Active Shape Models in the Poultry Processing Industry

Francis Suson Cagatin

Faculty of Electrical and Computer Engineering
University of Iceland
2012

Active Shape Models in the Poultry Processing Industry

Francis Suson Cagatin

60 ECTS thesis submitted in partial fulfillment of a
Magister Scientiarum degree in Electrical and Computer Engineering

Advisor(s)
Kristinn Andersen
Hrafnkell Eiriksson
Jóhannes R. Sveinsson

Faculty Representative
Magnús Örn Úlfarsson

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, May 2012

Active Shape Models in the Poultry Processing Industry
60 ECTS thesis submitted in partial fulfillment of a *Magister Scientiarum* degree in Electrical and Computer Engineering

Faculty of Electrical and Computer Engineering
School of Engineering and Natural Sciences
University of Iceland
Hjardarhagi 2-6
107, Reykjavik
Iceland

Telephone: 525 4000

Bibliographic information:
Francis Suson Cagatin, 2012, Active Shape Models in the Poultry Processing Industry, Master's thesis, Faculty of Electrical and Computer Engineering, University of Iceland, pp. 61.

Printing:
Reykjavik, Iceland, May 2012

Abstract

The Active Shape Model (ASM) algorithm is an image segmentation technique that takes advantage of the data derived from the training set. The model is built up from shape information obtained from analyzing points along object boundaries known as landmarks. This forms the basis in searching unknown images for target shapes.

The theory and process of building up the ASMs and image search is discussed. Examples are given to reinforce the discussion. Finally, the ASM is used on several poultry processing applications.

It is found that the inherent ability of ASMs to adapt to different variations seen during training make it an ideal candidate for applications in the poultry processing industry.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my heartfelt gratitude to all the people who have assisted me in the completion of this thesis.

My professor Dr. Karl Sölvi Guðmundsson for setting me up this path of study. Thank you for your patience and faith in me.

Much appreciation also to Marel ehf. for allowing me to work under their roof and providing all the test images. Thank you to Hrafnkell Eiriksson and Dr. Kristinn Andersen for providing me with several topics and allowing me to choose which one suited me the best. I am also grateful for their valuable insight and guidance during the course of this thesis.

Dr. Jóhannes R. Sveinson, my thesis supervisor for his incredible patience in reading and correcting the numerous drafts.

To my girlfriend, Ruth, thank you for all the love, unwavering support and persistent motivation.

1 Introduction

Humans learn the concept of shapes at a very early age. It is an important tool in the child's language development [**24**]. Toys such as the one shown in Figure 1.1 help stimulate the process of learning shapes. Toddlers learn about different shapes and colors as they place the right object in its slot. As we grow older, we develop the ability to distinguish shapes naturally and intuitively. This conceals the complexity at how the seemingly simple process of distinction is actually carried out.

Figure 1.1 A child playing with a shape toy.

Given a certain picture such as the one in Figure 2.1, it is trivial for anyone to name simple shapes like the triangle and square. This does not stop there; humans are able to identify more complex shapes even with great degree of variability. Dogs for example, come in all shapes and sizes. But presented with an image of a dog of a common breed, most persons will be able to identify it as a canine.

Most people do not know all the breeds of dog that exist but most probably have seen a breed or two. This gives them a rough estimate of how a dog would look like. This *a priori* knowledge is enough to enable them to classify, to some degree, if an animal presented to them is a breed of dog or another animal entirely.

In this thesis, the theory and application of the Active Shape Model (ASM) is explored. This technique is different in that it takes advantage of *a priori* information from a training set to search for object instances in unknown images.

1.1 Motivation and objective

The poultry processing industry in general pose a difficult challenge for image processing algorithms. Natural products, as seen in the food processing industry, are mostly non-rigid objects. As such there is a great variety in which even the same type of product appears. In a processing line that handles only chicken wings, for example, it cannot be expected that the wings appear in the same exact position every time. Some can be lying on one side, while the others on the other; the angle at which the wings open can be different for every piece and even the sizes of the portions themselves can be different. These are just some of the hurdles that an image processing application has to overcome in order to be effective in this type of operation.

The primary advantage of using ASMs is its ability to learn from training and allowance for large discrepancies during image search. We then aim to explore the theory of ASMs and investigate its effectiveness in applications in the poultry processing industry.

1.2 Thesis overview

This thesis is arranged into several chapters

- **Chapter 2: Discussion of shape** –A formal definition of shape and point distribution models is introduced.
- **Chapter 3: Data set** –A brief description of the SensorX machine and the test images used in this thesis is provided.
- **Chapter 4: Active Shape Models** –The theory and foundation of the Active Shape Model is presented.
- **Chapter 5: ASM Implementation and Results** –The ASM algorithm is trained with the X-ray images and the performance is discussed.
- **Chapter 6: ASM Applications in the Poultry Processing Industry** –The results from the ASM algorithm is used in conjunction with other techniques to solve different challenges in the poultry processing industry.
- **Chapter 7: Conclusion and Future Work** –Proposals for further studies resulting from this thesis are put forward and a conclusion is submitted.

2 Discussion of Shape

In this chapter we establish the definition of a shape. We also look into landmarks, the foundation of ASMs. Finally, we discuss point-distribution models.

2.1 Shape

Before any further discussion, we must first define *shape*. A formal definition is given by D. G. Kendall [**10**]:

Definition 1: *Shape* is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object.

Changes in location, scale and rotation are called Euclidean similarity transformations. Several objects are presented in Figure 2.1. We identify the first three shapes, Figure 2.1a, Figure 2.1b and Figure 2.1c, as having the same shape, they are all triangles. The differences in size, location and orientation do not hinder us from classifying it as such. It can be shown that by performing one or a combination of translation, rotation and scaling, these figures are exactly alike, e.g., Figure 2.1b can be scaled down to match Figure 2.1a, Figure 2.1c can be rotated to fit Figure 2.1b. The object in Figure 2.1d, however, cannot be classified as a triangle regardless of any similarity transforms. Hence, it is a different shape from the first three and we know it as a square. Therefore, a shape must be invariant to any similarity transformation.

2.2 Landmark

Now that we have defined what a shape is, we need to describe them. A good starting point is to look at the outline of the object in question. But instead of considering

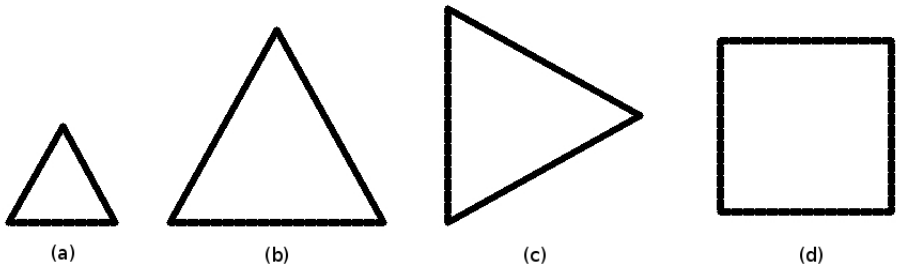


Figure 2.1 Objects (a), (b) and (c) have the same shape but different transformations. Object (d) has a shape that is different from the other three.

all the points that lie in the outline, we select only certain points that are of interest. We adopt the concept of a *landmark* as defined by Dryden and Mardia [**10**].

Definition 2: A *landmark* is a point of correspondence on each object that matches between and within populations.

These landmarks are further classified into three categories [**10**]:

1. **Anatomical landmarks** – Points assigned by an expert that corresponds between organisms in some biologically meaningful way.
2. **Mathematical landmarks** – Points located on an object according to some mathematical or geometrical property of the figure, e.g., at a point of high curvature or at an extreme point.
3. **Pseudo-landmarks** – Constructed points on an object located either around the outline or in between anatomical or mathematical landmarks.

In this manner, a shape is described by the set of landmark points acquired along its boundary. Although this is directly applicable to two-dimensional (2D) shapes, extension to three or more dimensions is also possible.

In the succeeding discussions, no distinction is made among the different landmark types. All landmark points whether anatomical, mathematical or pseudo-landmarks are called simply as landmarks or landmark points.

In this thesis only 2D shapes are considered. A shape described by a vector representation would then be:

$$\mathrm{z} = [x_1, y_1, x_2, y_2, \ldots, x_{n-1}, y_{n-1}, x_n, y_n]^T. \tag{2.1}$$

2.3 Point Distribution Models

Each object in Figure 2.1(a) - (c) can be transformed exactly to resemble the other two using similarity transformations. In less than ideal conditions, though, this is not the case. Normally, there are variations that exist among objects of the same shape that cannot be resolved by similarity transformations, i.e., one side may be a little longer or a vertex can be slightly skewed. These minor deformations are often acceptable and do not hinder the object from being classified into its respective shape. With this in mind, we build a model that describes a shape but allows for slight variations. This model is built upon the statistical information derived from analyzing known objects of the same shape class in the training set. This is a called *point distribution model* [**8**].

The foundation of point distribution models are landmarks. Each object in the training set must be labeled with a set of landmark points. In most cases, as is done in this thesis, there is one object for every training image. These objects do not necessarily have the same orientation across the training images. There can also be minor variations as described above. It is important that in the landmarking process, the landmark points correspond to the same point on the shape across all the training images regardless of orientation and slight deformations. For rigid objects or those that are regularly-shaped, the placement of landmark points is well-defined. Anatomical, mathematical, and pseudo-landmarks placed at equal distances in between are sufficient to describe the shape. For non-rigid objects like poultry or meat, the placement of landmarks can be ambiguous. The perimeters of these objects not only depend on the actual size of the cut but also how they are positioned during image acquisition. As a result, the normal process of placing pseudo-landmarks at equal distances from anatomical or mathematical landmarks becomes unreliable. Extra care must be taken so that each landmark corresponds to the same position across all images.

As mentioned above, the objects in the training images often have slight differences among them. Thus, after acquiring all the landmarks, we align them to a common reference frame. This is done by performing similarity transformations on the training images. This goes back to our definition of shape that all the translation, scale and rotation are filtered out.

Statistical information is built up after isolating geometric information from the training images. Using multivariate data introduced by the landmarks prevents this from being a straightforward process. Hence, we must employ dimension-reduction techniques to facilitate the operation. One such method is the P*rincipal Component Analysis (PCA).* This will be discussed further in Chapter 4.

3 Data Set

The primary aim of this thesis is to test the Active Shape Model (ASM) algorithm on non-rigid images found in the food processing industry. For this purpose, all the images used in this thesis are taken by the SensorX poultry bone detection machine manufactured and developed by Marel hf. The images were taken by several SensorX machines at different instances running in factory conditions. This is evident in some of the images as artifacts or foreign objects are visible. The images selected for the experiments contain only one fillet. Most of the images include the complete fillet within its limits. In some, however, the fillet is not completely contained in the image. This was done to explore the performance of the ASM algorithm in these conditions.

3.1 SensorX poultry bone detection machine

Figure 3.1 The Marel SensorX machine. Courtesy of Marel ehf.

The SensorX is an inspection machine that detects contaminants in poultry products. It uses X-ray technology to detect bones, metal, stone and glass. The X-rays are generated by a 500W source with an 1mm focal spot size having cone geometry. This is projected onto a belt which runs at 500mm/sec. The pixel size is about 0.6mm and the scanning width is 305mm. A line scan sensor captures the projected X-rays and builds the image up at a rate of 625 lines per second. The images produced are in grayscale.

Bones and other contaminants larger than 2mm are detected with an accuracy of 99% with only a 3% false positives rate.

3.2 Chicken breast fillet

All test and training images contain a chicken breast fillet. The United States Department of Agriculture (USDA) identifies the chicken breast as separated from the back at the shoulder joint and by a cut running backward and downward from that point along the junction of the vertebral and sternal ribs [**35**]. This is also known as the butterfly cut. It can

be sliced along the breastbone to separate them into equal halves. The ribs and other bones are taken out to complete the fillet. Figure 3.2 shows two samples of chicken breast fillets.

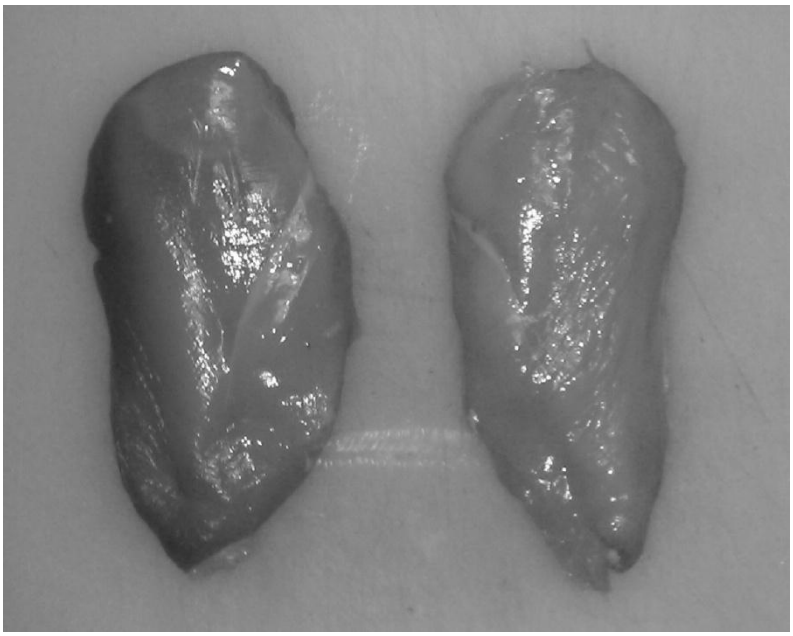


Figure 3.2 Some samples of chicken breast fillets.

4 Active Shape Models

This chapter discusses ASM. A review of existing literature is given to briefly explain the evolution of the algorithm. The ASM algorithm is broken down into two phases: the training phase and the search phase. Each of the steps in the two phases is reviewed. Finally, a multi-resolution approach to image search is examined.

4.1 History of related work

Numerous methods and algorithms have been developed to perform shape description and representation. These techniques extract information from the desired object for identification and recognition. This information can be in the form of shape descriptors. Peura and Iivarinen [**27**] used classical shape descriptors like convexity, variance and principal axes to describe objects. Zahn and Roskies [**43**] described shapes using the Fourier coefficients of a periodic function from the closed curve bounding the object, calling these Fourier descriptors. This was extended by Abter *et al.* [**1**] to accommodate affine deformations. Xie *et al*. [**41**] used skeletal segments computed using the medial axis transform to find correspondences in different instances of an object.

An entirely different approach was introduced by Fischler and Elschlager [**13**] with deformable models. In their representation, a model is a reference image composed of several rigid points or components. Each point is connected to another by a spring. The cost of fitting the model to an input image is measured by evaluating the tension that each spring experiences. In the same year, Widrow [**39**], [**40**] presented his 'rubber-masks'. In his technique, a model is iteratively distorted to fit an input image up to a desired degree of accuracy; thus creating a stretched template or a rubber-mask. This was especially helpful for applications where the target objects do not share the same exact shape, such as human chromosomes in Widrow's study. The model-based approach benefit from being able to adapt to differences across the input images to correctly identify the target objects. This is especially true for objects that have a high degree of variation among similar instances such as in biological structures. Hence, the extensive use of these methods is found in medical applications [**26**].

Kass *et al.* [**22**] proposed the active contour model. Energy-minimizing contours or snakes actively search for object boundaries. The snakes are acted upon by a combination of forces or energies.

1) The internal contour forces act to impose piecewise smoothness.
2) Image forces push the snake to prominent image features.
3) External forces move the snake near the desired boundaries. In the presence of these forces, the snake deforms as it finds a local minimum of the energy.

The constant minimization of the energy functional demonstrated by the snakes makes it an 'active' model. In addition to this, constraints can be set on the snakes so that they remain smooth and do not bend more than is allowed. Shapes are then represented by the snakes computed by this method. However, parameters can be set so that the snakes are guided on to a desired path. Yuille *et al.* [**42**] used this to find the human eye in images. The problem with this method is that another set of parameters must be determined whenever a new object is introduced.

Cootes *et al.* [**8**] proposed the ASM to address both generality and specificity. They argue that modeling variability in a theoretical manner can be very difficult. Therefore, the only viable means is to build up statistical information from a set of training images and learn the patterns of variability from it. Thus, the model is only allowed to deform in a manner consistent with the variations described in the training set. ASMs have been successfully implemented in different applications [**31**], [**32**], and [**21**].

Further developments of ASM methods have also been proposed. Klim *et al.* [**23**], for example, used different regression matrices improving the search range and speed of the algorithm. Ginneken *et al.* [**37**] employed a non-linear classifier and trained it with a set of feature vectors to improve the movement of landmark points. A direct extension of ASM is the Active Appearance Model (AAM) [**33**].

4.2 ASM overview

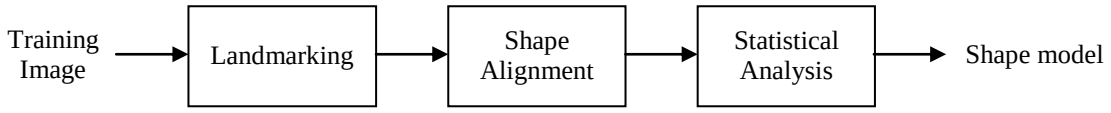


Figure 4.1 The major stages in training ASM.

Building active shape models is a three-step process. The first step is to capture information from the training set. This is accomplished by labeling the images with landmark points. An expert can place points on significant sites of the image. An alternative to this tedious task is to have landmarking done by an automatic method. Once the images are all labeled, alignment can be done. Statistical analysis is then performed on the aligned shapes, an example of which is the *Principal Component Analysis* (PCA). These steps are further discussed in the following sections.

4.3 Modeling shape variation

4.3.1 Acquiring landmarks

Labeling the training images is only done during training. Landmarks are the foundation of ASM. Hence, the process of acquiring a set of points must be done with great care to ensure the accuracy of the succeeding processes. Well-placed landmarks will also result in faster execution of the succeeding algorithms in the ASM technique.

Images of bones and organs often contain points of anatomical importance. Accordingly, landmarks on these images are primarily placed manually by medical doctors or experts in that field. In the separate experiments done by Eiriksson [**12**] and Stegmann [**33**], the images were annotated by medical professionals. For more ambiguous shapes such as weed leaves [**32**], salient or mathematical points are chosen on the outline of the shape and an equal number of pseudo-landmarks are placed in between them. Whichever method is used for choosing the landmarks, the goal is to have each landmark correspond to the same position on the shape across the training images. The number of landmarks, although not as critical, must also be chosen carefully. Too few points can cause inaccuracies during the search phase, with the object not being sufficiently described by the model. Too many points, on the other hand, increase the computational complexity of the process and consequently lengthen the completion time.

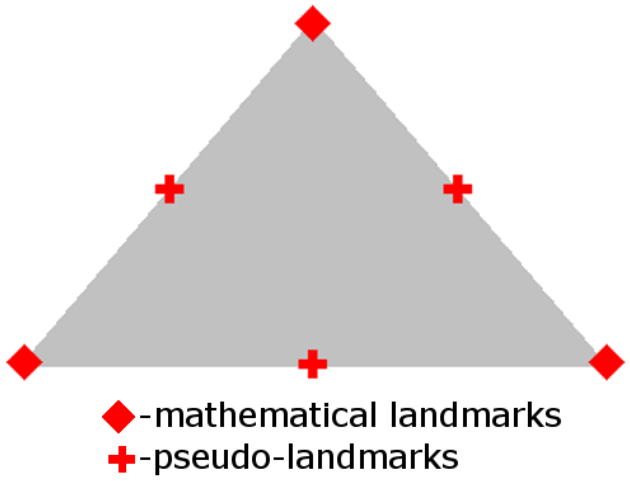


Figure 4.2 A landmarked triangle shape.

Figure 4.2 shows a triangle that is labeled with landmarks. A simple shape like this presents easily identifiable points that can be used as landmarks. Salient points such as the vertices of the triangle, marked with diamond indicators, are prime candidates for this. It is noticeable, though, that there are considerable distances between these mathematical landmarks. It is enough to represent a triangle with just three landmarks. But for more complex shapes, the distances between mathematical landmarks can be too great, or the mathematical landmarks themselves can be too few that it will not correctly represent the shape we are trying to build a model from. It is for this reason that we need to place points in between them. In Figure 4.2, the cross-shaped indicators are pseudo-landmarks.

While it may be trivial to mark simple shapes such as in Figure 4.2, labeling images with hundreds of points is very tedious and even more so in 3D or higher dimensions. Several automatic and semi-automatic methods have been developed to counteract this.

Walker *et al.* [**38**] used pairwise correspondence to generate landmarks. In their method, a bounding box containing the target shape is placed on all training images. Salient features found in each image are then located on all the other images. Global correspondence is established through an iterative scheme. This technique worked well on images that do not have significant changes in pose.

Hill *et al.* [**19**] took a slightly different approach. A pairwise corresponder is used to get the cost function of matching one image to another. A binary tree composed of paired images is built according to their cost function. The worst-match pair is placed at the bottom of the tree and the best-match pair is at the top. The top pair is averaged to get the mean shape. The landmark points are determined on the mean shape. These are propagated through the leaves of the tree using the pairwise corresponder employed previously. Rueda *et al.* [**28**] followed the same path of deriving the landmarks from a mean shape and propagated them through the training images. In their method, the training images are binarized and segmented prior to processing. A distance transform is then applied to the images and the mean is extracted. From the mean shape, the landmarks are selected by the c-scale method [**30**]. Parametric and distance transform propagation were compared as propagation methods for the landmarks.

Evaluating landmarks done manually involves several experts performing the landmarking process multiple times. This assesses the repeatability and reproducibility of the results [**33**]. For the purposes of this thesis, the landmarks are placed by the author. Mathematical and pseudo-landmarks points are chosen and located in each training image.

4.3.2 Shape alignment

Following Definition 1, we must remove the differences in size, position and rotation of the objects in the training set in order to properly describe the shape. By rotating, translating and scaling the shapes with respect to a set of axes, we effectively allow the landmarks to be used for statistical analysis. Only then can we build a reliable model.

Shape alignment is done by mapping one shape to another [**8**]. Let the base shape z_a contain n points and be represented by

$$z_a = [x_{a1}, y_{a1}, x_{a2}, y_{a2}, \dots, x_{an-1}, y_{an-1}, x_{an}, y_{an}]^T. \tag{4.1}$$

We then map shape z_b *to* z_a. Rotation and scale is introduced by the transformation, $M(s,\theta)$, of the shape vector

$$M(s,\theta)\begin{bmatrix} x_{bk} \\ y_{bk} \end{bmatrix} = \begin{pmatrix} (s\cos\theta)x_{bk} - (s\sin\theta)y_{bk} \\ (s\sin\theta)x_{bk} + (s\cos\theta)y_{bk} \end{pmatrix} \tag{4.2}$$

where s is the scale and θ is the rotation. Translation is done by adding a vector of variables to the shape vector

$$t_b = [t_{xb_1}, t_{yb_1}, \dots, t_{xb_n}, t_{yb_n}]. \tag{4.3}$$

In order to accomplish alignment, we attempt to minimize a weighted sum of squares of distances between z_b to the z_a. Let this sum be

$$E_b = \left(z_a - M(s_b, \theta_j)[z_b] - t_b\right)^T W\left(z_a - M(s_b, \theta_j)[z_b] - t_b\right) \tag{4.4}$$

where W is a diagonal matrix of weights. The weights are included to give more importance to landmark points that vary less across the training images. The points in the shape that move considerably relative to the other points not only increase the computational complexity of the algorithm but also do not add significant information. Thus by placing less emphasis on such points, we improve the performance of the algorithm. The weights are computed by

$$w_k = \left(\sum_{i=0}^{n-1} V_{D_{ij}}\right)^{-1}, k = 1,2, \dots, n, \tag{4.5}$$

where w_k is the vector of weights for point k. D_{ij} is the distance between points in one shape. $V_{D_{ij}}$ is the variance of the distances between points i and j over the training set. Points that have large movement will have large variances and consequently low weights. Stable landmarks have less movement and higher weights.

In order to align N shapes, we apply the method described previously in an iterative scheme. A common method of doing this is by using the generalized Procrustes analysis [**15**]. This is an algorithm that normally involves the following steps:

1. Determine a preliminary shape (e.g. first/last shape).
2. Align the training shapes to the preliminary shape.
3. Calculate the resulting mean from the aligned shapes.
4. Normalize the mean to chosen constraints.
5. Align the training shapes to the normalized mean.
6. Repeat from until convergence.

Convergence is reached when the computed mean only changes within a specified limit during iteration. The normalization step ensures that the algorithm converges. Establishing a preliminary shape provides a starting point of the alignment procedure. However, it does not affect the final mean shape. The experiments of Cootes *et al.* [**8**] showed that the scheme arrives at the same mean regardless of the preliminary shape chosen.

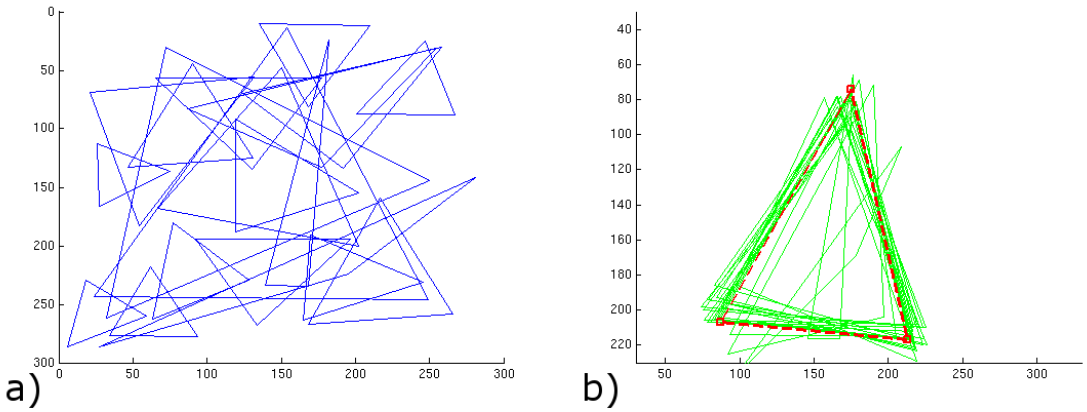


Figure 4.3 Unaligned (a) and aligned (b) sets of triangles.

Figure 4.3 (a) shows a plot of twenty triangles of different locations, sizes, and orientations. This was taken from a training set of twenty images, each containing one triangle and then manually landmarked. In this instance, only the vertices of the triangle were used for landmarking. For a simple shape such as this, the vertices are enough to model the shape and to show the effect of the alignment procedure. Figure 4.3 (b) is the plot of the aligned triangles. The mean shape is superimposed with the dotted outline. Looking closely at the image, we see there are triangles that appear in the center of the cluster. This shows that the alignment procedure does not alter the shape of the objects but rather only seeks to find the best fit through the similarity transformations.

4.3.3 Statistical analysis

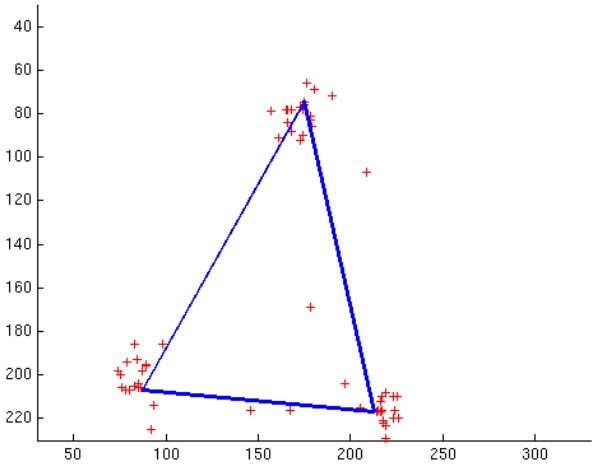


Figure 4.4 The mean shape of the training set with all the landmarks after the alignment process.

Once the objects are aligned, capturing statistical information from the training set is done next. Figure 4.4 shows a mean shape derived from aligning a set of twenty images containing triangles. The landmarks are overlaid on the mean showing the variability at which the landmarks are taken. It is important to note that these points are not independent and have some correlation between them. A point cannot move to a different location without affecting the adjacent points in the same object to some degree as they belong to the same object.

We let each aligned shape z_a to be a point in $2n$ dimensional space. A training set of N images then forms a cloud of N points in this space. Cootes *et al.* [**8**] identify the region at which the points occupy in the $2n$ dimensional space as the *allowable shape domain*. New shapes that conform to the training set can be formed by moving about in this domain. The high dimensionality of this space complicates the process of finding new shapes mathematically. A common method of dealing with high dimensional data is by using PCA.

PCA, also known as Karhunen-Loeve transform, was developed by Harold Hotelling in 1933 [**20**]. In this technique, a new set of bases is determined according to the variance in the data. This set of bases, or principal components, are orthogonal to each other and arranged according to the magnitude of the variance each one describes. In the process of transforming the original basis of the data set to the computed PCA basis, dimensionality reduction is achieved.

Principal components or modes are derived from the eigenvectors of the covariance matrix. The amount of variance contained in each mode is defined by its corresponding eigenvalue [**8**]. Therefore, the mode that corresponds to the largest eigenvalue describes the most variance and is also the most significant principal component. Conversely, the mode that corresponds to the smallest eigenvalue describes the least variance and is the least significant principal component.

The covariance matrix is computed as

$$S = \frac{1}{N}\sum_{k=1}^{N}(z_i - \bar{z})(z_i - \bar{z})^T \tag{4.6}$$

where $\bar{z}$ is the mean and is defined as

$$\bar{z} = \frac{1}{N}\sum_{i=1}^{N} z_i \ . \tag{4.7}$$

The eigenvalues, λ_k, and eigenvectors, p_k, are found by

$$Sp_k = \lambda_k p_k \ . \tag{4.8}$$

Each principal component gives a mode of variation to the shape being described. These modes allow the landmark points to move as the shape varies. As stated earlier, the eigenvalues identify the variance that each mode explains. However, not all the modes are needed to describe the shape. The number of modes can be chosen such that a significant percentage of the variance is represented. This reduces the dimensionality from *2n* dimensions to only *t* modes. In doing this we also reduce the amount of variance represented. As a result, the accuracy decreases. It now becomes a tradeoff between accuracy and compactness. We can assume that the modes with low variance only contain noise and can therefore be disregarded. The total variance is computed as

$$\lambda_{T=}\sum_{k=1}^{N}\lambda_k \ . \tag{4.9}$$

A shape can be generated by adding a linear combination of the selected modes to the mean shape, that is

$$z = \bar{z} + Pb \tag{4.10}$$

where $P = [p_1, p_2, \ldots, p_N]$ is the matrix of the chosen eigenvectors, and $b = [b_1, b2, \ldots, b_N]$ is a vector of weights.

By varying the weights, *b*, we can generate new shapes that are not present in the data set. However, constraints must also be set so that the formed shapes still conform to the shapes from training. A common limit is set at three standard deviations from the mean or

$$-3\sqrt{\lambda_k} \leq b_k \leq 3\sqrt{\lambda_k} \ . \tag{4.11}$$

This is chosen because the variance, σ^2, contained in each mode is equal to its corresponding eigenvalue, λ, and the range covered by $\pm 3\sigma$ is already 99.73% of the distribution [**33**]. However, this simplification allows all elements of the vector b to assume the value of $\pm\sqrt{3\sigma}$ at the same time. An improvement to this is to choose the elements of the vector b such that the Mahalanobis distance (D_m) is less than a suitable value, D_{max} [**8**]:

$$D_m^2 = \sum_{k=1}^{N} \left(\frac{b_k^2}{\lambda_k} \right) \leq D_{max}^2 \ . \tag{4.12}$$

From the training set of training images, we derive three modes corresponding to three elements for vector b. Figure 4.5 shows the new shapes formed by setting the first element of vector b to different values while keeping all the other elements to 0.

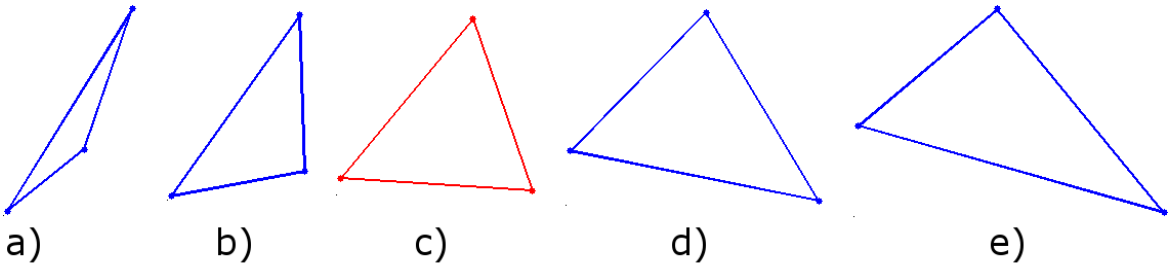


Figure 4.5 Deformations of the mean shape resulting from modifying b_1 by $-3\sqrt{\lambda_1}$ (a), $-1.5\sqrt{\lambda_1}$ (b), 0(c), $+1.5\sqrt{\lambda_1}$ (d), and $+3\sqrt{\lambda_1}$ (e).

4.4 ASM image search

With the models already built up, it is now possible to find the desired shape in unknown images. This phase is generally known as image search. The process starts with positioning the model to an initial location in the test image. Several methods can be used to accomplish this. The simplest of which is placing the model in an arbitrary spot in the image. Ad hoc methods [**32**] have also been used to improve the initial placement of the model. For general applications, genetic algorithms have been employed for this purpose [**18**].

From its starting position, the model moves, rotates, resizes and deforms until it finds the shape it was designed for. This is done in an iterative scheme hence limits are put in place to keep the algorithm from collapsing. Each iteration involves four steps:

1. Calculate best movement locations for each model point.
2. Align model to points computed from step 1.
3. Deform model to accommodate difference between aligned model to best movement locations.
4. Update parameters.

4.4.1 Selecting the movement location for each model point

In the iterative process of moving the model towards desired shape in the image, we need to identify the best points for which to move each model point. This involves analyzing the area around all landmarks and then computing the optimal location for each one. In this thesis, the search area for moving each point is *m* pixels on either side of the model point along the normal. If the model points are trained to be at the shape boundaries, this method allows each landmark to move closer to the shape edges only at a path that is normal to the shape model. This is illustrated in Figure 4.6; model point M is only allowed to move along the dashed line. Because the normal is directly affected by the orientation of the other model points, the best location for a certain model point may not necessarily fall within the normal passing through that point. In the case of Figure 4.6, this is not the problem since point B lies in the normal passing through point M. If point B did not lie on the normal, model point M cannot reach it on that iteration. In effect, more iterations are needed to reach the best location. A logical improvement to this method is expanding the search area from a 1D profile into a 2D area [**44**].

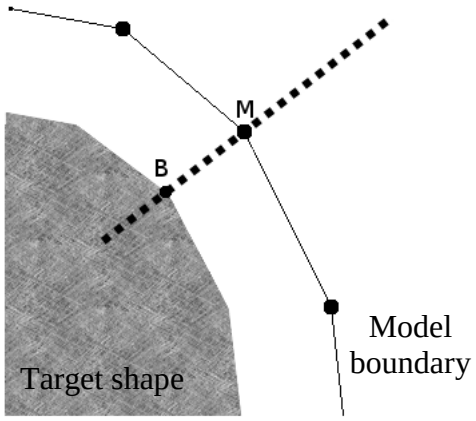


Figure 4.6 Searching for the best movement point along a normal to the model boundary.

The simplest method of best point selection is to use the strongest edge. That is, selecting the location in which the highest peak occurs in the derivative of the profile. This of course makes the point selection susceptible to noise and foreign objects in the image. Also, when the model points are within the shape boundaries, the strongest edge approach does not perform well.

A more robust alternative is to use gray level information built up from the training set. In this method, gray-level data is collected for every model point in each training image. A statistical database is then created from this. More details about the formation of the gray-level statistical description are discussed in Appendix A. During image search, a subset of the profile taken at each model point is compared to the gray level database. The center of the subprofile is the trial movement position. The Mahalanobis distance [**25**] between the subprofile and the corresponding gray level covariance matrix is computed. The point at which the distance is least marks the best update position for the model point. This ensures that the algorithm selects points that are in accordance with the gray level information for each landmark in the training set. As a result, the effects of noise and foreign objects are minimized.

4.4.2 Alignment

The current position of the model, *Z*, is compared to the suggested movement points, computed from the previous step. We then introduce transformation parameters to *Z* in order to map it to the suggested points, similar to the alignment process done in building the shape model.

4.4.3 Calculating model deformation

Aligning the model to the suggested best points is often not enough to capture the desired shape in an unknown image. After alignment, there are still differences that exist between the aligned model and the suggested points. These differences can only be compensated by deforming the shape model but only according to the shape variations seen during training.

The position of the model, *Z*, after the alignment stage is given by

$$Z = M(s,\theta)[z] + Z_c\ , \tag{4.13}$$

where Z_c is the center of the model location. We need to introduce an adjustment parameter, *dZ*, to this such that the points move closer still to the suggested points. We use (4.13) and add *dZ*

$$(Z + dZ) = M\big(s(1+ds),(\theta+d\theta)\big)[z+dz] + [Z_c + dZ_c]\ . \tag{4.14}$$

This becomes

$$M\big(s(1+ds),(\theta+d\theta)\big)[z+dz] = M(s,\theta)[z] + dZ - dZ_c\ . \tag{4.15}$$

Finally solving for *dz* yields,

$$dz = \Big[(M(s,\theta)[z] + dz - dZ_c)M\Big(\big(s(1+ds)\big)^{-1}, -(\theta+d\theta)\Big)\Big] - z. \tag{4.16}$$

Equation (4.16) allows the model to move to the suggested points. Directly using this formula, though, forms shapes that are not necessarily consistent with the shape model built during the training phase. For this reason, we cannot immediately use *dz* to deform the model. We introduce another vector, *db*, to approximate *dz* that only deforms according to the variations seen in training. Starting from (4.10), we find that

$$z + dz \approx \bar{z} + P(b + db)\ . \tag{4.17}$$

Thus, solving for *db* gives us

$$db = P^T dz\ . \tag{4.18}$$

4.4.4 Updating parameters

The parameters computed above bring the model closer to the target shape in the unknown image. We use the computed parameters in the present iteration to scale, rotate, translate and deform the model. This becomes the initial pose and shape of the model for the next iteration. This demonstrates the iterative nature of the ASM search algorithm. Updating the parameters is shown as:

$$Z_C \rightarrow Z_C + dZ_C \quad (4.19)$$
$$\theta \rightarrow \theta + d\theta \quad (4.20)$$
$$s \rightarrow s + sds \quad (4.21)$$
$$b \rightarrow b + db\ . \quad (4.22)$$

This update scheme treats all the elements of the adjustment vectors, dZ_C, $d\theta$, ds, db, with the same weight. A weight vector, w, can be multiplied to each adjustment vector such that elements that deviate more from the training set is given more importance. This introduces the benefit of accommodating large variations of certain points during image search if it was also seen during training.

Finally, the deformation of the model is checked so that it only deforms within certain limits. That is, we verify that the vector b lies within a hyper ellipsoid about the origin. This can be determined by computing the Mahalanobis distance, D_m, and confirming that it is less than a chosen D_{max}. In cases where the computed $D_m > D_{max}$, we scale b to stay within the limits and is computed by

$$b_k \rightarrow b_k \left(\frac{D_{max}}{D_m}\right), k = 1, \dots, N. \quad (4.23)$$

This ensures that the new model still conforms to the training information.

The new pose and shape of the model is compared to the location and form at the start of the iteration. Convergence is achieved when no significant change is recorded between them.

4.4.5 Multi-resolution method for image search

The ASM search method involves taking an 1-D profile at every model point. Sections of the profile are compared to the gray-level model to find the best movement location for each model point. The length of the profiles, therefore, becomes an important part in the algorithm. Selecting long profiles provides for a longer search area, thus, making it easier to "see" the best point for every landmark at each iteration. This has the disadvantage of making the process more computationally expensive and prone to error. Opting for short profiles, on the other hand, shortens the duration for each iteration since fewer comparisons have to be made at each model point. This reduces the area that the search covers. As a result, more iterations are needed to move the model points at the correct positions.

A good strategy is to use a multi-resolution approach [**9**]. We create several copies of the image in different resolution levels by employing Gaussian smoothing and subsampling. In

effect, we build a multi-resolution pyramid. With this, we allow the model to search large areas and move greater distances in the coarse levels. As we go up to the finer resolution levels, the search area is reduced. Therefore, the model is only made to move at small increments as it makes minor adjustments to capture the target object.

At level $l = 1$ of the pyramid lies the original image which contains the highest resolution. In order to obtain the image at $l = 2$, we use a 5x5 pixel Gaussian mask to smooth the image and then we subsample at every other pixel. This halves the number of pixels for the next level. We keep the number of pixels the same at all levels by resizing them back to the dimensions of the original image. The process is repeated until the desired number of resolution levels is achieved. In Figure 4.7 we use a more complex image to highlight the effects of representing an image in several resolution levels. As we progress through the levels, the amount of aliasing that occurs increases. As a result, details of the image are lost with every level as seen by the worsening pixelation.

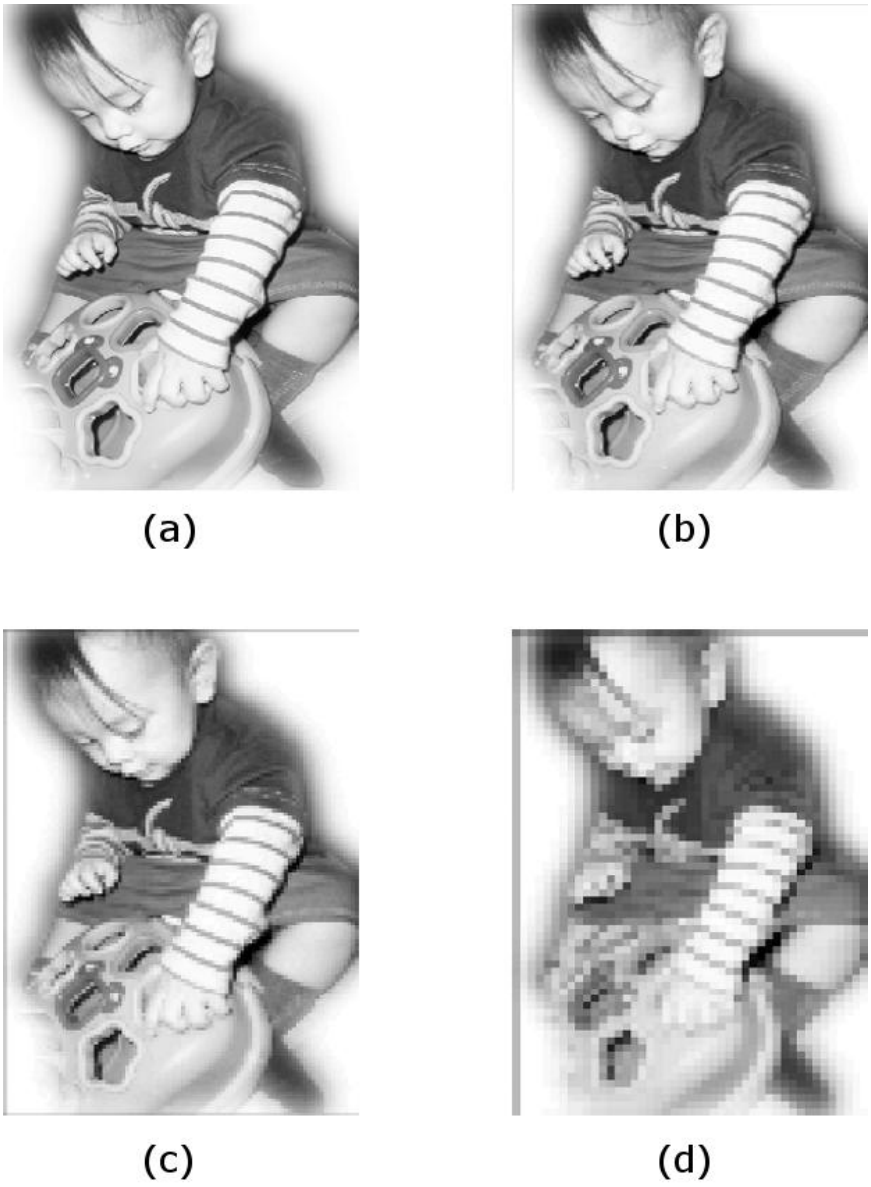


Figure 4.7 One image represented in different resolution levels. (a) Original image- Level 1. (b) Level 2. (c) Level 3. (d) Level 4.

In order to implement this approach, we need to build the gray-level models at each level of the multi-resolution pyramid. We perform Gaussian smoothing and subsampling to every training image. Then the resulting images from the same resolution level are used to build the ASM model for that level. In this thesis, three levels of the multi-resolution pyramid are used.

At every level we build a gray-level database for each landmark in the shape. However, capturing the actual pixel values at each profile directly makes the statistical analysis susceptible to noise and image acquisition conditions. Therefore, we perform differentiation and normalization on the profiles.

During image search, the unknown input image is decomposed into three levels similar to the *l* levels in training. Since we keep the same size for the images in all levels, the pixels of the image in *l + 1* cover an area that is twice as much as the pixels of the image in level *l*. Nevertheless, we still use the same number of pixels in each profile regardless of the level it is in. This allows the model to move larger distances in the coarse levels while in the finer levels, it can only make fine adjustments. We run *n* iterations at level *l* and use the final pose and location of the model as the initial state of the model as we start the algorithm again in level *l - 1*.

Figure 4.8 shows how the profile behaves as we progress through the resolution levels, each bar represents a pixel in the profile. Each pixel in Level 0, for example, occupies two pixel positions in the next level due to the Gaussian smoothing and subsampling. And this repeats as we go from one level to the next. If we take only seven pixels per profile and mark the bars as shown in Figure 4.8, we notice that as we progress through the levels, the pixel values that we get appear more similar. It will finally get to a point where the pixel values that appear in the profile are the same. This does not help the image search technique since it is based on finding the minimum Mahalanobis distance from the test subprofile to the training profile. A solution to this is to skip pixels on the normal during profile acquisition. In that way, we can obtain profile information that has enough variation to be able to make reliable calculations. More details on acquiring pixels for profiles is given in Appendix A.

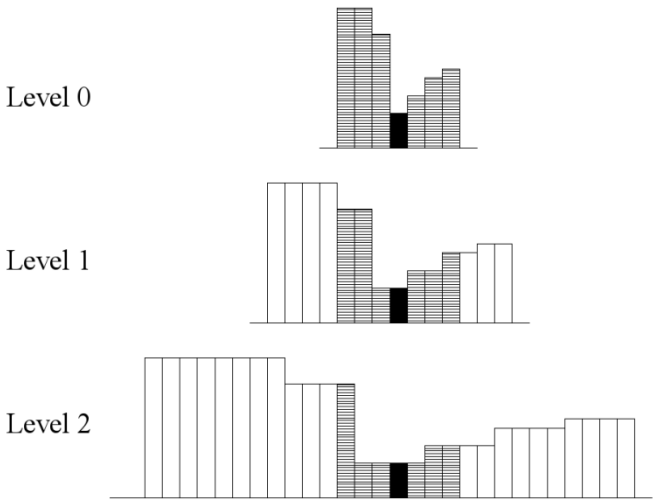


Figure 4.8 One gray-level profile at different resolution levels.

4.5 Conclusion

In this chapter we have gone through the training of the active shape model and its application in image search. During training, a point distribution model is built by recording landmark positions of the shape in the training images. We eliminate similarity transformation by removing scale, translation, and rotation effects from the landmark data. This is done in the shape alignment step. Once only shape information is left, we extract statistical information about the variations that can be seen for each model point. From this information, we create a model that can be deformed according to the variations seen in training.

The built model is used to find the target shape in an unknown image. It is first placed in an arbitrary location in the test image. A line normal to the model boundary and passing though model point *p* is used to compute the best movement position for *p*. This is done for all the *P* points in the model. The model is then rotated, scaled and translated to best fit the suggested points. At this point, differences between the suggested points and the repositioned model can only be resolved by deforming the model, but only within limitations to keep the model from being altered into a different shape. The algorithm is completed if no significant disparity is seen between the initial and final pose and shape of the model. Otherwise, another iteration is done to bring the model closer to the target shape in the unknown image.

A multi-resolution approach is employed to improve the performance of the ASM search algorithm. In training phase, each image is decomposed into several resolution levels. A shape model is built for every resolution level extracted in training. During image search, the unknown image is decomposed into the same number of resolution levels. The search is started at the coarsest level. The resulting pose, shape and location of the model are used as the initial settings for the search in the finer level. The final result of the image search is the pose and shape at the finest level.

5 ASM Implementation and Results

This chapter describes the setup used in the experiments to test the ASM algorithm. We identify the specifications of the computer that the algorithm was tested on. The images used for testing are also briefly discussed. Finally the results of the ASM implementation are shown and examined.

5.1 Computer specifications

The ASM algorithm was implemented in MATLAB R2010a running on Ubuntu Linux 10.04. This was installed on a laptop computer with a 1.80 GHz Intel Core 2 Duo processor with 2 GB of memory.

5.2 Test and training images

All the images used for this thesis were taken by the Marel SensorX machine in several different instances. The images are top-view X-ray captures of chicken breast pieces. Normally the entire chicken breast is cut into a shape called a butterfly. It is called such because the left and right portions of the chicken breast resemble the wings of a butterfly. In this application, only the images with portions on left side of the butterfly are used. Images with the portion on the right side of the butterfly are flipped and rotated to match portions on the left. This procedure increases the number of images for experimentation and also reduces the number of variables to be considered since "flipped" images are eliminated. Training images are selected such that there is only one object in each image and the entire fillet is completely within the limits of the image.

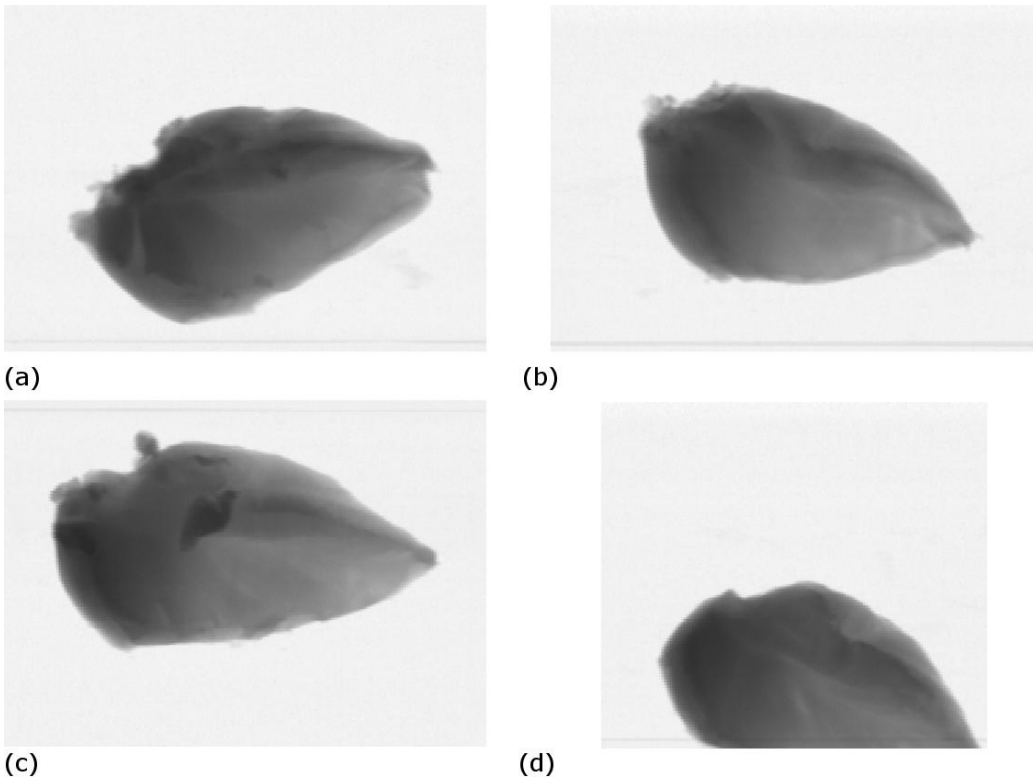


Figure 5.1 Some samples of training ((a) & (b)) and test((c) & (d)) images.

A total of fifty training images were used. Figure 5.1(a) and Figure 5.1(b) are examples of this. It is worth noting the differences in appearance between the two images. Even with size disparities, each training image was manually landmarked with 90 points. It falls upon the person performing the landmarking to carefully select the points such that each point reflects the same location across all the training images. For testing, a different set of images are chosen. Most of the test images are similar to Figure 5.1(c), with the object adhering to the criteria used for the training images. However, some images were selected outside of these conditions. This was done to evaluate the performance of the ASM algorithm under less than ideal conditions.

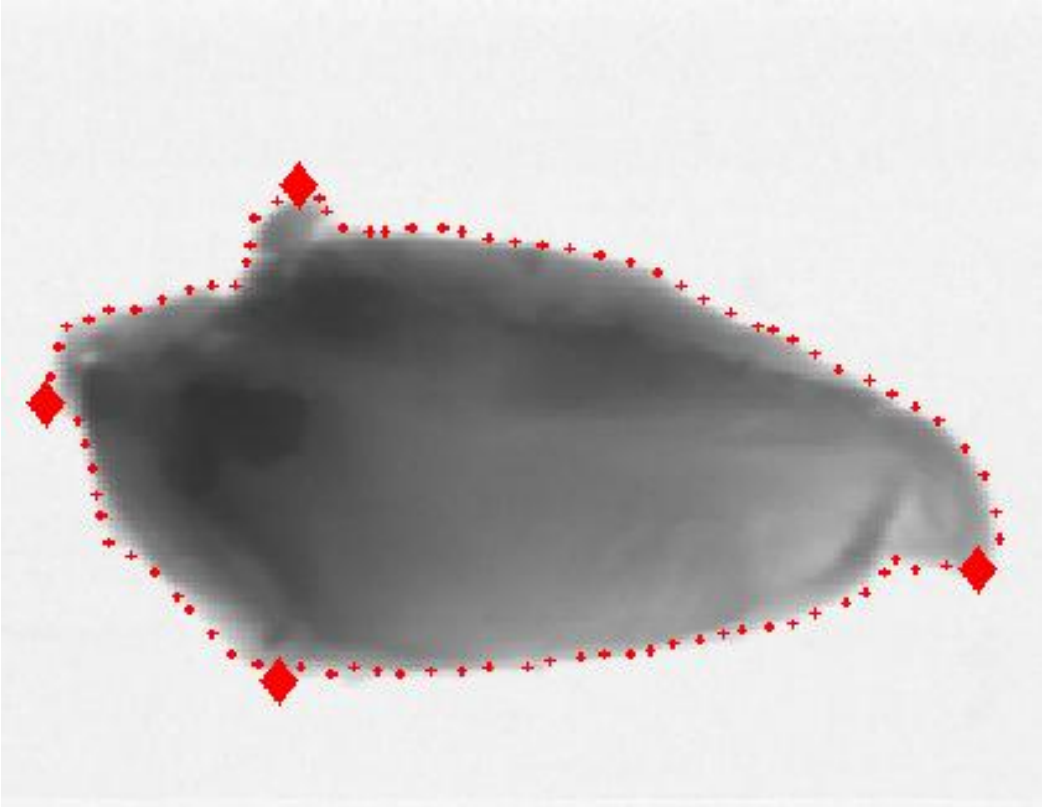


Figure 5.2 A manually landmarked chicken breast fillet.

Expectedly, the landmarking process for this type of images is more difficult from the ideal shape we have seen in Chapter 4. Figure 5.2 shows one training image that was manually labeled with ninety points. A quick look at the image reveals few salient features that can be used as key positions; those of which are marked with a diamond indicator. Several points are placed in between these to catch the contours better. If we take a different image of another chicken fillet, the points may well be differently located from the image above. This illustrates the difficulty with landmarking non-rigid shapes. The irregularity in the edges of the shape present opportunities for having variations in the location of each point when landmarking across several images. This is clearly seen in manual landmarking. Several labeled sets of the same image done by one person can still have some variations. While it is imperative to minimize these discrepancies, it is very difficult to completely eliminate them.

5.3 Alignment results

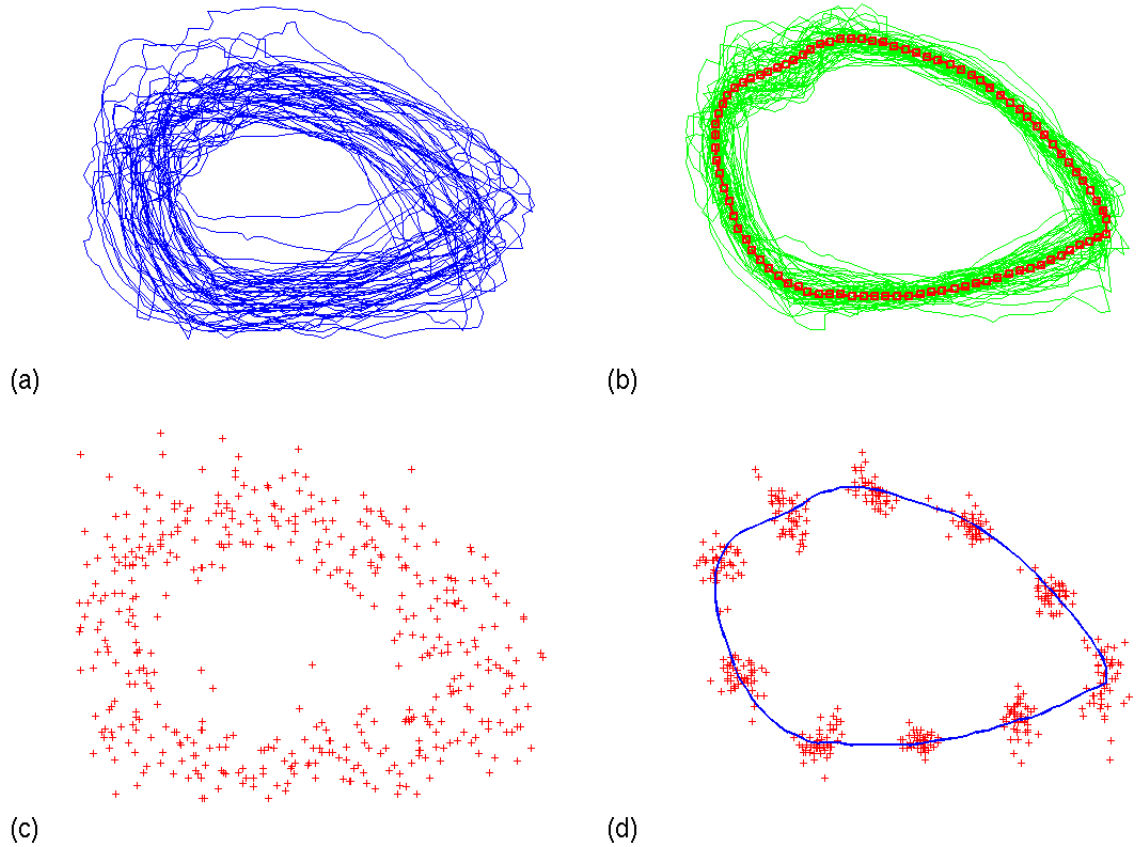


Figure 5.3 Plots of landmarks for the training images before (a) and after (b) alignment and a sample of their corresponding point clouds (c and d).

Figure 5.3 (a) presents a plot of all the training images in their original sizes and orientations. Figure 5.3 (b) shows the result of the alignment process. Even with the quantity of the landmarks and the disparity between the training shapes, the alignment algorithm was still able to perform well. We could see the rough shape of the chicken fillet from the aligned shapes. The superimposed mean then defines this shape more clearly. From the unaligned shapes in Figure 5.3 (a), we select landmarks at regular intervals from each of the training shapes and plot them in Figure 5.3 (c). This is also called a *Point Distribution Model* (PDM). Expectedly, we do not see any pattern arising from this. In Figure 5.3 (d), a PDM for the aligned shapes is taken from the same landmarks used in Figure 5.3 (c). Note that the clusters of points appear more distinct than in the Figure 5.3 (c). However, even though the objects were successfully aligned, there is still a considerable amount of variation in the location of each landmark. Evidently, there are landmarks which have more variation than others. The standard deviation seen in each landmark is shown in Figure 5.4. The most movement is seen at the landmark which corresponds to the point nearest to the end of the sternum. The cut in this area can be highly irregular since it is done against a bony section. This part of the fillet is also thinner, and being in the extremity, is more easily displaced in relation to the rest of the piece. The least deviation, on the other hand, lies on the thickest part of the fillet which is near the neck region. This part of the chicken has more meat leading to more even cuts.

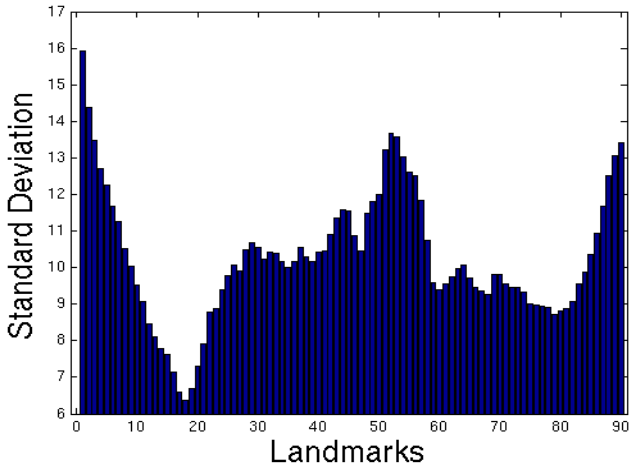


Figure 5.4 Standard deviation in each landmark.

5.4 Statistical analysis results

In Section 4.3.3 we discussed how statistical information is derived from the training set and used to generate new shapes that are consistent with the variations seen in training. In this thesis, there are ninety landmarks taken for each training image. This translates to one hundred and eighty dimensions where every shape moves around in. We explained previously that not all the dimensions or modes are necessary for shape generation. Some modes carry little or no information and thus can be disregarded. In order to identify which of these dimensions are worth taking into account, we use a dimension-reduction method such as PCA.

Table 5.1 The most important principal components from the training data.

Rank	Variance	%
1	4089.8	42.55
2	1524.5	15.86
3	1014	10.55
4	602.2	6.27
5	456.1	4.75
6	288.1	3.00
7	269.5	2.80
8	205.1	2.13
9	153.1	1.59
10	122.8	1.28
11	111.3	1.16
12	105.7	1.10
13	88.1	0.92
14	72.8	0.76
15	70.9	0.74
16	48.2	0.50
17	38.2	0.40
18	36.4	0.38
19	30.9	0.32
20	26.6	0.28

Table 5.1 shows the variances of the twenty most important principal components taken from the training data. The importance of each principal component is scored by taking its percentage against the sum of the variances of all principal components. It can be seen that the differences between the percentages of modes do not vary in a linear manner. In fact, the sum of the variances described by the first three already includes 68.97% of the total variance. We involve a few more components to increase the amount of variance that the model can describe. Using ten modes increases the percentage to 89.51%. This highlights the fact that the information is contained mostly in a few modes. Of course we can always take in more modes to increase the amount of variance that the model describes. However, this would not establish a significant improvement that is enough to justify the consequence of increasing the computational complexity.

By applying (4.10) we can see how the model deforms according to the data set. Theoretically, we can mold the mean model to resemble any of the objects we used in training. The accuracy at which we can copy the training shapes depends on the number of elements in vector *b*. A smaller quantity of elements may be sufficient provided that the target training shape is not too different from the mean shape, since most of the information is contained in the first few modes. However, more components of the vector *b* may be needed to pull the mean shape in accordance to the extreme shapes in the training set.

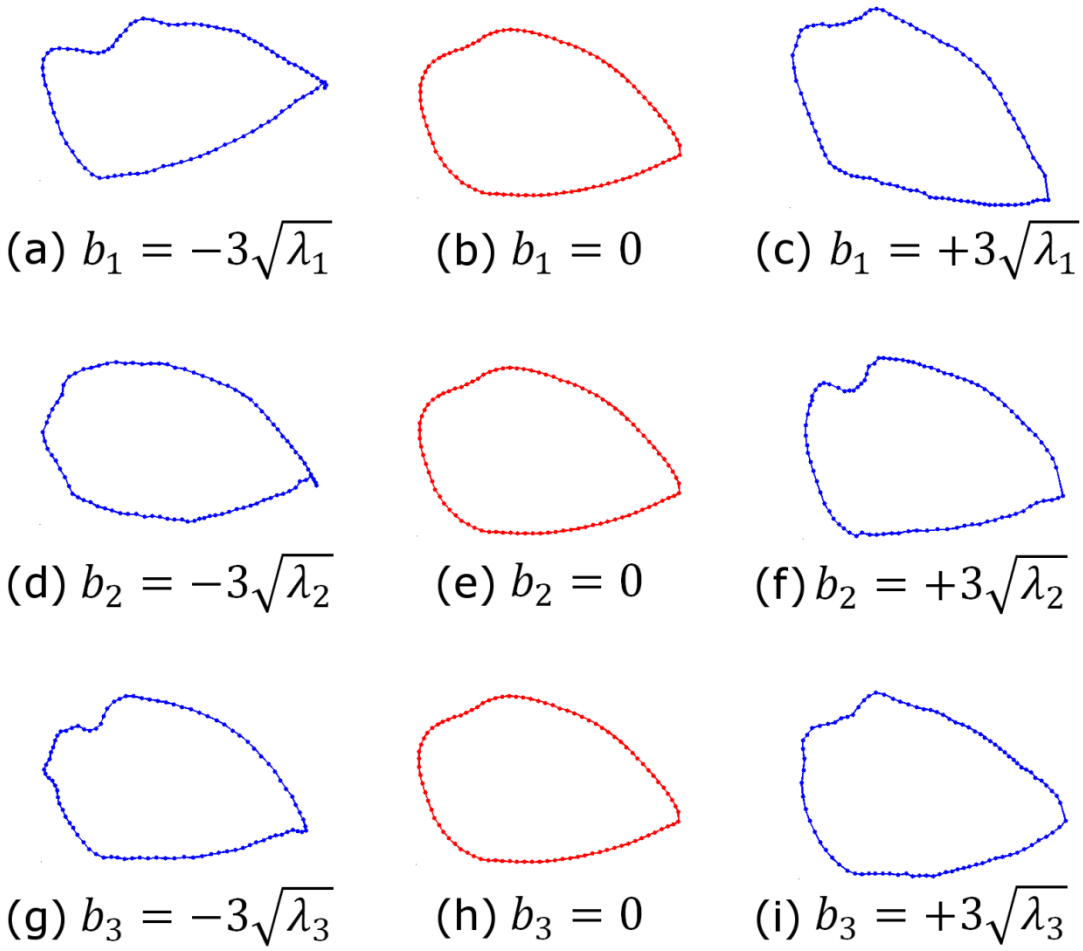


Figure 5.5 Deformations resulting from modifying the elements of vector b.

Figure 5.5 shows some of these deformations. For these images, the elements of b are set to zero apart from one of the first three modes. Only changes in the first three components are shown since they carry the most information and therefore the deformations are more pronounced. Expectedly, the shapes formed by modifying b_1 yielded shapes that are the most distant from the mean.

5.5 Search results

5.5.1 Strongest edge method

The objective of ASM is to find a known shape within an unknown image. In this section, we explore the results of using the ASM on the chicken fillet images. First we investigate the image search using the strongest edge method. We then compare the results to using the gray level model. Finally, we see the outcome of implementing the multi-resolution approach.

Choosing the initial location of the model on the target image before performing image search is not explicitly defined in the ASM algorithm, and is mostly left to the user. In this thesis, we use the same location of the mean during the training phase. Figure 5.6 shows the initial position of the model as it is superimposed on a test image. Clearly in this situation, the choice of starting point is good because most of the target object is already inside the model. From here, the model has to move, rotate, scale and deform in small extents to complete the search.

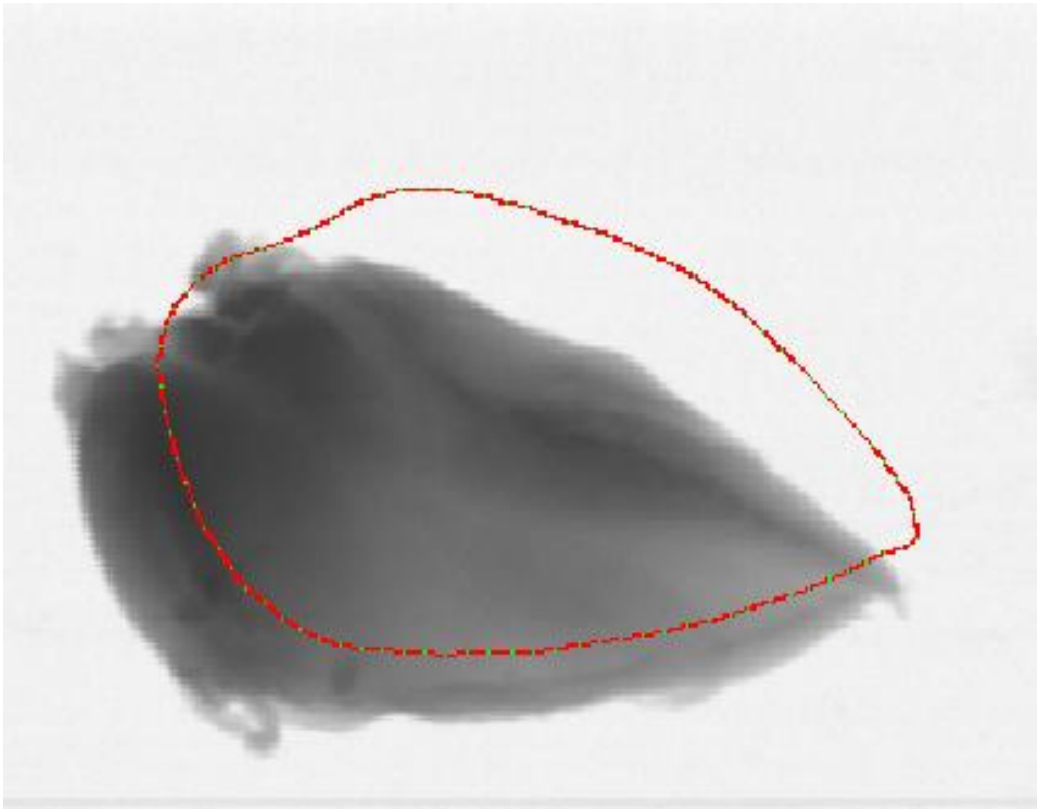


Figure 5.6 Input test image.

At each landmark point from the initial position of the model, profiles normal to the boundary are taken. This is illustrated in Figure 5.7. All the profiles have the same number of pixels contained in each one. Remember that the ideal situation is that the object boundary is contained in most, if not all, profiles. In this way, the algorithm will be able to move and deform the model correctly to the object outline with the least number of iterations. It is worth noting that in Figure 5.7, there is an offset between the locations of the model and the target object. As a result, there are some profiles that do not contain the object boundary. For the same reason, the resulting suggested points from these profiles will not be reliable.

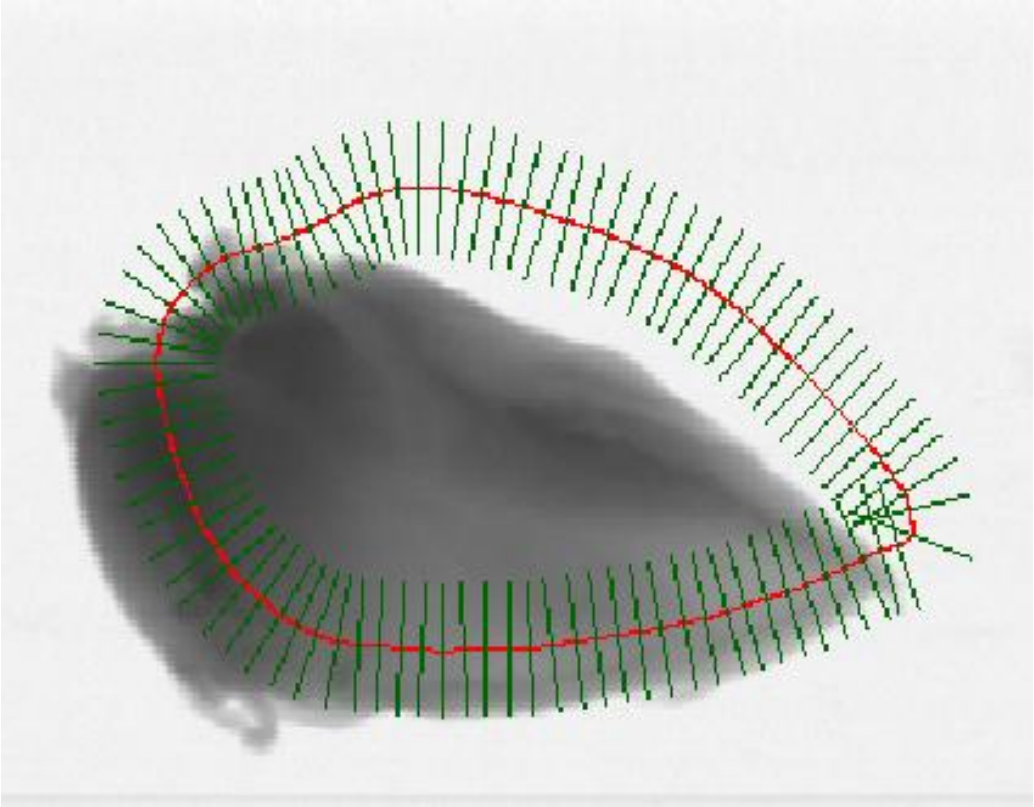


Figure 5.7 Search profiles for first iteration of image search.

Figure 5.8 shows a plot of the results of the best points search for each landmark. We can see that on the profiles where the object boundary was present, the strongest edge method was able to find the edges. This is helped by the fact that the target object is in a mostly clean and white background. Hence, the contrast between the object and its surrounding pixels are high enough for the strongest edge method to easily detect the edge. Occasionally, some profiles will contain more than one object boundary, such as seen on the tail-end of the object on Figure 5.7. The point selected will be the edge that has the higher difference in pixel levels. We see the effect of the method choosing the wrong edge in that situation on Figure 5.8. On profiles that are completely inside the object or on the background, the method will just choose the highest peak in the pixel profile. Obviously, this does not contribute to the end goal of finding the edges of the object. Thus, we rely on the suggested points that have actually found the edge to guide the succeeding stages of the ASM image search in moving, scaling, rotating and deforming the model for the next iteration. Needless to say, if there are more landmarks that grabbed the edges, then there will be less iterations required to complete the image search.

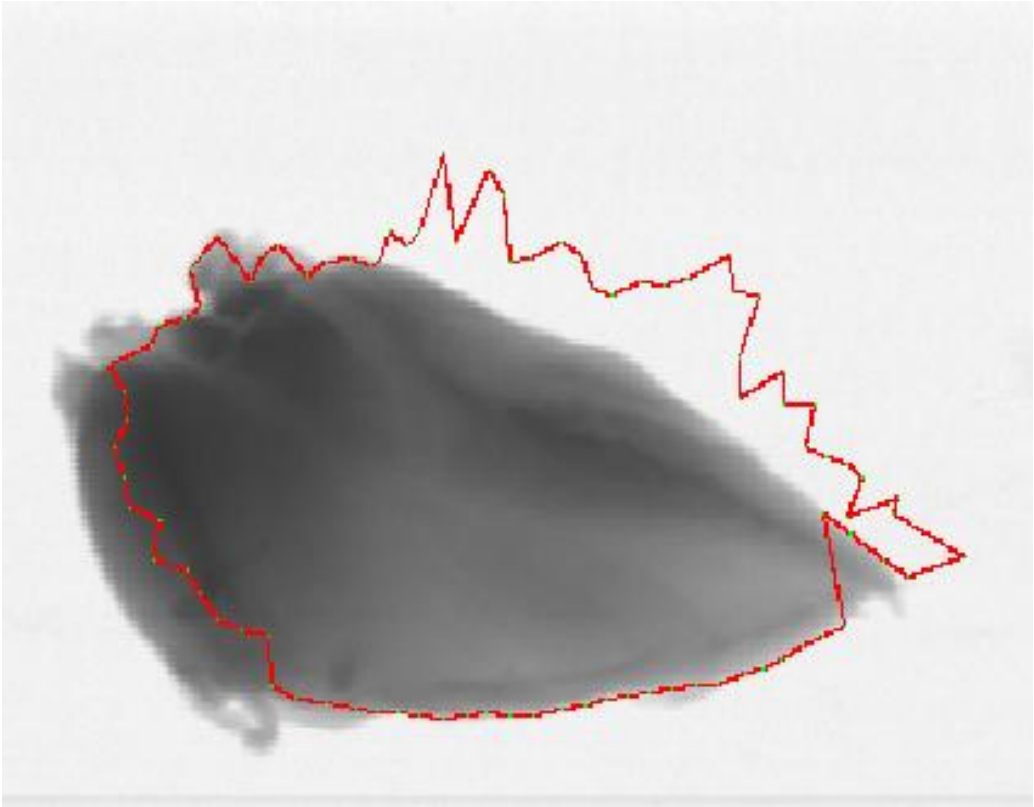


Figure 5.8 Suggested points in the first iteration of the image search.

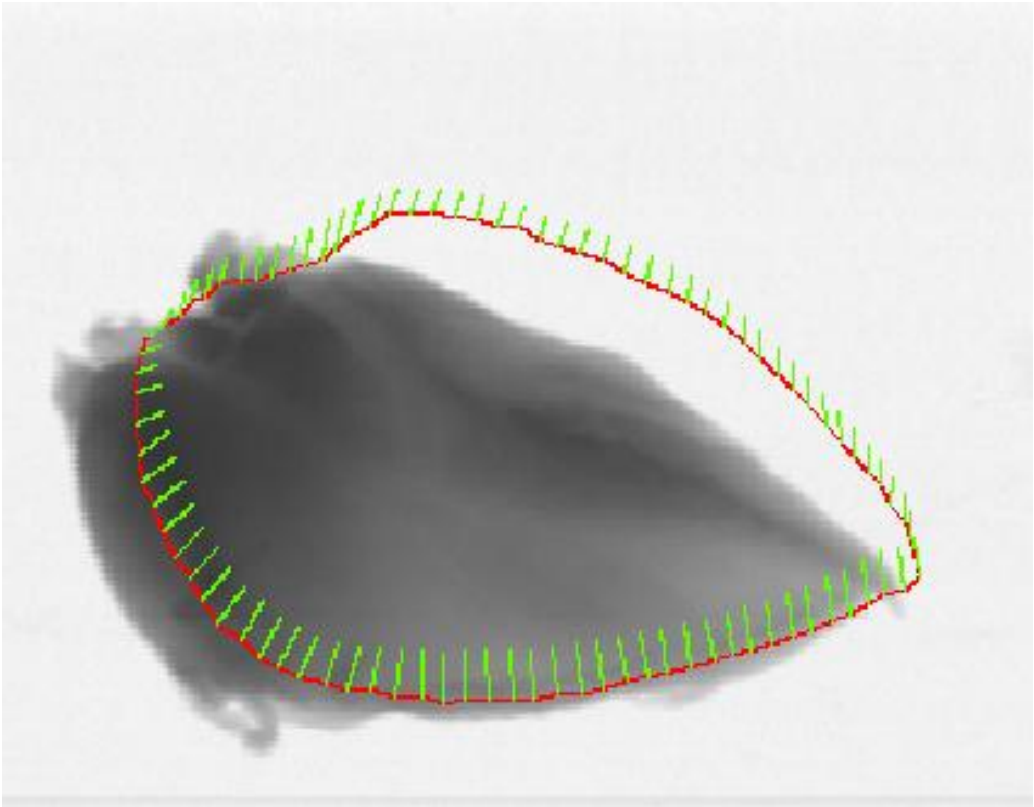


Figure 5.9 The resulting position of the model after the first iteration of the image search.

We move and deform the model to best fit the suggested movement points. Figure 5.9 illustrates the outcome of the first iteration of the image search method. Lines are drawn from the starting position of each landmark to its final position to emphasize the effect of the search technique. We observe that the greatest movement occurred where the model was able to reach the object boundary. The result of the first iteration becomes the starting position for the next one. In this example, it takes seven iterations for the method to complete. The outcome is shown on Figure 5.10. We observe that the model was able to correctly find the target. However, we notice that it is unable to deform to the exact contour of the object. This is not a fault of the strongest edge method. In fact, getting the suggested points from the strongest edge method on this location will capture the object boundary accurately. Nevertheless, the limits set for deformation keeps the model from completely conforming to these points. As a result, the model tries to change according to the suggested points but only within the specified constraints. This is clearly seen in this instance where the target is non-rigid and is therefore its boundary is highly variable. The objective of the algorithm is still accomplished, though, since the location and orientation of the target is found.

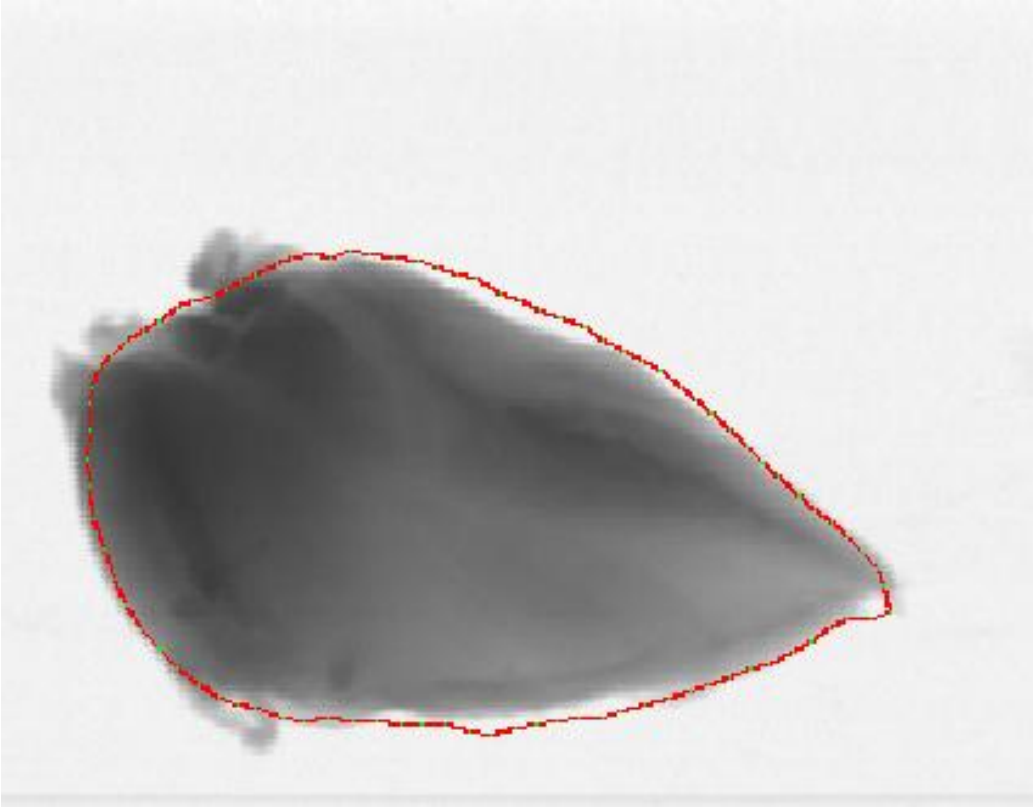


Figure 5.10 Final position of the model after image search using the strongest edge method.

Other examples of image search using the strongest edge method are shown on Figure 5.11. It must be said that the initial location and orientation of the model is the same on all image search trials. We see that on Figure 5.11(a), the model lands outside the limits of the test image. This indicates that the object in the image is actually smaller than the average size in the training set. In this situation, the model must scale down to fit the target. After ten iterations, the algorithm converges at the position displayed on Figure 5.11 (b). The number of model landmarks that were in close proximity to the object boundary and led to

the fast convergence. Figure 5.11 (c) is another image tested with the strongest edge method. In this trial, the algorithm took twenty three iterations to complete. Despite the apparent nearness of the initial location of the model landmarks to the object boundary, the algorithm needed more repetitions. This can be attributed to the shape of the object. The cut in the top portion of the fillet is wider than the average piece in training which caused the algorithm to iterate more to accommodate this difference.

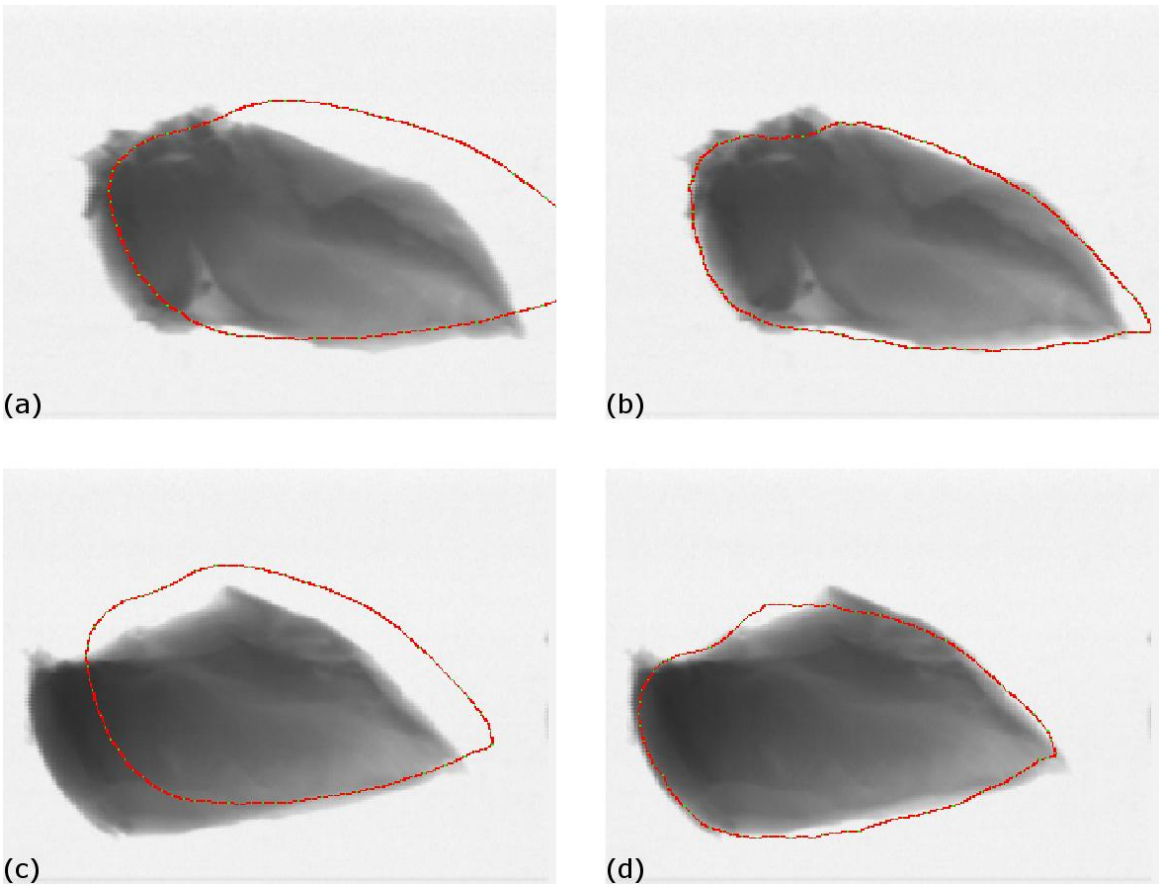


Figure 5.11 The initial (a) (c) and final (b) (d) models of two test images after image search using the strongest edge method.

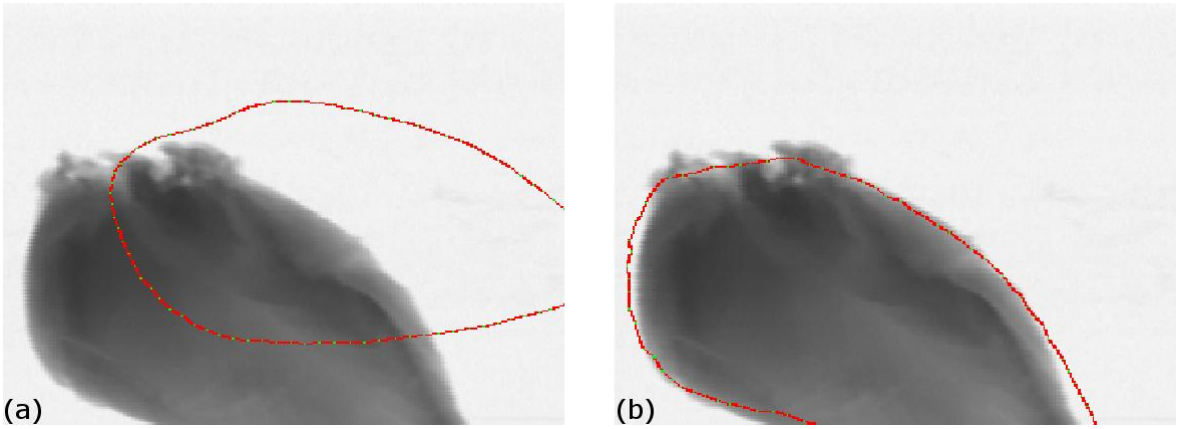


Figure 5.12 Initial model placement (a) and final result (b) of ASM image search with incomplete target object.

Figure 5.12 (a) shows an image with the fillet cut off at the bottom. Evidently, this is an alteration to the actual fillet of which the image was taken with. For the ASM image search, this is another impediment in finding the target object. The images used for training did not include incomplete fillets. It is for this reason that we cannot expect the algorithm to accurately find the object. Looking at Figure 5.12 (b) proves otherwise. We recall that the landmarks in ASM are not completely independent. During image search, each landmark is given its candidate movement point. But the actual position each landmark moves after each iteration is resolved by considering the positions of all movement candidates. This explains the result in Figure 5.12 (b). The model landmarks that were meant to search for object edges in the missing portion of the fillet are compensated by the landmarks that were able to seize object edges present in the image. This implies that ASM image search is able to tolerate missing features on its target object provided that there are enough landmarks that grabbed the object features present on the image.

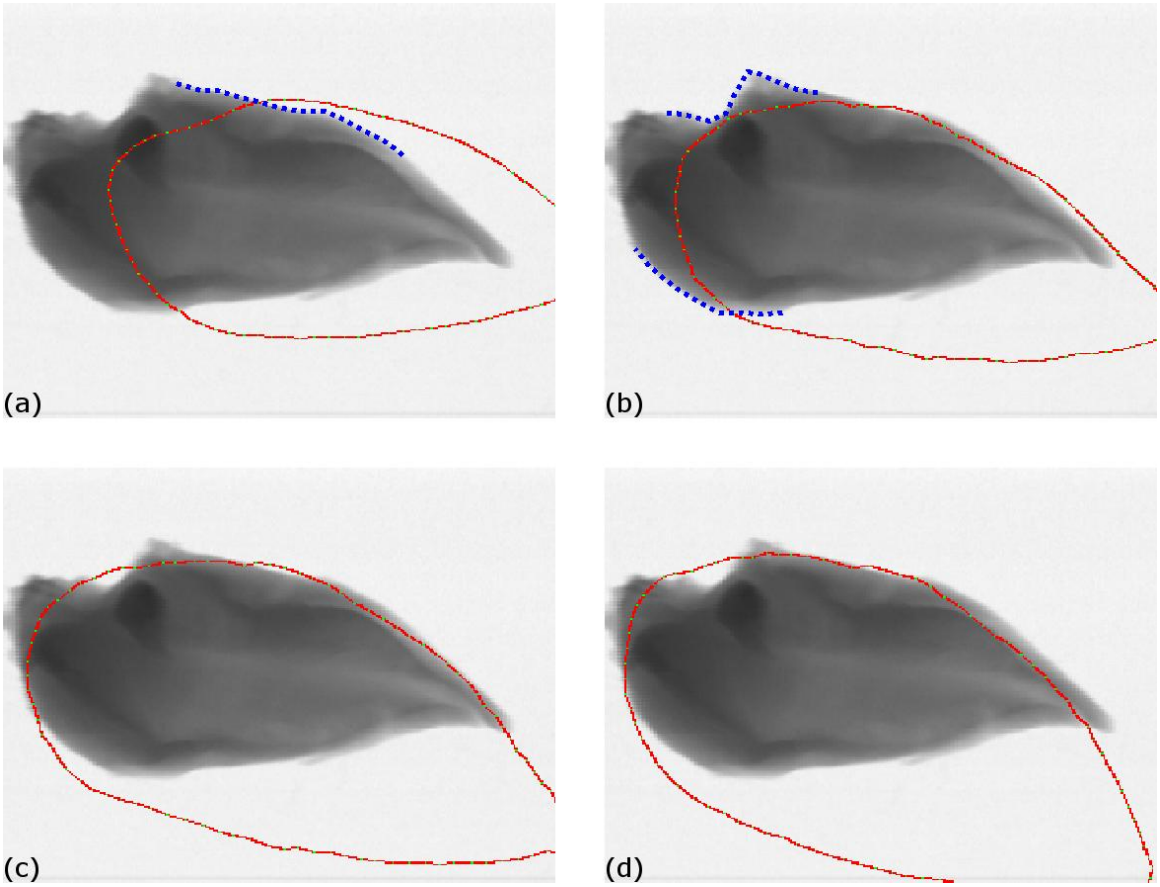


Figure 5.13 Non-convergence of image search because of model placement. (a)Initial model placement, (b)5th iteration, (c) 10th iteration, (d) 500th iteration

Given that in the strongest edge method, the model attaches itself to the points of largest contrast, it is very susceptible to noise and artifacts in the test images. When the model does not encounter any impurities in the image, the method performs well. The examples shown in Figure 5.6 to Figure 5.11 illustrate such instances.

However, in some test images, the method gets stuck not because of some stray object but rather the initial position of the model. In Figure 5.13 (a), we see a model that is positioned at the right side of the target at the start of the ASM image search. Figure 5.13 (d) shows

the result after 500 iterations. The model has scaled and rotated causing it to reach the bottom edge of the image. During the first few iterations, the model can only find the object edges indicated by the dotted line in Figure 5.13 (a). As a result, the model tries to rotate to attach to this edge as revealed in Figure 5.13 (b). In the same Figure, the edges indicated by the dotted line become exposed to the model. The model then rotates to cling to this without losing the edges it has already attached to. The consequence of this sequence of events is that the rotation of the model causes it to misalign from the fillet. The 10^{th} iteration shown in Figure 5.13 (c) bares a model that has rotated such that its tail end has extended towards the bottom edge of the image. Close inspection uncovers a visible gray line just above the bottom edge. As the strongest edge method perceives the line artifact as an object edge, the method will try and attach to it. After five hundred iterations, Figure 5.13 (d) shows little improvement over Figure 5.13 (c), establishing non-convergence at this test image.

The model placement at Figure 5.13 (a) is clearly not the best initial position for the ASM Image search. As a result, the image search reaches non-convergence. If we try and move the model more to the left of the image, we expect to get better results. Figure 5.14 tracks the results as we move the model further to the left. We see that shifting the model to the left from twenty-five pixels to ninety-nine pixels results in the convergence of the image search algorithm. Outside of this window, however, non-convergence still occurs. This shows the importance of placing the model at the right position for optimum results.

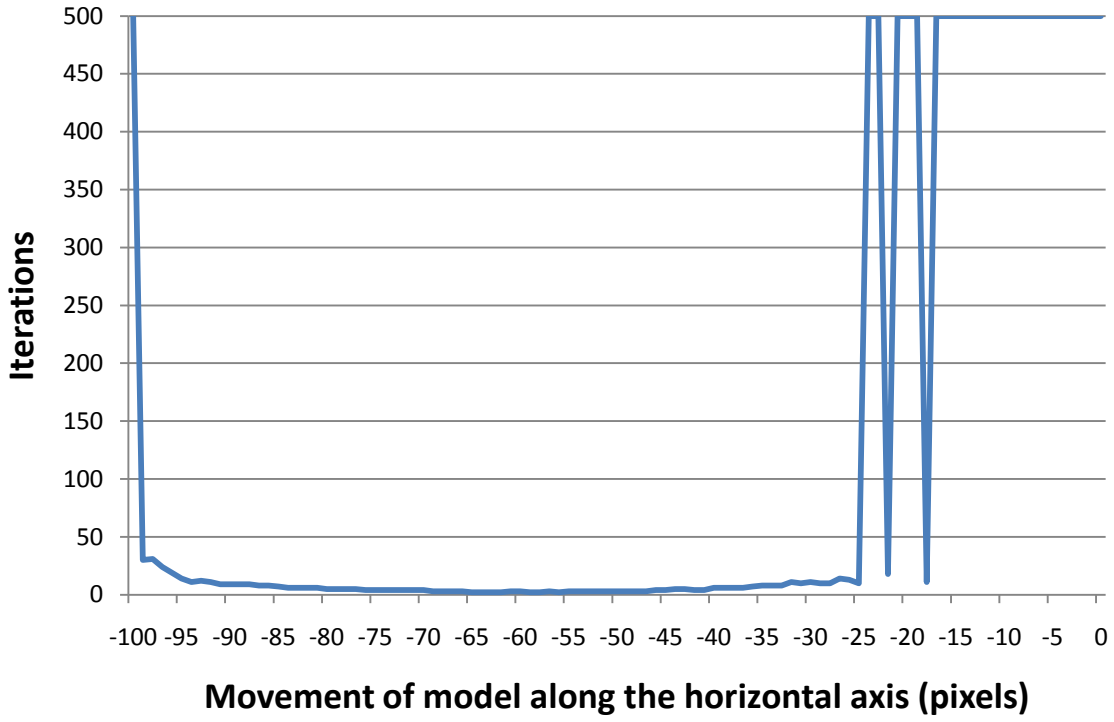


Figure 5.14 The effect of moving the initial placement of the model horizontally to the number of iterations needed to complete the image search algorithm.

5.5.2 Gray-level method

Another method for best point selection is the gray-level search. While the strongest edge method relies only on the highest peak in each derivative profile, the gray-level technique provides the advantage of using the statistical information taken for each landmark during training. This helps the technique to find the point that best resembles the profile of the landmark it was trained on. Figure 5.15 shows the effect of employing gray-level search in the first iteration of the test image in Figure 5.6. We see that the model was able to grab the object edge only on a number of landmarks. If we compare this to Figure 5.8, we see that there is a noticeable similarity. However, it took eight iterations to complete the ASM image search using the gray-level approach. This is one more than with the strongest edge scheme. In the second test image, seen in Figure 5.11(a), implementing the gray-level method took fourteen iterations of the ASM image search. Figure 5.16 (a) and Figure 5.16 (b) show the results of the image search using the gray-level method on the test images of Figure 5.6 and Figure 5.11 (a), respectively. These two instances demonstrate that if the model does not encounter any artifacts or irregularities during image search, using the strongest edge technique actually performs better.

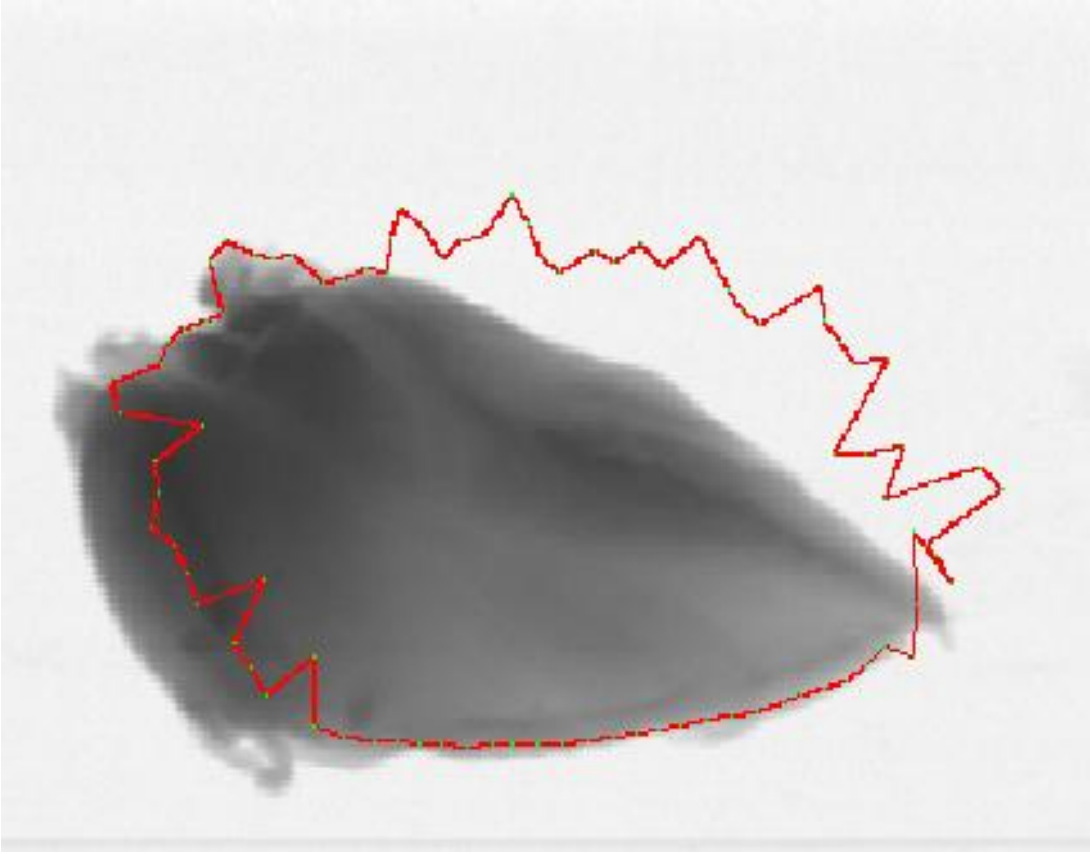


Figure 5.15 Suggested points in the first iteration of the image search.

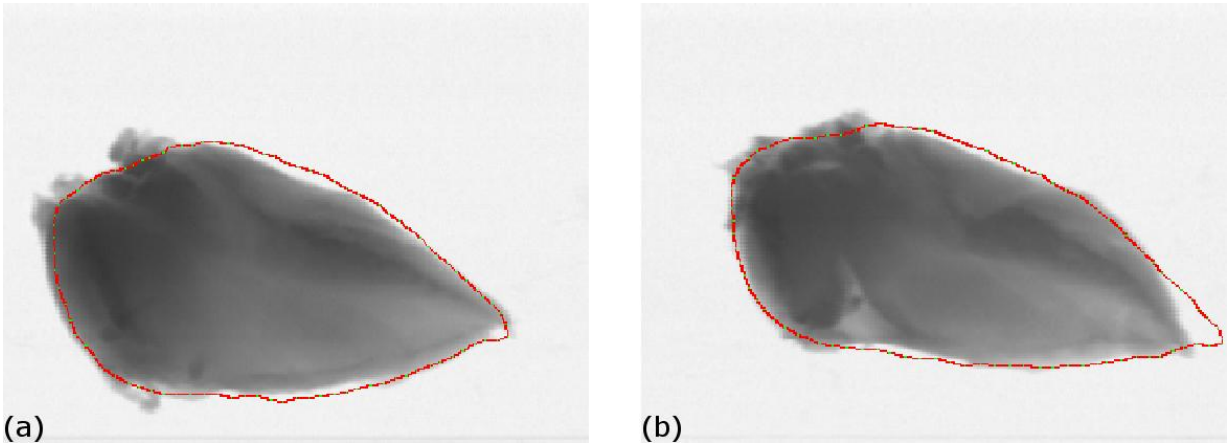


Figure 5.16 Final results of image search using gray-level method.

The advantage of the gray-level method is that it is less-prone to stray marks and artifacts in the test image. Figure 5.17 shows the result of using the gray-level method in the image search for the same test image examined in Figure 5.13 (a). We immediately notice that instead of getting stuck at the gray line at the bottom of the image similar to the result seen in Figure 5.13 (a), the model was able to correctly locate and bind to the target. The image search was able to converge after just twenty-two iterations.

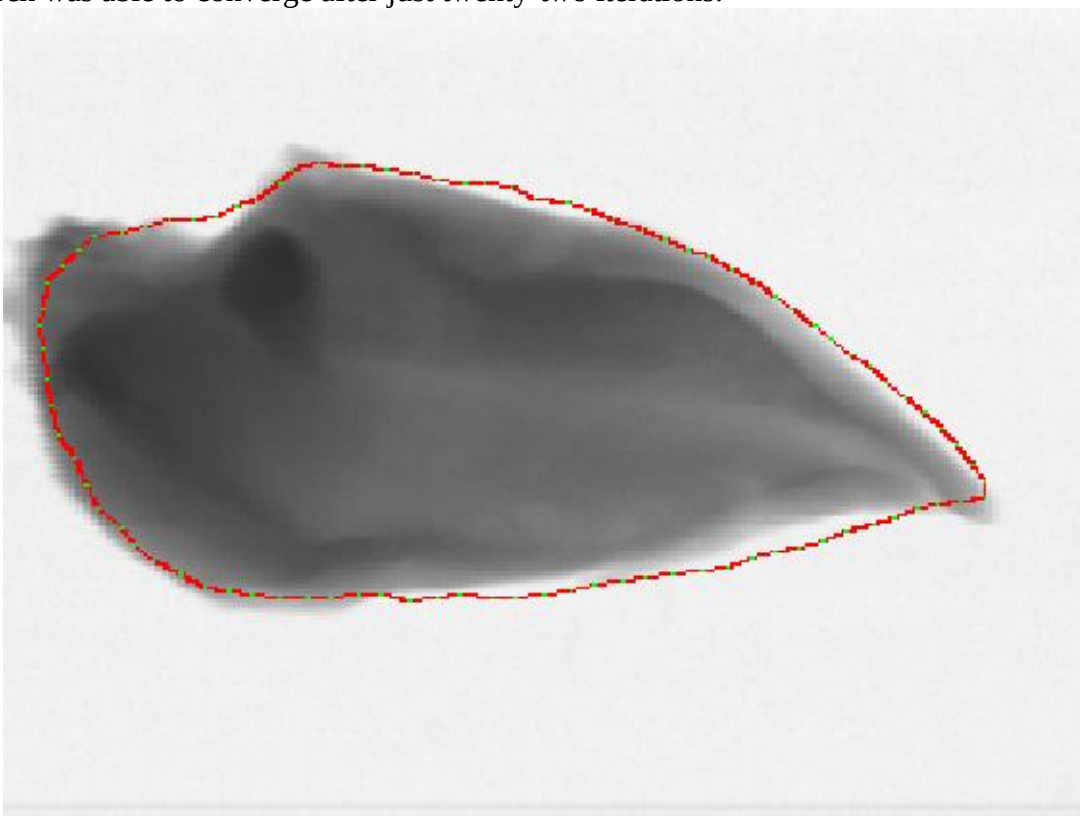


Figure 5.17 The result of implementing image search with the gray-level method for the test image in Figure 5.13(a)

Using the gray-level method to select the best points is not enough to produce useful results. The image search algorithm still needs to have the model be placed in a good initial position to achieve convergence. Figure 5.18 (a) shows one of the test images with the normal initial position of the model. Similar to Figure 5.13 (a), the model is also offset to

the right of the fillet. After reaching five hundred iterations, the result is shown in Figure 5.18 (b). The model was able to grab the upper right edges of the fillet, but it failed to expand to reach the other side. In Figure 5.18 (c), we move the model to fifty pixels to the left of the original initial position. With this small change in the location of the model, convergence was achieved after only four iterations. The result of the image search is shown in Figure 5.18 (d).

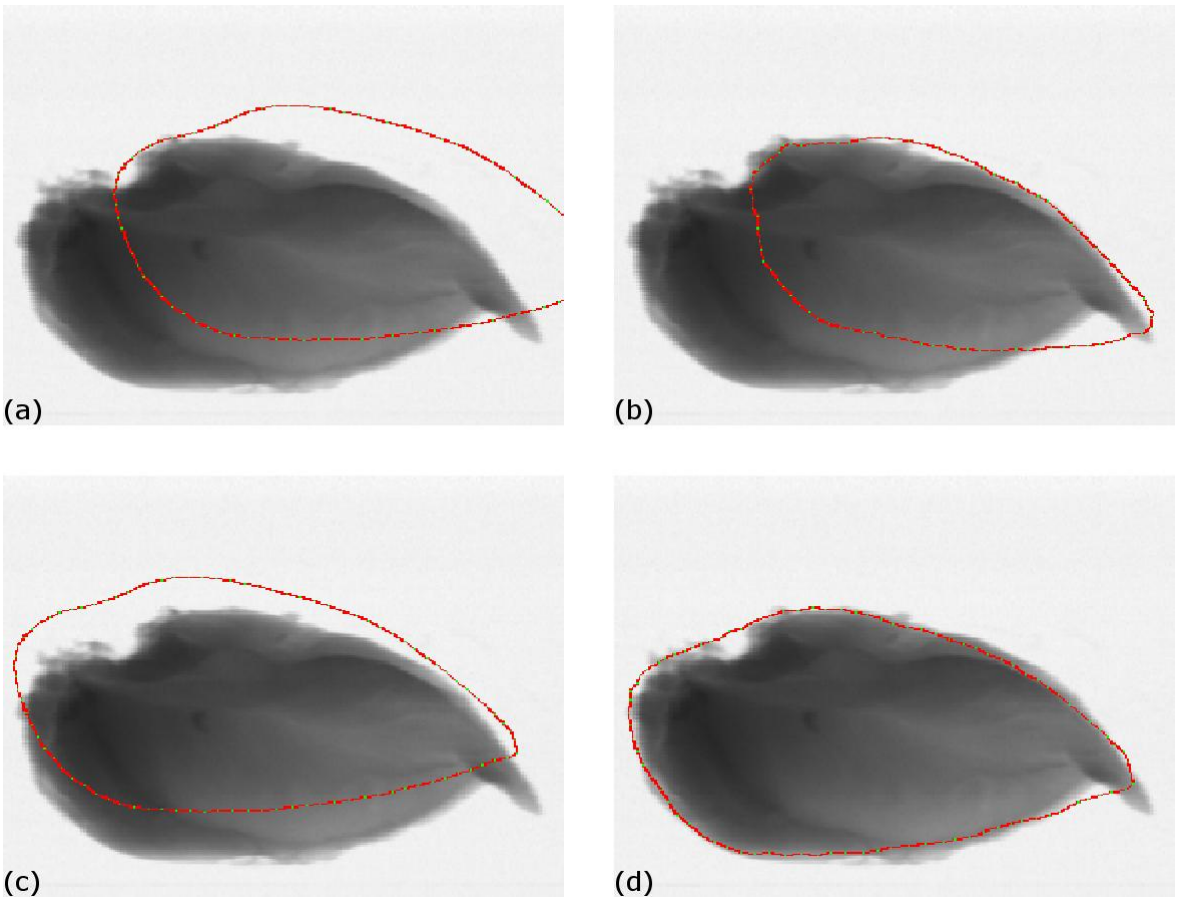


Figure 5.18 Comparing the effect of moving the initial placement of the model to the image search results.(a) Original position, (b) result after five hundred iterations from (a), (c) model moved to fifty pixels to the left of original position, (d) result after four iterations from (c).

Figure 5.19 plots the number of iterations needed for the ASM image search to converge while applying the strongest edge and gray-level methods to a set of ninety test images. In every test image, the same initial position is used for the model. We also set the limit of iterations to five hundred. We assume that once this limit is reached, the algorithm is already stuck in a local minimum and we do not proceed any further. The performance of each method is judged by how quickly the image search completes. Put simply, less iterations indicates better performance.

We notice in Figure 5.19 that the image search algorithm reaches the five hundredth iteration mark in five instances. We further observe that this situation occurred more while implementing the gray-level method of selecting the best points. Again it must be kept in mind, that the performance of the image search algorithm is not defined solely by the point-search method used. The initial model placement, similarity transformation and

deformation also play equally significant roles. The point-search method simply provides the initial direction while the other stages complete the iteration. Non-convergence can occur because of a number of factors. First of which is when the point-search method gets stuck because of a stray edge. When this happens, the model is prevented from getting to the fillet. The other important factor in deciding the success of the image search algorithm is the initial position of the model. The model has to see the object edges for it to move and deform to the right position and orientation. When no edges are seen, there is no way for the model to find the fillet. Also, there should be enough of the model landmarks that see the object edges to pull the model toward the fillet, otherwise non-convergence results.

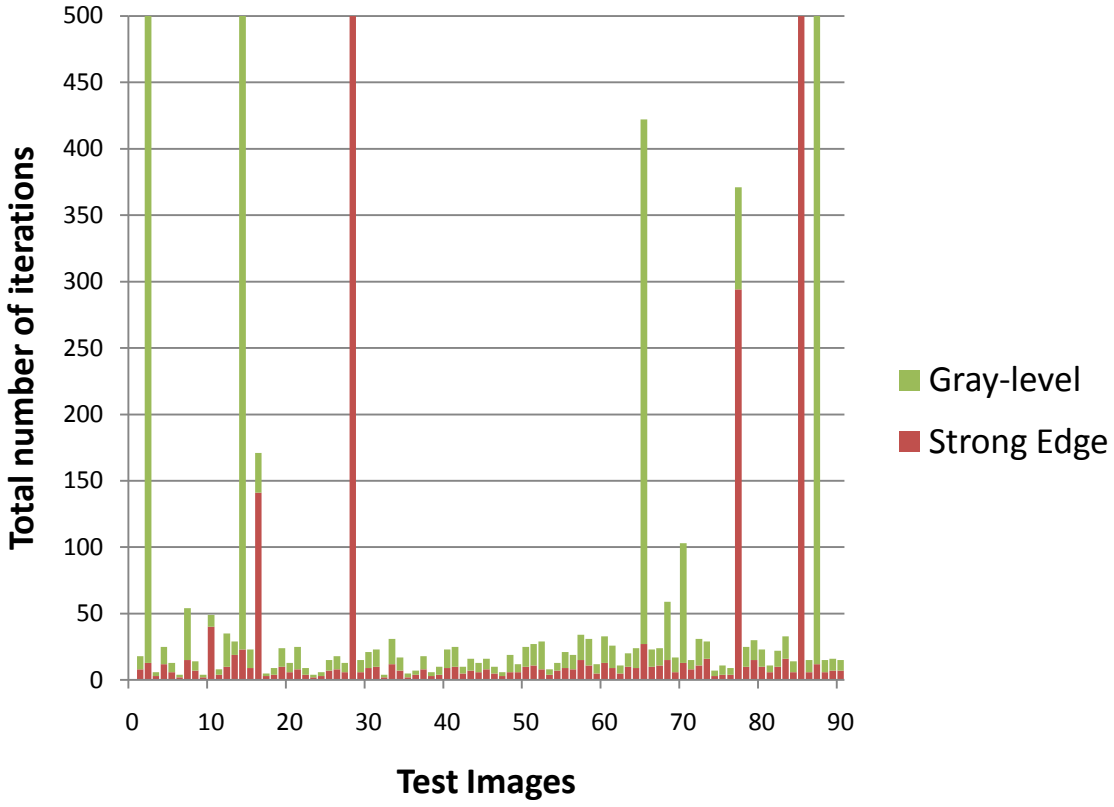


Figure 5.19 A comparison of the plots of the iterations needed by each test image to complete using the strongest edge and gray-level point-search methods.

5.5.3 Multi-resolution image search

We have seen in the previous sections that moving the starting position of the model has a great effect on whether the image search algorithm reaches convergence or not. Figure 5.13 and Figure 5.18 demonstrate that neither point-search method is immune to this. It is then that we explore the multi-resolution technique. The main advantage of using the multi-resolution technique in the image search algorithm is it eases the burden of placing the initial position of the model correctly to achieve convergence.

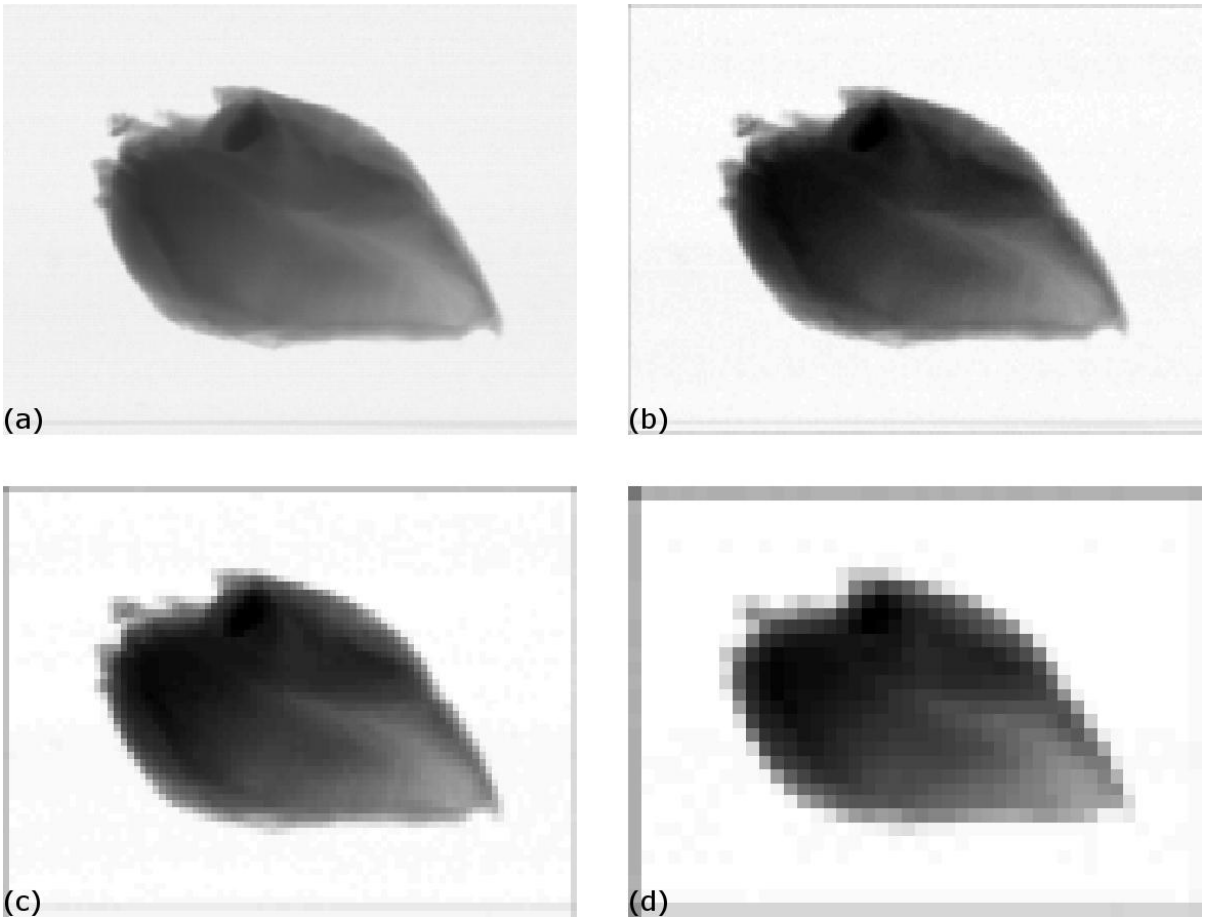


Figure 5.20 Multiresolution representation of a chicken fillet image. (a) Original image-Level 1. (b) Level 2. (c) Level 3. (d) Level 4.

Figure 5.20 (a)-(d) shows one of the test images in different resolution levels. We can clearly see the difference in resolution as we progress through the levels, from the original image seen in Figure 5.20 (a) until the 4^{th} resolution level in Figure 5.20 (d). In the image search, experiments, we focus on the gray-level point-selection method. Again it is stressed that the original initial position used in the previous experiments has been retained in the multi-resolution trials.

We look back at the test image used in Figure 5.18 (a)-(d). In Figure 5.18 (b) we saw that the image search resulted in non-convergence. But moving the model a few pixels to the left caused the model to correctly find the fillet in the image. In Figure 5.21 (a) we use the same image and keep the original position of the model and we apply the multi-resolution technique. Performing the image search in the coarsest resolution, at multi-resolution Level

3, the image search algorithm at this level converges after completing just six iterations. The result is shown in Figure 5.21 (b). We notice that the model has already moved a significant distance to the left, surrounding the fillet. Because of this, the image search algorithm needs very few iterations to converge in the succeeding multi-resolution levels. In fact, the Level 2 image search ends in only five iterations, the result of which is seen in Figure 5.21 (c). The Level 1 image search finishes after its first iteration, seen in Figure 5.21(d). We clearly see the differences in the distances travelled by the model in each multi-resolution level. In the coarse resolutions, the model is able to move longer distances while in fine resolution, it only makes subtle adjustments to its position. In effect, the performance of the image search algorithm becomes less reliant on the initial placement of the model.

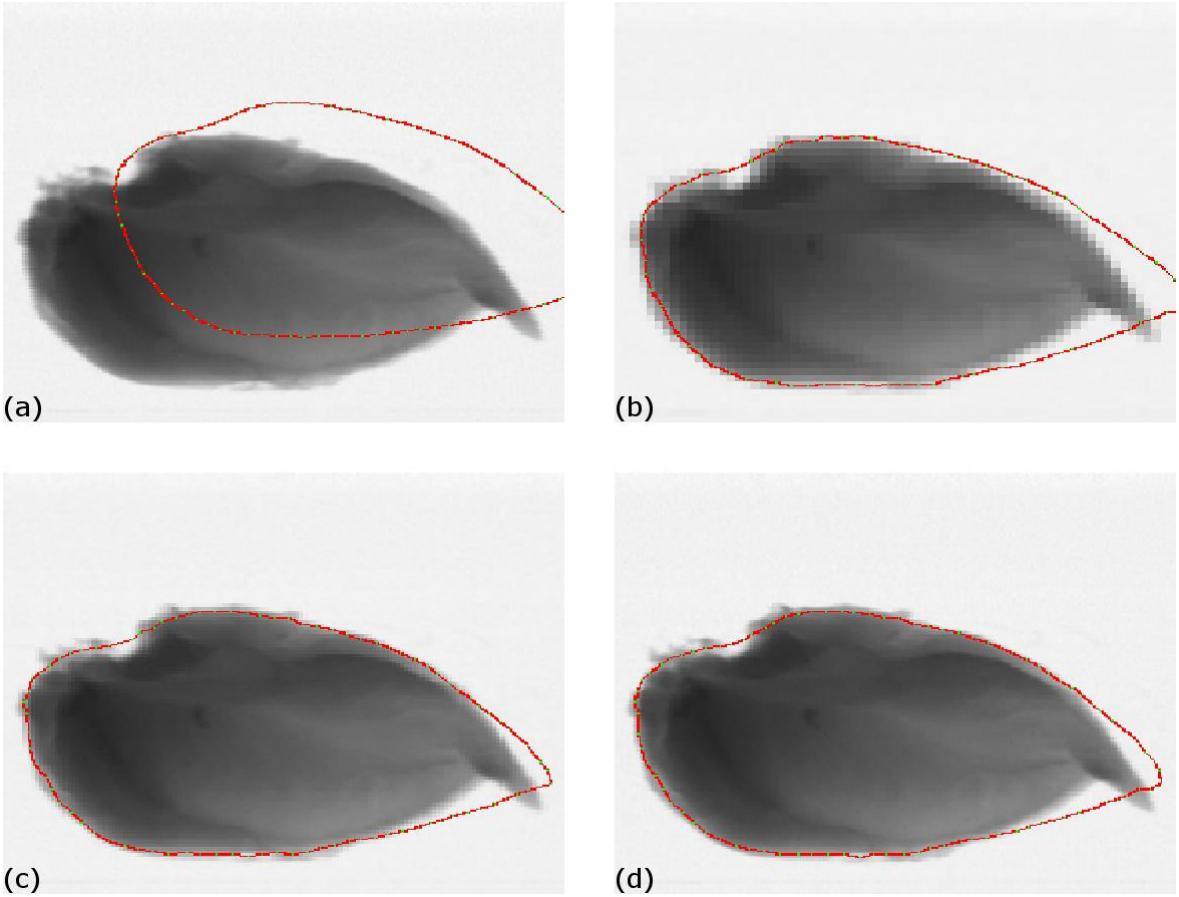


Figure 5.21 Multi-resolution image search results.(a) Initial position of model, (b) after six iterations in the multi-resolution Level 3, (c) after five iterations in the multi-resolution Level 2, (d) after one iteration in the multi-resolution Level 1

To further illustrate this point, we move the model fifty pixels further to the right as depicted in Figure 5.22(a). At this position, performing image search normally still ends in non-convergence. However, by implementing the multi-resolution technique, the image search algorithm is able to reach convergence. The result is shown in Figure 5.22(b).

With this in mind, we perform image search and apply the multi-resolution technique to all the test images. Figure 5.23 shows the number of iterations it took for the image search to converge across the test images. We immediately see that convergence was achieved in all the test images. None of the test images required more than one hundred iterations to

complete. This attests to the advantage of using multi-resolution technique in the image search.

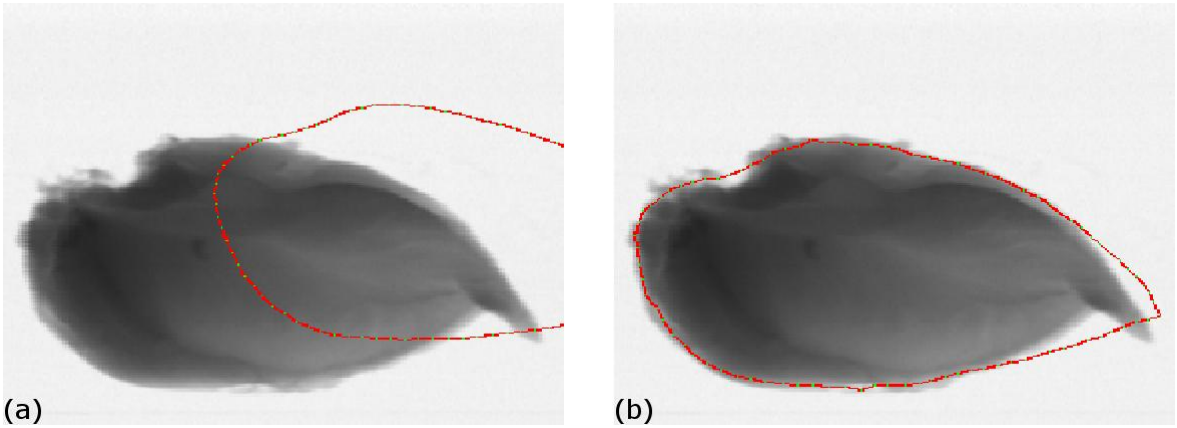


Figure 5.22 The effect of the multi-resolution technique in the image search algorithm on a poorly placed model. (a) The initial position of the model, (b) the final result of the image search after twenty-one iterations.

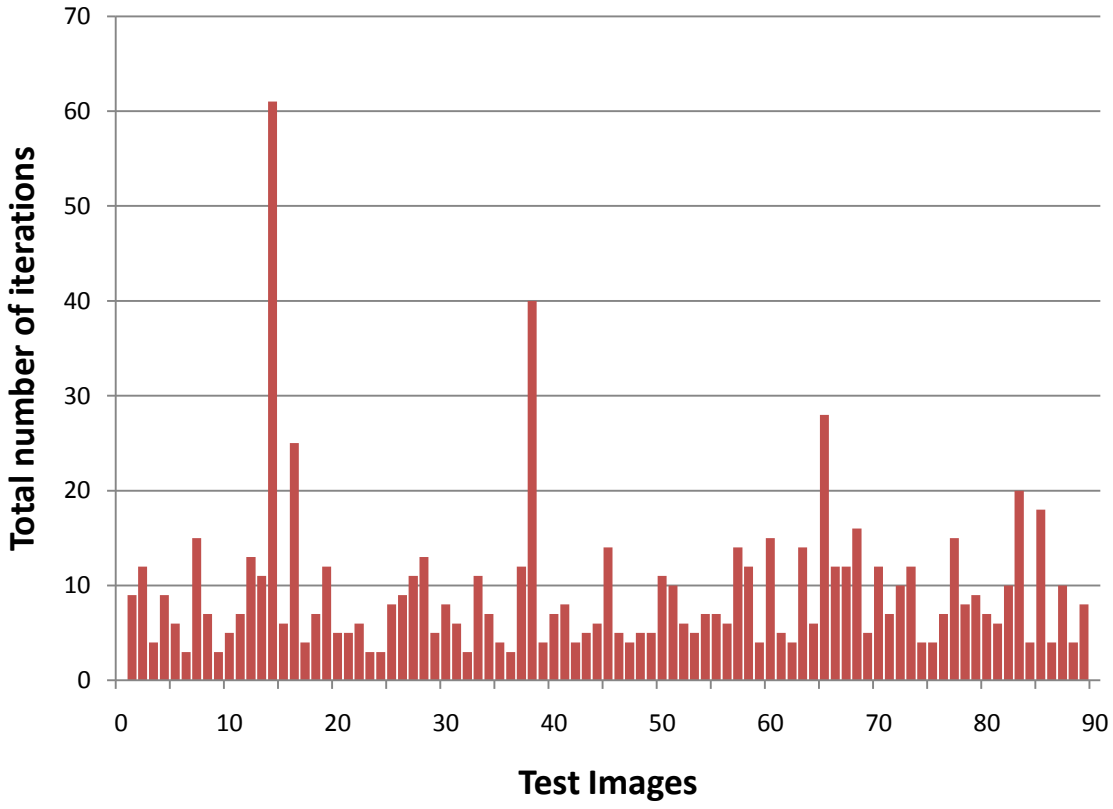


Figure 5.23 The plot of the iterations needed by the image search with the multi-resolution technique to converge for each test image.

5.5.4 Conclusion

In this chapter, we explored the conditions around the experiments for building the ASM model and image search. The computer hardware and specifications that the algorithm was run on were also described. We discussed the type of images that were used in the experiments.

Since the chicken fillets have a non-rigid nature, accurately landmarking these is not straightforward. There are only few salient features that can be used as mathematical landmarks and seen across several different images. To alleviate this insufficiency, we introduce pseudo-landmarks between the salient points. In this way we increase the number of landmarks thereby improving the performance of the ASM algorithm.

In the alignment stage, we saw the variation of the chicken fillet pieces across the test images. This contributes to the discrepancy of the locations for each landmark in the training set. Some landmarks have more movement while others were more stable. The quantity of landmarks dictated the number of dimensions that each shape resides in. However, the dimensions do not carry the equal amounts of information. In this regard, we aim to reduce the number of dimensions by using PCA and eliminate those that possess the least information.

For image search, the strongest edge and gray-level methods of selecting movement points were investigated. We found that for clean and noise-free images, the strongest edge method proved to be sufficient. However, in some images that contain noise and image artifacts, the gray-level method performed better. In both of these techniques, it is important to place the model correctly for image search. The initial position of the model can determine if the ASM search reaches convergence or not. To counteract this, we implemented the multi-resolution technique in the image search. With the multi-resolution operation, the model can be placed farther off the fillet but the ASM search still manages to converge.

6 ASM Applications in the Poultry Processing Industry

In many cases, imaging systems are combined with image processing techniques to perform specific tasks. Machine vision systems, as these are called, are commonly seen in industrial environments. The food industry has been one of the key beneficiaries of this technical development. According to Gunasekaran [**16**], it ranks among the top ten industries using machine vision technology.

Brosnan and Sun [**2**] acknowledge that previous studies on the applications of computer vision to the food industry focused on two areas: grading and product quality inspection. Automated inspection can either include separating questionable produce from a wholesome batch [**3**] [**11**] [**4**], or detecting contaminants in the products [**14**] [**17**]. Quality inspection was traditionally done by trained experts. However, this task is time-consuming and labour-intensive. In addition, inspectors are exposed to a multitude of health hazards [**36**]. These include phlebitis, muscokeletal disorders, shoulder, upper arm, upper back, neck and head problems. With automated visual inspection systems [**29**] gaining accuracy and speeds that are acceptable in industry standards, the technology is being used in the food industry [**34**].

6.1 Analyzing size and shape of detected object

In actual poultry processing plants, the throughput can range from tens to hundreds of pieces per minute. Poultry halves, breast quarters, leg quarters, legs, thighs, drumsticks, wings, and tenderloins are just some of the poultry cuts [**35**] identified by the USDA. Although the ideal condition is to have the same size, shape and weight of each piece to pass though the line, this is rarely the case due to the irregularity and non-rigid nature of poultry meat.

The weight of each piece is often the only data that is extracted on many food processing lines. While this is enough in many applications, the possibility of extracting more information out of the products can lead to more efficient production. Being able to get data about each piece that goes through the processing line is important for building up statistical information about the state of the production. The non-invasive nature of image acquisition and processing make it a strong candidate for this type of application. However, the methods that should be used must be flexible enough to adapt to the differences in each piece. For this reason, the ASM algorithm is well suited for it.

We again use the images of chicken breasts for training and testing. From the model we developed using the training set, we try and locate each piece in the test images. We assume that once the ASM search algorithm converges, the model has already found the chicken portion from the test image. We then proceed to take measurements of the outcome of the model, taking into account its length, width and area. Figure 6.1 illustrates how these measurements are taken. The area is found by counting all the pixels within the boundaries set by the landmark points. The length, represented as line L in Figure 6.1, is taken as the span of the line connecting the model landmarks having the greatest distance between each other. We then find the landmark points, x and y, on either side of L that has the farthest perpendicular distance to L. We identify these distances as W_1 and W_2. The

width is computed as the sum of W_1 and W_2. We note that because of the asymmetrical shape of the chicken breast, the lines connecting points x and y to L are not collinear. Figure 6.2 shows two of the test images that have been found by the ASM algorithm. The lines denoting the length and the width have been marked so that the measurements can be obtained. Aside from these three measurements, other information can also be extracted. The location of the centroid of the model is computed from the positions of the landmark points. We can also compute the angle at which the length line is oriented with respect to the horizontal axis.

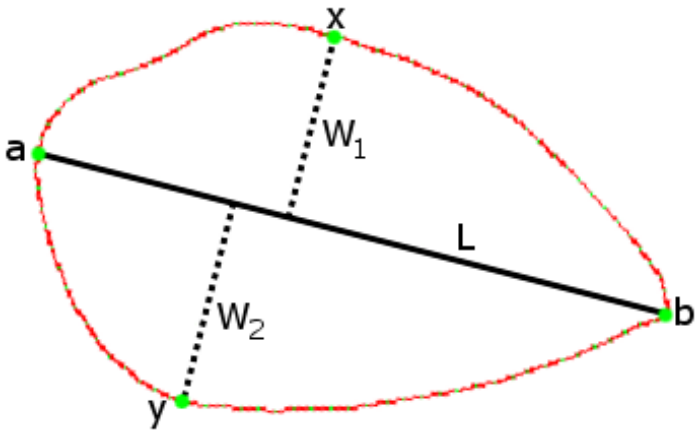


Figure 6.1 Measurements taken from model, where width is the sum of W_1 and W_2 and L is the length.

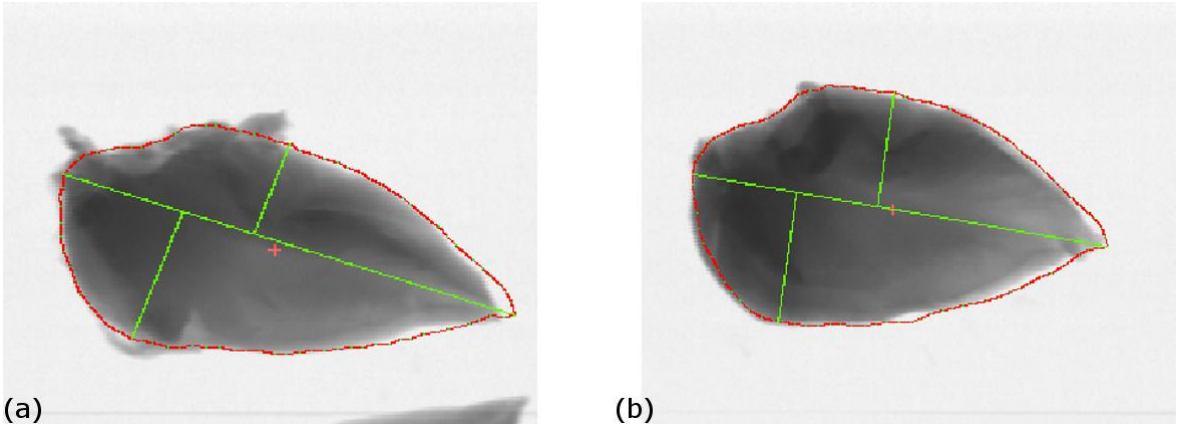


Figure 6.2 Chicken breasts (a) and (b) that were correctly located by the ASM search algorithm and labeled for measurement.

Performing the ASM search algorithm and obtaining the measurements from each test image, we get an overview of the entire batch. We are able to examine the histograms for the length (Figure 6.3), width (Figure 6.4) and area (Figure 6.5). From these initial measurements, we can derive statistical computations such as the mean and standard deviation.

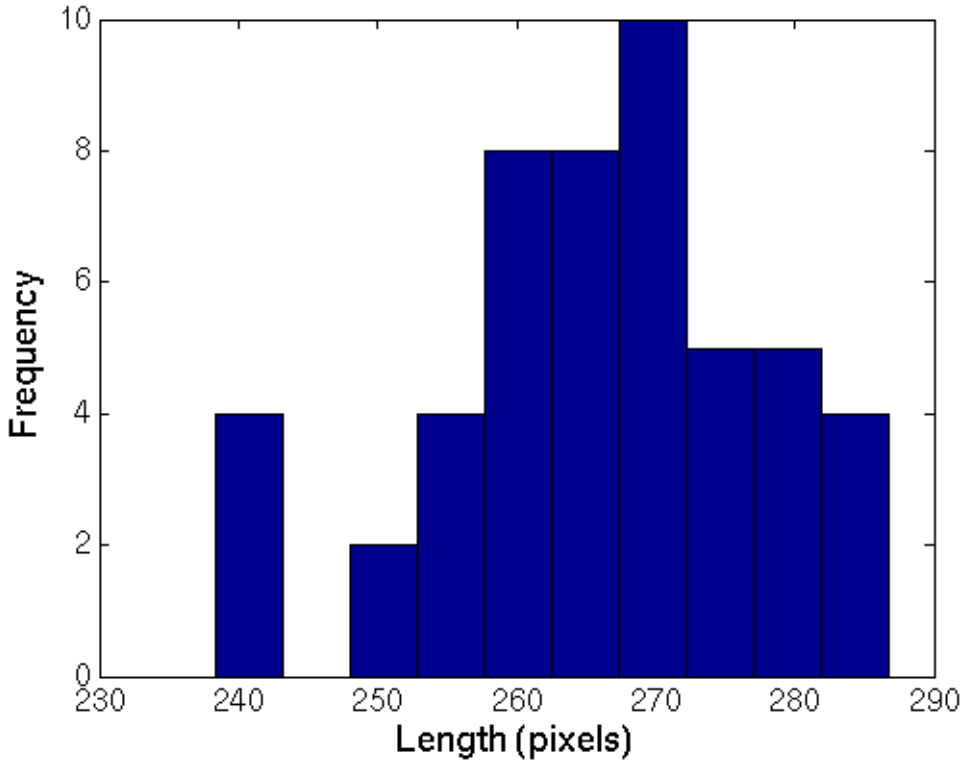


Figure 6.3 Histogram of the lengths of the chicken breasts in the test images.

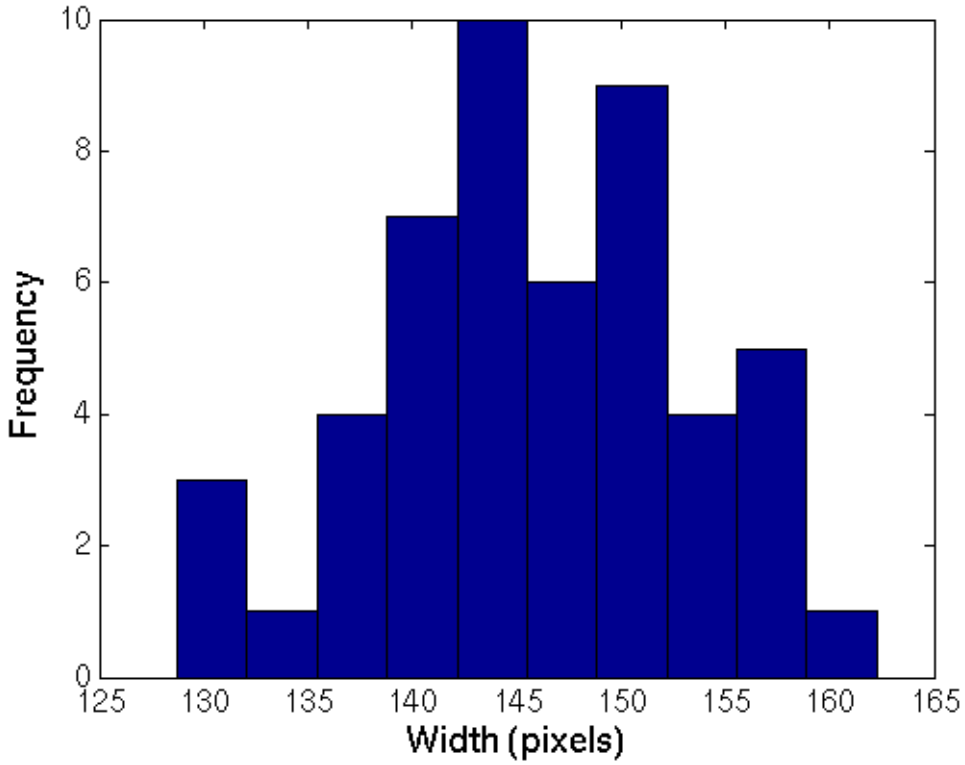


Figure 6.4 Histogram of the widths of the chicken breasts in the test images.

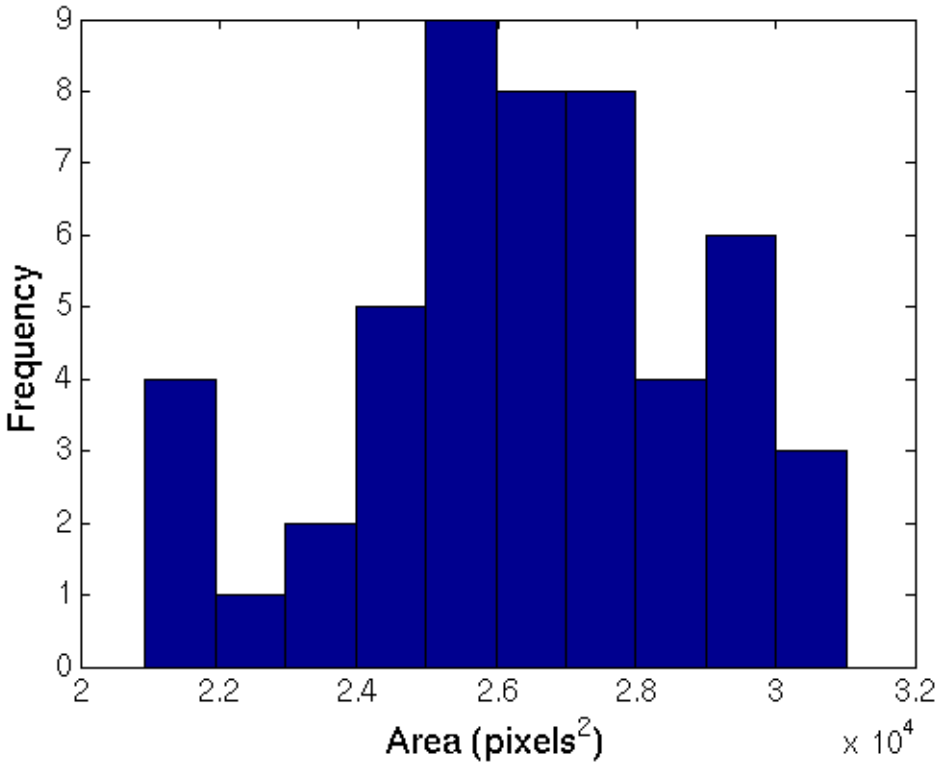


Figure 6.5 Histogram of the areas of the chicken breasts in the test images

Taking a statistical analysis of the length, width and area of the test images, we obtain the data in Table 6.1. We observe that the range of sizes in the test set is very wide. This presents an opportunity to categorize the chicken breasts with regards to the size and shape. It is a common practice to group products according to some defined metric such as weight, a process known as grading. The information about the size and shape of the chicken breasts will add another dimension to the grading process.

Table 6.1 Statistical computations for length, width and area.

	Length (pixels)	Width (pixels)	Area (pixels^2)	Centroid X	Centroid Y	Rotation (degrees)
Minimum	238.35	128.74	20937.50	132.00	107.00	-10.78
Mean	266.17	145.93	26443.07	164.88	145.12	11.70
Maximum	286.81	162.25	31011.50	191.00	172.00	24.96
Standard Deviation	11.80	7.51	2515.56	14.43	13.09	8.27

It is also possible to find the centroid of each chicken breast in the test image. Instead of using all the pixels in the boundary of the shape, we use only the landmarks. This makes the computation very simple. We can also give weights to each landmark to further improve the accuracy of the centroid. In this example, however, we treat all the landmarks equally. The angle of the line L in Figure 6.1 is used to determine the orientation of each chicken breast with respect to the horizontal. Table 6.1 also lists the statistics for the centroid and rotation for each of the test images. Both the location of the centroid and the

rotation of the chicken breasts can be used by packing machines to correctly pick up the products.

6.2 Locating bone fragments in chicken breasts

The SensorX , as discussed in Chapter 3, is primarily designed to detect contaminants such as bones on poultry products. It uses a set of algorithms to accurately identify the location of bones in the image. In factory applications, the SensorX is inserted into the processing line right after deboning or cutting to inspect for bone fragments. Figure 6.6 shows the basic configuration of the SensorX machine. Bone-free products pass through the system until it reaches the take-away conveyor for further processing. Products that have been found to have bones are redirected to workstations for bone removal. These are then inspected by the machine for the second time.

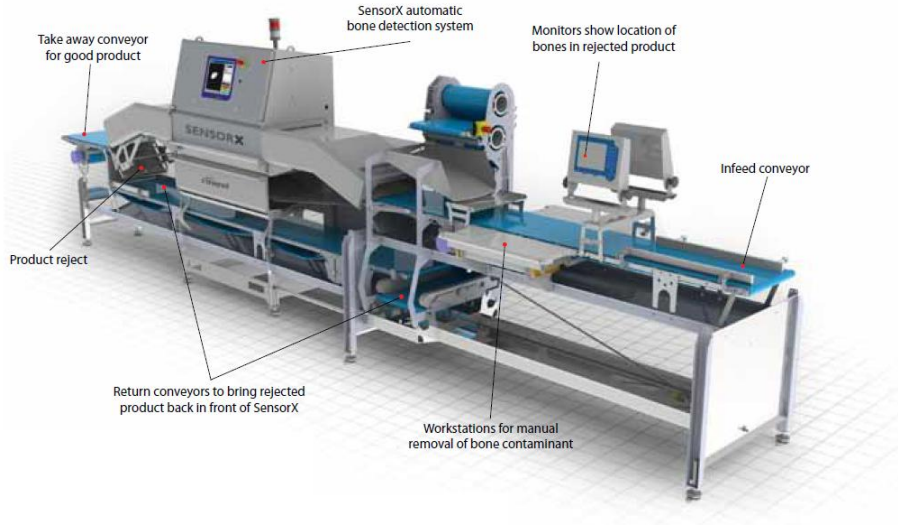


Figure 6.6 Basic SensorX in-line configuration. Courtesy of Marel Ehf.

The SensorX then records the number of products that passed bone-free and those that did not. This is used to evaluate the efficiency of the deboning or cutting process preceding the SensorX. It does not specify the amount of bone that was found in each piece or which part of the product the bone was discovered. These two details will help identify the cause of the bone incidence in the products which can be an incorrect setting of the deboning or cutting machines or staff making imprecise cuts in the products.

Obtaining the amount of detected bones and their location can be done separately using different techniques. In this example, we investigate a two-stage method to accomplish the objective. At the outset, we assume that the bone fragments have already been marked in the image. That is, they are highlighted to distinguish them from the rest of the image features. We again use chicken breasts in the training and test images. The training images are still the same as in the previous examples. For the test images however, we use a different set. All the test images have bone fragments marked with black pixels.

First, we have to find the shape, location, and orientation of each product in the image. This is done using the ASM algorithm. The outcome of the ASM search algorithm is used

for the next stage of the process. We want to map the bone fragments found in a test image such as in Figure 6.7 (a) to a reference shape Figure 6.7 (b). We perform this procedure for every bone pixel found in all the test images. After performing this operation for the entire batch of images, we obtain an image that contains all the bone incidences in the chicken breasts.

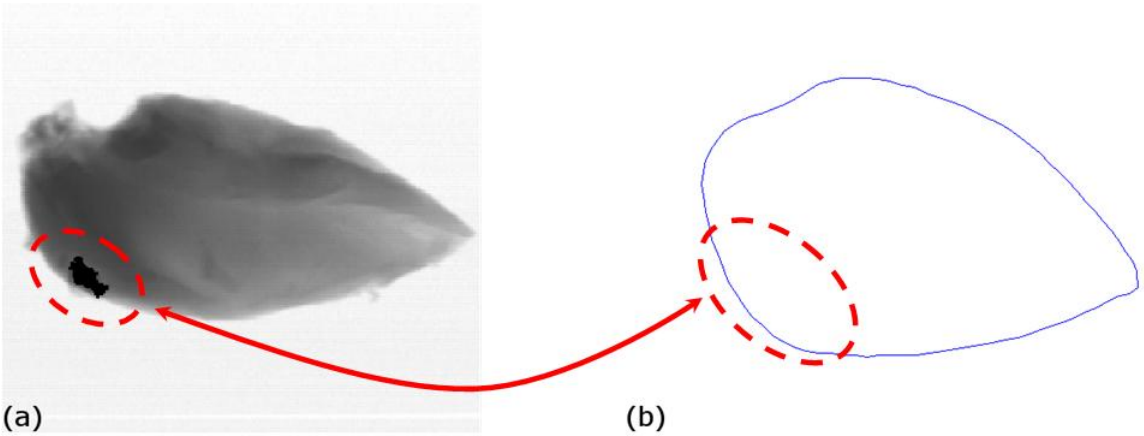


Figure 6.7 Mapping operation that locates the bone pixels in the test image(a), to a reference shape (b).

The mapping operation is more commonly known in image processing terms as warping. The Euclidean similarity transformation performed in the course of the ASM algorithm is also a form of image warp. In this application, we execute warping in a different manner. We adopt some of the steps in acquiring texture in Active Appearance Models (AAM) [**6**]. The AAM algorithm is the direct successor to ASM. Whereas ASM only deals with landmarks in the object and gathers information from the pixels surrounding those landmarks, AAM takes a step further and takes into account the texture found within the shape. Stegman [**33**] defines texture as

Definition 2: *Texture* is the pixel intensities across the object in question.

We are interested in the texture information as well but not quite in the same way as in the AAM algorithm. AAM combines the texture and the landmark information to find the target shape in the image. We only mean to look for the texture location of the bone pixels so that we may map it into a reference shape. The reference shape is the mean shape during the ASM training phase.

In order to have a consistent system of gathering texture information, AAM partitions the reference shape using Delaunay triangulation. With this method, the landmarks are connected by a mesh of triangles that each satisfies the Delaunay property. The circumcircle of every triangle must be empty of other points or landmarks. Illustrated in Figure 6.8, the circumcircle is the unique circle that passes through all three vertices of the triangle.

Because the shape of the chicken breasts is not entirely convex, some of the Delaunay triangles fall outside of the intended shape. Figure 6.9 (a) shows the model with some of the Delaunay triangles beyond the shape boundaries. These triangles do not contain any useful texture information as they are outside the shape and should therefore be removed.

For this reason, we need to constrain the Delaunay triangulation to include only triangles within the shape boundaries. In Figure 6.9 (b), only the Delaunay triangles inside the shape boundaries are included. In the course of the experiments, the Delaunay triangulation shown in Figure 6.9 (b) is implemented since we are only concerned about the bone fragments that appear within the boundaries of the chicken breast.

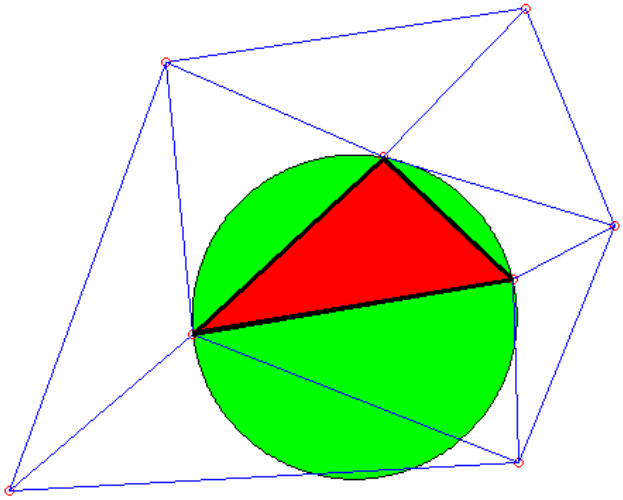
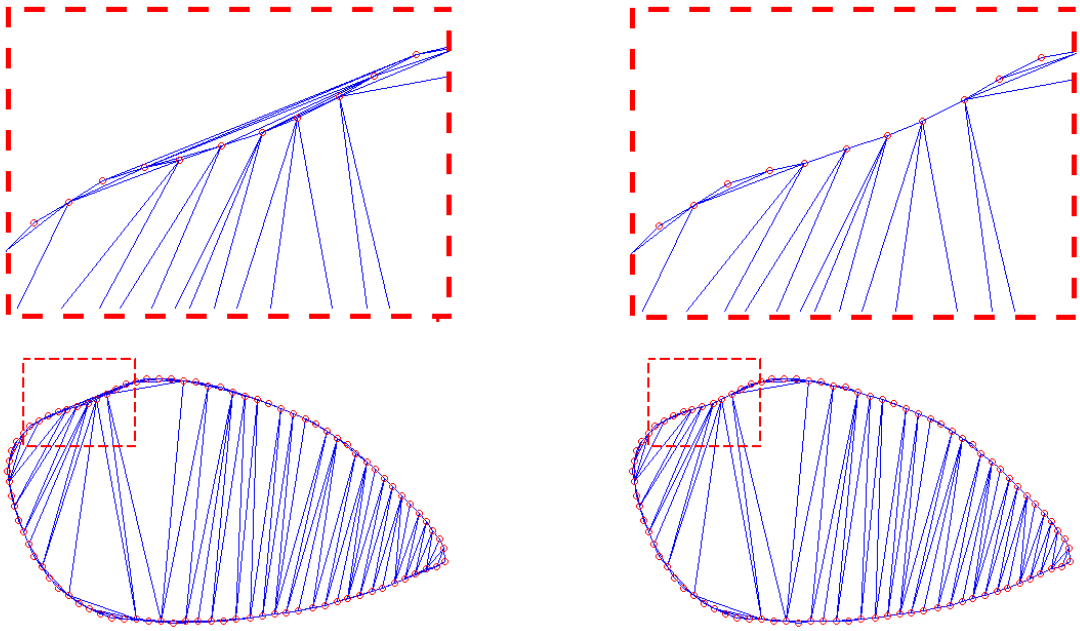


Figure 6.8 Set of points connected by triangles that comply with the Delaunay property.



(a) (b)

Figure 6.9 The model shape with unconstrained (a) and constrained (b) Delaunay triangles.

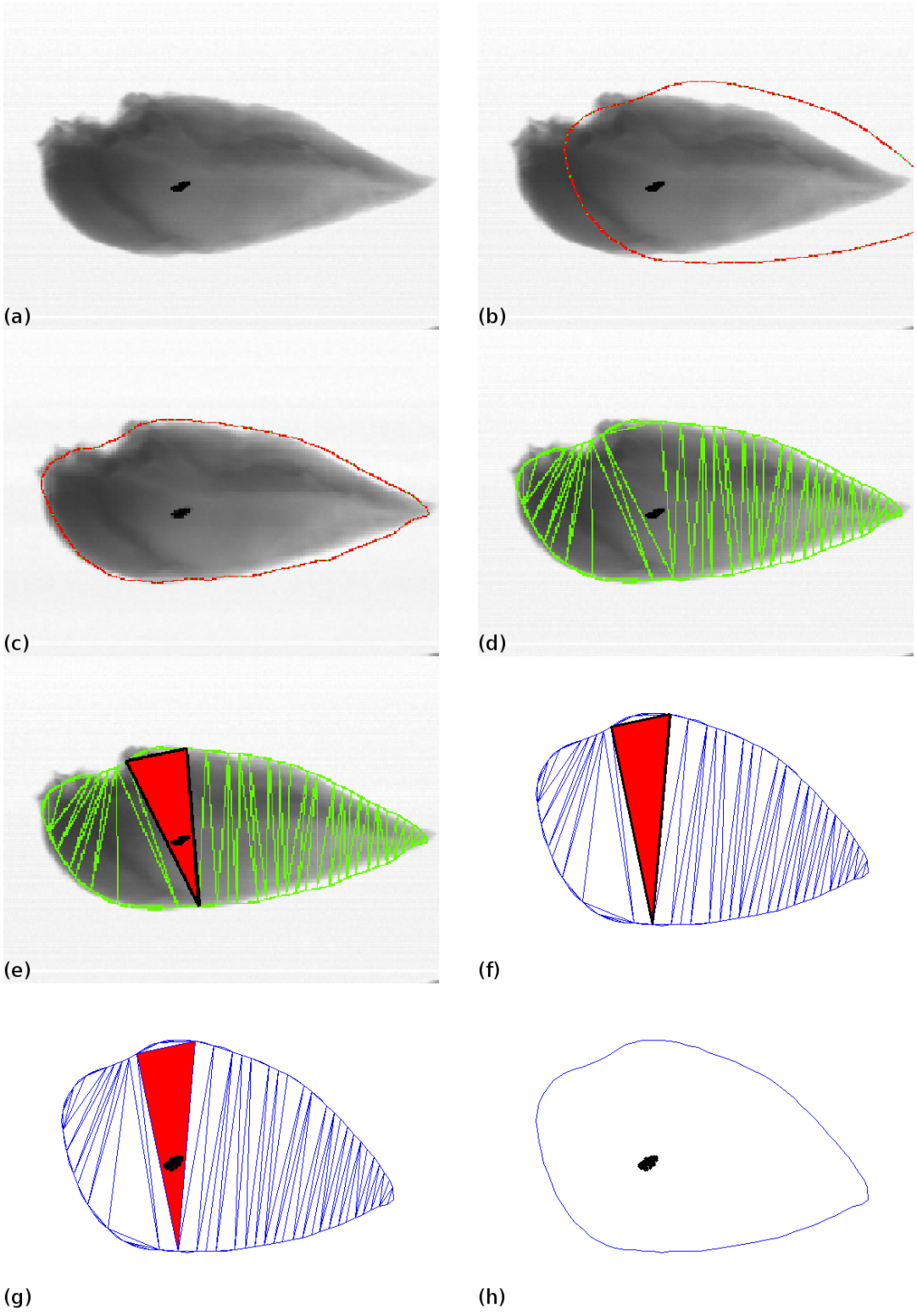


Figure 6.10 The process of the bone warping algorithm.(a)test image with bone fragments marked in black pixels, (b) placement of the ASM model to initialize search, (c) final result of ASM image search, (d) Delaunay triangulation on the result of the image search, (e) identifying the triangle/s that contains bone pixels, (f) location of the triangle in the reference model, (g) map of the bone pixels to the respective triangle in the reference model, (h) the bone pixels mapped to the reference model.

We do not warp each triangle from the target to the reference shape. Since we are only concerned with bone pixels, we identify which triangles each bone pixel belongs to. Those are the only triangles we warp into the reference shape. Each bone pixel is mapped to the corresponding triangle in the reference shape by an affine transformation. More information about the affine transformation is described in [**33**].

Figure 6.10 displays each stage in the bone-warping method. Figure 6.10 (a) is the test image with bone pixels. Figure 6.10 (b) and Figure 6.10 (c) show the initial and final positions of the ASM model before and after the search algorithm, respectively. The Delaunay triangulation done on the reference shape is replicated on the result of the ASM search as seen in Figure 6.10 (d). The triangles that contain the bone pixels are located and identified such as in Figure 6.10 (e). This triangle is the only one that will be warped. The bone pixels in this triangle will be warped to the corresponding triangle in the reference shape. The reference shape with the mapped bone pixels are shown in Figure 6.10 (h).

The bone warping process is done for each of the test images with bone. The location of the bone pixels found in each test image is recorded and shown in the reference image seen in Figure 6.11. We immediately notice that bone pixels are relatively absent in the middle portion of the reference shape. We also note that most of the bone pixels appear in the top portion of the reference shape and along the bottom side. The top part where the shape curves slightly inward is where the wings are cut from the breast. The bottom part is where the breast halves are joined together and held by the breast bone. Owing to the proximity of these to the locations where the meat is separated from bone, much of the bones pixels found are in these two areas.

Figure 6.11 Location of all the bone pixels seen by the bone-warp algorithm in all of the test images.

While it is indeed helpful to see where the bone pixels have appeared in the test images, the frequency at which the bone pixels occur at one location is also valuable information. Since the areas at which the bone pixels appear are not random, we expect the some bone pixels may overlap. Figure 6.12 reveals the rate of recurrence that each bone pixel position. A darker shade denotes more frequent occurrence. Here we see that the bone pixels appear at the same location only at a few instances. In a production standpoint, if this was a result of one continuous run, it indicates that a general recalibration of the cutting equipment may be in order.

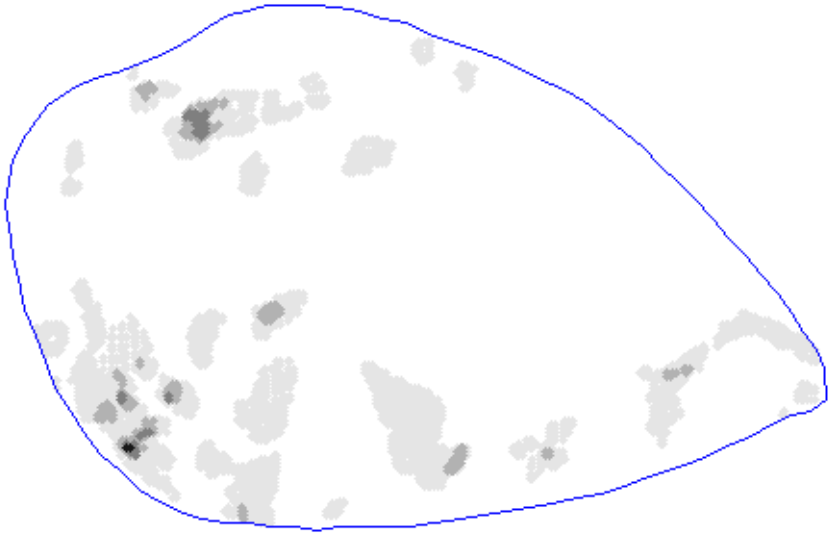


Figure 6.12 Distribution of bone pixels across all the test images showing the frequency of bones.

6.3 Conclusion

In this chapter we investigated two different applications for the ASM algorithms in poultry processing. The first application uses the final outcome of the ASM search algorithm to extract information about the test image. Because the model is able to capture the size, shape and location of the object in the image, measurements taken from the model is assumed to be the same as the object itself. Hence, measurements such as the length, width and size are measured and a statistical overview can be built with this information. The second application dealt with the bone fragments found in the chicken breasts. Using the outcome of the ASM search algorithm, the location of each bone pixel is mapped to a reference shape. In this manner, the occurrence of bone pixels and the frequency at which they occur can be tracked. With this data, appropriate measures can be done by the poultry line operator to eliminate or at least minimize the problem.

7 Conclusions and Future Work

7.1 Conclusions

The theory of the Active Shape Model algorithm has been examined in this thesis. Training is done to capture the statistical information about the shape that need to be found. A sufficient number of images are used to build up the Active shape model. If the number of training images is inadequate, there may not be enough variation described by the algorithm to correctly identify the shape during the search phase. However, there is also a possibility to over-train the algorithm by having too many training images.

Points, labeled as landmarks, are placed around the outline of the shape in each of the training images. Landmarks are placed in mathematically or anatomically important locations around the shape. When the distances between these points are too great, pseudo-landmarks are placed in between them to capture the shape more closely. Ideally these points are marked in the same place across the training images. This proves difficult especially for irregularly-shaped objects such as chicken breasts. However, careful manual landmarking can also do the job.

The positions of the landmarks in all the training images are recorded. Although the landmark sets contain the shape data, they also include some undesirable information. The rotation, translation, and scale are eliminated to extract the shape. For this purpose, shape alignment is performed. One of the shapes in the training set are used to serve as a reference shape. All the other landmark sets are aligned to this reference shape. With this method, only the shape and the variations that exist across the landmark sets remain after the alignment process.

Each of the aligned shapes is a point in a 2n dimensional space known as the allowable shape domain. Any point within this shape domain will have a resemblance in some extent to the training images. However due to the high dimensionality, mathematically finding a shape is a problem. PCA solves this by finding the most significant principal components. These are the eigenvectors of the covariance matrix. Most of the variation found in the training set is described with only a few of the principal components. In order to recreate any shape in the training set, only the linear combination of the mean shape and the principal components are needed.

The pixel information about each landmark point in the training images is also gathered during training. This will assist in selecting the best location to move each model point during the image search phase. Pixel samples extracted along paths normal to the shape boundary and passing through each landmark forms a statistical database of expected gray-level profiles.

During the image search, the model is placed in an arbitrary position in an unknown image. At each of the landmarks in the model, a gray-level profile normal to the shape boundary is extracted. The best movement positions for each model points are selected. The model is then rotated, scaled and translated to try and match the suggested landmark locations. This rarely results in a perfect match. The transformed model is then deformed according to the

established constraints to further conform to the suggested locations. When the disparity between the recommended positions and the deformed model is still above a set limit, the process is repeated.

To improve the performance of the algorithm, a multi-resolution approach is applied. In this case, an ASM model is trained for every resolution level. During image search, the test image is decomposed into the appropriate number of resolutions. The search is done starting from the coarsest resolution going up to the finest level. The outcome at each level is used as the starting point for the next. This allows the model to move at large distances at the lower resolution levels and make fine adjustments at the higher resolutions.

The flexibility and accuracy of the ASM algorithm lends itself well to applications where the target object is non-rigid. In this thesis, the algorithm is applied to poultry products. Since the ASM algorithm tries to find the target in the image and approximate the object boundaries, it is suited for gathering size information. The data collected from a number of images builds up a statistical summary of the test images. In a food processing line, this information is useful for measuring the quality of the batch.

Another application for ASM in the food processing industry is identifying the locations at which bone fragments occur in poultry pieces. The ASM model serves as the reference shape to which bone fragments found in the poultry cuts is mapped to. This helps identify problem areas in the processing line and eventually lead to solutions for eliminating the bone fragments.

7.2 Future work

The landmarking method employed for building up the ASM model consisted of manually placing points along the object boundary on the training images. The irregularity with the shape boundaries of the poultry pieces only make this task even more challenging. An automated method of landmarking will definitely help in this regard. Another area that can be improved upon is on the search phase of the algorithm. The algorithm tries to grab on to edges that mostly resemble the object it was trained on. On the event that the model starts on a location where it does not reach any object edge, a mechanism must be put in place such that the movement is towards a location that has a high probability of having the object. Two possibilities for this is to take into account *a priori knowledge* or using histograms.

Shape recognition of non-rigid objects such as food items is very challenging and difficult to accomplish with conventional image processing techniques. The capability of the ASM algorithm to accept a wide degree of variation in recognizing shapes makes it an ideal candidate for applications in the food processing industry. For more complicated images, AAMs can be used instead. This presents the advantage of utilizing texture information in addition to landmark data.

References

[1] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger, "Application of affine-invariant fourier descriptors to recognition of 3-d objects," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 640-647, 1990.

[2] T. Brosnan and D. W. Sun, "Inspection and grading of agricultural and food products by computer vision systems- a review," *Computers and Electronics in Agriculture*, vol. 36, pp. 193-213, 2002.

[3] K. Chao, Y. R. Chen, W. R. Hruschka, and F. B. Gwozdz, "On-line inspection of poultry carcasses by dual-camera system," *Journal of Food Engineering*, vol. 51, no. 3, pp. 185-192, February 2002.

[4] Y. R. Chen. (2006, August) SPI Web site. [Online]. http://spie.org/x8641.xml?ArticleID=x8641

[5] T. F. Cootes. (2011, June) ASM/AAM Lectures. [Online]. http://personalpages.manchester.ac.uk/staff/timothy.f.cootes/Lectures/asm_lecture.ppt

[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active Appearance Models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 681-685, June 2001.

[7] T. F. Cootes, A. Hill, C. J. Taylor, and J. Haslam, "The Use of Active Shape Models for Locating Structures in Medical Images," *Image and Vision Computing*, vol. 12, no. 6, pp. 355-366, July 1994.

[8] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active Shape Models - Their Training and Application," *Computer Vision and Image Understanding*, vol. 6, no. 1, pp. 38-59, January 1995.

[9] T.F. Cootes, C.J. Taylor, and A. Lanitis, "Active Shape Models: Evaluation of a Multi-Resolution Method for Improving Image Search," in *Proceedings British Machine Vision Conference*, York, 1994, pp. 327-336.

[10] I. L. Dryden and K. V. Mardia, *Statistical Shape Analysis*.: John Wiley & Sons, 1998.

[11] Z. Du, M. K. Jeong, and S. G Kong, "Band selection of hyperspectral images for automatic detection of poultry skin tumors," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 4, July 2007.

[12] H. Eiriksson, *Shape Representation, Alignment and Decomposition*. Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, Denmark: Informatics and Mathematical

Modelling, Technical University of Denmark, DTU, 2001.

[13] M. A. Fischler and R. A. Elschlager, "The Representation and Matching of Pictorial Structures," *IEEE Transactions on Computers*, vol. C-22, no. 1, pp. 67-92, January 1973.

[14] G. Ginesu, G. G. Giusto, V. Magner, and P. Meinlschmidt, "Detection of foreign bodies in food by thermal processing," *IEEE Transactions on Industrial Electronics*, vol. 51, no. 2, pp. 480-490, April 2004.

[15] J. C. Gower, "Generalized Procrustes Analysis," *Psychometrika*, vol. 40, no. 1, pp. 33-51, 1975.

[16] S. Gunasekaran, "Computer vision technology for food quality assurance," *Trends in Food Science and Technology*, vol. 7, August 1996.

[17] R. P. Haff and N. Yoyofuku, "X-ray detection of defects and contaminants in the food industry," *Sensing and instrumentation for Food Quality and Safety*, vol. 2, no. 4, pp. 262-273, December 2008.

[18] A. Hill, T. F. Cootes, and C. J. Taylor, "A Generic System for Image Interpretation Using Flexible Templates," in *British Machine Vision Conference*, 1992, pp. 276-285.

[19] A. Hill, C. J. Taylor, and A. D. Brett, "A framework for Automatic Landmark Identification Using a New Method of Nonrigid Correspondence," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 241-251, March 2000.

[20] H. Hotelling, "Analysis of a Complex of Statistical Variables into Principal Components," *Journal of Educational Psychology*, vol. 24, no. 6, pp. 417-441, September 1933.

[21] T.J. Hutton, S. Cunningham, and P. Hammond, "An Evaluation of Active Shape Models for the Automatic Identification of Cephalometric Landmarks," *European Journal of Orthodontics*, vol. 22, no. 5, pp. 499-508, 2000.

[22] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321-331, 1988.

[23] S. Klim, S. Mortensen, B. Bodvarsson, L. Hyldstrup, and H.H. Thorberg, "More Active Shape Model," in *Image and Vision Computing New Zealand 2003*, 2003.

[24] B. Landau, L.B. Smith, and S.S. Jones, "The importance of shape in early lexical learning," *Cognitive Development*, vol. 3, pp. 299-321, 1988.

[25] P. C. Mahalanobis, "On the Generalised Distance in Statistics," in *Proceedings of the National Institute of Sciences India*, 1936, pp. 49-55.

[26] T. McInerney and D. Terzopolous, "Deformable models in medical image analysis: A survey," *Medical Image Analysis*, vol. 1, no. 2, pp. 91-108, June 1996.

[27] M. Peura and J. Iivarinen, *Efficiency of Simple Shape Descriptors*.: World Scientific, 1997.

[28] S. Reuda, J. Udupa, and L. Bai, "A New Method of Automatic Landmark Tagging for Shape Model Construction via Local Curvature Scale," *Medical Imaging 2008; Visualization, Image-guided Procedures, and Modeling*, vol. 6918, pp. 69180N-69180N-12, April 2208.

[29] R. Rios-Cabrera, I. Lopez-Juarez, and H. Sheng-Jen, "An ANN analysis in a vision approach for potato inspection," *Journal of Applied Research and Technology*, vol. 6, no. 2, pp. 106-119, August 2008.

[30] S. Rueda, J. Udupa, and L. Bai, "Local Curvature Scale: A New Concept in Shape Description," *Medical Imaging 2008: Image Processing*, vol. 6914, pp. 69144Q-69144Q-11, April 2008.

[31] P. P. Smyth, C. J. Taylor, and J. E. Adams, "Automatic Measurement of Vertebral Shape Using Active Shape Models," *Image and Vision Computing*, vol. 15, no. 8, pp. 575-581, August 1997.

[32] H.T. Sogaard, "Weed Classification by Active Shape Models," *Biosystems Engineering*, vol. 91, no. 3, pp. 271-281, July 2005.

[33] M.B. Stegmann, *Active Appearance Models: Theory, Extensions and Cases*, 2nd ed. Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby: Informatics and Mathematical Modelling, Technical University of Denmark, DTU, 2000.

[34] D. W. Sun, "Inspecting pizza topping percentage and distribution by a computer vision method," *Journal of Food Engineering*, vol. 44, no. 4, pp. 245-249, June 2000.

[35] United States Department of Agriculture. (1998, April) Agricultural Marketing Service. [Online]. http://www.ams.usda.gov/AMSv1.0/getfile?dDocName=STELDEV3002393

[36] United States Department of Labor. (2011, October) Occupational Safety and Health Administration. [Online]. http://www.osha.gov/SLTC/ergonomics/powerpoint/chicken/index.html

[37] B. van Ginneken, A.F. Francgi, J.J. Staal, B.M. ter Haar Romeny, and M.A. Viergever, "A Non-Linear Gray-Level Appearance Model Improves Active Shape Model Segmentation," in *Mathematical Methods in Biomedical Image Analysis, 2001, MMBIA 2001 Workshop on*, 2001, pp. 205-212.

[38] K.N. Walker, T.F. Cootes, and C.J. Taylor, "Determining Correspondences for

Statistical Models of Facial Appearance," in *FG*, 2000, pp. 271-276.

[39] B. Widrow, "The 'Rubber-Mask ' Technique - I. Pattern Measurement and Analysis," *Patter Recognition*, vol. 5, no. 3, pp. 175-176, September 1973.

[40] B. Widrow, "The 'Rubber-Mask' Technique - II. Pattern Storage and recognition," *Pattern Recognition*, vol. 5, no. 3, pp. 199-211, Sept. 1973.

[41] J. Xie, P. Heng, and M. Shah, "Shape matching and modeling using skeletal context," *Pattern Recognition*, vol. 41, no. 5, pp. 1756-1767, May 2008.

[42] A.L. Yuille, P.W. Hallinan, and D.S. Cohen, "Feature Extraction from Faces Using Deformable Templates," *International Journal of Computer Vision*, vol. 8, no. 2, pp. 99-111, 1992.

[43] C.T. Zahn and Roskies Z., "Fourier Descriptors for Plane Closed Curves," *IEEE Transactions on Computers*, vol. C-21, no. 3, pp. 269-281, March 1972.

[44] Z. Zheng, J. Jiong, D. Chunjiang, X. Liu, and J. Yang, "Facial Feature Localization based on an improved active shape model," in *Information Sciences*, vol. 178, 2008, pp. 2215-223.

Appendix A - Selecting profile pixels at each landmark [5]

During image search, a method must be put in place to facilitate the selection of the best movement location for each landmark. In choosing a one-dimensional (1-D) profile normal to the shape boundary, we restrict the movement to only one dimension which simplifies the calculation for locating the best point. Even with the restriction, this method still allows enough flexibility to be able to move and deform the model to fit the target shape. Figure A.1 shows how the profile is drawn for one landmark point.

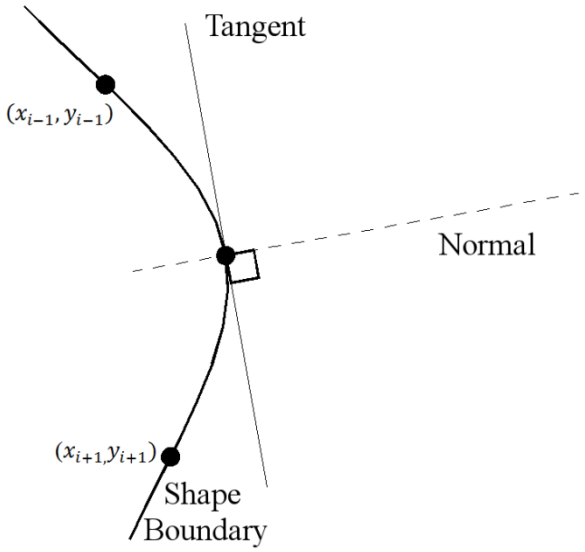


Figure A.1 A normal to the model boundary

In order to obtain the normal, we have to compute for the tangent line to the shape boundary at the current landmark. First we compute

$$d_x = x_{i+1} - x_{i-1} \text{ and } d_y = y_{i+1} - y_{i-1} \,. \tag{A.1}$$

We use this to compute for the tangent line

$$\left(t_x, t_y\right) = \frac{(d_x, d_y)}{\sqrt{\left(d_x^2, d_y^2\right)}} \,. \tag{A.2}$$

From the tangent line we derive the normal

$$\left(n_x, n_y\right) = \left(-t_y, t_x\right) \,. \tag{A.3}$$

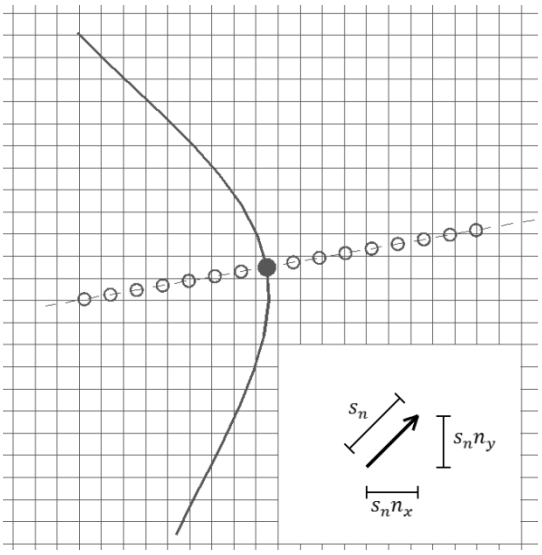


Figure A.2 Selecting points along the normal

If we let *(X, Y)* be the current landmark position we can choose points along the normal to belong to the profile using the equation

$$(X, Y) + i(s_n n_x, s_n n_y) \tag{A.4}$$

where $i = (\ldots,-2,-1,0,1,2,\ldots)$ and s_n is the steps along the normal that we want to consider. Being able to select the steps, s_n, is especially useful when dealing with different resolution levels. Shown in Figure A.2, one pixel in a resolution level occupies two pixel locations in the next higher resolution level. If the pixels in the profiles were selected to be always the same distance with each other at all resolution levels, it would come to a point wherein all the pixels selected for the profiles would be of the same value. This will not benefit the best point selection in the image search process since there would be no difference in the Mahalanobis distances between pixels in the profile. Also during gray-level modeling, having profiles with very similar pixel levels would result to nearly singular covariance matrices. This will not produce accurate results during image search.

A. Normalizing the gray-level profile

Once the profile is profile is obtained, the best movement point of the landmark can be found using the strongest edge method. We do this by looking for the highest peak produced by computing the derivative of the profile. This method works well when the landmarks are placed on the boundary, the background is clean and the target shape has a large difference in contrast to the background. But it is rare that these conditions are met and most often the strongest edge method fails to produce satisfactory results. The next solution is to make a statistical analysis of the gray-level profiles of each landmark. This is done on every training image during the training phase. Before we can start on this, however, we have to perform a conditioning step for the profiles. Since the training images are taken at different instances in a range of settings, some variations can be observed. Some images appear darker or brighter than others while most, if not all, images will contain some noise. In this section, we adapt the method used in [7] to eliminate or minimize the effects described above. We start with computing the derivative profile, *g*, of the j^{th} landmark point at image *i*

$$g_{ij} = p_k - p_{k-1}, \tag{A.5}$$

where *p* is the vector of pixel values taken at the profile. We then normalize this profile by

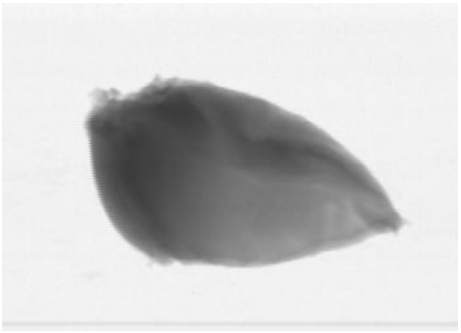
$$g_{ij}{}' = \frac{g_{ij}}{\sum_{k=1}^{n_p} \left| g_{ijk}{}' \right|}. \tag{A.6}$$

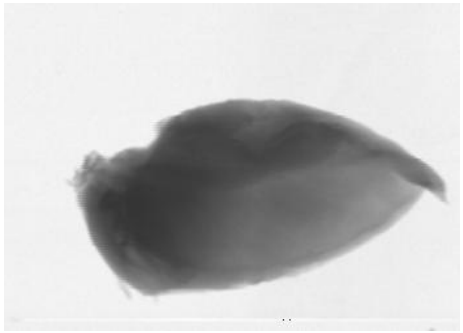
We then get the mean of the normalized derivative profiles for each landmark point across the images,

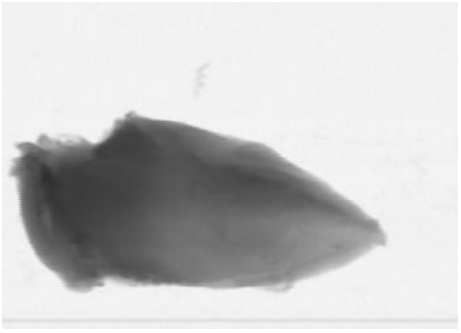
$$\overline{g_i} = \frac{1}{N_S} \sum_{j=1}^{N_s} g_{ij}{}' . \tag{A.7}$$

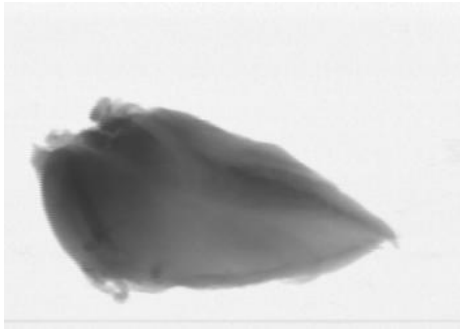
During training, we gather the mean normalized derivative profiles for each landmark point and compute the covariance matrix, S_{gi}. This would form a statistical database of expected gray-level profiles for each landmark. During image search, we also perform the methods described above after extracting the profiles.

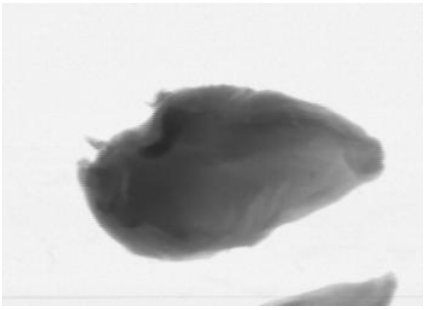
Appendix B –Training Images

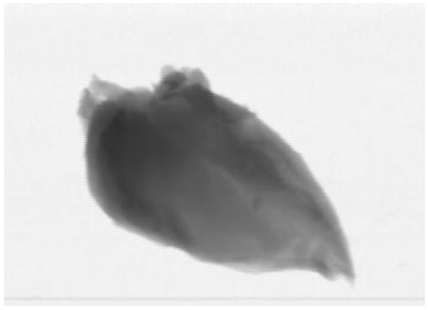
1 

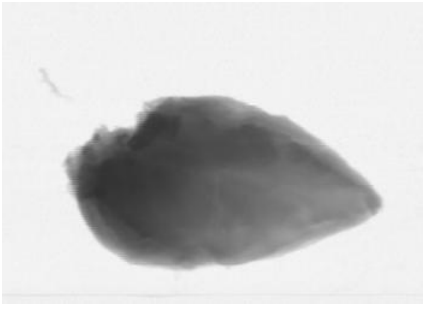
2 

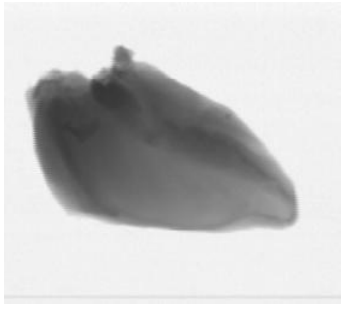
3 

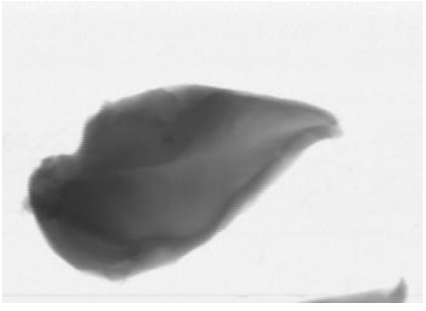
4 

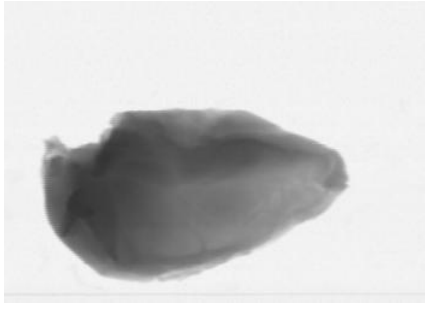
5 

6 

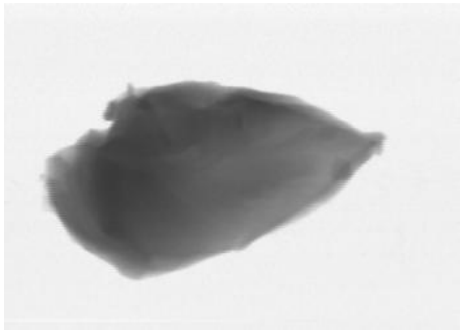
7 

8 

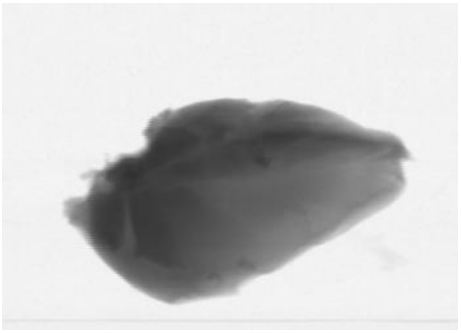
9 

10 

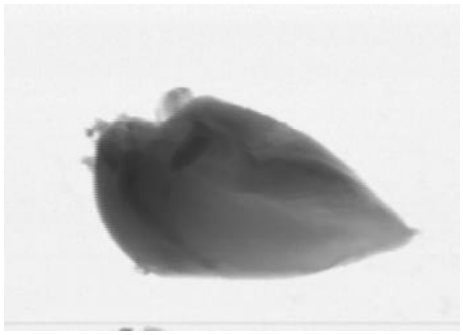
11



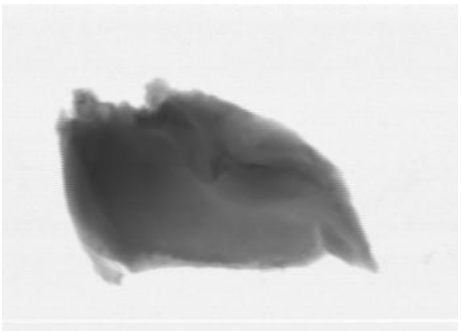
12



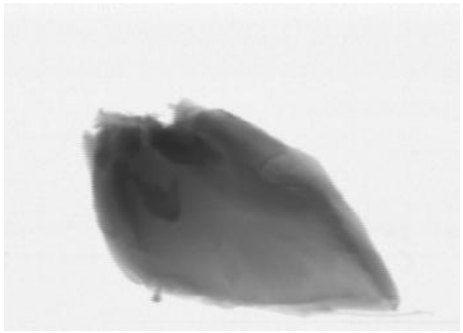
13



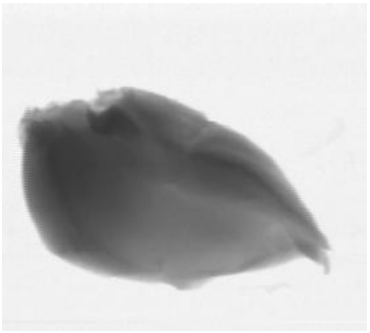
14



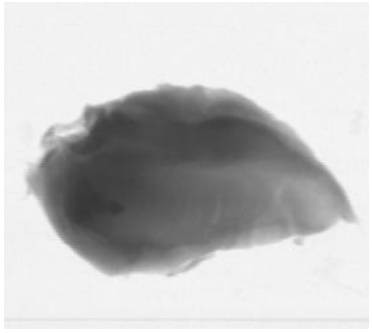
15



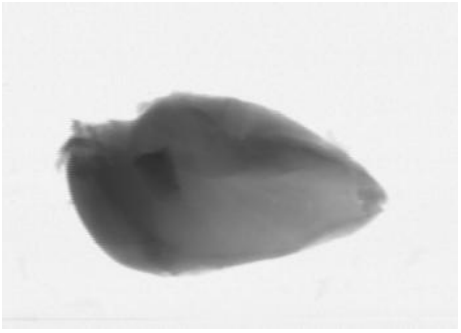
16



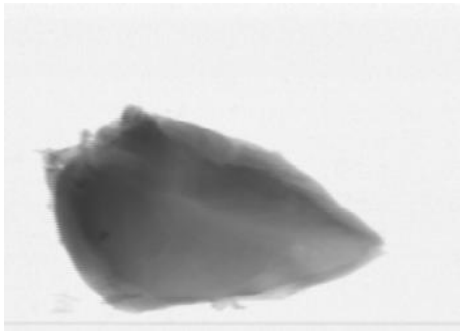
17



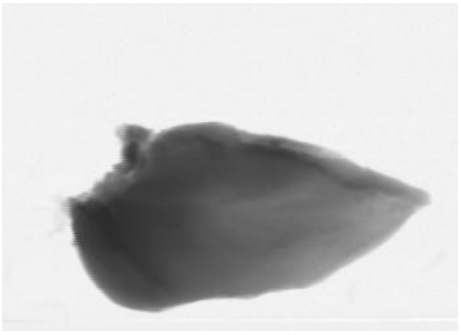
18

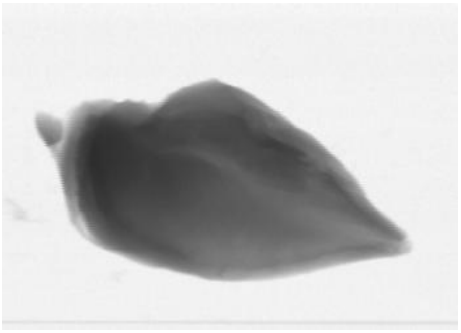


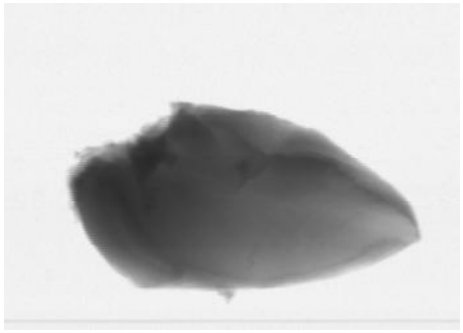
19

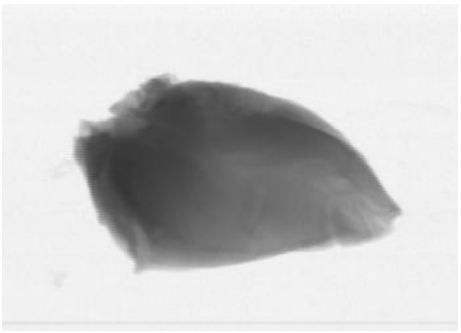


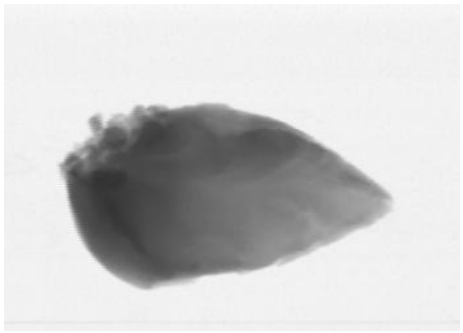
20

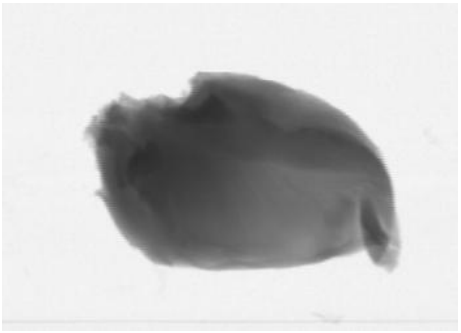


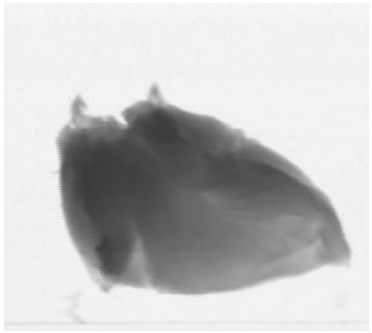
21 

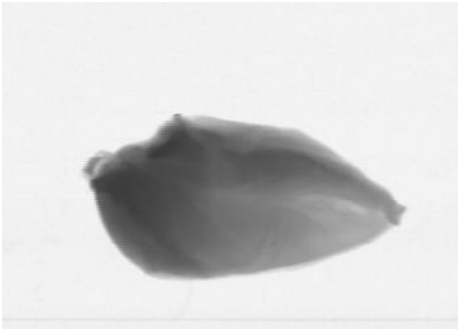
22 

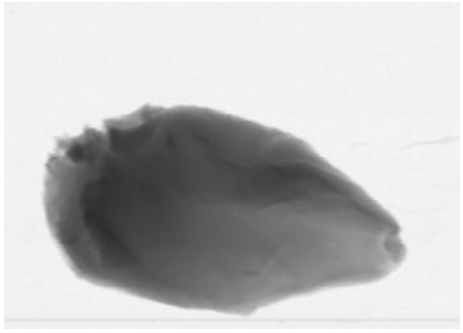
23 

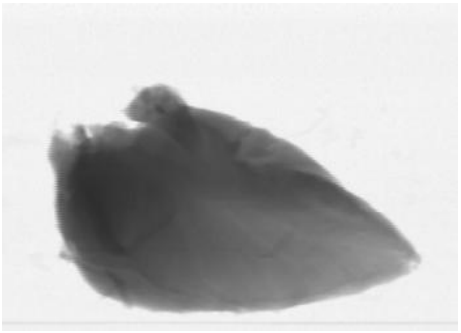
24 

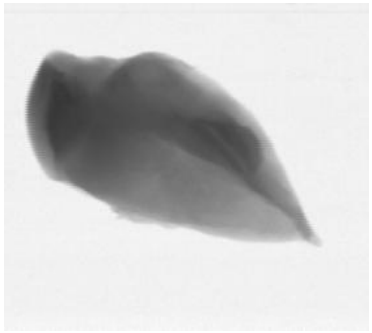
25 

26 

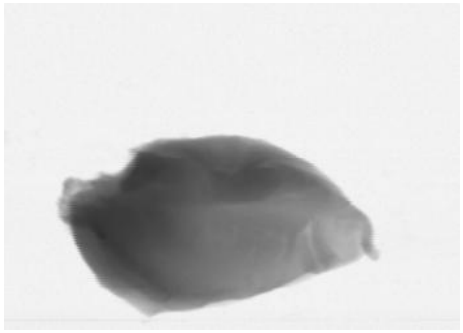
27 

28 

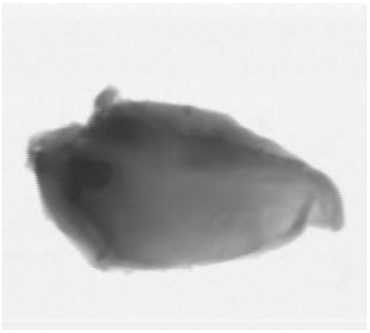
29 

30 

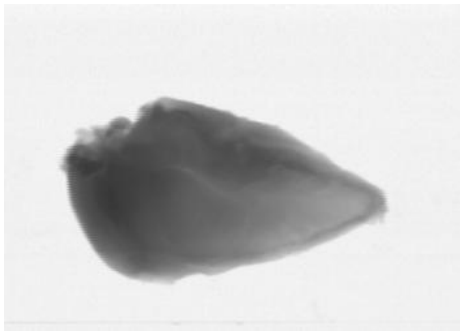
31



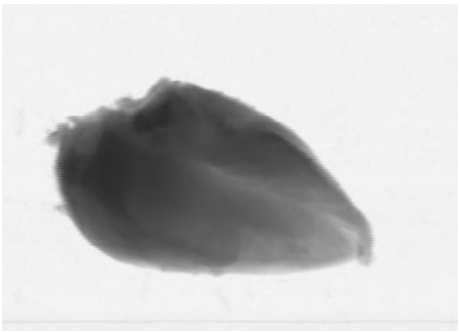
32



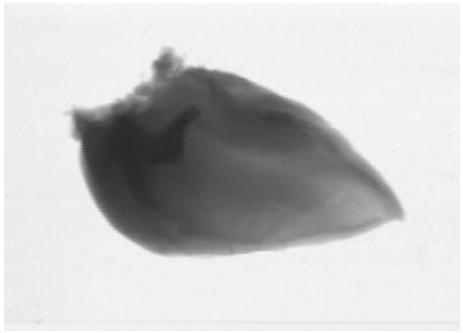
33



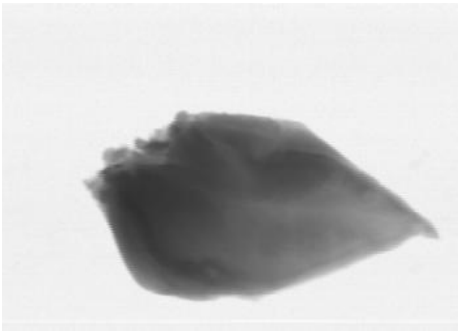
34



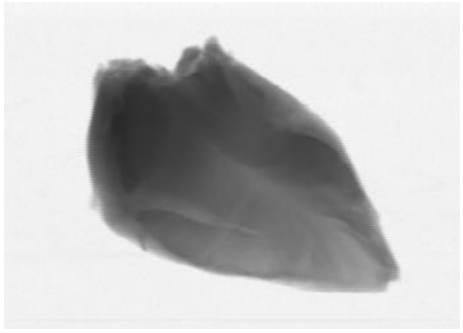
35



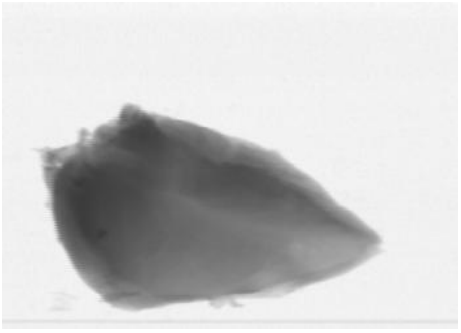
36



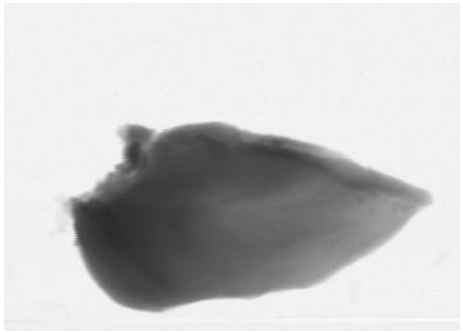
37



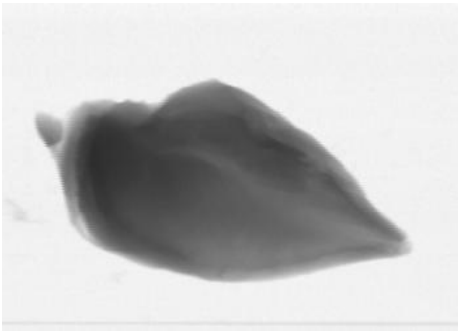
38

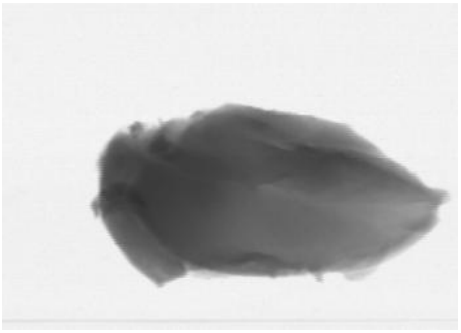


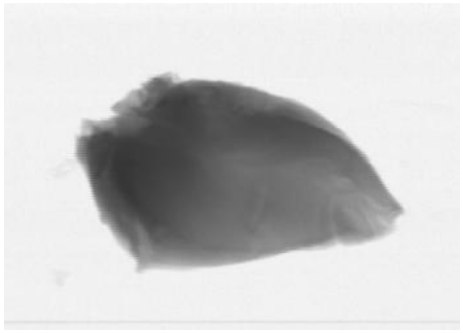
39

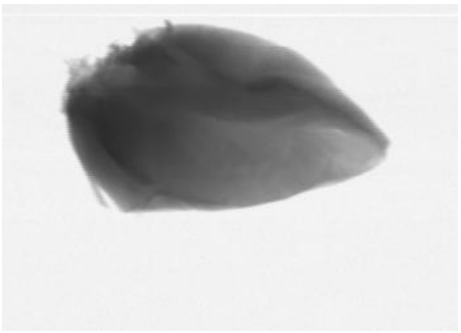


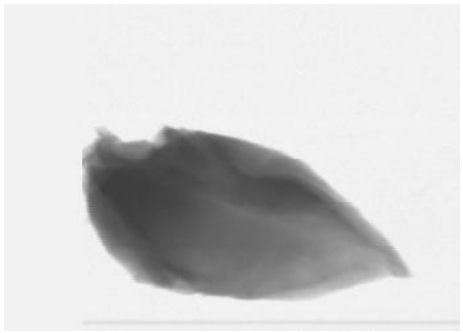
40

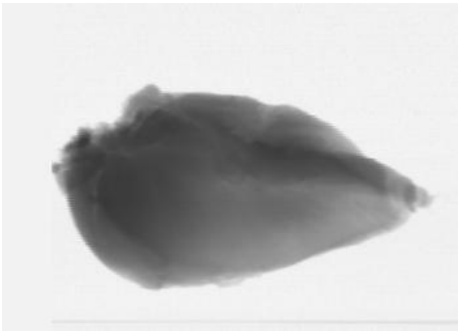


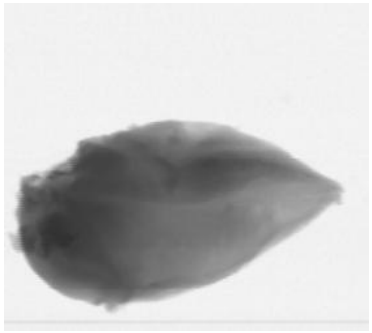
41 

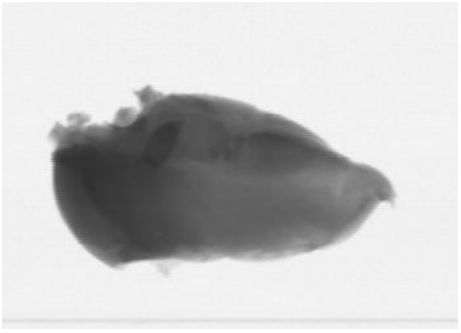
42 

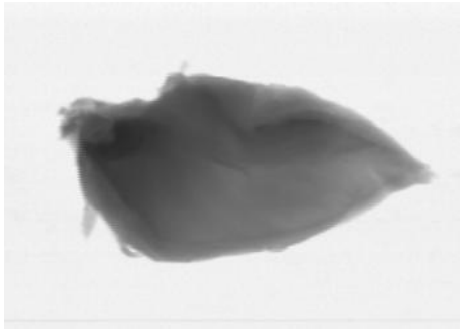
43 

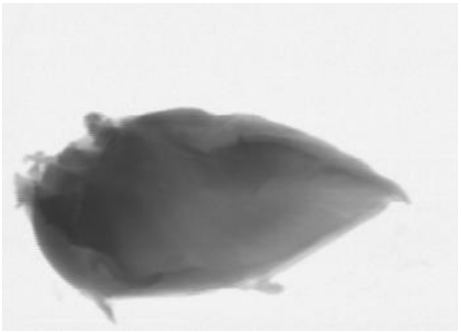
44 

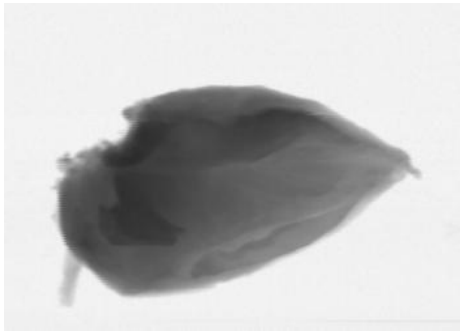
45 

46 

47 

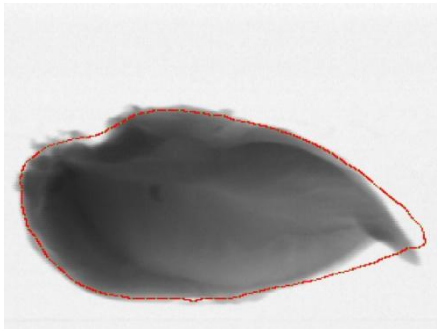
48 

49 

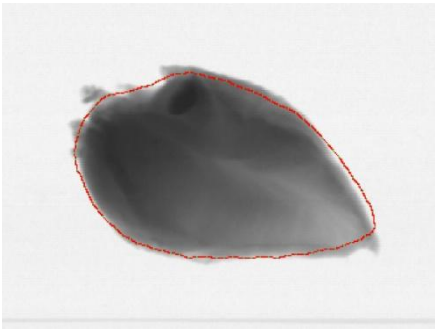
50 

Appendix C –ASM Results

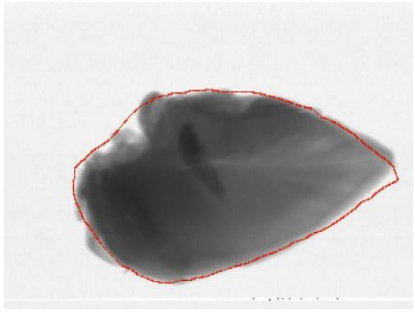
1



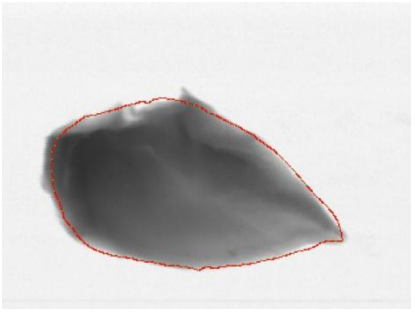
2



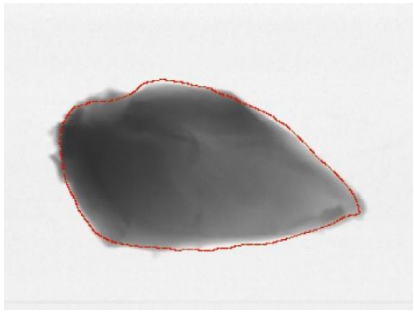
3



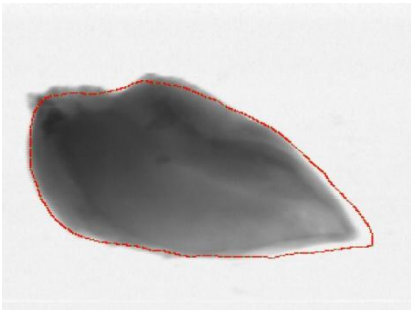
4



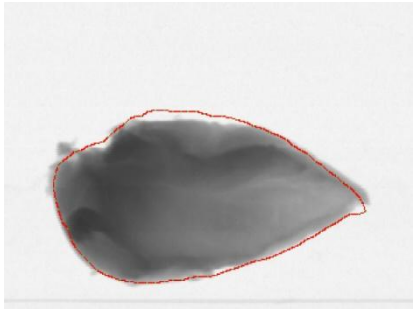
5



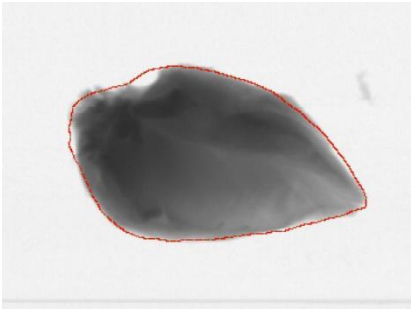
6



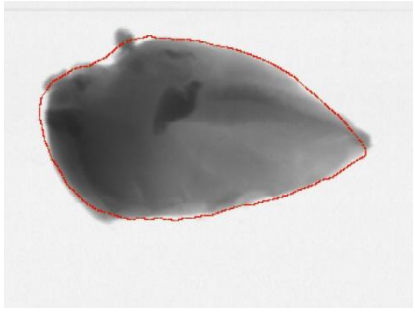
7



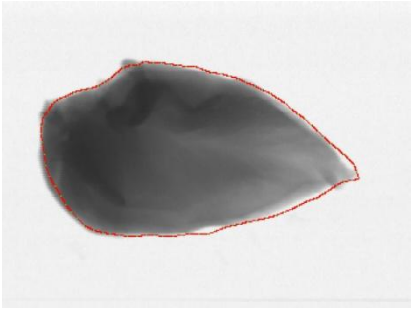
8

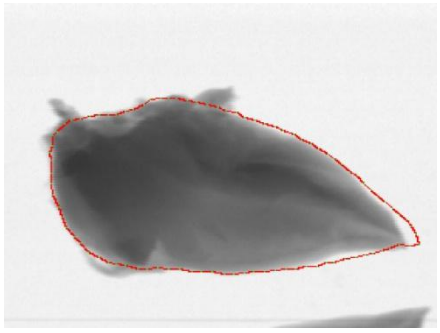


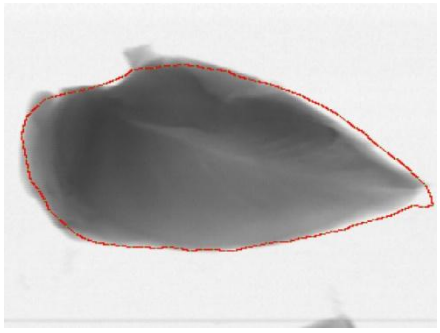
9

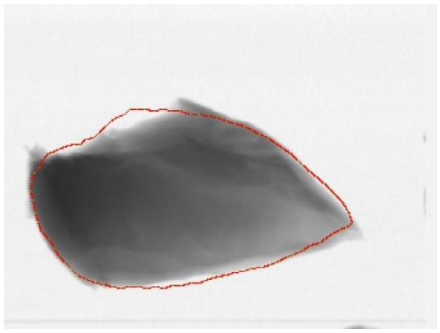


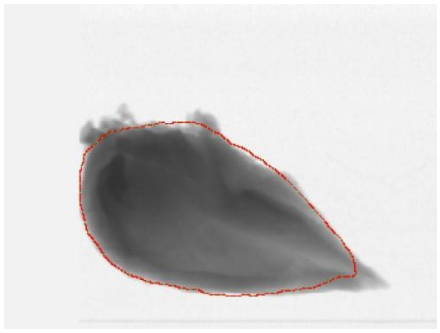
10

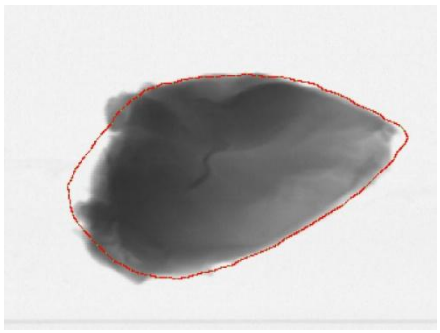


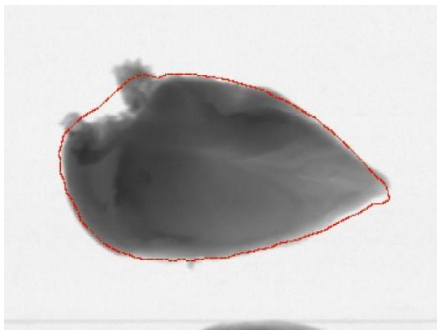
11 

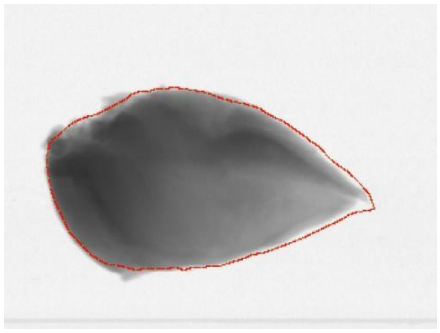
12 

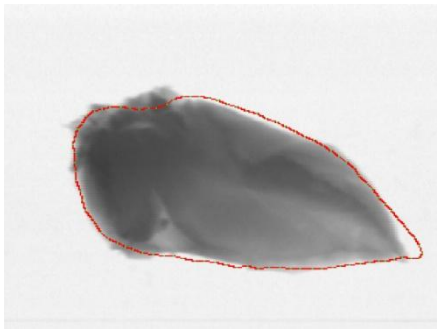
13 

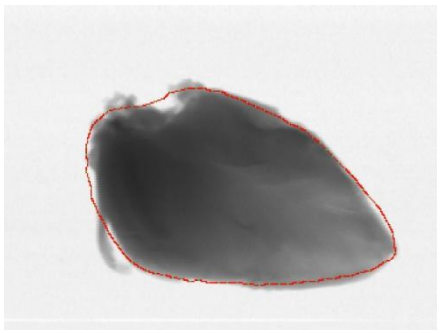
14 

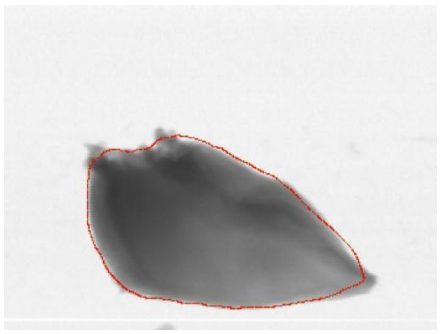
15 

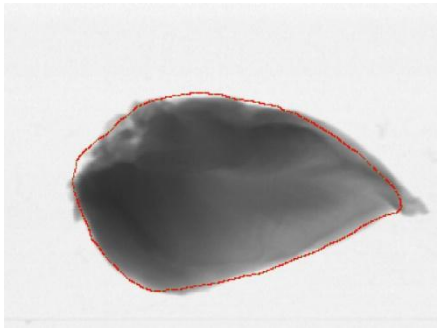
16 

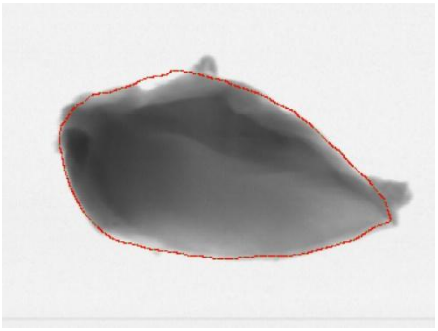
17 

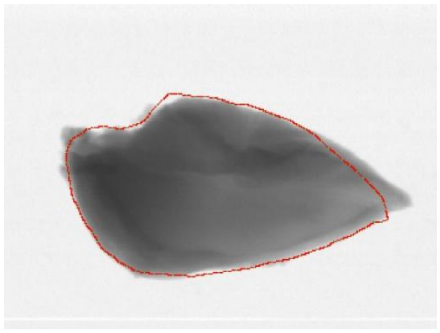
18 

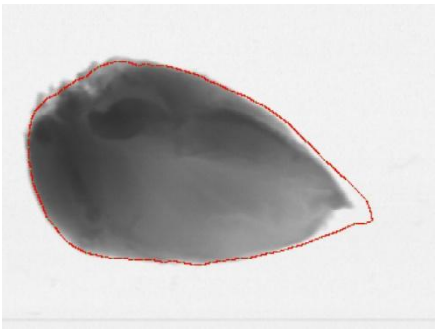
19 

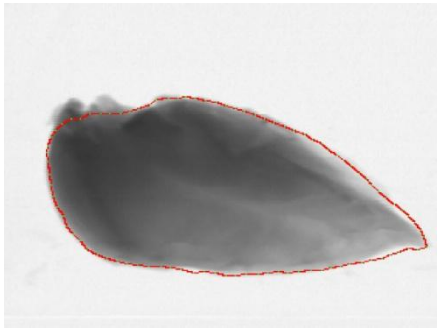
20 

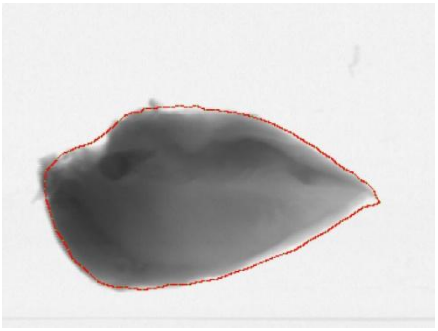
21 

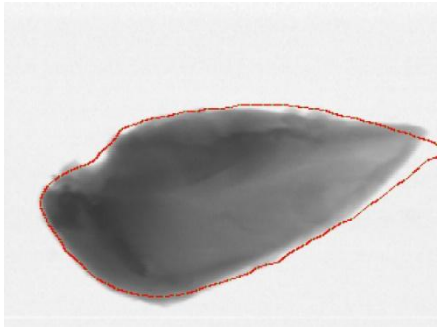
22 

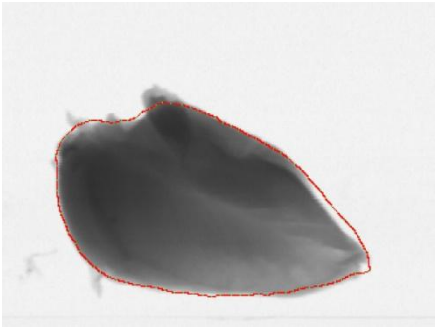
23 

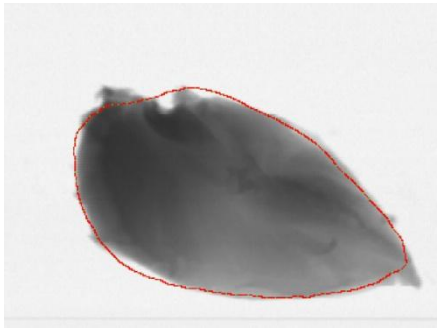
24 

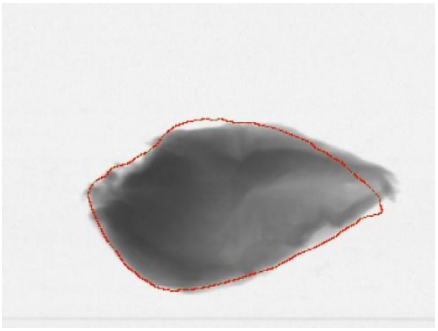
25 

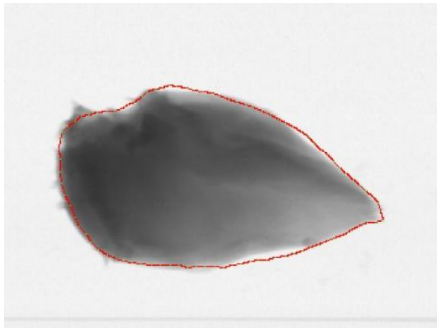
26 

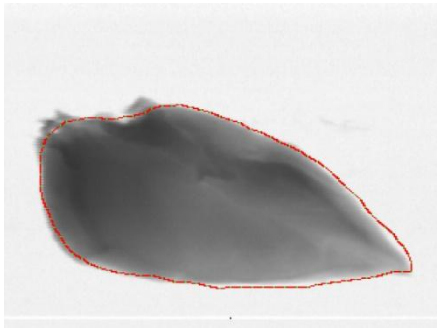
27 

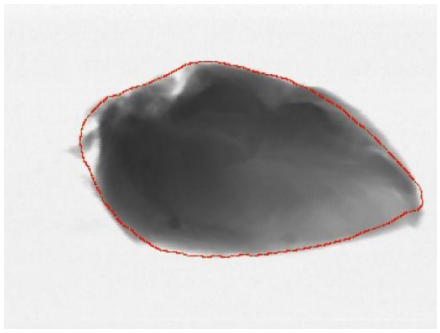
28 

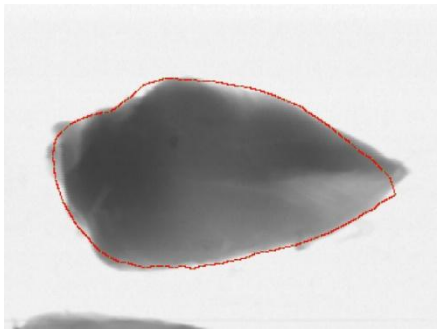
29 

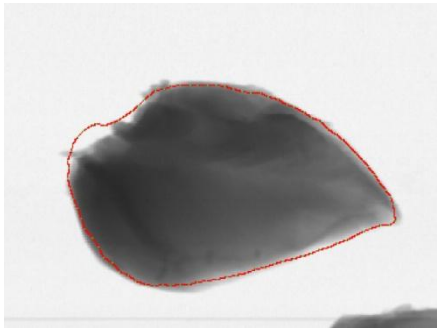
30 

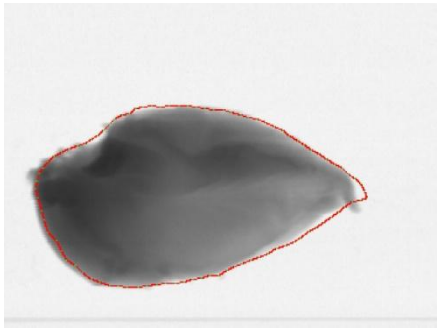
31 

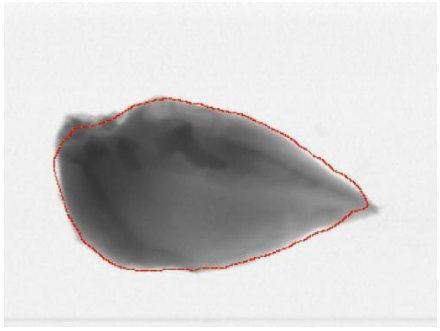
32 

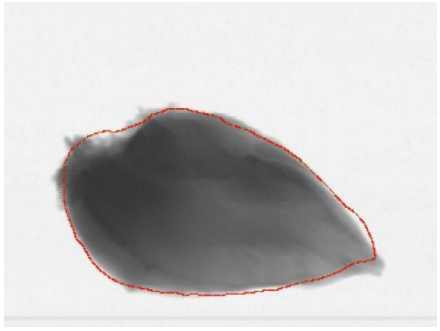
33 

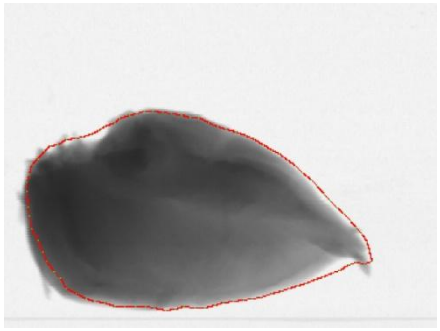
34 

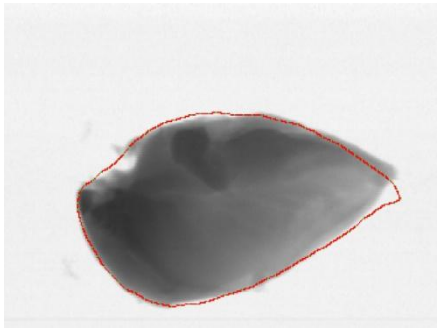
35 

36 

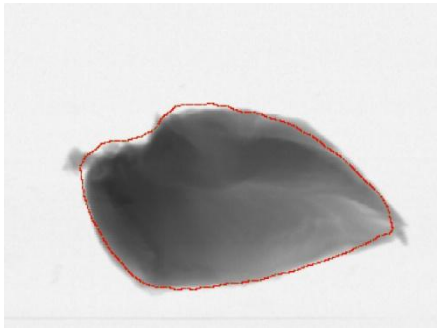
37 

38 

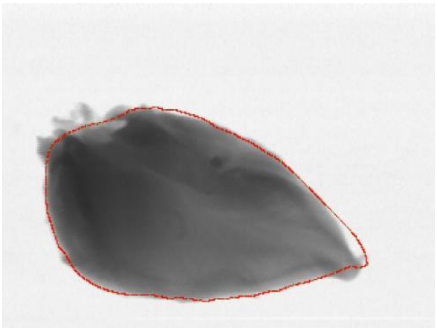
39 

40 

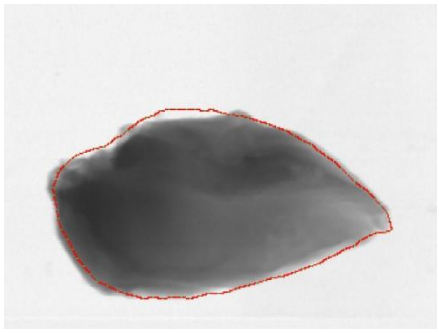
41



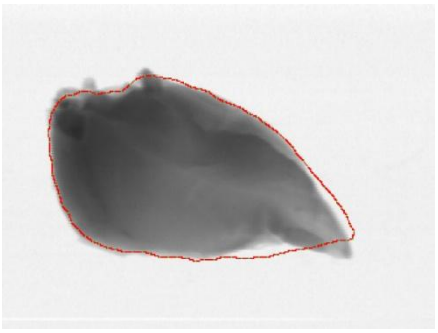
42



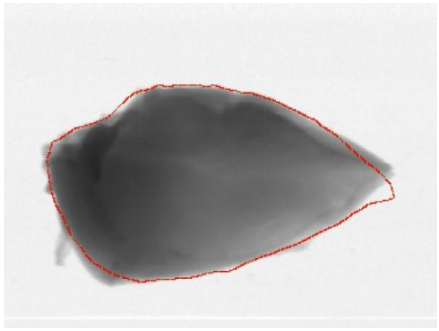
43



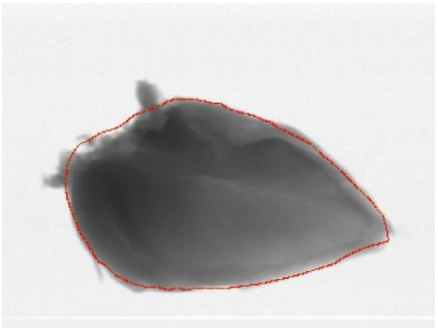
44



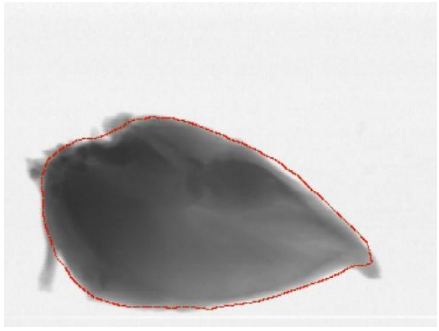
45



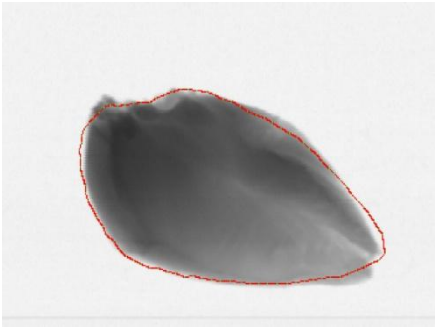
46



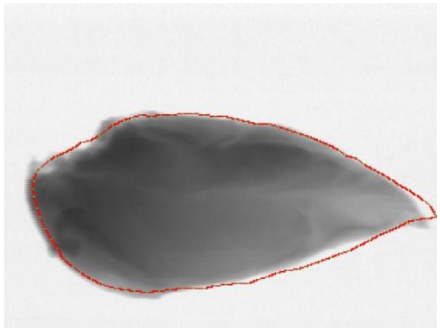
47



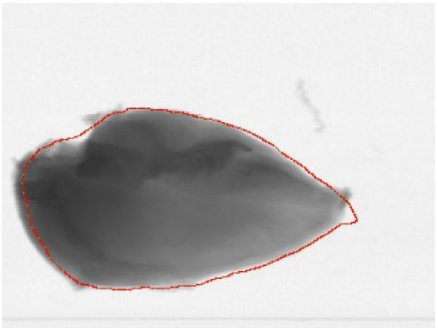
48



49

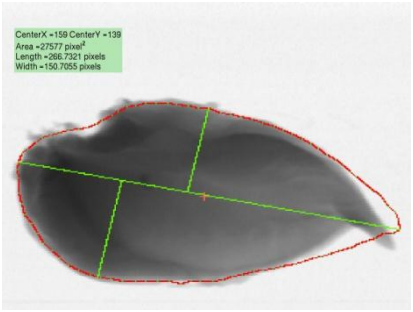


50

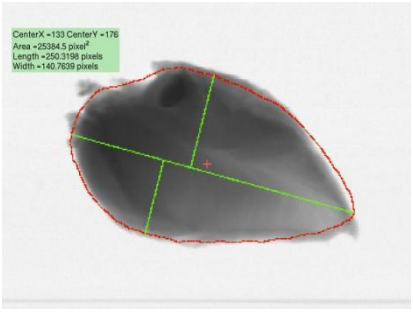


Appendix D – Information from Resulting ASM

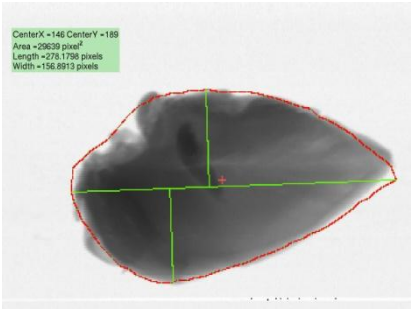
1



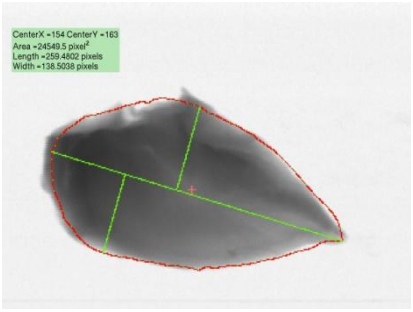
2



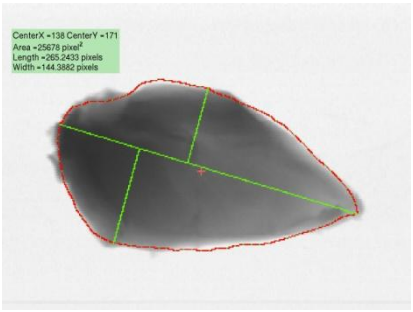
3



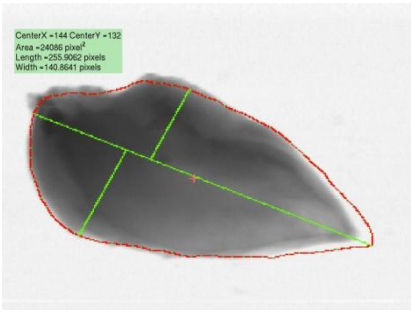
4



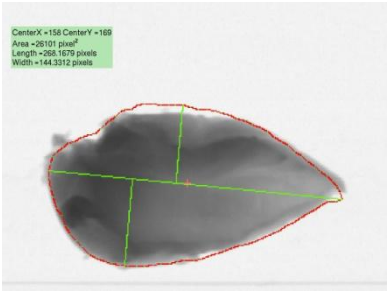
5



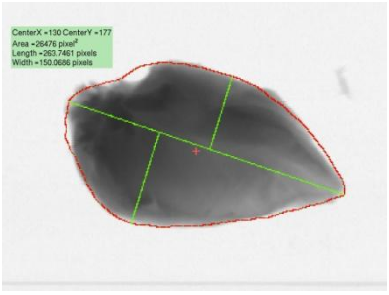
6



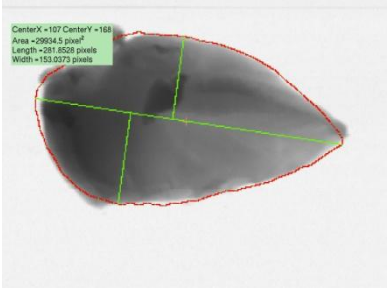
7



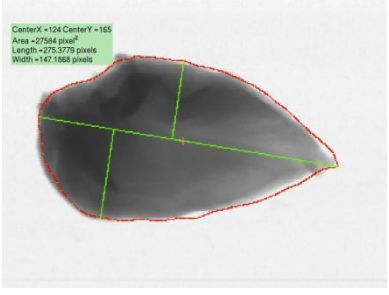
8



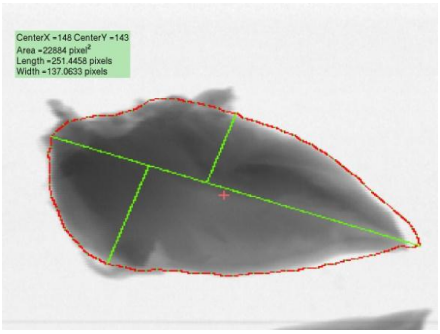
9



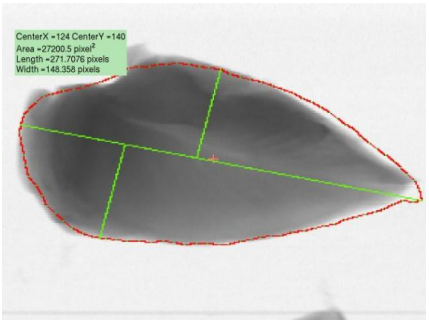
10



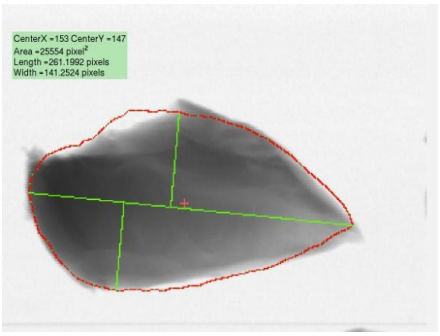
11



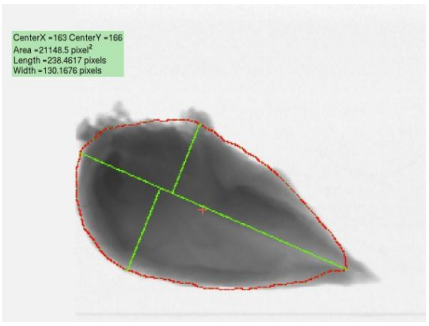
12



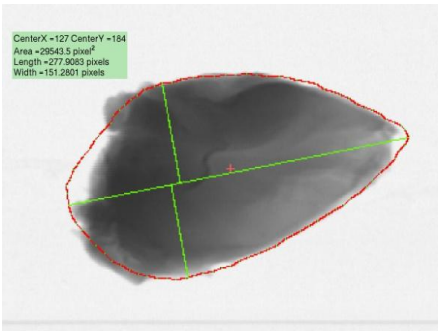
13



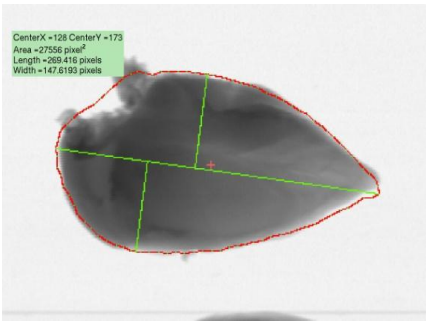
14



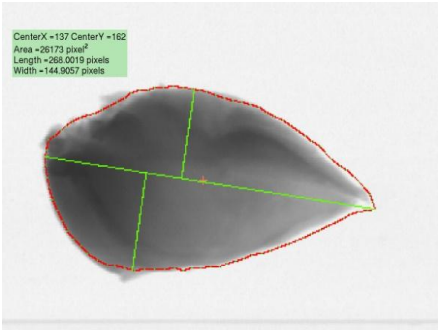
15



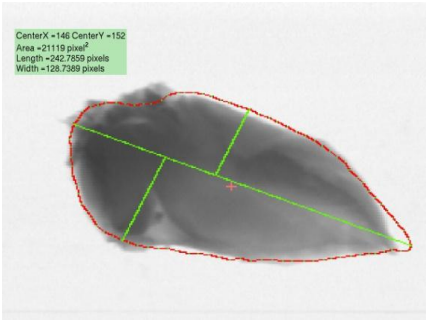
16



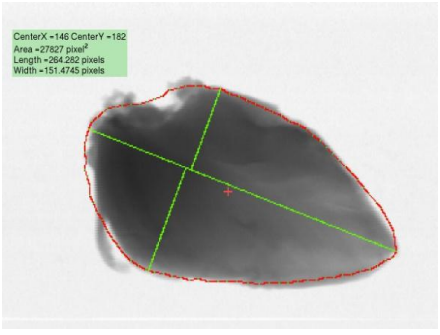
17



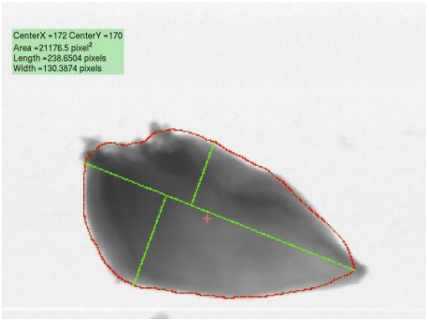
18



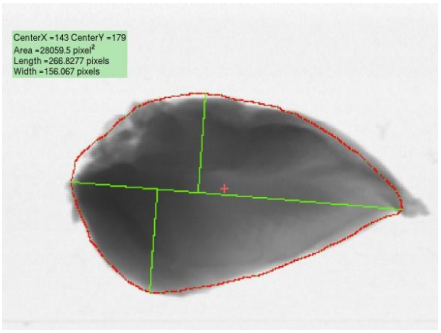
19



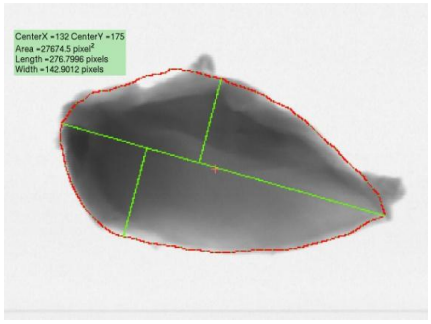
20



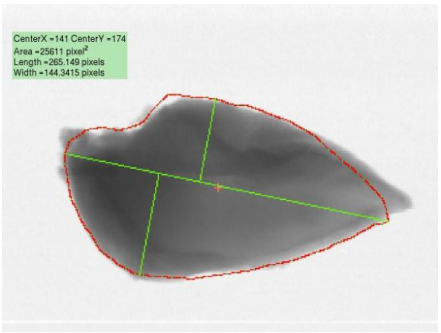
21



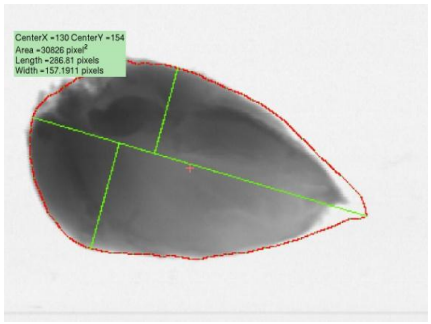
22



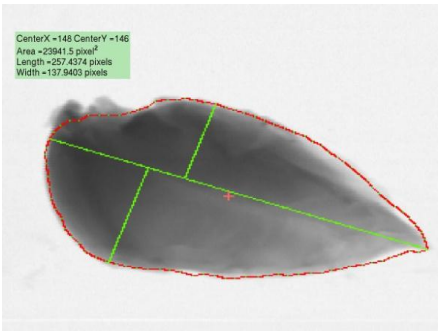
23



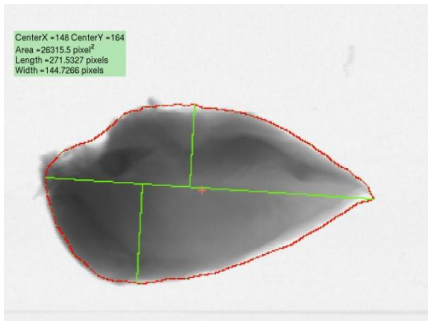
24



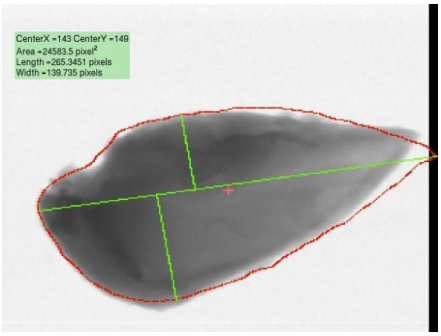
25



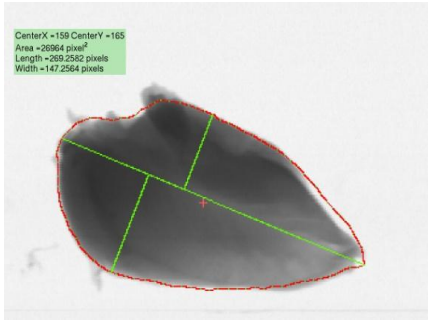
26



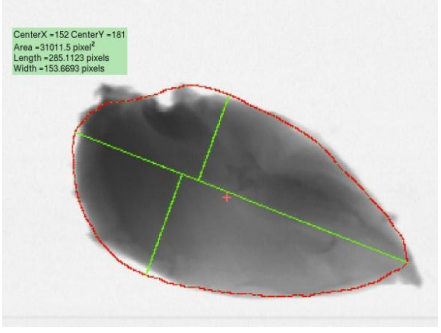
27



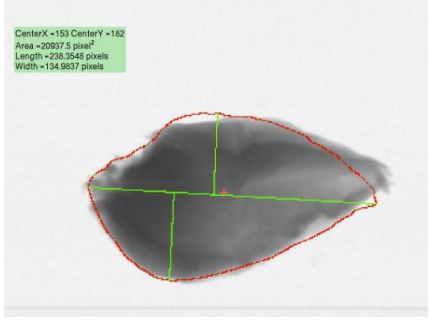
28



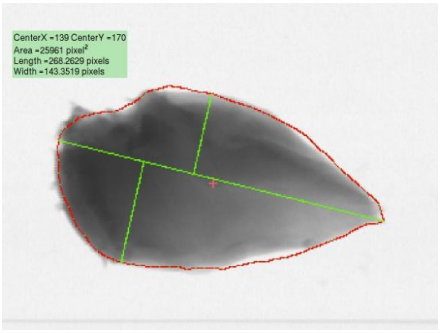
29



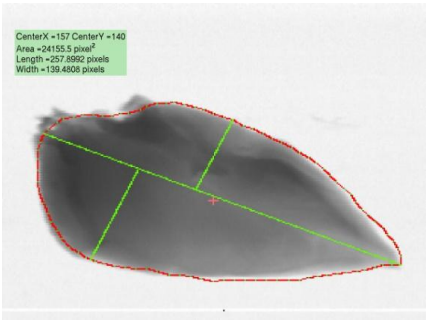
30



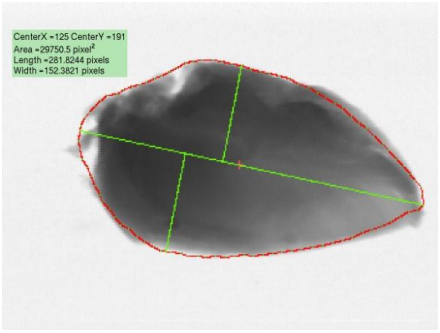
31



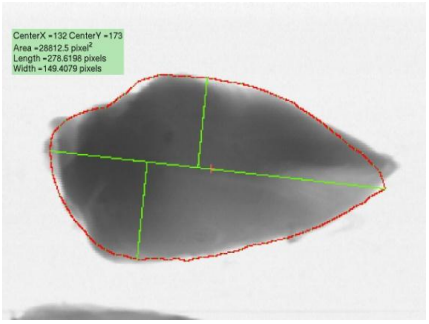
32



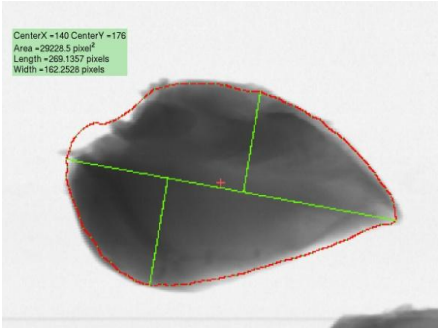
33



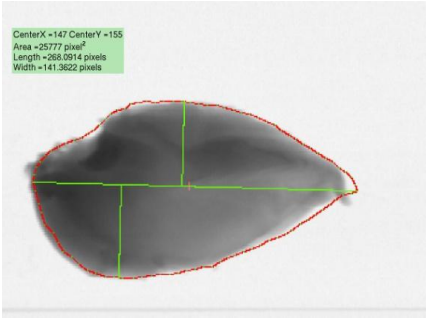
34



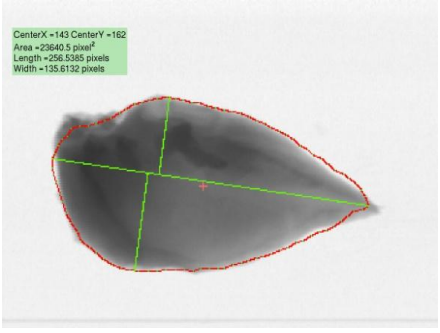
35



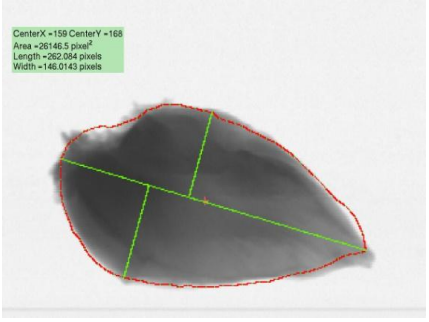
36



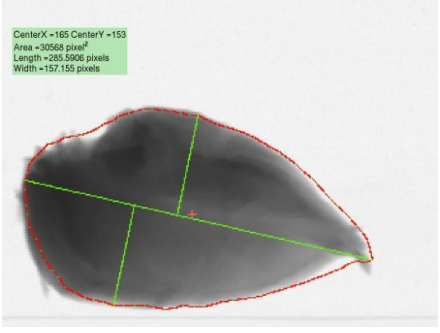
37



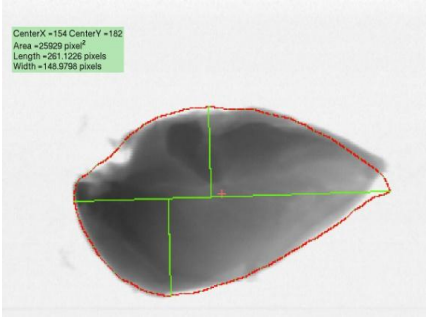
38



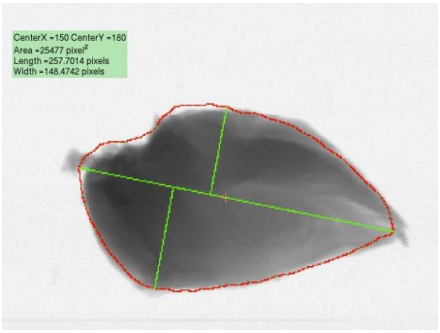
39



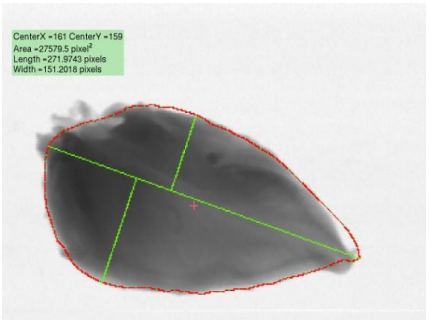
40



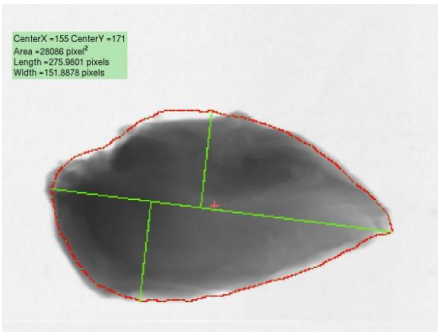
41



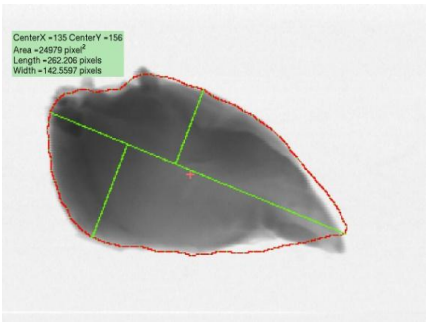
42



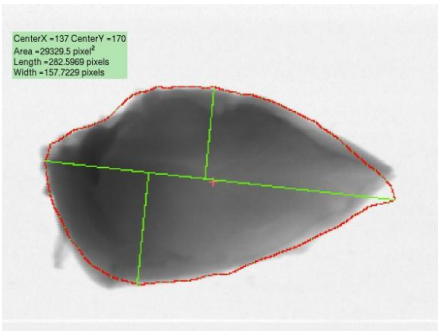
43



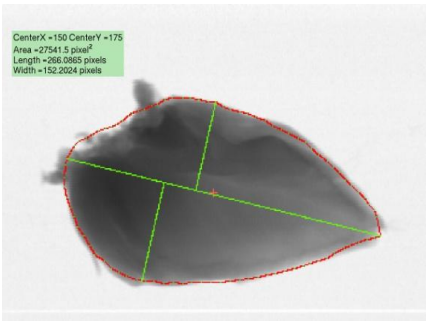
44



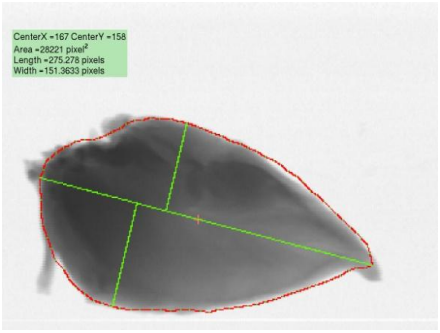
45



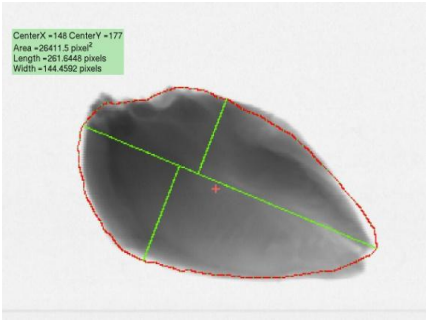
46



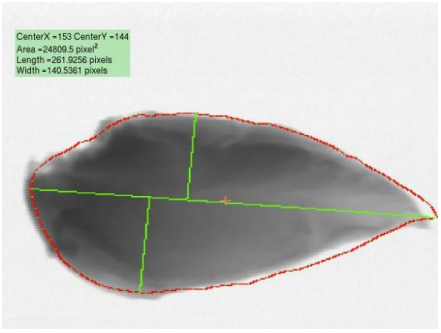
47



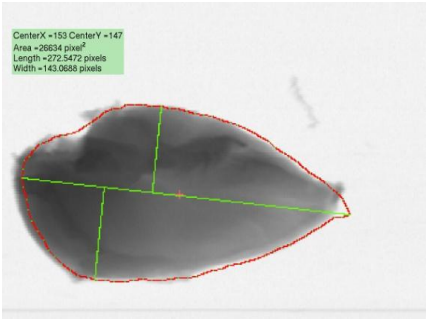
48



49

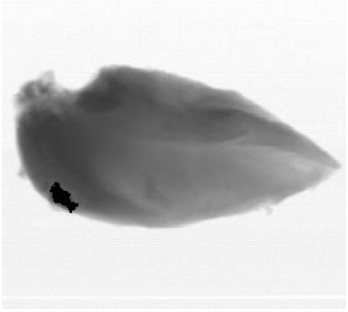


50

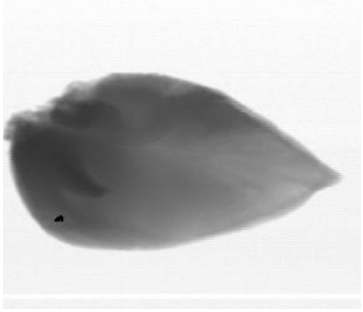


Appendix E – Test Images with Bone

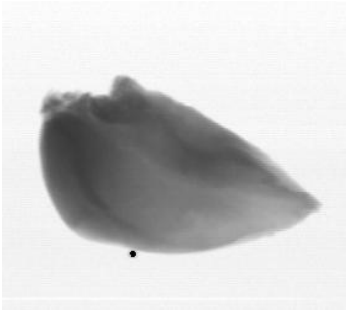
1



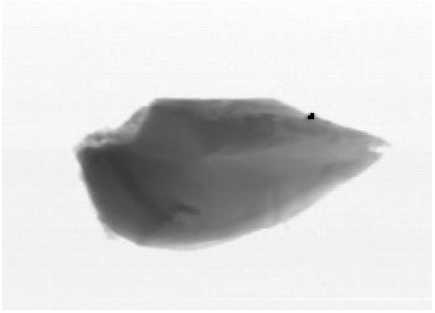
2



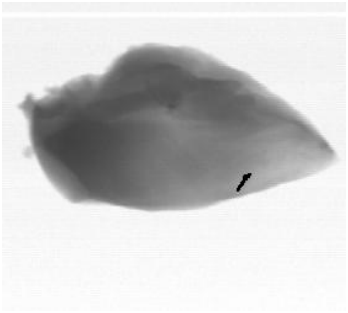
3



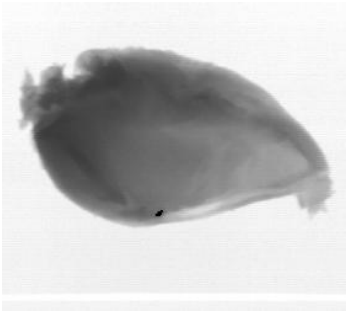
4



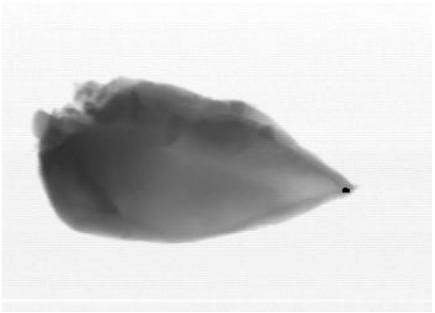
5



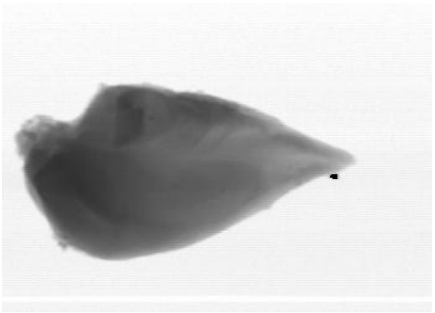
6



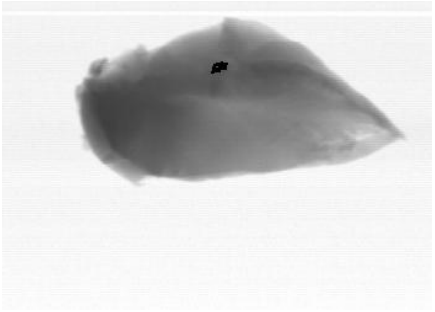
7



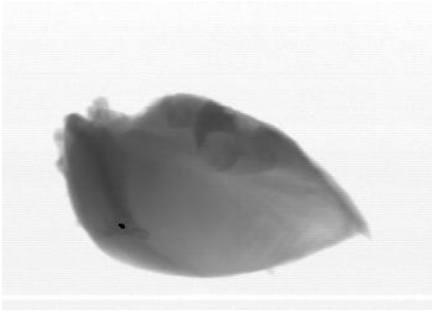
8



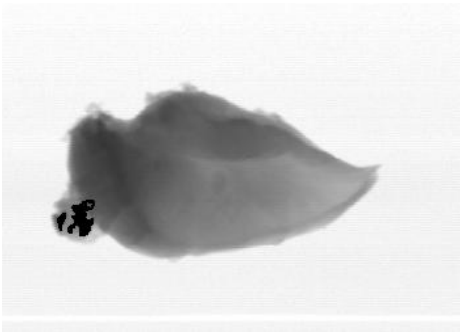
9



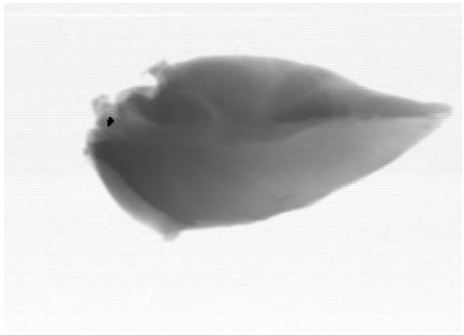
10



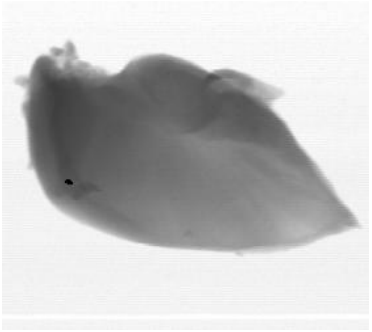
11



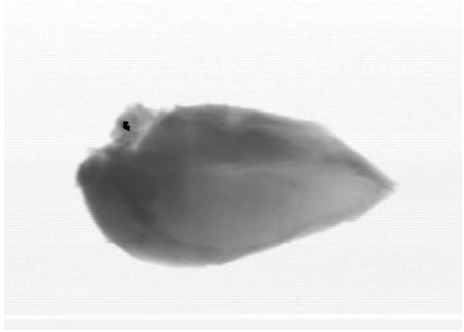
12



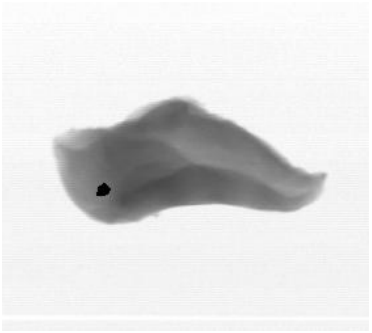
13



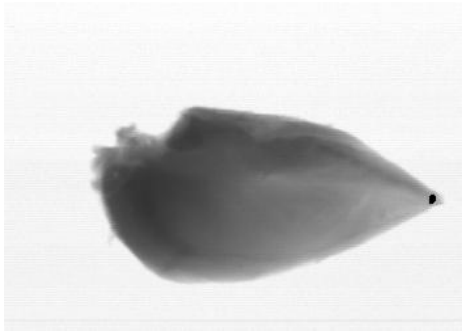
14



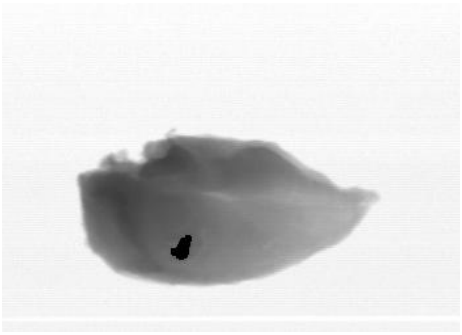
15



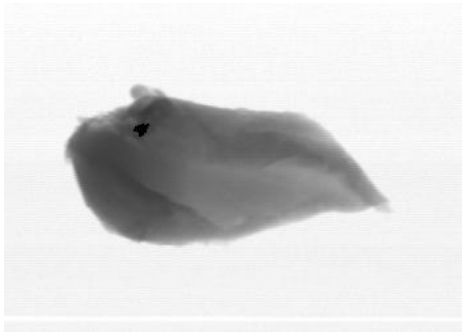
16



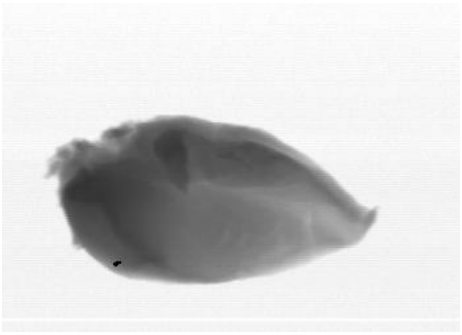
17



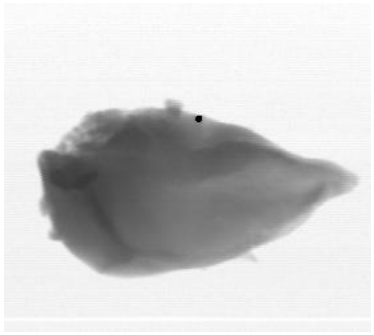
18

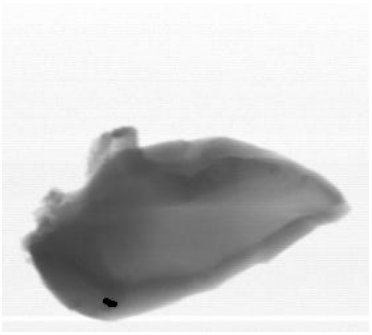


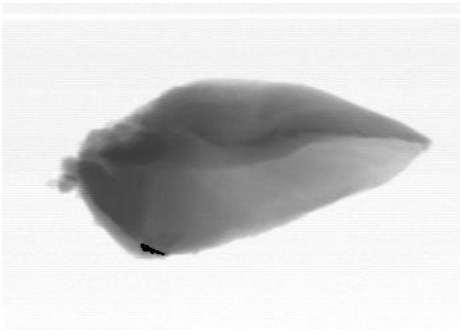
19

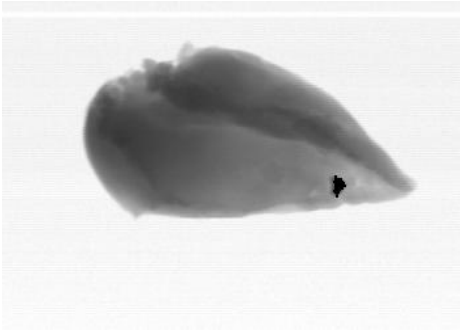


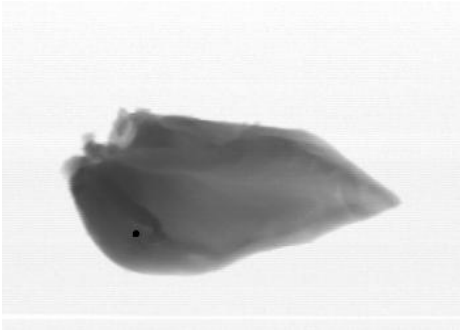
20

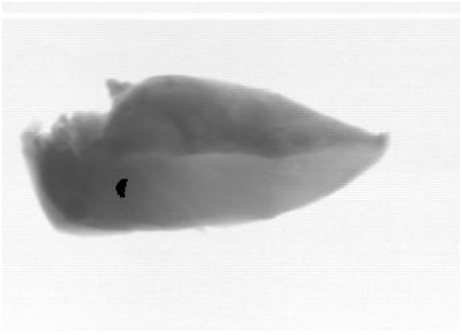


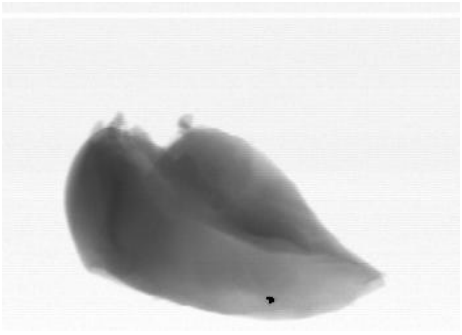
21 

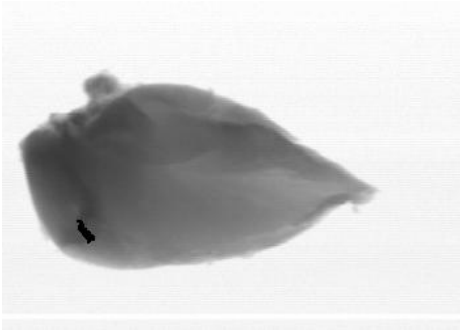
22 

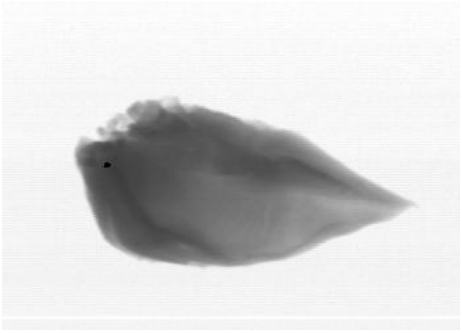
23 

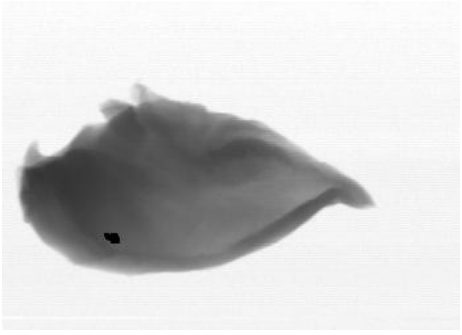
24 

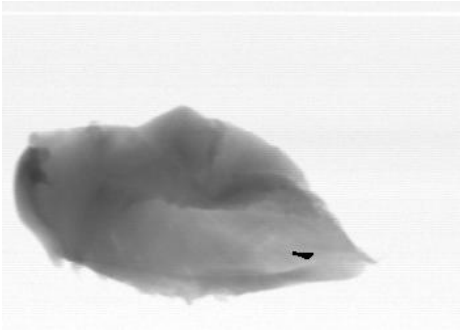
25 

26 

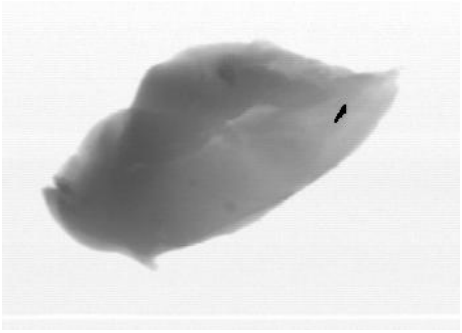
27 

28 

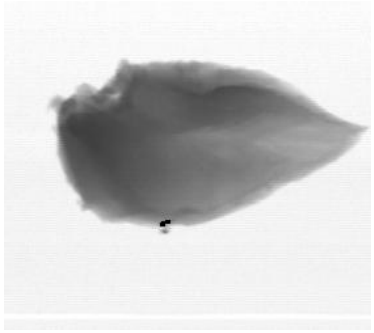
29 

30 

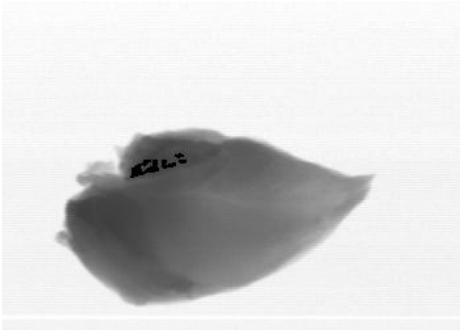
31



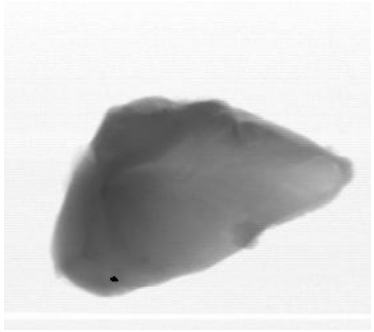
32



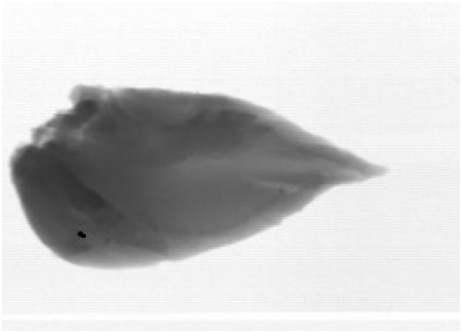
33



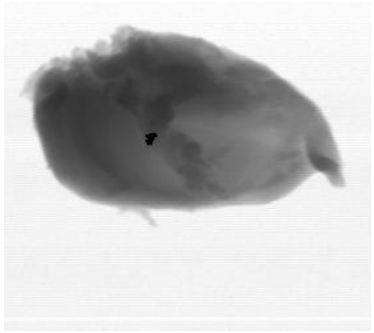
34



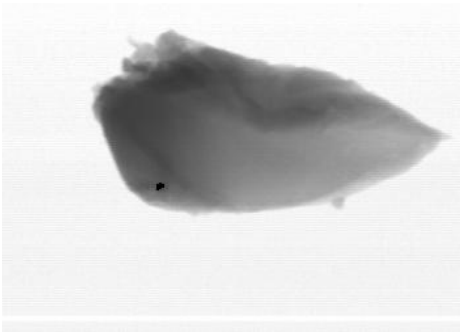
35



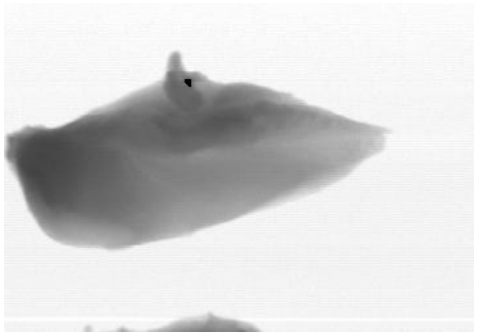
36



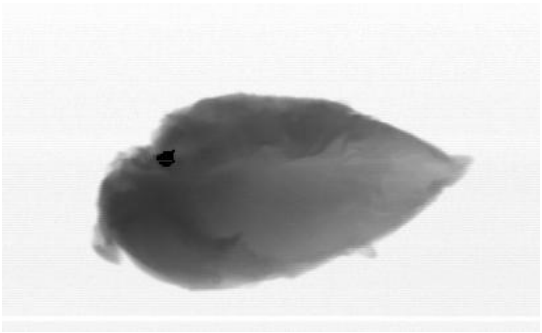
37



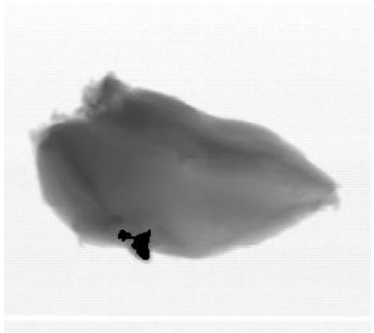
38



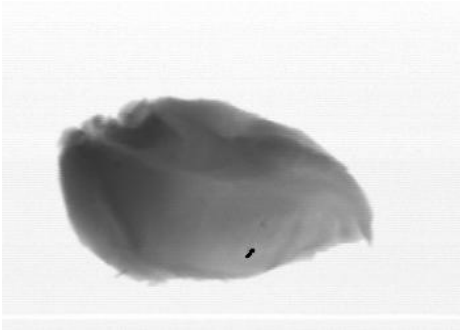
39



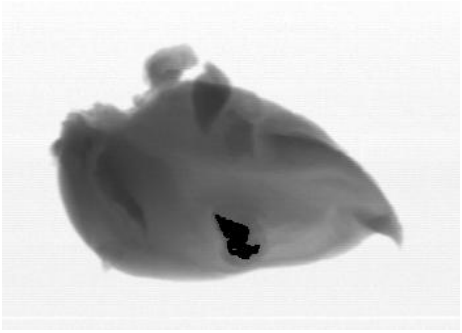
40



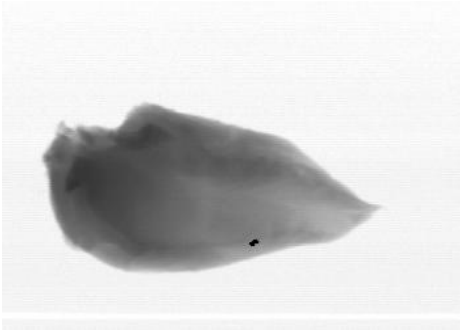
41



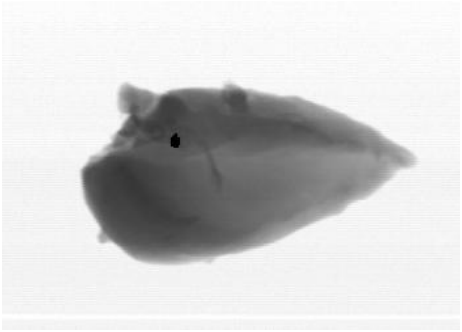
42



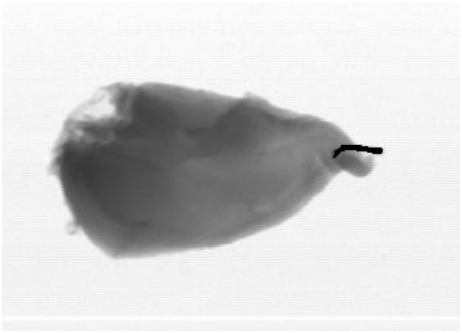
43



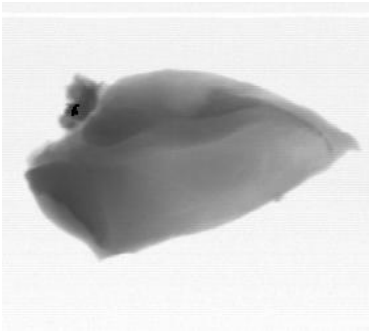
44



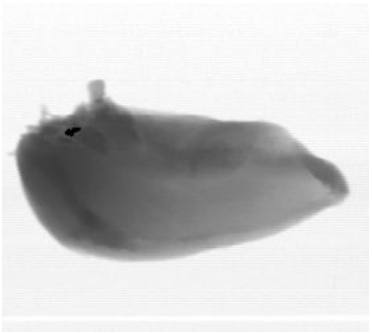
45



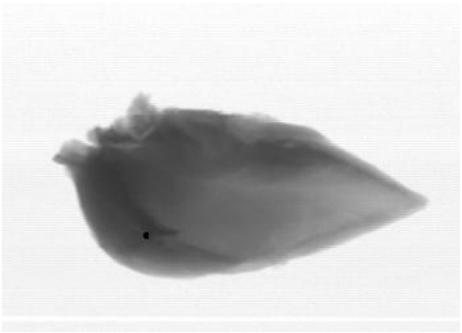
46



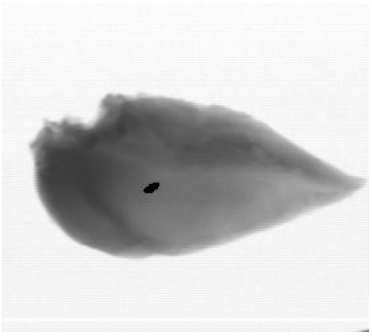
47



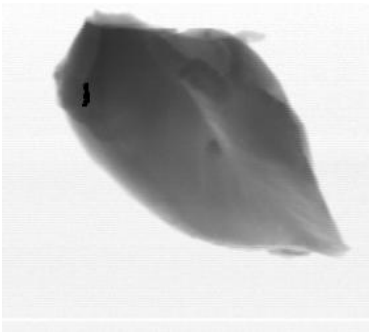
48



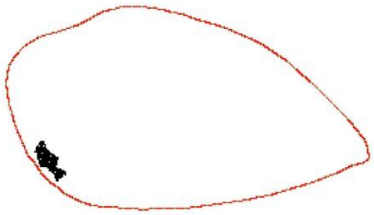
49



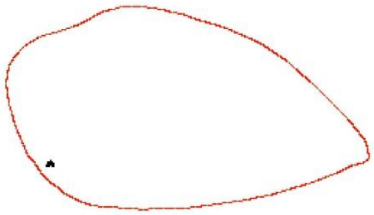
50



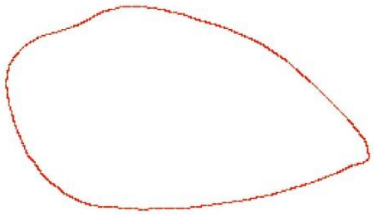
Appendix F –Bones Mapped to Reference Shape



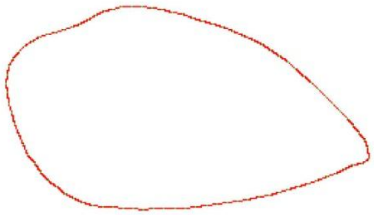
1



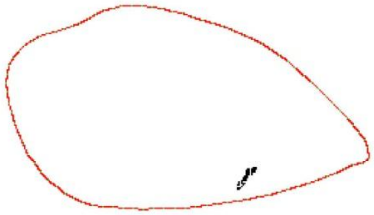
2



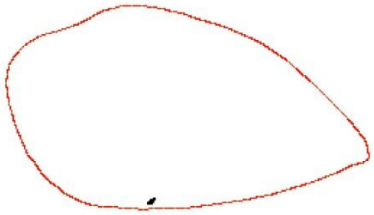
3



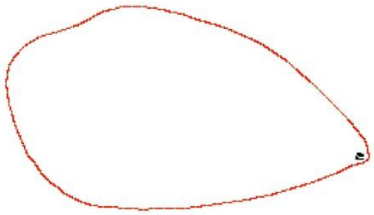
4



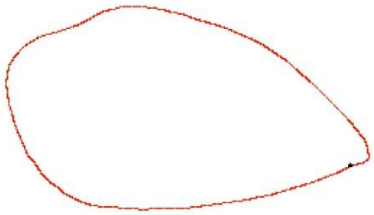
5



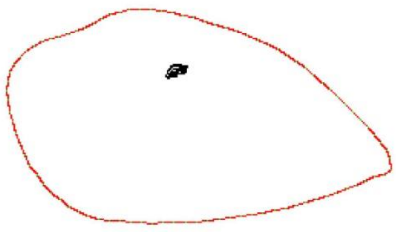
6



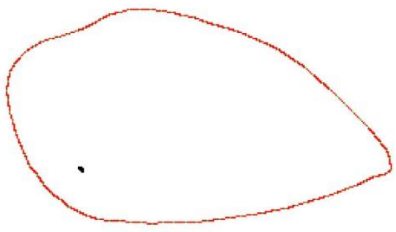
7



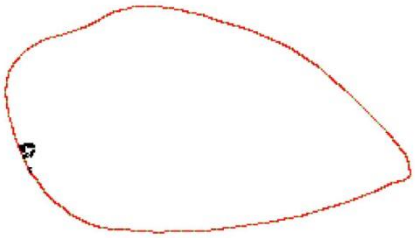
8



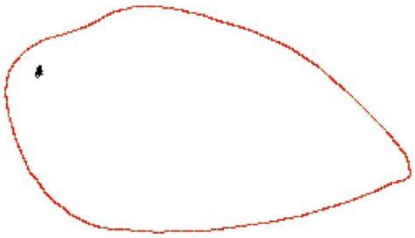
9



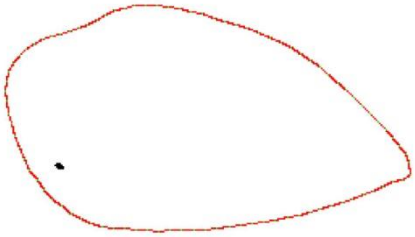
10



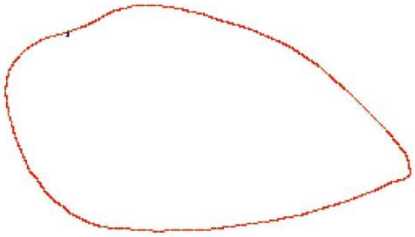
11



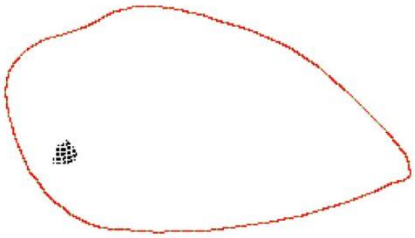
12



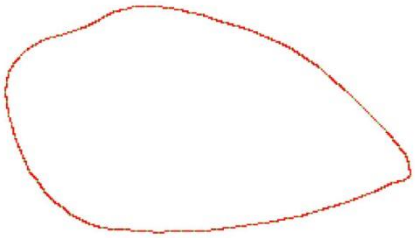
13



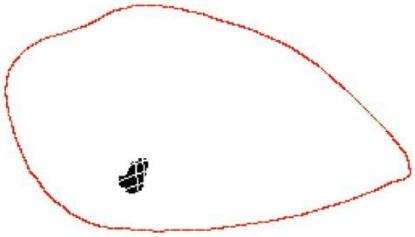
14



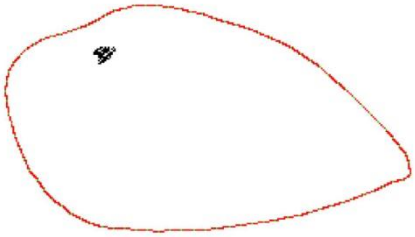
15



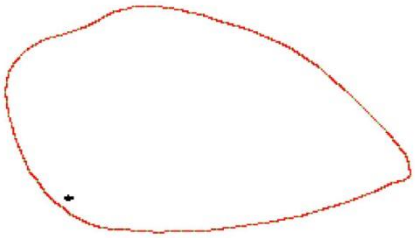
16



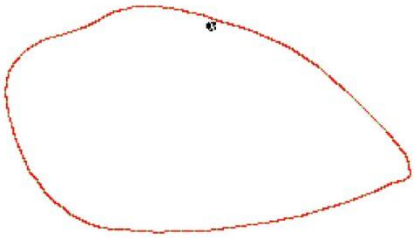
17



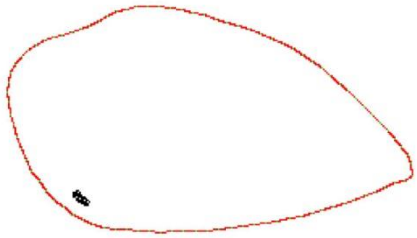
18



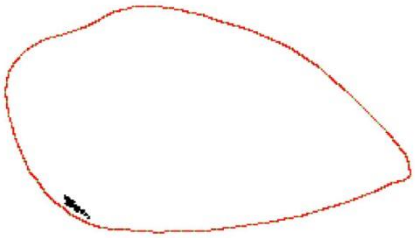
19



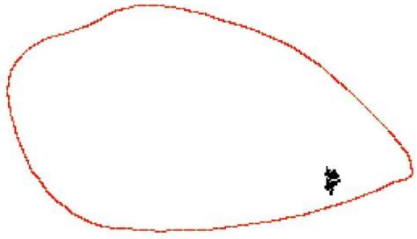
20



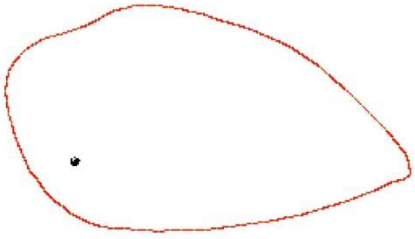
21



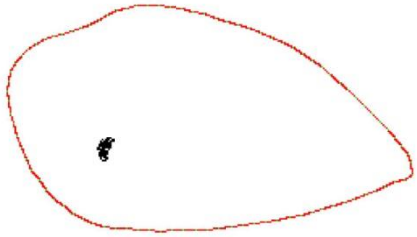
22



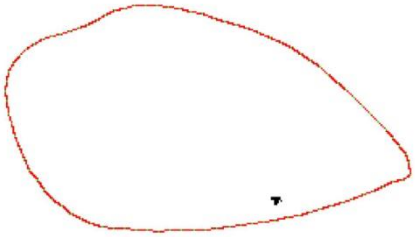
23



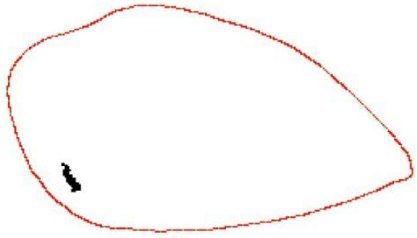
24



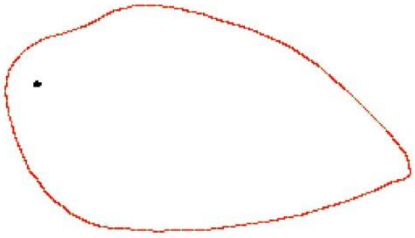
25



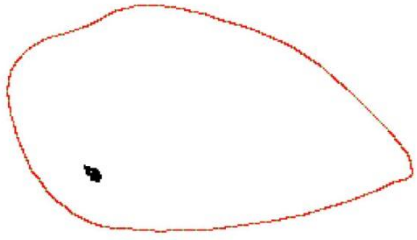
26



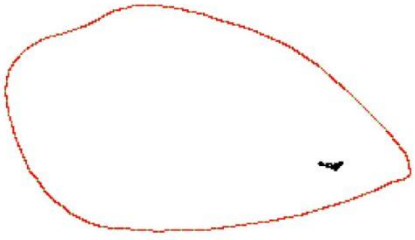
27



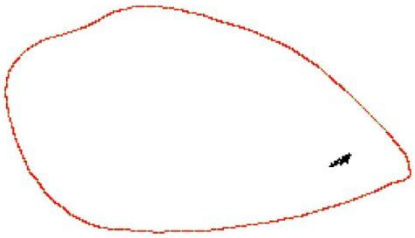
28



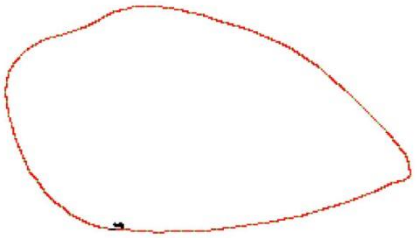
29



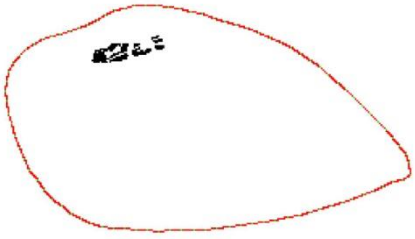
30



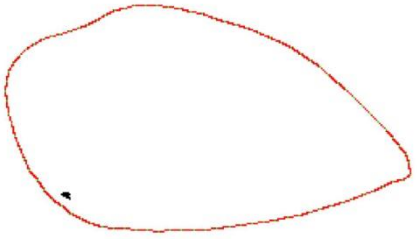
31



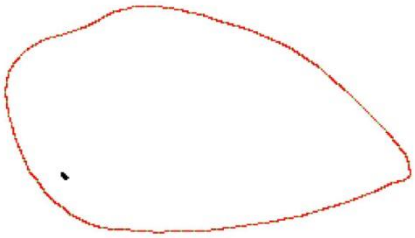
32



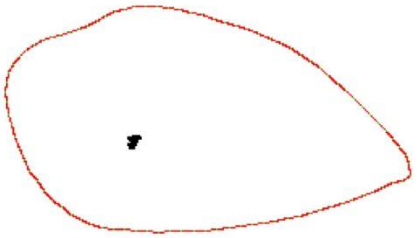
33



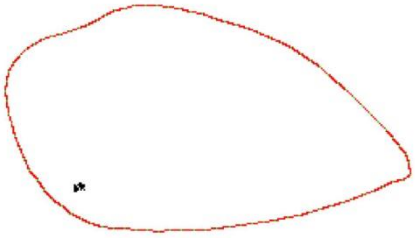
34



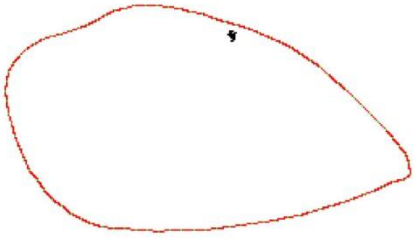
35



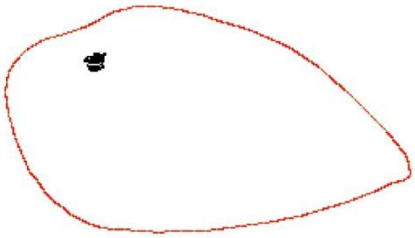
36



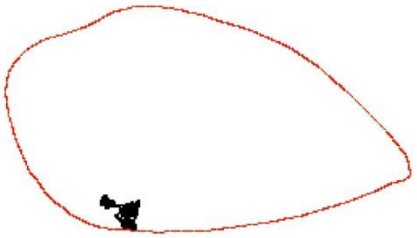
37



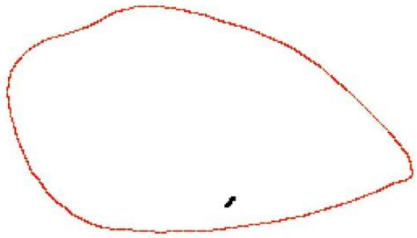
38



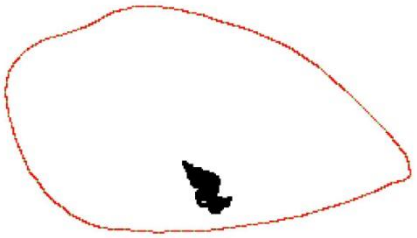
39



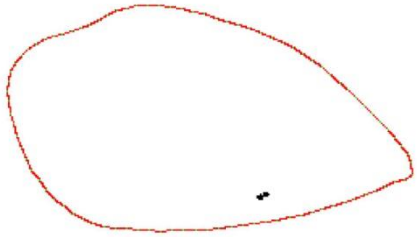
40



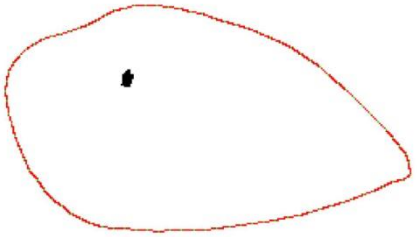
41



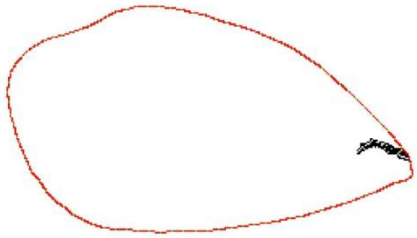
42



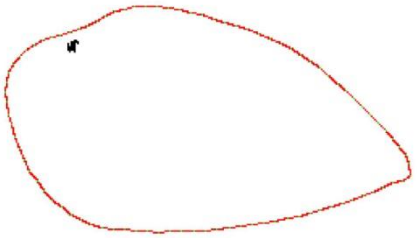
43



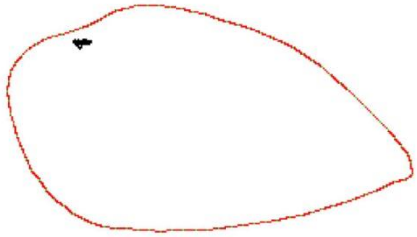
44



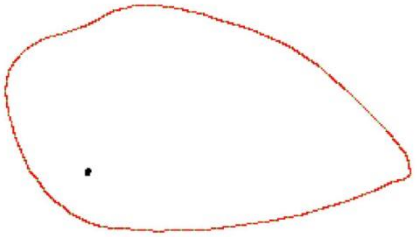
45



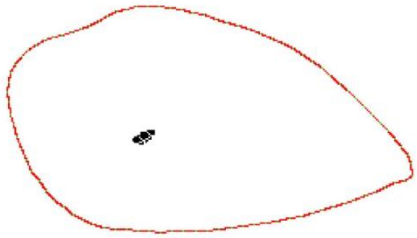
46



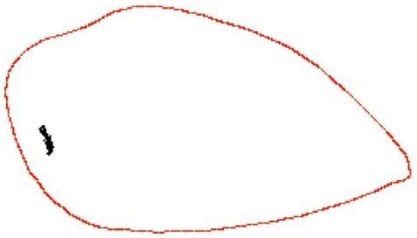
47



48



49



50