FACHHOCHSCHULE
WIENER NEUSTADT
University of Applied Sciences – Austria

Development and evaluation of a bone anatomy segmentation guide for micro-CT in mice

Master's Thesis

Submitted by:	**Stefan Rohrbacher, BSc**
Matriculation No.:	**01607307**
at	**Master's Program MedTech Functional Imaging, Conventional and Ion Radiotherapy**
Supervisor:	**Dipl.-Ing. Michael Rauter**

Wiener Neustadt, 2nd December 2024

FACHHOCHSCHULE
WIENER NEUSTADT
University of Applied Sciences – Austria

DECLARATION OF INTEGRITY

I hereby affirm that this research work was written solely by me and that I have not previously submitted this work at another educational institution for the purpose of receiving an academic degree. In particular, contributions by other persons in this work have been appropriately cited and the data gathered through the methods described have been accurately reproduced.

Ich erkläre an Eides statt, dass diese Arbeit ausschließlich von mir selbst verfasst wurde und ich diese Arbeit nicht zuvor an einer anderen Bildungseinrichtung zum Zwecke der Erlangung eines akademischen Grades vorgelegt habe. Insbesondere wurden Beiträge anderer Personen entsprechend kenntlich gemacht sowie die in dieser Arbeit verwendeten Daten entsprechend der dargestellten Verfahren gewonnen und richtig wiedergegeben.

Wiener Neustadt,	2nd December 2024	
	Date	Signature

Acknowledgements

First and foremost, I would like to thank my supervisor Dipl.-Ing. Michael Rauter for his invaluable feedback. Furthermore, I wish to thank my partner for her unconditional support and patience as well as my parents for their never ending support for all my ventures in live. Finally, I would like to thank David Kovacs, MA as he did not hesitate to help a complete stranger accomplish writing a thesis in LaTeX.

Abstract

Anatomical segmentation is a common task in microscopic computed tomography (micro-CT) studies. Segmentation quality and accuracy strongly depends on the tools chosen and operator experience. Therefore, it is important to educate newcomers to the field in the basics of micro-CT segmentation. Comparing segmentation performance of different operators before and after offering them a guide to reference with the goal of answering the research question, whether the usage of a segmentation guide improves segmentation quality when used by inexperienced operators. First, a ground truth segmentation was created by the author. Secondly, a guide explaining micro-CT segmentation with *3D Slicer* has been created. Third, testers were tasked with segmenting three bones on a mouse micro-CT scan. Then they were tasked with reading the guide and repeat their segmentation on the contralateral anatomical structures. The resulting segmentations were compared to the ground truth using the Hausdorff distance (HD) and Dice similarity coefficient (DSC). Seven testers did six segmentations each. 85.71% of users improved their segmentations after reading the guide and 81% of segmentations displayed better agreement with the ground truth. Due to the small sample size, the research question cannot be answered definitively. However, the results imply significant improvements if the study were to be repeated at a larger scale.

Keywords

micro-CT, segmentation, guide, 3D Slicer

Contents

List of Abbreviations

micro-CT	Microscopic computed tomography
AC	attenuation coefficient
AI	artificial intelligence
API	Application Programming Interface
bit	binary digit with only two possible values 0 and 1
BPF	Berkeley Packet Filter
CCD	charge-coupled device
CMOS	complementary metal–oxide–semiconductor
CNN	convolutional neural network
CPU	central processing unit
CSV	Comma-separated value
DCNN	deep convolutional neural network
DICOM	Digital Imaging and Communications in Medicine
DOF	depth of field
DSC	Dice similarity coefficient
DVH	dose volume histogram
FBP	filtered back-projection
FCN	fully convolutional network
FDK	Feldkamp-Davis-Kress algorithm
FN	False Negative
FP	False Positive
GB	Gigabyte
GHz	Gigahertz
GPU	graphics processing unit
HD	Hausdorff distance
HU	Houndsfield units
Hz	Hertz
isotropic	uniform in all directions
ISRA	image space reconstruction algorithm
ISRA-TV	total variation regularization ISRA
kVp	Kilovoltage peak
LUT	lookup table

mAs	Milliampere-seconds
MRB	Medical Reality Modeling Language bundle
MRI	magnetic resonance imaging
OAR	organs at risk
OS	operating system
pixel	the smallest addressable element in a raster image
PTV	planning target volume
RAM	Random Access Memory
ROI	region of interest
SVM	support vector machine
TN	True Negative
TP	True Positive
TV	total variation regularization
voxel	three-dimensional counterpart to a pixel

1. Introduction

This chapter is intended to provide an introduction to the thesis. In Section 1.1 the general motivation is discussed. The section is followed by outlining the research goals in Section 1.2. Section 1.3 gives an insight into the research methods used in order to answer the raised research question. Finally, Section 1.4 gives an outlook of the structure of the remaining thesis.

1.1 Motivation

Microscopic computed tomography (micro-CT) is a commonly used modality in industry and research. For research, it enables the study of small animal morphology [1]. A common task when conducting short-term or long-term studies with mice is the segmentation of the image datasets, as it plays an important role for quantitative image analysis [2]. Segmentation may be defined as the process of separating various image components. Furthermore, extracting parts relevant for subsequent analysis and comparison [1]. A common task may be in-vivo bone analysis for the investigation and monitoring of the progression of various bone pathologies. The first step of analysis is performed by segmenting the micro-CT volume. Secondly, morphological features relevant to the study can be extracted and then finally be analyzed. Streamlining the segmentation process is therefore crucial for the analysis step [1], [3].
For performing micro-CT dataset segmentation tasks, there exist a few capable free and open-source programs. Two actively maintained and well-known software packages are [4], [5]:

- ITK-SNAP [6] available under the `GNU General Public License` [7]
- 3D Slicer [8] available under a `BSD style license` [9]

Both programs equip the user with a sizeable number of segmentation tools, which can be grouped into manual, semi-automatic and fully automatic [1]. The accuracy and efficiency of the segmentation, however, strongly correlate with user experience and the segmentation algorithm chosen [5], [10], [11]. 3D Slicer's documentation website explains the software and many of the available tools in the standard distribution of 3D Slicer [12]. But it does not feature a recommended segmentation workflow. Despite this fact, there is a lack of general recommendations, instructions or guides on how to efficiently segment Microscopic computed tomography datasets.

1.2 Research Goals

Segmenting large datasets, such as those produced by a high-resolution micro-CT scan, results in large Random Access Memory (RAM) demands by the segmentation software. The 3D Slicer documentation states that the software will require approximately ten times more memory than the loaded dataset. The user therefore has to optimize the segmentation workflow if the software is to be used on a computer with limited amounts of RAM [13], [14]. Therefore, the author hypothesizes that users of 3D Slicer will benefit from a streamlined segmentation guide, and it will improve segmentation quality as well as workflow efficiency. As mentioned above, `3D Slicer` and `ITK-SNAP` provide the user with a sizeable amount of segmentation tools. The selection of the correct segmentation algorithm and workflow have a non-negligible influence on the segmentation quality as well as the time the user has to spend on it [15].
This thesis will explore potential segmentation workflows, as well as the creation of a guide for performing the mentioned segmentations. Additionally, the following research question will be answered:

> "Does the usage of a segmentation guide improve segmentation quality when used by inexperienced operators?"

1.3 Research Methodology

The practical work for this project will be completed by the author on his private computer, for technical information, see Table 1.

OS	NixOS 24.05
Kernel	Linux 6.6
Display	24 inch, 1920×1080 @ 60 Hz
CPU	AMD Ryzen 7 3700X 16 cores @ 3.6 GHz
GPU	AMD Radeon RX 580
Memory	64 GB
Harddrive	Samsung SSD 860 EVO 500 GB

Tab. 1: Segmentation computer specifications

The first step was the segmentation of the micro-CT scans, which served as a ground truth for finding possible efficient workflows with the tools included in `3D Slicer`. Then a guide has been created outlining two different possible workflows for bone segmentation. This guide has been handed out to test candidates in conjunction with the task of using the guide to segment a specific part of a scan volume. The candidates were asked to perform two segmentations and return their results to the author. The first segmentation they were asked to perform prior to reading the guide. The second segmentation they were asked to perform

after reading the guide. After receiving the results, the author calculated their accuracy by comparing the results of each test candidate with his own segmentation. Dice similarity coefficient (DSC) and Hausdorff distance (HD) were chosen as a similarity scores for comparison of segmentation quality between the guided segmentation versus the unguided one [16], [17]. This has been done using the `SlicerRT` extension for the 3D Slicer software package [18].

1.4 Structure of this Thesis

The next chapter is the frame (Chapter 2), where the necessary theoretical information will be explained.
In Chapter 3, the origin of the datasets and the evaluation of the segmentation will be described.
Afterwards, in Chapter 4, results of the segmentations will be listed.
Finally, Chapter 5 will deal with the results of this work, as well as the limitations and further subjects of research.

2. Background

This section is devoted to providing the theoretical background that was necessary for the study. Chapter 2 is structured as follows:
First and foremost, an introduction to the operating principle of a micro-CT will be provided in Section 2.1.
Secondly, Section 2.2 will give an overview of the composition of a guide.
Thirdly, an evaluation of the segmentation techniques and algorithms is given in Section 2.3.
Fourth, an overview of possible digital representation methods of segmentations is given in Section 2.4.
Fifth, possible segmentation evaluation methods are described in Section 2.5.
Finally, Section 2.6 will provide a brief overview of the relevant anatomical structures of mice.

2.1 micro-CT

Microscopic computed tomography (micro-CT) as a technology has been around since the 1980s and has since emerged as the preclinical analogue of clinical CT [19], [20]. In comparison with clinical CT, micro-CT features high spatial resolution, which makes it irreplaceable for non-destructive small animal studies [19]–[21].

2.1.1 Operating principle

The fundamental principle underlying clinical CT is that both the X-ray source and the detector rotate around the patient table as part of a gantry assembly simultaneously [22]. A visual representation of this concept can be seen in Figure 1.

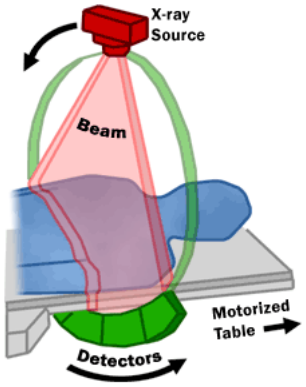


Fig. 1: Clinical CT basic setup [23]

As illustrated in Figure 1, the animal or inanimate object is positioned on a table between the X-ray source and the detector. In clinical CT, the X-ray tube is designed to produce a fan-shaped beam, which passes through the patient onto a concave detector array. In contrast, in most micro-CT the X-ray source produces a cone- or pyramid-shaped-beam, which passes through the patient onto a two-dimensional or flat detector [19], [24]. For reference, see Figure 2.

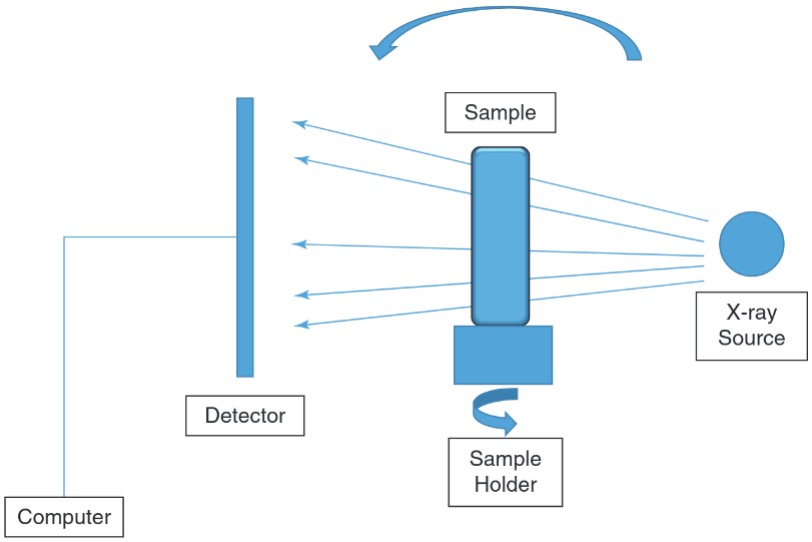


Fig. 2: micro-CT basic setup[25]

Micro-CT X-ray sources typically utilize a fixed tungsten anode and therefore are required to operate at lower voltage and current in comparison to the rotating anode employed in clinical CT. The achievable tube voltage is limited to a range between 20 and 100kVp and a current range spanning from 0.05 to 1mA.

$$I \propto (kV)^2 * (mA) \tag{1}$$

The number of X-ray photons depends on the tube current, and the energy of the photons depends on the tube voltage squared. Moreover, the intensity of the beam can be expressed as a function of the product of these two factors, see: Equation **(1)**. A flat panel detector typically contains a scintillator screen, which, when excited by X-rays, emits visible light [20], [24], [26]. The light is then detected with an image sensor, which can be either a charge-coupled device (CCD) or a complementary metal–oxide–semiconductor (CMOS) image sensor [24]. Micro-CT employs high-resolution flat panel detectors with pixel sizes of less than 100 μm^2 [19], [26]. They can, however, achieve isotropic voxel sizes in a range between 400 nm and 70 μm [25]. To produce a scan, the micro-CT captures several hundred, two-dimensional cone beam projections of different angles around the animal or object [19], [20], [26]. The rotation angles can be in the range of 0 to 180° or 0 to 360° [25]. Image information can

be described as the amount of attenuation of the X-ray beam at a specific path through the volume [25]. This attenuation is described by the following Equation **(2)**:

$$I_x = I_0 e^{-\mu x} \tag{2}$$

Where I_0 is the initial intensity of the beam, x is the distance the beam traveled, and μ is the linear attenuation coefficient. The linear attenuation coefficient (AC) is different for every tissue and material and describes its X-ray density. The result of Equation **(2)** is I_x, the intensity of the beam at a distance x from the source [25], [27]. This is performed simultaneously for all detector pixels in a single projection. If the process is repeated for multiple angles in small steps (see: Figure 3) the attenuation value for every voxel in the scan volume can be found. Generally speaking, increasing the amount of projections leads to a more accurate result [25], [27].

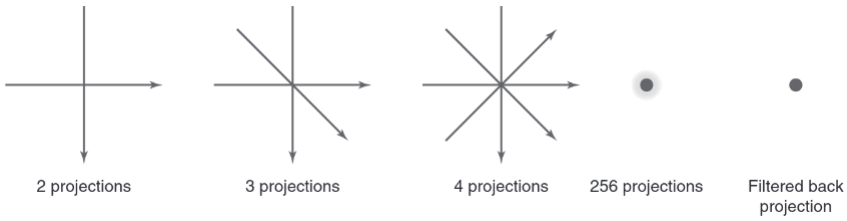


Fig. 3: micro-CT projections and reconstruction [25]

2.1.2 Data processing

Once all needed projections, as shown in Figure 3, are performed and stored on a computer, they can be reconstructed into the final image. Or in the case of a volume dataset, multiple images [25], [27]. A single projection can be described as a radiograph, which represents the intensity of the object at a particular angle [25], [27]. The radiographs differentiate either 65,536 (16-bit) gray values or 256 (8-bit) gray values, which are directly related to the X-ray absorption [25], [27]. To convert these two-dimensional radiographs into a three-dimensional dataset, the application of a reconstruction algorithm is required. Before any reconstruction algorithm can be utilized, however, the data needs to be preprocessed. This step compensates for factors such as detector response heterogeneity, defective pixels, X-ray intensity drift and electronic noise [27]. Next, the reconstruction algorithm can be applied. As mentioned above, there exist a multitude of reconstructed algorithms, however, the most widely used algorithm in micro-CT is the filtered back-projection (FBP). Specifically, the Feldkamp-Davis-Kress algorithm (FDK), which is well suited for micro-CT because of its simplicity and ability to handle data truncation in the z-axis or longitudinal direction [25], [27]. Building upon the forward- and back-projection of these simple algorithms are iterative algorithms

like the ISRA and ISRA-TV, which were utilized to reconstruct the datasets used in this study (see: Table 2 on page 33). As can be seen in Figure 4, this algorithm starts off by performing the scan and measuring the absorption profile images to obtain the measured data. Next, the current reconstruction image is forward and back-projected. The resulting estimation image then serves to calculate the error ratio between the estimation and the measured data. The error is then considered in the next iteration of forward and back-projection to update the estimated image. In the case of the ISRA-TV, which is used to improve reconstruction results for datasets with a small number of projections, total variation regularization (TV) enforces a sparse image gradient. Effectively, this modification of the ISRA algorithm enforces uniformity between edges [28]. Figure 4 shows TV being used to enhance the ISRA method by calculating the TV on the previous back-projected image and modifying the error ratio for the next iteration cycle [28].

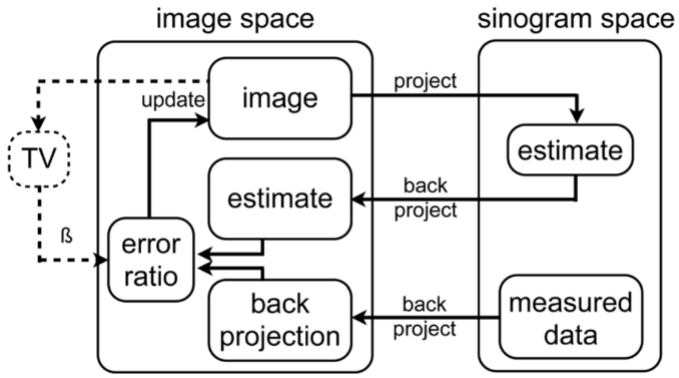


Fig. 4: ISRA and ISRA-TV iterative algorithm principle [28]

Figure 5 shows a comparison of reconstruction results between FDK, ISRA and ISRA-TV in relation to the number of projection views. When comparing the algorithm results with the lowest number (32) of projection views, it can be clearly seen that FDK suffers the most from streak artifacts and image noise. ISRA is able to correct for most streak artifacts, but still struggles with image noise. Finally, ISRA-TV is able to correct for streak artifacts and image noise [28].

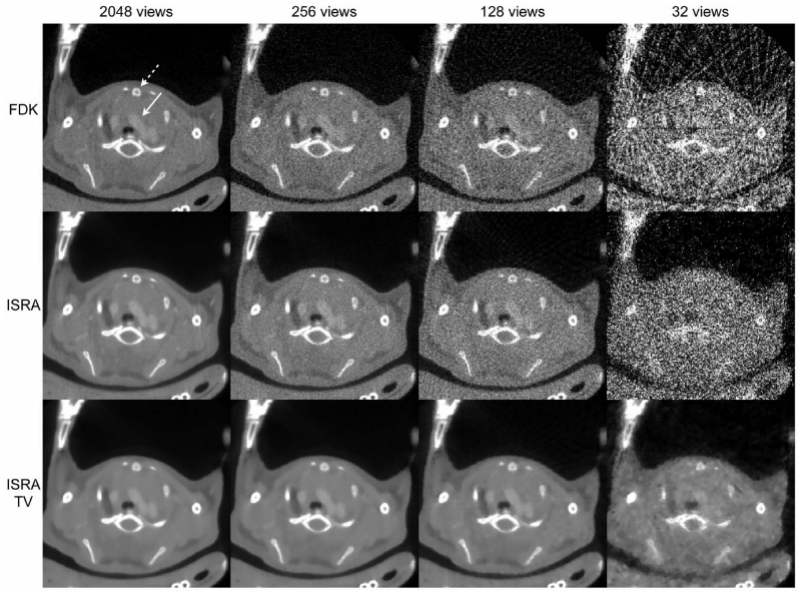


Fig. 5: FDK, ISRA and ISRA-TV reconstruction quality comparison [28]

2.1.3 Houndsfield units (HU)

Houndsfield units (HU) are directly related to the attenuation coefficient (AC) μ and can also be called the CT number.

$$HU = \frac{\mu - \mu_{water}}{\mu_{water}} * 1000 \tag{3}$$

Equation **(3)** shows how HU are computed. Where μ is the attenuation coefficient of a specific tissue or material and μ_{water} is the attenuation coefficient of water. It is calculated as the difference of the attenuation coefficient of a tissue and water, divided by the AC of water and finally multiplied by the factor 1000 [27]. Effectively, all HU values are calculated as a comparison to the attenuation coefficient of water. This results in some notable HU values on the Houndsfield scale. The most notable one being water, which is zero by definition of Equation **(3)**. Air, which has an attenuation coefficient close to zero, has a HU value of -1000. Bone tissue, which has about twice the attenuation coefficient of water, has a CT number of $+1000$. While most soft tissues have a HU value between -100 and $+100$ [27]. Figure 6 gives an overview of the most common clinical HU values.

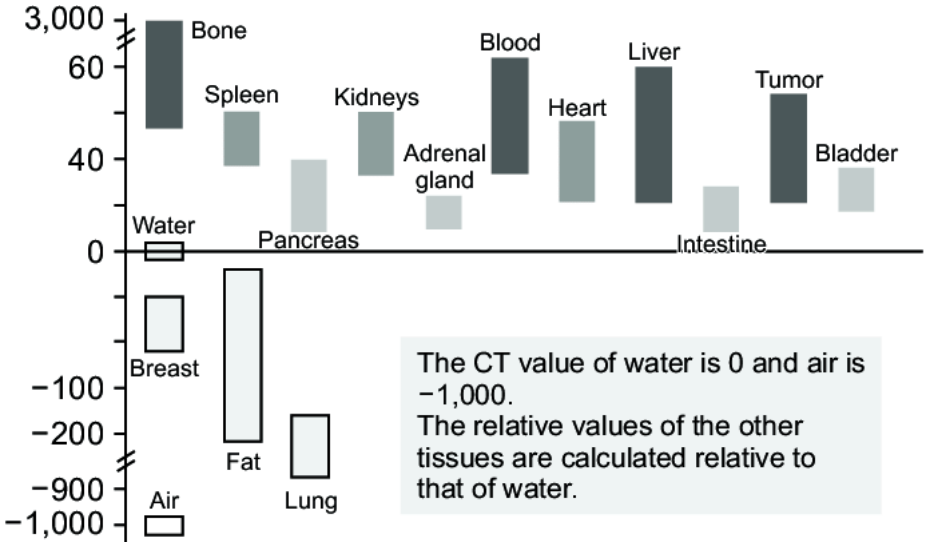


Fig. 6: HU for tissues in clinical setting [29]

2.2 Composition of a Guide

This section is intended to give a rudimentary overview of two commonly used frameworks in software documentation.

2.2.1 The Good Docs Project

> "The Good Docs Project educates and empowers people to create high-quality documentation by providing them with templates, resources, best practices, and tools to enhance their documentation in open source and beyond." ([30])

The Good Docs Project is used for documentation design by companies like GitLab (GitLab Inc., San Francisco, USA) and JetBrains (JetBrains s.r.o, Prague, Czech Republic). It actively discourages producing any sort of documentation before its content and structure have been extensively planned [30]. Instead, it instructs documentation authors to, first and foremost, think about the topic at hand. Especially, authors should evaluate the complexity of the topic and try to predict what kind of questions readers might have while reading the documentation. Furthermore, documentation authors are supposed to consider the ability of their target audience and what reasonable assumptions one can make about the prior knowledge of readers. Next, The Good Docs Project instructs documentation authors to create three fictional readers: a beginner, an intermediate and an expert. The beginner might represent an ordinary customer or user of some software or technology product. The intermediate might be a system administrator of said product and the expert could be a support person. The documentation author should now, come up with several problems each fictional reader would try to solve with the help of documentation [30]. Furthermore, the author is supposed to compose a table and score the likeliness each reader is going to use the documentation to

complete the task. The score ranges from 1, which is the lowest likelihood, to 3, which is the highest likelihood. Next, the author needs to sum up each row. The resulting *Critical Path Score* is then intended to give the author insight about the importance of different documentation content [30]. *Critical Path Score* ranges from 3[1], which is the lowest importance to the highest importance of 9[2]. Finally, based on all the information gathered in the process explained above, the author is instructed to choose the appropriate template provided by The Good Docs Project [30].
The following compressed list provides an overview of the available templates in the current edition of The Good Docs Project version 1.2.0:

Concept An explanation of a concept, context, or background information about a product or its features.

How-to A concise set of numbered steps to do one task with the product.

README Information users need to know about your project, including how users can engage with the project and get started with the tool.

Reference Specific, in-depth details about a particular topic.

Release notes Communicate new features, improvements, bug fixes, and known issues about a product to users and stakeholders.

Tutorial Instructions for setting up an example project using the product, intended for the purpose of hands-on learning.

Troubleshooting A list of common problems (referred to as "symptoms") experienced by users, an explanation of the causes, and steps to resolve the issue.

Bug report The bug report is an issue template that your users can fill out to provide you with higher-quality, actionable bug issues for your product.

Code of Conduct A Code of Conduct helps you govern your open source or developer community to ensure it remains healthy and open.

Contributing guide A CONTRIBUTING document tells users how they can contribute to your open source project and join the community.

Our team Helps you clearly communicate who belongs to your open source project or organization and how contributors can contact or work with them.

[1] 3 readers, each with a likelihood of 1. $3 * 1 = 3$
[2] 3 readers, each with a likelihood of 3. $3 * 3 = 9$

API quickstart API quickstarts describe the easiest way for readers to achieve a result that shows off the capabilities of the service.

API reference API references are technical manuals that provide API specifications and integration instructions to help your audience understand the service.

Contact support A contact support page typically includes a list of the communication channels, discussion forums, and links to other resources to assist users with issues that they are having with your product.

Glossary A reference document that lists and organizes terms and their definitions that are unique to your organization or which you use in a specific way.

Installation guide Explain all the necessary steps to install the product and set it up for further use.

Quickstart A quickstart introduces your users to your application for the first time. It focuses on the primary feature of the application and helps your users to start using the application as quickly as possible.

2.2.2 Diátaxis

The *Diátaxis documentation framework* by Daniele Procida [31], [32] is extensively used for technical documentation by software projects like NumPy [33], Django [34] and large companies like Canonical (Canonical Limited, Douglas, Isle of Man) and Cloudflare (Cloudflare, Inc., San Francisco, California, U.S.A.). Diátaxis bases itself on four basic concepts to divide documentation into four different types with different structures according to their goals.

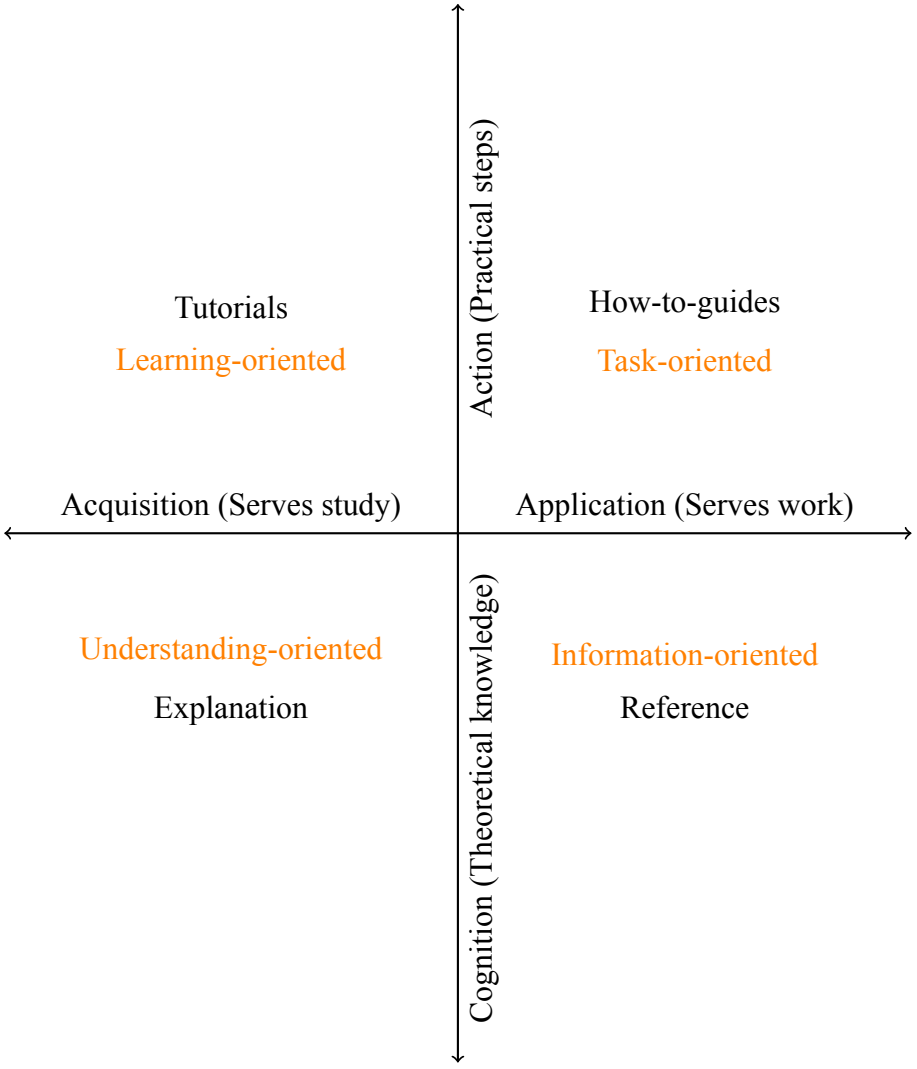


Fig. 7: Diátaxis Foundations [31]

As can be seen in Figure 7, the documentation author first ought to decide which scale their documentation applies to. The target audience may be interested in practical or theoretical knowledge, the author traces the y-axis of Figure 7 in a positive or negative direction. Procida then instructs documentation authors to decide if their target audience is interested in documentation that either serves further study or leads to directly applicable practical knowledge. Documentation writers should therefore trace the x-axis of Figure 7 in a positive or negative direction, finally arriving at the type of documentation they should compose for the task at hand [35]. Procida explains the purpose of his four types of documentation as follows:

Tutorial Teaches the user in basic competence in a skill by having them follow practical steps.

How-to Guide Helps users apply their skills to solve actual problems by showing steps.

Explanation Helps users understand theoretical concepts about their skill.

Reference Gives the user information about a skill without outlining the steps they need to take.

Additionally, Procida mentions that these types do not pose hard limits on what documentation can contain. They are rather meant to remind the author about the needs of users [31]. According to these criteria the guides practical section is based on Procida's structure for How-to guides, and its theoretical section is based on his structure for Explanation style documentation [31].

Tutorial

A tutorial should be interpreted as a practical activity according to Procida [31]. It aids the user in acquiring skills and knowledge by performing a meaningful and achievable task. Tutorials should therefore be seen as lessons. Furthermore, what a student does will not directly relate to what they will learn. Instead, they are meant to learn theoretical knowledge, for instance: how things relate and interact and by extension also how to interact with them [31], [35].

How-to guide

Procida implores that How-to guides need to have practical utility and a clear objective. They can only serve an already competent user. Thus, on page 73 the guide has explanations about the basic usage of 3D Slicer and its tools before instructing the user to follow a segmentation method themselves. Procida requires How-to guides to feature imperative language that does not allow for digression or explanation [31], [35].

Explanation

Explanation documentation, as stated by Procida, on the other hand, is defined as offering context to users. It is supposed to establish connections between concepts and deepen the user's theoretical understanding of the skill. While also offering the user choices and reasons to do something a particular way, or even explain why something is not recommended [31], [35].

Reference

Reference documentation material, as defined by Procida, contains theoretical knowledge that a user looks to in their work. In the case of software, like 3D Slicer, reference guide describes the software itself. This may include tools, buttons, default settings, recommended setting or APIs. Procida describes reference material as being like a map. Users get to look up information about a territory without having to explore it themselves. Although, reference material should not imply how to perform tasks, it should however show how to use something correctly [31], [35].

2.2.3 Pictograms

The guide, which can be seen on page 73, makes extensive use of pictograms to inform the reader about common errors, pitfalls to avoid, warning about resource-intensive operations, and, whether a tool is only available via an add-on. Inspiration for this kind of visual communication was taken from books published by O'Reilly (O'Reilly Media, Inc., Sebastopol, California, U.S.A.), specifically *Linux observability with BPF: advanced programming for performance analysis and networking* by David Calavera and Lorenzo Fontana [36]. Pictograms are stylized drawings that are supposed to represent familiar objects, ideas or abstract concepts to the reader [37]. In health information documents, they have been proven to enhance the readability, attractiveness and visual attention of the target audience [38], [39].

2.3 Segmentation of images and volumes

Segmentation of images is a process of delineating structures of interest. It can also be called contouring or annotation and is typically used to delineate anatomical structures, tumors, malignancies and various other parts of images. As it is a requirement for visualization of target structures, like a tumor, or quantification tasks, like volume and surface measurement, segmentation is a well-established procedure in medical image computing [12], [13].

2.3.1 AI based Segmentation

AI based image segmentation methods play an essential part in medical imaging analysis and improve the diagnostic process [40]. In general, these methods can be categorized into three groups.

Semantic segmentation

Semantic segmentation can be described as assigning a label to each pixel of an image. This is commonly performed via machine learning tools like a support vector machine (SVM) or a fully convolutional network (FCN). For this to be attainable, the list of possible labels must be defined before the segmentation task. Additionally, semantic segmentation aims to partition an image into mutually exclusive subsets, where each subset is a meaningful region of the original image [41], [42].

Instance segmentation

Semantic segmentation, like mentioned above, focuses on assigning a label from a predefined list to each pixel in an image. Instance segmentation, on the other hand, alters this behavior through an important concept: Differentiating between instances of objects belonging to the same class. Thus, it can be defined as:

> "… the task of finding a simultaneous solution to object detection as well as semantic segmentation." ([43])

However, in instance segmentations, not every pixel gets its own label, rather a bounding box is created to delineate an instance of a class. In comparison to semantic segmentations, the instance variant is commonly performed by a deep convolutional neural network (DCNN) or a convolutional neural network (CNN) [43], [44].

Panoptic segmentation

Panoptic segmentation combines the benefits of the two methods mentioned above [45]. Every pixel gets assigned a semantic label and a unique instance identifier, thus differentiating between instances of the same class and performing an exact image segmentation [45].

2.3.2 Classical Segmentation

> "Fully automatic, completely reliable segmentation in medical images is an unrealistic expectation […]" ([46])

Thus, many clinical and preclinical segmentation workflows still rely on classical segmentation tools [47]. While AI based image segmentation techniques revolve around pattern

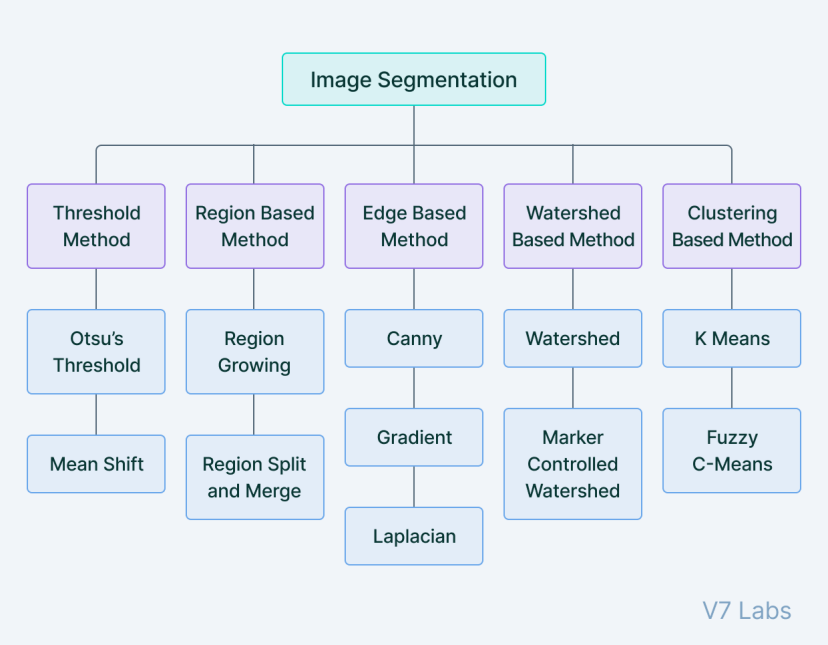


Fig. 8: Classical segmentation options [48]

recognition via some form of machine learning, classical image segmentation techniques rely on analysis of the qualities of each pixel and its neighborhood. These qualities may include properties like color, brightness, contrast and intensity. There exist a number of classical segmentation tools. Figure 8 gives an overview of the most commonly applied techniques [47], [49].

Edge Detection

Edge detection of grayscale images is defined as the task of finding sharp changes in pixel value intensity. These sharp changes in intensity indicate edges between objects in images.

Fig. 9: Edge detection

This entails susceptibility to image noise. In other words, if the image is rich in background noise, the quality of segmentation by edge detection will decline. Furthermore, if the pixel intensity gradient is less pronounced, edge detection quality is also impacted. Compare Figure 9, rows A and B. Row A has a sharply defined edge and thus, will pose no difficulty in edge detection tasks. Row B, however, displays less of a steep gradient and will thus, result in less accurate edge detection results.

Image intensity function

Fig. 10: Image intensity functions: step and ramp [50]

Conceptually, the image intensity function describes changes in intensity. Wherever the image intensity transitions from one value to another, the image intensity function changes slope accordingly, as can be seen in Figure 10. Edges are thereby signified by certain shapes of the function, such as steps, ramps and roofs. Edge detection methods work by analyzing the derivative of the image intensity function and can be categorized into two types of techniques based on the order of derivation they require [50].

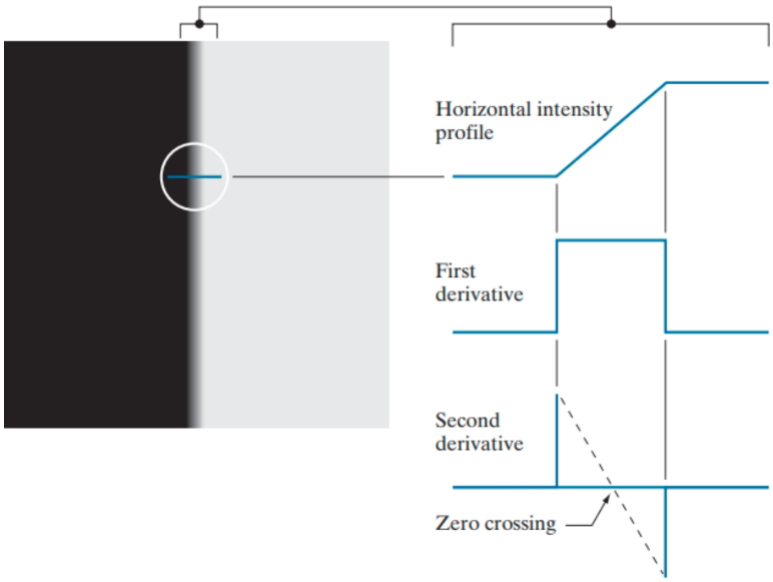


Fig. 11: Image intensity function derivatives: first, second and zero crossing [50]

First order derivative

The first derivative of an image intensity function, or horizontal intensity profile, measures the rate of change of pixel intensity (see: Figure 11). First order derivative detects edges by analyzing the intensity profile and looking for rapid changes. The first derivative can be approximated by using gradient operators like the Sobel operator. The latter is composed of 3×3 convolution kernels, which are specific to the image axis [50]. In other words, firstly, there is a horizontal kernel G_x, see Equation **(4)**, which is intended to emphasize changes in intensity in horizontal direction:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{4}$$

Secondly, the vertical kernel G_y, see Equation **(5)**, is used to highlight changes in intensity in vertical direction:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{5}$$

To conclude, the edge detection workflow using the Sobel kernels is as follows:

1. optionally reduce noise by applying Gaussian blur

2. convolute the image with the Sobel filter kernels to obtain edges in the respective x- and y-direction
3. calculate gradient magnitude
4. normalization of the output image

Second order derivative

The second derivative measures the rate of change of the first derivative. This technique analyzes the second order derivative of an image intensity function, or horizontal intensity profile, and look for so-called *zero crossings*. These zero crossings are defined as points where the second order derivative changes signs. In other words, it goes from negative to positive, or vice versa. Zero crossings therefore indicate the presence of edges in the rate of change of image intensity [50]. The second derivative can be approximated using the Laplacian operator, see Equation **(6)**:

$$Laplace = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{6}$$

To conclude, the edge detection workflow using the Laplacian kernel is as follows:

1. optionally reduce noise by applying Gaussian blur
2. apply Laplacian kernel
3. normalization of the output image

Region Growing

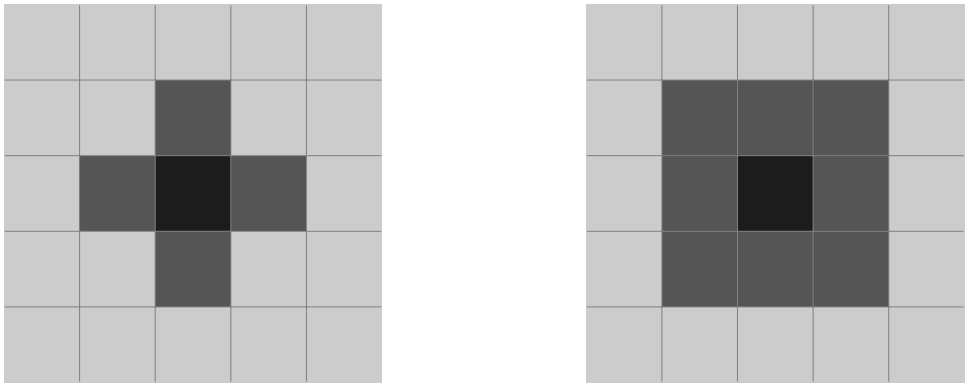
Fig. 12: Region growing seeds and result [51]

Region growing breaks down images into separate areas by grouping areas by their similarity. This is done by creating a selection of n seed pixels S, choosing some sort of principle of

similarity and then recursively grouping pixels adjacent to seeds according to their similarity score into regions R. Once a region stops growing, a new seed S, that does not belong to any region, is chosen and the process repeats [52], [53]. See Figure 12 for reference: It shows two times the same single slice of an MRI of the human brain. On the left image some seeds were defined, indicated by the green crosses on a lesion. On the right image, a region growing algorithm has put all adjacent pixels with similar intensity into the same group. To elaborate, the basic steps for a region growing algorithm are:

1. Selection of n seeds S
2. Selection of similarity score algorithm
3. Grouping of adjacent pixels according to their similarity into n regions R
4. Adjust similarity threshold or add seeds until result is satisfactory
5. Stop if similarity is lower than the defined threshold or all pixels belong to a group

The similarity score criteria can be any property of a pixel or group of pixels. Common criteria include color, intensity or texture [52], [53].



(a) 4-connected neighborhood (b) 8-connected neighborhood

Fig. 13: 4- and 8-connected neighborhoods

Additionally, a region growing algorithms can be configured to consider a seeds' 4-connected neighborhood (Figure 13a) or 8 connected neighborhood (Figure 13b) for similar pixels [52], [53]. Where the black center pixel represents a seed, and its gray neighborhood represents the considered pixels.

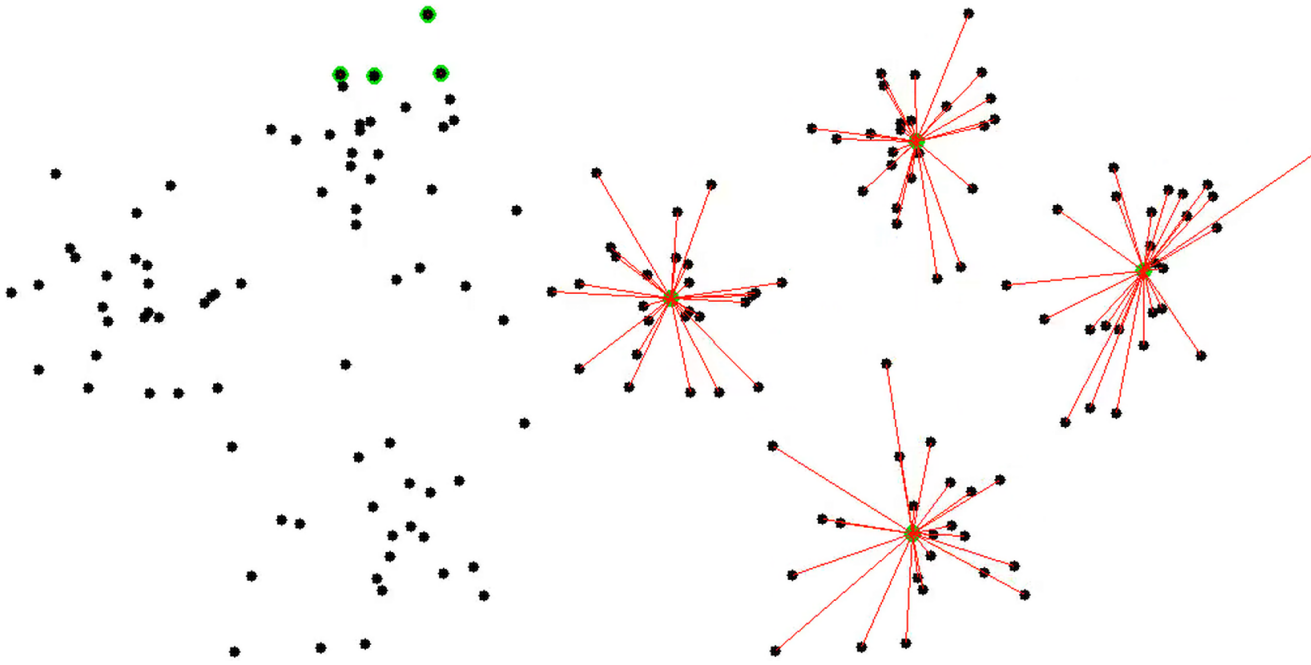
Threshold

A simple classical segmentation method is the threshold. Whereby the user defines a wanted pixel intensity range, and every pixel within that range is set to the binary value 1 and every other pixel is set to the binary value 0 [54].

Clustering

Clustering, specifically the k-means variant, refers to the division of a set of data into a specific number of groups. For this to be applicable, the items in a group need some kind of similarity. In image segmentation, such a similarity may be pixel properties like color or intensity. The algorithm then assigns all data points in a group according to their proximity to a seed [55]. The basic steps for a k-means clustering algorithm are as follows:

1. Initialize n clusters k
2. Distribute n initial seed points randomly across the image
3. Assign each data point to the closest seed point (see: Figure 14a)
4. Compute the center of each cluster and move the seed point there (see: Figure 14b)
5. Reassign each data-point to the new closest seed point
6. Repeat steps $3 - 5$ until it satisfies a tolerance value



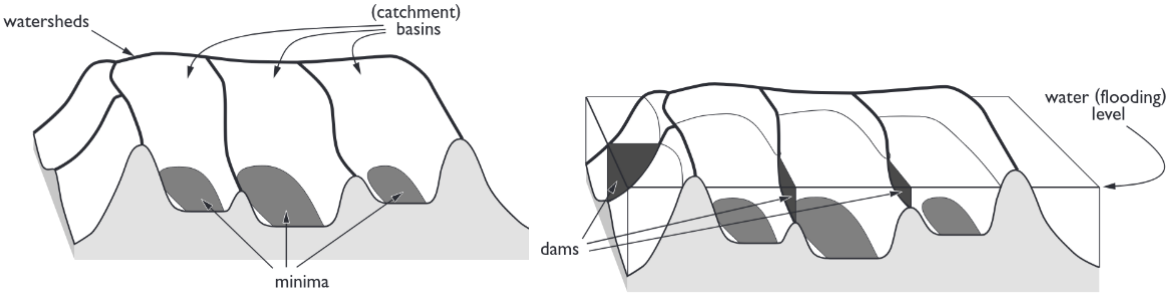
(a) k-means initial seed points [56] (b) k-means cluster centers [56]

Fig. 14: k-means clustering

A schematic overview of the k-means clustering algorithm can be seen in Figure 14. Figure 14a displays a set of data-points, represented as black dots, and a small number of cluster seeds, represented as green circles. The algorithm first tries to group the data-points according to their proximity to the seed points, represented as red lines. In Figure 14b the seed points have been relocated to better fit the dataset, and the clusters have changed accordingly.

Watershed

The watershed algorithm treats pixel intensities as elevations in a virtual landscape. In the latter, local minima are seen as the seed for *catchment basins*. On the other hand, local maxima, or so-called *watersheds* separate the basins from each other (see: Figure 15a). By *filling the basins with water* each pixel in an image is assigned to a label. This label depends on which basins water reaches it first. In order to avoid relabeling pixels, *dams* are created when the floodwater of two basins meet. Finally, by modifying the water height, the watershed algorithm can be adjusted to either create larger or smaller regions (see: Figure 15b) [57], [58].



(a) Watershed minima with basins [58] (b) Watershed dams and flooding level [58]

Fig. 15: Watershed landscape

2.4 Segmentation representation

Segmentations can be represented in a number of data structures. Figure 16 gives an overview of the most common representations. Of those shown, the binary labelmap is the most commonly used storage format. For most workflows, a single representation of data is not sufficient. An example of the requirement of several data structures within a single workflow is the calculation of a dose volume histogram (DVH) in radiation therapy. The planning target volume (PTV) and organs at risk (OAR) are typically segmented by defining their cross-section per image slice on a planning CT or MRI for the extent of the whole structure [59]. This results in a list of planar contours similar to Figure 16/D. These planar contours must then be voxelized into binary labelmaps, like in Figure 16/A. As for the next step, these binary labelmaps are used to compute dose distribution volumes and dose volume histogram. If a workflow requires 3D visualization, closed surfaces must be constructed. This can be done by converting, using either planar contours or binary labelmaps as the source dataset [12].

The following sections should give a short overview of the mentioned data structures and their use cases.

Fig. 16: Brain stem in different representations: A: Binary labelmap, B: Closed surface, C: Fractional labelmap, D: Planar contours, E: Ribbon mode [12]

2.4.1 Binary labelmap

Binary labelmaps (see: Figure 16/A) are the most common representation, as they have several benefits for segmentation software developers. The process of editing these labels is relatively straightforward, as any modifications to the label map entail the addition or subtraction of voxels to the segment label. Storage is simplified as well, as each voxel can only belong to a single segment. This enables the storage of the whole dataset in a single 3D array. If label overlap is desired, the base 3D volume needs to be duplicated in order to store the overlapping segment in a separate 3D array. As a result storage requirements for the segmentation are inflated, thus binary labelmaps tend to be memory inefficient for overlapping structures and large numbers of labels [13].

2.4.2 Fractional labelmap

Fractional labelmaps (see: Figure 16/C) are a segmentation representation, which improve on the concept of binary labelmaps mentioned above. Instead of a single voxel value representing a discrete label, in fractional labelmaps, each voxel value represents the probability that a structure is contained inside or outside that voxel. As a result, fractional labelmaps can store higher segmentation resolutions than the source dataset [12], [60]–[63].

2.4.3 Closed surface

Closed surface representations, like in Figure 16/B, are composed of a triangulated mesh, which in turn, is built of a series of vertices or triangles. In terms of segmentation, the mesh surface boundary defines the segmentation volume, so that every voxel encompassed by the mesh belongs to the segment [12], [63], [64]. In order to store a vertex representation, each triangle (vertex) needs to be stored separately. A schema of the representation can be seen in Figure 17a. The representation of a single voxel starts with the definition of its facet normal or unit normal vector in 3D cartesian space. This vector points outward from the vertex, effectively defining the basic orientation of the vertex. Then, the triangle corners are defined in 3D cartesian space (x, y, z). Figure 17b gives an overview of a possible triangle orientation. As illustrated, the first corner overlaps with the third corner at the point where $x = -1$ and $z = 1$. The second corner overlaps with the first only at position $z = 1$, whereas it overlaps with the third at both $y = 1$ and $z = 1$. Ultimately, the third corner overlaps with the first at the point $x = -1$ and with the second at the points $y = 1$ and $z = 1$.

```
facet normal n(i) n(j) n(k)
[...]
  vertex v1(x) v1(y) v1(z)
  vertex v2(x) v2(y) v2(z)
  vertex v3(x) v3(y) v3(z)
[...]
```

(a) Closed surface mesh schema

```
facet normal 0.000 0.000 1.000
[...]
  vertex -1.000 -1.000 1.000
  vertex  1.000  1.000 1.000
  vertex -1.000  1.000 1.000
[...]
```

(b) Closed surface mesh sample

Fig. 17: Surface mesh representation

To finally form a segmentation form these vertices, a series of them must be composed that encompasses the whole segmentation volume. This in turn results in a representation like in Figure 18.
The closed surface representation is ideal for 3D visualization of segmentations but has some downsides as well. First and foremost, they can be self-intersecting and thus invalid representations of reality. Furthermore, to edit a closed surface, the point coordinates of individual vertices must be modified, and the unit normal vector recalculated [12].

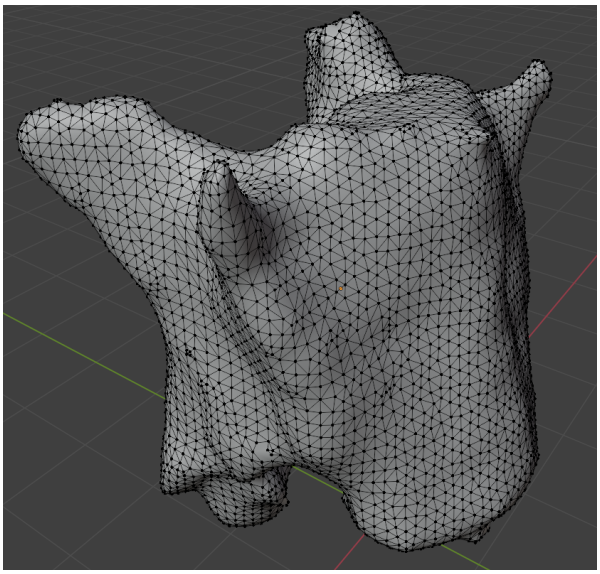
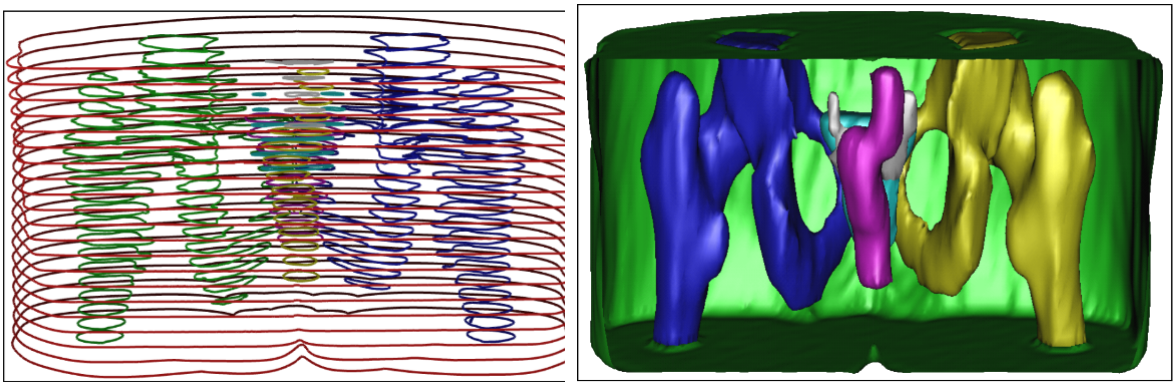


Fig. 18: Closed surface mesh of a mouse vertebrae segmentation

2.4.4 Planar contours and ribbons

Planar contours (see: Figure 16/D) as well as ribbon models (see: Figure 16/E) define a segmentation by encompassing it in rings or ribbons for each slice separately. They are created by tracing contours around a region of interest (ROI) in parallel planar slices [12], [65]. In 2D, planar contours are represented by polylines. In 3D representations, however, these polylines need to be stacked and gaps require interpolation. Then, the final volume is smoothed.



(a) Planar contours outlining structures of interest (b) Reconstructed surfaces from planar contours

Fig. 19: Planar contour representation [65]

As can be seen in Figure 19a planar contours or ribbons can be highly accurate in 2D. Experts can also draw polylines regardless of noise, low-resolution data, and overlapping pathologies. Furthermore, for radiation therapy, the DICOM standard requires segmentations to be stored as *structured sets*, essentially a series of planar contours. However, when it comes to 3D

visualization, planar contours as well as ribbons offer comparatively poor quality visuals [12], [65].

2.5 Segmentation evaluation

This section is intended to give an overview of the two supervised segmentation assessment algorithms used in this study.

2.5.1 Dice coefficient

The Sørensen–Dice coefficient, also known as Dice similarity coefficient (DSC), is described by the Equation **(7)** [16]:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|} \tag{7}$$

It can be used to describe the similarities of two segmented images or volumes. It is two times the intersecting voxels of segmentation x and y divided by the sum of the number of voxels of x and y.
When dealing with Boolean data, the formula can be transformed to the following Equation **(8)**:

$$DSC = \frac{2 * TP}{2 * TP + FP + FN} \tag{8}$$

It is two times the True Positive (TP) intersection divided by two times the True Positive plus the False Positive (FP) plus the False Negative (FN). In the context of volume comparison, this can be interpreted as two times the number of overlapping voxels between two segmentations divided by the total number of voxels in both segmentations [66].

2.5.2 Pompeiu-Hausdorff Distance

The *Pompeiu–Hausdorff distance* or simply *Hausdorff distance* is named after its creators Felix Hausdorff and Dimitrie Pompeiu [67].

> "[…] it is able to measure the degree of mismatch between two sets from the distance of the point of A that is farthest from any point of B, and vice versa." ([68])

In computer graphics, the Hausdorff distance can be used to compute the difference between two representations of the same 3D object. The Hausdorff distance is described by its general definition in Equation **(9)**. Where the Hausdorff distance H from a set of points

$A = a_1 \ldots, a_p$ to another set of points $B = b_1 \ldots, b_q$ is a maximum distance function $max(h(AB), h(BA))$ [17], [68], [69].

$$H(A, B) = max(h(AB), h(BA)) \tag{9}$$

The functions $h(A, B)$ and $h(B, A)$ (see: **(10)** and **(11))** are called directed Hausdorff distances from A to B and vice versa. Where $d(a, b)$ is the Euclidean distance from a to b. It identifies the nearest point in the set B for every point in the set A and isolates the largest distance between these points. As a result, the Hausdorff distance can be effectively put to use for measuring the mismatch between to sets of points [17], [68], [69].

$$h(A, B) = max_{a \in A}(min_{b \in B}\ d(a, b)) \tag{10}$$

$$h(B, A) = max_{b \in B}(min_{a \in A}\ d(b, a)) \tag{11}$$

To conclude, the algorithm works by iterating over the following steps [17], [68], [69]:

1. For all points in set A: find the minimum euclidean distance to points in set B
2. Compute the maximum of those minimal distances
3. Repeat steps 1 and 2 with reversed roles for points of sets A and B
4. Compute the maximum of the two maxima determined by the previous steps

2.6 Mouse bone anatomy

This section is dedicated to the bone anatomy of laboratory mice (Mus musculus) (see: Figure 20). Apart from the obvious size difference, mice are biologically comparable to humans (Homo sapiens), notable differences between human and mouse bone anatomy will be pointed out [70].

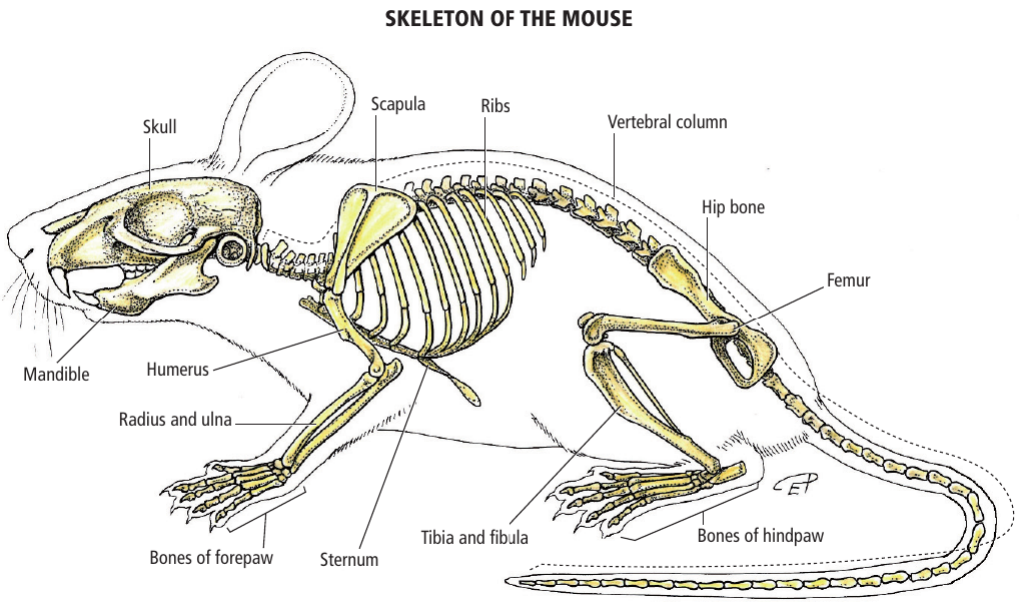


Fig. 20: Mouse bone model [70]

2.6.1 Skull

Most notably, the mouse skull reflects the importance of its sense of smell (olfaction), by its elongated facial features and large tooth size (see Figure 21). Meanwhile, the human head is shaped for a larger brain. In mice, the interparietal bone is well-developed compared to humans. Furthermore, in mice, the two mandibles are not fused together like in humans, they are rather joined by cartilaginous tissue [71]–[73].

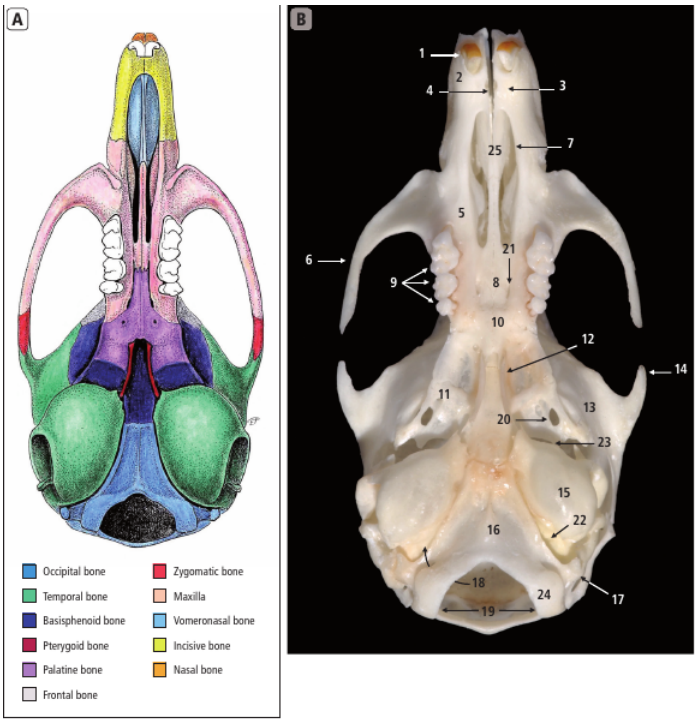


Figure 2-14. Cranium and facial bones. Ventral view. A) Diagram. **B)** Isolated skull.

1: Incisor tooth; 2: Incisive bone; 3: Palatine process (incisive bone); 4: Incisive canal; 5: Maxilla; 6: Zygomatic process (maxilla); 7: Palatine fissure; 8: Palatine process (maxilla); 9: Molar teeth; 10: Palatine bone; 11: Basisphenoid bone; 12: Pterygoid bone; 13: Temporal bone; 14: Zygomatic process (temporal bone); 15: Tympanic bulla (tympanic part, temporal bone); 16: Basioccipital bone; 17: Paracondylar process; 18: Hypoglossal canal; 19: Foramen magnum; 20: Foramen ovale; 21: Major palatine groove and major palatine foramen; 22: Jugular foramen; 23: Sphenotympanic fissure; 24: Occipital condyle; 25: Vomeronasal bone.

Fig. 21: Mouse Skull [70]

2.6.2 Spine

Mouse spines (see Figure 22) are made of 7 cervical vertebrae, 13 thoracic vertebrae, 6 lumbar vertebrae, 4 sacral vertebrae and 20-30 caudal vertebrae [73]. Mice can possess up to 60 vertebrae, whereas humans only possess 33[3]. The largest part of this difference is made up by the fact that mice possess a tail composed of 27-31 coccygeal vertebrae, while humans only possess 4 non-articulated coccygeal vertebrae. The remainder is due to minor differences in thoracic, lumbar and sacral vertebrae counts [71], [74], [75]. Rodent and human vertebrae additionally differ in vertebral body size.

Rodent vertebrae have a comparatively small body, as due to their quadrupedal nature, mice spines do not undergo as much axial stress as human spines [71], [73], [74].

[3] 34 with anatomical normal variant L6

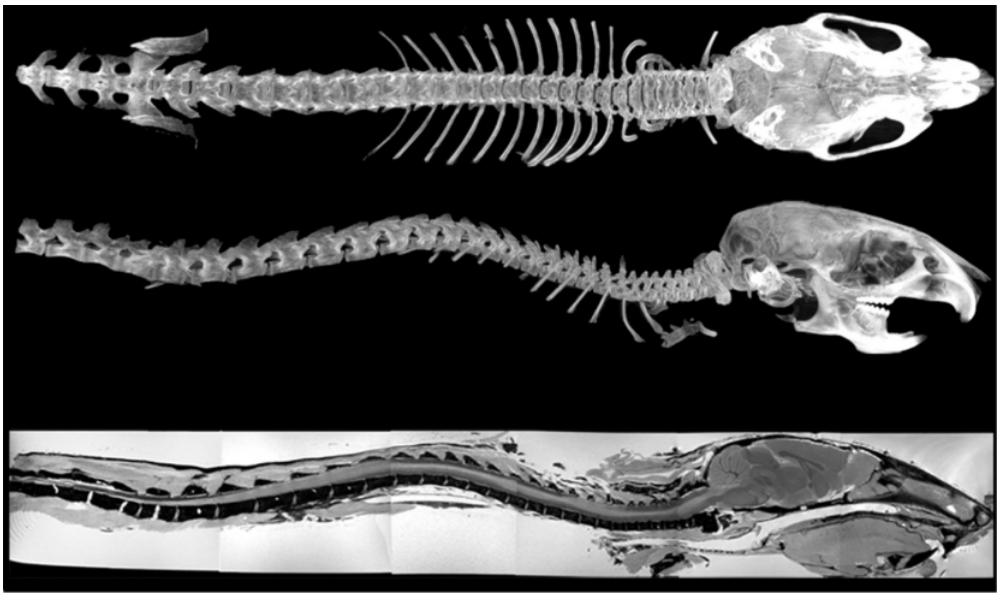


Fig. 22: Mouse spine X-ray and MRI [75]

2.6.3 Thorax and Pelvis

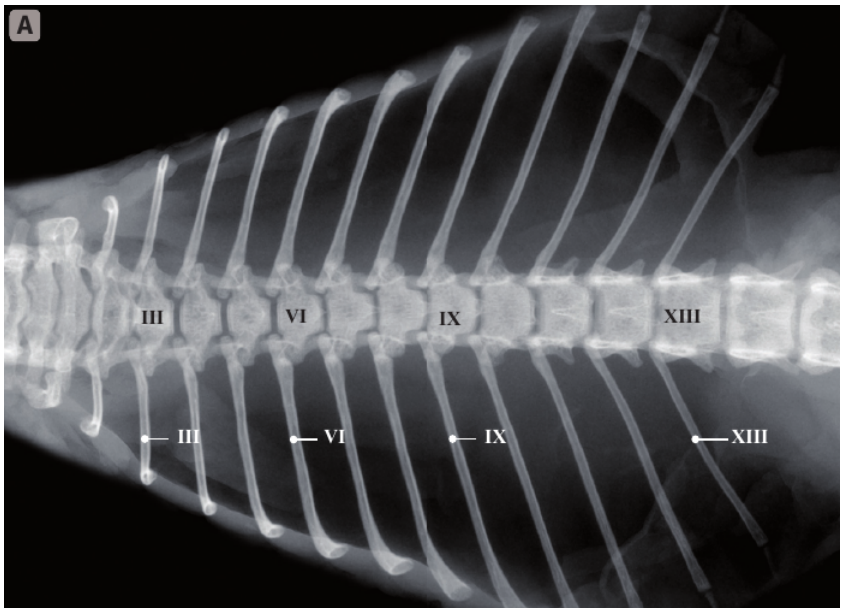
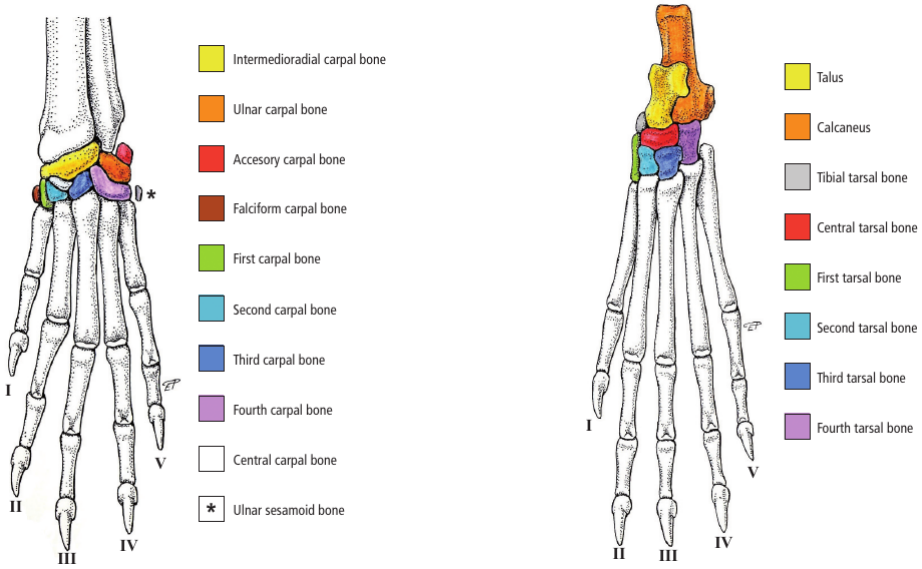


Fig. 23: Mouse thorax X-ray [75]

In many quadrupedal mammals the clavicle has dramatically reduced or even disappeared. Despite their quadrupedal nature, mice have a well-developed clavicle, that joins the sternum and acromion of the scapula, like in humans [70]. Mice possess 13 pairs of ribs (costae) as can be seen in Figure 23, including 7 pairs that are articulated to the sternum, also called true ribs and 3 pairs of asternal or false ribs. This is very similar to human anatomy, with the most

notable difference being that humans only have 12 pairs of ribs. The mouse pelvis, like in humans, is composed of the ilium, ischium and pubis [72]–[74].

2.6.4 Extremities



(a) Mouse front paw [74]

(b) Mouse hind paw [74]

Fig. 24: Mouse paw anatomy

As can be seen in Figure 24a after the mouse radius and ulna, the wrist joint is made up of 9 bones:
The carpal bones I-IV, the intermedioradial carpal bone, the ulnar carpal bone, the accessory carpal bone, the central carpal bone, and the proximal sesamoid bones. Which are articulated with 5 metacarpal bones and phalanges. Humans however, possess 8 carpal bones, which include the scaphoid, lunate, triquetrum, pisiform, trapezium, trapezoid, capitate and hamate [73].
As can be seen in Figure 24b, the mouse ankle is made up of 8 bones: The talus, calcaneus, tibial tarsal bone, central tarsal bone and tarsal bones I-IV. Whereas the human ankle joint is made up of 7 bones: Calcaneus, cuboid, talus, navicular, lateral cuneiform, middle cuneiform and medial cuneiform [73].
Supratrochlear foramen in humerus in mice but not in humans [73], [74] radius and ulna are independent and not fused as it occurs in other mammals.
The rodent fibula is half the length of those of humans and fuses with the tibia, whereas the human fibula articulates with the ankle [71], [73].

3. Materials and Methods

In this section, the various materials and methods used in this thesis will be described. Furthermore, the rationale of using the specific tools for a task will be explained.

3.1 Literature

To obtain a systematic overview of the topics micro-CT segmentation and guide authoring, a literature search was performed with the help of online search engines and databases like *Google Scholar*, *PubMed*, *ScienceDirect*, *IEEE Xplore* and *ResearchGate*. However, to get a comprehensive understanding of the used software *3D Slicer* and *SlicerRT* also less scientific resources had to be consulted. Namely, their respective documentation websites and also explanations in video form, mostly hosted on YouTube (Google LLC, San Bruno, California, USA). Keywords for resource search included terms like "3DSlicer", "micro-CT", "segmentation", "segmentation evaluation", "software documentation". To make sure the resources were up-to-date with current developments in image segmentation, literature older than 10 years was excluded. An exception was made for material explaining basic concepts that have been developed several decades ago, as can be seen in Chapter 2.

3.2 Datasets

All nine datasets used in this study were acquired at the competence center for preclinical imaging and biomedical engineering at the University of Applied Sciences, Wiener Neustadt. The total body scans were performed on a Molecubes X-Cube micro-CT machine (Molecubes NV X-Cube, MOLECUBES NV, Gent, Belgium). The specimens were non contrast enhanced mice, however, remnants of contrast agent could be seen in the vesica urinaria of all scanned animals. All scans were acquired with a tube voltage of 50kVp and a tube current exposure time product of 70mAs. Pixel spacing, the distance between the centers of neighboring pixels in the x/y-plane, was consistent across all scans with a distance of $0.1mm$ and the slice thickness was set to 0.1mm resulting in isotropic voxels.
A summary of all scan parameters can be seen in Table 2. The author has not performed the scans themselves, but rather repurposed a dataset from another study. Therefore, any information about scan parameters and reconstruction settings had to be taken from the DICOM metadata.

Manufacturer	Molecubes
Model	X-Cube
Tube voltage (kVp)	50
Tube current (mAs)	70
Exposure time (sec.)	42
Reconstruction diameter (mm)	70
Reconstruction algorithm	ISRA
Matrix size (pixels)	700×700
Bit depth (bit)	16
Slice thickness (mm)	0.1
Pixel spacing (mm)	0.1

Tab. 2: Dataset scan parameters

Dataset preparation

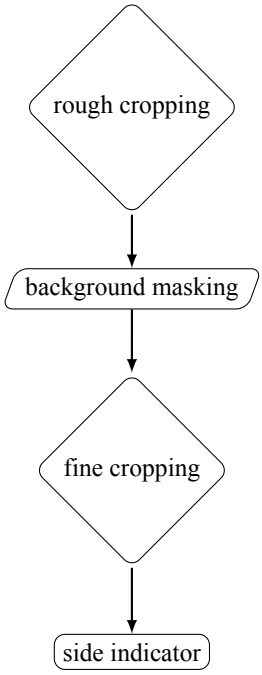


Fig. 25: Dataset preparation

As can be seen in Figure 25 several steps of preparation were performed before the dataset was handed out to testers in conjunction with the guide. All modifications were done using built in 3D Slicer tools. Firstly, a rough cropping of the dataset was performed. The goal was to reduce the size of the dataset to only include anatomical structures relevant for the segmentation assignment. This first rough cropping operation already reduced the dataset geometric size to approximately one fifth of its original dimensions.
Next, a background masking operation has been applied with the goal of reducing file size. In

order to achieve this, first a threshold operation was performed to segment background noise in the scan as well as anatomical structures composed of soft tissue. The resulting segment was the base for a masking operation where the segmented volume was uniformly overwritten with the HU value of -1000.
Thirdly, the remaining volume was cropped again, making only minor adjustments to the ROI. The goal was to get rid of small artifacts created by the first cropping operation. Finally, to aid the testers in determining if they are currently working on the left or right side anatomy of the dataset, side indicators were added to the volume. This was done, using 3D Slicers Engrave tool available via the *SlicerSegmentEditorExtraEffects* [76] plugin. Effectively adding an artificial segment displaying left (L) and right (R) when viewed in 3D view. The final dataset was reduced to the anatomical bone structures of both shoulder joints. Effectively containing sinistral and dextral clavicle, humerus and scapula.

3.3 Segmentation Software

3D Slicer is a free and open-source software for visualization, processing, segmentation, registration and analysis of medical, or other 3D image datasets. It is actively developed and supported by its core development team and ever-growing community to stay on top of cutting edge development in image processing. Furthermore, 3D Slicer's functionality can be extended via extensions written in C++ or Python [8]. 3D Slicer was selected in favor of ITK-SNAP because the author was already familiar with the software, while choosing ITK-SNAP would have entailed a familiarization phase.
A common extension for 3D Slicer is SlicerRT, which is designed to enable common radiotherapy workflows inside 3D Slicer. It includes tools for visualization, quantitative metrics computation, comparison and DICOM import and export [18]. Most importantly for this study, it includes a tool for segmentation comparison. In this tool, segments can be compared by simply loading two different datasets and choosing the appropriate segments to compare against each other. To conclude, it was used for segmentation comparison of the ground truth segmentation as well a s the testers aided and unaided segmentations.

3.4 Similarity score

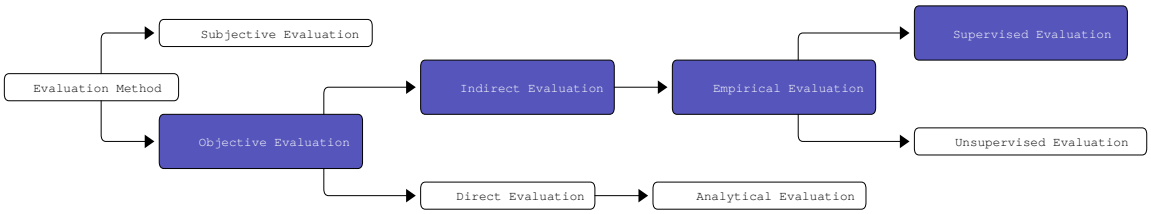


Fig. 26: Evaluation Method decision tree [77]

For comparison of the segmentations done by testers with the ground truth segmentation, two well established metric were selected. The Dice similarity coefficient (DSC) is a similarity score applicable to a multitude of datasets and has been used as such by numerous studies [78], [79]. However, the DSC is based on segmentation overlap and therefore has no concept of shape. Rayed, Islam, Niha, *et al.* as well as Wang, Wang, and Zhu recommend the employment and comparison of a second similarity metric, which calculates the surface distance of two segmentations [68], [81]. As a result, the Hausdorff distance (HD) was chosen by the author as a second metric. Coincidentally, both are readily available in 3D Slicer by the 3rd party extension called SlicerRT [18]. Both, the Dice similarity coefficient and the Hausdorff distance are examples of supervised, empirical, indirect and objective evaluation methods (see Figure 26), where the quality of a segmentation is determined by comparing it to a known high quality reference segmentation. The comparison is performed by calculating a similarity score and is thus objective but indirect. In contrast, a subjective evaluation would be the act of visually inspecting and subjectively assessing segmentation quality. Direct analytical evaluation would be the analysis of the segmentation algorithm itself. Unsupervised evaluation refers to the evaluation of a segmentation without comparing it to a reference. Instead, it is evaluated by calculating criteria that define a good segmentation [77].

3.5 Segmentation

A nearly full skeleton segmentation of the mouse specimens in the dataset was performed by the author. Depending on image quality, artifacts, and resolution, on average about 102 individual bones could be segmented in a single specimen. The only exception being the ribs, which were segmented pairwise. In other words, instead of creating individual segments for costa 1 left and right, a single segment was created called costae 1. As indicated the individual bones were segmented and labeled using their Latin names, as this is common practice in literature and research [70], [71], [74], [75]. Image resolution limitations were found to impair segmentation accuracy in the wrist and ankle joints and thus, the affected bones were excluded from segmentation. As most fully automatic segmentation tools included in 3D Slicer were found to have a high error rate when applied to the dataset. In this study, they were disregarded for usage in the ground truth segmentation. The ground truth segmentation was performed by the author using strictly manual and semi-automatic segmentation tools included in 3D Slicer. Extensive usage of anatomy atlases and anatomy browsers were performed during the segmentation [70], [71], [73]–[75], [82]. To ensure consistent segmentation quality, a strict order of operations was devised. Figure 27 provides a visual overview of the devised segmentation workflow. After successful completion of each task, the changes were written to a save file.

As working on large datasets in 3D Slicer pose a risk of unexpected software termination, the first step in the workflow was to perform operations which reduce the size of the dataset.

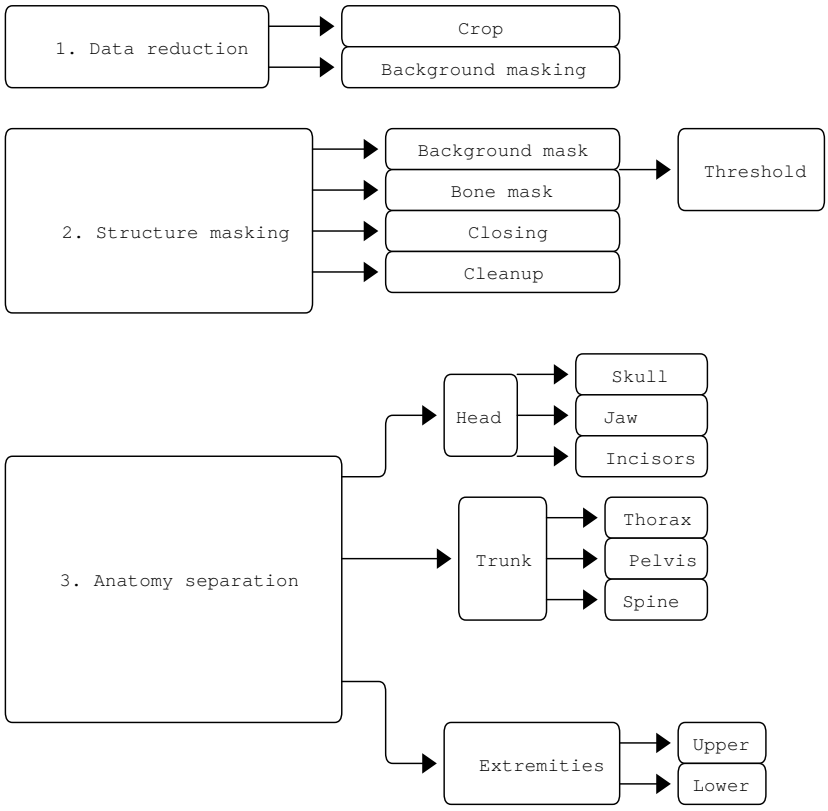


Fig. 27: Segmentation workflow

Firstly, a rough cropping of the volume was performed. The goal was to crop the volume down to the minimal required size, without affecting the anatomical structures of interest. Secondly, to further reduce the dataset size, a background masking operation was performed. The goal was to replace background noise outside relevant structures with a uniform HU value of -1000 HU. It was experimentally determined that this greatly improves dataset compression ratios. Next, structure masking was performed, whereby first a bone mask was created via the threshold tool. The threshold range was adjusted to encompass all bone structures, a lower threshold value around 550 HU was found to be optimal for most specimens. To compensate for small inconsistencies in the threshold operation, a closing operation was performed immediately after. 3D Slicer does not include a closing operation tool, thus the task was accomplished by using 3D Slicers *Margin grow* and *Margin shrink* tools in sequence. The same steps were performed to a background mask. For the actual segmentation step, the mouse anatomy was divided into the general body parts, such as head, trunk, and extremities. Which in turn were divided into the individual bones that compose them. Separation of the individual bones was done manually via paint operations in 2D and 3D. Whereas segmentation of larger bone volume was performed using semi-automatic tools like 3D Slicers *Islands* tool. A brief summary of segmented bones can be seen in Table 3.

Head	Os Hyoideum	**Trunk**	Sternum
	Mandible		Costae 1-13
	Skull		Clavicle (sin. & dext.)
	Incisors (sup. & inf.)		Scapula (sin. & dext.)
			cervical Vertebrae 1-7
			thoracic Vertebrae 1-13
			lumbar Vertebrae 1-6
			sacral Vertebrae 1-4
			Pelvis
			caudal Vertebrae 1-27
upper Extremities	Humerus (sin. & dext.)	**lower Extremities**	Femur (sin. & dext.)
	Radius (sin. & dext.)		Sesamoid (sin. & dext.)
	Ulna (sin. & dext.)		Tibia (sin. & dext.)
	front Paw (sin. & dext.)		Fibula (sin. & dext.)
			Patella (sin. & dext.)
			hind Paw (sin. & dext.)

Tab. 3: Segmented bones

3.6 Guide composition

A guide was created to be sent out to testers in conjunction with a small dataset. The Diátaxis framework, described in Section 2.2.2, was chosen as a basis. Whereby, the guide was determined to be required to be a composite of the Diátaxis concepts of a *Reference manual* and a *How-to-guide*. This is due to the following situation. According to Procida *How-to-guides* can only ever be useful to already competent users and may not allow explanations. However, it was assumed that the potential testers have little to no experience with segmentation of micro-CT datasets. Additionally, it was assumed that potential testers have little to no experience in segmentation with 3D Slicer. Therefore, the *How-to-guide* portion of the guide had to be paired with introductory information about micro-CT dataset segmentation using 3D Slicer. For this part Procidas Diátaxis concept of a *Reference manual* has been chosen. Additionally, this has been paired with extensive use of the pictograms displayed in this section.

Pictograms

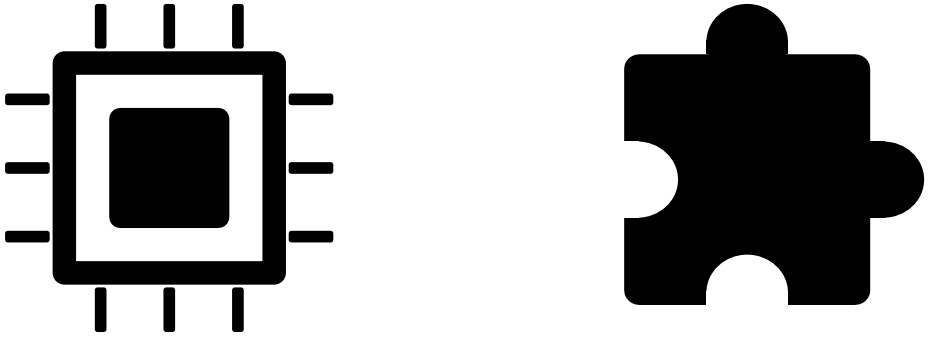
The following black and white pictograms were drawn by the author himself and were explained to readers of the guide in its introductory section:

(a) 2D symbol (b) 3D symbol

Fig. 28: Figure 28a and Figure 28b indicate whether a tool may be used in 2D or 3D mode or view.

(a) Hint symbol (b) Irreversible symbol

Fig. 29: Figure 29a indicates a hint or tip while Figure 29b informs the reader that an action may not be undone easily.



(a) Performance symbol (b) Plugin symbol

Fig. 30: Figure 30a informs the reader about a potential resource intensive operation and Figure 30b indicates that some tool may only be available after installing a 3rd party add-on.

3D Slicer displays the loaded dataset in 2D slice views by default. The user has the option to visualize the dataset in 3D via 3D Slicer's volume rendering capabilities. Considerably more useful, however, is the option to display already segmented parts of the dataset in 3D view. Most segmentation tools in 3D Slicer are only usable in 2D views and were therefore marked in the guide as such with Figure 28a. A smaller part of 3D Slicer segmentation tools can be used in 3D view and were therefore marked in the guide as such with Figure 28b. The 3D symbol, which when compared to the 2D symbol, is displayed with a depth of field (DOF) effect, giving it a 3-dimensional appearance. This is intended to draw the reader's attention to 3D Slicers 3D rendering view. In order to draw the readers' attention to a specific recommendation on how to use a tool, Figure 29a was utilized. The hint symbol can be described as a simple glowing light bulb, intended to suggest that it symbolizes an idea to consider. It was always inserted into the guide with a small section of text explaining to the user how to ideally utilize the current tool. 3D Slicer allows for many operations to be undone, some operations, however, have a permanent effect. In most cases, this is because the original data has been overwritten by the user's last operation. Unfortunately, 3D Slicer does not explicitly inform the user which operations potentially overwrite source data and are therefore impossible to undo. In the guide, Figure 29b was used to inform the reader about such issues. This irreversible symbol displays a counterclockwise circular arrow that is disrupted by a thunderbolt, which extends from the top right to the bottom left of the symbol. The counterclockwise circular arrow is intended to convey the idea of undoing something by "turning back time" and the thunderbolt interrupts or blocks this action and thus give the reader the idea that something cannot be undone. A similar problem occurs when using segmentation tools that are computationally expensive. These are tools which may work fine for most users on small datasets and lead the operator to believe they will work to the same extent on larger datasets. However, loading and segmenting large datasets with some tools, especially semi-automatic segmentation tools, will lead to crashes due to 3D Slicer running out of Random Access Memory (RAM). Thus, segmentation tools which may be computationally expensive were marked with Figure 30a paired with a short explanation in the guide. The performance symbol displays a central processing unit (CPU) which is intended to remind the reader that they are using a computer that has limited computational capabilities. Some segmentation tools referenced in the guide are only available to users after installing a third party plugin for 3D Slicer, wherever this is the case, readers were informed by an instance of Figure 30b. The plugin symbol displays a puzzle piece and is meant to convey the idea that some functionality must be intentionally added to 3D Slicer.

3.7 Research Design

For this study datasets of nine mice, scanned via micro-CT were taken (see Section 3.2). The author performed bone anatomy segmentation on these datasets via manual and semi-

automatic tools usable in the software called 3D Slicer (see Section 3.5). Then a guide was created by the author of this thesis (see Section 3.6). This guide would be sent out to testers in conjunction with a reduced dataset containing only six bone structures making up both shoulder joints (see Section 3.2). The participants were instructed to segment the anatomy of the left shoulder joint before reading the guide. This was done to get a baseline value of their segmentation skills. After completing segmentation of the left shoulder joint, the testers were instructed to read through the guide and finally, with the help of the guide, segment the right shoulder joint. The participant then saved their work in a Medical Reality Modeling Language bundle (MRB) file and sent it back to the author. Segmentation analysis was performed by loading the segmented dataset of a tester and the reference segmentation dataset into 3D Slicer. And computing both Dice similarity coefficient and Hausdorff distance via the 3D Slicer extension *SlicerRT*. All this was performed on a custom-built computer from 2019, which was running Linux, for more information see: Table 1. Analysis results were stored in Comma-separated value (CSV) files for further processing.

4. Results

Supervised segmentation evaluation was performed with the 3D Slicer extension *SlicerRT*. Segmentation results of seven testers, which did six segmentations each, have been analyzed. These 42 segmentations were compared to the reference segmentation via the Dice similarity coefficient and Hausdorff distance. Note that the Dice similarity coefficient is a dimensionless score between 0 and 1. The testers had to perform the initial segmentation without the help of the guide on page 73. They were told to perform this segmentation on the left side anatomy, which from here on will always be colored *blue* in plots. Another segmentation was performed by the testers after reading through the guide on the right side anatomy, which from here on will always be colored *red* in plots. To adequately display potential outliers Hausdorff distance (HD) results will always be given with the average and maximum distance from the ground truth in millimeters (mm).

4.1 Test person 1

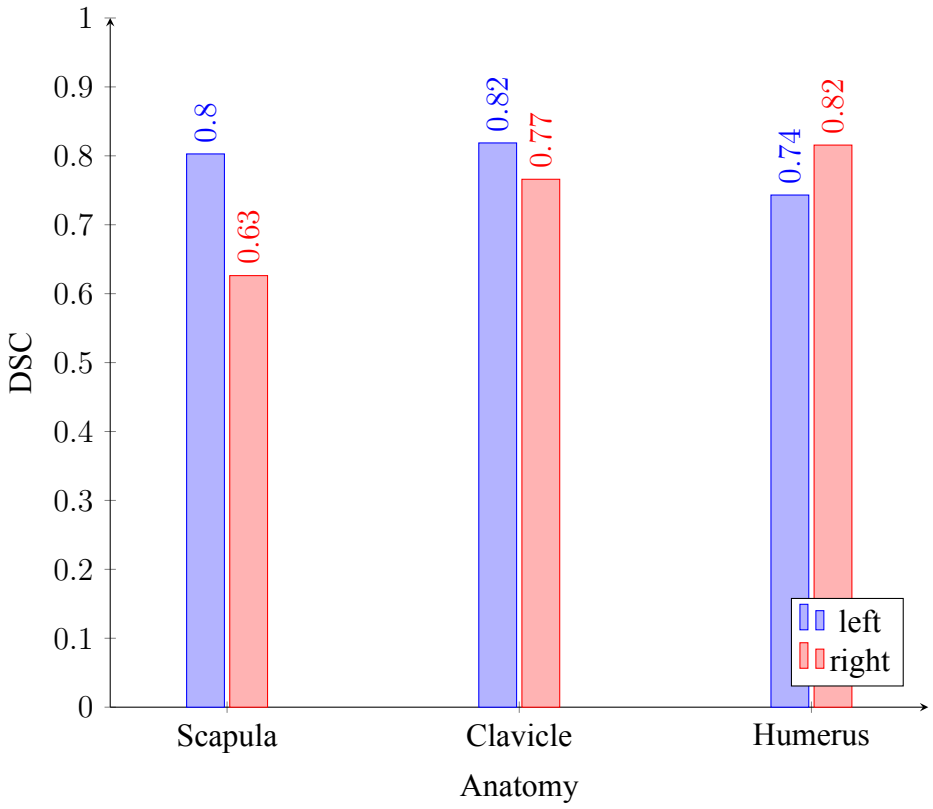
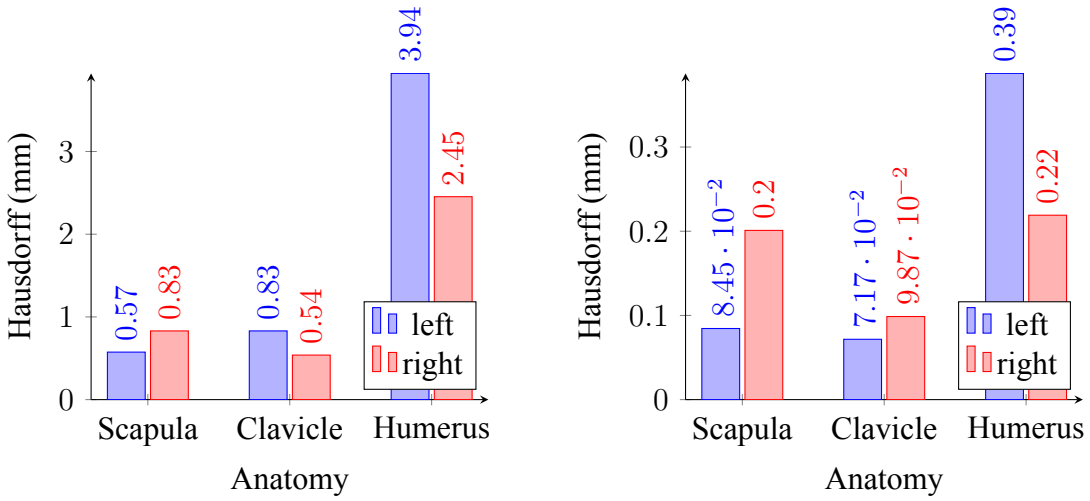


Fig. 31: DSC results: Test person 1 (more is better)



(a) HD maximum results: Test person 1

(b) HD average: Test person 1

Fig. 32: Hausdorff distance (HD) results: Test person 1 (less is better)

Figure 31 illustrates the segmentation comparison results via the Dice similarity coefficient (DSC) for test participant 1. The DSC-based segmentation overlap demonstrates a decline in accuracy for the scapula (-0.17) and clavicle (-0.05), while exhibiting an uptick for the humerus ($+0.08$). Figure 32 illustrates the segmentation comparison results via the Hausdorff distance (HD) for test person 1. Whereby Figure 32a displays the maximum distance in millimeters (mm) while, Figure 32b displays the average distance in millimeters. As can be seen, in Figure 32b, the segmentation of test person 1 on average was further from the ground truth in case of the scapula ($+0.1155mm$) and clavicle ($+0.027mm$). With regard to the humerus, the segmentation was found to be in closer alignment with the ground truth, exhibiting a distance of ($-0.17mm$).

4.2 Test person 2

Figure 33 depicts the segmentation comparison results via the Dice similarity coefficient (DSC) for test person 2. The segmentation overlap according to the DSC shows an increase in segmentation accuracy for all segmented anatomical structures: scapula ($+0.45$), clavicle ($+0.32$) and humerus ($+0.69$). Figure 34 depicts the segmentation comparison results via the Hausdorff distance (HD) for test person 2. Whereby Figure 34a displays the maximum distance in millimeters (mm) and Figure 34b displays the average distance in millimeters. As can be seen, in Figure 34b, the segmentation of test person 2 on average was slightly further from the ground truth for all segmented anatomical structures: scapula ($+0.39mm$), clavicle ($+0.3mm$) and humerus ($+0.07mm$).

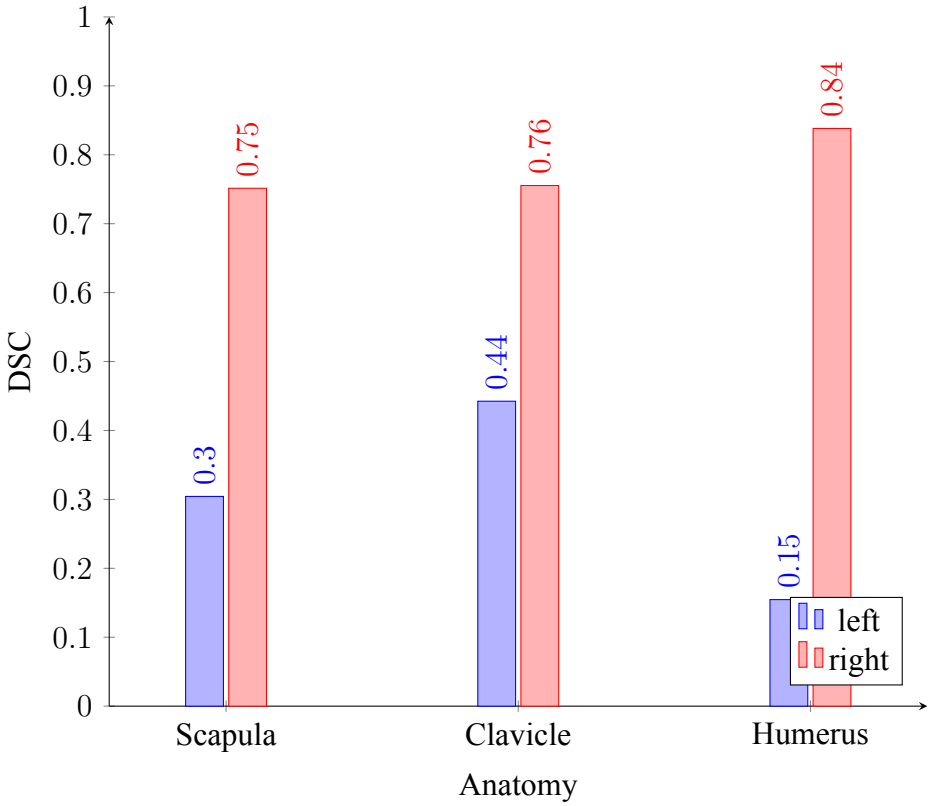
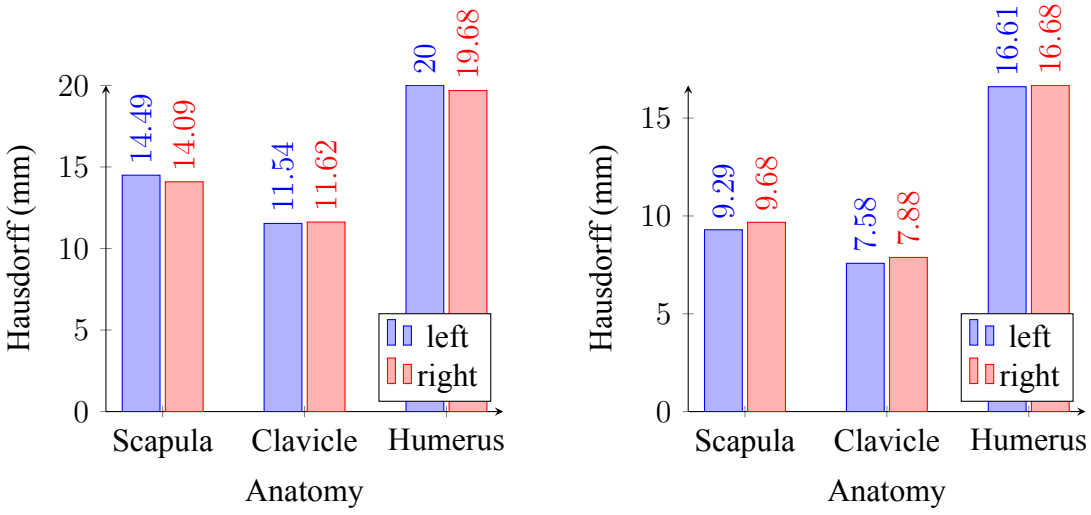


Fig. 33: DSC results: Test person 2 (more is better)



(a) HD maximum results: Test person 2

(b) HD average: Test person 2

Fig. 34: Hausdorff distance (HD) results: Test person 2 (less is better)

4.3 Test person 3

Figure 35 serves as a depiction of the segmentation results compared with the Dice similarity coefficient (DSC) for test participant 3. The segmentation overlap shows an uptick in seg-

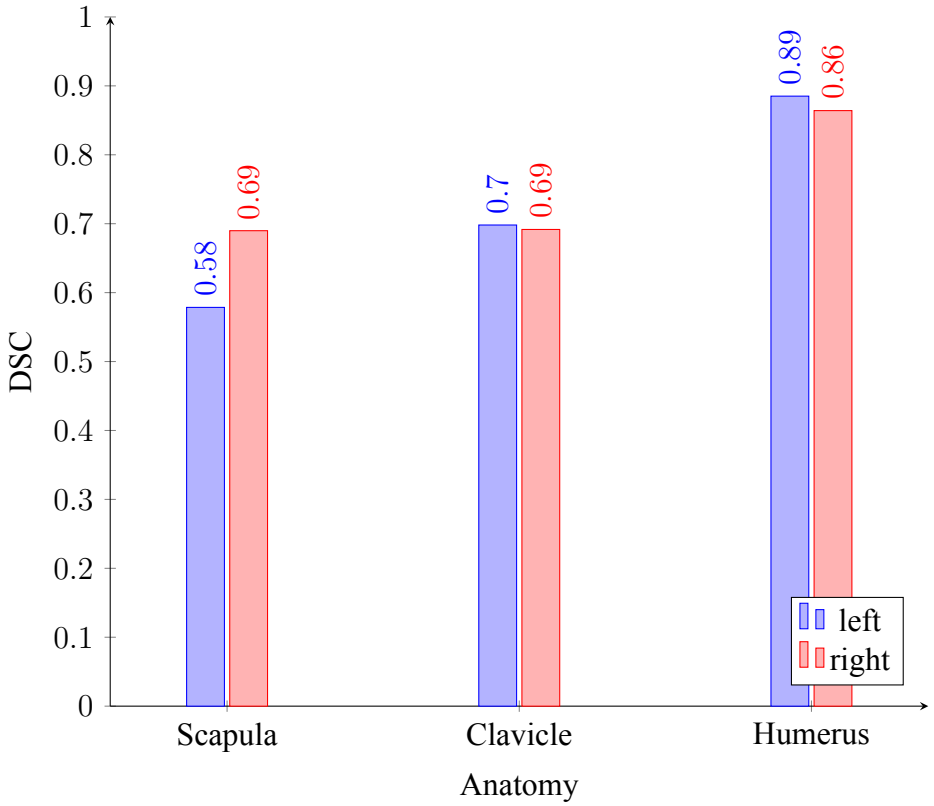
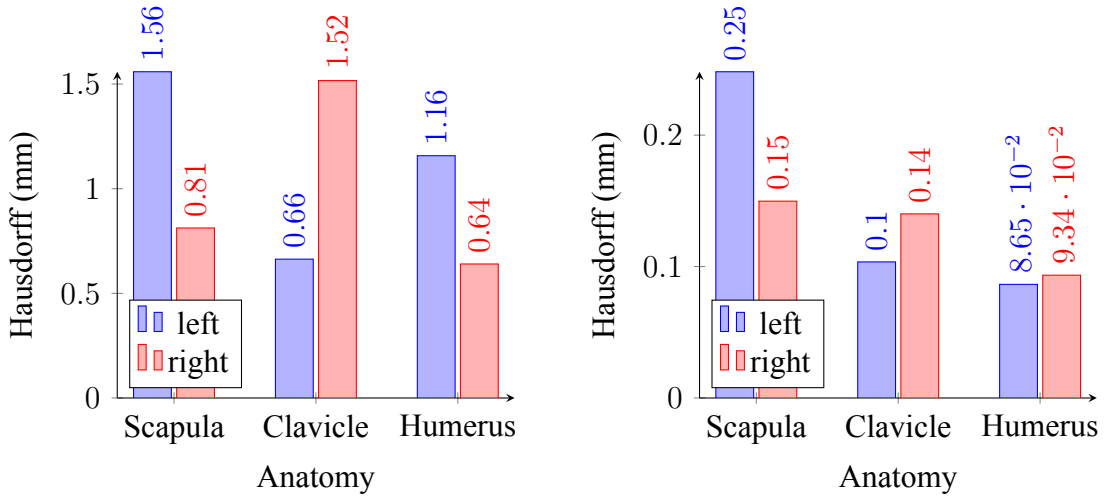


Fig. 35: DSC results: Test person 3 (more is better)



(a) HD maximum results: Test person 3

(b) HD average: Test person 3

Fig. 36: Hausdorff distance (HD) results: Test person 3 (less is better)

mentation accuracy for the scapula ($+0.11$) and a decrease for the clavicle (-0.01) as well as the humerus (-0.03). Figure 36 illustrates the segmentation results for test participant 3 compared via the Hausdorff distance (HD). Whereby Figure 36a displays the maximum distance in millimeters (mm) and Figure 36b displays the average distance in millimeters. As

can be seen, in Figure 36b, the segmentation of test person 3 on average was actually closer to the ground truth for the scapula ($-0.1mm$) and further from it for clavicle ($+0.04mm$) and humerus ($+0.0069mm$).

4.4 Test person 4

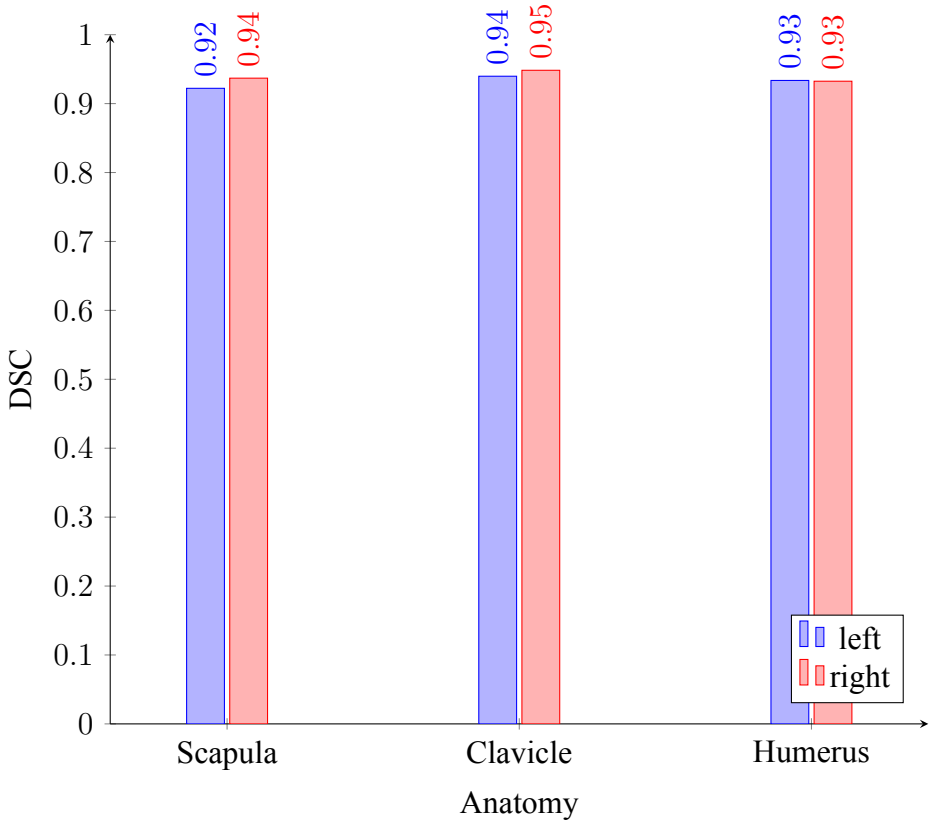
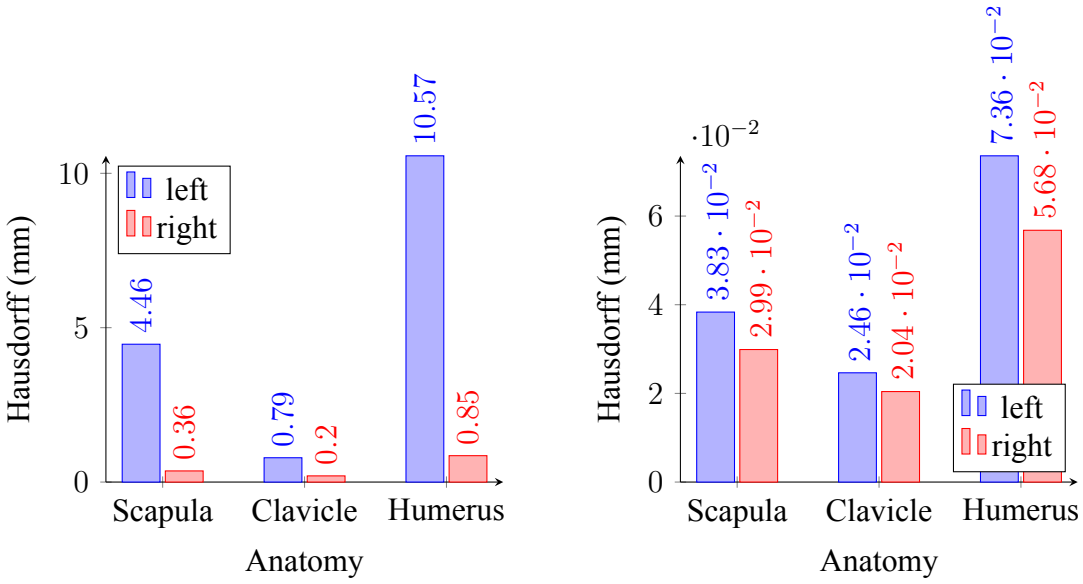


Fig. 37: DSC results: Test person 4 (more is better)

The segmentation comparison results using the Dice similarity coefficient (DSC) for test person 4 are shown in Figure 37. Two of the three segmentations, scapula ($+0.02$) and clavicle ($+0.01$), show an improvement in segmentation accuracy. Whereas, the humerus exhibits no change in segmentation overlap as determined by the DSC. The segmentation comparison results using the Hausdorff distance (HD) for test person 4 are shown in Figure 38. Whereas Figure 38a shows the average distance in millimeters, Figure 39a shows the greatest distance in millimeters (mm). As shown in Figure 38b, test person 2's segmentation was, on average, closer to the ground truth in all cases: scapula ($-0.0084mm$), clavicle ($-0.0042mm$) and humerus ($-0.0168mm$).



(a) HD maximum results: Test person 4

(b) HD average: Test person 4

Fig. 38: Hausdorff distance (HD) results: Test person 4 (less is better)

4.5 Test person 5

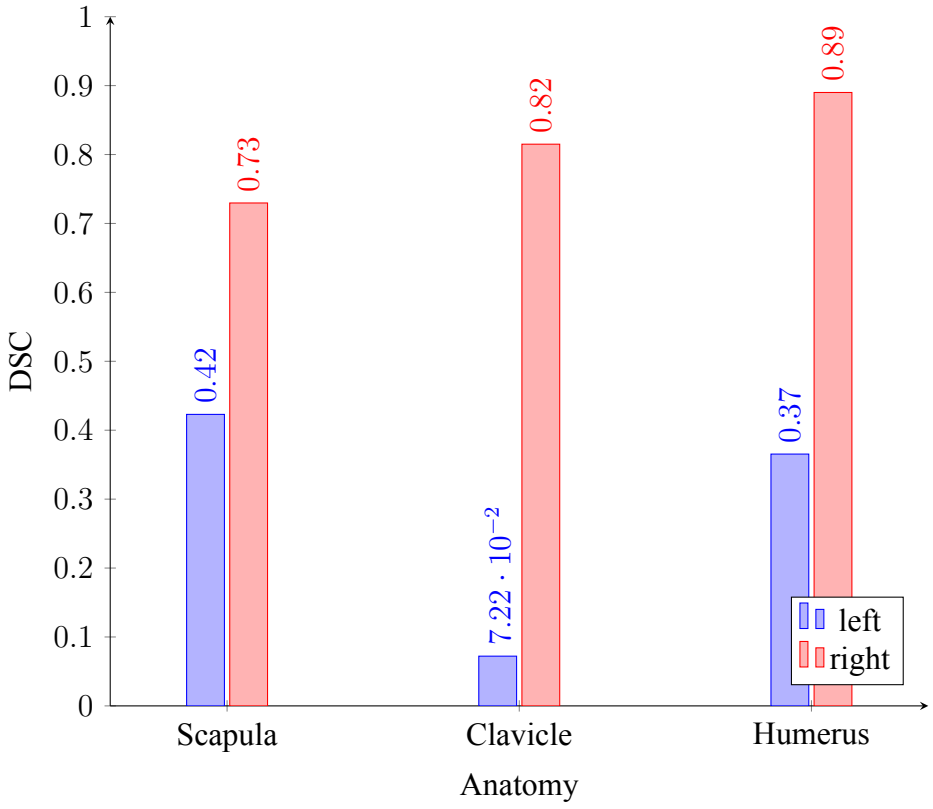
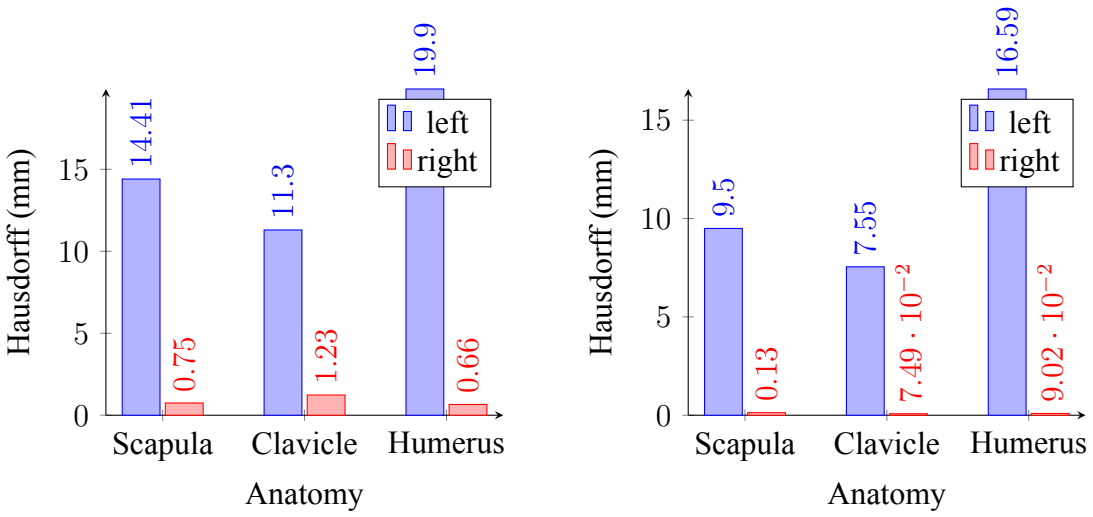


Fig. 39: DSC results: Test person 5 (more is better)



(a) HD maximum results: Test person 5

(b) HD average: Test person 5

Fig. 40: Hausdorff distance (HD) results: Test person 5 (less is better)

Figure 39 illustrates the segmentation comparison results using the Dice similarity coefficient (DSC) for test person 5. The DSC's segmentation overlap indicates that all segmented anatomical structures—the scapula ($+0.31$), clavicle ($+0.7478$), and humerus ($+0.52$)—have improved segmentation accuracy. The segmentation comparison results for test participant 5 using the Hausdorff distance (HD) are shown in Figure 40. Whereas Figure 40a displays the maximum distance in millimeters (mm) and Figure 40b displays the average distance in millimeters. Figure 40b shows that, on average, test person 5's segmentation of the scapula ($-9.37mm$), clavicle ($-7.4751mm$), and humerus ($-16.4959mm$) was closer to the ground truth in every instance.

4.6 Test person 6

The segmentation comparison results using the Dice similarity coefficient (DSC) for test person 6 are shown in Figure 41. The clavicle ($+0.14$), humerus ($+0.08$), and scapula ($+0.09$) all exhibit improved segmentation accuracy, according to the DSC's segmentation overlap. The segmentation comparison results using the Hausdorff distance (HD) for test person 6 are shown in Figure 42. Whereas Figure 42b shows the average distance in millimeters, Figure 42a shows the greatest distance in millimeters (mm). Test person 6's segmentation, as shown in Figure 42b, was generally closer to the ground truth in all cases: scapula ($-0.0365mm$), clavicle ($-0.0615mm$), and humerus ($-0.0528mm$).

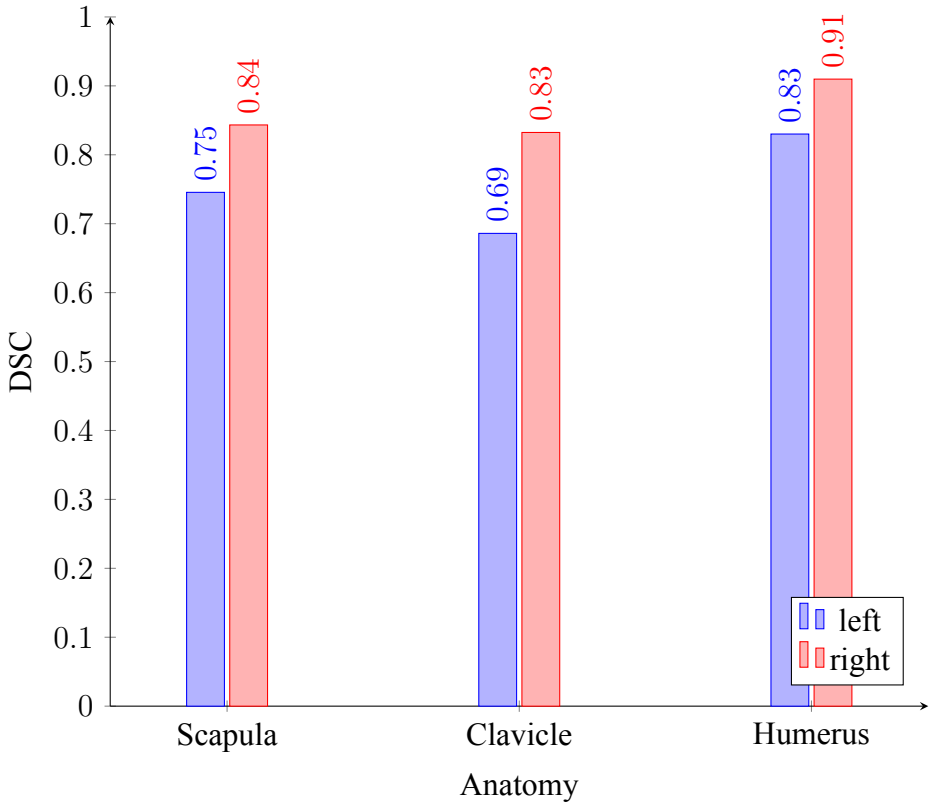
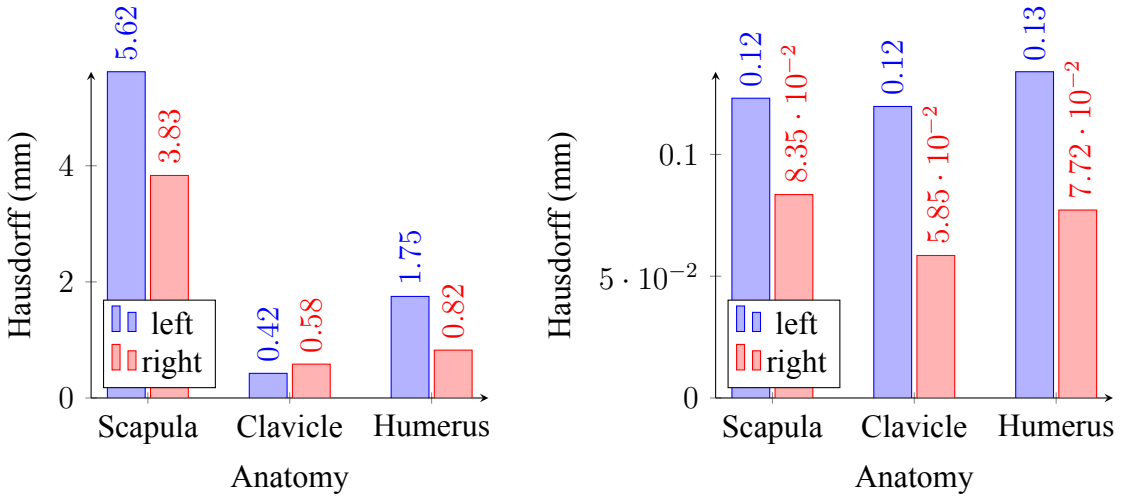


Fig. 41: DSC results: Test person 6 (more is better)



(a) HD maximum results: Test person 6

(b) HD average: Test person 6

Fig. 42: Hausdorff distance (HD) results: Test person 6 (less is better)

4.7 Test person 7

Figure 43 depicts the segmentation comparison results via the Dice similarity coefficient (DSC) for test person 7. The segmentation overlap according to the DSC shows an increase

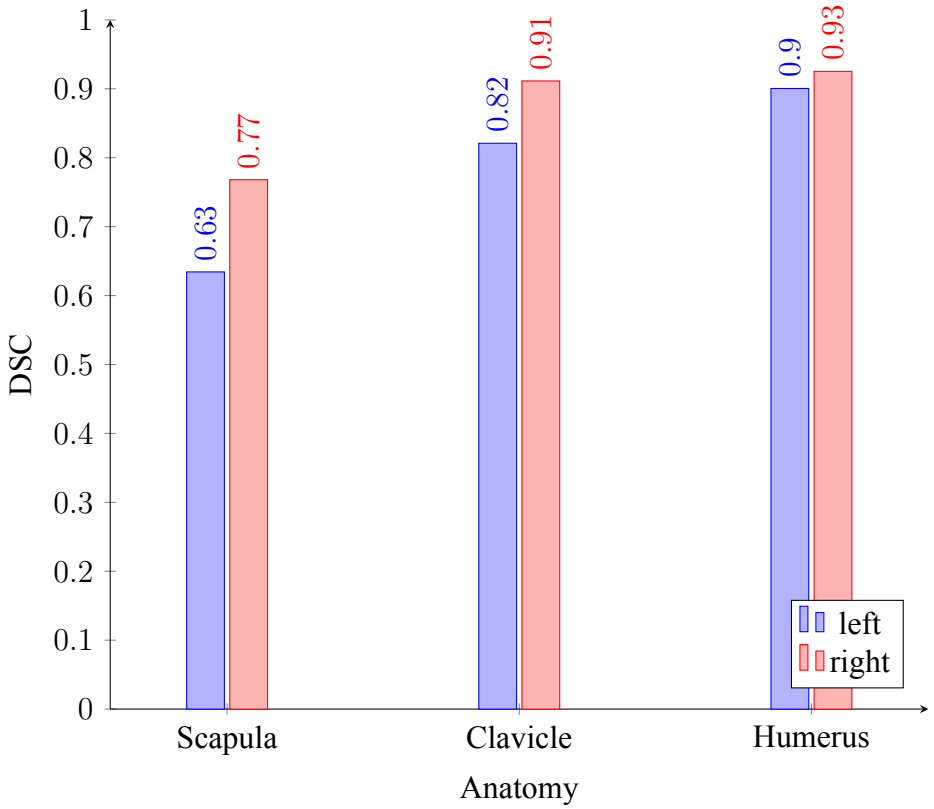
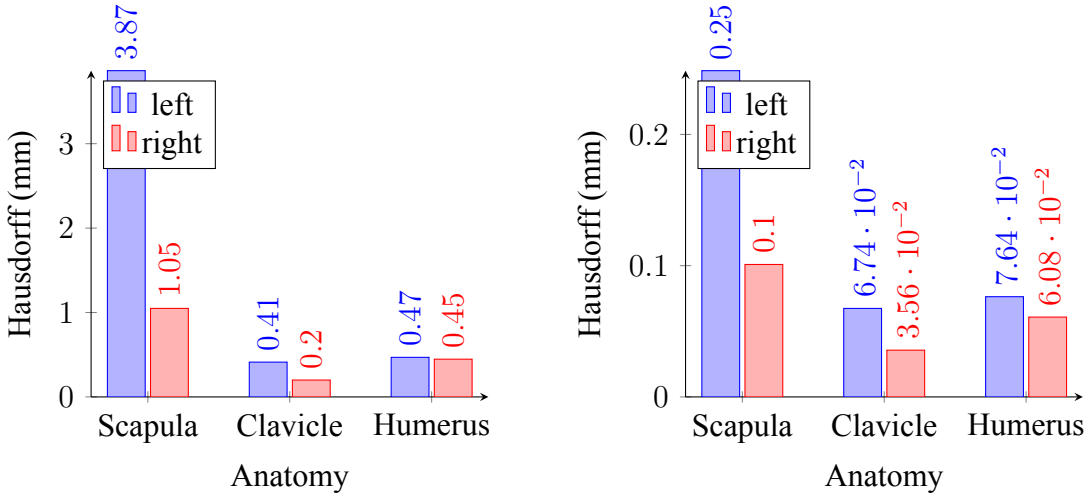


Fig. 43: DSC results: Test person 7 (more is better)



(a) HD maximum results: Test person 7

(b) HD average: Test person 7

Fig. 44: Hausdorff distance (HD) results: Test person 7 (less is better)

in segmentation accuracy in all segmented anatomical structures: scapula (+0.14), clavicle (+0.09) and humerus (+0.03). Figure 44 depicts the segmentation comparison results via the Hausdorff distance (HD) for test person 7. Whereby Figure 44a displays the maximum distance in millimeters (mm) and Figure 44b displays the average distance in millimeters. As can

be seen, in Figure 44b, the segmentation of test person 7 on average was closer to the ground truth in all cases: scapula ($-0.15mm$), clavicle ($-0.0318mm$) and humerus ($-0.0156mm$).

5. Discussion

This chapter provides an overview of the relevance and interpretation of the results in Chapter 4. However, due to the fact that this study has a small number of participants (n=7), it has to be noted that a statistical significance of the results has not been achieved. Similar to Andersen, Darvann, Murakami, *et al.*, segmentation accuracy will be interpreted via overlap using the Dice similarity coefficient (DSC), where 90% overlap will be considered as *excellent overlap* [83]. As well as, border distance using the Hausdorff distance (HD), where $0.4mm$[1] will be considered as *acceptable agreement*.
Furthermore, Chapter 6 provides a final conclusion of the topics covered by this thesis and discusses possible future research.

The following sections (Section 5.1 - Section 5.3) feature tables (Table 4 - Table 9), which provide an overview of segmentation improvements for each test person, separated by segmented anatomy and comparison metric. Where the values shown in the tables reflect the difference of their segmentation accuracy between pre- and post-guide performance.

5.1 Segmentation evaluation: Scapula

As can be seen in Table 4, segmentation of the scapula with the guide improved in six out of seven testers. The mean values of the segmentation overlap between the ground truth and the test person's segmentation increased by 13.57%. Thus, six out of seven testers appear to derive benefit from reading the guide when tasked with segmenting a mouse scapula when comparing their performance using the Dice similarity coefficient.

Test person	**overlap (%)**
1	-17
2	45
3	11
4	2
5	31
6	9
7	14
Average	13.57

Tab. 4: Evaluation: DSC scapula

[1] 4 times the voxel size of $0.1mm$

Five out of seven testers showed improved results when the scapula was segmented using the guide, as shown Table 5. Between the test persons segmentation and the ground truth, the average distance between the segmentation borders dropped by $1.3085mm$. Consequently, it appears that reading the guide helped five out of seven participants with the mouse scapula segmentation task.

Test person	**distance (mm)**
1	0.1155
2	0.39
3	-0.1
4	-0.0084
5	-9.37
6	-0.0365
7	-0.15
Average	-1.3085

Tab. 5: Evaluation: HD scapula

5.2 Segmentation evaluation: Clavicle

Depicted in Table 6 is the clavicle segmentation improvement in five out of seven participants. On average, the segmentation overlap between the ground truth and test person's segmentation increased by 17.82%. To conclude, five out of seven testers seem to have benefited from reading the guide when tasked with segmenting a mouse clavicle, as compared to their performance using the Dice similarity coefficient.

Test person	**overlap (%)**
1	-5
2	32
3	-1
4	1
5	74.78
6	14
7	9
Average	17.82

Tab. 6: Evaluation: DSC clavicle

Table 7 shows that segmentation of the clavicle with the guide improved in four out of seven testers. The average distance between segmentation borders between the ground truth and the test person segmentation decreased by $1.029mm$. This entails that four out of seven testers benefited from reading the guide when tasked with segmenting a mouse clavicle, as shown by their results using the Hausdorff distance as a metric.

Test person	distance (mm)
1	0.027
2	0.3
3	0.04
4	−0.0042
5	−7.4751
6	−0.0615
7	−0.0318
Average	−1.02937

Tab. 7: Evaluation: HD clavicle

5.3 Segmentation evaluation: Humerus

As can be seen in Table 8 segmentation of the humerus with the guide improved in four out of seven testers. The mean value of the segmentation overlap between the ground truth and the test person's segmentation increased by 17.71%. Thus, four out of seven testers appeared to have benefited from reading the guide when tasked with segmenting a mouse humerus when comparing their performance using the Dice similarity coefficient.

Test person	overlap (%)
1	−5
2	69
3	−3
4	0
5	52
6	8
7	3
Average	17.71

Tab. 8: Evaluation: DSC humerus

As can be seen in Table 9 segmentation of the humerus with the guide improved in five out of seven testers. On average the distance between segmentation borders between the ground truth and the test person segmentation decreased by $2.382mm$. Therefore, five out of seven testers seem to have benefited from reading the guide when tasked with segmenting a mouse humerus when comparing their performance using the Hausdorff distance.

Test person	distance (mm)
1	-0.17
2	0.07
3	0.0069
4	-0.0168
5	-16.4959
6	-0.0528
7	-0.0156
Average	-2.382

Tab. 9: Evaluation: HD humerus

5.4 Interpretation and comparison with Literature

In their study, Andersen, Darvann, Murakami, *et al.* compared manual segmentation between MRI and CT scans of the same patients performed by different operators and defined the results via the Dice similarity coefficient (DSC) and Hausdorff distance (HD) [83]. A DSC value of 0.9 or 90%, was defined as *excellent agreement* by the researchers. Moreover, they defined a border distance (HD) equal to four times the acquired voxel size as *acceptable agreement* [83]. In this study the author introduces the term of *acceptable agreement* for the DSC for an overlap of 80%.
As indicated in Chapter 4, *excellent agreement* after reading the guide, according to the Dice similarity coefficient was achieved by several testers. Test person 4 for all three anatomical structures, test person 6 for humerus segmentation and test person 7 for clavicle and humerus segmentation. Accordingly, the highest level of agreement after reading the guide was attained for humerus segmentation by 42.86% of testers (see: Table 10).

Anatomical structure	testers in agreement (total)	testers in agreement (%)
scapula	0/7	0
clavicle	1/7	14.29
humerus	3/7	42.86

Tab. 10: *Excellent agreement: DSC*

Acceptable agreement after reading the guide, according to the DSC was achieved by all testers for the humerus, testers 4, 5, 6, and 7 for the clavicle and testers 4 and 6 for the scapula. Consequently, the highest level of consensus after reading the guide was achieved for humerus segmentation with 100% of testers achieving full agreement (see: Table 11). The second-highest level of agreement was achieved for clavicle segmentation by 57.15% of testers.

Anatomical structure	agreement (total)	agreement (%)
scapula	2/7	28.57
clavicle	4/7	57.15
humerus	7/7	100

Tab. 11: *Acceptable agreement: DSC*

Acceptable agreement after reading the guide, according to the HD was achieved for all anatomical structures by six out of seven testers (see: Table 12). Test person 2 exhibited a discrepancy from the rest of the group. Consequently, the majority of testers demonstrated a high degree of agreement following the reading of the guide for all segmented structures.

Anatomical structure	agreement (total)	agreement (%)
scapula	6/7	85.71
clavicle	6/7	85.71
humerus	6/7	85.71

Tab. 12: *Acceptable agreement: HD*

6. Conclusion and Limitations

The primary objective of this thesis was to investigate whether individuals with limited experience in micro-CT segmentation tasks may derive benefit from the use of a guide or reference material prior to undertaking the task. This is a relevant topic of investigation because the process of segmentation represents a crucial step in the context of quantitative image analysis studies. Providing researchers with a head start in segmentation tasks is likely to result in improved segmentation speed and accuracy. In this study, a guide was created which describes the process of segmenting mouse bone anatomy in micro-CT datasets. The guide was distributed to the testers with the task of performing segmentation without any help from the guide on 3 bones. Subsequently, the task was repeated on the contralateral side with the aid of the guide. The resulting segmentations were then compared to a gold standard segmentation and analyzed. The limited sample size did not allow for a statistically significant demonstration of improvement in the segmentations. However, the results suggest that a larger number of testers would likely show more notable enhancements in segmentation accuracy. In conclusion, the present study was unable to demonstrate statistically significant improvements due to the limited sample size. Nevertheless, future research could provide a more definitive answer to the research question of whether the use of a segmentation guide improves segmentation quality when employed by individuals with limited experience.

Future Research

This thesis provides a basis for future research in the area of medical imaging segmentation documentation and guidance. Further points of research can be inferred from the limitations described in Chapter 6.

First and foremost, as this paper only managed to gather results from a limited number[1] of testers, the challenge with participant numbers needs to be addressed in follow-up research. This could be solved via the following steps. Firstly, it would be beneficial to allocate more time for testers to complete the task. Secondly, it would be beneficial to engage a greater number of testers from the outset. Secondly, the segmentation performance of different guide structures may be evaluated by strategic *A/B testing* as described by Brata and Brata [84]. Whereby, testers are divided into two groups without their knowledge. Version A of a guide is handed out to group A and version B of a guide is handed out to group B. By comparing the performance of the two groups, the success rate of different guide formats can be indirectly evaluated [84].

[1] n=7

Furthermore, to ensure a consistent bone mask across all testing participants a fixed HU threshold range should be defined [85].
Therefore, additional research is required to answer the research question of whether inexperienced users benefit from a segmentation guide.

List of Figures

List of Tables

Bibliography

[1] T. Perciano, D. Ushizima, H. Krishnan, D. Parkinson, N. Larson, D. M. Pelt, W. Bethel, F. Zok, and J. Sethian, "Insight into 3D micro-CT data: Exploring segmentation algorithms through performance metrics," *Journal of Synchrotron Radiation*, vol. 24, no. 5, pp. 1065–1077, Sep. 1, 2017, issn: 1600-5775. doi: 10.1107/S1600577517010955. [Online]. Available: https://scripts.iucr.org/cgi-bin/paper?S1600577517010955 (visited on 02/05/2024).

[2] A. Sheppard, S. Latham, J. Middleton, A. Kingston, G. Myers, T. Varslot, A. Fogden, T. Sawkins, R. Cruikshank, M. Saadatfar, N. Francois, C. Arns, and T. Senden, "Techniques in helical scanning, dynamic imaging and image segmentation for improved quantitative analysis with X-ray micro-CT," *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, vol. 324, pp. 49–56, Apr. 2014, issn: 0168583X. doi: 10.1016/j.nimb.2013.08.072. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0168583X14000457 (visited on 02/05/2024).

[3] V. C. Korfiatis, S. Tassani, and G. K. Matsopoulos, "An Independent Active Contours Segmentation framework for bone micro-CT images," *Computers in Biology and Medicine*, vol. 87, pp. 358–370, Aug. 2017, issn: 00104825. doi: 10.1016/j.compbiomed.2017.06.016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0010482517301804 (visited on 02/08/2024).

[4] A. Virzì, C. O. Muller, J.-B. Marret, E. Mille, L. Berteloot, D. Grévent, N. Boddaert, P. Gori, S. Sarnacki, and I. Bloch, "Comprehensive Review of 3D Segmentation Software Tools for MRI Usable for Pelvic Surgery Planning," *Journal of Digital Imaging*, vol. 33, no. 1, pp. 99–110, Feb. 2020. doi: 10.1007/s10278-019-00239-7. [Online]. Available: http://link.springer.com/10.1007/s10278-019-00239-7 (visited on 02/05/2024).

[5] M. Mandolini, A. Brunzini, G. Facco, A. Mazzoli, A. Forcellese, and A. Gigante, "Comparison of Three 3D Segmentation Software Tools for Hip Surgical Planning," *Sensors*, vol. 22, no. 14, p. 5242, Jul. 13, 2022, issn: 1424-8220. doi: 10.3390/s22145242. [Online]. Available: https://www.mdpi.com/1424-8220/22/14/5242 (visited on 02/05/2024).

[6] P. A. Yushkevich, J. Piven, H. C. Hazlett, R. G. Smith, S. Ho, J. C. Gee, and G. Gerig, "User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability," *NeuroImage*, vol. 31, no. 3, pp. 1116–1128, Jul.

2006, issn: 10538119. doi: 10.1016/j.neuroimage.2006.01.015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1053811906000632 (visited on 02/05/2024).

[7] G. G. P. License, "Gnu general public license," *Retrieved December*, vol. 25, p. 2014, 1989. [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html#license-text (visited on 02/07/2024).

[8] R. Kikinis, S. D. Pieper, and K. G. Vosburgh, "3D Slicer: A Platform for Subject-Specific Image Analysis, Visualization, and Clinical Support," in *Intraoperative Imaging and Image-Guided Therapy*, F. A. Jolesz, Ed., New York, NY: Springer New York, 2014, pp. 277–289, isbn: 978-1-4614-7656-6. doi: 10.1007/978-1-4614-7657-3_19. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-7657-3_19 (visited on 09/17/2023).

[9] A. Gaudeul, "Public Provision of a Private Good: What is the Point of the BSD License?" *SSRN Electronic Journal*, 2005, issn: 1556-5068. doi: 10.2139/ssrn.933631. [Online]. Available: http://www.ssrn.com/abstract=933631 (visited on 02/06/2024).

[10] H. Aydin, M. M. Sofu, A. Salkaci, S. Evrimler, and A. Karaibrahimoglu, "RELIABILITY, REPRODUCIBILITY AND TIME-EFFICIENCY OF VOLUMETRIC EVALUATION OF LUNG TUMORS BY 3D-SLICER SEGMENTATION WIZARD," *MEDICAL DIAGNOSIS*, p. 105, 2020.

[11] D. Argüello, H. G. Sánchez Acevedo, and O. A. González-Estrada, "Comparison of segmentation tools for structural analysis of bone tissues by finite elements," *Journal of Physics: Conference Series*, vol. 1386, no. 1, p. 012 113, Nov. 1, 2019, issn: 1742-6588. doi: 10.1088/1742-6596/1386/1/012113. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1386/1/012113 (visited on 10/21/2023).

[12] C. Pinter, A. Lasso, and G. Fichtinger, "Polymorph segmentation representation for medical image computing," *Computer Methods and Programs in Biomedicine*, vol. 171, pp. 19–26, Apr. 2019, issn: 01692607. doi: 10.1016/j.cmpb.2019.02.011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169260718313038 (visited on 02/07/2024).

[13] Slicer Community. "3D Slicer image computing platform." (Nov. 22, 2022), [Online]. Available: https://www.slicer.org/ (visited on 09/17/2023).

[14] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.-C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, J. Buatti, S. Aylward, J. V. Miller, S. Pieper, and R. Kikinis, "3D Slicer as an image computing platform for the Quantitative

Imaging Network," *Magnetic Resonance Imaging*, vol. 30, no. 9, pp. 1323–1341, Nov. 2012, issn: 0730725X. doi: 10.1016/j.mri.2012.05.001. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0730725X12001816 (visited on 02/08/2024).

[15] Y. Liu, J. Zhang, Z. She, A. Kheradmand, and M. Armand, "SAMM (Segment Any Medical Model): A 3D Slicer Integration to SAM," 2023. doi: 10.48550/ARXIV.2304.05622. [Online]. Available: https://arxiv.org/abs/2304.05622 (visited on 10/21/2023).

[16] L. R. Dice, "Measures of the Amount of Ecologic Association Between Species," *Ecology*, vol. 26, no. 3, pp. 297–302, Jul. 1945, issn: 0012-9658. doi: 10.2307/1932409. [Online]. Available: https://esajournals.onlinelibrary.wiley.com/doi/10.2307/1932409 (visited on 02/06/2024).

[17] F. Hausdorff, *Grundzüge der Mengenlehre*, Repr. Berlin 1914. New York: Chelsea, 1978, isbn: 978-0-8284-0061-9.

[18] C. Pinter, A. Lasso, A. Wang, D. Jaffray, and G. Fichtinger, "SlicerRT: Radiation therapy research toolkit for 3D Slicer," *Medical Physics*, vol. 39, no. 10, pp. 6332–6338, Oct. 2012, issn: 0094-2405. doi: 10.1118/1.4754659. [Online]. Available: https://aapm.onlinelibrary.wiley.com/doi/10.1118/1.4754659 (visited on 10/24/2023).

[19] D. Clark and C. Badea, "Advances in micro-CT imaging of small animals," *Physica Medica*, vol. 88, pp. 175–192, Aug. 2021, issn: 11201797. doi: 10.1016/j.ejmp.2021.07.005. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1120179721002490 (visited on 09/21/2024).

[20] E. L. Ritman, "Current Status of Developments and Applications of Micro-CT," *Annual Review of Biomedical Engineering*, vol. 13, no. 1, pp. 531–552, Aug. 15, 2011, issn: 1523-9829. doi: 10.1146/annurev-bioeng-071910-124717. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev-bioeng-071910-124717 (visited on 09/21/2024).

[21] K. Keklikoglou, C. Arvanitidis, G. Chatzigeorgiou, E. Chatzinikolaou, E. Karagiannidis, T. Koletsa, A. Magoulas, K. Makris, G. Mavrothalassitis, E.-D. Papanagnou, A. S. Papazoglou, C. Pavloudi, I. P. Trougakos, K. Vasileiadou, and A. Vogiatzi, "Micro-CT for Biological and Biomedical Studies: A Comparison of Imaging Techniques," *Journal of Imaging*, vol. 7, no. 9, p. 172, Sep. 1, 2021, issn: 2313-433X. doi: 10.3390/jimaging7090172. [Online]. Available: https://www.mdpi.com/2313-433X/7/9/172 (visited on 09/21/2024).

[22] T. Flohr, "CT Systems," *Current Radiology Reports*, vol. 1, no. 1, pp. 52–63, Mar. 2013, issn: 2167-4825. doi: 10.1007/s40134-012-0005-5. [Online]. Available: http://link.springer.com/10.1007/s40134-012-0005-5 (visited on 09/21/2024).

[23] FDA, *Drawing of CT fan beam and patient in a CT imaging system*, Jun. 14, 2019. [Online]. Available: https://www.fda.gov/radiation-emitting-products/medical-x-ray-imaging/computed-tomography-ct (visited on 09/21/2024).

[24] R. Baba, Y. Konno, K. Ueda, and S. Ikeda, "Comparison of flat-panel detector and image-intensifier detector for cone-beam CT," *Computerized Medical Imaging and Graphics*, vol. 26, no. 3, pp. 153–158, May 2002, issn: 08956111. doi: 10.1016/S0895-6111(02)00008-3. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0895611102000083 (visited on 09/21/2024).

[25] K. Orhan, *Micro-Computed Tomography (Micro-CT) in Medicine and Engineering*. Cham: Springer International Publishing, 2020, isbn: 978-3-030-16640-3. doi: 10.1007/978-3-030-16641-0. [Online]. Available: https://link.springer.com/10.1007/978-3-030-16641-0 (visited on 09/22/2024).

[26] D. Clark and C. Badea, "Micro-CT of rodents: State-of-the-art and future perspectives," *Physica Medica*, vol. 30, no. 6, pp. 619–634, Sep. 2014, issn: 11201797. doi: 10.1016/j.ejmp.2014.05.011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1120179714001008 (visited on 09/21/2024).

[27] H. Li, H. Zhang, Z. Tang, and G. Hu, "Micro-computed tomography for small animal imaging: Technological details," *Progress in Natural Science*, vol. 18, no. 5, pp. 513–521, May 2008, issn: 10020071. doi: 10.1016/j.pnsc.2008.01.002. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1002007108000658 (visited on 09/22/2024).

[28] B. Vandeghinste, S. Vandenberghe, C. Vanhove, S. Staelens, and R. Van Holen, "Low-Dose Micro-CT Imaging for Vascular Segmentation and Analysis Using Sparse-View Acquisitions," *PLoS ONE*, vol. 8, no. 7, A. Muñoz-Barrutia, Ed., e68449, Jul. 1, 2013, issn: 1932-6203. doi: 10.1371/journal.pone.0068449. [Online]. Available: https://dx.plos.org/10.1371/journal.pone.0068449 (visited on 09/19/2024).

[29] H. Jung, "Basic Physical Principles and Clinical Applications of Computed Tomography," *Progress in Medical Physics*, vol. 32, no. 1, pp. 1–17, Mar. 31, 2021, issn: 2508-4445, 2508-4453. doi: 10.14316/pmp.2021.32.1.1. [Online]. Available: http://www.progmedphys.org/journal/view.html?doi=10.14316/pmp.2021.32.1.1 (visited on 09/27/2024).

[30] C. Shorter, R. Macklin, A. Rock, F. Brand, R. Murugesan, and T. Luedtke, *The Good Docs Project*, version v1.2.0, The Good Docs Project (TGDP), Jun. 13, 2024. [Online]. Available: https://thegooddocsproject.dev/ (visited on 09/24/2024).

[31] D. Procida. "Diátaxis documentation framework," Diataxis. (Dec. 2023), [Online]. Available: https://diataxis.fr/ (visited on 09/08/2024).

[32] E. Aghajani, C. Nagy, M. Linares-Vásquez, L. Moreno, G. Bavota, M. Lanza, and D. C. Shepherd, "Software documentation: The practitioners' perspective," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, Seoul South Korea: ACM, Jun. 27, 2020, pp. 590–601, isbn: 978-1-4503-7121-6. doi: 10.1145/3377811.3380405. [Online]. Available: https://dl.acm.org/doi/10.1145/3377811.3380405 (visited on 08/04/2024).

[33] C. R. Harris, K. J. Millman, S. J. Van Der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. Van Kerkwijk, M. Brett, A. Haldane, J. F. Del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 17, 2020, issn: 1476-4687. doi: 10.1038/s41586-020-2649-2. [Online]. Available: https://www.nature.com/articles/s41586-020-2649-2 (visited on 09/21/2024).

[34] *Django*, version 1.5, Lawrence, Kansas: Django Software Foundation, 2013. [Online]. Available: https://www.djangoproject.com/ (visited on 09/21/2024).

[35] D. Procida, "Python Docs Community Workshop: Introduction to Diataxis," Sep. 27, 2022. [Online]. Available: https://www.youtube.com/watch?v=710uQqIqsWk (visited on 09/13/2024).

[36] D. Calavera and L. Fontana, *Linux Observability with BPF: Advanced Programming for Performance Analysis and Networking*, First edition. Beijing [China] ; Sebastopol, CA: O'Reilly Media, Inc, 2019, 162 pp., isbn: 978-1-4920-5020-9.

[37] R. Dowse, "Pharmacists, are words enough? The case for pictograms as a valuable communication tool," *Research in Social and Administrative Pharmacy*, vol. 17, no. 8, pp. 1518–1522, Aug. 2021, issn: 15517411. doi: 10.1016/j.sapharm.2020.10.013. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S155174112031144X (visited on 09/19/2024).

[38] P. S. Houts, C. C. Doak, L. G. Doak, and M. J. Loscalzo, "The role of pictures in improving health communication: A review of research on attention, comprehension, recall, and adherence," *Patient Education and Counseling*, vol. 61, no. 2, pp. 173–190, May 2006, issn: 07383991. doi: 10.1016/j.pec.2005.05.004. [Online]. Available:

https://linkinghub.elsevier.com/retrieve/pii/S0738399105001461 (visited on 09/19/2024).

[39] L. E. Mansoor and R. Dowse, "Effect of Pictograms on Readability of Patient Information Materials," *Annals of Pharmacotherapy*, vol. 37, no. 7-8, pp. 1003–1009, Jul. 2003, issn: 1060-0280. doi: 10.1345/aph.1C449. [Online]. Available: http://journals.sagepub.com/doi/10.1345/aph.1C449 (visited on 09/19/2024).

[40] I. Qureshi, J. Yan, Q. Abbas, K. Shaheed, A. B. Riaz, A. Wahid, M. W. J. Khan, and P. Szczuko, "Medical image segmentation using deep semantic-based methods: A review of techniques, applications and emerging trends," *Information Fusion*, vol. 90, pp. 316–352, Feb. 2023, issn: 15662535. doi: 10.1016/j.inffus.2022.09.031. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1566253522001695 (visited on 11/09/2024).

[41] S. Hao, Y. Zhou, and Y. Guo, "A Brief Survey on Semantic Segmentation with Deep Learning," *Neurocomputing*, vol. 406, pp. 302–321, Sep. 2020, issn: 09252312. doi: 10.1016/j.neucom.2019.11.118. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231220305476 (visited on 09/30/2024).

[42] J. Long, E. Shelhamer, and T. Darrell. "Fully Convolutional Networks for Semantic Segmentation." arXiv: 1411.4038 [cs]. (Mar. 8, 2015), [Online]. Available: http://arxiv.org/abs/1411.4038 (visited on 09/30/2024), pre-published.

[43] L.-C. Chen, A. Hermans, G. Papandreou, F. Schroff, P. Wang, and H. Adam, "MaskLab: Instance Segmentation by Refining Object Detection with Semantic and Direction Features," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT: IEEE, Jun. 2018, pp. 4013–4022, isbn: 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00422. [Online]. Available: https://ieeexplore.ieee.org/document/8578520/ (visited on 09/30/2024).

[44] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: State of the art," *International Journal of Multimedia Information Retrieval*, vol. 9, no. 3, pp. 171–189, Sep. 2020. doi: 10.1007/s13735-020-00195-x. [Online]. Available: https://link.springer.com/10.1007/s13735-020-00195-x (visited on 09/30/2024).

[45] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. "Panoptic Segmentation." arXiv: 1801.00868 [cs]. (Apr. 10, 2019), [Online]. Available: http://arxiv.org/abs/1801.00868 (visited on 09/23/2024), pre-published.

[46] L. Grady and G. Funka-Lea, "An Energy Minimization Approach to the Data Driven Editing of Presegmented Images/Volumes," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2006*, R. Larsen, M. Nielsen, and J. Sporring, Eds., red. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell,

M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, and G. Weikum, vol. 4191, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 888–895, isbn: 978-3-540-44727-6. doi: 10.1007/11866763_109. [Online]. Available: http://link.springer.com/10.1007/11866763_109 (visited on 10/06/2024).

[47] Y. Yu, C. Wang, Q. Fu, R. Kou, F. Huang, B. Yang, T. Yang, and M. Gao, "Techniques and Challenges of Image Segmentation: A Review," *Electronics*, vol. 12, no. 5, p. 1199, Mar. 2, 2023, issn: 2079-9292. doi: 10.3390/electronics12051199. [Online]. Available: https://www.mdpi.com/2079-9292/12/5/1199 (visited on 04/17/2024).

[48] V7labs. "An Introduction to Image Segmentation: Deep Learning vs. Traditional [+Examples]," V7labs. (Dec. 7, 2021), [Online]. Available: https://www.v7labs.com/blog/image-segmentation-guide#h4 (visited on 10/01/2024).

[49] A. Bali and S. N. Singh, "A Review on the Strategies and Techniques of Image Segmentation," in *2015 Fifth International Conference on Advanced Computing & Communication Technologies*, Haryana, India: IEEE, Feb. 2015, pp. 113–120, isbn: 978-1-4799-8488-6. doi: 10.1109/ACCT.2015.63. [Online]. Available: http://ieeexplore.ieee.org/document/7079063/ (visited on 09/29/2024).

[50] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3. ed. Upper Saddle River, NJ: Pearson/Prentice Hall, 2007, isbn: 978-0-13-168728-8.

[51] K. Pandya. "Exploring Image Segmentation with the Region Growing Algorithm," Medium. (Apr. 19, 2024), [Online]. Available: https://medium.com/@khushihp7903/exploring-image-segmentation-with-the-region-growing-algorithm-4972dae63680 (visited on 10/04/2024).

[52] M. Jourlin, "Various Contrast Concepts," in *Advances in Imaging and Electron Physics*, vol. 195, Elsevier, 2016, pp. 27–60, isbn: 978-0-12-804813-9. doi: 10.1016/bs.aiep.2016.04.002. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1076567016300325 (visited on 10/02/2024).

[53] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277–1294, Sep. 1993, issn: 00313203. doi: 10.1016/0031-3203(93)90135-J. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/003132039390135J (visited on 10/04/2024).

[54] B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, p. 146, Jan. 1, 2004, issn: 1017-9909. doi: 10.1117/1.1631315. [Online]. Available: http://electronicimaging.

spiedigitallibrary.org/article.aspx?doi=10.1117/1.1631315 (visited on 10/02/2024).

[55] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image Segmentation Using K - means Clustering Algorithm and Subtractive Clustering Algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015, issn: 18770509. doi: 10.1016/j.procs.2015.06.090. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877050915014143 (visited on 10/06/2024).

[56] A. A. Shabalin, *K-means clustering. K-means++*. 2024. [Online]. Available: https://shabal.in/visuals/kmeans/1.html (visited on 10/06/2024).

[57] B. Preim and C. Botha, "Image Analysis for Medical Visualization," in *Visual Computing for Medicine*, Elsevier, 2014, pp. 111–175, isbn: 978-0-12-415873-3. doi: 10.1016/B978-0-12-415873-3.00004-3. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780124158733000043 (visited on 10/01/2024).

[58] H. K. Hahn and H.-O. Peitgen, "IWT-interactive watershed transform: A hierarchical method for efficient interactive and automated segmentation of multidimensional gray-scale images," presented at the Medical Imaging 2003, M. Sonka and J. M. Fitzpatrick, Eds., San Diego, CA, May 16, 2003, p. 643. doi: 10.1117/12.481097. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.481097 (visited on 10/06/2024).

[59] N. G. Burnet, "Defining the tumour and target volumes for radiotherapy," *Cancer Imaging*, vol. 4, no. 2, pp. 153–161, 2004, issn: 1470-7330. doi: 10.1102/1470-7330.2004.0054. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1434601/ (visited on 09/25/2024).

[60] S. K. Warfield, C.-F. Westin, C. R. G. Guttmann, M. Albert, F. A. Jolesz, and R. Kikinis, "Fractional Segmentation of White Matter," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI'99*, C. Taylor and A. Colchester, Eds., red. by G. Goos, J. Hartmanis, and J. Van Leeuwen, vol. 1679, Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 62–71, isbn: 978-3-540-66503-8. doi: 10.1007/10704282_7. [Online]. Available: http://link.springer.com/10.1007/10704282_7 (visited on 09/26/2024).

[61] A. Noe and J. C. Gee, "Partial Volume Segmentation of Cerebral MRI Scans with Mixture Model Clustering," in *Information Processing in Medical Imaging*, M. F. Insana and R. M. Leahy, Eds., red. by G. Goos, J. Hartmanis, and J. Van Leeuwen, vol. 2082, Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 423–430, isbn: 978-3-540-45729-9. doi: 10.1007/3-540-45729-1_44. [Online]. Available: http://link.springer.com/10.1007/3-540-45729-1_44 (visited on 09/26/2024).

[62] K. Sunderland, C. Pinter, A. Lasso, and G. Fichtinger, "Fractional labelmaps for computing accurate dose volume histograms," presented at the SPIE Medical Imaging, R. J. Webster and B. Fei, Eds., Orlando, Florida, United States, Mar. 3, 2017, 101352Y. doi: 10.1117/12.2254978. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.2254978 (visited on 09/26/2024).

[63] J. L. Weber, "3D Slicer as a platform for the automatic segmentation of real-time MRI data," Technische Hochschule Köln, 2022, 79 pp. [Online]. Available: https://elib.dlr.de/187800/ (visited on 09/26/2024).

[64] T. Delamé, C. Roudet, and D. Faudot, "From A Medial Surface To A Mesh," *Computer Graphics Forum*, vol. 31, no. 5, pp. 1637–1646, Aug. 2012. doi: 10.1111/j.1467-8659.2012.03169.x. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2012.03169.x (visited on 09/26/2024).

[65] D. Weinstein, "Scanline surfacing: Building separating surfaces from planar contours," in *Proceedings Visualization 2000. VIS 2000 (Cat. No. 00CH37145)*, IEEE, 2000, pp. 283–289, isbn: 0-7803-6478-3. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/885706 (visited on 09/27/2024).

[66] P. Schelb, A. A. Tavakoli, T. Tubtawee, T. Hielscher, J.-P. Radtke, M. Görtz, V. Schütz, T. A. Kuder, L. Schimmöller, A. Stenzinger, M. Hohenfellner, H.-P. Schlemmer, and D. Bonekamp, "Comparison of Prostate MRI Lesion Segmentation Agreement Between Multiple Radiologists and a Fully Automatic Deep Learning System," *RöFo - Fortschritte auf dem Gebiet der Röntgenstrahlen und der bildgebenden Verfahren*, vol. 193, no. 05, pp. 559–573, May 2021, issn: 1438-9029. doi: 10.1055/a-1290-8070. [Online]. Available: http://www.thieme-connect.de/DOI/DOI?10.1055/a-1290-8070 (visited on 09/12/2024).

[67] T. Birsan and D. Tiba, "One Hundred Years Since the Introduction of the Set Distance by Dimitrie Pompeiu," in *System Modeling and Optimization*, F. Ceragioli, A. Dontchev, H. Futura, K. Marti, and L. Pandolfi, Eds., vol. 199, Boston: Kluwer Academic Publishers, 2006, pp. 35–39, isbn: 978-0-387-32774-7. doi: 10.1007/0-387-33006-2_4. [Online]. Available: http://link.springer.com/10.1007/0-387-33006-2_4 (visited on 02/06/2024).

[68] K. Sim, M. Nia, C. Tso, and T. Kho, "Brain Ventricle Detection Using Hausdorff Distance," in *Emerging Trends in Applications and Infrastructures for Computational Biology, Bioinformatics, and Systems Biology*, Elsevier, 2016, pp. 523–531, isbn: 978-0-12-804203-8. doi: 10.1016/B978-0-12-804203-8.00034-1. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128042038000341 (visited on 10/17/2024).

[69] N. Gegoire and M. Bouillot, "Hausdorff Distance between convex polygons," McGill University, 1998. [Online]. Available: https://cgm.cs.mcgill.ca/~godfried/teaching/cg-projects/98/normand/main.html (visited on 10/17/2024).

[70] J. Ruberte, P. N. Schofield, J. P. Sundberg, A. Rodriguez-Baeza, A. Carretero, and C. McKerlie, "Bridging mouse and human anatomies; a knowledge-based approach to comparative anatomy for disease model phenotyping," *Mammalian Genome*, vol. 34, no. 3, pp. 389–407, Sep. 2023. doi: 10.1007/s00335-023-10005-4. [Online]. Available: https://link.springer.com/10.1007/s00335-023-10005-4 (visited on 09/27/2024).

[71] C. Jerome, B. Hoch, and C. S. Carlson, "Skeletal System," in *Comparative Anatomy and Histology*, Elsevier, 2018, pp. 67–88, isbn: 978-0-12-802900-8. doi: 10.1016/B978-0-12-802900-8.00005-1. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/B9780128029008000051 (visited on 09/27/2024).

[72] M. Halle, V. Demeusy, and R. Kikinis, "The Open Anatomy Browser: A Collaborative Web-Based Viewer for Interoperable Anatomy Atlases," *Frontiers in Neuroinformatics*, vol. 11, Mar. 27, 2017, issn: 1662-5196. doi: 10.3389/fninf.2017.00022. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fninf.2017.00022/full (visited on 10/18/2023).

[73] W. Platzer, *Taschenatlas Anatomie. Bd. 1: Bewegungsapparat / Werner Platzer*, 11., überarb. Aufl. Stuttgart: Thieme, 2013, 467 pp., isbn: 978-3-13-492011-6.

[74] J. Ruberte, A. Carretero, and M. Navarro, *Morphological Mouse Phenotyping: Anatomy, Histology and Imaging*. Madrid, Spain: Academic Press, 2017, isbn: 978-0-12-812972-2.

[75] M. Harrison, A. O'Brien, L. Adams, G. Cowin, M. J. Ruitenberg, G. Sengul, and C. Watson, "Vertebral landmarks for the identification of spinal cord segments in the mouse," *NeuroImage*, vol. 68, pp. 22–29, Mar. 2013, issn: 10538119. doi: 10.1016/j.neuroimage.2012.11.048. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1053811912011603 (visited on 10/09/2023).

[76] A. Lasso, *SlicerSegmentEditorExtraEffects*, PerkLab, Mar. 5, 2024. [Online]. Available: https://github.com/lassoan/SlicerSegmentEditorExtraEffects (visited on 09/23/2024).

[77] Z. Wang, E. Wang, and Y. Zhu, "Image segmentation evaluation: A survey of methods," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5637–5674, Dec. 2020, issn: 0269-2821, 1573-7462. doi: 10.1007/s10462-020-09830-9. [Online]. Available: https://link.springer.com/10.1007/s10462-020-09830-9 (visited on 09/30/2024).

[78] A. W. Setiawan, "Image Segmentation Metrics in Skin Lesion: Accuracy, Sensitivity, Specificity, Dice Coefficient, Jaccard Index, and Matthews Correlation Coefficient," in *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, Surabaya, Indonesia: IEEE, Nov. 17, 2020, pp. 97–102, isbn: 978-1-7281-8283-4. doi: 10.1109/CENIM51130.2020.9297970. [Online]. Available: https://ieeexplore.ieee.org/document/9297970/ (visited on 09/12/2024).

[79] İ. Ataş, "Performance Evaluation of Jaccard-Dice Coefficient on Building Segmentation from High Resolution Satellite Images," *Balkan Journal of Electrical and Computer Engineering*, vol. 11, no. 1, pp. 100–106, Jan. 30, 2023, issn: 2147-284X. doi: 10.17694/bajece.1212563. [Online]. Available: http://dergipark.org.tr/en/doi/10.17694/bajece.1212563 (visited on 09/12/2024).

[80] M. E. Rayed, S. S. Islam, S. I. Niha, J. R. Jim, M. M. Kabir, and M. Mridha, "Deep learning for medical image segmentation: State-of-the-art advancements and challenges," *Informatics in Medicine Unlocked*, vol. 47, 2024, issn: 23529148. doi: 10.1016/j.imu.2024.101504. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352914824000601 (visited on 10/18/2024).

[81] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "MESH: Measuring errors between surfaces using the Hausdorff distance," in *Proceedings. IEEE International Conference on Multimedia and Expo*, Lausanne, Switzerland: IEEE, 2002, pp. 705–708, isbn: 978-0-7803-7304-4. doi: 10.1109/ICME.2002.1035879. [Online]. Available: http://ieeexplore.ieee.org/document/1035879/ (visited on 10/16/2024).

[82] T. Rautenkranz, *Sectional-anatomy*, Mar. 15, 2018. [Online]. Available: https://www.sectional-anatomy.org (visited on 10/13/2024).

[83] T. N. Andersen, T. A. Darvann, S. Murakami, P. Larsen, Y. Senda, A. Bilde, C. V. Buchwald, and S. Kreiborg, "Accuracy and precision of manual segmentation of the maxillary sinus in MR images—a method study," *The British Journal of Radiology*, vol. 91, no. 1085, p. 20 170 663, May 1, 2018, issn: 1748-880X. doi: 10.1259/bjr.20170663. [Online]. Available: https://academic.oup.com/bjr/article/doi/10.1259/bjr.20170663/7449367 (visited on 11/12/2024).

[84] K. C. Brata and A. H. Brata, "User experience improvement of japanese language mobile learning application through mental model and A/B testing," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 3, p. 2659, Jun. 1, 2020, issn: 2088-8708, 2088-8708. doi: 10.11591/ijece.v10i3.pp2659-2667. [Online]. Available: http://ijece.iaescore.com/index.php/IJECE/article/view/20880 (visited on 10/27/2024).

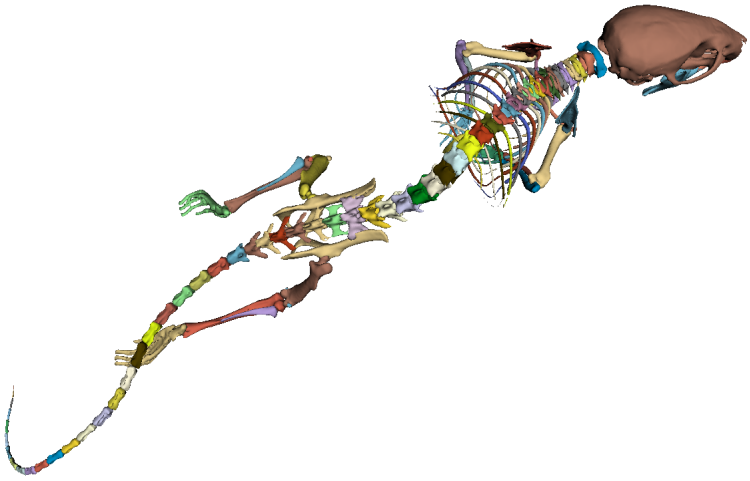
[85] M. B. Chavez, E. Y. Chu, V. Kram, L. F. De Castro, M. J. Somerman, and B. L. Foster, "Guidelines for Micro–Computed Tomography Analysis of Rodent Dentoalveolar Tissues," *JBMR Plus*, vol. 5, no. 3, e10474, Mar. 2021, issn: 2473-4039. doi: 10.1002/jbm4.10474. [Online]. Available: https://academic.oup.com/jbmrplus/article/7499109 (visited on 10/28/2024).

Appendix A: Segmentation guide

FACHHOCHSCHULE
WIENER NEUSTADT
University of Applied Sciences – Austria

MicroCT Segmentation Guide for Mice

STUDENT NAME:	**Stefan Rohrbacher**
STUDENT ID:	**01607307**
COURSE NAME:	**MedTech**
SUPERVISOR:	**Dipl. Ing. Michael Rauter**
DATE OF SUBMISSION:	**November 19, 2024**



Deutschlandsberg, August 27, 2024

Contents

List of Figures

Glossary

CPU Core Processing Unit. 3

FOV Field of View. 7

GPU Graphics Processing Unit. 3

Houndsfield unit (HU) A mapping of the linear attenuation coefficient of the X-ray beam on to an arbitrary scale called Houndsfield scale[1]. 9

MRB Medical Reality Bundle (.mrb, .zip): MRB is a binary format encapsulating all scene data (bulk data and metadata). Internally it uses zip format. [2]. 13

OpenVR "[…] an API and runtime that allows access to virtual reality hardware from multiple vendors without requiring that applications have specific knowledge of the hardware they are targeting"[3]. 4

ROI Region of Interest. 51

VRAM GPU memory. 4

1 Introduction

This guide is intended to help newcomers get started with segmentation tasks in a software called "3D Slicer". 3D Slicer is a comprehensive and fully featured program for visualization, processing, segmentation, registration and analysis of 3D datasets. Unfortunately, this also means that it can be overwhelming for someone who lacks experience working with such tools. This guide aims to alleviate this issue at least for segmentation workflows. It tries to explain tools and possible workflows in an easy-to-follow manner and enable the reader to complete their segmentation without needing to watch multiple tutorials or read bits of documentation scattered around the internet.

1.1 Using the guide

1. **Jump around using links:** The guide makes extensive use of hyperlinks, which enable the reader to easily jump to important information. Make use of this feature to save yourself some time. Note that most but not all PDF readers will support this feature.

2. **Jump to information you require:** This guide covers many tools and functionalities of 3D Slicer. You will most certainly not use every tool. Use the table of contents to jump to tools that are important for the task at hand.

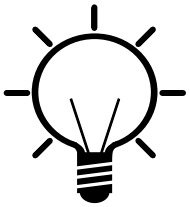
3. **Make a backup before experimenting:** Sometimes 3D Slicer makes it difficult to undo an operation. If you wish to experiment with a tool you have never used before it is advisable to duplicate your dataset and save it to a separate file. So that in case something does not turn out the way you intended, and you do not know how to undo the unwanted change, you can always load a known good copy of your dataset.

1.2 Conventions used in this Guide

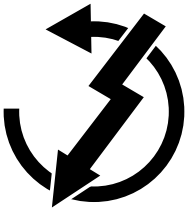
(a) 2D symbol (b) 3D symbol

Figure 1: 2D (figure 1a) and 3D (figure 1b) symbols signifying if a tool can be used in 2D or 3D

If you are already familiar with 3D Slicer and want to start with the segmentation right away, skip to page 47 for the Quick Start Guide.

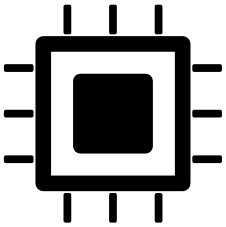


(a) Hint symbol

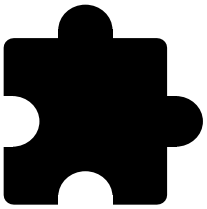


(b) Irreversible symbol

Figure 2: Hint (figure 2a) symbol signifying tip or hint and Irreversible(figure 2b) symbol signifying irreversible operation



(a) Performance symbol



(b) Plugin symbol

Figure 3: Performance(figure 3a) symbol signifying resource intensive operation and Plugin (figure 3b) symbol signifying 3rd party plugin tool

1.3 Installation

Basic installation instructions for 3D Slicer:

1. Browse to: `https://download.slicer.org`
2. Download the installer of the latest stable release[1] for your operating system
3. Run the installer and follow its instructions

1.3.1 Installation on macOS

On macOS, 3D Slicer can be installed in two different ways:

Either via the provided package on the downloads page, or via Homebrew: `https://formulae.brew.sh/cask/slicer`
For the conventional installation refer to the 3D Slicer wiki: `https://slicer.readthedocs.io/en/latest/user_guide/getting_started.html#mac`

[1] Version 5.6.2 at the time of writing this document

1.3.2 Installation on Linux

3D Slicer ships with all its dependencies on Windows and macOS.
On Linux it is required to install some dependencies via your distributions package manager.
The 3D Slicer wiki gives information about required packages and their respective names on different distributions: `https://slicer.readthedocs.io/en/latest/user_guide/getting_started.html#linux`.
To make this process easier, this guide provides a distrobox manifest file (see page: 50), which allows the user to create a Linux container holding all necessary dependencies without polluting the host system.

To run 3D Slicer via the container run:

```
distrobox assemble create --file /path/to/manifest_file.ini
distrobox enter slicerbox -- ./path/to/Slicer/binary/Slicer
```

1.4 System Requirements[4]

Note: as of November 19, 2024

1.4.1 Official supported operating systems

- Windows: 10 or 11
- macOS: 11 or later
- Linux:
  - Ubuntu 20.04 or later
  - Debian 10 or later
  - Fedora 35 or later
  - CentOS 7 or later

1.4.2 Hardware requirements

- Memory: At least 4 Gigabytes, more is strongly recommended.
  The 3D Slicer Wiki mentions needing about 10 times more memory than the amount of data you plan to load.
  3D Slicer on its own uses about 456 Megabytes of RAM. After loading a 26.17MiB Dataset it consumes about 614 MB.
- CPU: 3D Slicer uses multi-threading for some calculation and will thus benefit from a multicore CPU.
- GPU: A dedicated GPU is recommended but not required as 3D Slicer only uses it for interactive volume rendering. If you restrict your usage to the 2D views, you will hardly ever use your graphics card.

1.4.3 Hardware recommendations

The following points are not strictly needed in order to use 3D Slicer, but they make working with the software a lot easier.

- Input devices: 3D Slicer supports mice, touchpads, pens, graphic tablets and OpenVR headsets. The easiest input method though is a 3 button Mouse with a mouse wheel.
- Internet connection: for downloading extensions, online documentation and sample data sets.
- VRAM: for interactive volume rendering it is recommended to have more GPU texture memory than the data set you plan to load.
- A large display or monitor with a decent screen resolution. A 14-inch screen or larger is strongly advised. Screen resolution does not need to exceed 1920x1080, but going below 720p is also not advisable.

1.5 Interface and Usage

Upon first launch you will be greeted by the 3D Slicers welcome screen (figure 4) The welcome screen has some quick links to useful modules like load data and

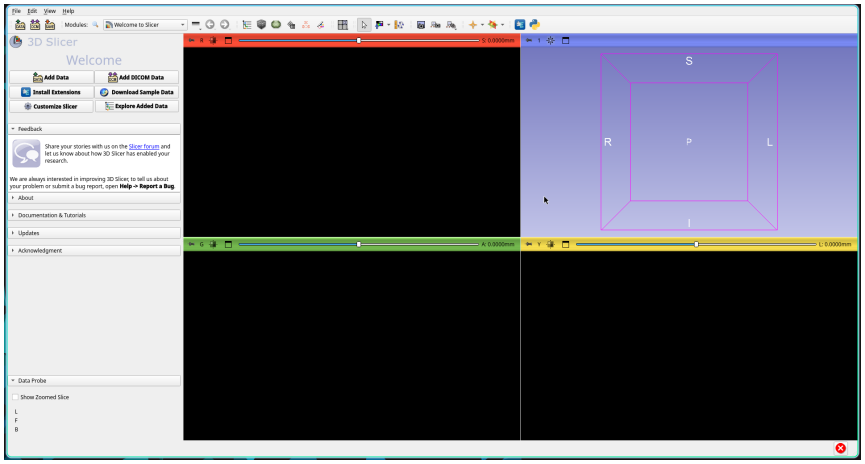


Figure 4: 3D Slicer welcome screen

download datasets.

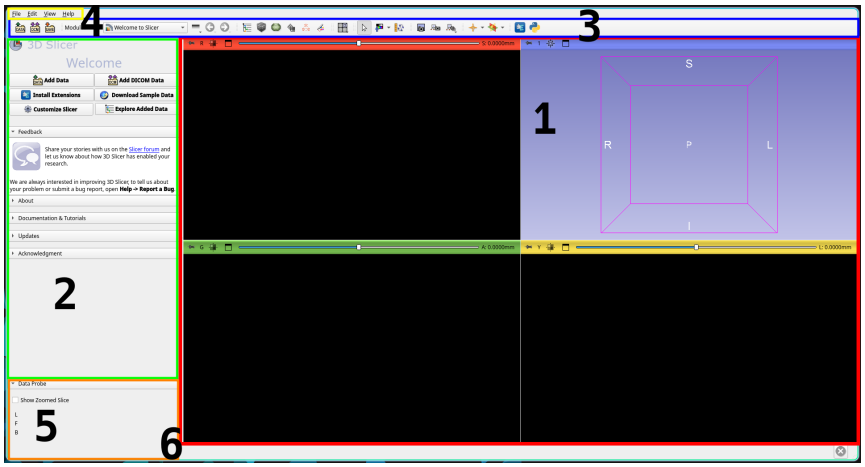


Figure 5: 3D Slicer window

The 3D Slicer window consists of several areas, which provide different tools to interact with the program. A quick overview of these areas is provided by figure 5:

1. View area
2. Module panel
3. Toolbar
4. Application menu
5. Data probe
6. Status bar

1.5.1 The view area

By default 3D Slicer shows the view area (figure 6) with three 2D slice views and the interactive 3D view at the top right.

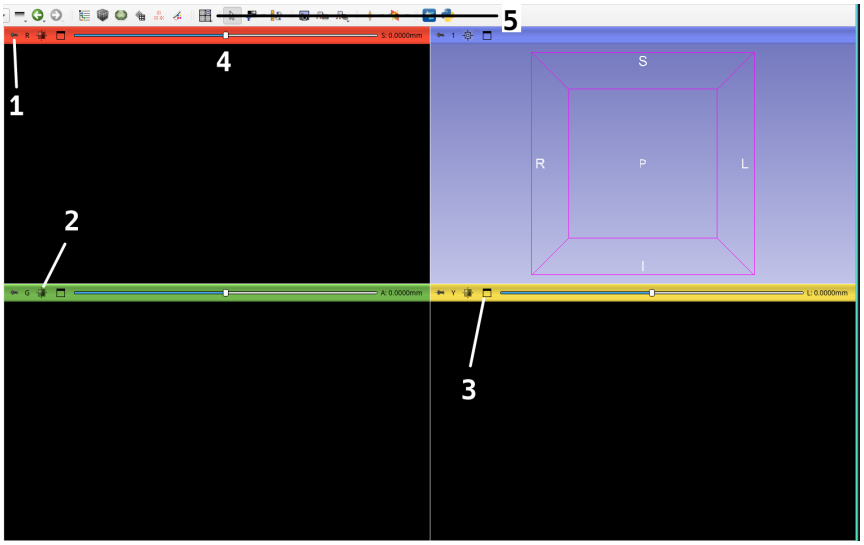


Figure 6: 3D Slicer view area

3D Slicer calls this the `Four-up` view.

There is a numerous amount of view presets to choose from by clicking on the `change view` option in the toolbar: figure 6:5.

Each slice view has its own scroll bar (figure 6:4). The number on the right end of the scroll bar shows by how much a single tick scrolls through the dataset. In other words: it shows the slice thickness of the view.
Scrollbar controls:

- left click and drag the scrollbar
- mouse wheel
- `B` and `F` Keys
- up, down, left and right arrow keys

To maximize a single slice view, click on the `Maximize view` button (figure 6:3). Click it again to return to the previous layout. Double left-clicking on the slice has the same effect.

'Reset FOV' can be used to reset the zoom level and shift the slice into its isocenter (figure 6:2). The `R` key has the same effect.

The pin icon (figure 7:1) brings up a submenu.
Here (figure 7:2) it is possible to quickly change the displayed dataset if more than one has been loaded.
Change the displayed slice plane by clicking on figure 7:3.
Toggle slice visibility in 3D views: figure 7:4.
Clicking on the `double arrow` button (figure 7:5) will reveal more configuration options, which this guide will not cover, as they are not usually needed for segmentation tasks. However, the 3D Slicer documentation provides more information about this submenu: `https://slicer.readthedocs.io/en/latest/user_guide/user_interface.html#slice-view`

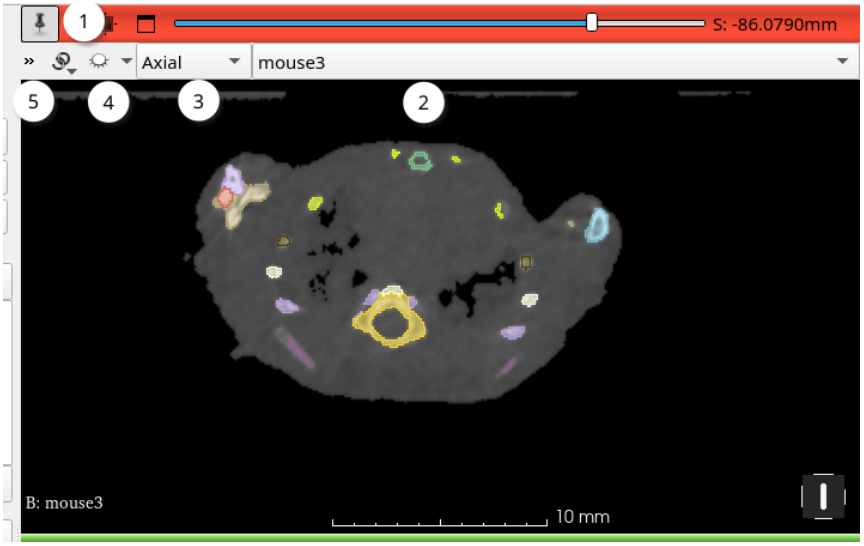


Figure 7: 3D Slicer pin menu

1.5.2 The module panel

Figure 5:2 the module panel displays options, tools and information of the currently loaded module.

1.5.3 The Toolbar

Figure 8: 3D Slicer toolbar

The far left end of 3D Slicers toolbar holds buttons for loading and saving data. Loading data from a file or directory can be done via the `Add Data` button' figure 8:1.
Keyboard shortcut: `Ctrl + o`
The next button (figure 8:2) loads the `Add DICOM Data` module, which can also load data from files or directories and save it to 3D Slicers on disk DICOM database.
Saving data can be done via clicking on the `Save Data` button (figure 8:3).
Keyboard shortcut: `Ctrl + s`
Figure 8:4 marks the module toolbar. Clicking on the lens button opens a full text search for all installed modules.
Keyboard shortcut: `Ctrl + f`
Loadable modules can also be found via the dropdown list right next to the lens icon. 3D Slicer keeps track of the recently used modules and allows to quickly return to the most recent eight[2] modules. Either via the `Modules history` dropdown list or the forward and back buttons (figure 8:4). The toolbar has a designated part for `Favorite Modules` (figure 8:5), which can be fully customized via the settings menu.
Figure 8:6 can be used to change the view layout, like mentioned in section 1.5.1.
3D Slicer has different mouse modes, where depending on the selected mode, the mouse has different functionalities. By default, `translate/rotate view` is selected (figure 8:7).
Keyboard shortcuts:
`right click + mouse drag` to zoom in and out.
`Ctrl + mouse wheel` to zoom in and out.
`Shift + left mouse click + drag` pan view.
`Shift + move mouse` display ROI under mouse in all views, 3D Slicer calls this the crosshair.
The icon to the right is called `adjust window/level`.
`Left click` and drag to create a ROI, 3D Slicer then tries to find the optimal window and center for that ROI.
Alternatively hold `Ctrl` and `left mouse`, then drag the mouse to adjust the window manually.
The final button of this part of the toolbar toggles an additional markups toolbar on and off, this guide will not go into further detail as this is not needed for segmentation.

[2] the amount can be configured in the settings menu

3D Slicer has a build in screenshot and screen recording module as can be seen in figure 8:8.
To toggle visibility of the crosshair, that has been mentioned above the left button in figure 8:9 can be used. The button to the right of that toggles slice intersection lines on and off.
Finally, on the right end (figure 8:10) of the toolbar 3D Slicer has buttons to open a Python console window and opening the `Extension Manager`.

1.5.4 The Application Menu

See figure 5:4.

- `File`: Loading and saving data, download sample data, unloading data and close 3D Slicer
- `Edit`: Cut, Copy, Paste and the `Settings Menu`
- `View`: Show or hide additional windows and toolbars. Open the `Extension Manager`, `Python console`, `Error Log` and change the view layout.
- `Help`: Access online documentation.

1.5.5 The Data Probe

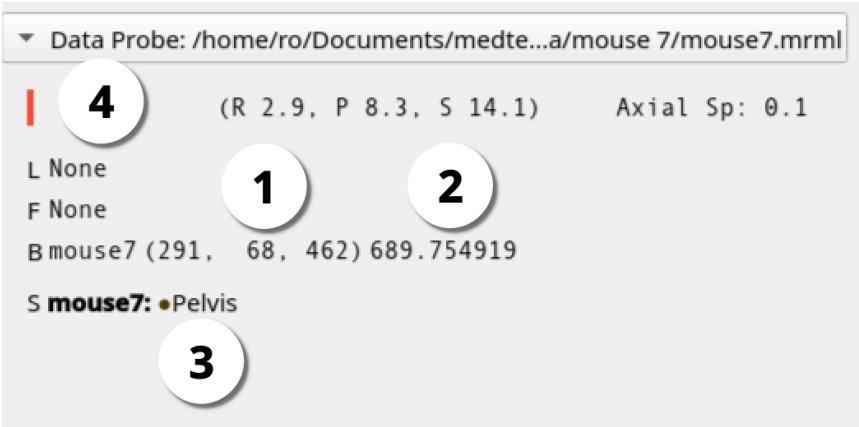


Figure 9: 3D Slicer Dataprobe

The `Data Probe` shows useful information about the data currently being displayed. More specifically it informs the user about the ROI under the cursor position.
To start using it, just hover the mouse cursor over a ROI, if the cursor is moved, the data probe will update in real time.
Most notable information shown:
The numbers in brackets in figure 9:1 display the x-, y- and z-Position of the ROI. After that, outside the brackets, it displays the Houndsfield unit (HU)

value of that ROI (figure 9:2).
In figure 9:3 the data probe shows which segmentation ROI belongs to if any.
Figure 9:4 displays the currently loaded dataset.

1.5.6 The Status Bar

The status bar located on the bottom of the 3D Slicer window, is of no great importance during normal operation. Notably the `X` icon on the far right opens the error log, which may be useful for debugging.

2 Basic usage

From this point onwards, this guide makes the following assumptions:

1. 3D Slicer has been successfully installed
2. The data you wish to work on has already been acquired
3. The computer running 3D Slicer has access to the data

2.1 Loading Data

3D Slicer can load data in two different ways. Loading data from file or folder or load data from folder and save to database (8:2). The database option is advisable if you are dealing with a larger number of studies, as it enables quick switching between studies. If only a single study needs to be loaded, usage of the database is not necessary. Clicking on the `Add Data` Button (8:2) or using the keyboard shortcut `Ctrl + o` brings up a file picker dialogue.

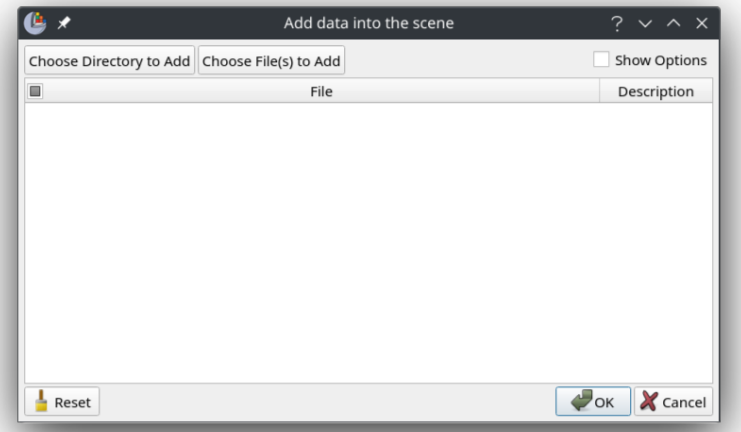


Figure 10: 3D Slicer file picker

Depending on whether the data is contained in a single or multiple files, click either `Choose Directory to Add` or `Choose File(s) to Add`. Browse to your data, select it and confirm your selection by clicking `Choose`. Show additional import options by setting a checkmark at `Show Options`. Here it is possible to load the dataset as a Label map, force 3D Slicer to ignore similar files, automatically center the volume, ignore orientation information in the DICOM header, show or hide the volume and set the color table. After confirming the import, depending on size of the dataset, some patience may be required. If

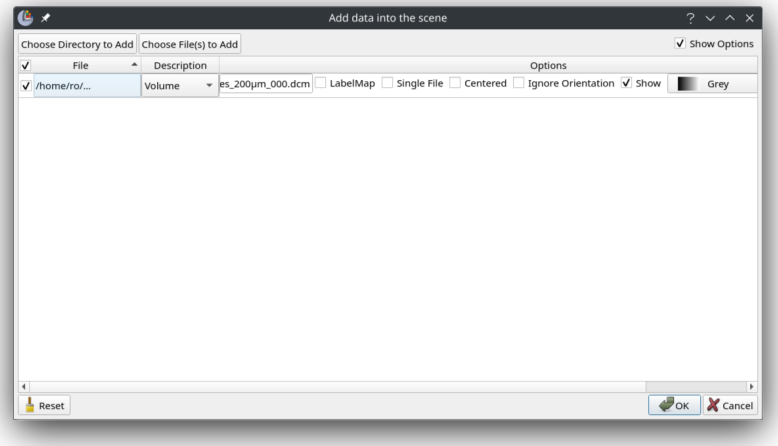


Figure 11: 3D Slicer file picker options

the import was successful and the dataset has not been set to be hidden after import, 3D Slicer will automatically populate the view area.

2.2 Saving Data

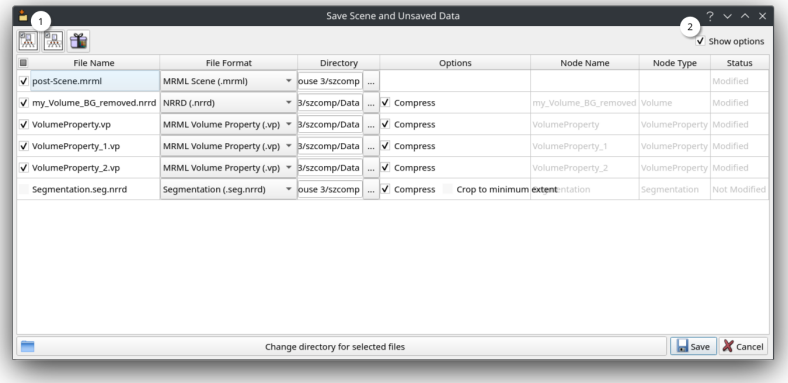


Figure 12: 3D Slicer file save menu

Save your work in 3D Slicer by clicking on the `Save Data` button in the figure 8:3. This will open a popup window which holds information about what

data will be saved and the format it will be saved to. 3D Slicer defaults to bundling up your data into a single file. Change this by clicking on the bundle icon (figure 12:1) and set a checkmark at `Show options` (figure 12:2). The window should now look like Figure 12. The first column shows the file names, which are derived from their names in the `Data` module and their file format.

Let us take a look at what 3D Slicer is going to save:

post-Scene.mrml "MRML file is a xml-formatted text file with scene metadata and pointers to externally stored data files"[2] (Metadata file)

my__Volume__BG__removed.nrrd volume data in a non-DICOM general purpose multidimensional format

VolumeProperty.vp volume property file storing information on volume rendering (Metadata file)

Segmentation.seg.nrrd "Segmentation labelmap representation"[2] storing the segmentation as a multidimensional volume

3D Slicer defaults to compressing all data before saving and there should be no reason to turn this behavior off. The second column is dedicated to the file format of the mentioned files. 3D Slicer supports a number of different file formats (see: `https://slicer.readthedocs.io/en/latest/user_guide/data_loading_and_saving.html#supported-data-formats`). However, exporting for example a volume or segmentation in a different format, say STL for 3D printing, I recommend choosing the `Data` module over changing file formats in the save dialogue. Compression can be turned off or on per file in the option's column. And somewhat importantly, the last column, the status column, shows if a file has been modified since the last save.

Clicking on the "Bundle" icon (Figure 12:1) collapses the file view, as this instructs 3D Slicer to not save individual files, but bundle them in a MRB file.

Pick a directory by clicking on `Change directory for selected files` and confirm via the save icon. Depending on the file size this operation may require some patience.

2.3 Addressing Performance issues

If loading the data took a long time or scrolling in the 2D views is sluggish, it might be worth considering reducing the size of your dataset. Make sure to save after every operation to reduce the loss of progress in case of a crash.

2.3.1 Cropping

Locate the `Crop Volume` module either via the dropdown menu (figure 8) `Converters -> Crop Volume` switch to it. Or click the lens icon to use the text search (figure 13) Before any cropping can happen it is required to do some setup in the newly opened module panel.

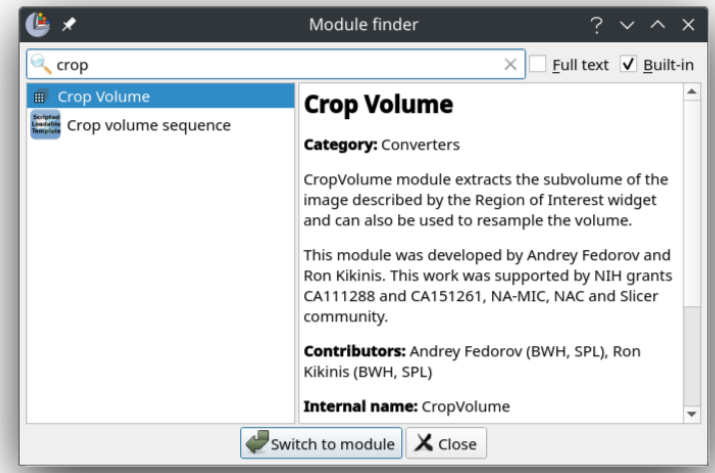


Figure 13: 3D Slicer module switcher

See figure 14 for reference. Under `IO -> Input Volume` make sure the dataset you loaded is selected. Create a new ROI under `IO -> Input ROI` choose `Create new ROI as…` and give it a distinctive name. Also make sure it is not hidden by looking for an open eye next to `Display ROI`.

To get a better visualization of the volume that is going to remain define the ROI in the `Volume rendering` module.

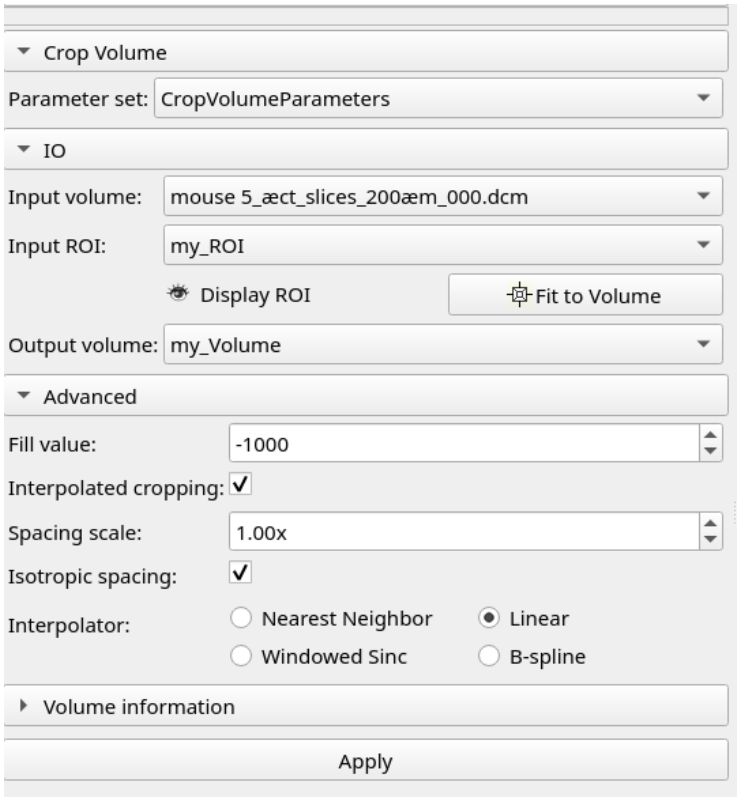


Figure 14: 3D Slicer cropping module panel

Next choose a name for your output volume under `IO -> Output volume -> Create new volume as…` and give it a distinctive name. Under `Advanced -> Fill value` choose a HU value for everything outside your ROI. It should be easily distinguishable from the structure or tissue you are segmenting. For bone segmentation I recommend choosing -1000 HU[3]. Checking the tick box `Advanced -> Interpolated cropping` will ensure that the output volume has the same dimensions as the input volume. Also check the box: `Advanced -> Isotropic spacing` with a `Spacing scale` of 1x to ensure the voxels stay isotropic. Now adjust your ROI in the 2D view areas by clicking and dragging the colored dots at the edges and corners of the ROI. Crop out as much excess volume as possible without affecting the anatomy you wish to segment. Check positioning and size of your ROI by scrolling through the 2D view on all three planes, after that click `Apply`. This operation might require some patience depending on the size of the dataset. Note that there will be no visual confirmation the cropping operation is finished. However, you can check by switching to the `Data` module. If 3D

[3]The HU value of air

Slicer is unresponsive, the operation is not done yet.

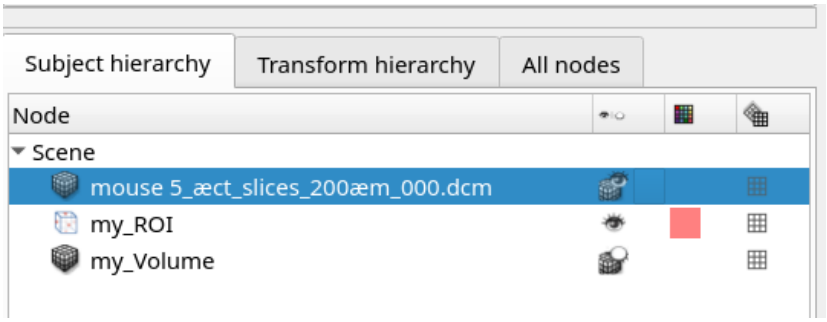


Figure 15: 3D Slicer data module

Under `Subject hierarchy -> Node -> Scene` (figure 15) you should see:

1. the loaded dataset
2. the newly created ROI
3. the newly created volume

Show your cropped volume by clicking on the closed eye on the same line as its name. Hide the ROI by clicking on the open eye next to its name. 3D slicer will then automatically hide the original dataset and populate the view area with the cropped volume dataset.

2.3.2 Masking

In the last step we cropped out the part of the source volume which does not contain any material or tissue of interest. This step is dedicated to homogenizing the volume which was not cropped out in the last step. Switch to the `Segment Editor` module (figure 16:1). Make sure your cropped volume is selected as the `Source volume` (figure 16:2). Click on the `+ Add` button (figure 16:3). You should see a new segment appear in the segments list below. Double click to rename it to "Background" or leave its default name. Make sure your "Background"segment is select and then activate the `Threshold` tool (figure 16:4). Shift the lower threshold limit until the background is covered by the segment color in the 2D views. Afterward shift the upper threshold limit until your tissue of interest is definitely no longer covered by the segment color. Before applying (figure 16:7) the threshold segmentation make sure that 3D Slicer is allowed to edit `Everywhere` and overwrite all other segments (figure 16:6).

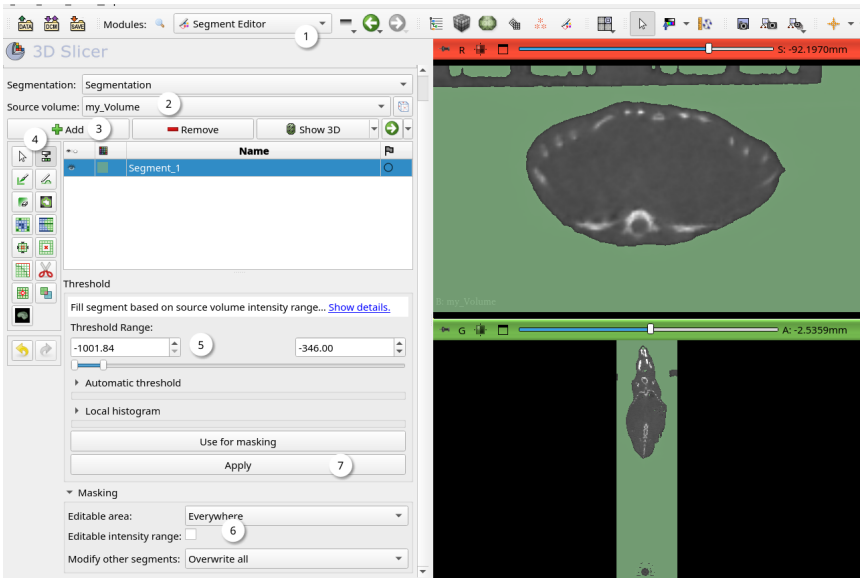


Figure 16: Threshold background

Next, activate the `Scissors` tool (figure 17:1) with your "Background" segment selected. This tool provides a simple way of selecting materials and tissue that are not easily distinguished by HU value, like the MicroCT couch and positioning tools. Change its operation mode to `Fill inside` (figure 17:2) and its shape to `Rectangle` or `Free-form` (figure 17:3). In the 2D views you can now click and drag to create a shape, upon releasing the left mouse button, the tool will add the volume inside the shape to your "Background" segment.

The scissors tool "punches" through the volume, this makes it very easy to accidentally select something intentionally. Always make sure there is no ROI in the area on the previous or next slices and check your positioning on the other planes. In most cases it will be easier to work on small sections of your volume, rather than selecting a large volume in one single step.

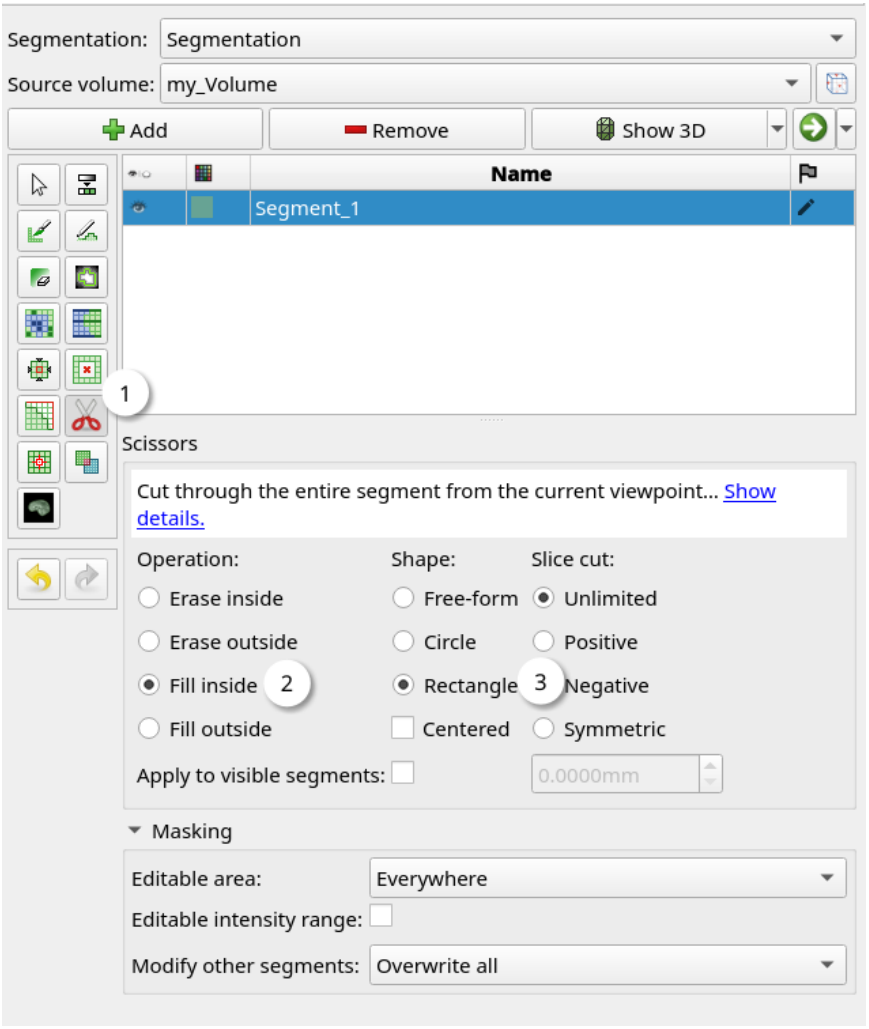


Figure 17: Scissors background

Activate the `Mask volume` tool (figure 18:2) and make sure your "Background" segment is selected (figure 18:1). As for the operation mode, choose `Fill inside` (figure 18:3). The fill value (figure 18:4) should be the same as in the cropping step (section 2.3.1). Select your cropped source volume as the `Input Volume` (figure 18:5). For the result, create a new volume by clicking on `Output Volume -> Create new Volume as…` (figure 18:6) and give it a distinctive name.

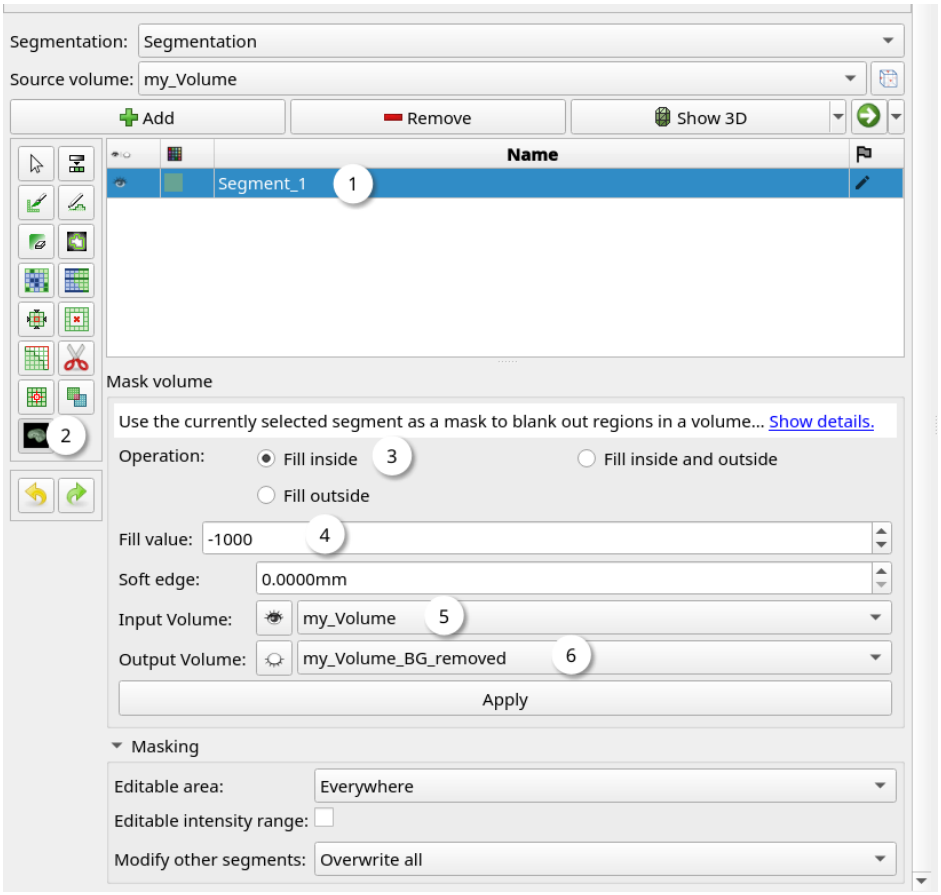
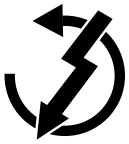


Figure 18: Mask background



Always create a new volume when working with the `Mask volume` tool. It overrides the individual voxel intensity values, this *can not be undone* as 3D Slicer does not store the original intensity values in its undo history.

2.3.3 Cleanup

Switch to the `Data` module. Here we see all the items we have created so far.

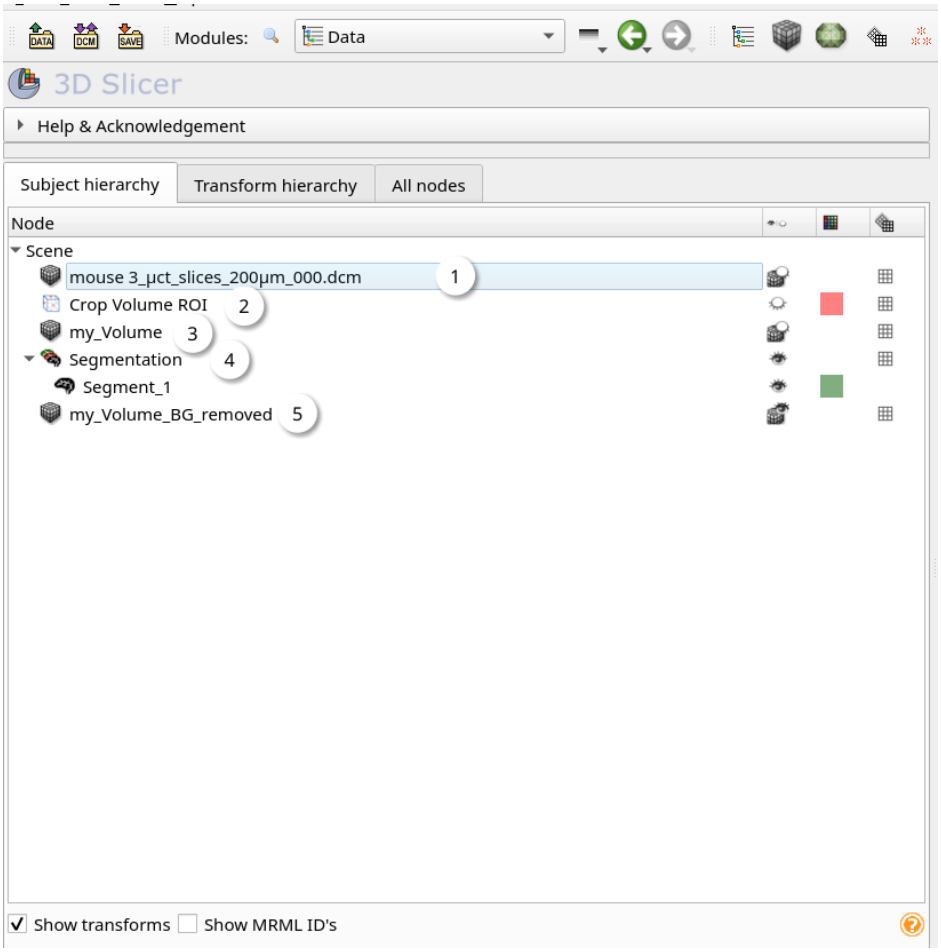


Figure 19: Data cleanup

Figure 19:1 is the DICOM dataset from the MicroCT scanner. Figure 19:2 is the cropping ROI created in section 2.3.1 and figure 19:3 is the volume that resulted from the cropping operation. The segmentation item (figure 19:4) holds all segmentations created in the `Segment Editor` module. So far it only holds the "Background" segment created in Section 2.3.2. And the final item (figure 19:5) is the most recent volume resulting from the `Mask volume` operation. Items figure 19:1-4 can be deleted to save harddrive space and RAM while working in 3D Slicer. In this example the data volume on disk could be decreased from 309 megabytes to (source DICOM data) to 15 megabytes. Which equates to

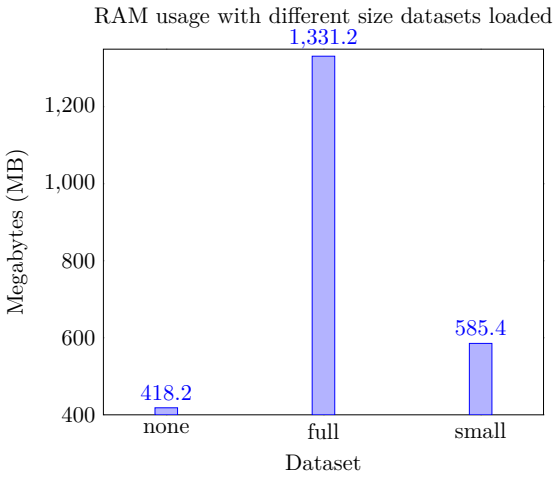


Figure 20: RAM usage comparison

approximately a 95% reduction in file size. As a result of this, 3D Slicers RAM consumption has dropped from 1.3 gigabytes to 585 megabytes, which equates to approximately 55% reduction. See Appendix A.2 for more information.

Figure 20 is a RAM usage comparison of 3D Slicer under different workloads, with the intention of showing the user what amount of resource consumption to expect.
`none` refers to 3D Slicers base RAM usage without any dataset loaded.
`full` refers to the RAM usage with a raw dataset of 309 megabytes file size.
`small` refers to the RAM usage with the same dataset as above but reduced down to 15 megabytes file size using the method explained in section 2.3.

3 Segmentation tools

With your dataset loaded and prepared switch to the `Segmentation Editor` module. Before you start to segment *ALWAYS* make sure the masking options are set correctly.

Editable area which areas are affected by your tool

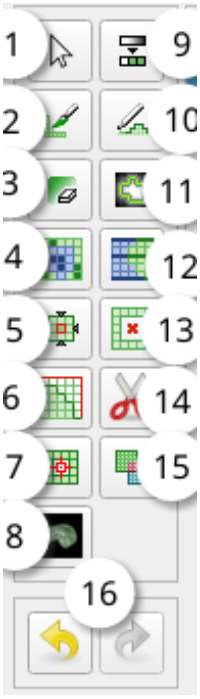
- **Everywhere** no restrictions, tool is usable anywhere in the view areas
- **Inside all segments** tool can overwrite any existing segment, does not work outside of segments
- **Inside all visible segments** tool can overwrite any existing segment that is not hidden, does not work outside of segments
- **Outside all segments** inverse of `Inside all segments`, tool works only outside existing segments
- **Outside all visible segments** inverse of `Inside all visible segments`, tool works only outside existing non-hidden segments
- **Inside `user created segments`** tool can override only selected existing segment, does not work outside of segment

Editable intensity range restrict tool to affect only specific HU value range

Modify other segments define interaction with other existing segments

- **Overwrite all** tool can overwrite all existing segments
- **Overwrite visible** tool can overwrite all existing non-hidden segments
- **Allow overlap** tool does not overwrite existing segments, instead segmented areas are shared between the affected segments

3.1 Built in segmentation tools



1. `No editing`
2. `Paint`
3. `Erase`
4. `Grow from seeds`
5. `Margin`
6. `Smoothing`
7. `Islands`
8. `Mask volume`
9. `Threshold`
10. `Draw`
11. `Level tracing`
12. `Fill between slices`
13. `Hollow`
14. `Scissors`
15. `Logical operators`
16. `Undo and Redo`

Figure 21: Segmentation tool

3.1.1 No editing

Selected by default. Enables inspection of 2D and 3D views without accidentally modifying a segmentation.

3.1.2 Paint

2D 3D

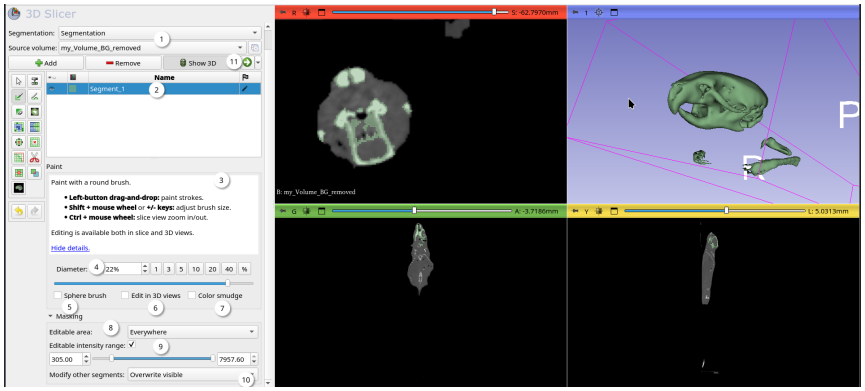


Figure 23: Paint tool

The `Paint` tool is the most widely applicable manual segmentation tool. It is also the basis for most semi-automatic segmentation tools. To get started, first make sure the correct volume dataset and segmentation dataset are loaded (figure 23:1). Create a new segment to segment a new object or organ. Or select an existing segment to continue working on it. Navigate to the region you wish to work on in the 2D view areas and activate the `Paint` tool. Check your masking options before making any modifications (figure 23:10). 3D Slicer displays the most important keyboard options in the help area (figure 23:4). Left-click and drag to draw, Shift+Mouse wheel to change the brush size, Ctrl+mouse wheel to zoom, middle mouse drag and move to pan. Figure 23:4 displays the brush size modification options, click and drag the slider to increase the brush size, alternatively click one of the size presets. In most cases setting the brush size via this menu will be not necessary, instead use the keyboard shortcuts. Keeping the mouse in the view area during segmentation enforces a non-disruptive workflow. Make the `Paint` tool affect more than one slice by making the brush spherical (figure 23:5). The sphere brush can also be used in the 3D view if enabled (figure 23:6). If you wish to allow overlapping segments, enable `Color smudge` (figure 23:7).

3.1.3 Erase

Erase segmentation with brush. Inverse of Paint tool, for usage see section 3.1.2.

3.1.4 Grow from seeds

2D 

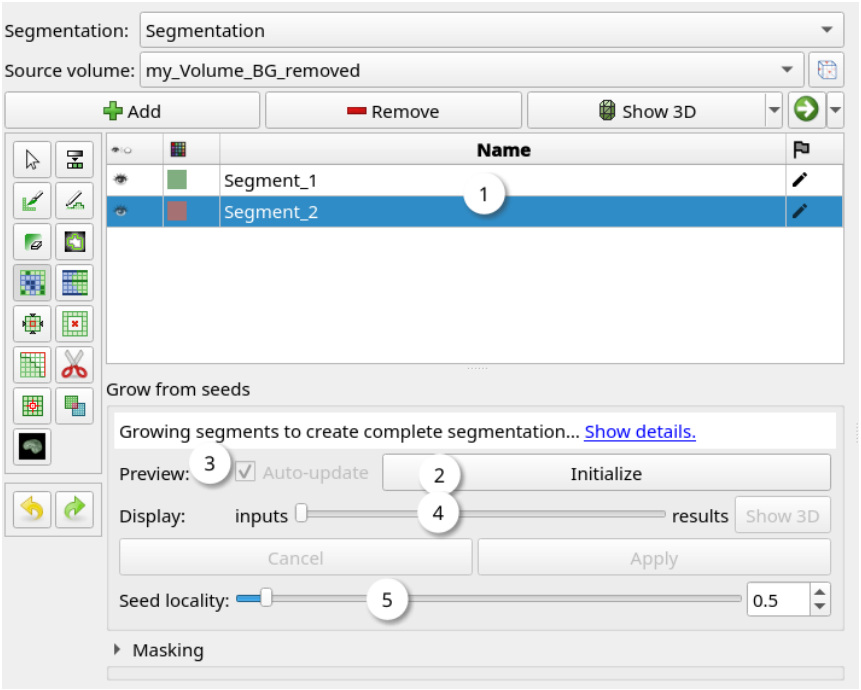


Figure 25: Grow from seeds tool

Semi-automatic segmentation tool based on seeds and distinctive HU changes. To get started create at least two segments (figure 25:1), one for the "Background" and one for the actual segmentation. Use the paint tool to create seeds in those segments. Try to evenly distribute seeds throughout your segmentation area in all three planes. Additionally, try to define the border between your segments manually. Click `Initialize` and wait for the 2D views to display the segmentation preview. Change the opacity of the preview via the slider (figure 25:4) or display it in the 3D view. The first iteration will most likely be not satisfactory. Turn of `Auto-update` (figure 25:3) and switch back to the `Paint` tool. Add seeds in all areas `Grow from seeds` segmented incorrectly. Switch back to `Grow from seeds` and click `update`. If the segmentation does

not grow enough or grows disproportionate, change the seed location modifier (figure 25:5). Increasing the modifier forces the segmentation to be more localized and vice versa. Repeat this cycle of adding seeds and letting the segmentation update until the segmentation is satisfactory. Click apply to confirm the segmentation, delete the "Background" segment if its no longer useful.

3.1.5 Margin

2D

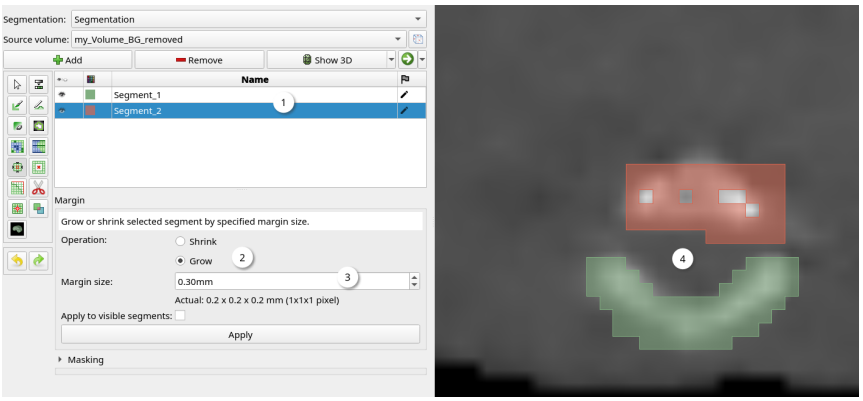


Figure 27: Margin tool

Grow or shrink a segmentation by specific amount. Select the segment you wish to modify (figure 27:1). Pick `Grow` or `Shrink` depending on your needs. Set the size to grow or shrink by (figure 27:3) and click `Apply` to confirm the operation.

This tool can be used to fill small holes by first growing a segment slightly and then immediately shrinking it by the same amount. Or it can be used to separate two segments which have some overlap by shrinking both.

3.1.6 Smoothing

2D

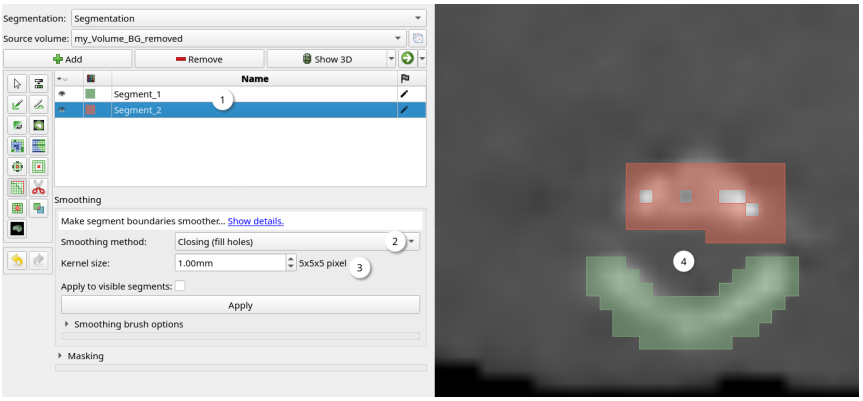


Figure 29: Smoothing tool

Smooth segmentation or fill small holes. Select the segment you wish to modify (figure 29:1). Select a smoothing method (figure 29:2) and kernel size (figure 29:3). If you are not sure which kernel size is best for your dataset, set the kernel size as small as possible in order to avoid large modifications. Undo if the result is not satisfactory, increase the kernel by a small amount and try again.

Median removes small extrusion while leaving smooth areas mostly unchanged

Opening removes extrusions smaller than the kernel size

Closing fills holes and sharp corners smaller than the kernel size

Gaussian smooths all contours and shrinks the segment

Joint smoothing affects all visible segments, smooths multiple segments at once

3.1.7 Islands

2D 3D

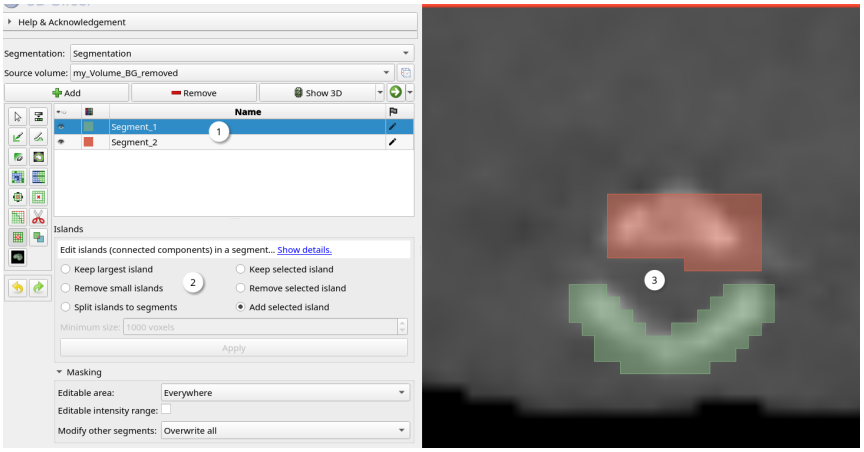


Figure 31: Islands tool

Multitool operating on disconnected segmentation areas or connected segmentation areas with different labels.

Keep largest island deletes all islands but the largest one

Remove small islands deletes all islands smaller than the specified minimum size

Split islands to segments deletes all islands smaller than the specified minimum size, the remaining islands will be automatically assigned a new segment each

Keep selected island click on an island you wish to keep, all other islands will be deleted

Remove selected island click on an island you wish to delete, all other islands will be unaffected

Add selected island click on an island you wish to join with the selected segment, it will inherit the label of the segment

3.1.8 Mask volume

2D 

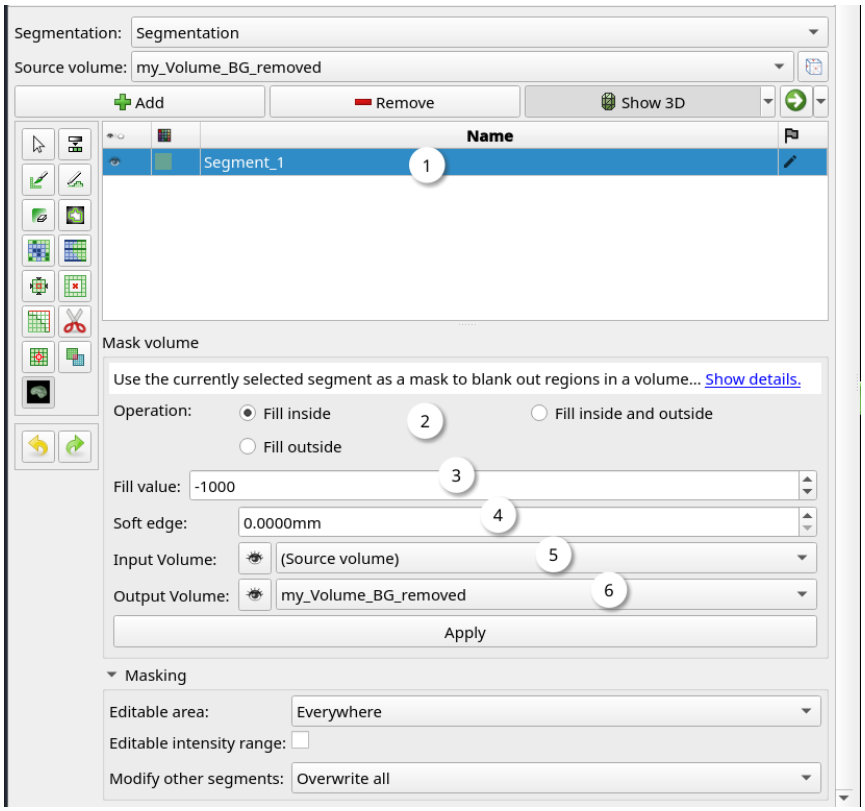


Figure 33: Mask volume tool

Tool to overwrite parts of a volume.

`Mask volume` overwrites HU values in your dataset and 3D Slicer does not save the previous values in its undo history. Thus, if you accidentally overwrite something unintentionally you will not be able to undo your action. When using this tool *always* save your progress and make a backup copy of your save file.

To get started, create and select a segment you wish to modify (figure 33:1).

The segment can for example be the background you wish to blank out. Or it can be an object or organ you wish to blank out or preserve. In the example Figure 33 `Mask volume` was used to blank out the background, some support structures and just preserve the scanned animal. This was achieved by using the `Threshold` tool (section 3.1.9) to segment the air. Then the `Scissors` tool (section 3.1.14) was used to segment the positioning and support structures. Next, select an operation mode (figure 33:2) and fill value (figure 33:3).

Fill inside fill volume inside selected segment with `Fill value`

Fill outside fill volume outside selected segment with `Fill value`

Fill inside and outside takes two fill values (inside and outside), fill volume inside and outside with these values respectively

The tool can also smooth the edge between segments with a blur. Increase the `Soft edge` (figure 33:4) value above zero. Before applying the operation, make sure the correct `Input Volume` is selected and create an `Output Volume`.

3.1.9 Threshold

2D

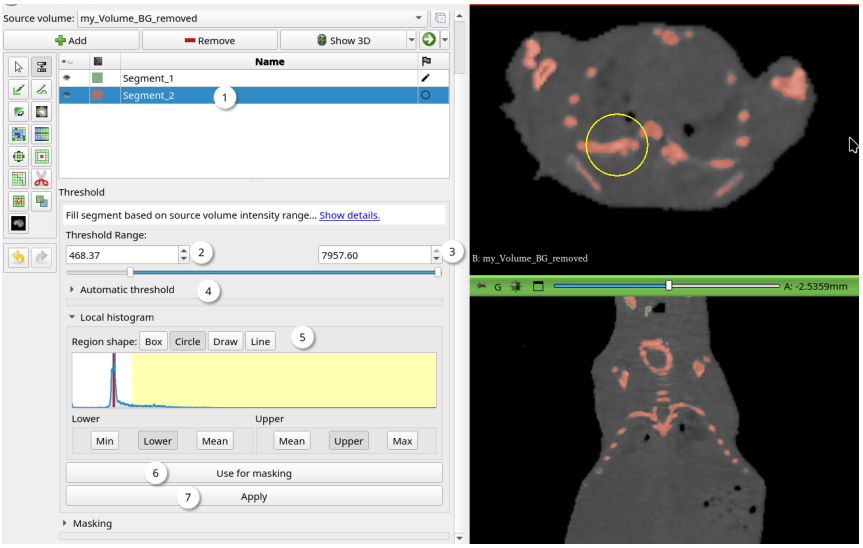


Figure 35: Threshold tool

Segment by HU value range.
First, select the segment to threshold (figure 35:1). Next, pick a lower (figure 35:2) and upper (figure 35:3) limit either by clicking and dragging the sliders

or by typing in the HU values manually. A dynamically updated preview will be available in the 2D views. Automatic threshold algorithms (figure 35:4) can be used to determine the limits but in most cases clicking and dragging the slider will be faster. You may want to see a histogram of a ROI and pick your lower and upper limit from the histogram (figure 35:5). In order to do that, pick a ROI shape and click and drag it in a 2D view area. Click on the histogram to choose your limits. If you do not want to save your threshold to a segment but intend to use it for masking with the `Paint` tool, click on `Use for masking` (figure 35:6). This will activate the `Paint` tool with your threshold limits set as the masking option.

3.1.10 Draw

2D

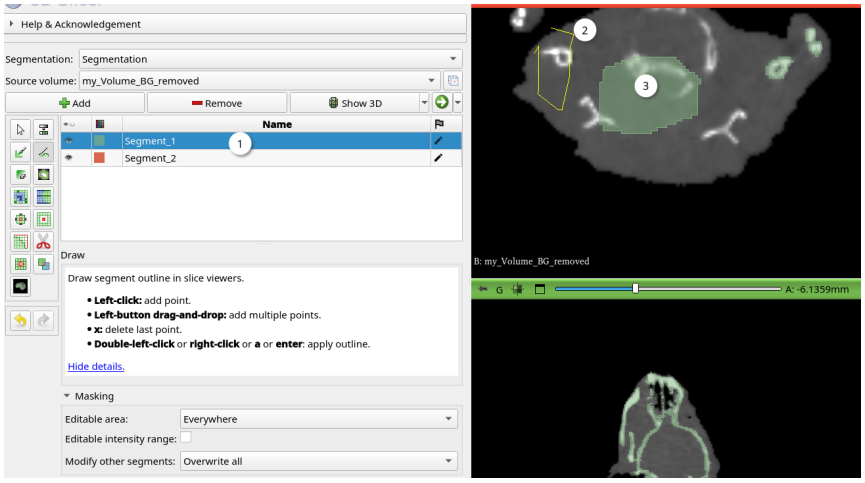


Figure 37: Draw tool

Simple drawing tool. Make sure the correct segment is selected (figure 37:1). Click on a 2D view to create points, double click to connect the last point with the first point. The encompassed area will be filled in (figure 37:3).

3.1.11 Level tracing

2D

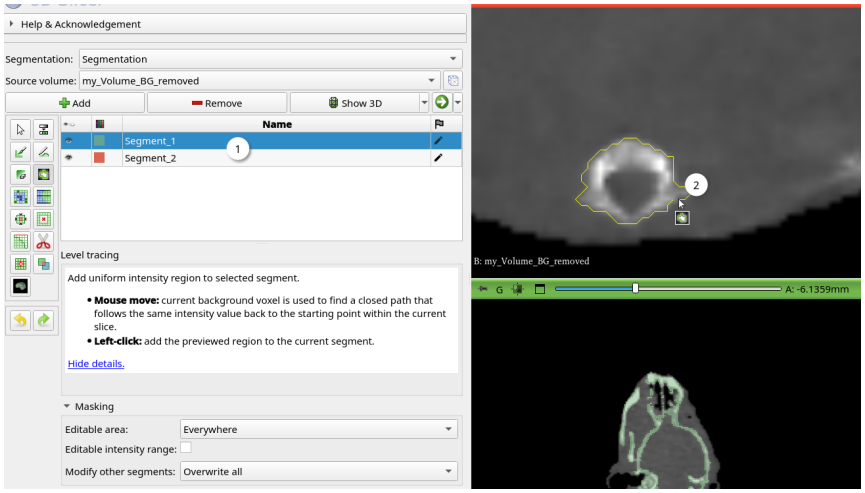


Figure 39: Level tracing tool

`Level tracing` has a similar role as the `Draw` tool (section 3.1.10). But instead of manually defining points and connecting them, the tool automatically tries to find areas of similar intensities. Hover your mouse over a 2D view to see the outline of an area, click to confirm (figure 37:2).

3.1.12 Fill between slices

2D 

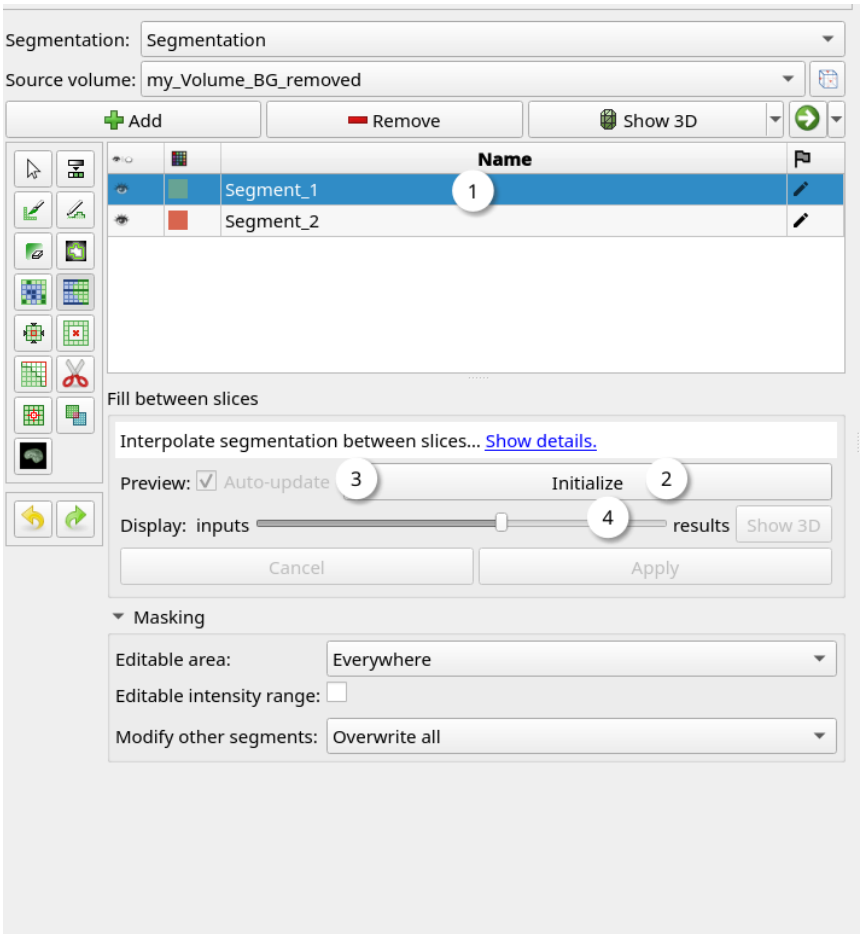


Figure 41: Fill between slices tool

`Fill between slices` is used similar to `Grow from seeds` (section 3.1.4). Start by creating at least one segment and segment a slice with a manual tool. Skip at least one slice and segment the next slice. Repeat the last 2 steps until you have covered the desired segmentation volume. Click on `Initialize` (figure 41:2) and wait for the interpolation preview to appear in the 2D view area. The first segmentation will most likely not satisfactory. Deactivate `Auto-update` (figure 41:3), switch to the `Paint` tool and manually paint over the slices with

inaccuracies. Switch back to the `Fill between slices` tool and click `Update`. Repeat until the segmentation is satisfactory.

3.1.13 Hollow

2D

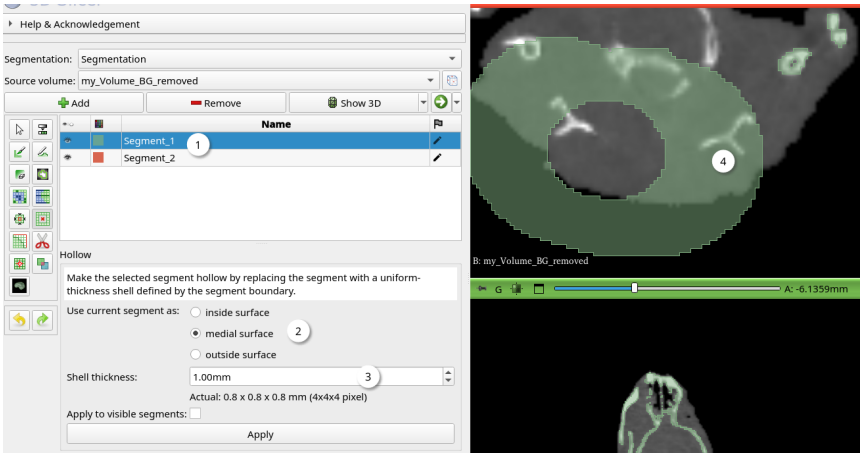


Figure 43: Hollow tool

Hollows out a segment (figure 43:4) by replacing it with a border of a specified thickness (figure 43:3). To use this tool, select the desired segment (figure 43:1), select if the segment should represent the `inside`, `medial` or `outside` border. Finally set the border thickness (figure 43:3) and click apply.

3.1.14 Scissors

2D **3D**

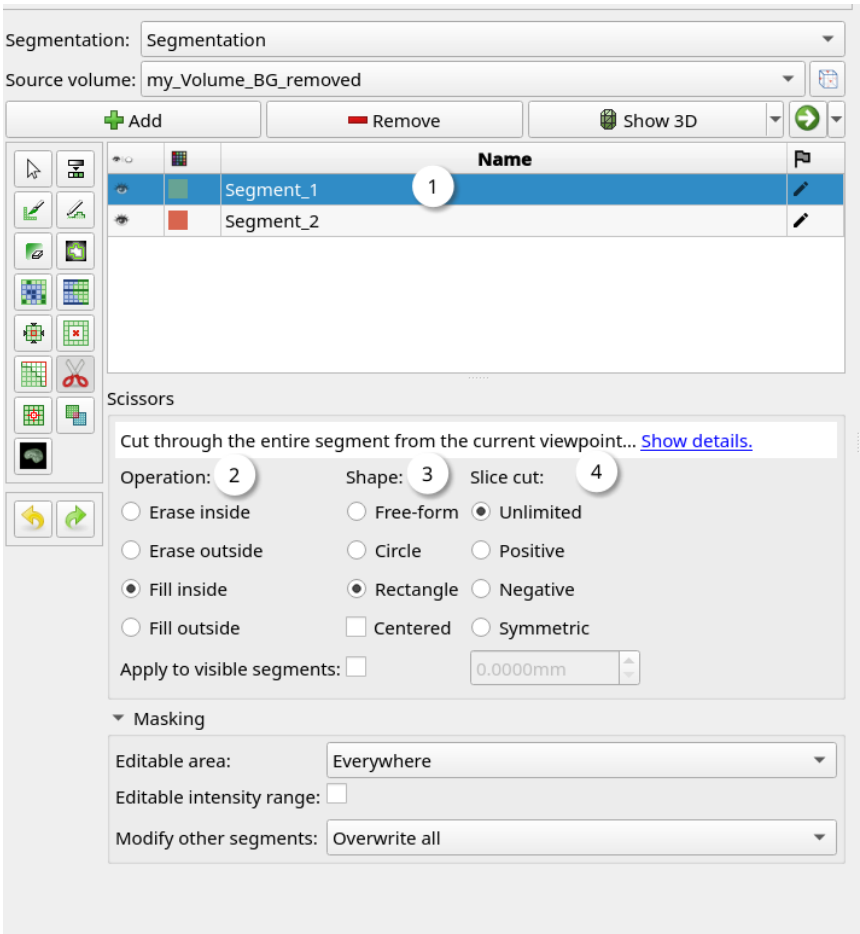


Figure 45: Scissors tool

`Scissors` can be used to cut or fill through a segment. Select a segment to cut (figure 45:1) and pick a mode of operation (figure 45:2).

Erase inside erases already segmented area inside ROI

Erase outside erases already segmented area outside ROI

Fill inside fills ROI with segment label

Fill outside fills everything but the ROI with segment label

Pick a ROI shape (figure 45:3). `Centered` means that the ROI will not be drawn from its edge but the center, this may be useful for drawing circular ROIs. Finally, choose which slices should be affected by the `Scissors` tool (figure 45:4).

Unlimited affect all slices

Positive only affect slices with a higher slice number than the current one

Negative only affect slices with a lower slice number than the current one

Symmetric affect both higher and lower slices numbers but only a specific distance

3.1.15 Logical operators

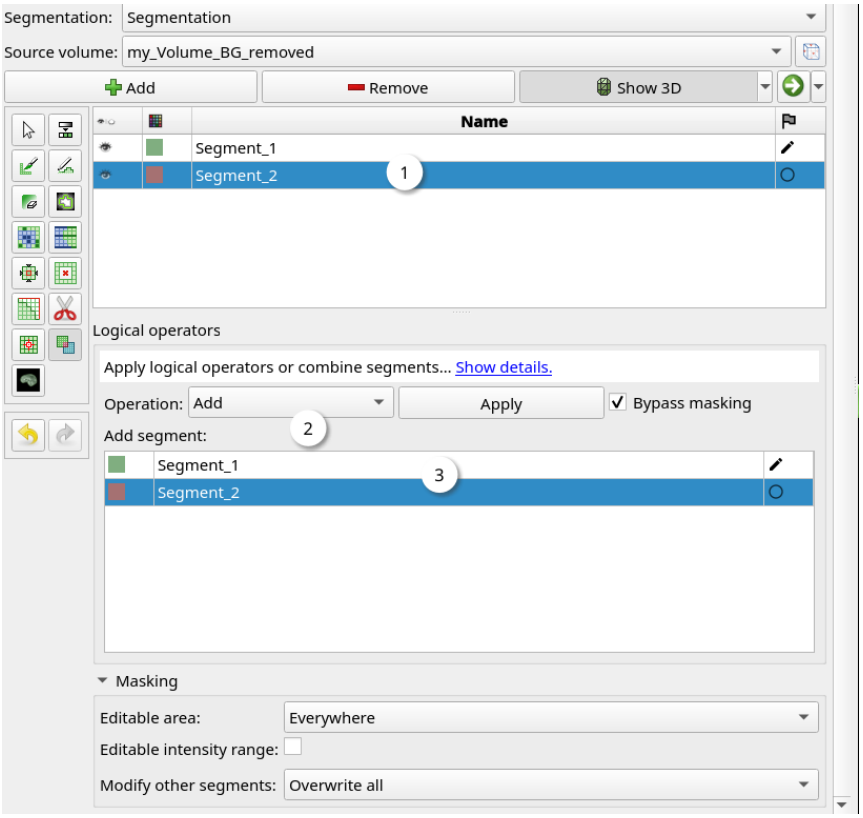


Figure 46: Logical operators tool

Boolean operators processing up to two segments at a time. First, make sure the correct segment is selected (figure 46:1). Second, pick an operation (figure 46:2).

Copy copy the segmentation of the modifier segment to the active segment, effectively replacing its segmentation

Add combine active and modifier segment

Subtract remove overlapping parts of the modifier segment from the active segment

Intersect only keep overlapping parts of the modifier segment and the active segment

Invert invert the active segment

Clear clear the active segment, the same can be achieved via the right click menu and selecting `Clear selected segments`

Fill fill the whole volume with the active segment

Finally, if applicable, pick the modifier segment (figure 46:3) and click apply.

3.1.16 Undo and Redo

Undo and Redo work the same as just about in any other software. The only caveats are that 3D slicer allows only 9 undo operations and that some operations cannot be undone (section 3.1.8)

3.2 Additional segmentation tools

Some tools are not included with the download of 3D Slicer. There can be several reasons for this. Some may not be considered stable enough yet. Some may have considerable overlap in their use case with a built-in tool. And some may just not fit 3D Slicer's development direction. Fortunately, 3D Slicer provides an API which enables the development of extensions. One such extension is "SlicerSegmentEditorExtraEffects"[5] by Andras Lasso. This extension expands the tool section in the `Segmentation Editor` module with the following tools:

1. `Draw Tube`
2. `Engrave`
3. `Fast Marching`
4. `Flood Filling`
5. `Split Volume`
6. `Surface Cut`
7. `Watershed`
8. `Local Threshold`

Figure 47: SlicerSegmentEditorExtraEffects

Install the extension by clicking on the `Extensions Manger` in the toolbar (figure 8:10). In the newly opened window, click on `Install Extensions`, then use the search bar to find "SegmentEditorExtraEffects". Click install and wait for 3D Slicer to ask you for a restart. The extra tools will be available after restarting 3D Slicer.

3.2.1 Draw tube

2D 3D

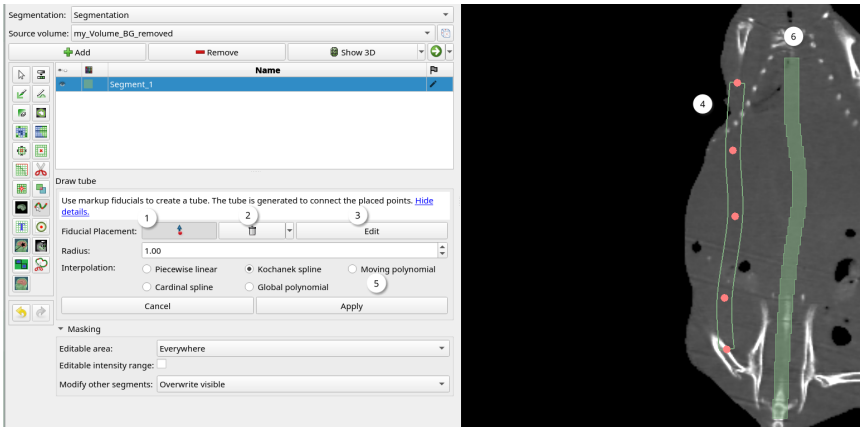


Figure 49: Draw tube tool

Draws a tube of fixed radius along a path of points. First, make sure the desired segment is selected. Click on `Place a control point` (figure 49:1) to start drawing. Create as many control points as you need by clicking on your volume in the 2D or 3D views. Exit the point creation mode by double left-clicking on the last point or right-clicking anywhere in the view area. The points can now still be moved freely by clicking and dragging them around. If you wish to delete the last added point, click on the trashcan icon (figure 49:2). Pick a tube radius in millimeters and an interpolation algorithm (figure 49:5) by experimenting with the available options and reviewing changes in the preview (figure 49:4). Click `Apply` to confirm your changes (figure 49:6). The result however can still be changed by clicking `Edit` (figure 49:3).

3.2.2 Engrave

2D 3D

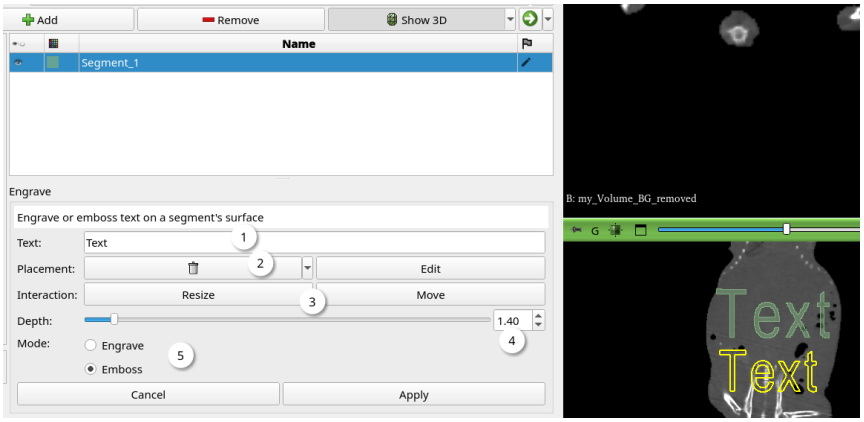


Figure 51: Engrave tool

Draws or carves out text on a segments surface. First insert your desired text string (figure 51:1). Next, click on `Place a control point` (figure 51:2) and place your text by clicking on the desired position in either the 2D or 3D view area. Change size and location by selecting either (figure 51:3) and manipulating the text in the view area. Change the depth of penetration in millimeters by entering a value or dragging the slider (figure 51:4). Finally choose an operation mode (figure 51:5):

Engrave carve the text out of an existing segment, requires prior segmentation

Emboss stamp the text on a segment, requires no prior segmentation

3.2.3 Fast Marching

2D 

Works similar to `Grow from seeds` (Section 3.1.4) with the big difference that it does not need a "Background" segment and can be used on a single segment at a time. `Fast Marching` is also considerably faster than `Grow from seeds` `Maximum volume` refers to relative amount of your dataset `Fast Marching` will try to segment. Control segmentation leakage via the `Segment volume` slider.

3.2.4 Flood Filling

"Add to the current segment all similar intensity voxels near the clicked position. Generally "Local threshold" effect is recommended instead of this effect because this effect often either cannot prevent leaking into other structures or provides incomplete segmentation."[5] This tool has been found to be unreliable and tends to crash often.

3.2.5 Split Volume

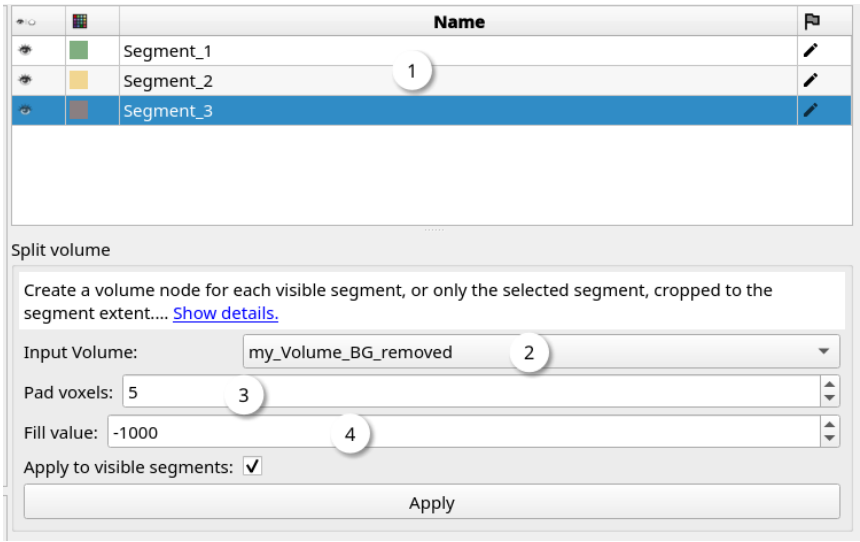


Figure 55: Split volume tool

Creates a new volume for each segment and requires at least two existing segments. Start by segmenting parts of your dataset that you wish to split (figure 55:1). Make sure to select the correct input volume (figure 55:2). Set the amount of padding voxels around each segment (figure 55:3) and the fill value outside the segmentation (figure 55:4). After clicking `Apply` the `Data` module will show some new volumes according to the number of segments you split.

3.2.6 Surface Cut

2D 3D

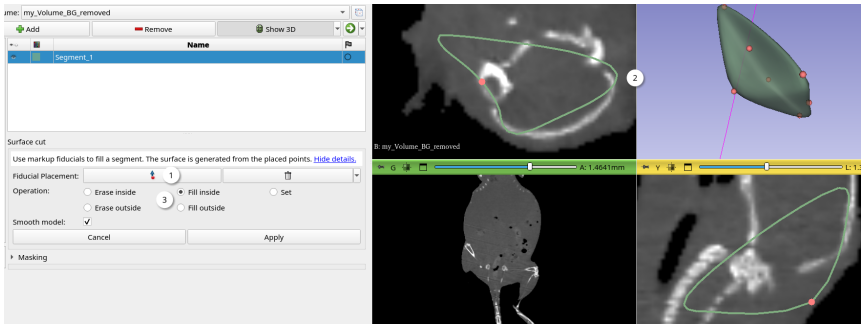


Figure 57: Surface cut tool

Works similar to `Draw tube` (section 3.2.1) and `Scissors` (section 3.1.14). Start by creating fiducial markers (figure 57:1) in the 2D or 3D views (figure 57:2). Exit fiducial creation mode by either double-clicking on the last fiducial or right-clicking anywhere in the view area. Modify fidicual placement by clicking and dragging them to the desired position. Choose an operation mode:

Erase inside erases already segmented area inside fiducial volume

Erase outside erases already segmented area outside fiducial volume

Fill inside fills fiducial volume with segment label

Fill outside fills everything but the fiducial volume with segment label

Set fills fiducial volume with segment label

3.2.7 Watershed

2D 

Works almost identical to `Grow from seeds`, with the exception that segmentation smoothing factor can be customized via the `Object scale` setting. For usage see: Section 3.1.4.

3.2.8 Local Threshold

2D 

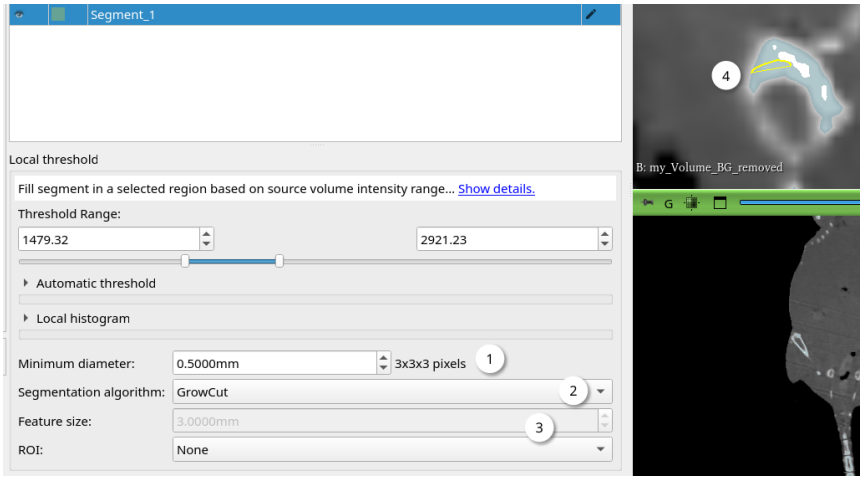


Figure 60: Local threshold tool

ROI based threshold tool. Start by drawing a ROI in the 2D view area (figure 60:4). The segmentation algorithm (figure 60:2) will automatically set the lower and upper threshold limit. If the segmentation is leaky, reduce the `Minimum diameter`. This will prevent the segmentation algorithm from considering bridged areas with bridges smaller than the specified amount. If the watershed algorithm is selected (figure 60:2), `Feature size` can be used to control the amount of smoothing the algorithm performs. If you wish to locally restrict the threshold, create and draw a ROI (figure 60:3) before using the tool.

4 Performing segmentations

Segmenting is now a matter of applying a selection of the tools mentioned above in a meaningful order. This guide will focus on bone segmentation as this was the authors use case, but the general workflow should apply to most anatomical structures. The next two sections will show you possible workflows. If you already face performance issues when loading the dataset, start by following the instructions in section 2.3.

4.1 Manual

1. **Identify the target structure** Navigate to your target structure using the 2D views or visualize it in the `Volume Rendering` module.
2. **Isolate the target structure** Crop out as much unneeded volume as possible using the method described in section 2.3.1. This will not only improve performance, it will also make navigation easier.
3. **Generate a mask for the structure** Using the threshold tool (section 3.1.9) generate a mask for your target structure, in case of this guide bone structures. This will make it almost impossible to accidentally segment unwanted tissue.
4. **(Optional) close holes and smooth edges** If the threshold tool left some holes throughout your target structure it is possible to fix this using a `Closing` operation (section 3.1.5).
5. **Cut away unwanted parts** Use the `Scissors` tool (section 3.1.14) in 2D or 3D view to cut away unwanted structures from the mask segment.
6. **Create segments** Create a segment for each structure that is going to be segmented and name them accordingly (figure 61:1).
7. **Manually separate structures** Use manual tools like `Draw, Paint, Draw tube` to separate elements. Overwriting the mask segment created in the third step. This can be achieved by selecting the mask segment as the `Editable area` in the `Masking` options (figure 16:6, figure 61:3). Make sure to separate your target structure carefully in all planar views (figure 61:4).
8. **Connect remaining islands** Use the `Islands` tool (figure 61:2) to connect the middle segment of the last step, effectively finishing the segmentation.

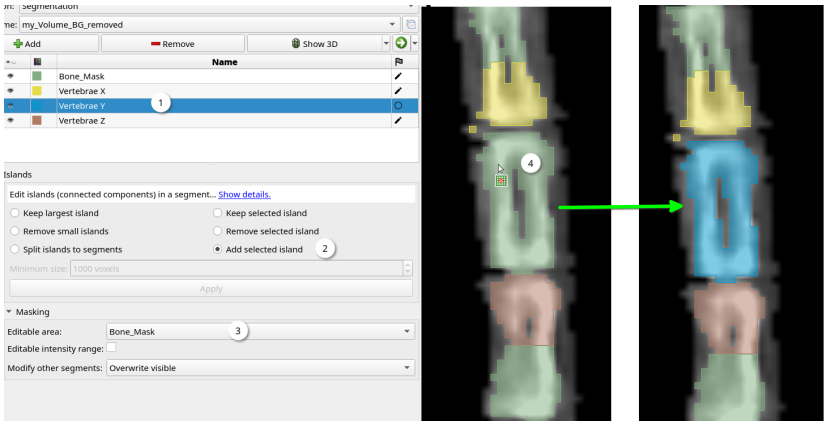


Figure 61: Manual segmentation via islands

4.2 Semi-Automatic

1. **Identify the target structure** Navigate to your target structure using the 2D views or visualize it in the `Volume Rendering` module.

2. **Isolate your target structure** Crop out as much unneeded volume as possible using the method described in section 2.3.1. This will not only improve performance, it will also make navigation easier.

3. **Determine threshold for masking** Using the threshold tool (section 3.1.9) generate a mask for your target structure, in case of this guide bone structures. Alternatively just use the threshold tool to determine the upper and lower limits and use them in the masking tab of the chosen semi-automatic segmentation tool.

4. **Create segments** Create a segment for each structure that is going to be segmented and name them accordingly. If you plan to use `Grow from seeds` (section 3.1.4) also create a background label.

5. **Paint seeds** Manually paint seeds inside the structures you wish to segment. Make the seeds evenly spread throughout the structure and add extra seeds where structures border. When using `Grow from seeds` also do this for the background segment.

6. **Perform semi-automatic segmentation** Let the tool calculate a segmentation based on your seeds. Improve the segmentation by going back to step 5 and adding more seeds. Repeat until the result is sufficiently accurate.

References

[1] T. D. DenOtter and J. Schubert, "Hounsfield Unit," in *StatPearls*, Treasure Island (FL): StatPearls Publishing, 2024. pmid: 31613501. [Online]. Available: http://www.ncbi.nlm.nih.gov/books/NBK547721/ (visited on 07/09/2024).

[2] R. Kikinis, S. D. Pieper, and K. G. Vosburgh, "3D Slicer: A Platform for Subject-Specific Image Analysis, Visualization, and Clinical Support," in *Intraoperative Imaging and Image-Guided Therapy*, F. A. Jolesz, Ed., New York, NY: Springer New York, 2014, pp. 277–289, ISBN: 978-1-4614-7656-6. DOI: 10.1007/978-1-4614-7657-3_19. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-7657-3_19 (visited on 09/17/2023).

[3] ValveSoftware, *OpenVR SDK*, version 2.5.1, Washington: Valve Corporation, Mar. 27, 2024. [Online]. Available: https://github.com/ValveSoftware/openvr (visited on 07/07/2024).

[4] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, *et al.*, "3D Slicer as an image computing platform for the Quantitative Imaging Network," *Magnetic Resonance Imaging*, vol. 30, no. 9, pp. 1323–1341, Nov. 2012, ISSN: 0730725X. DOI: 10.1016/j.mri.2012.05.001. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0730725X12001816 (visited on 02/08/2024).

[5] A. Lasso, *SlicerSegmentEditorExtraEffects*, PerkLab, Mar. 5, 2024. [Online]. Available: https://github.com/lassoan/SlicerSegmentEditorExtraEffects (visited on 09/23/2024).

A Appendix: Code

A.1 distrobox manifest file

```
[slicerbox]
image=docker.io/library/fedora:latest
# Xorg/GUI
additional_packages="itk mesa-libGLU libglvnd alsa-lib alsa-utils freetype libSM pulseaudio-libs-glib2 libXcomposite libGL libXdamage libXrandr libXrender libXft libxkbcommon-x11 libXxf86vm libXext libXcursor libXi libXtst libX11"
# data manipulation
additional_packages="dcmtk libxcrypt vtk teem"
# QT5
additional_packages="qt5-qtbase qt5-qtbase-common qt5-qtbase-gui qt5-qtdeclarative qt5-qttranslations qt5-qtwayland qt5-qtxmlpatterns"
# Plugin Support
additional_packages="libffi libnsl nss"
# AI support
additional_packages="conda python3-tinygrad"
#export=" "
init=false
nvidia=false
pull=true
root=false
replace=true
start_now=false
```

A.2 Resource usage comparison

```
# output of: ps_mem -S -p $(pidof SlicerApp-real)


# 3D Slicer without any dataset loaded
Private  +   Shared  =  RAM used   Swap used        Program

416.0 MiB +   2.2 MiB = 418.2 MiB     0.0 KiB        SlicerApp-real
---------------------------------------------
                        418.2 MiB     0.0 KiB
=============================================

# 3D Slicer with the full dataset loaded
Dataset: 309M    ./pre-Scene.mrb

Private  +   Shared  =  RAM used   Swap used        Program

1.3 GiB +   6.0 MiB =   1.3 GiB     0.0 KiB        SlicerApp-real
---------------------------------------------
                          1.3 GiB     0.0 KiB
=============================================


# 3D Slicer with the reduced dataset loaded
Dataset: 15M     ./post-Scene.mrb

Private  +   Shared  =  RAM used   Swap used        Program

579.4 MiB +   6.0 MiB = 585.4 MiB     0.0 KiB        SlicerApp-real
---------------------------------------------
                        585.4 MiB     0.0 KiB
=============================================
```

B Appendix: Shortcuts used in this Guide

Table 1: 3D Slicer shortcuts

`right click + mouse drag` (up/down)	zoom in and out
`Ctrl + mouse wheel`	zoom in and out
`middle click + mouse drag`	pan/translate view
`left arrow/right arrow`	show previous/next slice
`up arrow/down arrow`	show previous/next slice
`b/f`	show previous/next slice
`Shift + mouse move`	move crosshair in all views/ show ROI in all views
`left click + v`	toggle slice visibility in 3D view
`b`	reset zoom and pan
`g`	toggle segmentation visibility
`Ctrl + o`	add/load data
`Ctrl + s`	save data to file
`Ctrl + w`	close scene
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view
middle click and mouse drag	pan/translate view