

Part III

Segmentation

Chapter 10

Introduction

Yoo [49] defined segmentation as the partitioning of a dataset into contiguous regions (or sub-volumes) whose member elements (e.g. pixels or voxels) have common, cohesive properties.

The human eye can partition an image in split-seconds into sub-volumes based on intensity differences, edge detection and shape recognition. To implement a computer program to do the same is one of the most challenging tasks in the field of image processing.

There are three main methods to partition an image. These are: manual, semi-automatic and automatic segmentation.

Manual segmentation is based on hand contouring by an experienced user. As each 3D image slice has to be segmented separately, manual segmentation is a very time consuming method. Due to the high level of human involvement, the result will be very accurate. However, previous studies [38] have shown relatively large variations between segmentations of the same image performed by different imaging experts.

Semi-automatic segmentation programs reduce processing time by at least one order of magnitude, as only a couple seed points and segmentation parameters have to be set. The computer then performs the subsequent image partitioning automatically, and the segmentation is terminated by manual interaction or a preset number of iterations. Due to the fact that the segmentation is now based on mathematical equations, the repeatability increases significantly, reaching 100% if the number of iterations are identical. Especially for comparative studies, this is a significant improvement. Nevertheless, the segmentation accuracy is now dependant on the image quality and the efficiency of the partitioning algorithms.

Fully automated methods do not require any human interaction. The segmentation parameters and termination condition are calculated based on the image information or fixed pre-settings. Usually these programs are written for a very specific

task and cannot be used for other segmentations. The accuracy of the segmentation is now wholly dependent on the implementation of the algorithms. However, if implemented correctly, the repeatability for all images is 100% and the processing time minimal.

There are already several freeware segmentation programs available on the Internet. Slicer [14] implemented by Gering et al., Analyze [37] created by Robb and Hanson or itkSNAP [50] maintained by Cognitica Corporation, to mention a few.

itkSNAP was chosen for this study for the following reasons:

- **Open source:** The complete source code of itkSNAP is available and can be freely changed as desired.
- **Program language:** itkSNAP is based purely on ITK [19], VTK [46] and FLTK [12]. These are the toolkits mainly used in this lab¹. Additionally, previously implemented functions can easily be included in the source code.
- **Semi-automatic segmentation:** itkSNAP is semi-automatic and therefore capable of segmenting diverse objects with a fair amount of accuracy, precision and speed.
- **Level set snake:** The user can choose whether to run a "geodesic active contour level set" or a "region competition level set" segmentation. Both methods are well known and perform well.
- **Graphical user interface:** The use of this program is simplified by the excellent graphical user interface (GUI). Furthermore, the segmentation results are directly applied to three 2D slices; sagittal, transverse and coronal through the input image. This significantly eases the control of the segmentation process for the user.

The functionalities of itkSNAP described in chapter 11 have been widely extended during this project. All extensions will be introduced in chapter 12.

After the implementation of all new features, they were also tested for accuracy and precision (see chapter 13). This evaluation greatly aided the final rat knee joint segmentation presented in chapter 14.

¹Bone Imaging Laboratory, University of Calgary, Canada

Chapter 11

itkSNAP

itkSNAP was originally implemented by Turello et al. at the Swiss Federal Institute of Technology. Over the last 6 years, various changes and extensions (such as the graphical interface and manual segmentation tool) were added by members of the University of North Carolina, Chapel Hill.

In 2004, this semi-automatic segmentation tool was included in the InsightApplications library [19].

11.1 Segmentation Method

SNAP stands for "Snake Automated Partitioning", which is the basic concept behind this segmentation program. In 1987, Kass et al. introduced the principle of "active snakes". In a two dimensional image, a snake can be visualized as a flexible rubber band. On one hand, the active snake tries to fit the image gradient, but on the other hand, the rubber band tries to minimize overall deflection energy. In 2002, a similar but more efficient approach was introduced by Goldberg et al. The snake is now driven by external and internal forces F , acting on the snake C .

$$\frac{\partial C(p, t)}{\partial t} = F(C(p, t), I) \cdot N(p) \quad (11.1)$$

$C(p, t)$ is the position of the snake parameterized by p , at time t . I denotes the image intensity and $N(p)$ the normal of the snake at p . Figure 11.1 shows a snake with internal and external forces acting on the rubber band.

In itkSNAP the force F is defined in two ways. The first is the "geodesic active contour" method by Caselles et al. [6], and the second is the "region competition" method first introduced by Zhu et al. [5]

11.1.1 Geodesic Active Contouring

This method uses image edges to define the forces acting on the snake. The edge image (feature image) $g(I)$ is obtained by calculating the intensity derivatives of the input image I . A detailed description of this image conversion is given in subchapter [12.2](#).

The total force F is defined as a weighted sum of a balloon force F_{bal} , a curvature force F_{curv} and an advection force F_{adv} .

$$F = \alpha \cdot F_{bal} + \beta \cdot F_{curv} + \gamma \cdot F_{adv} \quad (11.2)$$

α , β and γ are force weights adjustable by the user. The balloon force is large in areas of uniform intensities and decelerates the snake, if the edge filtered image has low values (see figure [11.1](#)).

$$F_{bal} = g(I) \quad (11.3)$$

The curvature force considers the snake's curvature κ at point p and, if the curvature is convex, pulls the snake inwards or, if the curvature is concave, pushes the snake outwards. Thus, the curvature force smoothens the segmentation.

$$F_{curv} = -\kappa \cdot g(I) \quad (11.4)$$

For most feature images $g(I)$, the minimum value of zero is rarely reached. Even in border areas with very strong contrast. Therefore, force F is rarely zero and the snake continues to grow. To prevent the snake from "leaking", a third force is needed.

The advection force accelerates the snake's expansion if the curve normal is parallel to a strong feature image edge. If the snake passes the edge, the advection force will start to pull the snake inwards and finally stop on the object's surface.

$$F_{adv} = -\nabla g(I) \cdot N(p) \quad (11.5)$$

11.1.2 Region Competition

Region competition is driven by a probabilistic model of image intensity. The probability that an image voxel belongs to the object $P_{obj}(I)$ or to the background $P_{bg}(I)$ is calculated. The snake converges to places where the probability is equal. The object intensity probability is given as

$$P_{obj}(I) = \frac{1}{\pi} \cdot \left(\tan^{-1} \left(\frac{I - a}{\kappa \cdot (b - a)} \right) + \tan^{-1} \left(\frac{b - I}{\kappa \cdot (b - a)} \right) \right) \quad (11.6)$$

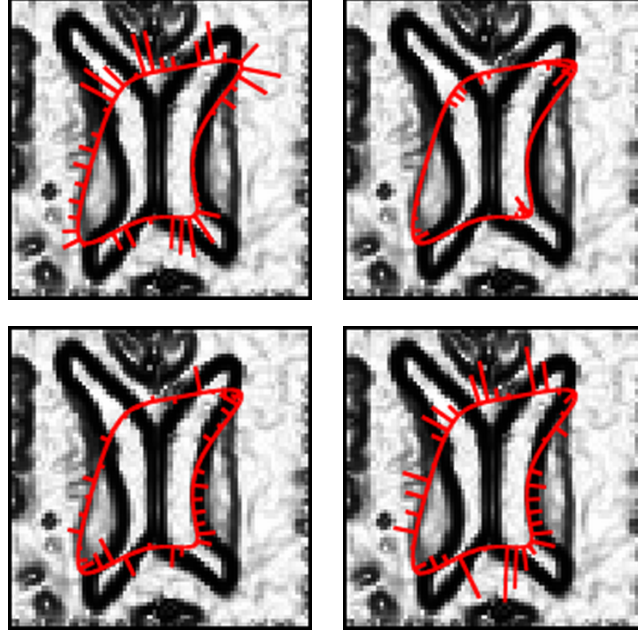


Figure 11.1: Upper left: Balloon force acting on the snake. Upper right: Curvature force acting on the snake. Lower left: Advection force acting on the snake. Lower right: Combined forces acting on the snake.

where a and b are the lower and upper threshold values defined by the user and κ is a curve smoothing factor. The background probability is

$$P_{bg}(I) = 1 - P_{obj}(I) \quad (11.7)$$

Thus, the feature image results in

$$g(I) = P_{obj}(I) - P_{bg}(I) = 2 \cdot P_{obj}(I) - 1 \quad (11.8)$$

The total region competition force F only consists of the α -weighted probabilistic image force F_{prob} and a β -weighted curvature force F_{curv} .

$$F = \alpha \cdot F_{prob} + \beta \cdot F_{curv} , F_{prob} = g , F_{curv} = -\kappa \quad (11.9)$$

Hence, the snake grows in foreground areas according to the two acting forces. In addition, this method is capable of letting the snake shrink in background areas, where the probability $P_{obj}(I) < 0.5$.

11.1.3 Level-Set Method

In the previous subchapters, the total force acting in a single point on the snake curve, has been evaluated. In 3D images, a sphere (balloon), instead of a curve, is used. To calculate all the forces F acting on the balloon-snake at each time step, the level set method by Osher et al. is used.

$$\frac{\partial \phi}{\partial t} = F \cdot |\nabla(\phi)| \quad (11.10)$$

The active snake is defined as a level set of the scalar-value function ϕ . Usually, points where $\phi < 0$ are defined as being inside the snake, whereas points with $\phi > 0$ are defined as outside the snake-surface.

An exhaustive iterative computation of the forces over the whole image area would be too time consuming. Thus, diverse algorithms like sparse field, narrow band, and dense level set have been implemented to speed up the evolution by reducing the number of voxels taken into consideration.

The explicit procedure used to find the final snake parameters are beyond the scope of this snake contouring overview. A detailed description can be found in [43], [49] and [50].

Chapter 12

mySNAP

mySNAP is an extended version of the original itkSNAP segmentation program. Among the new features, the user has the option to choose between four or five different pre-filters to denoise the original input image (see subchapter [12.1](#)). For the geodesic active contour snake, three more sophisticated edge filters have been implemented. The parameters of all filters can be user-adjusted interactively and the resulting feature image changed on-the-fly. This greatly simplifies the parameter optimization and saves time.

Furthermore, mySNAP is compatible with images stored in AIM format. The image loading and saving process is governed by a user-friendly wizard. All filter parameters and additional segmentation information is saved in the AIM-internal log file.

Figure [12.1](#) shows a flow chart of all new features provided in mySNAP. The following subchapters will focus on the functionality of these extensions.

12.1 Denoise Filter

An image filter changes the appearance of an image, and are generally used to increase the image information content.

The filters presented in this subchapter try to remove noise, to increase the quality of the final feature image used in the segmentation.

All four denoise filters implemented in mySNAP are briefly described below. A fifth filter (bilateral image filter [\[18\]](#)) is also implemented, although due to its very slow performance, it was removed from the GUI again. However, the filter code is still available in the mySNAP program and can be reactivated through a short series of changes in the GUI settings.

If the user enters the segmentation mode in mySNAP, the wizard first asks to

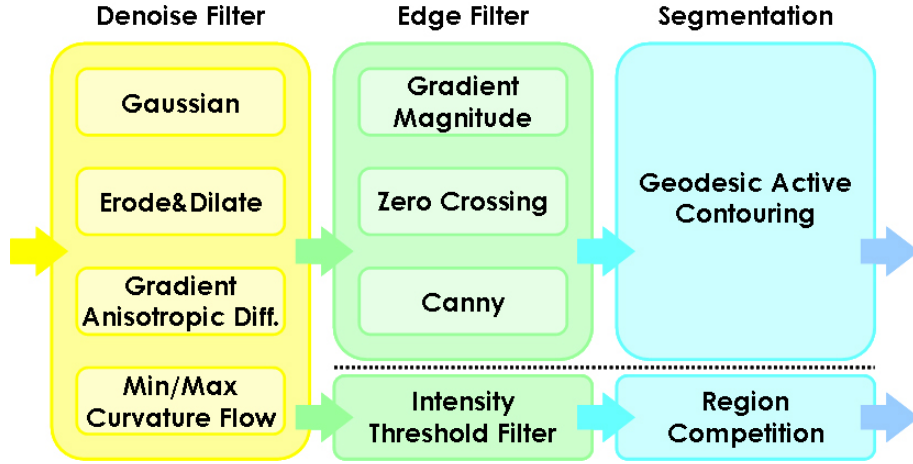


Figure 12.1: Flow chart of mySNAP extensions.

apply a denoise filter to the original image. This increases the image quality for the subsequent generation of feature images. Pre-filtering can also be skipped by checking the appropriate button in the denoise filter window.

12.1.1 Discrete Gaussian Image Filter

Blurring is a traditional approach for removing noise from an image. The goal is to attenuate high frequency components in an image by applying a Gaussian kernel. A N -dimensional Gaussian distribution $G(r)$ has the form

$$G(r) = \frac{1}{\sqrt{2\pi} \cdot \sigma^2} \cdot e^{-r^2/(2\sigma^2)} \quad (12.1)$$

where r is the blur radius and σ is the standard deviation of the distribution. Figure 12.2 shows the corresponding discrete one-dimensional curve of the convolution kernel. The user defines a maximum error, from which the kernel size is calculated.

In figure 12.3 the effect of the discrete Gaussian image filter is illustrated.

12.1.2 Gradient Anisotropic Diffusion Filter

A major drawback of Gaussian smoothing (or linear filtering in general) is that each voxel is treated identically. There is no distinction between image noise and edge borders.

Non-linear diffusion tries to distinguish between noise and edges. In edge regions of high gradient values, smoothing is reduced, whereas in all other regions, image

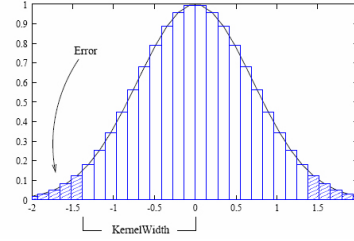


Figure 12.2: Discrete Gaussian distribution with a user defined kernel size.

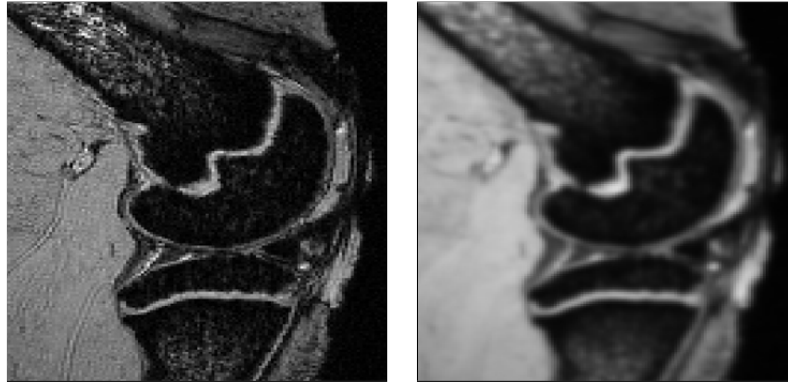


Figure 12.3: Left: Original test image. Right: Gaussian blurred test image.

filtering is enhanced.

The gradient anisotropic diffusion filter produces a Gaussian smoothed image, which is the solution to the heat equation 12.2, with a variable conductance term to limit edge smoothing.

$$\frac{\partial g(x, y, z, t)}{\partial t} = \nabla \cdot \nabla(g(x, y, z, t)) \quad (12.2)$$

$g(x, y, z, 0)$ is the input image and $g(x, y, z, t) = G(\sqrt{2t}) \otimes f(x, y, z)$ is the cross-product of the original image $f(x, y, z)$ with $G(\cdot)$, a time step t dependent Gaussian distribution.

For this particular filter, the conductance term is a function of the gradient magnitude of the image at each point. Hence, the diffusion in edge regions is reduced as shown in figure 12.4.

12.1.3 Min/Max Curvature Flow Filter

As the gradient anisotropic diffusion filter, the min/max curvature flow algorithm is a non-linear edge-preserving smoothing filter.

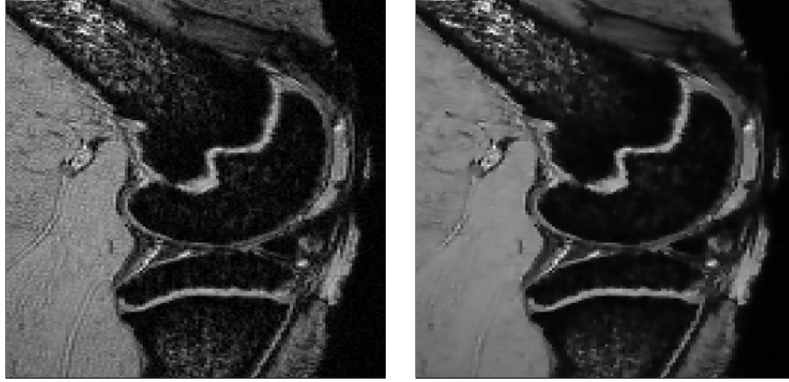


Figure 12.4: Left: Original test image. Right: Gradient anisotropic diffusion filtered test image.

The curvature flow filter uses a level set approach, where iso-intensity contours are defined as the level sets. The level sets I then undergo a diffusion equation where speed is proportional to the curvature κ of the contour.

$$I_t = \kappa \cdot |\nabla(I)| \quad (12.3)$$

Level sets with high curvature will diffuse faster than level sets with low curvature. Hence, noise particles will be smoothed faster than larger object borders (see figure 12.6).

For the min/max approach κ is defined as

$$\kappa = \begin{cases} \max(\kappa, 0) : a < t \\ \min(\kappa, 0) : a \geq t \end{cases} \quad (12.4)$$

where a is the average intensity calculated from all neighboring voxels within a user-defined radius. T is the average intensity along the line perpendicular to the gradient of the iso-intensity curves. Thus, diffusion is turned on or off depending on the scale of the noise to be removed. For $a < t$ bright spots in a mainly dark area are removed more quickly. Dark spots in a bright area are blurred for $a \geq 0$. The behavior of this filter is illustrated in figure 12.5.

12.1.4 Erode&Dilate Filter

This linear filter is based on the simple erosion and dilation filter. An erosion filter (see figure 12.7 B) checks all neighboring pixels within the user-defined kernel radius

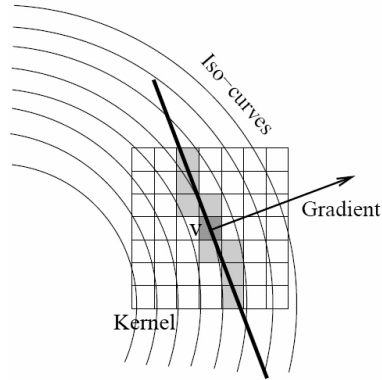


Figure 12.5: Iso-level sets through voxel v with corresponding gradient vector and kernel.



Figure 12.6: Left: Original test image. Right: Min/Max curvature flow filtered test image.

and sets the center value to the lowest intensity of all considered voxels. The dilate filter (see figure 12.7 C) performs the opposite function and sets the center value to the highest intensity in the kernel range.

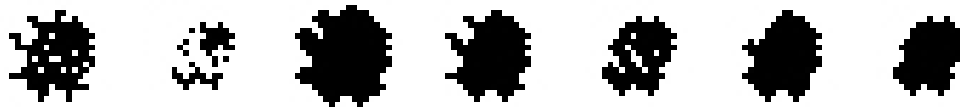


Figure 12.7: A) Original B) Single erosion. C) Single dilation. D) Dilation followed by erosion. E) Erosion followed by dilation. F) Opening filter followed by closing filter. G) Closing filter followed by opening filter.

In imaging terms, an erosion followed by a dilation is called an opening filter, whereas a dilation followed by an erosion is known as a closing filter. A closing filter fills gaps (figure 12.7 D), and an opening filter smoothens surfaces as shown in figure 12.7 E.

Combining these two filters leads to very efficient noise removal (figure 12.8). The user-defined kernel radius defines the maximum size of a noise particle which can be removed from the image. Furthermore, if the filter is reapplied iteratively on an image, intensity drifts can also efficiently removed. This is especially helpful for subsequent region competition segmentations. The user additionally defines the number of iterations and which filter (erode or dilate) to begin with.

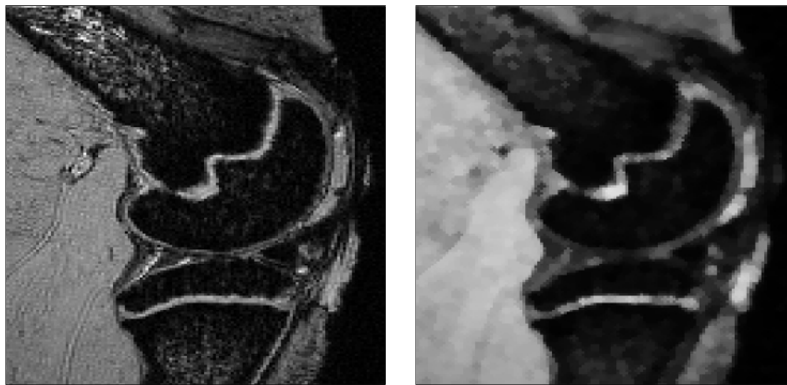


Figure 12.8: Left: Original test image. Right: Erode&Dilate filtered test image.

12.2 Edge Filter

The edge filters attempts to create an effective feature image prior to geodesic active contour segmentation. The appropriate filter wizard asks the user to choose one of three different edge detection approaches described below.

12.2.1 Gradient Magnitude Filter

Gradient filtering is probably the simplest and most rudimentary approach to find the edges of an object. In an image, the gradient can be approximated using a simple finite difference approach. A simple mask as shown in figure 12.9 is convolved with the image in all three dimensions.

$$G_x(i) = I_x(i + 1) - I_x(i - 1) \quad (12.5)$$

$$G_y(i) = I_y(i + 1) - I_y(i - 1) \quad (12.6)$$

$$G_z(i) = I_z(i + 1) - I_z(i - 1) \quad (12.7)$$

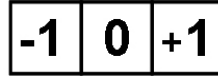


Figure 12.9: Gradient filter mask, used separately in x y and z direction.

The final approximation of the gradient magnitude G is given by

$$G(i) = \sqrt{G_x(i)^2 + G_y(i)^2 + G_z(i)^2} \quad (12.8)$$

where G_x , G_y and G_z are the convolution result for each voxel i of the image I in x, y and z direction.

To increase the advection force in the feature image, the gradient intensities are magnified with a non-linear intensity remapping function.

$$I_{out}(i) = \frac{1}{1 + \left(\frac{I_{in}(i) - I_{min}}{\kappa \cdot (I_{max} - I_{min})} \right)^p} \quad (12.9)$$

$I_{in}(i)$ is the intensity of voxel i in the gradient filtered image. I_{max} and I_{min} are the upper and lower extremes of the gradient image respectively. κ is the curve smoothing factor and p the exponent of the denominator, defining the steepness of the remapping curve. $I_{out}(i)$ is the remapped input intensity value. κ and p are defined by the user. The filter is applied to the example image as illustrated below (figure 12.10).



Figure 12.10: Left: Original test image. Right: Gradient magnitude filtered test image.

12.2.2 Zero Crossing Filter

As the name suggests, the zero crossing detector looks for zero crossings in the Laplacian-filtered image. The Laplacian filter is known as an edge-enhancement filter and is defined as the second special derivative of an image with intensities I .

$$L(x, y, z) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} + \frac{\partial^2 I}{\partial z^2} \quad (12.10)$$

The Laplacian L of an image highlights regions of rapid intensity change and is often used for edge detection.

Since an image is discrete, the Laplacian can be approximated by an image convolution with the following 3D kernel.

$$\begin{pmatrix} (-1,-1,-1) & (-1,-1,-1) & (-1,-1,-1) \\ (-1,-1,-1) & (-1,26,-1) & (-1,-1,-1) \\ (-1,-1,-1) & (-1,-1,-1) & (-1,-1,-1) \end{pmatrix}$$

Zero crossings always lie on closed contours, and therefore the output from a zero crossing detector is usually a binary image consisting of pixel-thickness lines corresponding to the position of the zero crossing points.

The zero crossing filter is very susceptible to noise and usually has to be heavily smoothed with a denoise filter. Figure 12.11 shows the filter behavior for a Gaussian smoothed test image.

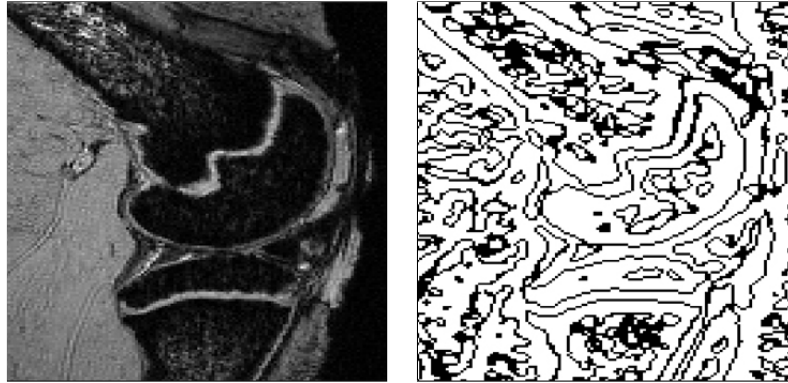


Figure 12.11: Left: Original test image. Right: Zero crossing filtered test image.

12.2.3 Canny Edge Detection Filter

The Canny operator was designed to be an optimal edge filter, meaning that all intensity edges in the image should be detected and marked at the exact location of their occurrence. Noise should be ignored and edges should form a gap-free surface.

There are four major steps used in the edge-detection filter:

1. **Denoise:** The raw image is smoothed to reduce noise. This step was removed from the Canny filter source code, as the input image is already pre-treated with one of the four denoise filters described previously (subchapter 12.1).
2. **Second derivative:** Edges are detected by calculating the second directional derivatives of the smoothed image. This is performed using the zero crossing filter previously introduced.
3. **Non-maximum Suppression:** The sign of the third derivative is evaluated to find the correct extrema. An exact mathematical derivation can be found in the original publication by J. Canny [4].
4. **Thresholding:** Finally, hysteresis thresholding is applied to the product of the gradient magnitude and zero crossings, to find and link edges.

A Canny filtered test image is depicted in figure 12.12.

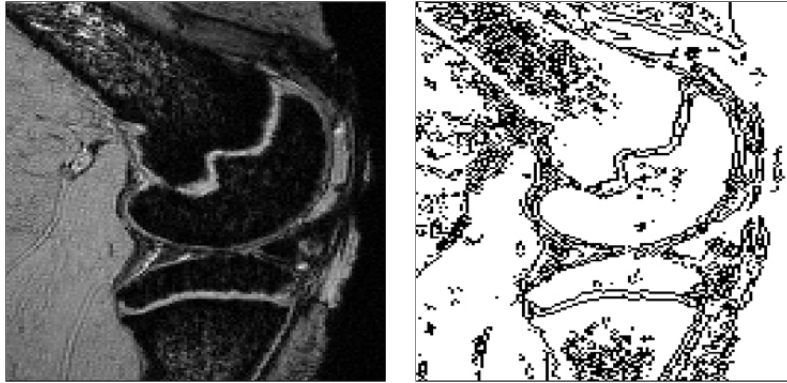


Figure 12.12: Left: Original test image. Right: Canny edge detection filtered test image.

12.3 AIM compatibility

Since AIM-format is the predominate file-format at this lab¹, the mySNAP's compatibility with this file-format was a key feature for this project.

mySNAP is now capable of loading AIM files of char or short word type. A wizard guides the user step-by-step through the different loading phases.

Once mySNAP has finished partitioning the image, the segmented data can be saved in various file formats including AIM files of char and short word type. Similar to loading files, a wizard guides the user through the saving steps. The voxel intensity in the AIM file corresponds to the assigned segmentation label number. Hence, if desired, multiple segmentation results can be saved in a single file.

Previously saved filtered images or segmentations can be loaded in advance, to speed up the segmentation process. This is especially useful if multiple registered images are to be segmented, as the seed points can be picked once and quickly loaded for use in all subsequent segmentations.

12.4 Logfile

To keep track of the filters and segmentations applied to an image, mySNAP provides an option to save all filter and segmentation parameters, elapsed time and all statistical data in a separate log file (txt-format).

If the segmentation is saved as an AIM-file, the log file content is automatically written to the AIM internal log file.

¹Bone Imaging Laboratory, University of Calgary, Canada

Chapter 13

Accuracy and precision

Segmentation is probably the most crucial component in the generation of an accurate three dimensional knee joint model. For the data acquisition and registration, the image content is mostly unchanged.

The accuracy of a segmentation is limited by the image voxel-size. A voxel may only belong, or not belong, to a segmented region. Unlike registration image interpolation, there are no mid-ranging gray-levels. Furthermore, segmentation is strongly dependent on image quality. Contrast and image distortions are of major concern, as they are key to successful partitioning of an image.

This chapter illustrates the efficiency of different filtering and segmentation methods, dependent on the image quality.

13.1 Method

As for the previous two accuracy and precision evaluations, segmentation results must be compared to a ground truth value. Therefore, a 3D image including a well-defined object has to be created. A volumetric "E" with decreasing thicknesses for each horizontal bar, was found to be most appropriate for modeling tissues of different shapes and sizes.

A white "E" on a black background is not a very realistic model of a CT- or MRI-acquired image. Thus, the image quality needs to be reduced in a controlled manner.

There are two coefficients which mathematically describe the image quality.

These are,

Signal-to-noise ratio (SNR)

$$SNR = \frac{\bar{I}_o}{\sqrt{\frac{\sum_{i=1}^N (I_b(i) - \bar{I}_b)^2}{N}}} \quad (13.1)$$

Contrast-to-noise ratio (CNR)

$$CNR = \frac{\bar{I}_o - \bar{I}_b}{\sqrt{\frac{\sum_{j=1}^M (I_o(j) - \bar{I}_o)^2}{M} + \frac{\sum_{i=1}^N (I_b(i) - \bar{I}_b)^2}{N}}} \quad (13.2)$$

$I_o(j)$ and $I_b(i)$ are the intensities of voxel j in the object and voxel i in the background, respectively. Similarly $\bar{I}_o$ and $\bar{I}_b$ are the mean intensities of all voxels M in the object and all voxels N in the background.

SNR is defined as the mean intensity of the object divided by the standard deviation of the background noise. CNR is defined as the image contrast (i.e. the difference in object and background intensities) divided by the standard deviation of the noise in the whole image.

Figure 13.1 shows a single slice through the segmentation test image with decreasing SNR and CNR values.

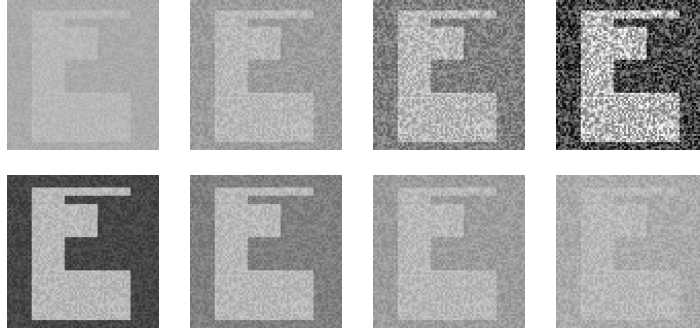


Figure 13.1: Top: Decreasing SNR (4,2,1,0.5 starting left) and constant CNR (2). Bottom: Decreasing CNR (8,4,2,1 starting left) and constant SNR (2).

In the accuracy and precision experiments, SNR was kept constant at a realistic noise level and CNR was varied between 0.5 and 4. These contrast and noise values are similar to the image quality observed in MR and CT datasets.

To get an overall view of the segmentation accuracy, the test image with a SNR and CNR of 2, was segmented with all 24 filter combinations and segmentation

methods available in mySNAP.

In a second step, the 6 best segmentation combinations were tested for increased (4) and decreased (1 and 0.5) *CNR* values.

Some segmentations were repeated multiple times to evaluate the precision of the mySNAP partitioning program.

Other test sequences evaluated the influence of the number of seed points used in the segmentation. Size, position and number of initial points were varied.

Finally the segmentation parameters were tuned, optimized, and tested separately and in combination. All these measurements were performed on a Gauss-Gradient filtered image.

13.2 Result

The enormous amount of data evaluated throughout the project will not be fully presented in this subchapter. The graphs below (figures 13.2 to 13.6) illustrate the general findings of this measurement series. The corresponding numerical results are listed in tables F.1 to F.4. A complete list of all the segmentation accuracy and precision results can be found on the enclosed DVD.

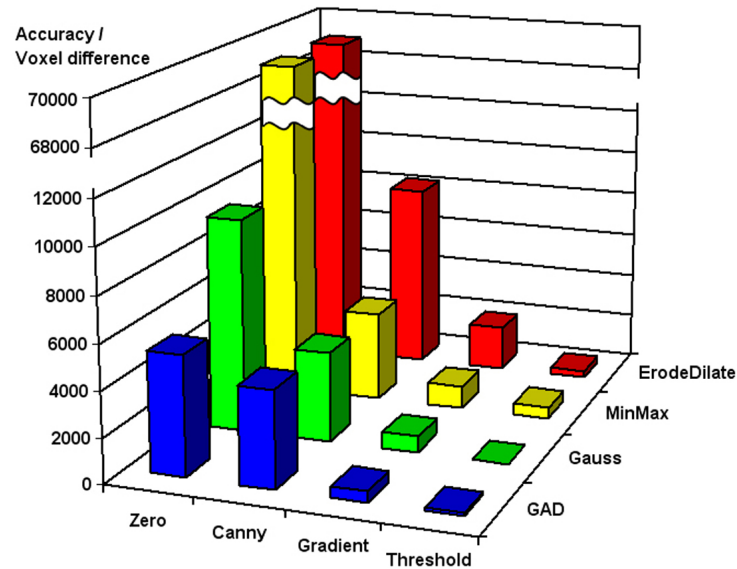


Figure 13.2: Over-all accuracy comparison for all the possible filter combinations.

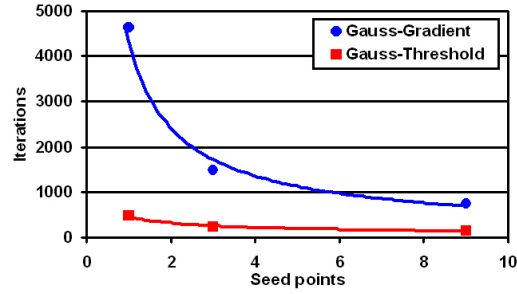


Figure 13.3: Exponential dependency of the segmentation processing time on the number of seed points.

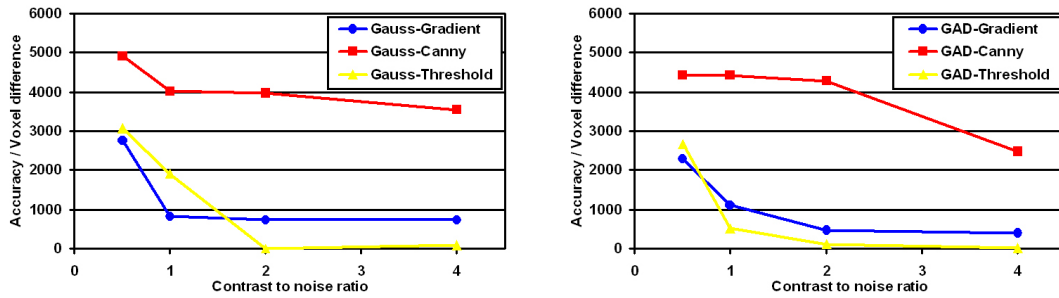


Figure 13.4: Change in accuracy for the 3 best Gaussian- (left) respectively GAD- (right) filtered segmentations, with increasing CNR values.

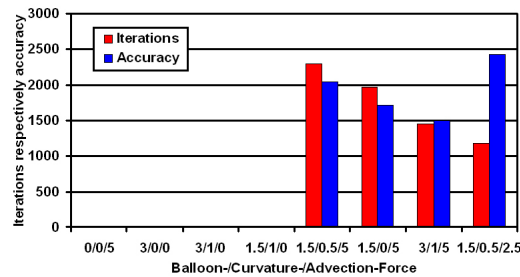


Figure 13.5: Accuracy and speed properties for various segmentation parameter settings.

13.3 Discussion

In the overall segmentation accuracy test shown in figure 13.2, large variation of the segmentation accuracy between different filter combinations can be seen. The threshold filter shows excellent results regardless of the denoise filter chosen. The highest accuracy could be achieved with the Gauss and Gradient Anisotropic Diffusion (GAD) filter, with a volumetric error of only 0.0028% and 0.1766% respectively.

As expected, segmentation of corners and edges present the greatest challenge for the program. Provided the gradient edges are solid and continuous, the segmentation "bubble" will remain inside the boundaries and no leakage will occur. In these cases, the repeatability of the segmentation is 100% independent of the number, size and position of the seed points.

The segmentation processing speed depends eminently on the number of initial seed points (see figure 13.3). This is especially true for edge-based segmentation, where an exponential decrease in the number of iterations is observable, if the number of seed points is increased from 1 to 9. For 9 seed points the processing time can be reduced by a factor of 5. The acceleration does, however, greatly depend on the position of the initial points. The snake bubble growth is much slower in thin and complex structures than in simple large regions. Therefore, it is recommended to place more seeds in slow performing areas than in areas with fast bubble growth.

It is also crucial, for the edge-based segmentation that all initial bubbles lie within the edge boundaries. Otherwise, the snake will start growing outside of the segmentation region.

It was generally observed that the region snake performed faster than the edge snake. The average voxel expansion of the region snake in the study was 478 voxels/iteration, whereas the edge snake only grew by 56 voxels/iteration.

In figure 13.4, the best six filter combinations were tested on their segmentation consistency for decreasing contrast to noise ratio. In most cases, the GAD-Threshold filter combination provided the best segmentation results. However, the GAD-Gradient filter performed better for very low contrasts (SNR 2, CNR 0.5).

The Canny filter tended to underestimate the true volume by approximately 4%, although, for thin curved surfaces (e.g. cartilage), the performance may improve.

It should be noted that none of the experiments included intensity drifts or local distortions, as are often observed in MR and CT images. Such artifacts may significantly reduce the dominant position of the threshold filter, as the region snake relies on a uniform image intensity in the segmentation region.

For the first four segmentation parameter optimization tests, either the snake did not expand (figure 13.6 A) or the segmentation leaked (figure 13.6 B). A seg-

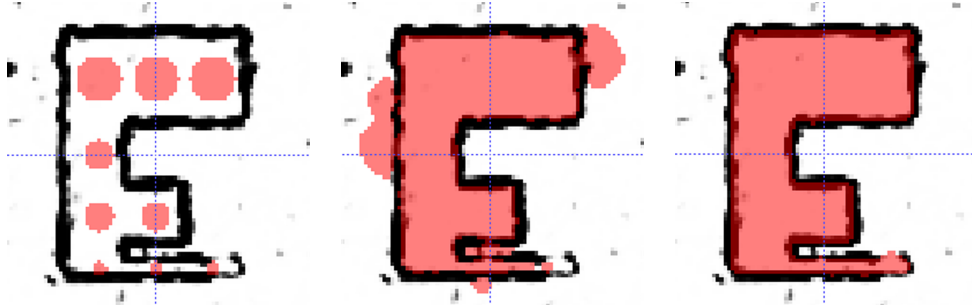


Figure 13.6: A) No expansion, wrong parameter settings. B) Leakage, wrong parameter settings. C) Successful registration, correct parameter settings.

mentation without a balloon-force did not expand, and contrary to this, examples with a too high balloon-force leaked. To prevent the segmentation from leaking, the advection-force must be greater than zero. However, the advection-force slows the segmentation process. By combining the three forces appropriately, an optimal accuracy can be found within a reasonable processing time (see figure 13.5 second bar-pair from the right, and figure 13.6 C).

Chapter 14

Sample Segmentation

With all information collected in the segmentation accuracy and precision experiments, tissue from aligned images could separately be extracted.

14.1 Method

The registered (contrast-enhanced) CT and MR images have been used for segmentation. Of over 40 components in a rat knee joint, 23 were selected and separately segmented with mySNAP. For each partition, the most suitable filter combination and parameter settings were evaluated and applied. The final result was undergoing a visual control by overlaying the semitransparent segmentation on the original image. If necessary, segmentation was re-performed or manually corrected.

All segmentations were saved in a separate AIM file (char word-type) including a log file of all information on the filter types and parameter settings used.

14.2 Results

Only two examples of the 23 individual segmentations are presented here. These are representative of all region and edge snake segmentations.

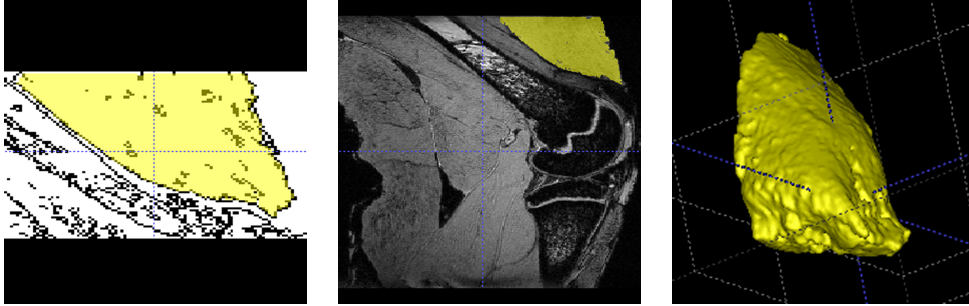


Figure 14.1: Partitioned "Tensor Fascia Lata" (yellow) from MRI dataset. Left: During the segmentation. Middle: After segmentation, overlaying the original image. Right: 3D visualization.

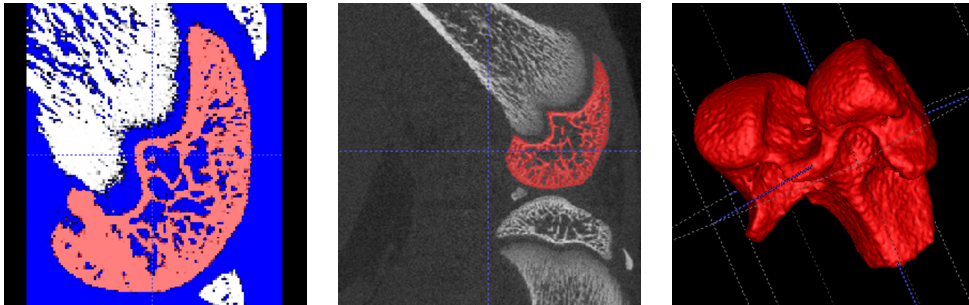


Figure 14.2: Partitioned femur growth plate (red) from CT dataset. Left: During the segmentation. Middle: After segmentation, overlaying the original image. Right: 3D visualization.

All segmentations were processed in 10 to 20 minutes.

14.3 Discussion

In general, both soft-tissue and bone structures were segmented well with the mySNAP program.

In the case of CT images, the high contrast and uniform intensity distribution of bone made these images appropriate for the fast and highly accurate region snake method (see figure 14.2).

Strong intensity drifts occurred in the contrast-enhanced MR images, especially in the muscles. The threshold method failed to set proper boundary conditions for the segmentation. With the gradient and especially the Canny filter, the darkened tissue borders were accurately detected. Hence, most soft-tissue was extracted with an edge-based segmentation method (see figure 14.1).

Cartilage could not be accurately partitioned in the MR images. Due to the rather large voxel size of 70x70x70 μ m, only 6 voxels fit into the average cartilage diameter of 400 μ m. One voxel more or less in diameter incurs an error of $\pm 16.7\%$. For future FE knee joint kinetics analysis, an inaccuracy of this magnitude is not tolerable. Especially concerning the fact that cartilage plays a major role in force dissipation in knee joints.

Because of this, the cartilage was extracted from the contrast-enhanced CT image, previously described in chapter 5.4, and the error was reduced to reasonable 5%.

