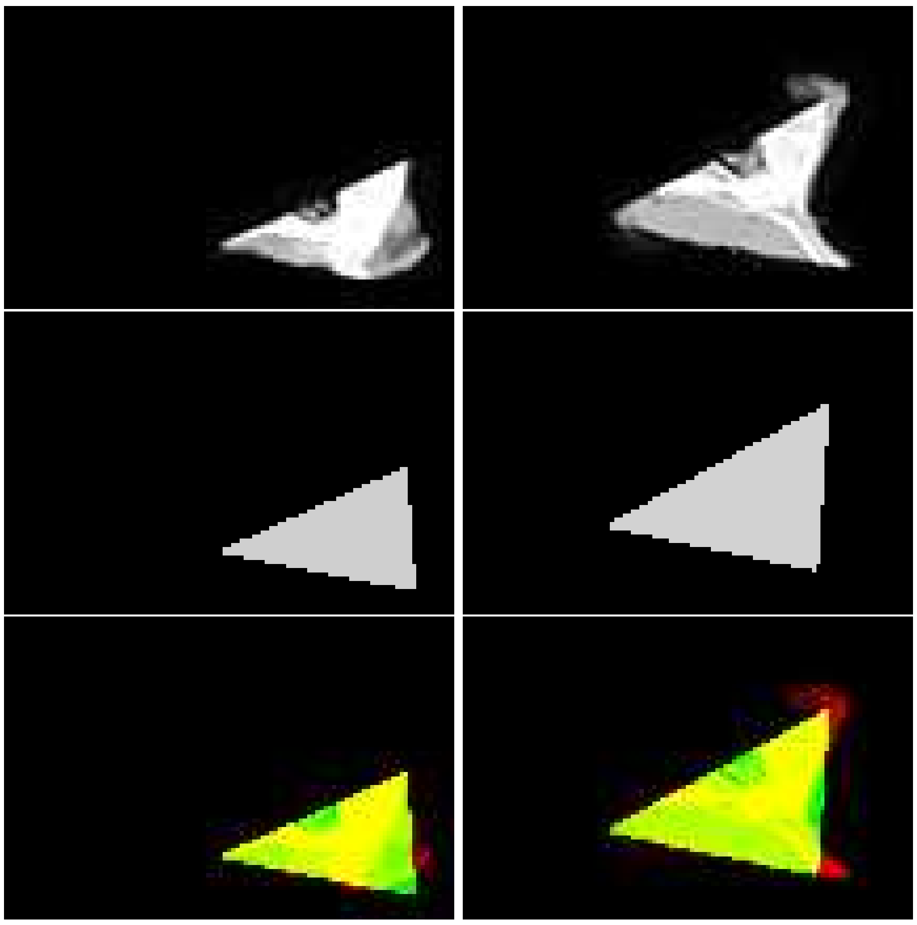

Figure 4.2: Left: Slice 8. Right: Slice 30. Top: Spine. Middle: Fittet model of spine. Bottom: Model on top of the spine.

and the whole pig middle are straight. To transfer the spine, the top point of the pig middle must be found, see figure 4.3 left. First we notice the splitting line (blue curve in figure 4.3 right), where the carcass was cut in two, is very close to being a straight line. The top point is on this line, so if we can find the line, can also find the top point. The procedure is:

1. threshold the image at 200
2. take the vertical derivative of the thresholded image, see figure 4.3 middle

3. do a Hough transform to get a slope/intersection-description of the line candidates in the image

4. remove line candidates with very negative slope

5. select line candidate with the most votes from the Hough transform

6. search in direction of the line

7. the top point is the first pixel with background intensity, see figure 4.3 right

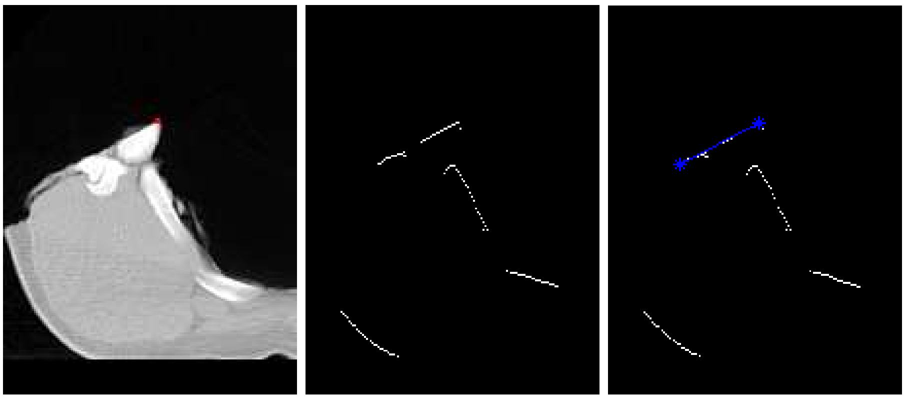

Figure 4.3: Left: ● marks the to point. Middle: Edge map of thresholded image. Right: Splitting line found by Hough tranform. The point to the right is the top point.

Now the top point has been found, the spine model can be transfered to the image of the whole pig middle. Assuming the correspondence between slices is correct, we can for each slice in the pig middle find the corresponding three points in the spine model. Following notation of figure 4.4, matching of the cut and the uncut spine to each other is done in the following manner:

1. P1, Q1, Q2, Q3 and the splitting line of the pig middle is known

2. place the top points P1 and Q1 of the two spines on top of each other

3. rotate the spine model so Q2 is on the splitting line, this is P2

4. project the Q3 along the line Q2-Q3 onto the perimeter of the pig middle, this is P3

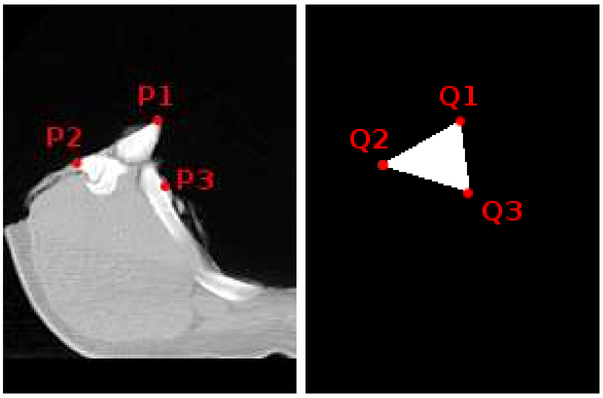

Figure 4.4: Left: Pig middle with point notation. Right: Spine model with point notation.

The projection along the line Q2-Q3 to find P3 is done to assure that the cutting point exist on the perimeter of the middle. Especially the points P2 and P3 are interesting, since they define the cutting line for the spine, one of the things we are looking for.

4.1.2 Ribs

The cut off ribs is flat and smoothly bended structure, see figure 4.5. The resolution in the z-direction is low, as with the spine, making it difficult to tell one rib from the other. To find out where the ribs were cut off, a couple of ideas was tried out before the one eventually used. The first idea was that since the ribs are a rigid structure, it would be possible to rigid register the rib surface of the cut off ribs on the rib surface of the whole pig middle using iterative closest point (ICP). Apparently, the ribs are not a rigid structure as assumed. It turns out that cutting off the ribs and placing them in the scanner causes some deformation. This is because there is meat between each of the rigid ribs, which of course is soft and therefore flexible.
Next idea was to do a connected component analysis of the ribs to match ribs one by one. Unfortunately, the resolution in the z-direction is so low that is it makes it very difficult to make an algorithm that can seperate out the individual ribs. Since we have no method for finding correspondence between the cut off ribs and the ribs on the pig middle, we instead try to measure the rib length slice by slice.

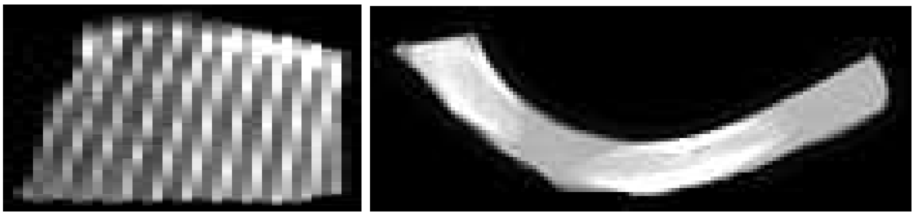
Figure 4.5: Left: Ribs seen from top. Right: Ribs seen from side

4.1.2.1 Measuring Rib Length

This will describe a method for measuring rib length. The procedure is:

1. threshold the image at -500
2. find points in the thresholded image
3. fit circle to points in iterative way using Nelder-Mead Simplex Search:
   (a) given center (c_x, c_y), radius r and image points (x, y)
   (b) compute cost as $\sum(|(x, y) - (c_x, c_y)| - r)^2$
   (c) take step according to Nelder-Mead
   (d) if converged then stop, else go to 2
4. image points to polar coordinates with (c_x, c_y) as center
5. fit polynomial to image points in polar coordinates
6. sample along the polynomial and find end points of rib
7. go from polar to cartesian coordinates
8. measure distance between sample points and sum to get the rib length

Figure 4.6 shows the circle estimated for the rib shown in figure 4.5 right. Since most of the ribs naturally has a shape close to the on of a circle arc, circle fitting is an easy way to straighten it out. The straightent out rib (figure 4.7) will of course still have some curvature in the polar domain, but much less. The use of a polynomial to find the medial axis of the rib ensures a smooth medial axis, and turns out to be a very robust way of doing it compared to e.g. thinning or skeletonization, which are prone to error if there are spatial irregularities. Figure 4.8 shows the found medial axis for the rib including end points. Figure

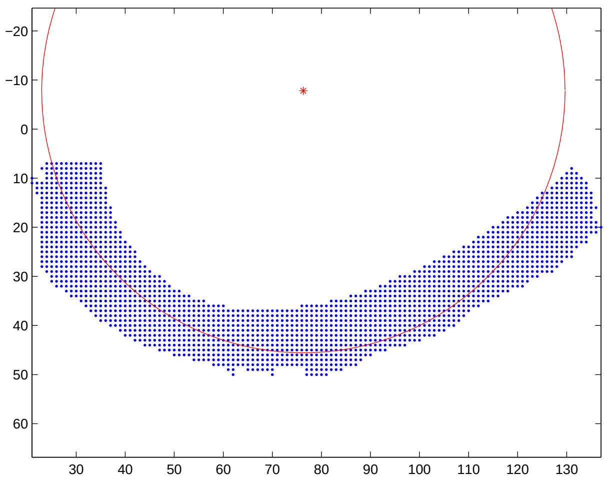

Figure 4.6: Image points fittet with a circle.

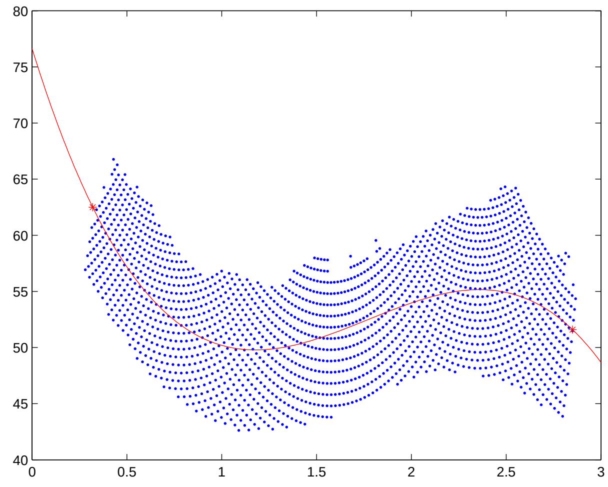

Figure 4.7: Polar transformed image points fittet with a polynomial.

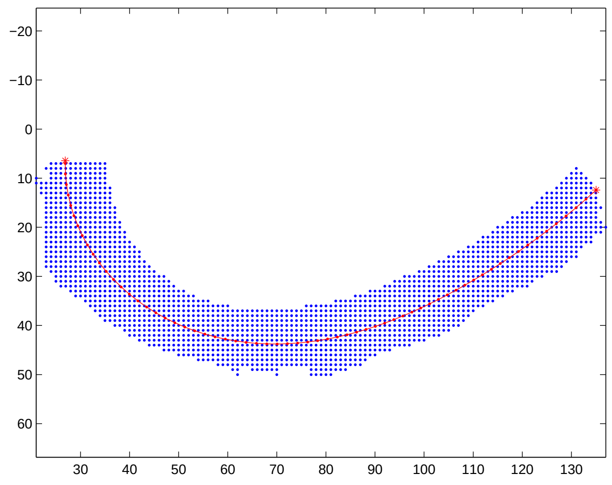

Figure 4.8: Image points with measuring path and end points plottet on top.

4.9 is a plot of the ribs in figure 4.5. One should be carefull when deducing anything about the rib length directly from figure 4.5 since this is a projection onto a plane and therefore removes some of the curvature information.

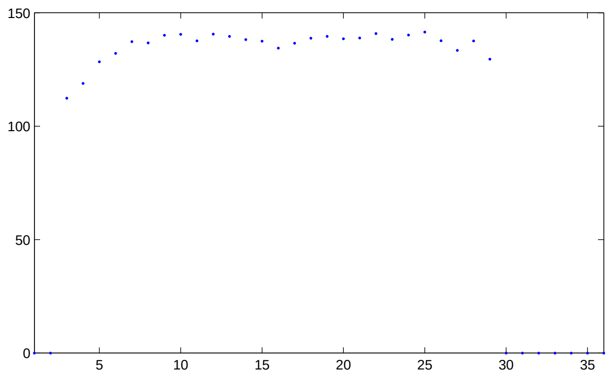

Figure 4.9: Measured length of the cut off ribs from middle 2557.

4.1.2.2 Rib Cutting

Having measured the length of the cut off ribs, these can now transfered onto the whole pig middle to find where the ribs are cut off. Given a rib length L_{rib} at a specific slice on the whole pig middle we find the rib cutting point in the following way:

1. find the spine cutting point P3, which defines where the spine and the ribs are seperated

2. from a point above the pig middle, shot out radial lines with a distance 0.05 radians, see figure 4.10

3. mark where every line hits the surface of the pig middle

4. measure distance between marks and sum up until equal to the given rib length L_{rib}

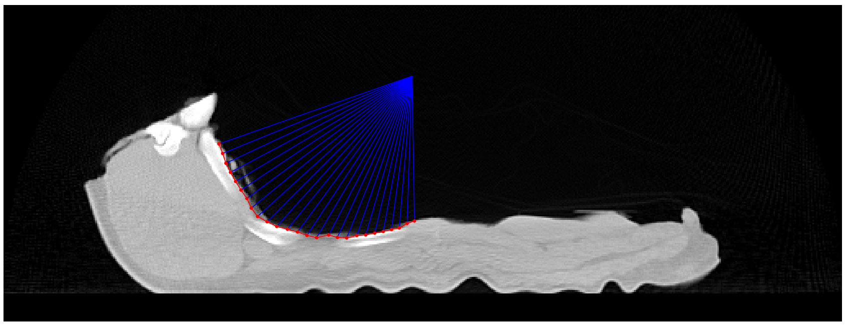

Figure 4.10: Illustration of rib cutting process.

As mentioned earlier, it is not possible to find corresondence between the cut off ribs and the ribs on the whole pig middle. The results in section 4.2 show that the individual ribs of the cut off ribs almost have equal length. For this reason, the only thing we need to find an approximate cut of the ribs on the pig middle, is the point P3 and the mean rib length of the corresponding cut off ribs. Note that the cut of the ribs is only defined where the ribs are present, i.e. in the lower part of the pig middle it does not make sense to have a rib cutting point.

4.1.3 Back

The back is the biggest product of them all. It is very flexible, but during the scanning it has been placed in such a way that it is easy to find slice correspondence between the cut off back and the back on the pig middle. Knowing slice correspondence makes it possible to develop a simple way of finding the place on the skin where the back and the belly are seperated. Note that we only find the place on the skin, not the whole cut. Assuming that the skin is unstretchable, like we did for the thin plate spline transformation, we can find the width of the cut off back and transfer it onto the whole pig middle.

4.1.4 Back Cutting

The first step is to measure the cut off back width. We are only interested in the width of the skin, since this is assumed to be unstretchable. When the back was scanned, it was scanned with the skin side down towards the shelf. The flexibility of the back would make the skin lie almost flat against the shelf. Given a slice of the back, see figure 4.11 left, the procedure for finding the skin is:

1. make a mask that covers the lowest 5 pixels on the back, see figure 4.11. This forms a preliminary region of interest.
2. Threshold the back image at 0. Skin has a intensity above 0. See figure 4.11.
3. perform an `AND`-operation on the mask and the thresholded image.
4. find for each image column in the `AND`-image the pixel closest the bottom of the image.
5. measure distance between pointson the found skin and sum to get the width

It is a simple way of finding the skin but it works because the skin side is towards the shelf and the skin has an intensity that is different from that of the subcutaneous fat. Repeating the above for each slice in the cut off back, the width along the back can be found. Figure 4.13 shows the width for each slice in the cut off back from middle 2557.

To actually find where on the pig middle the cutting point should be we also need to find the skin on the whole pig middle. Fortunately the same procedure

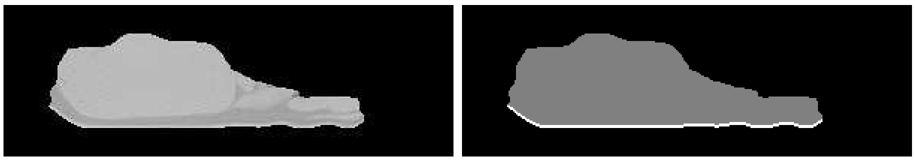

Figure 4.11: Left: Slice 30 from the back from pig middle 2557. Right: White is the actual mask, gray is outline of the back.

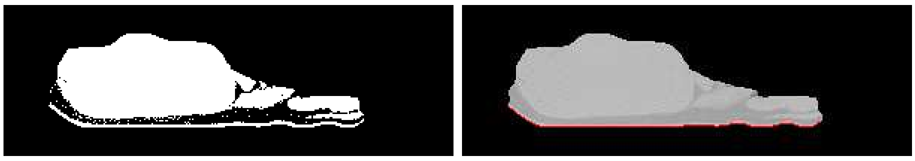

Figure 4.12: Left: Back thresholded at 0. Right: The found skin edge.

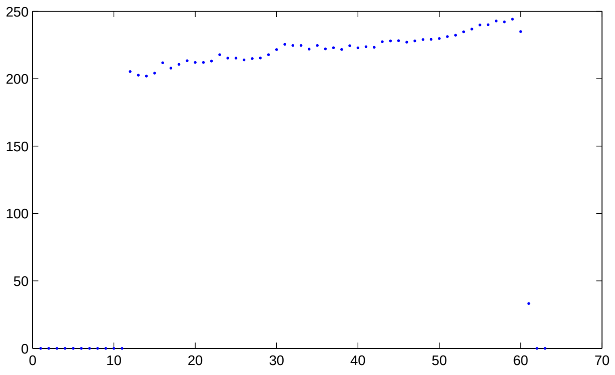

Figure 4.13: Back width for each slice in the cut off back from middle 2557.

can be used as used for the cut off back, see figure 4.14. The only difference is just the size of the object.

Finding the cutting point on whole pig middle is done by measuring cumulated sum of the distances between the found skin points. The point where the cumulated sum is equal to the width of the cut off back, is the point we are looking for.

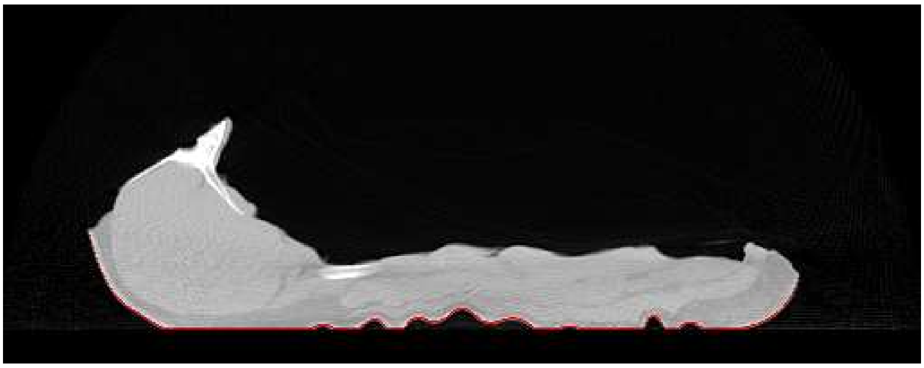

Figure 4.14: The found skin edge on the whole pig middle.

4.2 Results

This section will sum up the results of using the ad hoc methods for finding the cuttings.

4.2.1 Rib Length Statistics

To gain more knowledge about the cutting process of the ribs a study how the rib length varies. It is interesting to know if all the individual ribs are cut at the same length or not. For this reason we make small statistical study. For each of the cut off ribs we fit a straight line to the measured rib length, see figure 4.15. We only include points "inside" the rib, i.e. we leave out the measured lengths near the two ends of the rib. The slope α of these lines can then be studied. The mean of the 35 α's for all the cut off ribs is $0.37\frac{mm}{cm}$ with a 95% confidence interval of $[0.2068; 0.5332]\frac{mm}{cm}$, see figure 4.16 for histogram. From this we can conclude that the individual ribs are not cut off at the same length since the confidence interval does not contain 0. It is a bit hard to argue that this would have any consequence at all in practise. A 30 cm long rib would be cut off with a difference of $0.37\frac{mm}{cm} \cdot 30cm = 11mm$. The ribs are approximately 150 mm long, meaning that the maximum error is in the order of 5-10%.

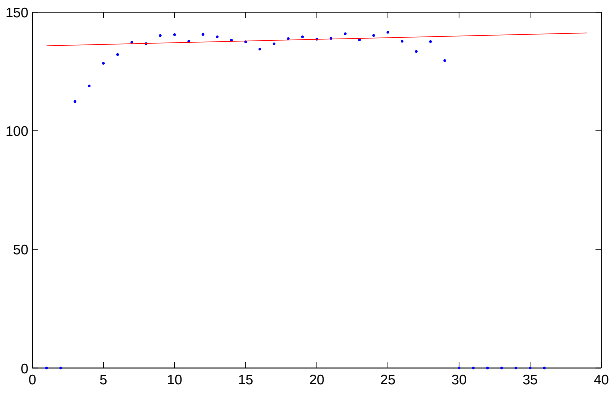

Figure 4.15: Line fittet to the measured length of the cut off ribs from middle 2557.

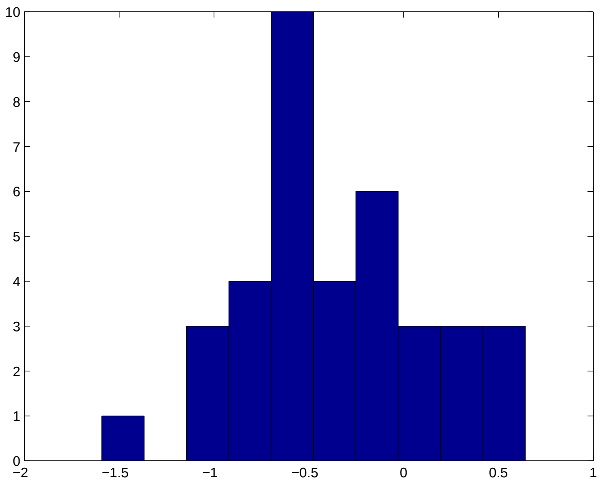

Figure 4.16: Histogram of α for the cut off ribs.

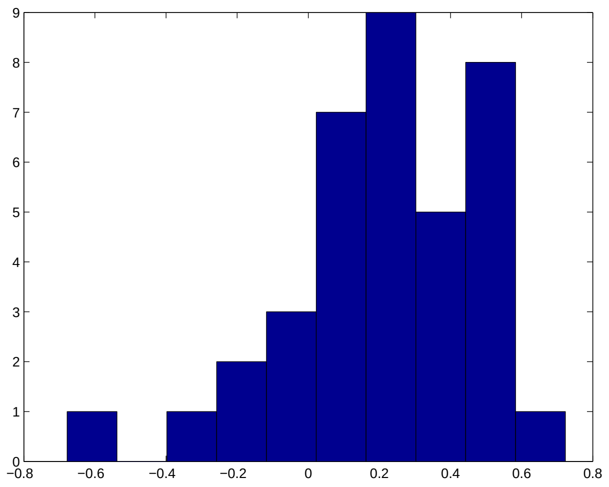

Figure 4.17: Histogram of α for the cut off backs.

4.2.2 Back Width Statistics

Like with the ribs, it is interesting to study how the width varies from back to back. Again, we fit a straight line to the measured width and record the slope α, see figure 4.15 and 4.13. The mean of the α's is $0.2053\frac{mm}{cm}$ with a 95% confidence interval of $[0.1118; 0.2987]\frac{mm}{cm}$. Again, we can reject with 95% certainty that the cuts are straight, since the confidence interval does not contain 0. A back is around 50 cm long, meaning that there is an absolut error on the cut in the order $0.2053\frac{mm}{cm} \cdot 50cm = 10mm$.

4.2.3 Transfered Cuttings

Transfering all the cuttings using the methods described in the previous sections, we get a pig middle with four cuttings points in each slice. There might be one or two of the end slices in the pig middle which do not have all cutting points, but since it is only one or two and they are in the ends of the pig middle, it is not so important. Two examples of transfered cuttings are shown in the next.

4.2.3.1 Pig Middle 2557

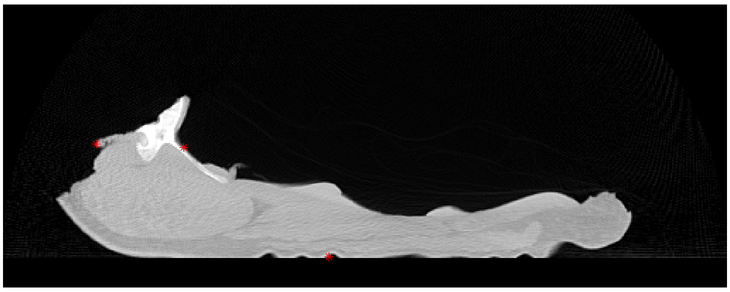

Figure 4.18: Cutting points on pig middle 2557 slice 15.

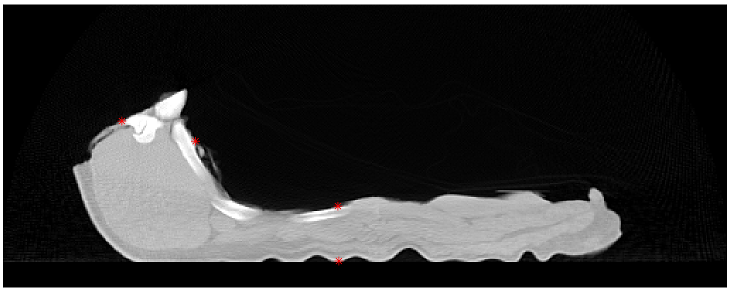

Figure 4.19: Cutting points on pig middle 2557 slice 30.

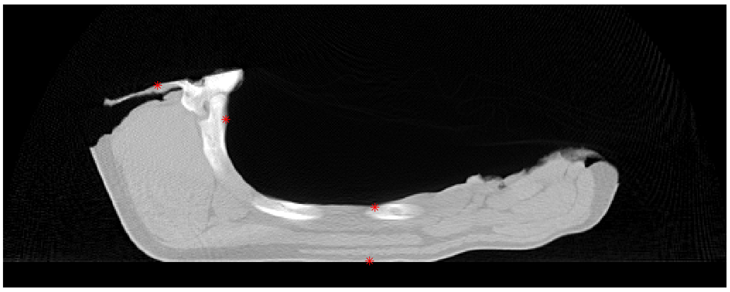

Figure 4.20: Cutting points on pig middle 2557 slice 45.

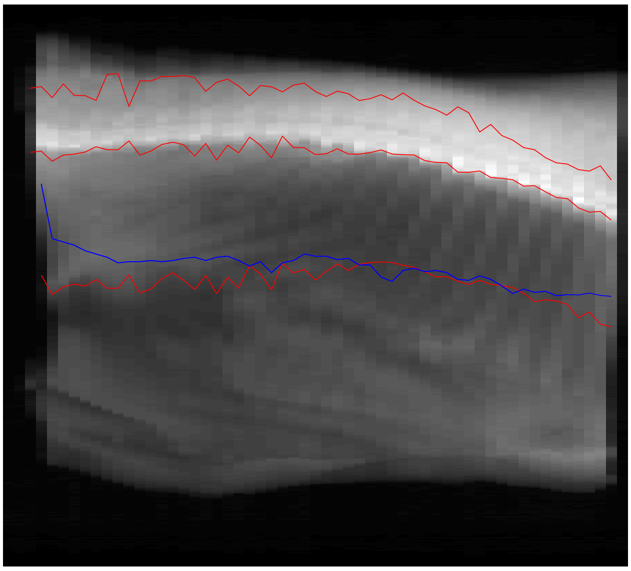

Figure 4.21: Cutting points on pig middle 2557 projected onto xz-plane. The rib cutting point is plottet even in slices where it does not make sense. Two upper red are the spine cuts, blue line is the back cut and the lower red is the rib cut.

4.2.3.2 Pig Middle 8803

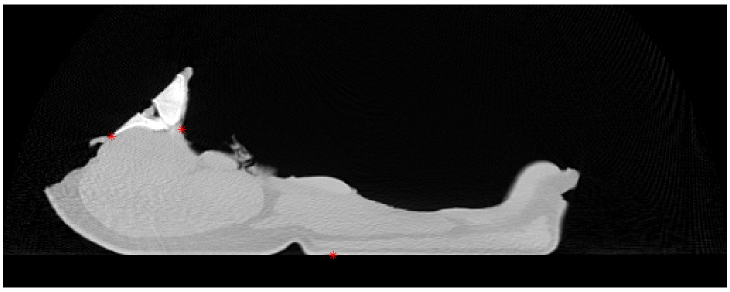

Figure 4.22: Cutting points on pig middle 8803 slice 15.

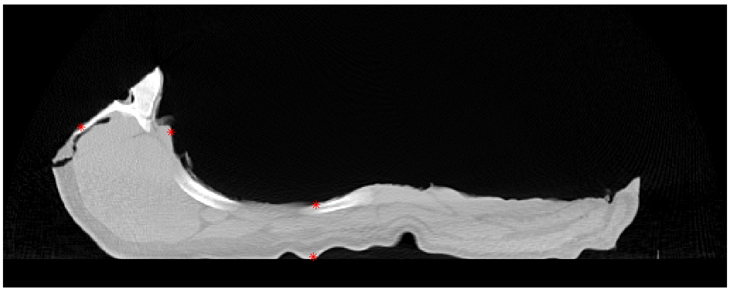

Figure 4.23: Cutting points on pig middle 8803 slice 30.

4.2.4 Discussion

There is no doubt that the ad hoc methods described here are limited in their precision. It is also difficult to find a way to test the methods, since they are ad hoc methods and therefore does not generalise to other kinds of data. Still, visual inspection of the results, show that the results are good enough at least as toy data to test other methods.

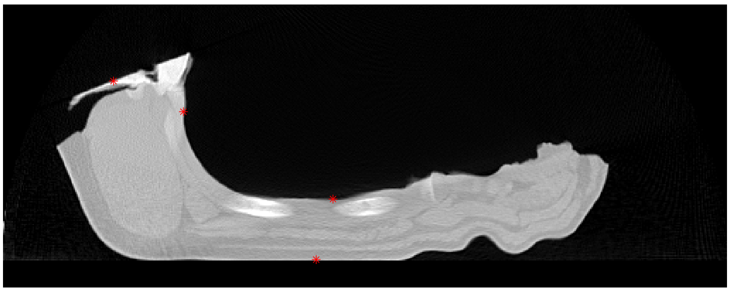

Figure 4.24: Cutting points on pig middle 8803 slice 45.

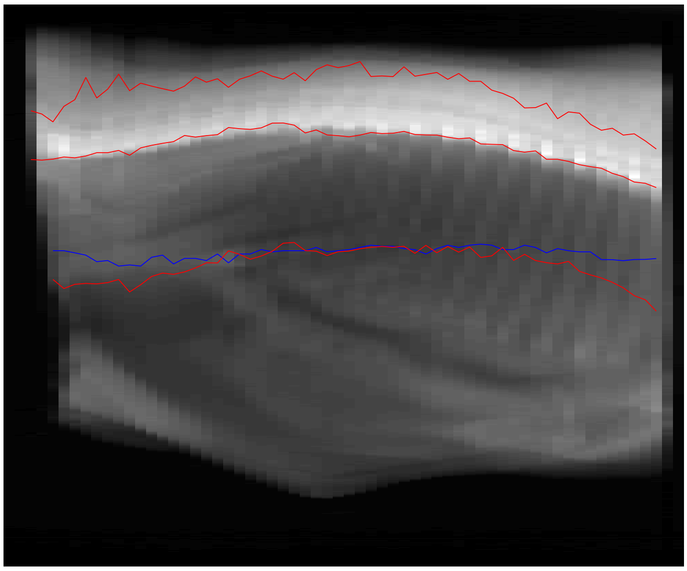

Figure 4.25: Cutting points on pig middle 8803 projected onto xz-plane. The rib cutting point is plottet even in slices where it does not make sense. Two upper red are the spine cuts, blue line is the back cut and the lower red is the rib cut.

Chapter 5

Statistical Shape Model of Cuttings

Now that we have the cuttings on every of the pig middles, it is interesting to study the variation of these cuttings from pig middle to pig middle. A such will give us a clue about where the middle cutting machine is the least precise. To model the shape variation of the cuttings, a common reference for is needed. When we have common reference we can start to analyse the variations.

5.1 Preliminaries

This section contains theory for a method used several in the next section.

5.1.1 Robust Smoothing Spline

Some times it is necessary to model a set of points by a certain function f, often times this function f is a polynomial, due to their nice and well-understood properties. Though polynomials often are excellent choices, they often fail when they are used for extrapolation. This is because there is no constraints on the curvedness of the polynomial, so when they leave the domain of data support they "explode", see figure 5.1. Instead, 1-dimensional thin plate splines, called smoothing splines, can be used. In the 1-dimensional case, where N points $\boldsymbol{x}$ are mapped onto N points $\boldsymbol{y}$, we have

$$\begin{bmatrix} \boldsymbol{A}+\lambda\boldsymbol{I} & \boldsymbol{P} \\ \boldsymbol{P}^T & \boldsymbol{Z} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{a} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} \tag{5.1}$$

where $\boldsymbol{A}_{i,j} = |\boldsymbol{x}_i - \boldsymbol{y}_j|$, $\boldsymbol{P}_{i,:} = (1, \boldsymbol{x}_i)$ and $\boldsymbol{Z}$ and $\boldsymbol{z}$ are 2×2 and 2×1 zero matrices respectively. $\boldsymbol{I}$ is the $N \times N$ identity matrix, λ is a scalar, $\boldsymbol{w}$ is $N \times 1$ and $\boldsymbol{a}_y$ is 2×1. This is basically the same as equation (3.24), except for the cahnge in dimension and the U-function. The solution scheme is also similar to the one described earlier.

Although non-parametric of nature, it is still poossible to adjust the spline using λ. The problem is to determine how large a λ should be. With polynomials a closed form is available for the solution that minimizes the residual error variance. In a sense, the same exist for the smoothing spline, but in that case it will be $\lambda = 0$. Instead, if the variance of the errors on the data points where known, it would be possible to find the λ that make the data point variance σ_y^2 equal to the residual error variance σ_{res}^2.

Assume the underlying function for the data points $\boldsymbol{y}$ is the smooth function $S(\boldsymbol{x})$ and the noise on the data points are normal distributed with $\epsilon \sim N(0, \sigma_y^2)$

$$y_i = S(x_i) + \epsilon \tag{5.2}$$

Now consider the difference of two successive points y_i and y_{i+1}

$$y_{i+1} - y_i = [S(x_{i+1}) + \epsilon_{i+1}] - [S(x_i) + \epsilon_i] \approx \epsilon_{i+1} - \epsilon_i \tag{5.3}$$

where it is assumed that $S(x_{i+1}) - S(x_i) \approx 0$ because S is smooth. Hence the variance of the difference $var(y_{i+1} - y_i)$ is

$$var(y_{i+1} - y_i) \approx var(\epsilon_{i+1} - \epsilon_i) = var(\epsilon_{i+1}) + var(\epsilon_i) = 2\sigma_y^2 \tag{5.4}$$

So to find an estimate of σ_y^2 we can average the $N-1$ point differences

$$\hat{\sigma}_y^2 = \frac{1}{2(N-1)} \sum_{i=1}^{N-1} (y_{i+1} - y_i)^2. \tag{5.5}$$

If no extreme outliers are present, the presented method for variance estimation should give a reliable result[22].

Having an estimate of the variance we can setup a minimization problem. Let $f(\lambda, \boldsymbol{x}, \boldsymbol{y})$ be the smoothing spline

$$\underset{\lambda}{min} \quad var(f(\lambda, \boldsymbol{x}, \boldsymbol{y}) - \boldsymbol{y}) - \hat{\sigma}_y^2 \tag{5.6}$$

The minimization itself can be done in a simple fashion by iteratively applying a Newton steps and checking for negative vaules of λ, or mapping the λ to a function that is spans the whole $\mathbb{R}$. Strictly speaking it is a constrained problem that should be solved using Lagrange multipliers, but since it is so simple as it is, Lagrange multipliers is a bit overkill [15].
Figure 5.1 shows 200 points from the function $S(x) = \tanh(\frac{x-100}{10}) + \frac{x}{100}$ added som noise with $\sigma^2 = 0.01$, a fittet polynomial of degree 8 and a smoothed spline. The spline is a better fit than the polynomial and it is also possible to extrapolate the spline in points close to the domain of support. Table 5.1.1 shows the residual variance of fittet polynomials degree 2 to 15. The residual variance of the spline is $\sigma_{res}^2 = 0.0079$, so the spline seems to overfit just a little bit. The polynomial degree must at least be 15 to match the spline fit.

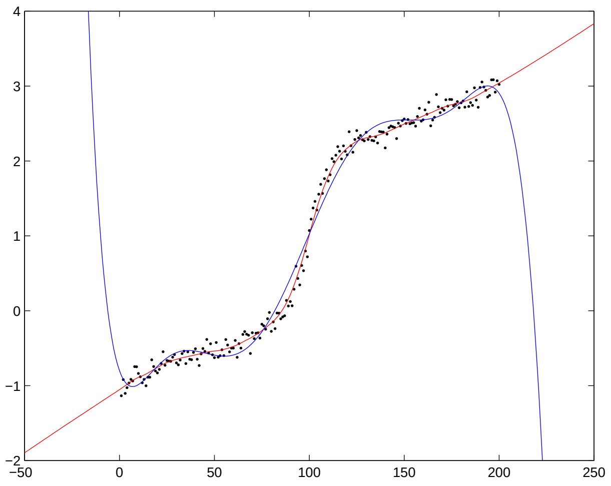

Figure 5.1: Data ●, polynomial degree 8 — and smoothing spline —.

Poly. deg.	2	3	4	5	6	7	8
σ^2_{res}	0.1631	0.0635	0.0634	0.0296	0.0295	0.0148	0.0147
Poly. deg.	9	10	11	12	13	14	15
σ^2_{res}	0.0076	0.0076	0.0040	0.0040	0.0021	0.0021	0.0011

5.2 Inter-middle registration

In order to capture the difference between the cuttings on the different middles, it is necessary to do an inter-middle registration. This means that all the middles has to be registered to the so called *mean pig middle*. The mean pig middle μ is the pig middle which is the mean of all the pig middles in the hyperspace in which they exist. The problem is that we do not know the mean pig middle, but we can estimate it on basis of data. The acutal image registration is done using the nonrigid B-spline based free-from deformation described previously. It did not do well registering products to whole pig middles, but for this purpose where there is one-to-one correspondnce it works well. To find the mean pig middle, the following steps were done:

1. choose a temporary mean pig middle μ_{tmp} in the set. We choose the one with the median volume to avoid being out in the extremes of the pig middle distribution
2. register all the pig middles to μ_{tmp}
3. store transformation for each of these registrations
4. find the mean of all the transformations
5. transform μ_{tmp} according to this mean transformation to get μ
6. re-register all pig middles to μ

Doing this and we will have a common reference for all the pig middles. This means that we in theory can go from any point in a pig middle to the same point in any other pig middle [21]. After the cutting points have been transfered onto the mean pig middle, they are interpolated in every slice using a smoothing spline. This is done to make sure all slices in the mean pig middle have all cutting points.

Figure 5.2 shows an projection of the mean pig middle. The shape of the whole middle looks fine, but if one looks at the bone structure, one can see some bending of the ribs. This is because the registration method does not know that the bones really are rigid structures that cannot be bended. Figure 5.3 show some slices of the mean pig middle. They are reasonable, the edges between meat and fat are sharp, and not blurry as one could fear.

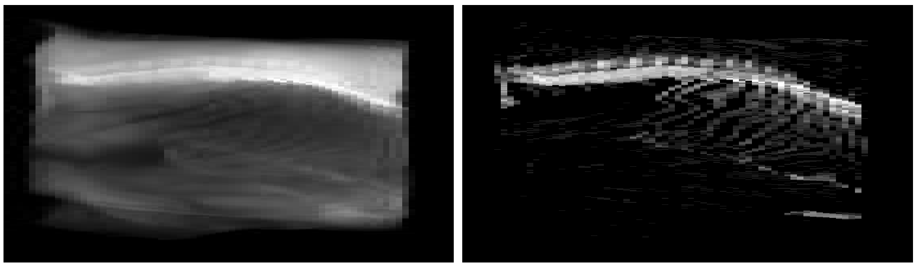

Figure 5.2: Left: Mean pig middle projected onto the xz-plane. Right: Bones in the mean pig middle projected onto the xz-plane.

5.3 Statistical Shape Model

Having a common reference, we can now build a statistical shape model of the cuttings.

5.3.1 Theory

Given is a point set $\boldsymbol{x}$ in an d-dimensional space describing N shapes. First the mean shape is found

$$\boldsymbol{x}_\mu = \frac{1}{N}\sum_{i}^{N}\boldsymbol{x}_i \tag{5.7}$$

and then the covariance of the shapes is found

$$\boldsymbol{S} = \frac{1}{N-1}\sum_{i}^{N}(\boldsymbol{x}_i - \boldsymbol{x}_\mu)(\boldsymbol{x}_i - \boldsymbol{x}_\mu)^T \tag{5.8}$$

The eigenvectors $\boldsymbol{\phi}_i$ and eigenvalues λ_i of the covariance is found (this is a so called principal component analysis(PCA)). Each eigenvector describe a direction of variation in the dataset, the corresponding eigenvalue quantifies how much it describes. Sorting the eigenvectors by their corresponding eigenvalues, therefore gives you a method for simplifying your dataset. If d is large, the precision of the PCA will be low, so instead it may be useful to find the eigenvalues and the eigenvector using a singular value decomposition (SVD) [16]. The data can now be modelled by

$$\boldsymbol{x} = \boldsymbol{x}_\mu + \sum_{1}^{p}\boldsymbol{\phi}_i\boldsymbol{b}_i \tag{5.9}$$

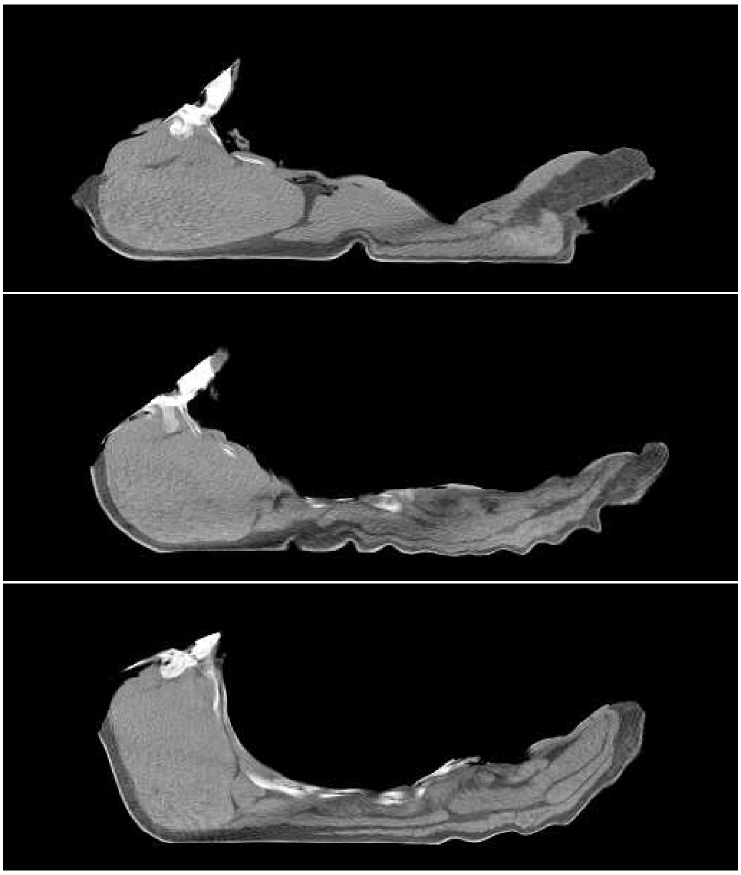

Figure 5.3: Mean pig middle. Top: Slice 15. Middle: Slice 30. Bottom: Slice 45.

where

$$\boldsymbol{b} = \boldsymbol{\phi}^T(\boldsymbol{x} - \boldsymbol{x}_\mu) \tag{5.10}$$

If the parameter b_i is constrained to $\pm 3\sqrt{\lambda_i}$, the model will only describe shapes that are within the range of the original data. The parameter p determines how much of the variation in the dataset we want to model. Often p is chosen such that the model describes 95% of the variation. The value of p is found as the p for which the sum of the p largest eigenvectors is equal to 95% of the sum of all eigenvectors:

$$\sum_1^p \lambda_i = 0.95 \sum_1^d \lambda_i. \tag{5.11}$$

5.3.2 Results

We build the shape model from the location of the cuttings points in each slice. There are 56 slices in the mean pig middle giving a total of $2 \times 4 \times 56 = 448$ parameters. Figure 5.4 shows the eigenvalue plot of the shape model. By using the 13 eigenvectors with the largest eigenvalues, you get a model that covers 95% of the data set. The first and second mode of the shape model are shown

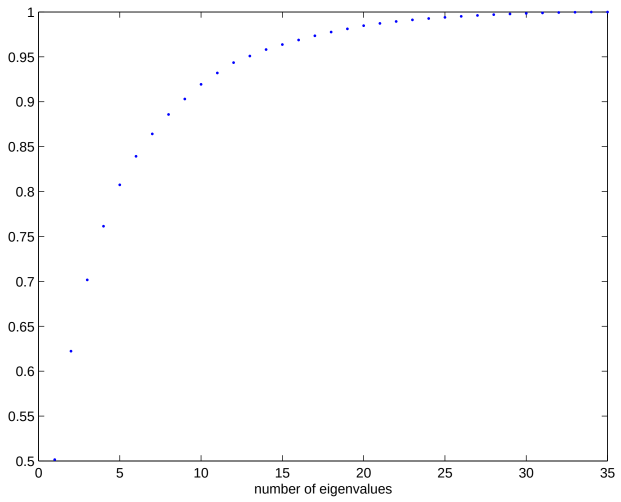

Figure 5.4: Eigenvalue plot for the shape model.

in figure 5.5 and 5.6. Blue is mean, red is $+3\sqrt{\lambda}$ and green is $-3\sqrt{\lambda}$. The first mode seems to describes the variation in the rib cutting points and some of the spine cutting points. The second mode describes a mostly variation on the spine cutting points.

5.4 Virtual Cutting

Since we in theory are able to go back and forth between the different cooordinate systems of the pig middles, we can do a so called virtual cutting. The virtual cutting is the mean cuts on the mean pig middle that are transformed back onto any pig middle that has been registered to the mean pig middle. In theory it works, but the software used to do the registration is not able to do

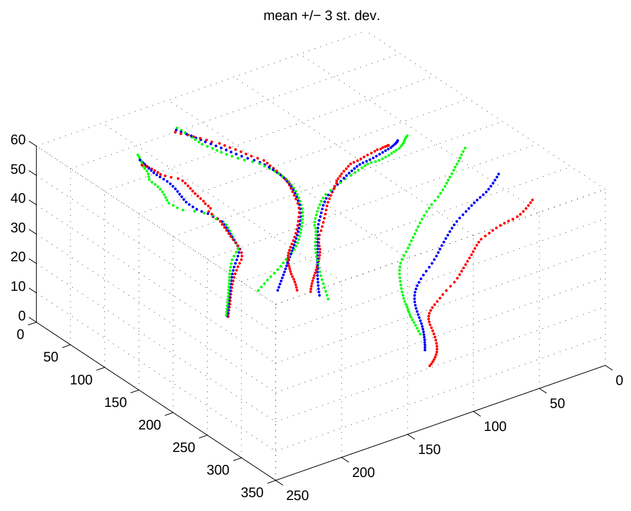

Figure 5.5: First mode of the shape model.

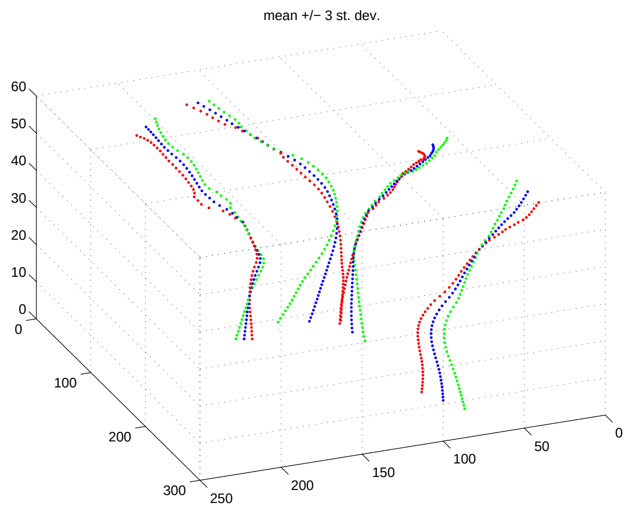

Figure 5.6: Second mode of the shape model.

the correct transformation back. This is an example of the forward/inverse existence problem discussed earlier. The result of registering a pig middle back and forth is the result shown in figure 5.7. The problem is that the inverse cannot be estimated good enough by the software so the is a lot of spatial errors. How it estimated and why it fails, is not known, since the software is a black box.

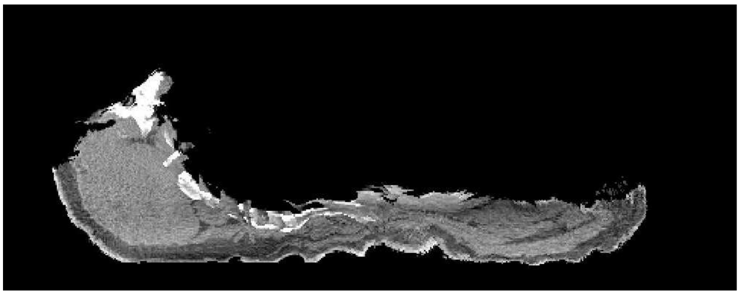

Figure 5.7: Pig middle transformed back and forth.

5.5 Discussion

The quality of the cutting points that were inputted into the shape model was not as good as one could have wished for. We now that the there is a systematic error on the rib cut point (we used the mean length of the rib), and this is can suddenly be seen in the shape model in form of the first mode. Theory behind the shape model is strong enough and we also care about precision by using SVD, but if the data is of less good quality, the result will the same.
The problem with the virtual cutting has nothing to do with data, but is connected to the way the inverse is estimated.

Chapter 6

Conclusion

6.1 Conclusion

This thesis has described the topic of modelling cuttings of pig carcasses. First the data was pre-processed to remove the shelves in the scanner. Also a method for reducing bit resolution with truncation and scaling was studied and applied to data.

The theory behind image analysis was investigated and a few image registration methods were implemented and applied to data. The use of image analsysis for finding cuttings was unsuccesfull.

A number of ad hoc methods for finding the cuttings were developed with succes, though their precicion is not as good as one could wish for. The methods were applied to the whole data set to get a full set of cutting points to use in further analysis.

Statistical shape models were studied and one was build on basis on the cutting points. 13 eigenvectors showed to be enough to describe 95% of the data variation. It demonstrated the principle of statistical shape models, but due to the low precision on the data, it was not useful in practice.

A virtual cutting method was briefly touched, but due to software problems, it did not give good results.

6.2 Future work

For a fully succesfull project one or more of these things may be considered:

1. the data has too low resolution in the z-direction, isotropic voxels best, but anything smaller than 10 mm is an improvement.
2. development of volume constraining image registration method, e.g. [17].
3. development of a registration method that has both forward and inverse, or at least the possibility to estimate it well.

Appendix A

Pig middle scanning numbers

Table of pig middle scanning numbers.

Carcass	Uncut	Cut	Excluded
8803	27	26	
8829	29	28	
260	4	3	
8864	31	30	
2720	34	33	
3256	36	35	
3338	38	37	
5962	40	39	
6875	42	41	
6926	44	43	
2533	46	47	
3290	48	49	
2601	6	5	
4193	50	51	
4597	52	53	
5526	54	55	
6137	56	57	
6976	58	59	
6979	60	61	X
1403	63	64	
1534	65	66	
1805	67	68	
2557	69	70	
4152	7	-	X
3218	71	72	X
5371	73	74	
5991	75	76	
9502	77	78	
4288	9	8	
2	80	81	
1654	82	83	
2223	84	85	
7227	86	87	
8362	88	89	
9064	90	91	
2522	93	94	
2615	95	96	
4332	97	98	
6551	99	-	X

Appendix B
Software

The implemented software (Matlab files) can be downloaded as an burnable ISO-image from `www.student.dtu.dk/~s011572/thesis` or contact the author by email `allan@lyckegaard.dk`.

Bibliography

[1] Butchering a 1/2 Pig. `http://www.btinternet.com/~happydudevir/butchering.html`.

[2] Danish Meat Association. Statistik 2005 svin, 2005.

[3] P. Besl and McKay. A method for registration of 3-d shapes. *Trans. PAMI*, 14(2), 1992.

[4] D.L.G.Hill C. Studholme and D.J. Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition*, 32(1):71–86, Jan 1999.

[5] Simon Jackson and Richard Thomas. *Cross-Sectional Imaging Made Easy*. Churchill Livingstone, 2004.

[6] H.B. Nielsen K.Madsen and O. Tingleff. Methods for non-linear least squares problems. Lecture note, IMM, March 2004.

[7] H.B. Nielsen K.Madsen and O. Tingleff. Optimization with constraints. Lecture note, IMM, March 2004.

[8] Thomas M. Lehmann, Claudia Gonner, and Klaus Spitzer. Survey: Interpolation methods in medical image processing. *IEEE Transactions on Medical Imaging*, 18(11):1049 – 1075, Nov 1999.

[9] Tony Lindenberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):77–116, 1998.

[10] Mathworks. *Matlab Documentation, Image Processing Toolbox.*

[11] Jan Modersitzki. `http://www.math.mu-luebeck.de/safir/flirt-matlab.shtml`.

[12] Jan Modersitzki. *Numerical Methods for Image Registration.* Oxford Science Publisher, 2004.

[13] J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, pages 308–313, Jan 1965.

[14] NEMA. `http://medical.nema.org/`.

[15] H.B. Nielsen P.E. Frandsen, K. Jonasson and O. Tingleff. Unconstrained optimization. Lecture note, IMM, March 2004.

[16] Lars Kai Hansen Rasmus Elsborg Madsen and Ole Winther. Singular value decomposition and principal component analysis. Technical report, IMM, DTU, 2004.

[17] T. Rohlfing and C.R. Maurer. Volume-preserving nonrigid registration of mr breast images using free-form deformation with an incompressability constraint. *IEEE Trans. on Medical Imaging*, 2003.

[18] D. Rueckert, L.I. Sonoda, and C. Hayes et al. Nonrigid registration using free-form defoemations: Application to breast mr images. *IEEE Transactions on Medical Imaging*, 18(8):712–721, August 1999.

[19] J. A. Schnabel, D. Rueckert, M. Quist, J. M. Blackall, and et al. A generic framework for non-rigid registration based on non-uniform multi-level free-form deformations. In *In Fourth Int. Conf. on Medical Image Computing and Computer-Assisted Intervention (MICCAI '01)*, pages 573–581, October 2001.

[20] M. Sonka and J.M. Fitzpatrick. *Handbook of Medical Imaging, Volume 2. Medical Image Processing and Analysis.*, volume 2. SPIE, 2000.

[21] T.F.Cootes and C.F.Taylor. Statistical models of appearance for computer vision. Technical report, University of Manchester, 2001.

[22] Larry Wasserman. *All of Statistics.* Springer Textsin Statistics, 2004.

[23] Pascal Cachier Xavier Pennec and Nicholas Ayache. Understanding the "demon's algorithm": 3d non-rigid registration by gradient descent. In *MICCAI 99*, 1999.

[24] Ming-Hsuan Yang and Jongwoo Lim. A direct method for modeling non-rigid motion with thin plate spline. *Computer Vision and Pattern Recognition*, 2005.