Fractional labelmap representation of anatomical structures

by

Kyle Remington Sunderland

A thesis submitted to the
School of Computing
in conformity with the requirements for
the degree of Master of Science

Queen's University
Kingston, Ontario, Canada
September 2017

Abstract

INTRODUCTION: In medical imaging software, structures are often represented using image volumes known as labelmaps. Conversion of structures to labelmap representations results in a loss of information which impacts medical imaging algorithms, including those in radiation therapy (RT) treatment planning. RT treatment planning systems are used to optimize radiation delivery to structures, which are represented as planar contours. Errors from conversion affect metrics such as dose volume histograms (DVHs) that are the primary metric for RT plan optimization. The goal of this thesis is to develop fractional labelmaps as a structure representation that preserves more structural information and reduces conversion errors. **METHODS:** The effect of voxelization on treatment planning metrics was tested by comparing DVHs calculated using varying voxel sizes. An algorithm was implemented that triangulates planar contours to closed surface mesh and handles features such as branching, keyhole contours, and end-capping. Closed surfaces were converted into fractional labelmap representations, where each voxel represents occupancy between 0% and 100%. Existing segmentation methods were modified to allow fractional labelmaps to be edited directly. **RESULTS:** Algorithms were implemented in the open-source medical imaging platform 3D Slicer[1], and the SlicerRT[2] toolkit. Voxel size was found

[1] slicer.org
[2] slicerrt.org

to affect DVH accuracy for structures with small features or in regions with a high dose gradient. The planar contour to closed surface triangulation algorithm produced qualitatively good surfaces. Fractional labelmaps from closed surfaces were found to be up to 19.1% better at representing structure volume than binary labelmaps, with an average improvement of 6.8%. DVHs calculated from fractional labelmaps were found to be more accurate than DVHs calculated using binary labelmaps. Fractional segmentation methods were found to create good quality segmentations. **CONCLUSION:** Labelmap voxel size was found to be a contributing factor for DVH accuracy. Accurate conversion algorithms were implemented for planar contour to closed surface and closed surface to fractional labelmap conversions. When used for structure representation and DVH calculation, fractional labelmaps were more accurate than binary labelmaps at the same resolution. Fractional labelmaps were found to be an effective tool for structure representation in radiotherapy that could be expanded to other use cases.

Statement of Co-Authorship

This work was completed under the supervision and mentorship of Dr. Gabor Fichtinger, Csaba Pinter, and Dr. Andras Lasso. The research presented in this thesis was published in the following publications:

- "Reconstruction of surfaces from planar contours through contour interpolation", Kyle Sunderland, Boyeong Woo, Csaba Pinter, Gabor Fichtinger (SPIE Medical Imaging 2015) [1]
- "Effects of voxelization on dose volume histogram accuracy", Kyle Sunderland, Csaba Pinter, Andras Lasso, Gabor Fichtinger (SPIE Medical Imaging 2016) [2]
- "Analysis of dose volume histogram deviations using different voxelization parameters", Kyle Sunderland, Csaba Pinter, Andras Lasso, Gabor Fichtinger (14th Imaging Network Ontario Symposium) [3]
- "Fractional labelmaps for computing accurate dose volume histograms", Kyle Sunderland, Csaba Pinter, Andras Lasso, Gabor Fichtinger (SPIE Medical Imaging 2017) [4]

Acknowledgments

I would like to begin by thanking Dr. Gabor Fichtinger for his mentorship, encouragement and dedication. I have relied on and appreciated your guidance and direction throughout my time in the Perk Lab, both as an undergraduate and graduate student. I am grateful for the opportunities you have presented to me. Thank you for believing in me and seeing my potential.

I would also like to express my thanks to Csaba Pinter for providing feedback and direction for my projects in the Perk Lab. Csaba, without your help and patience I would not be the programmer that I am today. I will apply the lessons that you have taught me to continue to improve my programming skills.

I appreciated the insight and assistance that Dr. Andras Lasso has provided to me. Andras, you always set me on the right course. When I was confused, or unsure of how to proceed, you were always able to direct me towards the solution.

I am gratefully to my family for their love and encouragement, and for being there when I became overwhelmed. Thanks to my friends and colleagues in the Perk Lab for your support and friendships.

Thank you all.

Contents

List of Tables

List of Figures

Glossary

RT Radiotherapy. i, 1–3, 5, 7, 8, 13, 15, 16, 18, 44, 68–70

TPS Treatment Planning Software. i, 2, 4–8, 10–13, 15, 16, 18, 36, 44, 63, 68, 70

DVH Dose Volume Histogram. i, ii, vii, ix, x, 2–7, 10, 11, 14, 15, 17–20, 36, 37, 40, 44–48, 50–52, 62, 63, 65, 66, 70, 71

DSC Dice similarity coefficient. vii, 38, 39, 44–46, 62, 68, 71

CT Computed Tomography. ix, 8, 9, 14, 26, 48

EBRT External Beam Radiation Therapy. 2, 20

TCP Tumour Control Probability. 3, 4

NTCP Normal Tissue Complication Probability. 3, 4

DICOM Digital Imaging and Communications in Medicine. 8, 42

V95 Percent of a structure's volume that receives 95% of the desired amount of radiation. 10

D5 The minimum dose that is received by 5% of a structure. 11

GPU Graphics Processing Unit. 32, 59, 68

GLSL OpenGL Shading Language. 33, 59, 60

VD Volume Difference. 37, 46, 48, 63

DTA Dose to Agreement. 37, 46, 48, 63

GTV Gross Tumour Volume. 45, 46, 48, 50, 52, 53, 56–58

PTV Planning Target Volume. 45, 46, 48, 50, 52, 53, 56–58

CTV Clinical Target Volume. 45, 46, 48, 50, 52, 53, 56–58

Chapter 1

Introduction

1.1 Background

In medical imaging software, anatomical structures are commonly represented using 3D image volumes, referred to as labelmaps. Within a labelmap, each cubic voxel contains a value that indicates if the structures are present inside the voxel. This method of representation can be used successfully for large structures, however issues can arise when representing structures containing features with sub-voxel thickness. Converting these structures to a labelmap representation causes large regions of false positive and false negative classifications within the voxels. These classification errors result in volume differences that can represent a large fraction of the total structure volume. The errors caused by these differences can have negative implications for many different medical imaging applications. The focus of this thesis is on the effect and mitigation of labelmap representation issues with respect to radiation therapy (RT) treatment planning.

1.1.1 Radiation Therapy

Cancer is a disease that is expected to be diagnosed in approximately 206,200 individuals in Canada during 2017, and is predicted to cause approximately 80,800 deaths [5]. The term cancer refers to a number of diseases that are classified by the abnormal division and invasive expansion of cells. As the affected cells continue to divide and grow, they form large growths of cancer cells known as tumours. Rapid tumour growth disrupts normal tissue function for the regions of the body in which they grow, causing many negative side effects. If left untreated, the cancer can spread to critical regions of the body, which may eventually result in death.

In order to treat various types of cancer, one common method is the use of external beam radiation therapy (EBRT). X-ray radiation is generated in a beam using a linear accelerator which rotates on a gantry around the patient's body, administering a level of radiation to the tumour that will kill the cancer cells (see Figure 1.1). The goal of EBRT treatment is to deposit a high cumulative dose of radiation that is lethal to the tumour (and a small margin), while minimizing the amount that is absorbed by the surrounding normal tissue. In order to achieve the best possible outcome, plans are developed and optimized prior to administration using RT treatment planning systems (TPSs).

1.1.2 Dose Volume Histogram

Dose volume histogram (DVH) is a metric used to represent the distribution of radiation within segmented structures (see Figure 1.2). Cumulative DVH represent the minimum amount of radiation that is received by a proportion of the structure's volume. The x-axis represents the amount of radiation, either in Gy or as a percentage

Figure 1.1: Woman Prepared for Radiation Therapy (Michael Anderson - National Cancer Institute / Wikimedia Commons / Public Domain)

of the total desired radiation dose. The y-axis represents the volume of the structure, either in volume units, or a percentage of the total structure volume.

When analyzing the optimality of an RT treatment plan, DVHs are often used as a metric for measuring how effective a plan is at depositing dosage in the correct locations, and minimizing the effect on healthy tissue. Drzymala et al. worked to evaluate the effect that differences in DVHs had on metrics such as tumour control probability (TCP), and normal tissue complication probability (NTCP) [6]. In order to evaluate the potential effects of DVH accuracy with respect to TCP and NTCP

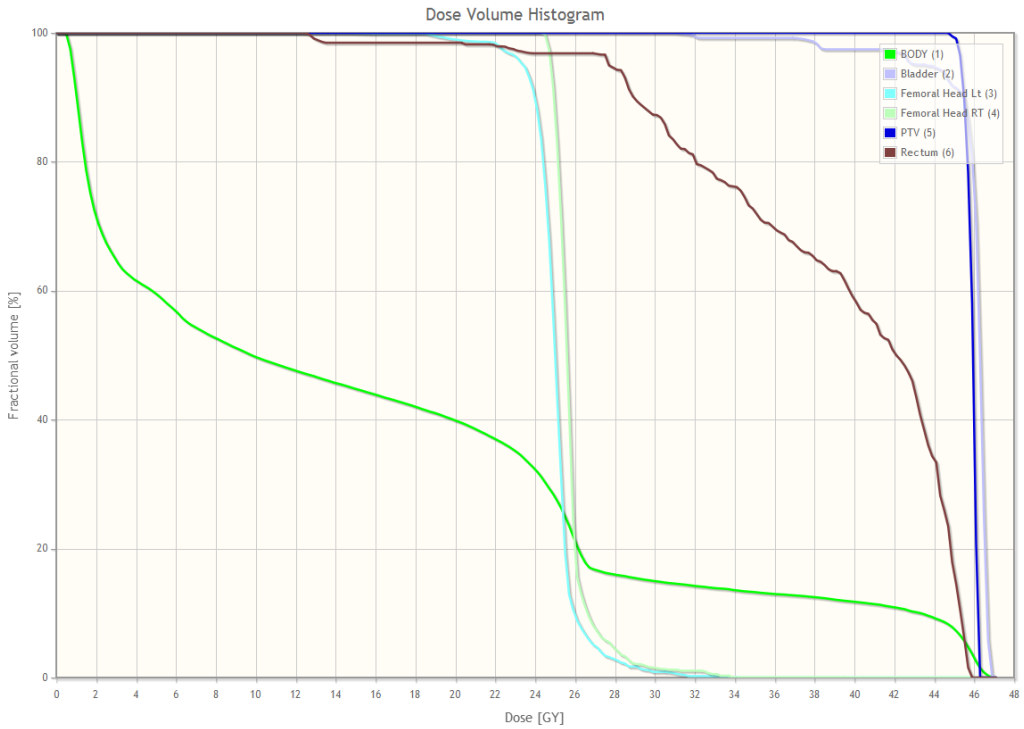

Figure 1.2: DVH calculated for RANDO^{TM} prostate phantom structures

metrics, they provided a concentric cubic phantom and dose volume to four different institutions. They used the phantom structures and dose volume to create DVHs using their own TPS methods. They found that relatively small differences between DVHs could result in significant differences in the derived TCP and NTCP metrics, which could result in sub-optimal plans being selected over more optimal ones. As a result of their findings, they stressed that it is important that DVHs should not to be used as the sole determining factor when comparing plans, and should be supplemented by additional metrics to ensure that process of selecting the best plan is as accurate as possible.

To ensure the quality of radiotherapy TPS, the American Association of Physicists in Medicine commissioned a task group to identify all of the features that should be

taken into consideration when developing quality assurance program for RT TPS [7]. One particular section of their report detailed the factors involved in the calculation of DVH that should be analyzed through quality assurance to ensure RT treatment plan accuracy. Some of the factors listed were: volume region of interest identification, structure identification, voxel dose interpolation, structure volume, histogram bins and limits, DVH calculation, DVH types, DVH plotting and output, plan and DVH normalization, dose and volume region of interest grid effects, use of DVHs from different cases, as well as other factors. Included with their list was a summary of the ways in which errors in the various factors could impact the accuracy of the results. In addition to their analysis of the factors affecting DVH calculation, they also analyzed a large number of other potential factors throughout the treatment planning process. Their conclusions were that a quality assurance plan would be an important step in ensuring effectiveness of patient treatment, but that the task of implementing these procedures was large and unfinished.

In order to evaluate the accuracy of DVH, there must be a ground truth that can be used as a reference for comparison. Nelms et al. developed a method for evaluating the accuracy of DVHs calculated from TPS by comparing them to exact DVHs constructed using simple analytical shapes and doses [8]. They created structures such as spheres, cones and cylinders in different orientations using implicit mathematical functions, and created a dose volume consisting of a simple 1 dimensional gradient. In order to calculate DVHs for these structures, the mathematical functions for the structures were integrated with the dose gradient to calculate the dose distribution for the structure volume, providing an exact solution Various DVHs were calculated using commercial TPSs, such as Pinnacle^3 (Philips, Eindhoven, North

Brabant, Netherlands) and PlanIQ (Sun Nuclear Corporation, Melbourne, Florida, USA) by sampling the analytical functions into contours at varying thicknesses. The DVHs from these TPSs were then compared to the analytically derived DVHs by comparing how far each point was from the analytical solution. They found that there were several errors introduced in the TPS DVH calculation process that should be easily avoidable. In the Pinnacle software system, they found that errors were caused when voxels created by end-capping in the contour conversion process were included in the structures total volume, but were not utilized in the DVH calculations. They also found that the Pinnacle DVH suffered due to a lack of supersampling in the calculation process, causing the points in the DVH to be offset from the true value. The authors were able to demonstrate that their system was an effective method for evaluating the accuracy of various TPSs.

Differences in structure volume is a problem that can contribute significantly to the accuracy of DVHs. Ackerly et al. conducted research on the observation that the volumes of structures created from the same contours varied between different TPSs [9]. The authors first segmented structures into contours using Cadplan (Varian Medical Systems, Palo Alto, CA, USA), and then exported the results. They used other TPSs to calculate the volumes of the exported contours, and found that they had significantly different reconstructed volumes when compared to the Cadplan results. It was discovered that since Cadplan does not factor in slice thickness for contours contained on the first or last slices in the structure, there was a loss of volume in all structures, while the other TPSs took this into account by extending their contours by $\frac{1}{2}$ slice thickness. These results showed that different TPSs can calculate different volumes for the same structure, and that volume uniformity cannot be assumed.

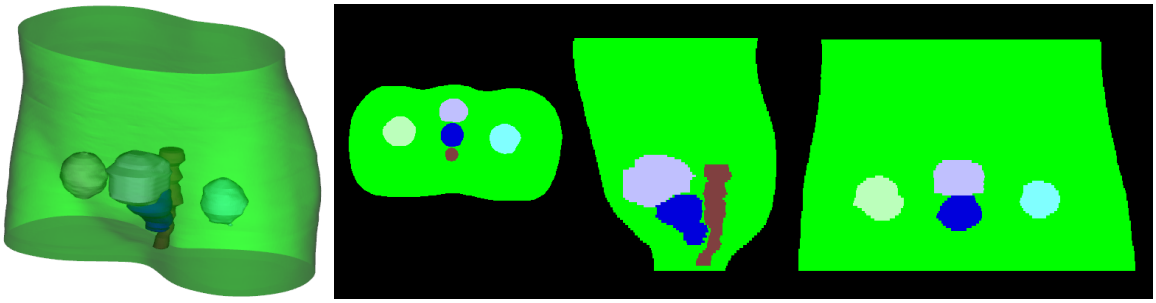

Figure 1.3: Closed surface mesh representing a RANDO^{TM} prostate phantom (left), and axial, saggital, and coronal slices from the corresponding binary labelmap (right)

In commercial TPSs, voxel size for structure volumes is a factor that can have a significant impact on the accuracy of DVH. Corbett et al. researched the effect of voxel size on DVHs derived in the treatment of prostate seed implants [10]. They evaluated the effects of voxel size by first randomly selecting five patients who had undergone seed implantation. The patient records containing prostate contours and seed coordinates were used to calculate dose distributions with a range of voxel sizes from 0.5mm to 5mm. They found that when using larger voxels, the volume of the structure that receives 200% of the target dose was 30-43% larger than the same structure at small voxel resolution. For a voxel size of 1mm x 1mm x 1mm, the DVH were found to be $\pm$5% of the expected value.

1.1.3 Voxelization

Voxelization, similar to its 2D counterpart, rasterization, is the process of converting a 3D structure into an image volume. With binary voxelization, voxels in the resulting image can be one of two states: inside or outside the structure (see Figure 1.3). This method of conversion is used by RT TPSs to quickly construct labelmaps from 3D structure representations.

In many TPSs, binary labelmaps are often stored using 1 byte of memory for each voxel in the image volume. This means that out of the 8 bits in each voxel, only one is actually used to represent the structure. In order to utilize this memory storage effectively, the remaining 7 bits could be used to store the fraction of each voxel that is occupied by the structure, instead of a binary voxel classification. This method of representation, known as fractional labelmap representations, have previously been used for functions such as automatic image segmentation of brain tissue [11] [12]. These methods have returned different types of fractional segmentations: the probability that a voxel is filled with a particular tissue classification type [11], or the probability that a voxel contains abnormal white matter tissue [12].

The DICOM standard defines labelmap representations in the segmentation (SEG) data-type, and contains a property for determining if labelmap classifications are binary or fractional (PS3.3: 2017b - DICOM Information Object Definitions C.8.20.2 Segmentation Image Module [1]) [13]. This standard includes several other properties, such as occupancy or probability classifications for fractional labelmaps, data format, and the maximum scalar value.

1.2 Motivation

In the Digital Imaging and Communications in Medicine (DICOM) standard, radiation oncology structure segmentations are stored as a series of parallel contours segmented from parallel computed tomography (CT) scan slices (see Figure 1.4). As a result, computational algorithms are implemented in RT TPSs which perform the conversion from a set of contours into an image volume known as a binary labelmap

[1] http://dicom.nema.org/medical/DICOM/current/output/chtml/part03/sect_C.8.20.2.html

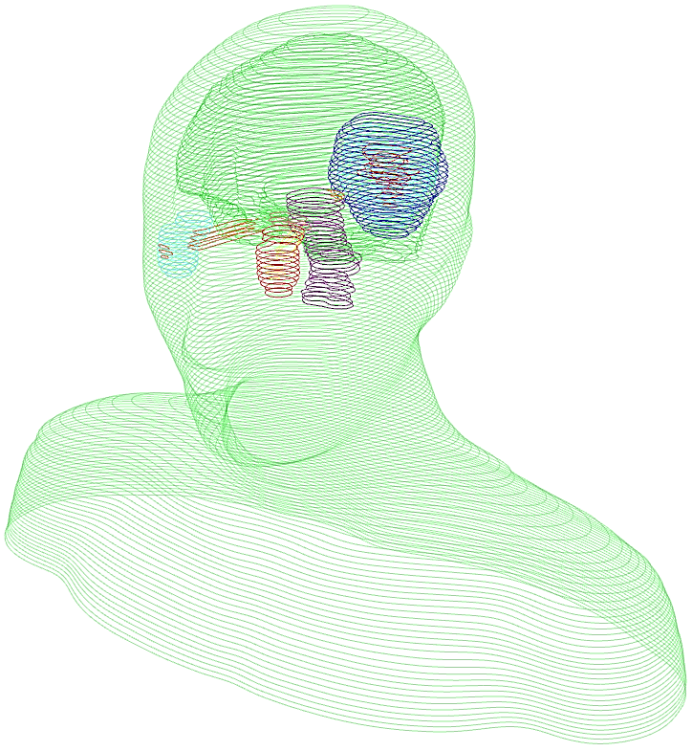

Figure 1.4: RANDO® ENT structures segmented from CT image slices into stacks of parallel contours

(see Figure 1.5). As an intermediate step, this conversion process often utilizes a triangulation algorithm to create a closed surface mesh from the contours, which interpolates between the slices and allows for smoother surface visualization and voxelization. This triangulation step is necessary since planar contours only represent structure boundaries within the slice plane, and contain no information about structure geometry in the out-of-plane axis. If the structures are not interpolated, the closed surface representation would have a jagged, staircase-like appearance between slices and would be less accurate when voxelized. Triangulation of the contours into a closed surface mesh solves this issue and approximates the original structure geometry by linearly interpolating the surface between adjacent slices.

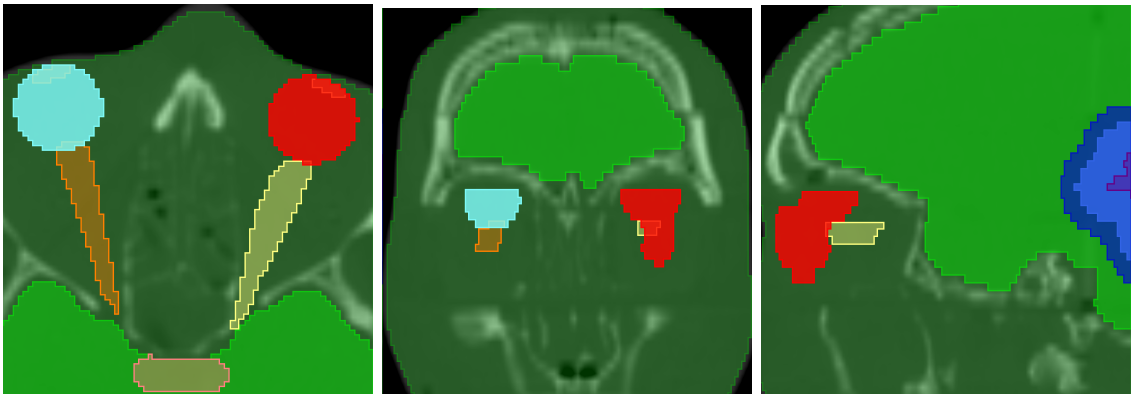

Figure 1.5: Binary labelmaps representing the structures shown in Figure 1.4

Even with an intermediate triangulation, the process of voxelization is not perfect, which results in information being lost in the process of conversion. This is due to the existence of regions with false positive and negative classifications caused by voxel size limitations, whether converting contours directly or through intermediate closed surfaces into a binary volume (see Figure 1.6). Differences in voxelization between TPSs are exacerbated by the fact that each software package uses a different algorithm for both the contour to closed surface conversion, and voxelization processes, and their methods are not disclosed. The result of these circumstances is that the treatment plans generated by different software systems may produce labelmaps that vary. This disparity in the structure segmentation causes differences for metrics such as DVHs of the same structures, the effects of which are small in structures with large features, but have a more pronounced effect in structures containing small features.

Differences between the closed surface and the binary labelmap generated by voxelization can cause issues in metrics that are used in TPS algorithms. Structure metrics such as V95 (volume of the structure receiving at least 95% of the desired

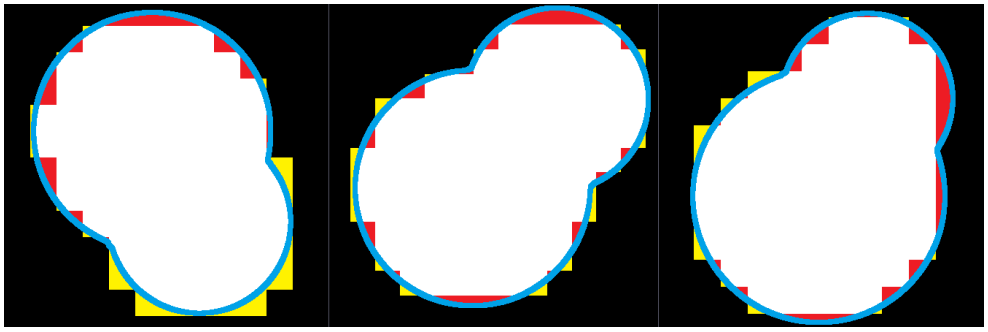

Figure 1.6: Axial, coronal and saggital views of a binary labelmap generated from a structure using a voxel size of 2.5mm showing true positive and false positive/negative regions in white/yellow/red respectively

dose) and D5 (minimum dose of radiation that is received by at least 5% of the structure) are obtained from DVHs. These primary metrics are used by inverse planning TPS, such as the one by Li et al. [14] to determine how effective a treatment plan is at irradiating the target structure, while minimizing the dose received by other more sensitive organs such as the optic chiasm and brain stem, which are some of the most closely guarded structures. If DVHs that are calculated using voxelized labelmaps are not accurate, then the metrics derived from them may be unreliable, which could influence the process of plan optimization. The current iteration of DVHs are used by the TPS to adjust dose that structures are predicted to receive, however if the DVHs are not accurate, then it could drive the plan towards a poorer solution, and may become stuck in a locally optimal plan. This issue may not be discovered before treatment begins, since the available DVH metrics would back up the current plan selection instead of a more optimal one.

The accuracy of the treatment planning process can be improved by increasing labelmap resolution and reducing voxel size to create larger binary labelmaps, which has the effect of increasing structure granularity. The trade-off is that the labelmaps

now require larger amounts of memory and computing time during both the conversion steps, and in TPS algorithm calculations. In many medical image computing research software platforms, labelmaps are stored by allocating 1 byte of memory for each voxel by default. Using a 1 byte datatype for each voxel is useful, since it allows for simpler memory access than labelmaps with a binary datatype classification.

The memory required to store labelmaps could be optimized to use 1 bit for each voxel, however another possibility is to utilize the wasted bits in each voxel by storing more structurally relevant information.

Representing structures using fractional instead of a binary labelmaps increases the amount of information that can be stored in the same amount of memory. In fractional labelmaps, the value of each voxel is representative of the fraction of the total voxel volume that is occupied by the structure. The conversion of other representation methods into fractional labelmaps requires a longer initial computation time when compared to binary labelmap conversion, but once it is complete, the TPS algorithm computation is only increased by a small constant factor. By utilizing the extra information contained in fractional labelmaps during the calculation of TPS metrics, it should be possible to increase the accuracy of the calculations, without increasing execution time. A few groups have also expressed interest in the development of a fractional labelmap representation for use in applications such as prostate segmentation for deep learning research, and atlas based segmentation of MRI.

1.3 Objective

The objective of my thesis is to develop a fractional labelmap representation for segmented anatomical structures, that allows for more of the original structure information to be preserved, while using the same amount of memory as a binary labelmap at the same resolution. In addition, this work aims to improve the overall quality of voxelized structure representation, which will be addressed through the implementation of accurate conversion methods, such as the planar contour to closed surface algorithm. The fractional labelmap representation and accurate conversion methods will be used to reduce the severity of errors introduced during the voxelization process.

In order to accomplish this, the effect of structure representation and conversion on dosimetry analysis and radiotherapy treatment planning in general will be evaluated by developing a method to quantitatively analyze how differences in structure representation can affect RT treatment plan quality. In addition, it will be useful to find a metric that can reliably gauge the amount of information that is lost during the conversion from planar contours and closed surfaces to labelmap representations. This knowledge will be utilized to mitigate the effects of contour voxelization on treatment plan accuracy.

When DICOM-RT structures are imported into commercial TPS, they are stored using planar contour representations. For these structures to be voxelized and used in RT analysis, they must first be converted into closed surface representations. In order to accomplish this, an algorithm will be implemented, that can convert planar contour representations to closed surface meshes. The algorithm will also perform pre/post-processing steps such as keyhole separation and end capping.

In order to improve treatment planning metrics, one objective is to determine how changing the size of image voxels affects plan accuracy. Since structures are typically voxelized at the same resolution as the dose volume, structures with areas of sub-voxel thickness could be negatively affected by a loss of information. By investigating how changes in voxel size influence the accuracy of treatment planning metrics, I will attempt to identify potential issues and areas of improvement in structure representation.

Due to the nature of voxel size limitations of binary labelmaps, structural information is lost during the contour conversion process. One approach to improve the accuracy of labelmap structure representation is to store more information in the same amount of space by fully utilizing any unused memory. By representing each labelmap voxel as a fractional occupancy value, instead of a binary inside/outside classification, it should be possible to increase the accuracy of structure representations, without increasing image dimensions or memory usage. This will be accomplished by creating an algorithm that can convert closed surface mesh into fractional labelmap images, storing each image voxel using 1 byte of memory. A conversion rule that can convert the fractional labelmaps back into a closed surface mesh will also be implemented.

The closed surface to fractional labelmap conversion is useful for pre-existing images, however it may be necessary to segment fractional labelmap segmentations from CT images, independent of contour and surface model representations. It is therefore necessary to implement segmentation tools that can modify fractional labelmap representations, independent of any structure conversion algorithms. I will modify the existing functions for editing binary labelmaps so that they can be used to edit fractional labelmaps in the same manner.

1.4 Contributions

Using structural analogues defined by analytical functions, I created binary labelmap segmentations and dose volumes using varying voxel sizes. I calculated and compared DVHs for labelmaps at each voxel size to determine the effect of voxel size on DVH accuracy. I evaluated the differences between the labelmaps using Dice Similarity Coefficient, Hausdorff distance, and total structure volume.

I implemented the planar contour to closed surface conversion algorithm which was integrated within the open-source SlicerRT toolkit [15], one of the most downloaded extensions for the 3D Slicer [16] medical imaging platform. The base algorithm, including steps for correspondence and branching, was implemented in Python by Boyeong Woo. I converted the algorithm from Python to C++ and optimized the execution speed so that it could be used as an automatic conversion step for users importing structures into SlicerRT. I improved the algorithm by adding additional steps to handle important features such as end-capping and keyhole separation. I also provided user support, and fixed issues that were identified due to the differences in contours generated from software by different vendors.

I implemented an algorithm to convert closed surface mesh into fractional labelmap representations. I modified the 2D segmentation visualization classes in 3D Slicer to support the visualization of fractional labelmap representation. I evaluated the accuracy of closed surface to fractional labelmap conversion by comparing the original closed surface mesh volume to the volume of binary and fractional labelmaps, at the same resolution. I tested the accuracy of fractional labelmaps for use in RT TPS by comparing DVHs calculated using fractional labelmaps and those calculated with binary labelmaps against DVHs from commercial TPS.

I implemented a conversion algorithm to convert fractional labelmaps back to closed surfaces using marching cubes. I analyzed the accuracy of closed surface mesh created from fractional labelmaps and binary labelmap surface mesh by comparing the distance to the original closed surface.

The closed surface to fractional labelmap and fractional labelmap to closed surface algorithms were integrated into the Segmentations module in the open-source 3D Slicer core. This allowed the fractional labelmap conversion methods representations to be used for many other applications beyond the RT TPS use case.

Another important feature of labelmap segmentation is the ability to use a variety of tools to create and edit segmentations from scratch, without the need for pre-existing structures. I modified the existing binary labelmap segmentation tools in the Segment Editor module in 3D Slicer to allow users to create and edit fractional labelmaps without relying on any intermediate conversion steps. Segmentation methods, such as Logical Operations, Smoothing, and Island Effects required only minimal changes, while other methods, such as Paint, Threshold, Scissor, and Draw/Level-tracing required more modifications to edit fractional labelmaps. These segmentation methods are currently being integrated into the Segment Editor module in the open-source 3D Slicer core.

Chapter 2

Methods

2.1 Effects of Voxelization

The effects of changing voxel size on the accuracy of DVH was published in a paper at the SPIE: Medical Imaging conference in 2016 [2] and in an abstract at the Imaging Network Ontario (ImNO) Symposium in 2016 [3].

2.1.1 Implicit Structures

In order to test the effect of varying voxel size on the accuracy of DVH, it is necessary to create structures which can be accurately sampled at any resolution, without relying on closed surface to binary labelmap algorithms. This is accomplished by creating structures represented using a combination of mathematical implicit functions that serve as analogues for structures segmented from a physical phantom dataset. Specifically, Boolean addition and subtraction of equations is used to combine shapes such as spheres, cylinders, and cones to construct the phantom structures. These equations are combined to resemble the structures from the RANDO® ENT phantom segmentation, however they are not intended to be an exact replica (see Figure 2.1).

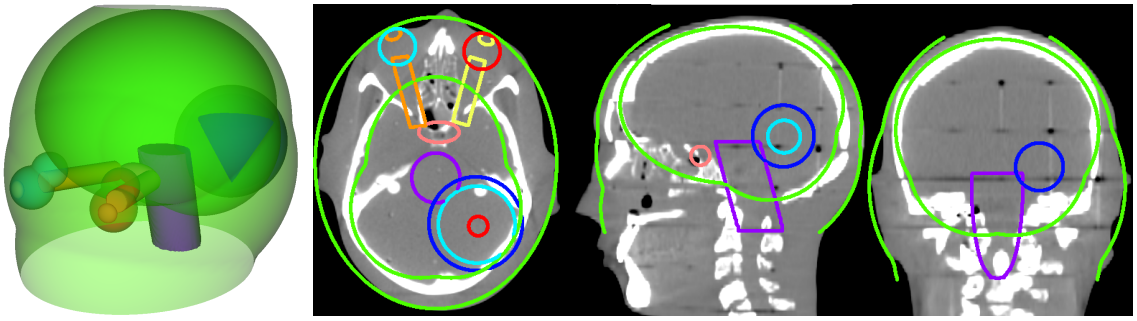

Figure 2.1: Surface mesh visualization created from the implicit structures using marching cubes

One potential concern is the existence of infinitely sharp edges in the regions of intersection between implicit functions. This will ultimately cause a slight volume difference when the implicit functions are voxelized, however the effect is minimal, and would be reduced as voxel size is lowered. Segmentations of real patient anatomy using commercial TPS can also create structures with sharp edges and complex geometry, which can also contribute to a loss of information during voxelization. Using implicit mathematical functions to define the geometry of structures has been previously used as RT phantoms [17], for testing Monte Carlo simulations for radiation dose predictions [18], and for verifying DVH calculations [8].

2.1.2 Binary Labelmap

The voxelization process using implicit functions is simpler and more accurate than the voxelization of closed surface mesh, which allows labelmaps to be constructed at a higher resolution that would typically be possible with traditional segmentations. Binary labelmaps are constructed from the implicit phantom structures by evaluating the functions at the centre of each voxel. A given voxel is considered to be inside the structure if value of the implicit function is less than or equal to zero. For voxels where

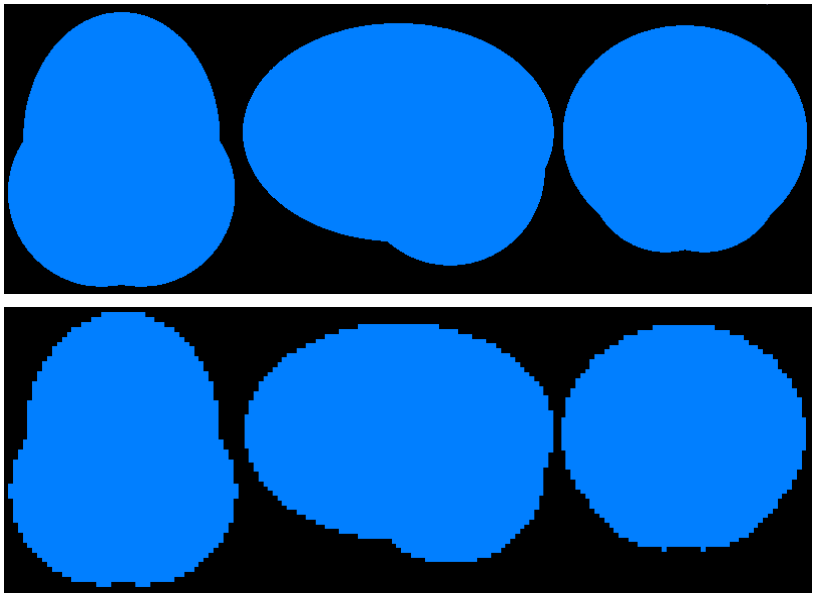

Figure 2.2: Binary Labelmaps generated using small (top) and large (bottom) voxel sizes

the value is greater than zero, the voxel is considered to be outside the structure. Each voxel in the binary labelmap is iterated through, marking voxels inside the structure with a value of 1 and outside with 0. Two types of binary labelmaps are created in this manner, one labelmap at a low resolution using voxels with a typical size of 2.5mm edges, while the other labelmap is created using the smallest possible voxel size between 0.05mm and 0.3mm depending on the total structure size (see Figure 2.2). The range of voxel sizes for the high resolution labelmap is necessary, since the amount of memory required for large structures may exceed the amount of allocatable memory if a voxel size of 0.05mm is used.

2.1.3 Dose Volume

In order to test the effect of voxel size on DVH accuracy it is necessary create a corresponding dose volume, which is an 3D image volume that represents the intensity and

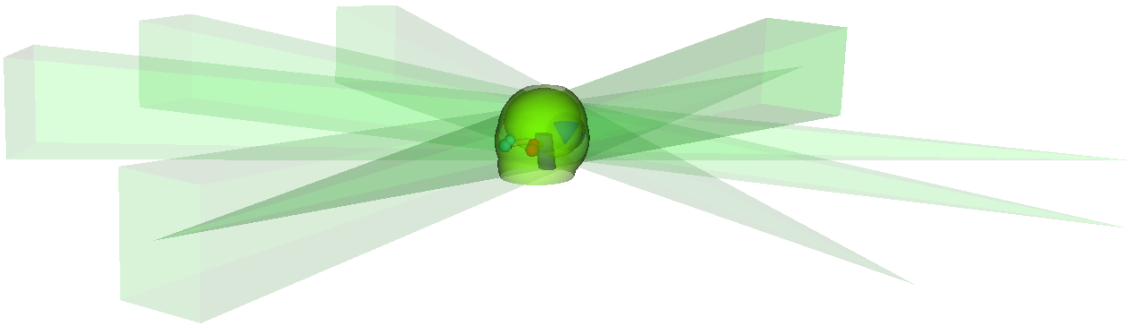

Figure 2.3: Beam layout used to generate the artificial dose volumes

spatial distribution of radiation throughout the structures. For analysis purposes, the dose volume is constructed using the same resolution and voxel size as the labelmap that is being analyzed. In doing so, the dose volumes will not need to be interpolated or resampled for use in DVH calculations, since the centres of voxels in both image volumes line up exactly.

The dose volume was created by simulating an EBRT plan that is based on an existing example. This plan was set up by arranging five beams around a simulated isocentre, using a 100cm source-axis distance and a $10cm^2$ focal region (see Figure 2.3). Dose is computed at the centre of every voxel in the dose volume by summing the contribution for each of the beams, which are calculated using the value of clinical tissue phantom ratio, cross-plane dose profile, and in-plane dose profile (see Figure 2.4). The resulting dose volume was not intended to be a perfect reproduction of a clinical plan and was merely an approximation that could be calculated at any resolution.

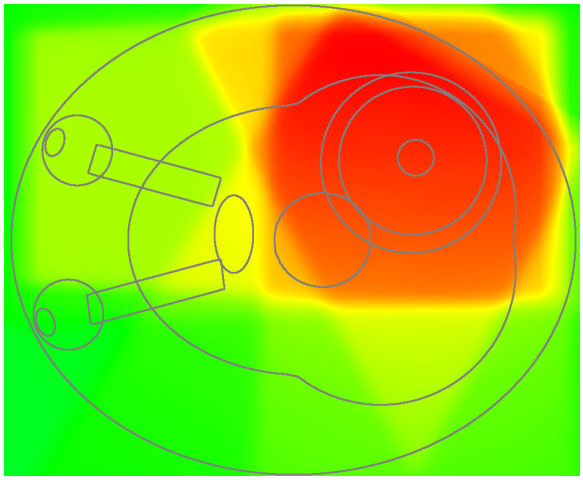

Figure 2.4: Artificial dose volume created from the beam layout in Figure 2.3

2.2 Conversion

2.2.1 Planar Contour to Closed Surface Conversion

The planar contour to closed surface algorithm was published in a paper at the SPIE: Medical Imaging conference in 2015 [1].

Conversion of planar contours to a closed surface representation is a process that has been implemented many times for medical imaging applications [19]. Within this thesis, the focus of the planar contour to closed surface conversion was on the implementation of a functional algorithm for use in RT treatment planning, rather than the development of a novel conversion method .

The triangulation of the contours was implemented based on a method developed by Fuchs et al. that triangulates the contours by minimizing the length of edges that span between the contours [20].

Methods are also utilized from the triangulation algorithm developed by Meyers et al. in order to handle the problem of correspondence and branching [21].

The planar contour to closed surface algorithm is implemented in a manner that is intended to create a triangulation that preserves the original contour points within the surface, while avoiding the removal of old points or the addition of new points to the surface. One exception to this rule is the removal of points during the keyhole contour pre-processing step.

Internal contours are contours whose points are entirely surrounded by another contour on the same slice. Points in contours are often represented by using a clockwise ordering for external contours while internal contours are represented using a counter-clockwise ordering. Some of the structures that have been used to test the algorithm do not follow this rule however, and can have contours with an arbitrary orientation. For this reason all contours are converted to have a clockwise ordering before being triangulated.

Keyhole Removal

The DICOM-RT standard defines keyholes within planar contour representations as narrow channels used to combine separate internal and external contours on a single slice together into a single large contour (see Figure 2.5) [13].

When triangulating planar contours into a closed surface representation, these contours need to be separated in order for the triangulation process to proceed correctly. If the contours are not separated, then the triangulation process will attempt to link the inner and outer contours together in a continuous manner, resulting in an incorrect triangulation.

The keyholes are separated in a pre-processing step, during which, each point is compared to every other point in the contour. If the distance between two points

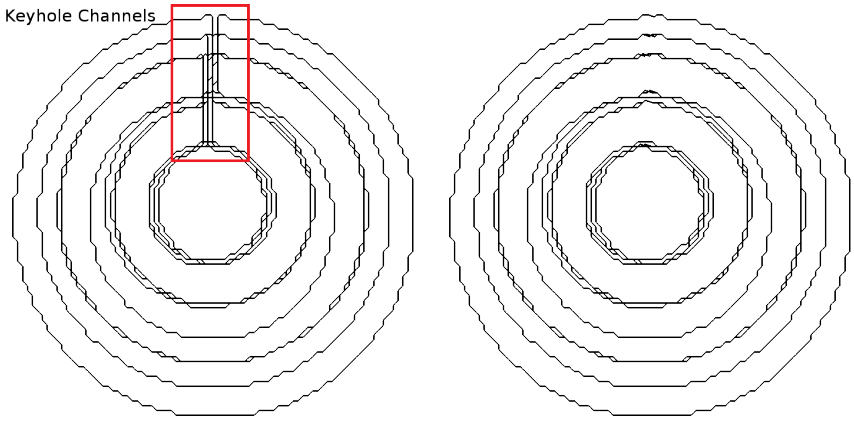

Figure 2.5: Fabricated keyhole contours before (left) and after (right) removal

is below a certain threshold and they are not adjacent to each other in the contour, then they are found to be in conflict with each other. This process continues until all of the points have been analyzed, at which point the algorithm iterates over the list of points and recreates the contours by checking which of the points are in conflict. The first contour is created and inserted to a stack of contours, before adding the points to the contour that are not in conflict. If a conflicting point is encountered the current contour is set aside and a new contour is started and pushed onto the stack. Points that are not in conflict are added to the new contour until a point is found that is in conflict with a previously checked point. The contour is then popped off of the stack and added to a list of completed contours. This process continues until all of the points have been evaluated, at which point the final step is to ensure that all of the contours represent closed loops.

Correspondence and Branching

Creating a triangulation between planar contours requires prior knowledge about which contours should be triangulated together, which is known as correspondence.

Algorithm 1 Keyhole Removal Pseudocode

```
Let α be the minimum distance between two points
Let β be the minimum points of separation
conflicts ← []
for each point i in contour do
    for each point j in contour do
        if | point(i) - point(j) | < α and |i − j| < β  then
            conflict(i) ← j
        end if
    end for
end for

contours ← []
current contour ← []
finished contours ← []
for each point i in contour do
    if conflict(i) > i and conflict(i − 1) < i then
        current contour ← []
        contours.push(current contour)
    else if conflict(i) < i then
        finished contours.insert(contours.pop()
        current contour ← contours.peek()
    else
        current contour.insert(i)
    end if
end for
for c in contours do
    finished contours.insert(c)
end for
```

The correspondence between contours is determined by comparing the axis aligned bounding boxes in the xy-plane of contours that are contained on adjacent z-axis slices [21]. When a contour bounding box is found to overlap with more than one contour on an adjacent slice, then a potential branching region exists.

If a branching region is detected within a contour, then the contours must be divided into separate regions in order to handle the branching geometry triangulation. Before triangulation, contours with multiple correspondence are separated into smaller individual contours according to their closest neighbouring contour, with several duplicated points along the seams to ensure connectivity. Each of the separated regions are then processed as independent contours that are triangulated separately to their corresponding contour, resulting in a branching surface mesh.

Triangulation

In order to triangulate sets of contours together, a greedy PointWalk algorithm was originally used to convert contours to closed surface mesh. This method was often found to create incorrect triangulations, so an alternative was needed that could correctly triangulate contours together.

The new triangulation algorithm works by creating triangles between the contours using a dynamic programming algorithm based on the method developed by Fuchs et al. [20] that minimizes the length of edges between the contours. To begin, the algorithm finds two points, one from each contour, that are closest together to use as a starting location. The dynamic programming table is created starting from these two points, where the table axes are each representative of the points in one of the contours, in counter-clockwise order. The starting location represents the first edge in the triangulation, between the two closest points, from there the triangulation is built by calculating the cumulative length of the edges used to create the triangles. At each cell in the table, a triangle is created using two edges: an edge from the previous cell that has the shortest triangulation length, and an edge connecting the two points

represented by the current cell. When the edge from one of the previous cells is calculated, the value of the current cell is set to be the sum of the edge represented by the current cell and the value of the previous cell. Iteration is completed when the last two points in the contours are fully triangulated, at which point the table is backtracked to find the path representing the triangulation with the shortest possible length of contour spanning edges.

Seal Mesh

Since the triangulation algorithm only creates triangles between the contours, there are still holes in the surface mesh exterior that lie within the contours at the top and bottom of the structure. These contours are identified by iterating through all of the surface mesh triangles and matching them to the contours that they connect, making note of if they lie above or below the contour. If the triangle centre is above the slice plane in the z-axis, then the contour was triangulated from above. If the triangle centre is below the slice plane in the z-axis, then the contour has been triangulated from below. After all of the triangles have been checked, any contour that has not been triangulated from both above and below the contour plane needs to be sealed.

Each contour in the full segmentation represents a CT slice of the structure that has a non-zero thickness, extending $\frac{1}{2}$ slice thickness both above and below the contour. This means that if the inside of the external contours was simply triangulated, it would cause a loss of volume when compared to the actual segmented structure. In order to correctly represent the slice thickness, the external contours are duplicated and extruded from the surface mesh by $\frac{1}{2}$ slice thickness. The newly extruded end-cap contour is then rasterized in the xy-plane and eroded until it takes up approximately

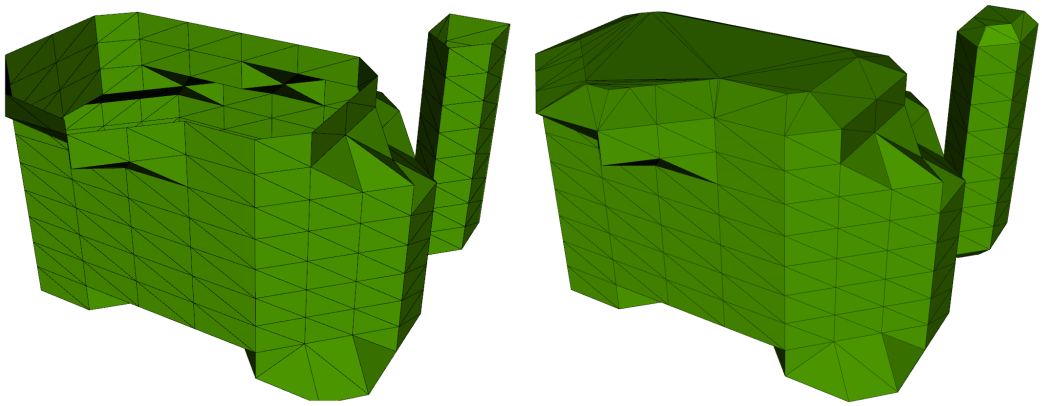

Figure 2.6: Structures before (left) and after end capping (right)

50% of the original contour area. Using this method, contours generated are able to retain the original shape of the contour, even in structures that have irregular geometry. Decimation is used on the resulting contour in order to reduce the number of points to an amount that is consistent with the original contours. The contour is then triangulated to the main surface mesh using the previously outlined triangulation method. The final step to finishing the closed surface is to triangulate the internal area of the end-cap contour using an ear-clipping algorithm (see Figure 2.6).

2.2.2 Closed Surface to Fractional Labelmap Conversion

The closed surface to fractional labelmap algorithm was published in a paper at in the SPIE: Medical Imaging conference in 2017 [4].

Voxel Volume Calculation

In the ideal case, each voxel in a fractional labelmap segmentation would represent the volume of the closed surface that is contained within the voxel as a fraction of the total voxel volume. Unfortunately, there is no fast and simple solution to calculate

the volume of the intersection between each voxel and the surface mesh. Furthermore, the extra information that could be obtained from this method would be made largely irrelevant once the data is compressed into a 1 byte representation.

Oversampled Binary Labelmap

In order to quickly estimate the volume of a surface mesh that is contained within each voxel, the first approach developed was to generate an oversampled binary labelmap at a high resolution, which can be resampled into a fractional labelmap with a regular voxel size. Oversampling was applied to each dimension uniformly, which increased the resolution in each axis by the same factor.

Using 1 byte of memory to represent each voxel allowed a range of 256 values, between 0 and 255, to be utilized. In order to use this extra storage effectively, while still oversampling the labelmap by a uniform amount in each direction, it was necessary to choose an oversampling factor that did not exceed the maximum value of 255 when cubed. This meant that an oversampling factor of 6 (6^3=216) was the maximum amount of uniform oversampling possible, as 7 (7^3=343) would go beyond the available range.

The approach that resulted from this was to create a binary labelmap from the closed surface mesh that was oversampled by a factor of 6 relative to the desired fractional labelmap resolution. The fractional labelmap was then constructed from the oversampled labelmap by summing the number of filled voxels in the oversampled binary labelmap that lie within the voxel bounds at the desired resolution. Using an oversampling factor of 6 resulted in fractional values between 0 (0% occupancy) and 216 (100% occupancy) in each voxel. This method worked well for small structures,

however for large structures such as the body, it was difficult to reliably allocate the oversampled labelmap in memory, due to its size.

The problem with memory allocation was avoided by dividing the image data into smaller extents that were processed independently and combined to generate the final fractional labelmap. This approach was found to be slow since the closed surface voxelization method that was used had a substantial pre-processing overhead. Since the small extent regions could be calculated independently of each other, multithreading was used to voxelize each of the smaller extents in parallel. By multithreading, the execution time of the algorithm was found to decrease substantially, however another approach was later implemented that increased the algorithm's performance without multithreading.

Offset Binary Labelmap

The principles behind the oversampling method can still be applied by considering the oversampled binary labelmap as a grid of 216 points within each voxel. At the original resolution, the goal of the conversion algorithm is to determine the number of these points that lie within the structure (Figure 2.7). Instead of constructing a single high resolution labelmap, it is possible to create 216 different binary labelmaps at the original resolution, by shifting the origin in each labelmap so that the centre of each voxel is at one of the 216 grid points, instead of the original voxel centre. The value of each fractional labelmap voxel is calculated by summing the number of times the corresponding voxels are filled in the 216 binary labelmaps. Since the binary labelmaps are computed independently, the memory requirement of the algorithm

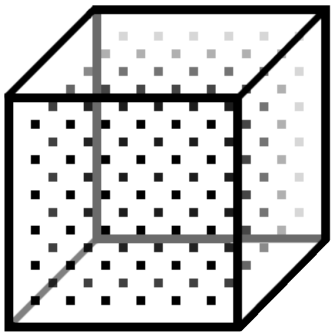

Figure 2.7: Oversampled offset voxel locations. By shifting the origin to one of the offset points, the voxelization algorithm can sample that location within each voxel, instead of at the centre of the voxel

is reduced so that only two labelmaps are allocated at any given time, one binary labelmap, and the output fractional labelmap.

The solid voxelization algorithm computes the binary labelmap from the closed surface on a slice by slice basis, calculating the intersection of the closed surface with the slice plane and rasterizing the intersecting edges. By creating a modified class that is based on the same algorithm, it is possible to cache these slice intersections, so that they do not need to be recalculated in each of the offset labelmap calculations. Between these optimizations, the offset method is currently the fastest approach implemented for calculating fractional labelmaps.

2.2.3 Fractional Labelmap to Closed Surface Conversion

The fractional labelmap to closed surface algorithm was published in a paper at the SPIE: Medical Imaging conference in 2017 [4].

The process of converting fractional labelmaps back into a closed surface representation is completed using a marching cubes algorithm [22]. Using an existing

marching cubes implementation, the iso-surface value is set to represent 50% occupancy within the voxel, which is 108 for a scalar range of 0 to 216. This results in a watertight interpolated surface mesh that can be further smoothed or decimated as the user requires. As a final, optional step, the surface normals of the mesh are calculated to present a smoother visualization.

2.3 Fractional Effects

Most segmentation tools contain methods for modifying binary labelmap representations, and are not compatible with fractional labelmaps. To facilitate modification of fractional labelmap representations without relying on binary labelmap to closed surface to fractional labelmap conversion, existing binary labelmap segmentation methods have been modified to allow fractional labelmaps to be edited using the same algorithms. This includes methods such as Paint, Threshold, Draw, Level-Tracing, Scissor, GrowCut, Slice Interpolation, Island Effects, Smoothing, and Logical Operations.

2.3.1 Paint Effect

The Paint and Erase effects allow users to click on the master image data to add values within a cylindrical or spherical brush to the segment labelmap (see Figure 2.8). The brush is represented visually by a closed surface mesh that is placed at the mouse pointer location. In order to apply the paint effect, the brush position is recorded periodically in world coordinates as the user clicks and drags their cursor around the image. When the left mouse button is released, the closed surface mesh is transformed to each of the points and voxelized into a binary volume. These images are combined

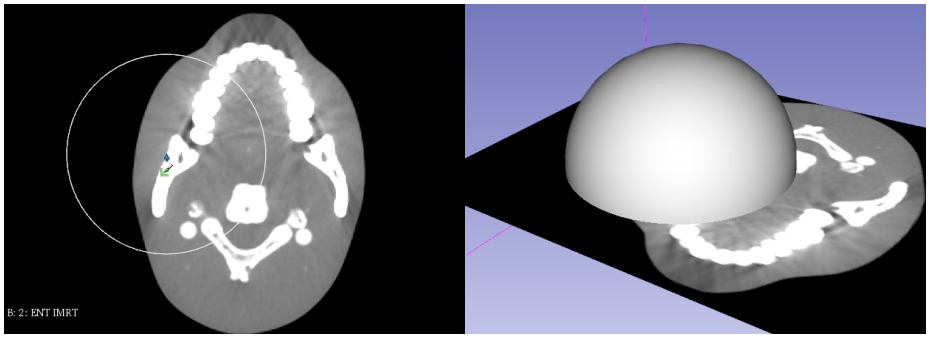

Figure 2.8: Paint effect using a spherical brush segmentation in 2D (left) and 3D (right) views

together into a single image that is resampled to the correct orientation and combined with the segment using a maximum operation (for Paint), or is inverted and then combined with a minimum operation (for Erase).

The first approach to create a fractional labelmap from the existing Paint effect was to voxelize the closed surface mesh brush into an oversampled binary labelmap. The oversampled geometry was calculated so that the image bounds would be identical to the original brush, however the number of voxels in the image was increased by a factor of 6 in each axis. Using the oversampled image, the fractional labelmap was constructed by counting the number of filled voxels inside the bounds of voxels in the original image. This created a fractional labelmap with values between 0 and 216, similar to the closed surface to fractional labelmap conversion process. This approach was found to be functional and accurate, however it took significantly longer than the same binary operation, prompting the need for a faster alternative method using GPU acceleration.

Using GPU accelerated methods, the Paint effect was modified to create a fractional labelmap by executing a custom shader over the brush image data. Within the fragment shader, centre of each voxel is transformed from texture coordinate space

into world coordinates. The brush is defined in the shader as an analytical sphere by computing the distance in world space coordinates from the brush location to the centre of the current voxel. Voxels that lie on the edge of the sphere, are sampled in the fragment shader at 216 points within the current voxel. The fragment shader returns a value that represents the fraction of the 216 points that are within the sphere (between 0.0 and 1.0). For voxels that are not on the edge of the sphere, the fragment shader returns either 0.0 or 1.0, depending on if the voxel was outside or inside the sphere, respectively.

2.3.2 Threshold Effect

Labelmaps are created with the Threshold effect by using a user-specified scalar range representing the minimum and maximum values in the master image that are considered to be inside the segment. For binary segmentation, voxels that are within the scalar range are represented with a value of 1 while voxels outside the scalar range are represented with a value of 0.

The first approach used to create a fractional labelmap by thresholding was to oversample the master image data using linear interpolation by a factor of 6 in each axis. This caused issues however, since the amount of memory required to fully oversample the master image data was greater than could be reliably allocated in memory, resulting in a segmentation fault. To get around this issue, the master image data was divided into smaller extent regions that would have approximately the same dimensions as the original image when oversampled. This solved the crashes related to memory allocation, however the algorithm was still observed to be slow compared to the original binary operation.

Using a GPU accelerated method, the speed of the Threshold method was increased by implementing the algorithm as a GLSL shader. The master image data is passed to the shader as a 3D texture which is used by the fragment shader to sample the values of 216 evenly spaced points within each of the voxels using linear interpolation. The fragment shader output is a value between 0.0 and 1.0 corresponding to the fraction of sampled points that are within the specified threshold.

2.3.3 Draw/Level Tracing Effect

Draw creates a labelmap brush by allowing the user to place a number of points which define a 2D polygon in one of the slice planes. The polygon is then converted into a binary labelmap by rasterizing the polygon into a single binary image slice. Level tracing uses a similar method, however it creates the polygon by tracing a contour around voxels in the master image data with a specified intensity.

Creating a fractional labelmap from the input polygon uses a method that is similar to the closed surface to fractional labelmap conversion, but in two dimensions. The polygon is converted into a high resolution binary labelmap that is oversampled in the slice plane by a factor of 6. It is then resampled into a fractional labelmap by summing the number of 36 binary voxels into each fractional voxel. To account for slice thickness, each of the 36 points adds 6 to the final voxel value instead of 1, which results in a fractional value between 0 and 216.

2.3.4 Scissor Effect

The Scissor segmentation method allows the user to specify the mode (erase/fill, inside/outside), and the brush shape (free-form, circle, rectangle) that is used to edit

the labelmap. Using these shapes, the user creates a closed surface representation in any orientation and view, that is voxelized into a binary labelmap and used to modify the image segment with the specified operation. To adapt this method for fractional labelmaps, the voxelization procedure utilizes the previously outlined closed surface to fractional labelmap conversion algorithm to create a fractional representation of the scissor. The fractional labelmap is then used to modify the selected segment with the specified operator.

2.3.5 Other Effects

Existing effects such as GrowCut and Slice Interpolation could not be easily modified for use with fractional labelmaps. The GrowCut algorithm operates by iteratively expanding user-created regions, or "seeds" that have been placed on the image, based on background intensity [23]. These methods are approximated for fractional labelmaps by resampling the fractional input segments into high resolution binary labelmaps using linear interpolation and thresholding, increasing dimensions of the labelmap by a default oversampling factor of 3. The oversampling factor takes into account factors such as memory and computation speed, but can easily be adjusted based on user needs. Once the binary algorithms are complete, the resulting labelmaps are resampled back into fractional representations at the original resolution.

For binary labelmaps, the Margin Shrink/Grow effects use an erode and dilate kernel to alter structure boundaries, treating the voxels as discrete values. Similar kernels are used to apply erode and dilate operations to the fractional labelmap image, treating the voxels as continuous, rather than discrete, values.

Island Effects allow the user to remove, isolate, and split voxel islands that occur in the binary labelmap. This method is not well suited for fractional labelmaps, so the Island Effects are handled by converting the fractional labelmap into a binary labelmap using thresholding. Voxels that are considered empty have a value of 0, while all other voxels have a value of 1. The Island Effects create binary image masks that are applied to the fractional labelmap by copying or removing voxels within the mask.

Smoothing effects are implemented using a number of methods: Gaussian and Median smoothing do not need any modifications for fractional labelmaps, and can be used with the same methods. Opening uses an image erode followed by a dilation, while closing uses a dilation followed by an erosion. Minimal changes are required to support Gaussian and Median smoothing of fractional labelmaps, since the filters are already able to operate on fractional data. The opening and closing effects use continuous variations of the erode and dilate methods, which typically use filters that expect discrete non-continuous values. Joint smoothing is accomplished by converting the fractional labelmap into a closed surface, running a smoothing filter on the mesh, and then converting the smoothed mesh back into a fractional labelmap.

The following Logical Operations were added by modifying the binary Labelmap Operations: Copy, Invert, Add, Subtract, Intersect, and Clear/Fill. These rules can be consistently applied to both binary and fractional with only minor adjustments. Copy duplicates the values of the selected labelmap into another. The Inversion operation inverts the value in each of the voxels to that the maximum value is set to the minimum, and vice versa. Addition of labelmaps is implemented by calculating

the maximum value of the voxels between the two labelmaps. Subtraction operation is implemented by inverting the subtracting labelmap, and then calculating the minimum value of the voxels between the two labelmaps. Intersection calculates the minimum value of all voxels between two labelmaps. Clear and fill operations set the value of all voxels to the minimum and maximum scalar range values, respectively.

2.4 Analysis

2.4.1 DVH Comparison

In order to determine the effectiveness of a radiotherapy treatment plan, the first step is to quantitatively evaluate the differences between DVHs. This is a topic that was addressed by Ebert et al. in their paper: "Comparison of DVH data from multiple radiotherapy TPS" [24]. The authors developed a method to compare DVH data using a formula that calculates a measure of similarity by comparing the volume and dose of each bin (see Equations 2.1 and 2.2 from [24]).

$$\gamma_i = min\{\Gamma[(d_i, v_i), (d_r, v_r)]\}\forall\{r = 1..P\} \tag{2.1}$$

$$\Gamma[(d_i, v_i), (d_r, v_r)] = \left[\left(\frac{100 \cdot (v_r - v_i)^2}{\Delta V_R \cdot V_{Tot}}\right) + \left(\frac{100 \cdot (d_r - d_i)^2}{\Delta D_R \cdot D_{max}}\right)\right]^{1/2} \tag{2.2}$$

These formulas calculate a gamma value for each of the bins in the DVH, which represents how similar the bins are to a reference DVH. If a bin has a gamma value less than 1, then the points are considered to be in agreement. If the gamma value is greater than 1, this indicates that the points do not have similar dose or volume values. The values ΔV_R and ΔD_R indicate the volume-difference (VD) criterion and

dose-to-agreement (DTA) criterion, while V_{Tot} and D_{max} indicate the total volume and maximum dose in the DVH. By calculating the gamma value for each bin in the DVH, the authors quantify the differences between DVHs by calculating the percentage of bins in which the gamma value was less than zero for specific values of ΔV_R and ΔD_R.

2.4.2 Effects of Voxelization

In order to evaluate the effect of voxelization and voxel size on DVH accuracy, high and low resolution labelmaps are generated from the implicit functions. Both of the labelmaps are used with dose volumes at the same resolution to calculate DVHs for each structure. The DVH comparison module is used to evaluate the accuracy of low resolution DVHs against the high resolution ground truth DVHs. Using VD and DTA criteria values of 1%/1% and 3%/3% , the analysis are computed twice for each structure. The 3% and 1% criteria have previously been used by research applications for the coarse and fine comparison of structure volume and dose [25].

Dice Similarity Coefficient

Computing a metric for the similarity between two binary images can be accomplished by using the Dice similarity coefficient (DSC). When using the DSC to compare binary images, it measures the proportion of voxels that overlap between both labelmaps [26]. The labelmap voxels are iterated through at the high image volume resolution. If the voxel is filled in both images, then the number of true positive voxels is incremented. If the voxel is filled in the low resolution labelmap, but not in the high resolution labelmap, then it is marked as false positive. If the voxel is not filled in the low resolution labelmap, but is in the high resolution labelmap, then it

is marked as false negative. The DSC is calculated from the true positive (TP) and false positive/negative (FP, FN) values using Equation 2.3. While the DSC is able to measure the similarity between structures, it is influenced by the total structure size.

$$DSC = \frac{2TP}{2TP + FP + FN} \tag{2.3}$$

Hausdorff distance

The Hausdorff distance is used to calculate the similarity between two sets of points (A and B) [27]. By comparing the distance from each point in A to the closest point in B, several metrics can be calculated, including the maximum (see Equation 2.4), and average (see Equation 2.5) Hausdorff distances [28]. Unlike the DSC, Hausdorff distance is less affected by total structure size when calculating structures similarity, however it is not as effective at measuring differences between small structures.

$$H_{Maximum}(A, B) = \max_{a \in A} \min_{b \in B} ||a - b|| \tag{2.4}$$

$$H_{Average}(A, B) = \frac{1}{|A|} \sum_{a \in A} \min_{b \in B} ||a - b|| \tag{2.5}$$

2.4.3 Closed Surface to Fractional Labelmap

The conversion process from closed surface to fractional labelmap is tested against binary labelmap representations in different ways. Structure volume is used as the first metric to compare the volume of the fractional and binary representations for a given structure against the ground truth value of the original volume of the closed surface mesh. Volume is calculated for the binary and fractional labelmaps by summing

the fractional value of each of the labelmap voxels, and multiplying by the volume of a single voxel. For binary labelmaps, the fractional voxel value is either 0 or 1, corresponding to the actual value in the labelmap. The fractional voxel value for the fractional labelmap is calculated using Equation 2.6. The binary and fractional labelmaps are compared to the original closed surface mesh by calculating the absolute difference between the volume of the labelmaps and the volume of the closed surface mesh. The differences between binary and fractional volume are also calculated as a percentage of the total structure volume. The ground truth volume is calculated using an existing implementation based on the divergence theorem [29].

$$fractional\ voxel\ value\ = \frac{value - minimum}{maximum - minimum} \tag{2.6}$$

Metrics such as the DSC and Hausdorff distance that are normally used to compare binary labelmaps are not used for the fractional labelmap comparison. These evaluation methods are designed to compare two sets of binary or spatial data against each other, and are normally unable to compare fractional data. The metrics are therefore not used, since they would require significant modifications in order to be used for comparing binary and fractional labelmaps.

Fractional DVH

The accuracy of fractional labelmaps is evaluated for use in DVH calculations by modifying the algorithm for DVH calculation to allow either fractional or binary labelmaps to be used in the calculation process. The fractional voxel value is integrated into DVH calculation by calculating the fractional value behind each dose voxel, which is then multiplied by the voxel size and added to the DVH calculation.

The accuracy of the DVHs calculated using fractional labelmaps is evaluated by calculating DVHs for binary and fractional labelmaps of the same structures, and comparing these results to DVHs calculated using Eclipse™ (Eclipse™ radiation therapy treatment planning system, Varian Medical Systems, Inc., Palo Alto, CA, USA) and CERR (Computational Environment for Radiotherapy Research), a widely used Matlab-based open-source RT research toolkit [30]. These DVHs are compared using the previously mentioned metrics developed by Ebert et al. [24].

2.4.4 Fractional Labelmap to Closed Surface

The conversion from fractional labelmap back to closed surface representation is evaluated by creating binary and fractional labelmaps from RANDO® ENT dataset structures, and converting the labelmaps back into closed surface representations, without smoothing or decimation. Closed surfaces created from both fractional and binary labelmaps are compared by calculating the maximum and average directed Hausdorff distance from the reconstructed surfaces to the original closed surface mesh. The same method is also used for the reverse comparison, from the original surface to the reconstructed surfaces.

Changes between the volumes of the surface meshes are also evaluated by calculating the total mesh volume for each structure. The difference in volume between the surface meshes from binary/fractional labelmaps and the original closed surfaces is calculated using both cubic centimetres (cc), and as a percentage of the original structure volume.

Chapter 3

Results

3.1 Software

The conversion algorithms and analysis were implemented in 3D Slicer using the Segmentations and Segment Editor Module, along with the SlicerRT toolkit.

3.1.1 3D Slicer

3D Slicer is a free open-source medical imaging software tool that is used in institutions across the world [16] [1]. The most recent major version of Slicer has been downloaded over 299,000 times across Linux, Mac and Windows platforms. It was developed using the Visualization Toolkit (VTK)[2], which contains powerful image processing and 3D mesh algorithms. The base 3D Slicer program includes useful features, such as image visualization, segmentation, registration, volume rendering, and DICOM import and export functionality. The 3D Slicer platform has an easily extensible structure, supporting the creation of modules and extensions that add additional functionality to the software using both Python and C++.

[1] slicer.org
[2] vtk.org

Segmentations

One of the recent additions to the list of core 3D Slicer modules is the Segmentations module [31]. The Segmentations module enables the consolidation of different segmentation representations into a single data structure for easy storage and conversion. In each segmentation, the original structure data is stored as the master representation. Using the original master representation, structures can be converted into other representation types using various conversion algorithms. If the original representation data is modified, then the converted representations must be reconverted.

The Segmentations module defines a number of representation types, including: binary labelmap, closed surface, planar contours, ribbon model, and fractional labelmap. Representations can be created from the master representation type by performing a series of consecutive conversions linking a path from the original representation to the desired representation type. It is easy for developers to implement and register additional conversion rules between any two of the defined representation types.

3D Slicer also includes a Segment Editor module that allows users to easily edit binary labelmap segment representations using a variety of common segmentation methods. In addition to the default effects included in the base version of the Segment Editor, it is also possible for developers to create and utilize their own effects for editing segmentations.

3.1.2 SlicerRT

SlicerRT is an open-source RT toolkit that was developed for 3D Slicer in the Laboratory for Percutaneous Surgery (Perk Lab) at Queen's University [15][3]. It provides a powerful and extensible platform for RT research and supports the ability to import DICOM-RT files into Slicer from commercial RT TPS. It offers functionality such as external beam planning, DVH calculation, DVH comparison, dose comparison, isodose generation, and segment comparison. SlicerRT was designed through the input of researchers at the Ontario Consortium for Adaptive Intervention in Radiation Oncology, and has been adopted for use in many clinical and research applications around the world.

3.2 Effects of Voxelization

The classes that were developed to test the effects of voxelization were implemented in 3D Slicer [16] using Python and C++. These modules utilized the External Beam Planning module in SlicerRT to arrange the beam locations for dose volume calculation [15]. Analysis of the labelmaps were completed by using the Segment Comparison and Segment Statistics modules for measuring the structure volume, Hausdorff distance, and DSC. In addition, the Dose Volume Histogram and DVH Comparison modules were used to calculate the DVHs, and evaluate their differences, respectively.

The volume of structures represented using the labelmaps with a large voxel size were found to have a difference of up to 9.3% compared to the volume of small structures represented using the small voxel size labelmap. Some of the largest differences

[3] slicerrt.org

were in the left and right lens structures, which had differences of 9.3% and 5.0%, while the left and right nerve structures had differences of 6.3% and 8.5%.

Maximum Hausdorff distances for the structures were in the range of 1.8mm to 3.0mm (see Table 3.1). Average Hausdorff distance remained relatively consistent at around 0.5mm, except for three outliers: body, brain stem and gross tumour volume (GTV). The brain stem had an average Hausdorff distance of 0.9mm, the body had an average of 0.9mm, and the GTV structure had an average Hausdorff distance of 1.5mm. Based on the results for the current structures, the maximum Hausdorff distance was found to be a poor metric for evaluating the effects of voxelization on closed surface mesh, while the average distance showed more promising results.

DSC was able to calculate the contribution of false positive and negative regions in the results, which was affected by the size of structural features. For structures with large features, such the body, brain, brain stem, planning target volume (PTV), clinical target volume (CTV), and GTV, the DSC was found to be over 0.90 when comparing the similarity between small and large voxel size binary labelmaps. In structures with small features, such as optic nerves, optic lenses, and optic chiasm, the voxel size was larger relative to total structure volume, resulting in a DSC between 0.79 and 0.90 (see Table 3.1).

By plotting DVHs calculated using the labelmaps with small and large voxel sizes, visible differences were observed between the two graphs (see Figure 3.1). Structures with large features such as the body ,brain, CTV, and GTV were found to have only small or negligible differences between the DVHs. For structures with small features, like the optic lenses and nerves, the DVHs were more likely to be affected by the change in voxel size. Certain structures such as the brain stem were also found to be

Table 3.1: Comparison of volume, Hausdorff distance, and DSC between the same implicit structures using labelmaps with small and large voxel sizes

Structure	Small Voxel Size (mm)	Small Voxel Structure Volume (cc)	Large Voxel Structure Volume (cc)	Volume Difference (%)	Max Hausdorff Distance (mm)	Average Hausdorff Distance (mm)	Dice Similarity Coefficient
Body	0.30	2990.68	3016.11	0.9	3.0	0.9	0.99
Brain	0.30	1380.47	1379.13	0.1	2.1	0.5	0.99
Brain Stem	0.15	54.45	54.63	0.3	1.9	0.9	0.96
PTV	0.15	179.60	179.59	0.0	2.1	0.5	0.97
CTV	0.15	113.10	113.11	0.0	2.0	0.5	0.97
GTV	0.15	24.01	24.53	2.2	2.1	1.5	0.95
Optic Nerve - Left	0.05	4.97	4.66	6.3	2.0	0.5	0.88
Optic Nerve - Right	0.05	5.34	4.89	8.5	1.9	0.6	0.88
Orbit - Left	0.05	9.99	10.02	0.3	2.1	0.5	0.93
Orbit - Right	0.05	10.29	10.27	0.3	2.1	0.5	0.93
Optical Lens - Left	0.05	0.33	0.34	5.0	2.0	0.5	0.79
Optical Lens - Right	0.05	0.33	0.30	9.3	1.8	0.5	0.79
Optic Chiasm	0.05	2.87	2.84	1.0	2.0	0.5	0.90

affected by the change in voxel size. This resulted in visible effects, such as a staircase-like effect and structures with consistent discrepancies between DVHs calculated from small and large voxel labelmaps (see Figure 3.2).

DVHs from the small and large voxel labelmaps were compared using the DVH Comparison module in 3D Slicer, using 1% and 3% DTA and volume agreement criteria (see Table 3.2). When compared to the small voxel labelmaps, the DVH agreement for structures with small features was found to have a wide range of 28.48% to 99.19% agreement when using 1%/1% DTA and VD criteria, and a range of 53.66% to 99.19% agreement when using 3%/3% criteria. Large featured structures were more resilient to the change in voxel size, resulting in 100.0% agreement for many structures, with only the GTV decreasing to 97.7% agreement. The brain stem however, was prone to differences, even with its larger volume, decreasing to 59.23% with 1%/1% DTA and VD criteria, and back to 100.0% when using 3%/3% agreement criteria.

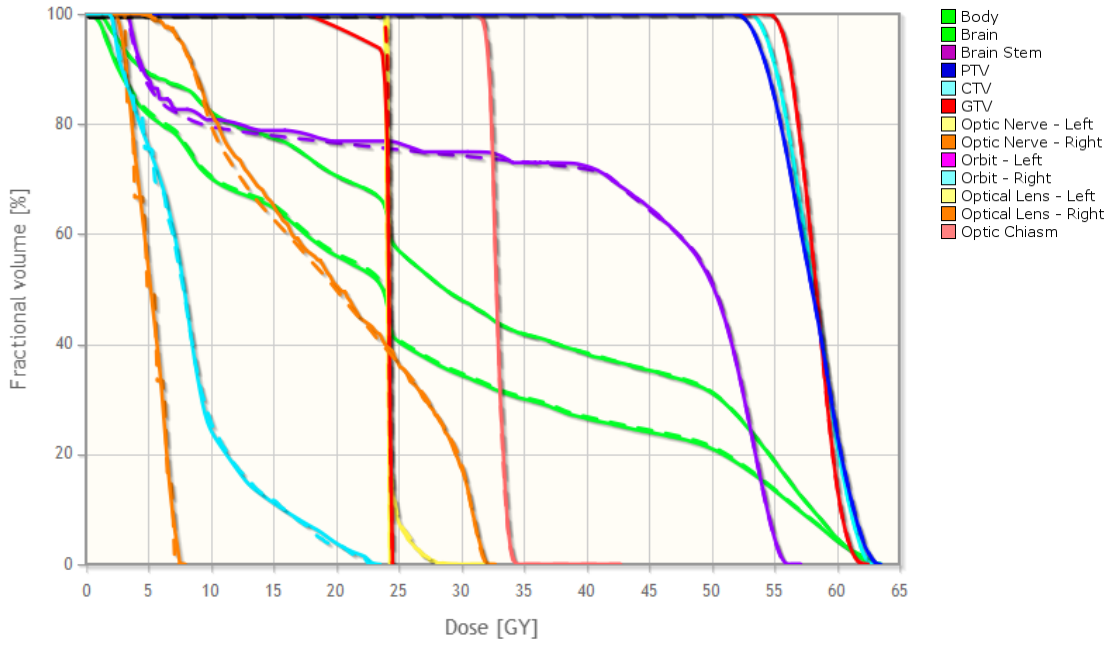

Figure 3.1: Comparison of DVHs calculated for implicit structures using labelmaps with small (solid) and large (dashed) voxel sizes

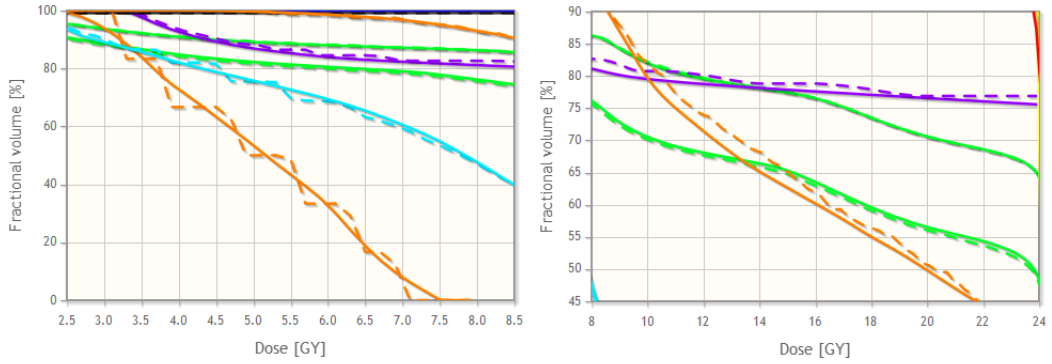

Figure 3.2: Enlarged regions of Figure 3.1 showing differences between implicit structure DVHs calculated from small voxel labelmaps (solid) against DVHs calculated from large voxel labelmaps (dashed). This shows a staircase effect (left) and consistent differences betwen DVHs (right)

Table 3.2: Comparison between implicit structure DVHs calculated from small voxel labelmaps against DVHs calculated from large voxel labelmaps

Structure	**Reference Voxel Size** (mm)	**Structure Volume** (cc)	**Surface Area** (cm^2)	**Agreement Acceptance %** ($1\%/1\%DTA/VD$)	**Agreement Acceptance %** ($3\%/3\%DTA/VD$)
Body	0.30	2990.68	812.90	100.0	100.0
Brain	0.30	1380.47	614.89	100.0	100.0
Brain Stem	0.15	54.45	84.03	59.2	100.0
PTV	0.15	179.60	153.78	100.0	100.0
CTV	0.15	113.10	112.99	100.0	100.0
GTV	0.15	24.01	52.16	97.8	100.0
Optic Nerve - Left	0.05	4.97	18.62	90.2	94.5
Optic Nerve - Right	0.05	5.34	19.94	28.5	59.4
Orbit - Left	0.05	9.99	23.76	75.0	79.8
Orbit - Right	0.05	10.29	24.59	50.4	76.5
Optical Lens - Left	0.05	0.33	2.39	99.2	99.2
Optical Lens - Right	0.05	0.33	2.39	51.2	53.7
Optic Chiasm	0.05	2.87	10.51	92.6	93.0

3.3 Planar Contour to Closed Surface Conversion

Planar contour to closed surface conversion was converted from Python to C++ and integrated into SlicerRT [15] as a conversion rule for the Segmentations module.

The algorithm was tested on hundreds of datasets containing structures segmented from CT images of both real individuals and phantoms. It was found to produce fully closed surfaces which did not contain any errant geometry for most structures that were tested, even in those containing branching regions (see Figure 3.3).

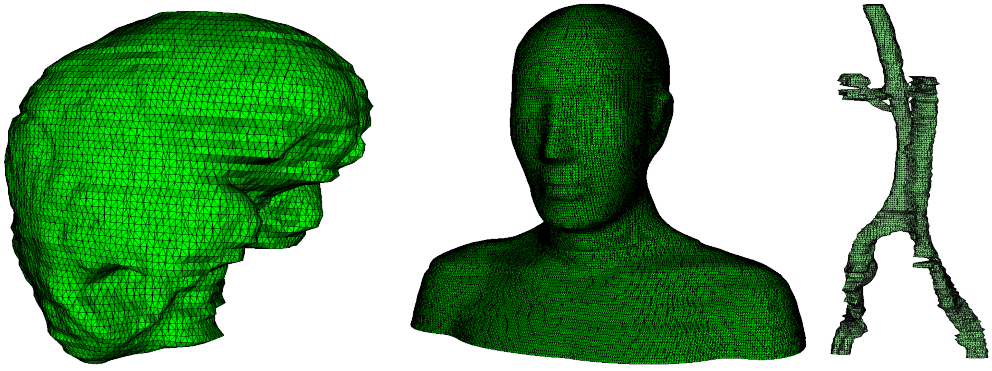

Figure 3.3: Closed surface mesh created from sets of planar contours representing brain (left), head and neck (centre), and vessels (right)

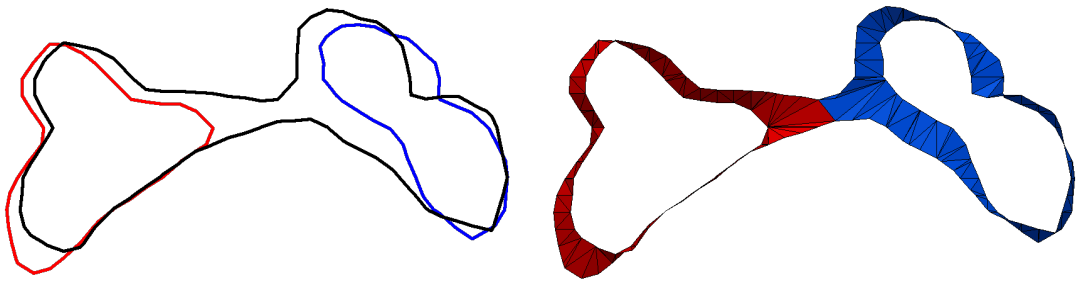

Figure 3.4: Demonstration of the triangulation of a branching contour, showing both the contours (left) and the triangulated surface (right) separated into individual branches (shown in red and blue)

3.4 Closed Surface to Fractional Labelmap Conversion

The closed surface to fractional labelmap conversion algorithm was integrated into the Segmentations module in the 3D Slicer core.

3.4.1 Accuracy

The effectiveness of fractional labelmaps at representing structures was tested by calculating and comparing the volume of structures in closed surface, binary labelmap, and fractional labelmap representations. Considering the closed surface to be the ground truth volume, the difference in total structure volume between the ground truth was compared to binary and fractional labelmaps at the same resolution (see Table 3.3). The volume of structures represented using fractional labelmaps was found to be closer to the ground truth than the binary labelmap structure volume. As a percent of the total ground truth structure volume, the improvements were found to be inversely relative to the size of structural features, since the improvements were small in structures with large features and large in structures with small features. The right optic nerve, which has a volume of 1.89cc, saw the largest

Table 3.3: Improvement in volume representation between closed surface to binary and fractional labelmap conversion algorithms for RANDO® ENT phantom structures

Structure	Closed Surface Volume (*cc*)	Binary Volume (*cc*)	Fractional Volume (*cc*)	Binary Volume Difference (%)	Fractional Volume Difference (%)	Improvement Between Binary and Fractional Volume (%)
Body	8031.29	8053.87	8031.24	0.3	0.0	0.3
Brain	1113.30	1114.65	1113.12	0.1	0.0	0.1
Brain Stem	30.99	31.42	30.98	1.4	0.0	1.4
PTV	126.18	127.03	126.17	0.7	0.0	0.7
CTV	69.52	70.11	69.52	0.8	0.0	0.8
GTV	8.00	8.46	7.99	5.9	0.1	5.8
Optic Nerve - Left	1.55	1.76	1.54	13.7	0.0	13.7
Optic Nerve - Right	1.89	2.26	1.89	19.3	0.3	19.1
Orbit - Left	8.30	8.29	8.29	2.4	0.0	2.4
Orbit - Right	8.56	8.56	8.56	2.4	0.0	2.4
Optical Lens - Left	0.12	0.14	0.12	17.6	0.2	17.4
Optical Lens - Right	0.11	0.13	0.11	10.6	0.1	10.5
Optic Chiasm	1.01	1.15	1.00	14.2	0.1	14.1

improvement for fractional labelmap of 19.1% of the original volume of the closed surface mesh, when compared to the binary labelmap volume (2.26cc).

In order to compare the accuracy of fractional DVH calculations, DVHs were calculated for the binary labelmap using the original method, while the fractional labelmap DVHs were calculated using the modified algorithm. These calculations were performed on the set of structures from the RANDO® ENT phantom alongside a clinical dose volume (see Figure 3.5).

The DVHs were evaluated using the DVH Comparison module in SlicerRT against DVH from Eclipse™ and CERR [30]. When comparing the DVHs, it was found that the fractional labelmaps had an agreement acceptance percentage that was equal to or greater than that of the binary labelmaps for most structures (see Table 3.4). In particular, the brain stem structure saw the most improvement, increasing by 13.5% and 2.5% relative to the binary labelmap for Eclipse™ and CERR, respectively. There were a few structures in which the agreement acceptance percent decreased

Figure 3.5: Visualization of fractional (solid) and binary (dashed) DVHs calculated in SlicerRT for RANDO® ENT phantom structures

in the fractional labelmap relative to the binary labelmap, however in each of these cases, the total difference was less than 1%.

3.4.2 Performance

The system specifications of the computer that was used to test the performance of closed surface to fractional labelmap conversion used an Intel® Core™ i7-4790K CPU @ 4.00GHz, Nvidia GeForce® GTX 750 Ti (2048 MB), and 16GB of RAM.

The closed surface to fractional labelmap algorithm was found to take an average of 7.35s (n=5) to create fractional representations for all structures in the RANDO® phantom, while the closed surface to binary labelmap algorithm took an average of 0.52s (n=5) for the same structures (see Table 3.5).

Table 3.4: Comparison between DVHs from Eclipse™ and CERR against DVHs calculated with binary and fractional labelmaps in SlicerRT from RANDO® ENT phantom structures

Structure	Binary and Eclipse (%)	Fractional and Eclipse (%)	Eclipse Improvement (Fractional - Binary) (%)	Binary and CERR (%)	Fractional and CERR (%)	CERR Improvement (Fractional - Binary) (%)
Body	100.0	100.0	0.0	100.0	100.0	0.0
Brain	100.0	100.0	0.0	100.0	100.0	0.0
Brain Stem	41.6	99.1	57.5	55.1	96.5	41.4
PTV	98.7	99.7	1.0	97.5	97.1	-0.3
CTV	98.4	99.1	0.6	97.5	97.5	0.0
GTV	98.4	98.4	0.0	98.1	98.1	0.0
Optic Nerve - Left	86.7	92.7	6.0	86.9	86.9	5.1
Optic Nerve - Right	93.3	93.3	0.0	93.6	93.6	0.0
Orbit - Left	95.9	95.2	-0.6	94.6	94.6	0.3
Orbit - Right	96.2	96.8	0.6	93.3	93.3	0.6
Optical Lens - Left	99.7	99.7	0.0	99.4	99.4	0.0
Optical Lens - Right	98.7	98.4	-0.3	98.7	98.7	0.0
Optic Chiasm	79.4	80.6	1.3	81.2	87.2	-0.6

Table 3.5: Average execution time of closed surface to fractional labelmap and closed surface to binary labelmap algorithms (n=5) on RANDO® ENT phantom structures

Structure	Closed Surface Volume (cc)	Number of Triangles	Average Fractional Conversion Time (s)	Average Binary Conversion Time (s)
Body	8031.29	92652	4.32	0.41
Brain	1113.30	21815	0.87	0.05
Brain Stem	30.99	2580	0.24	0.01
PTV	126.18	4674	0.29	0.01
CTV	69.52	3258	0.25	0.01
GTV	8.00	959	0.19	0.00
Optic Nerve - Left	1.89	532	0.17	0.00
Optic Nerve - Right	1.55	503	0.17	0.00
Orbit - Left	8.30	957	0.19	0.00
Orbit - Right	8.56	980	0.19	0.00
Optical Lens - Left	0.12	112	0.16	0.00
Optical Lens - Right	0.11	114	0.16	0.00
Optic Chiasm	1.01	280	0.17	0.00

When using fractional labelmaps for DVH calculation, the execution time took an average of 1.88s (n=5) for all structures in the RANDO® phantom, while binary labelmaps were found to take an average of 1.67s (n=5) for the same structure, representing an average increase in computation time of 0.22s.

3.4.3 Visualization

Variable opacity is a method of visualization that allows the user to estimate individual voxel occupancy at a glance. Opacity is adjusted so that voxels with 0% occupancy have 0% opacity, and voxels with 100% occupancy have 100% of the current maximum opacity level for the segment (see Figure 3.6). This method of visualization is most useful for applications such as probabilistic segmentations, where voxel values represent the probability that a given segment will exist within each voxel. These types of segmentations are usually calculated as the output of atlas-based segmentation algorithms [32].

For structures in which the segmentation represents a more accurate measure of occupancy in each voxel, the structures that are represented have defined boundaries that separates the interior and exterior. This type of representation requires alternative visualization method that creates a delineated representation around each structure that attempts to accurately reconstruct the original geometry. To visualize a hard edge around the fractional labelmap, each slice is resampled at a high resolution using linear interpolation. The slice is then thresholded to generate a hard edge around values in the segmentation that are greater than the specified occupancy threshold. By default, the value used in threshold visualization is set to represent 50% occupancy in the labelmap. Using this method for rendering fractional labelmaps in the 2D view, structures are visualized using clearly defined borders that delineate structure boundaries. This is a better method for preserving the defined edges of the original structure, which is important for real structures where the occupancy values are used to represent concrete boundaries, however for probabilistic representations, the variable opacity method may be preferred.

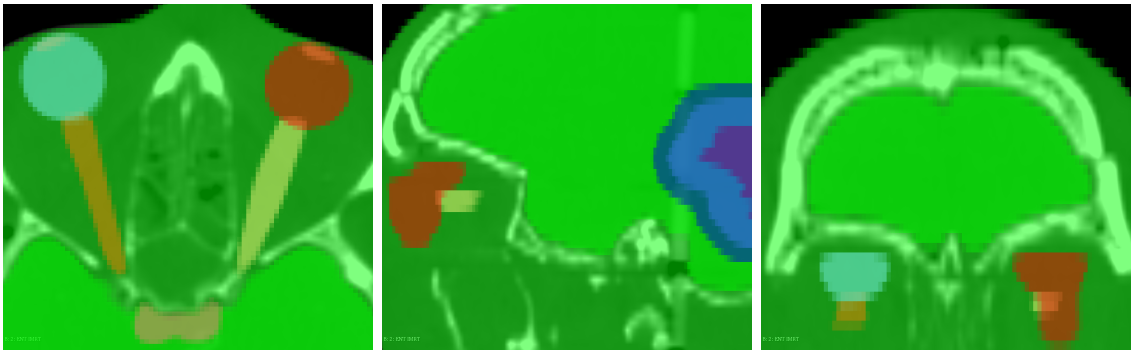

Figure 3.6: Smooth-edged fractional labelmap using the opacity visualization method with nearest neighbour interpolation

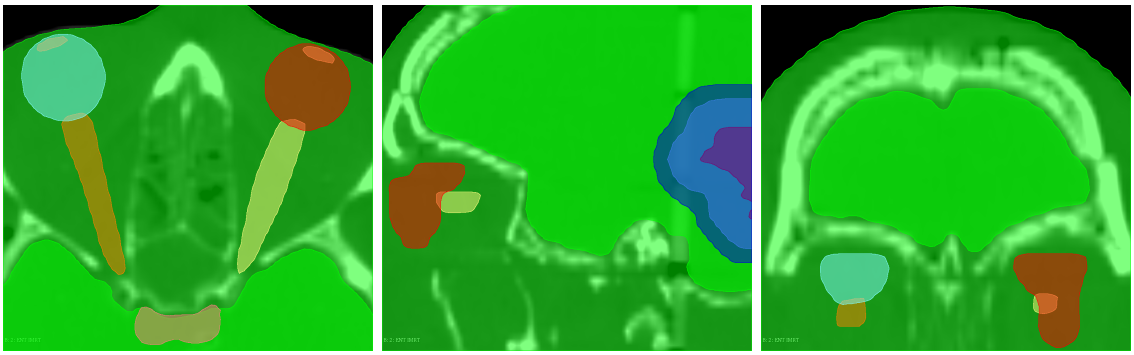

Figure 3.7: Hard-edged fractional labelmap using the thresholding visualization method with linear interpolation

Comparing the visualization of fractional and binary labelmaps at the same resolution to the original closed surface demonstrates that the fractional labelmap representation is better at representing the original structure than the binary labelmap representation (Figure 3.8). Structures in the binary labelmap visualization appear jagged and have a step-like effect, while the fractional representations is interpolated between adjacent voxels, resulting in an appearance that is more similar to the original closed surface.

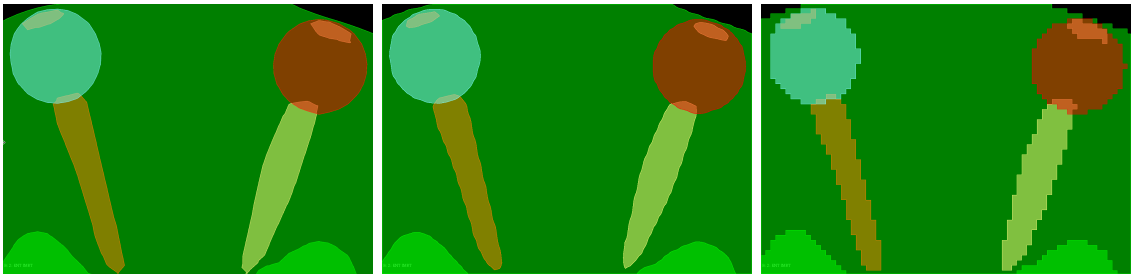

Figure 3.8: Comparison between the 2D representations of closed surfaces (left) to fractional (centre), and binary labelmaps (right) at the same resolution

3.5 Fractional Labelmap to Closed Surface Conversion

The fractional labelmap to closed surface conversion algorithm was integrated into the Segmentations module in the 3D Slicer core.

3.5.1 Accuracy

The binary and fractional labelmap to closed surface algorithms were tested by converting original closed surface meshes to binary and fractional labelmaps, and then reconstructing the closed surfaces from these labelmap volumes. Both the binary and fractional labelmaps were created from the original closed surface meshes at the same image resolution. Surface meshes created from the fractional labelmaps were found to be visually closer when compared to the original closed surface representations than the surfaces generated from the binary labelmaps (see Figure 3.9).

The average and minimum Hausdorff distances from the reconstructed surfaces to the original closed surface meshes were found to be lower when using surface meshes reconstructed from fractional labelmaps when compared to the reconstructed surface meshes triangulated from binary labelmaps (see Table 3.6). Conversely, inverse comparisons of the the average and maximum Hausdorff distances from the original

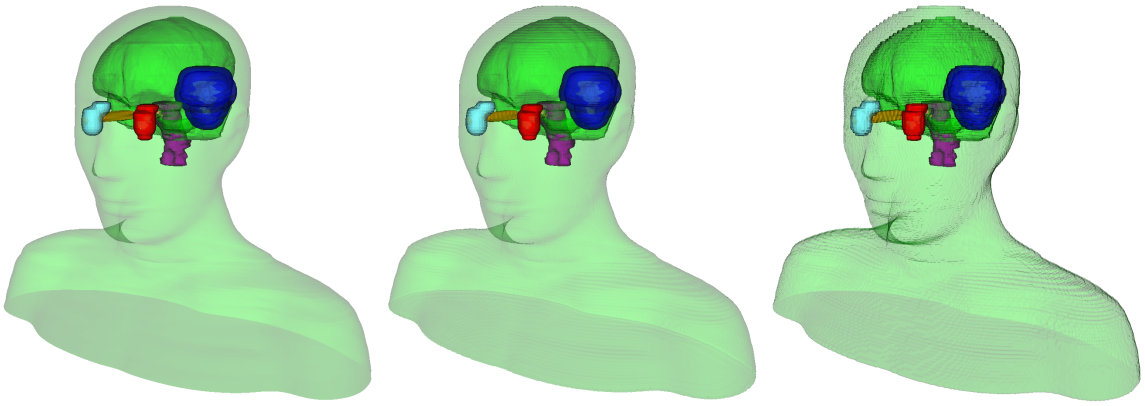

Figure 3.9: Comparison between the original closed surfaces (left) and the closed surfaces created from fractional (centre) and binary labelmaps (right)

Table 3.6: Directed Hausdorff distance from points in surface meshes created from binary and fractional labelmaps to points in the original closed surfaces for RANDO® ENT phantom structures

	Binary To Original Surface Distance		**Fractional To Original Surface Distance**		**Improvement**	
Structure	**Max** (*mm*)	**Avg** (*mm*)	**Max** (*mm*)	**Avg** (*mm*)	**Max** (*mm*)	**Avg** (*mm*)
Body	1.3	0.9	0.4	0.2	0.9	0.7
Brain	1.2	0.4	0.5	0.1	0.7	0.3
Brain Stem	1.2	0.3	0.5	0.1	0.7	0.2
PTV	1.2	0.3	0.3	0.1	0.9	0.2
CTV	1.2	0.3	0.5	0.1	0.7	0.2
GTV	1.2	0.4	0.7	0.2	0.6	0.2
Optic Nerve - Left	1.2	0.3	0.5	0.1	0.7	0.2
Optic Nerve - Right	1.3	0.4	0.7	0.2	0.6	0.3
Orbit - Left	1.2	0.3	0.4	0.1	0.8	0.2
Orbit - Right	1.2	0.3	0.3	0.1	0.9	0.2
Optical Lens - Left	0.9	0.3	0.4	0.1	0.5	0.1
Optical Lens - Right	0.8	0.2	0.6	0.1	0.3	0.1
Optic Chiasm	1.2	0.3	0.5	0.1	0.6	0.2

closed surface meshes to the reconstructed surfaces were found to be smaller in the surface meshes created from binary labelmaps than in meshes derived from fractional labelmaps (see Table 3.7).

Without smoothing, the volumes of surfaces reconstructed from fractional labelmaps were found to be closer to the original closed surface volumes in all except

Table 3.7: Directed Hausdorff distance from points in the original closed surfaces to points in surface meshes created from binary and fractional labelmaps for RANDO® ENT phantom structures

	Original To Binary Surface Distance		Original To Fractional Surface Distance		Improvement	
Structure	**Max** (mm)	**Avg** (mm)	**Max** (mm)	**Avg** (mm)	**Max** (mm)	**Avg** (mm)
Body	0.5	0.1	1.9	0.2	-1.3	-0.1
Brain	0.8	0.1	2.2	0.2	-1.3	-0.0
Brain Stem	0.6	0.1	1.0	0.2	-0.4	-0.1
PTV	0.5	0.2	0.6	0.1	-0.1	0.0
CTV	0.6	0.2	0.8	0.2	-0.2	-0.0
GTV	0.7	0.2	2.6	0.6	-1.8	-0.4
Optic Nerve - Left	2.2	0.2	3.3	0.4	-1.1	-0.2
Optic Nerve - Right	6.1	0.5	6.4	0.8	-0.3	-0.3
Orbit - Left	0.5	0.1	0.7	0.2	-0.2	-0.1
Orbit - Right	0.6	0.1	0.7	0.1	-0.1	-0.0
Optical Lens - Left	1.4	0.2	1.2	0.3	0.3	-0.2
Optical Lens -Right	0.9	0.2	1.1	0.3	-0.2	-0.2
Optic Chiasm	0.6	0.1	1.3	0.3	-0.7	-0.2

Table 3.8: Volume comparison between original closed surface representations and closed surfaces created from binary and fractional labelmaps for RANDO® ENT phantom structures

Structure	**Closed Surface Volume** (cc)	**Fractional Surface Volume** (cc)	**Binary Surface Volume** (cc)	**Fractional Difference** (%)	**Binary Difference** (%)	**Improvement** (%)
Body	8031.29	8033.19	8052.93	0.0	0.3	0.3
Brain	1113.30	1112.72	1114.26	0.1	0.1	0.0
Brain Stem	30.99	30.80	31.29	0.6	1.0	0.4
PTV	126.18	126.02	126.88	0.1	0.6	0.4
CTV	69.52	69.37	69.97	0.2	0.7	0.4
GTV	8.00	7.76	8.34	3.0	4.3	1.3
Optic Nerve - Left	1.55	1.45	1.70	6.2	9.7	3.5
Optic Nerve - Right	1.89	1.79	2.19	5.7	15.8	10.1
Orbit - Left	8.30	8.20	8.43	1.1	1.7	0.6
Orbit - Right	8.56	8.48	8.70	1.0	1.7	0.7
Optical Lens - Left	0.12	0.09	0.12	23.1	3.5	-19.6
Optical Lens - Right	0.11	0.09	0.11	22.4	3.6	-18.9
Optic Chiasm	1.01	0.94	1.11	6.5	10.3	3.8

the two smallest structures: the Lens - Left (-19.6%) and Lens - Right (-18.9%) (see Table 3.8).

Table 3.9: Average execution time comparison for binary labelmap to closed surface conversion and fractional labelmap to closed surface conversion (n=5) for RANDO® ENT phantom structures

Structure	Original Closed Surface Volume (cc)	Binary Execution Time With Smoothing (s)	Binary Execution Time Without Smoothing (s)	Fractional Execution Time (s)
Body	8031.29	1.16	0.38	0.42
Brain	1113.30	0.23	0.07	0.08
Brain Stem	30.99	0.02	0.01	0.01
PTV	126.18	0.04	0.01	0.01
CTV	69.52	0.03	0.01	0.01
GTV	8.00	0.01	0.00	0.00
Optic Nerve - Left	1.55	0.00	0.00	0.00
Optic Nerve - Right	1.89	0.00	0.00	0.00
Orbit - Left	8.30	0.01	0.00	0.00
Orbit - Right	8.56	0.01	0.00	0.00
Optical Lens - Left	0.12	0.00	0.00	0.00
Optical Lens - Right	0.11	0.00	0.00	0.00
Optic Chiasm	1.01	0.00	0.00	0.00

3.5.2 Performance

Binary labelmap to closed surface and fractional labelmap to closed surface conversions were both implemented using the same marching cubes algorithm [22], resulting in similar execution times. The conversion of binary labelmaps to closed surface representations often utilizes a smoothing post-processing step to create structures that are visually appealing. When comparing the binary labelmap to closed surface conversion of meshes using smoothing against fractional labelmap to closed surface conversion without smoothing, the fractional conversion method was found to be 0.98s faster on average (see Table 3.9).

3.6 Fractional Effects

All of the binary effects used in the Segment Editor were modified to allow fractional labelmaps to be edited using the same methods. Currently, these modified effects are being integrated into 3D Slicer, and will be available for all users in the near future.

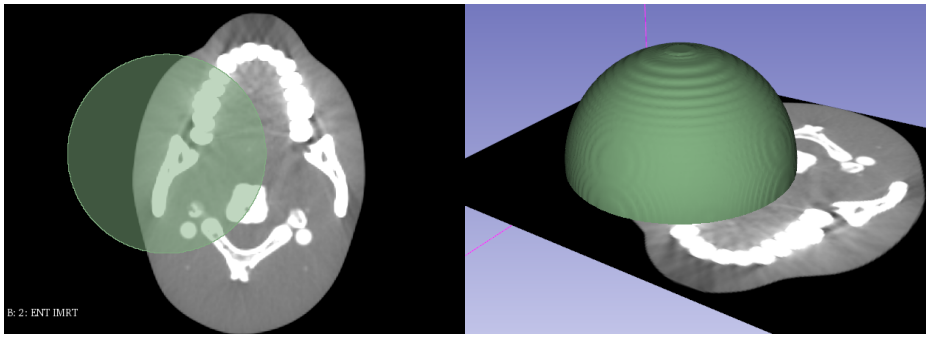

Figure 3.10: Brush results using the sphere brush shown in Figure 2.8. Fractional labelmap (left) and closed surface created from the fractional labelmap (right)

The CommonGL classes developed by Steve Pieper [4] are designed to allow VTK image data to be easily passed into GLSL shaders as 3D textures. The GLSL shader is rendered on each slice of the image data, before being read back from the GPU and stored in a VTK image data representation. These classes were used within the fractional segmentation methods to implement fast GPU accelerated effects.

The Paint and Erase brush effects were able to create smooth fractional labelmaps at the user-specified radius (see Figure 3.10). Using the GPU accelerated methods, fractional Paint and Erase were still found to be noticeably slower than the same effects using binary labelmaps.

Using the modified Threshold effect, fractional labelmaps were created from the master image data that accurately followed the structures contours in the background image. Linear interpolation and resampling was found to create fractional labelmaps that appeared to accurately follow the contours of the background interpolated image (see Figure3.11). The final threshold operation implemented using GLSL was found to take ∼0.91s to threshold a 512x512x139 image volume.

[4] github.com/pieper/Slicer/tree/add-commongl

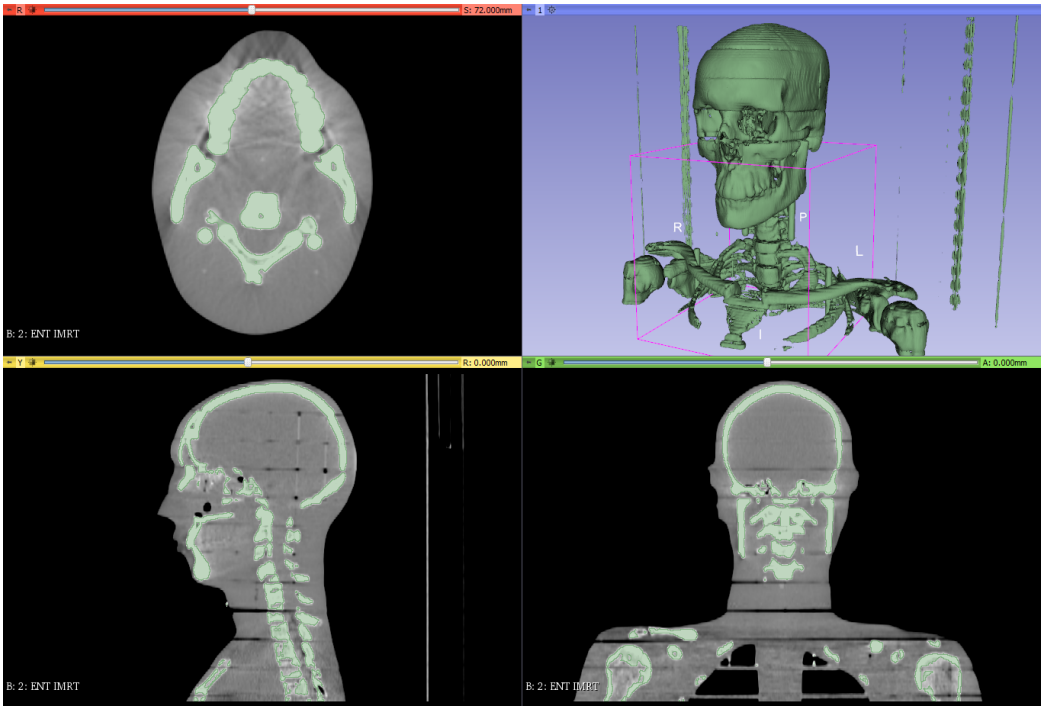

Figure 3.11: Fractional Threshold from CT of a phantom using scalars between 150 and 3071 Hounsfield units

The Level Tracing and Draw effects were able to create fractional labelmaps for the polygons defined by the user (see Figure 3.12). Due to the single slice nature of the segmentation methods, Draw and Level-Tracing were found to have faster execution times when compared to other fractional effects, and did not take noticably longer than the original binary effects.

The Scissors effect was found to create satisfactory fractional labelmaps from the closed surfaces defined by the user (see Figure 3.13). Execution speed for the Scissor effect was found to take approximately 6s to create a labelmap for a shape that would be completed in less than 1s by same the binary effect.

Many of the remaining effects that utilized the same or similar algorithms to the binary labelmap effects, such as Smoothing (except Joint Smoothing), Shrink/Dilate,

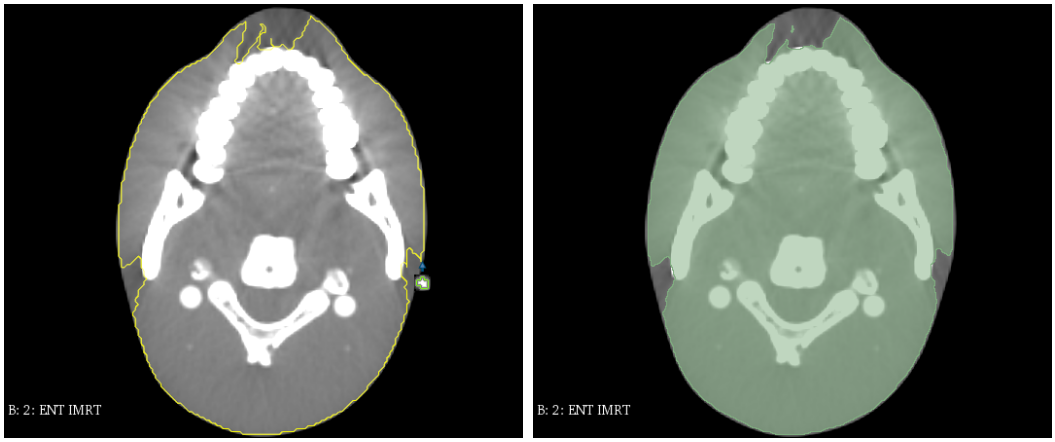

Figure 3.12: Fractional Level-Tracing effect before (left) and after (right)

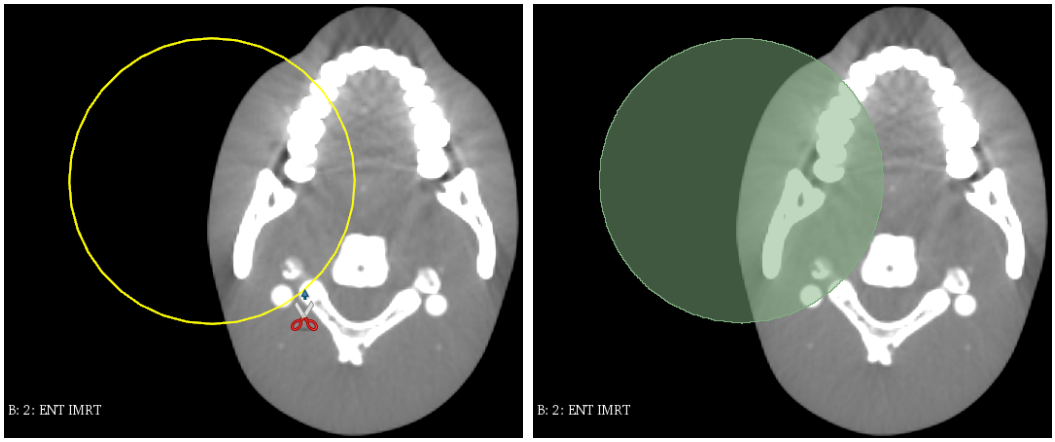

Figure 3.13: Fractional Scissors effect before (left) and after (right)

Islands, and Logical Operators, were found to create good fractional labelmap representations in the same amount of time and at the same resolution as the original binary effects. For the remaining effects, such as GrowCut, Slice Interpolation, and Joint Smoothing, the results were found to be acceptable, although they had an execution time greater than ∼16s for segmentations that could be completed by the same binary labelmap effect in less than ∼1s (see Figure 3.14).

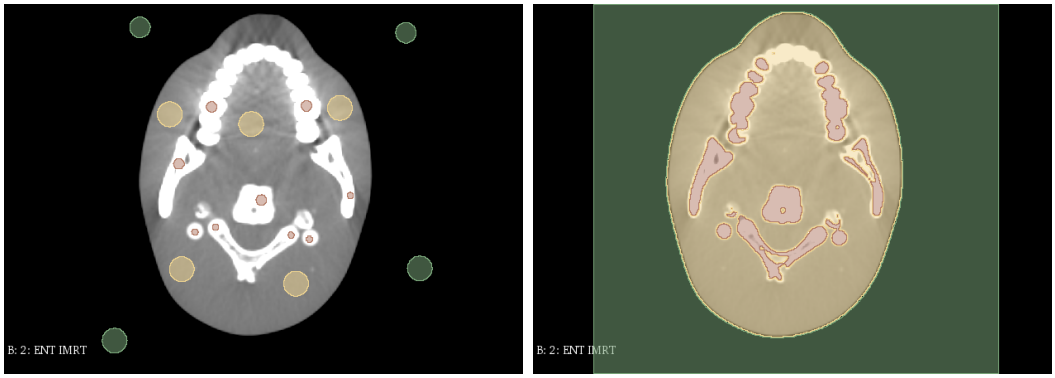

Figure 3.14: Fractional GrowCut effect seeds (left) and results (right)

3.7 Discussion

3.7.1 Effects of Voxelization

Structures with small features, such as the optical lenses and the nerves were found to have the largest differences in total structure volume between the large and small voxel size labelmaps, while structures with larger features had minimal volume differences.

The DSC was found to reflect large changes in total structure volume as a result of voxelization, indicating potential sensitivity to DVH errors in structures with small features. For organs at risk, it is important that the resulting DVHs are as accurate as possible, since sensitive structures in areas exposed to high intensity radiation could be damaged if a sub-optimal treatment plan is used. The size of voxels in the labelmap and dose volumes affected the accuracy of calculations for structures with small features since voxel size is directly responsible for the number of voxels used to represent the structures.

While the maximum Hausdorff distance did not reflect any DVH calculation problems, the average Hausdorff distance was able to detect issues with the brain stem and GTV structures that were not identified by other metrics. The brain stem and GTV

structures were created using angled cylinder and cone shapes that contained surfaces that were not aligned with the labelmap axis. As the structures were voxelized, the surface orientation relative to the labelmap axis resulted in a poorer voxelization. While the GTV was located in a region with a homogeneous dose intensity, the brain stem spanned an area with a high dose gradient, which caused the DVH to have a acceptance percent as low as 59.2% when using 3%/3% VD and DTA criterion. The body structure also had a high average Hausdorff distance, however this was found to be a result of voxelization issues along the top and bottom of the structure, in addition to the larger voxel size (0.3mm X 0.3mm X 0.3mm) that was used for the small voxel labelmap.

Since DVHs are the primary metric used to direct the optimization of RT treatment plans, it could be expected that the errors caused by voxelization would affect the plan optimization. Even though the individual differences caused by the voxelization artifacts shown in Figure 3.2 are quantitatively small, they may still be able to influence the plan optimization direction. The exact impact of these artifacts on the treatment planning process has yet to be determined and is a topic that still needs to be evaluated.

It is difficult to conduct comparisons between DVHs from different TPSs, since differences in steps such as contour triangulation, end-capping, oversampling, voxelization of border voxels, and more, can all affect the results of TPSs. Without a ground truth baseline, it is not trivial to determine which of these results are more correct. A solution to this problem could be found in the methods developed by Nelms et al. for verifying DVH calculations [8]. By comparing calculated DVHs to DVHs

derived from analytical shapes and dose gradients, it would be possible to definitively measure the accuracy of various TPSs.

3.7.2 Planar Contour to Closed Surface Conversion

The planar contour to closed surface conversion was found to be successful in the majority of observed cases. There are exceptions to the algorithm that can cause errant or missing triangles between vertices, due to the previously mentioned issues, in addition to several with unknown causes. These outliers have generally been handled on a case-by-case basis to help resolve the issues as they arise, however no single cause or solution has yet been found.

The contour branching method worked well for most simple cases that were tested, such as the one in Figure 3.3, however branching issues were found with structures that exhibited rapid change and internal contours (see Figure 3.16). In the case of rapidly changing contour shapes between slices, the triangles all tend to converge on a single point in the smaller contour, since it represents the closest point and the smallest possible edge between the contours. For structures that contain internal contours, the branching algorithm can connect internal and external contours together in manner that results in an incorrect triangulation (see Figure 3.15). If the user encounters an issue with triangulation, it is still possible to voxelize the structure by converting the structure to a Ribbon model representation, which can be converted to a binary labelmap.

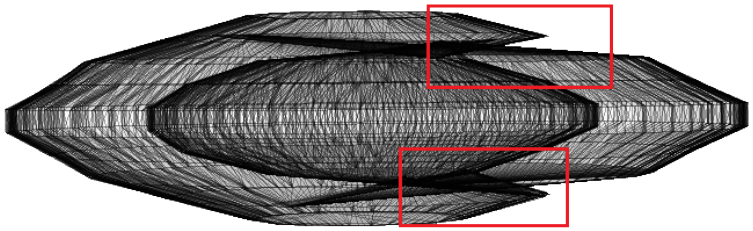

Figure 3.15: Issues with triangulation caused by internal contours that are closer to the contours on the next layer than the external contours. Constructed from the keyhole example in Figure 2.5

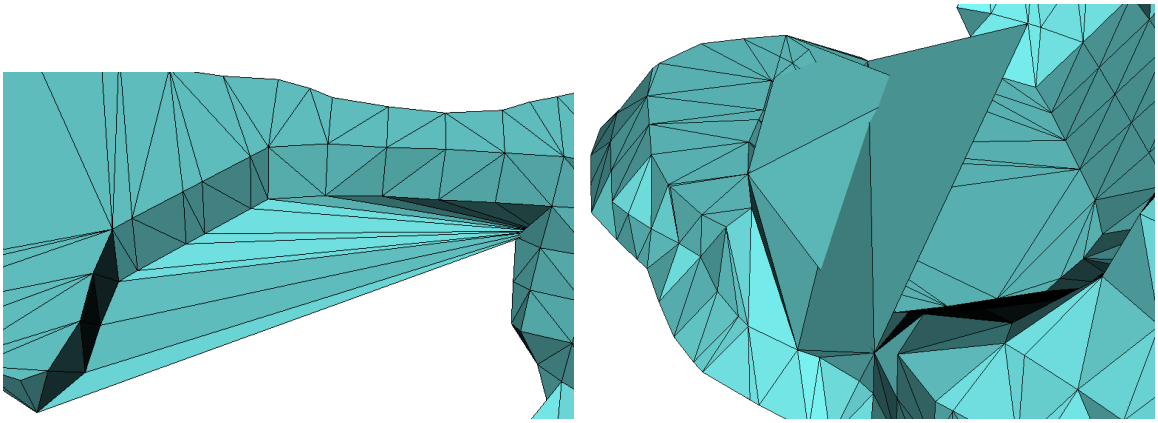

Figure 3.16: Issues with rapid change (left), causing the convergence of many triangles at the same point in a smaller contour. Problems with internal contour triangulation and branching (right)

3.7.3 Closed Surface to Fractional Labelmap Conversion

The conversion of closed surfaces to fractional labelmaps was originally implemented in a manner that was inefficient, resulting in a number of optimizations that were required to increase the speed of the algorithm to a usable level. Although the conversion takes approximately 14 times longer to execute when compared to binary labelmaps, it has improved dramatically from its first iteration. These improvements are a result of implementing methods that were better able to manage the large

amount of memory required by the algorithm, however there is still potential for optimization in the conversion process.

Although the execution time of DVH calculations using fractional labelmaps was found to be greater than the same calculations when using binary labelmaps, the difference is unnoticeable to the user. This assumes that the steps to convert planar contours into labelmap representations are not factored into the calculation cost. Taking into account the overhead of conversion, fractional labelmap representations offer a trade-off between longer execution times and more accurate calculations. In the cases where the DVHs from fractional labelmaps had a lower accuracy than the DVHs from binary labelmaps, the errors could be attributed to differences from the Eclipse™ calculations, specifically in sensitive steps such as contour triangulation.

3.7.4 Fractional Labelmap to Closed Surface Conversion

With fractional labelmaps, the interpolation that is introduced by the marching cubes algorithm allows the structure to be visually appealing, without the need for smoothing, although the option is still available for users that would like a smoother surface. The fractional labelmap to closed surface conversion is able to create smooth surface representations quickly. This is useful for manual segmentation since it allows the user to be quickly shown 3D feedback of the results, without requiring an extra smoothing step.

The fractional labelmap to closed surface conversion is not able to accurately reconstruct the original segmentation geometry for all structures. At the edges of structures, regions can exist in which multiple layers of voxels have less than 50% occupancy. As a result, these regions are often ignored in the triangulation, which

results in a potential loss of volume, as can be seen in the left and right lens structures (see Table 3.8). This may also be the cause of the results in Table 3.7, which showed that the points in the original closed surfaces were closer to the surfaces reconstructed from binary labelmaps, than the surfaces reconstructed from the fractional labelmaps.

3.7.5 Fractional Effects

The fractional effects were able to produce good representations when compared to binary labelmaps produced with the same methods. This may be attributed to the extra information stored in the fractional labelmap, but could also be a result of the methods used to generate the brush. Binary brushes are generated using a closed surface mesh that is created using a limited number of polygons. Fractional brush volumes are created from an analytical representation, which eliminates artifacts that are caused by the closed surface polygons.

Even with the use of GPU accelerated algorithms, fractional effects were found to take longer than the original binary effects. For methods such as Draw, Level-Tracing, Logical Operations and Smoothing, these differences are barely noticeable, however the execution speeds for effects such as Paint, Threshold, Scissor, Slice Interpolation and GrowCut are more impeding. This is to be expected, since the work done by many of the algorithms is approximately equivalent to the creation of 216 binary labelmaps. The implementation of an algorithm to convert closed surfaces to fractional labelmaps on GPU may help to improve the execution of effects such as the Scissors algorithm. Effects such as Slice interpolation and GrowCut might also benefit from being implemented on GPU. The Paint/Erase effects were able to run at

an acceptable speed, but further optimization is still needed to increase the efficiency of painting with a large number of points.

3.7.6 Future Work

In cases where the planar contour to closed surface conversion fails, or if complex structures are detected, it would be useful to notify the user and offer an alternative conversion pathway to create a binary labelmap representation. If a complex structure is detected, the structure could be routed through a different set of simpler conversion algorithms, which should reduce the wait time.

Currently, the best method of comparison to evaluate the similarity and accuracy of fractional labelmaps, is to compare the total structure volume, however it may also be useful to develop additional metrics for fractional labelmap comparison. This could be accomplished by modifying existing metrics such as DSC and Hausdorff distance for use with fractional labelmaps. It would be useful to implement methods that detect differences and estimate the amount of information lost during the conversion process, without increasing computation time significantly. Using these methods, the user could be notified if a large discrepancy exists between the volume of the original and converted representation, which could identify structures in which the accuracy of TPS algorithms might be affected.

The closed surface to fractional labelmap conversion could also benefit from GPU accelerated methods. Eisemann and Dcoret et al. developed a method for the solid voxelization of surface mesh using a single-pass rendering method in OpenGL [33]. A similar method was also developed by Shwarz and Seidel that implements a solid voxelization algorithm in CUDA [34]. CUDA is not currently an option for use in 3D

Slicer, however it may be beneficial to implement a platform agnostic version of the algorithm. Either of these methods could potentially be adapted to create a fractional labelmap by sampling a greater number of points in each voxel.

The original use case for this algorithm was conceived in order to address the issue of sub-voxel representations for structures in an RT TPS context. This led to the implementation of the original versions of the fractional labelmap conversion algorithm within SlicerRT. It became evident that the application of fractional labelmaps could extend beyond RT applications, and so the algorithms were integrated into the 3D Slicer core. Since then, certain groups have expressed interest in the application of fractional labelmap representations in 3D Slicer for other applications, such as prostate segmentation for deep learning research, and atlas based segmentation of MRI.

Chapter 4

Conclusions

The goal of this thesis was to implement methods for the creation and modification of fractional labelmaps, with the goal of increasing the accuracy of segmentation for structures in radiotherapy (RT) treatment planning systems (TPSs). Fractional labelmaps were evaluated against binary labelmaps by comparing the differences in volume for labelmaps generated from closed surface representations. Accuracy was also tested by comparing Dose Volume Histograms (DVHs) calculated from binary and fractional labelmaps using the SlicerRT radiation therapy research toolkit against ground truth DVHs from commercial TPSs. The volume of fractional labelmap representations was found to be much closer to the original closed surface volume than binary labelmaps, especially in structures with small features. DVHs that were calculated using fractional labelmaps were closer to the ground truth DVHs than ones calculated using binary labelmaps. From these results, it was evident that fractional labelmaps are an accurate method of representation that can be used for many of the same applications as binary labelmaps in RT TPS with equivalent or better accuracy.

One of the related goals for this thesis was to find a metric that could quickly gauge the amount of information lost in the contour conversion process. Several metrics such

as maximum and average Hausdorff distance, Dice Similarity Coefficient (DSC), and structure volume were tested to evaluate differences between high and low resolution binary labelmaps for the same structures. While the maximum Hausdorff distance was not able to detect differences caused by voxelization for simple structures, the average Hausdorff distance was found to increase in structures that are susceptible to DVH calculation issues. DSC was found to reflect the amount of information lost through voxelization, relative to the total structure size. Measuring the difference in volume between structures was found to be a fast and efficient method for estimating the amount of information lost from the voxelization process.

The planar contours to closed surface algorithm that was implemented in SlicerRT was able to produce qualitatively good surface meshes for most of the structures tested. There were still issues triangulating features such as internal contours and rapid changes between slices, however these structures were uncommon and the voxelization of these meshes were often still acceptable.

Converting closed surface to fractional labelmaps was completed by creating 216 binary labelmaps from the surface mesh and shifting the origin of the labelmap on each iteration. The resulting labelmaps were summed together to create a fractional labelmap with 217 different potential values. The conversion of closed surfaces to fractional labelmaps was found to be an accurate process that was able to represent structure volume more accurately than binary labelmaps created from the same closed surfaces. Fractional labelmaps were also found to be better than binary labelmap representations for use in DVH calculation. This showed that fractional labelmaps are an accurate method of segmentation representation that can be used for many of the

same algorithms as binary labelmaps, while allowing more information to potentially be utilized.

The process of converting fractional labelmaps back into closed surface representations was completed by using the marching cubes algorithm to reconstruct a surface at the 50% occupancy isosurface. When comparing closed surfaces created from the same structures using binary and fractional labelmaps, the surfaces created from fractional labelmaps were found to be closer to the original surface. The surface meshes created from fractional labelmaps were closer in appearance to the original surfaces than the same surface meshes created from binary labelmaps.

In order to facilitate the segmentation of fractional labelmaps, existing methods were modified to allow users to segment fractional labelmaps without having to rely on intermediate conversion pathways. These methods created qualitatively good representations using fractional labelmaps.

The conversion from planar contours to binary labelmaps was found to be a process that resulted in a loss of information due to the discrete nature of the binary labelmap representation. To reduce the amount of information lost during the conversion process, a fractional labelmap representation was implemented that was able to preserve more relevant structural information for use in medical imaging algorithms. Fractional labelmaps were found to be a promising alternative to binary labelmaps which are capable of improving the accuracy of anatomical structure representation, increasing the quality of patient care for many potential use cases.

Bibliography

[1] K. Sunderland, B. Woo, C. Pinter, and G. Fichtinger, "Reconstruction of surfaces from planar contours through contour interpolation," in *SPIE Medical Imaging*, pp. 94151R–94151R, International Society for Optics and Photonics, 2015.

[2] K. Sunderland, C. Pinter, A. Lasso, and G. Fichtinger, "Effects of voxelization on dose volume histogram accuracy," in *SPIE Medical Imaging*, pp. 97862O–97862O, International Society for Optics and Photonics, 2016.

[3] K. Sunderland, C. Pinter, A. Lasso, and G. Fichtinger, "Analysis of dose volume histogram deviations using different voxelization parameters," in *ImNO*, 2016.

[4] K. Sunderland, C. Pinter, A. Lasso, and G. Fichtinger, "Fractional labelmaps for computing accurate dose volume histograms," in *SPIE Medical Imaging*, pp. 101352Y–101352Y, International Society for Optics and Photonics, 2017.

[5] C. C. S. A. Committee, "Canadian Cancer Statistics 2017," *Canadian Cancer Society*, 2017.

[6] R. Drzymala, R. Mohan, L. Brewster, J. Chu, M. Goitein, W. Harms, and M. Urie, "Dose-volume histograms," *International Journal of Radiation Oncology* Biology* Physics*, vol. 21, no. 1, pp. 71–78, 1991.

[7] B. Fraass, K. Doppke, M. Hunt, G. Kutcher, G. Starkschall, R. Stern, and J. Van Dyke, "American Association of Physicists in Medicine Radiation Therapy Committee Task Group 53: quality assurance for clinical radiotherapy treatment planning," *Medical physics*, vol. 25, no. 10, pp. 1773–1829, 1998.

[8] B. Nelms, C. Stambaugh, D. Hunt, B. Tonner, G. Zhang, and V. Feygelman, "Methods, software and datasets to verify DVH calculations against analytical values: Twenty years late (r)," *Medical physics*, vol. 42, no. 8, pp. 4435–4448, 2015.

[9] T. Ackerly, J. Andrews, D. Ball, M. Guerrieri, B. Healy, and I. Williams, "Discrepancies in volume calculations between different radiotherapy treatment planning systems," *Australasian Physics & Engineering Sciences in Medicine*, vol. 26, no. 2, pp. 90–92, 2003.

[10] J.-F. Corbett, J. Jezioranski, J. Crook, and I. Yeung, "The effect of voxel size on the accuracy of dose-volume histograms of prostate 125i seed implants," *Medical physics*, vol. 29, no. 6, pp. 1003–1006, 2002.

[11] A. Noe and J. C. Gee, "Partial volume segmentation of cerebral MRI scans with mixture model clustering," in *Biennial International Conference on Information Processing in Medical Imaging*, pp. 423–430, Springer, 2001.

[12] S. K. Warfield, C.-F. Westin, C. R. Guttmann, M. Albert, F. A. Jolesz, and R. Kikinis, "Fractional segmentation of white matter," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 62–71, Springer, 1999.

[13] National Electrical Manufacturers Association, 1300 North 17^{th} Street, Rosslyn, VA, USA, *NEMA PS3 / ISO 12052, Digital Imaging and Communications in Medicine (DICOM) Standard*.

[14] N. Li, M. Zarepisheh, A. Uribe-Sanchez, K. Moore, Z. Tian, X. Zhen, Y. J. Graves, Q. Gautier, L. Mell, L. Zhou, *et al.*, "Automatic treatment plan re-optimization for adaptive radiotherapy guided with the initial plan dvhs," *Physics in medicine and biology*, vol. 58, no. 24, p. 8725, 2013.

[15] 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, 2012.

[16] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.-C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, *et al.*, "3D Slicer as an image computing platform for the Quantitative Imaging Network," *Magnetic resonance imaging*, vol. 30, no. 9, pp. 1323–1341, 2012.

[17] H. Zaidi and B. M. Tsui, "Review of computational anthropomorphic anatomical and physiological models," *Proceedings of the IEEE*, vol. 97, no. 12, pp. 1938–1953, 2009.

[18] J. Peter, M. P. Tornai, and R. J. Jaszczak, "Analytical versus voxelized phantom representation for Monte Carlo simulation in radiological imaging," *IEEE transactions on medical imaging*, vol. 19, no. 5, pp. 556–564, 2000.

[19] D. Huijsmans, W. Lamers, J. Los, and J. Strackee, "Toward computerized morphometric facilities: A review of 58 software packages for computer-aided three-dimensional reconstruction, quantification, and picture generation from parallel serial sections," *The Anatomical Record*, vol. 216, no. 4, pp. 449–470, 1986.

[20] H. Fuchs, Z. M. Kedem, and S. P. Uselton, "Optimal surface reconstruction from planar contours," *Communications of the ACM*, vol. 20, no. 10, pp. 693–702, 1977.

[21] D. Meyers, S. Skinner, and K. Sloan, "Surfaces from contours," *ACM Transactions On Graphics (TOG)*, vol. 11, no. 3, pp. 228–258, 1992.

[22] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *ACM siggraph computer graphics*, vol. 21, pp. 163–169, ACM, 1987.

[23] V. Vezhnevets and V. Konouchine, "GrowCut: Interactive multi-label ND image segmentation by cellular automata," in *proc. of Graphicon*, vol. 1, pp. 150–156, 2005.

[24] M. Ebert, A. Haworth, R. Kearvell, B. Hooton, B. Hug, N. Spry, S. Bydder, and D. Joseph, "Comparison of DVH data from multiple radiotherapy treatment planning systems," *Physics in medicine and biology*, vol. 55, no. 11, p. N337, 2010.

[25] H. Zhen, B. E. Nelms, and W. A. Tomé, "Moving from gamma passing rates to patient DVH-based QA metrics in pretreatment dose QA," *Medical physics*, vol. 38, no. 10, pp. 5477–5489, 2011.

[26] K. H. Zou, S. K. Warfield, A. Bharatha, C. M. Tempany, M. R. Kaus, S. J. Haker, W. M. Wells, F. A. Jolesz, and R. Kikinis, "Statistical validation of image segmentation quality based on a spatial overlap index 1: Scientific reports," *Academic radiology*, vol. 11, no. 2, pp. 178–189, 2004.

[27] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 9, pp. 850–863, 1993.

[28] M.-P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," in *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1, pp. 566–568, IEEE, 1994.

[29] A. M. Alyassin, J. L. Lancaster, J. H. Downs, and P. T. Fox, "Evaluation of new algorithms for the interactive measurement of surface area and volume," *Medical physics*, vol. 21, no. 6, pp. 741–752, 1994.

[30] J. O. Deasy, A. I. Blanco, and V. H. Clark, "CERR: a computational environment for radiotherapy research," *Medical physics*, vol. 30, no. 5, pp. 979–985, 2003.

[31] C. Pinter, A. Lasso, and G. Fichtinger, "Dynamic management of segmented structures in 3D Slicer," in *ImNO*, 2016.

[32] M. Cabezas, A. Oliver, X. Lladó, J. Freixenet, and M. B. Cuadra, "A review of atlas-based segmentation for magnetic resonance brain images," *Computer methods and programs in biomedicine*, vol. 104, no. 3, pp. e158–e177, 2011.

[33] E. Eisemann and X. Décoret, "Single-pass GPU solid voxelization for real-time applications," in *Proceedings of graphics interface 2008*, pp. 73–80, Canadian Information Processing Society, 2008.

[34] M. Schwarz and H.-P. Seidel, "Fast parallel surface and solid voxelization on GPUs," in *ACM Transactions on Graphics (TOG)*, vol. 29, p. 179, ACM, 2010.