2022

A Method for Predicting Dose Changes for HN Treatment Using Surface Imaging

Sarah E. Holler
Virginia Commonwealth University

A Method for Predicting Dose Changes for HN Treatment Using Surface Imaging

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University

by
Sarah Elizabeth Holler
Bachelor of Science, Bucknell University, May 2016

Director: Dr. Christopher Guy,
Assistant Professor, Department of Radiation Oncology

Virginia Commonwealth University

Richmond, Virginia

December, 2022

Acknowledgements

During my time at Virginia Commonwealth University working on this thesis, I had the privilege of working under the direction of two advisors and receiving the benefits of their combined perspectives and clinical expertise. I am thankful to Dr. Laura Padilla, my advisor when I began this project, for her clinical knowledge and helping me grow as a physicist and researcher. More than that, I'm thankful for her believing in me enough to embark on this project and inspiring me throughout my time at VCU. I am also grateful to Dr. Chris Guy for taking over from Dr. Padilla seamlessly, providing another clinical and research perspective, and guiding me through the second half of the work. I would also like to thank Dr. William Song for his encouragement and his compassionate approach to directing the graduate program. I also am appreciative of Dr. Lulin Yuan and Dr. Shiyu Song for providing invaluable insight into the direction of this work. The contributions of Dr. Bridget Quinn in finding patients and contouring structures ensured the success of this work. Additionally, I appreciate Dr. Frank Corwin for joining this project so recently and providing a unique perspective. I am also very thankful for the financial support provided by the VCU Health Department of Radiation Oncology.

Finally, I would like to thank my fellow students and colleagues throughout the years who offered guidance, feedback, and emotional support: Katie Goracke, Siqiu Wang, Matthew Riblett, Nicky Mahon, Dr. Ford Sleeman, the professors, and perhaps most importantly, my husband, Benjamin Lewis. I couldn't have gotten here without each and every one of you.

Table of Contents

List of Tables

List of Figures

Abbreviations

3DCRT	Three-Dimensional Conformal Radiotherapy
Adam	Adaptive Moment Estimation
ART	Adaptive Radiation Therapy
CBCT	Cone Beam Computed Tomography
CNN	Convolutional Neural Network
CT	Computed Tomography
CTV	Clinical Target Volume
dCT	Deformed Computed Tomography
DIR	Deformable Image Registration
DNN	Deep Neural Network
DOF	Degrees of Freedom
DSC	Dice Similarity Coefficient
DVH	Dose-Volume Histogram
FBP	Filtered Backprojection Algorithm
FDK	Feldkamp Algorithm
FLE	Fiducial Localization Error
FOV	Field Of View
GPU	Graphics Processing Unit
GTV	Gross Target Volume
HD	Hausdorff Distance
HDS	Single-sided Hausdorff Distance
HN	Head and Neck
HNC	Head and Neck Cancer
HPV	Human Papillomavirus
HU	Hounsfield Units
IGRT	Image Guided Radiotherapy

IMRT	Intensity Modulated Radiation Therapy
JI	Jaccard Index
kV	Kilovoltage
MAE	Mean Absolute Error
MI	Mutual Information
MLC	Multi-Leaf Collimator
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NMI	Normalized Mutual Information
NPC	Nasopharynx Cancer
OAR	Organs At Risk
OBI	Onboard Imager
OPC	Oropharynx Cancer
OPX	Oropharynx
pCT	Planning Computed Tomography
PET	Positron Emission Tomography
PTV	Planning Target Volume
QOL	Quality of Life
ReLU	Rectified Linear Unit
RM	Robbins and Monro Gradient Descent
RMSE	Root Mean Squared Error
ROI	Region of Interest
RT	Radiation Therapy
SAD	Sum of Absolute Differences
SI	Surface Imaging
SIB	Simultaneous Integrated Boost
SRS	Stereotactic Radiosurgery

SSD Sum of Square Differences

TNM Tumor-node-metastasis

TPS Treatment Planning System

TPU Tensor Processing Unit

TRE Target Registration Error

VMAT Volumetric Modulated Radiation Therapy

Abstract

A METHOD FOR PREDICTING DOSE CHANGES FOR HN TREATMENT USING SURFACE IMAGING

By Sarah Elizabeth Holler, B.S.

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

Virginia Commonwealth University, 2022

Director: Dr. Christopher Guy, Assistant Professor, Department of Radiation Oncology

Head and neck cancer (HNC) is commonly treated with radiation therapy. A conventionally fractionated radiotherapy treatment course for HNC lasts six to seven weeks, and during this time, the patient's anatomy may change substantially. The tumor and organs at risk may all reduce in volume, and patients often experience weight loss due to side effects of radiation such as xerostomia, dysphagia, and loss of taste. Anatomical changes lead to a reduction in treatment quality due to decreased setup reproducibility and altered dose deposition compared to the original plan. This may necessitate a mid-course resimulation and replan; however, few clinics have developed a standardized method for determining if patients need replanning, which may lead to overlooking patients who need it or wasting resources on patients who do not. This work investigates a new method for determining when to resimulate and replan HNC patients by utilizing their topographic anatomical changes to predict differences in planned versus delivered dose distributions.

The first part of this work presents a method for deformable image registration of CT to CBCT, developed using 15 retrospectively selected HN patients, which addresses the challenges of inaccurate Hounsfield units and truncated field of view present in the CBCT. The registration method was validated on 10 additional HN patients using contour comparison, with average

DSC of 0.82, 0.74, 0.72, and 0.69 for the mandible, cord, and left and right parotid. This is comparable to previous results found in the literature. The registration method was then used to generate dose maps and surface contours for 47 patients for the second part of this work.

The second part of this work involved the development of a U-Net which takes the original dose distribution, the original surface, and the treatment day surface as input and predicts the treatment day dose distribution as output. Each of the 47 patients had three deformed CTs evenly spread throughout the treatment course for the generation of treatment day surface and dose, resulting in 141 data sets for training and testing of the network. The network training was performed with 135 of these data sets, while the remaining 6 were used for testing. The average RMSE and MAE between the true and predicted dose distributions for the 6 test cases was 4.25 and 2.15.

The deformable image registration method was successful at producing reasonable deformations between CT and CBCT for the purpose of dose-of-the-day calculation. This work proved feasibility of the dose prediction neural network and shows promise for future clinical applicability.

Chapter 1 Introduction

The introductory chapter of this thesis presents the purpose of this work and background information relevant to it.

As oncology advances, treatments are becoming more and more customized to individual patients, with the goal of minimizing side effects while effectively killing cancer cells. This is especially important in radiation oncology, where the lines between effective treatment and not eradicating the cancer or causing lethal toxicity can be razor thin. In the head and neck (HN) region, delivery of excessive dose to the organs at risk (OARs) can result in severely decreased quality of life (QOL) due to dry mouth, loss of taste, and difficulty swallowing, among other symptoms[1–4]. On the other hand, underdosing the tumor can leave the door open for failed treatment, metastasis, or recurrence. While more complex treatments such as intensity modulated radiation therapy (IMRT) and volumetric modulated arc therapy (VMAT) have improved both outcomes and side effects, there is still progress to be made in the form of adaptive radiation therapy (ART). ART seeks to adapt treatments mid-course to account for anatomic changes which affect dose deposition. This thesis describes the exploration of neural networks as a tool for more efficient, data-guided ART, specifically for HN cases. The eventual goal of this work is to have a fully-fledged clinical tool which radiation oncologists could use to quickly and easily determine when a treatment adaptation is necessary.

This work uses computed tomography (CT) and cone beam computed tomography (CBCT) images from pre-treatment and three mid-treatment time points to generate estimated dose distributions and external body surfaces. The neural network is trained, tested, and

validated on the generated data. This is the first work of its kind to use surface information via neural network as a means to predict dose distributions.

1.1 Head and Neck Cancer

Head and neck cancer (HNC) was the seventh most common cancer globally and accounted for 3% of all cancer incidence and 1.5% of all cancer deaths in the US in 2018, with lip and oral cancer specifically contributing more deaths than any other HN site[5,6]. HNC encompasses a number of primary cancer sites within the region, often arising from the squamous cells of mucus membranes. They can be split into regions within HN: oral cavity, sinonasal cavity, pharynx, nasopharynx, and larynx. While in the past, tobacco and alcohol have been the leading cause of HNC, tied to 75% of cases at one point, human papillomavirus (HPV) has since become the leading cause, now accounting for over 70% of HNC[7].

In the tumor-node-metastasis (TNM) staging system, HNC tumors are staged by tumor size and/or location, lymph node involvement, and presence of metastases. Tumor characteristics are designated as Tis (for in situ), T1, T2, T3, T4a, and T4b, with some variation depending on disease site, and increasing numbers indicating increasing tumor size and spread; nodal involvement has similar designations and similar variations with disease site. Distant metastasis is indicated by M0 for no distant metastasis and M1 if distant metastasis is present. Each of the three scores contributes to determination of the overall stage of disease. For example, an oral cavity mass measured between 2 and 4 cm in its largest dimension, with involvement of a single same-sided lymph node, and no distant metastasis, would be designated as T2, N1, and M0, and diagnosed as Stage III. For oropharynx cases, HPV status must be determined as well. HPV-positive patients have a more favorable prognosis, as HPV-

mediated cancers tend to have better treatment response and HPV-positive patients tend to be younger and fitter with fewer coexisting conditions than HPV-negative patients. Nodal involvement is the greatest predictor of patient prognosis in HNC, and there are six nodal groups commonly investigated by physicians: submental, upper, lower, and middle cervical, posterior triangle of the neck, and supraclavicular. Metastasis is usually not seen until the late stages of HNC, and generally occurs via the lymphatic system.

The primary curative modalities are surgery and radiation, although chemotherapy is often used in conjunction. Radiation therapy may be used definitively or adjuvantly following surgery, with or without chemotherapy, depending on tumor staging and location. Surgery or radiotherapy alone are generally sufficient to cure Stage I and II disease, which accounts for 30-40% of HNC patients. Surgical removal of lymph nodes or prophylactic radiation of the neck may be considered to reduce chances of recurrence or further spread. For the more than 60% of cases who present with Stage III or IV disease, multimodality treatment is used to increase chances of survival while attempting to preserve function and quality of life. Survival rates for locally advanced disease are less than 50% at five years. Survival rates for HNC are generally better than cancers of different anatomic origin, due to the tendency of tumors in this region to spread locally, as opposed to distantly. However, side effects of treatment are of great concern, especially as the population of HNC survivors trends younger.

1.2 Current Practice of Head and Neck Radiotherapy

Radiation therapy (RT) of HNC is commonly used definitively, pre- or post-operatively, and/or in conjunction with chemotherapy. Treatment techniques have evolved over the years, from simple 2D or 3D conventional treatments to three-dimensional conformal radiotherapy

(3DCRT) to intensity modulated radiation therapy (IMRT) and volumetric modulated arc therapy (VMAT), made possible by improvements in imaging and beam delivery. Conventional treatments use relatively few beams shaped by jaws, often requiring decisions to be made about sacrificing normal tissue function. 3DCRT uses forward planning, where the number of beams and their parameters are chosen iteratively as planning goals are assessed after beams have been designed or adjusted, and multiple beams made by static multi-leaf collimators (MLC) shapes to conform to the target. IMRT uses inverse planning and beams modulated by MLCs to conform the dose to the target even further. In inverse planning, the TPS generates a fluence map to meet predefined planning goals, and then beam parameters and MLC positions are computed to achieve the desired fluence. The modulation may be achieved via a step-and-shoot (static) delivery technique, where the field is divided into segments and the beam is off while the leaves move to form the different beam shapes, or a sliding window (dynamic) delivery technique, where radiation is delivered while the leaves migrate and the timing of the movements determine the intensity of the radiation. VMAT is an extension of IMRT where radiation is delivered while the gantry and the MLCs simultaneously move, with varying speeds and dose rates. 3DCRT, IMRT, and VMAT are markedly better than 2D conventional techniques due to the ability to shape the beams (3DCRT) or dose (IMRT and VMAT) to the target, thereby sparing the surrounding normal tissues. Studies and systematic reviews of literature have also demonstrated that IMRT and VMAT may further improve acute and late toxicities over 3DCRT, with equivalent or better tumor control and survival[8–11].

When RT is prescribed for a patient's treatment using these advanced treatment techniques, they must first undergo a treatment simulation. During simulation, the patient is

oriented on the CT scanner table in the position planned for treatment, using the indicated immobilization devices. The planning CT (pCT) scan is taken at simulation with all devices in place. The primary immobilization device for HNC is a thermoplastic mask, which covers the head and may extend past the shoulders (Fig 1.1). This mesh-like mask is malleable while warm and molds to the patient's anatomy. As it cools, the mask hardens. This same mask is used throughout the patient's treatment course to reproduce the simulation position and minimize intra-fraction motion. Minimizing inter- and intra-fraction motion, enabled by properly fitted immobilization devices alongside proper inter- and intra-fraction imaging and monitoring, allows smaller PTV margins and accurate delivery of the planned dose distribution while avoiding potential geometric miss, which is critical to avoid underdosing of the target or overdosing of normal tissue. In the case of a mask which does not cover the shoulders, straps may be equipped to pull the patient's shoulders down.

Fig. 1.1 Thermoplastic mask used for immobilization for HNC radiotherapy.

A treatment plan is designed based on the pCT by a dosimetrist or physicist in conjunction with the physician; one to two weeks after simulation, the patient begins treatment. For HNC, it is common to use a simultaneous integrated boost (SIB) treatment plan for patients being treated definitively with radiation. An alternative method, sequential boost, involves delivering a "boost" to the tumor volume following delivery of a 3DCRT or IMRT treatment course; however, SIB has been found to achieve superior conformality, potentially significantly reducing doses to OARs[12,13]. SIB plans often consist of a high-risk target receiving 70 Gy, an intermediate risk target receiving 60 Gy, and a low risk target receiving 54 Gy (Fig. 1.2). Prescriptions for post-operative patients are lower, with the high-risk target receiving 60-66 Gy[14]. When designing the patient's treatment plan, the physician draws the gross target volume (GTV) contour around the bulky disease visible on imaging. An expansion is applied to account for microscopic spread of disease beyond what is visible on available imaging, resulting in the clinical target volume (CTV). A second expansion is utilized to account for random and systematic setup errors, resulting in the planning target volume (PTV), to which the physician prescribes dose. A typical prescription calls for 95% of the PTV to receive the prescription dose, while not exceeding a maximum dose of 115% of the prescription dose. As mentioned previously, in the case of SIB, each PTV in the set is designated as either high-, intermediate-, and low-risk. High-risk PTVs typically include the primary disease, intermediate PTVs include regional microscopic spread, and low-risk PTVs include the electively treated lymph nodes in the neck. In addition to being optimized on the targets, the treatment plan must spare OARs as much as possible and adhere to standardized dose constraints to avoid severe toxicities. For

HNC, critical OARs include the parotid glands, spinal cord, brainstem, and mandible, whose constraints are listed in Table 1.1.

Table 1.1 Critical OARs in HNC		
Structure	**Constraint**	**Endpoint**
Parotid gland	Mean dose < 26 Gy	Reduced salivary function
Spinal Cord	Max dose < 45 Gy (< 40 Gy if achievable)	Myelopathy
Spinal Cord +5mm	Max dose < 50 Gy	
Brainstem	Max dose < 54 Gy	Neuropathy or necrosis
Mandible	Max dose < 70 Gy	Osteoradionecrosis

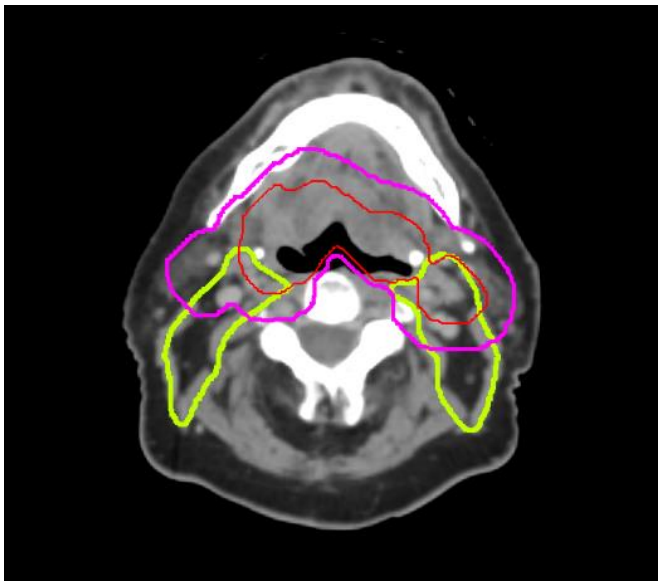


Fig. 1.2 Planning CT for SIB treatment with high-risk PTV indicated in red, intermediate-risk PTV indicated in magenta, and low-risk PTV indicated in lime.

This treatment scheme is generally delivered in 33 treatment sessions, called fractions, of 2.12/1.82/1.64 Gy each, 5 days per week for 7 weeks. At each fraction, kilovoltage (kV) orthogonal images or cone-beam computed tomography (CBCT) scans can be used to verify patient setup. One strategy for pre-treatment imaging for HNC patients is to image daily with orthogonal kV images and weekly with CBCT. The bony anatomy contained in kV orthogonal images is only a surrogate for the soft tissue position but sufficient for treatment alignment on a daily basis (Fig. 1.3). CBCT provides better soft tissue contrast for visualization of the external surface of the patient as well as the gross disease and can be used to monitor anatomical changes during the course of radiotherapy (Fig. 1.4). However, CBCT possesses its own unique set of flaws which limit its use to setup imaging. Unlike CT, the FOV is truncated and the HU values acquired by CBCT are inaccurate, making it challenging to use for dose calculation. These challenges will be expanded upon in Chapters 2 and 3.

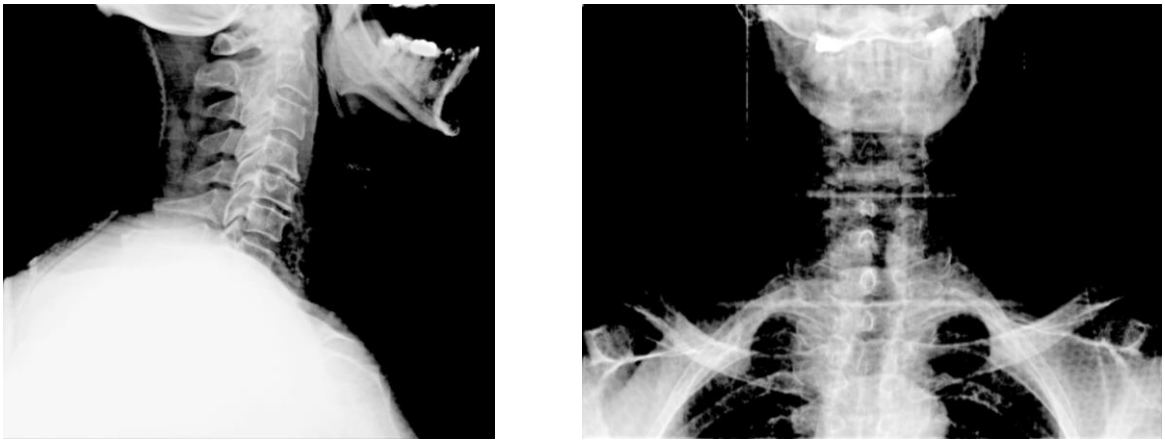


Fig. 1.3 Orthogonal kV images taken during HN RT.

Throughout the course of treatment, the patient's anatomy is known to change. Some change is intended, specifically tumor volume shrinkage; targets have been observed to shrink

by over 3%/week[15–21]. Other changes are undesirable side effects of normal tissue resulting from the radiation treatment. The parotid glands have been observed to shrink by up to 21.3% over the course of treatment, though these changes vary from patient to patient, and the ipsilateral parotid gland generally shrinks more than the contralateral[22]. Both target and parotid shrinkage are associated with cell death and vascular damage due to radiation. Additionally, patients may lose weight due to side effects such as xerostomia, dysphagia, and loss of taste. Anatomical changes significantly affect the delivered dose distribution to the targets and OARs, resulting in differences from the original treatment plan.

Upon inspection of the pre-treatment volumetric imaging, the physician may decide that the patient's anatomy differs significantly from the anatomy within the pCT. In such cases, the physician may order a new simulation and subsequent treatment plan to be used for the remaining treatment fractions. This may also occur if there is increased difficulty setting up the patient for treatment due to anatomical changes and ill-fitting immobilization devices often caused by patient weight loss. After ordering a resimulation, the physician may elect either to continue treatment with the original plan until the new plan is ready or to hold treatment until the new plan is ready. This means that while the new plan is being prepared, the patient is either receiving potentially suboptimal treatment or not receiving any treatment, both of which may negatively affect the patient's outcomes and toxicities. Multiple studies have found that hot spots increased, while coverage of the PTV suffered, and the OARs received higher doses when patients lost significant weight during treatment and were not replanned[16,23–27].

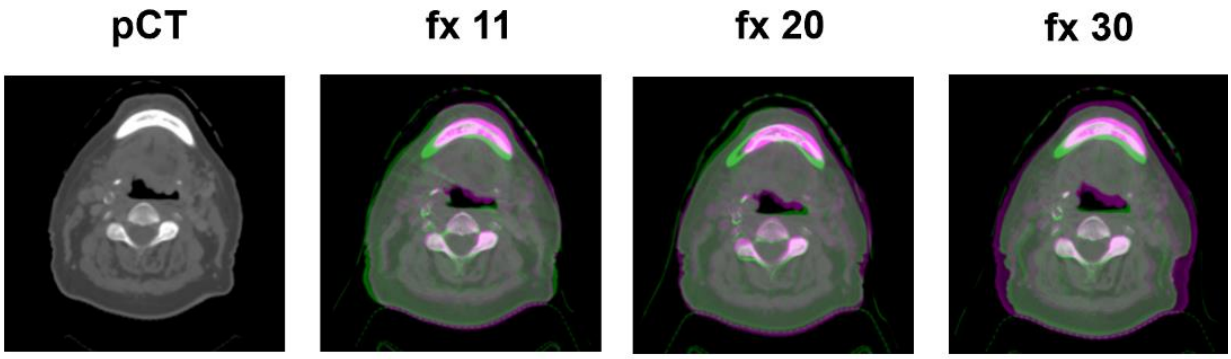


Fig. 1.4 CBCT images (green) taken over the course of HN RT registered with the pCT (magenta).

1.3 Adaptive Therapy Strategies

Adaptive therapy is the practice of changing a patient's treatment plan due to anatomic or biologic factors during the treatment course. This can occur in three different ways: offline adaptation, online adaptation, and intrafraction adaptation. Intrafraction adaptation seeks to address changes in the patient anatomy on the order of seconds, which may be a result of breathing motion or other bodily functions. Online adaptation and offline adaptation are similar to each other in that they both result in the generation of an entirely new treatment plan. However, online adaptation is done at the treatment machine with the patient on the table. This is commonly done on linacs with additional imaging capabilities, such as magnetic resonance imaging (MRI) or CT-on-rails, where high quality images of the patient can be captured each treatment day. However, some studies have sought to develop artificial intelligence methods for online adaptation using CBCT to generate new contours and plans, with demonstration of feasibility for the process[28,29]. Offline adaptation is done between treatment fractions and may include repeat simulation if onboard imaging is not sufficient for planning. Various adaptive planning methods have been explored for many anatomical sites,

including HN, lung, liver, and prostate[30–33]. Regardless of the site, it has been demonstrated that adaptive therapy, when executed on the right patients at the proper time, results in improved target coverage and reduced dose to the OARs. Any adaptive therapy method requires careful planning to avoid disruption to the clinic and establish an efficient workflow, given the resources required for adaptation.

The process of replanning is known to be time-consuming, taking multiple days to produce a new plan[24,34,35]. Though there is no current standard method for HNC ART, individual clinics have executed a variety of methods to standardize this process to only replan the patients who benefit most[24]. One study used a nomogram to predict parotid gland overdose, using the planning CT and a CT from the first treatment week to obtain the volume of the CTV planned to receive 70Gy, the difference between that volume in the pCT and the second CT, and the mean parotid gland dose difference between the pCT and the second CT; the nomogram correctly predicted patients who would have a >2.5 Gy overdose to the parotid glands, indicating they would benefit from weekly adaptive planning[36]. Another study developed a patient selection method based on multivariate linear regression which could theoretically predict, based solely on pre-treatment parameters, which patients would benefit most from adaptive therapy; a separate study used similar parameters to create a decision tree[37,38]. Brown et al. simply used pre-treatment nodal size to determine patients more likely to need a replan; in other work, the same authors determined more specifically that nasopharynx cancer (NPC) patients with large nodes may require replanning at week 3 of treatment, while oropharynx cancer (OPC) patients with large nodes may require replanning at week 4 of treatment[39,40]. Deformable image registration (DIR) between the planning CT and the daily

CBCT has also been used to recalculate the dose at each fraction, leading to fewer unnecessary replans and no missed cases, though this was a small study[41]. Other studies have indicated that adaptive planning is more effective in definitive treatments than in adjuvant treatments, which could help with patient selection[42]. Brown et al. also investigated virally mediated HNC (Epstein Barr positive or HPV positive), as these cancers are known to be highly responsive to radiotherapy, and found that those who presented with larger nodes were at a higher risk for needing replanning[43].

Many studies use pretreatment indicators of risk for needing a replan, which can be less accurate and predictive than desired. Relying on daily DIR to assess dosimetric changes, while more accurate, is time-consuming and may not be feasible in the average clinic. However, multiple studies note significant benefit to patients who have been properly selected for adaptive therapy. Reduced dose to OARs, improved target coverage, reduced toxicities, and improved overall survival and locoregional control have all been seen in various studies[16,27,44–51].

1.4 Surface Imaging in Radiotherapy

Surface imaging is a non-radiographic imaging technique used in RT to guide setup and treatment in much the same way as its radiographic counterparts, without the ionizing radiation. There are several different commercially available surface imaging systems, all operating with similar methodology. Systems compare a reference surface of the patient to a surface obtained in real time during treatment and report deviations in the form of error bars on 6 degrees of freedom (DOF). One surface imaging system is AlignRT (VisionRT, London, United Kingdom), with three camera and projector pods so the patient can be imaged

regardless of gantry angle (Fig. 1.5). The reference surface is either extracted from a user-selected contour associated with the pCT (i.e. the body contour) or acquired in the treatment room by the system. Surfaces are described by vertices and triangular faces, and the density of points used for surface definition varies according to software settings. The comparison surface is the patient's surface as they are immobilized on the treatment table. The projectors cast visible light that is colored red and in the form of a pseudo-random speckled pattern onto the patient's body. Optical cameras use the observed textural pattern and triangulation to create a 3D map of the patient surface[52–55]. The surface captured by the cameras may not include all of the anatomy in the reference surface, since the cameras only capture the anterior surface of the patient, if the patient is laying in a supine position on the treatment table. A rigid registration algorithm compares the two surfaces within the user-defined region of interest (ROI). Positioning uncertainty of AlignRT has been reported to be <1mm and <1°[56]. In a typical clinical workflow, therapists will position the patient on the table with the appropriate immobilization devices from simulation followed by the use of surface imaging to align the patient more accurately and precisely before exiting the room and performing radiographic imaging or starting treatment. Surface imaging may also be used to monitor the patient throughout treatment, as it does not impart ionizing radiation dose to the patient.

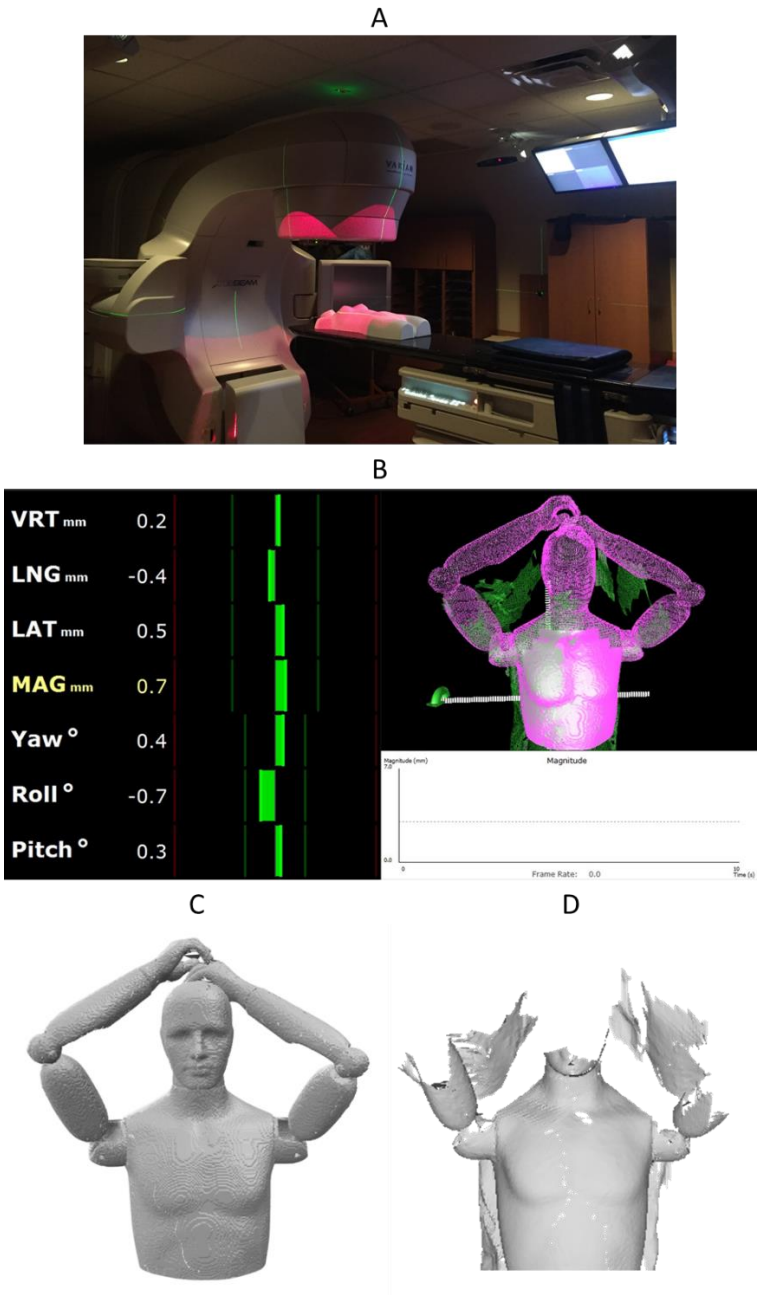


Fig. 1.5 Images demonstrating AlignRT System. A: Projection of red speckle pattern on phantom. B: Software display depicting alignment of reference surface and real-time surface along with error bars. C: Reference surface, generated from body contour. D: Real-time surface obtained using AlignRT cameras.

Surface imaging has a variety of applications in RT, including reducing margins for targets and OARs by improving setup, improving patient comfort via removal of immobilization devices, minimizing radiographic imaging dose, and allowing gating when breathing motion is relevant to treatment[52,57,58]. It is particularly useful in regions where the surface is a good surrogate for the treatment target, such as with the shallow targets seen in breast. Previously, it has been investigated for and found use in treatment sites such as breast, accelerated partial breast irradiation, chest wall, abdominal and pelvic sites, extremities, intracranial stereotactic radiosurgery (SRS), and head and neck[52,55]. Of particular interest are SRS studies, where high precision in positioning is especially critical. Surface imaging studies in SRS have used surface imaging in place of the typical frame and mask system and found that with an open face mask or no mask at all, similar precision could be achieved, and greater patient comfort was possible[58]. Additionally, setup with surface imaging took significantly less time than the conventional frame-based approach. In HN studies, patients were outfitted with open face masks to reduce patient anxiety, and setup errors between closed and open face masks were found to be comparable[56,59–61]. Open face masks also allow for more flexibility in using bite blocks and may reduce the bolus effect that causes skin toxicity. All cited studies used the patient surface as a surrogate for internal anatomy and were able to detect intrafraction surface changes with reasonable accuracy, suggesting that surface imaging could be extended to detect interfraction surface changes with similar accuracy.

1.5 Specific Aims

Given the known anatomic and dosimetric changes observed during a HNC patient's treatment course and the difficulty in determining the need for adaptive planning, the overall

aim of this work was to develop a neural network to aid adaptive therapy decisions, using machine learning capabilities to detect the effect of anatomic changes on the planned dose distribution. In pursuit of that aim, it was necessary to develop a deformable image registration workflow to handle the CT-CBCT registration task, which would enable weekly dose calculation without altering the current clinical imaging strategy.

The hypothesis of this work is that changes in the patient's surface can be used to predict changes in the radiation dose distribution and therefore determine and inform the treatment team of when re-simulation and re-planning should be considered to minimize differences between the expected and the delivered dose distributions.

Specific Aim 1

To develop a methodology for creating a synthetic CT and calculating delivered dose during a treatment course.

Specific Aim 1.1

To develop an approach for deformably registering a planning CT to a CBCT to address field of view and artifact issues.

Specific Aim 1.2

To evaluate the accuracy of the synthetic CT creation on 10 new HN cases.

Specific Aim 1.3

To calculate delivered dose and quantitatively compare planned dose distribution with delivered dose distribution in HN patients.

Specific Aim 2

To develop a tool that requires user input of patient surfaces and predicts changes in dose distribution.

Specific Aim 2.1

To create a neural network which uses initial patient surface, initial dose distribution, treatment day surface, and treatment day dose distribution as training data.

Specific Aim 2.2

To evaluate the dose prediction network using dose statistics and a cohort of reserved cases.

Chapter 2 Introduction to Deformable Image Registration

This chapter reviews the fundamentals of image registration as it relates to the work presented in Chapter 3.

2.1 Image Registration

Image registration is the process of finding the geometric relationship between two images. The purpose of image registration in RT may be to guide treatment planning or patient setup or to simply provide additional information about a patient's disease. Typical images used in RT include (but are not limited to) CT, CBCT, magnetic resonance imaging (MRI), positron emission tomography (PET), radiographs, and surface images. Apart from radiographs, which are two-dimensional, the modalities listed here all produce three-dimensional image sets. Image registration may be mono- or multi-modality. For instance, a patient being treated with RT a second time in the same anatomical region would need mono-modality image registration of the CTs from their first and second treatments to ensure appropriate sparing of OARs, while a patient whose tumor is not well-visualized on CT may need multi-modality registration of their CT with a PET or MRI for accurate target delineation. There are many aspects of image registration to consider when attempting to register two images, but the most important are the transformation model, the registration cost function, and the optimization procedure.

2.1.1 Transformation Model

Geometric transformations map points from one image space, X, to their corresponding points in a second image space, Y[62]. A transformation T acts on a point x in X to produce a transformed point $x' \in Y$,

$$x' = T(x), \tag{2.1}$$

where x and x’ are column vectors. A successful registration results in x’ and y being equal if point y in Y corresponds to x in X. A registration error is the existence of a displacement between y and x’. Transformations can be divided into rigid and nonrigid transformations. Rigid transformations preserve distances, straightness of lines, and angles, while deformable transformations may not preserve any of these features as the deformation is allowed to vary nonuniformly throughout the image volume.

Rigid transformations consist of translations and rotations, given by

$$x' = Rx + t, \tag{2.2}$$

where the rotation is given by R, a 3x3 orthogonal matrix, and the translation is given by t, a three-dimensional vector. In medical images, translations may occur in the superior-inferior, anterior-posterior, and lateral directions, and the rotations may occur around those axes. These transformations are useful when the object of interest is rigid anatomy, such as bones, or other anatomy which maintains rigidity between images. The degrees of freedom, or number of parameters, for the rigid transformation shown in equation 2.2 is 6, a rotation and translation in each physical dimension.

To model deformable transformations, a significantly larger number of degrees of freedom are necessary than for rigid transformations. At the extreme, an independent translation for each voxel of the fixed image can be calculated, equating to 3 x n voxels in degrees of freedom in a 3D image. Not only is such a transformation model computationally prohibitive, but also medical images deal with anatomy of patients which does not deform with enough complexity to require such a free-form transformation model. Instead, deformable

transformations are parameterized to a much smaller number of degrees of freedom. This not only speeds up the registration algorithm, but also enforces certain properties on the resulting deformation.

A common parametric approach to deformable transformation is B-splines. B-splines are a basis set of piecewise polynomials which define the space by dividing it into a set of hexahedrons, whose corners are called knots or control points. The control points are arranged on a regular grid over the fixed image, different from the voxel matrix of the image, and the transformation of a single point can be calculated from only a couple of surrounding control points. The equation for a B-spline transform is as follows:

$$T_\mu(x) = x + \sum_{x_k \in N_x} p_k \beta^3 \left(\frac{x - x_k}{\sigma}\right) \quad (2.3)$$

where the control points are represented as x_k, $\beta^3(x)$ represents the cubic B-spline polynomial, p_k represents the B-spline coefficient vectors, σ is the control point spacing, and N_x is the set of all control points within the compact support of the B-spline. In this case, the degree of the B-spline, m, is 3, a common choice in DIR. If p_k are chosen well, the transformation will be degree m – 1 times continuously differentiable, ensuring the polynomials join smoothly across the interfaces of the rectangles.

Importantly, the transformation maps points from the fixed (or target) image to the moving (or source) image. This is referred to as backward mapping, where pixel values of voxels in the deformed image are pulled from the moving image, and is preferred to forward mapping, where pixel values of voxels in the moving image are pushed to the deformed image. Forward mapping of deformable transformation results in needing to interpolate pixel values in the

deformed image, since voxels in the moving image will not be an exact match to voxels in the deformed image.

2.1.2 Objective Function

The registration objective function, or cost function, defines the criteria by which images are matched. The objective function is often a linear combination of similarity metrics and penalty terms which quantify the goodness of match resulting from a given deformation field. The general form of this cost function is then:

$$C(T) = -S(T) + \gamma P(T), \tag{2.4}$$

where C is the overall cost, S is the similarity metric, P is a penalty term, and γ is a weight to control the effect of similarity versus regularity, all calculated on the transformation T. The two main types of similarity metrics are geometric-base and intensity-based, and these may be used in tandem as a hybrid cost function.

Geometric similarity metrics can take the form of point sets in each image; the aim of the registration in this case is to align corresponding markers between images. These markers may be drawn from the anatomy or may be implanted fiducials or some other extrinsic landmark. The transformation aligns the markers as closely as possible, minimizing the registration error between the points. However, the designation of the points is often not perfect and will carry its own error, called fiducial localization error (FLE). In a constrained problem such as rigid registration, aligning well-placed landmarks will result in minimal registration error for other points. Yet in deformable transformations, the selection of transformation model can greatly affect the accuracy of registration of non-landmark points.

Intensity-based methods use the scalar values of pixels or voxels to inform transformation. Under these methods, a similarity measure calculated over the pixel or voxel values is optimized to determine the correct registration. The similarity measure should be carefully selected to account for the properties of the images being registered and the relationship between the intensity values found in the two images. An ideal similarity metric returns low values when the same tissue types are being compared and high values when differing tissue types are compared. With this in mind, similarity metrics can be divided based on whether the registration is a mono-modal or multi-modal problem.

Mono-modal image registration involves two images of the same modality, i.e. CT-to-CT registration. In this case, intensities from the same anatomical feature in both images are expected to be equivalent or exhibit a linear relationship. If the intensities are equivalent, sum of square differences or absolute differences (SSD or SAD) are acceptable similarity metrics. If a linear relationship between intensities exists, the appropriate similarity metric would be cross correlation or correlation coefficient.

In the case of multi-modality registration, images of different types are registered (i.e. MR-to-CT). Anatomically-corresponding tissues often have intensities that are no longer equivalent or even linearly related. To address this, information theoretic measures have been employed. Of these, mutual information (MI) may be the most well-known. MI seeks to minimize entropy between the resulting deformed image and the fixed image. One derivation of mutual information begins with the joint histogram of two images. With two aligned images A and B', the joint histogram charts the intensities of each image. When the joint histogram is normalized, the probability distribution function is obtained:

$$PDF[j,k] = \frac{HIST[j,k]}{\sum_{j,k} HIST[j,k]} \tag{2.5}$$

where HIST is the joint histogram calculated for two images related by a transformation. Then, the joint entropy H can be calculated using the PDF:

$$H = -\sum_{j,k} PDF[j,k] \log PDF[j,k] \tag{2.6}$$

MI, as opposed to entropy, takes into account the change in the histogram caused by transformation. To calculate MI, one must calculate the marginal entropies H(A) and H(B') as well as the joint entropy H(A,B'):

$$H(A) = \sum_{j} (\sum_{k} PDF[j,k] \log \sum_{l} PDF[j,l]) \tag{2.7}$$

$$H(B') = \sum_{k} (\sum_{i} PDF[i,k] \log \sum_{l} PDF[j,k]) \tag{2.8}$$

The MI equation then follows:

$$MI(A,B') = H(A) + H(B') - H(A,B'). \tag{2.9}$$

When the images are correctly aligned, entropy will be at its minimum and MI will be at its maximum. However, it may be possible for the images to get misaligned and still end up with a maximum in MI; therefore, MI is not overlap invariant. An extension of MI, called normalized mutual information (NMI), was developed to solve this problem:

$$NMI(A,B') = \frac{H(A) + H(B')}{H(A,B')} \tag{2.10}$$

However, it also has been demonstrated to not be overlap invariant. Other approaches have been explored, but each has its drawbacks.

Penalty terms, used to regularize transformations, are often calculated using the first or second order spatial derivatives of the transformation and added to the cost function[63]. Weights can be applied to both the similarity metric and the penalty term to control the effect of one versus the other. Any penalty term can be used with any similarity metric, but bending energy is potentially the most common form. The equation for the bending energy penalty term in 2D is:

$$\mathcal{P}_{BE}(\mu) = \frac{1}{P}\sum_{\tilde{x}_i}\left\|\frac{\partial^2 T}{\partial x \partial x^T}(\tilde{x}_i)\right\|_F^2 \quad (2.11)$$

where $\tilde{x}_i$ are points over which the term is evaluated, and P is the number of these points. With this penalty term applied, the cost function will increase with sharp gradients of the transformation, such as high expansion abutting high compression, which reduces folding in the transformation.

2.1.3 Optimization Methods

The goal of optimization is to find the optimal registration according to an objective function, which is generally composed of a similarity metric and a regularization term constraining the transformation. Optimization methods solve problems with variables that are either continuous or discrete. Continuous optimization methods take the form:

$$\theta_{t+1} = \theta_t + \alpha_t g_t(\theta_t) \quad (2.12)$$

where θ is the vector of transformation parameters, t indicates the iteration number, α_t is the step size, and g indicates the search direction. The search direction is determined using the entire objective function. A commonly used continuous optimization method is gradient descent, which follows the direction that decreases the cost function magnitude

$$g = -\nabla_\theta(\theta) \tag{2.13}$$

However, a drawback of gradient descent and other deterministic gradient methods is the computational demand of calculating derivative information due to the high dimensionality of the data and search space. Instead, stochastic gradient descent uses an update formula which approximates the gradient to reduce the computational demand. A specific application of stochastic gradient descent developed by Robbins and Monro (RM) decreases the step-size with time to decrease inaccuracy[64].

Other optimization methods include the Quasi-Newton method, which uses second-order information about the cost function to achieve better theoretical convergence than gradient descent[65]. Another type of approach includes conjugate gradients, which can be used to minimize quadratic or generally nonlinear cost functions, and can also achieve higher precision. However, both of these approaches are computationally expensive and take far longer to implement for minimal improvement in precision, which makes stochastic gradient descent the preferred approach for many DIR problems.

2.2 Deformable Image Registration for CT-to-CBCT Registration

Deformable image registration is a valuable tool in radiotherapy, where its use ranges from calculating composite treatment plan doses to using multiple modalities to improve target and OAR delineation to automatic segmentation based on contour atlases. As outlined previously, different tasks require different choices in transformation, similarity metric, optimization method, and more. Each task also possesses its own unique challenges. CT-to-CBCT registration has been explored for “dose of the day” calculation to assist in dose monitoring as well as adaptive therapy[66–69]. While dense deformation field methods such as the

viscous fluid method have been investigated, most studies use cubic B-splines and have found them to be sufficient for CT-to-CBCT registration[66,68–71]. Many studies have used hybrid similarity metrics, most commonly providing anatomical information in the form of contours alongside an intensity measure, with some authors using selectively applied rigidity penalties as well[66,67]. Commercial and open-source algorithms have both been tested, and target registration errors in the range of 2-3 mm were reported[66,67]. However, the issue of the reduced field of view of the CBCT has very rarely been addressed in DIR studies.

2.3 Challenges of CT-CBCT Deformable Image Registration

The first aim of this work seeks to address one of the principal problems in deforming a pCT to a CBCT, which is the reduced FOV of the dCT stemming from using a truncated CBCT as the fixed image. The pCT must be deformed to the CBCT to acquire both the correct HU and the anatomy of the day to produce an accurate dose calculation for neural network input. However, designating the pCT as the moving image and the CBCT as the fixed image in traditional DIR applications results in the pCT being truncated to the smaller FOV of the CBCT to produce a reasonable registration. The pCT has an array size of 512 x 512 and a typical voxel size of 1.17 x 1.17 x 3 mm^3 (FOV 60 x 60 cm^2) (Philips Brilliance Big Bore, Koninklijke Philips N.V., Amsterdam, Netherlands). The CBCT array size is 512 x 512 with a typical voxel size of 0.511 x 0.511 x 1.99 mm^3 (FOV 26.17 x 26.17 cm^2) (Varian Truebeam, Varian Medical Systems, Palo Alto, CA, USA). In HN cases, the reduced FOV results in missing anatomy superior and inferior of the CBCT as well as in the posterior and lateral directions (Fig. 2.1). Radiation often passes through anatomy not visible on the CBCT; replacing the missing information is crucial to an accurate dose of the day calculation.

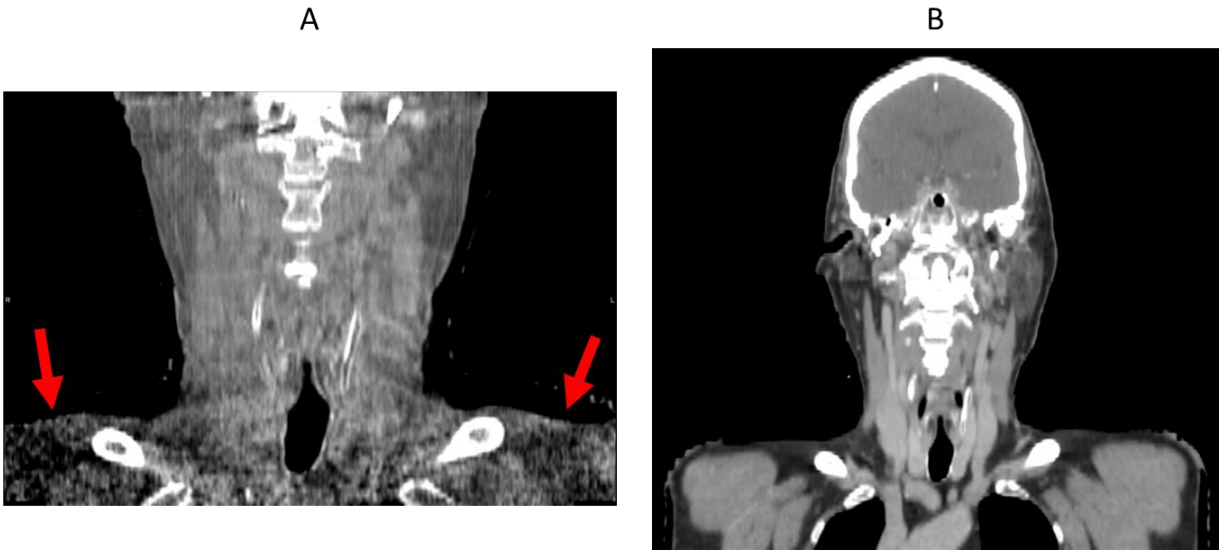


Fig. 2.1 CBCT (left) and pCT (right) for an example patient, showcasing the reduced FOV and image quality of the CBCT as compared to pCT. Arrows indicate regions of highest noise.

Choosing the correct similarity metric for DIR is imperative to obtaining accurate registrations. CT and CBCT differ in both their geometry and reconstruction methods, resulting in reduced image quality in CBCT (Fig. 2.1). While CT relies on a two-dimensional fan-beam geometry and a curved detector, meaning that rays on the edge of the detector have traveled approximately the same distance as rays at the center of the detector, CBCT uses a three-dimensional cone-beam geometry and a flat detector. The cone-beam X-ray geometry and flat detector results in a longer path-length from source to detector for rays on the periphery compared to those in the center. Additionally, the Feldkamp algorithm (FDK) utilized for CBCT reconstruction, an extension of the filtered backprojection algorithm (FBP) used in CT, makes several approximations which limit its accuracy; however, the FDK algorithm does not require significant computational resources to produce short reconstruction times, ensuring its continued use. Most importantly, CBCT lacks the collimation of detectors present in CT to reduce scatter contamination, which cannot be corrected through reconstruction algorithms.

These differences between CT and CBCT result in reduced image quality and increased artifacts in CBCT. Additionally, the image quality and artifacts are spatially variant, with image quality degrading further from the central plane, similar to images produced with the FBP algorithm. Due to this spatially variant difference in intensity between CT and CBCT, previous studies have had limited success with intensity-based registration methods, unless an intensity correction step is taken iteratively. This suggests that a mutual information metric may be better suited for CT/CBCT DIR. This informed the development of the DIR method detailed in the next chapter.

Chapter 3 Development of a Deformable Image Registration Method for CT-CBCT Registration

This chapter summarizes the work performed to achieve specific aim 1 as outlined in Chapter 1.5.

3.1 Initial Exploration

Initial work on the DIR method explored the use of a commercially available medical image software (MIM v6.7.9, MIM Software Inc, Beachwood, OH) with DIR capabilities. At the time, this DIR software possessed limited flexibility, allowing the user to choose between "image-based" and "normalized image-based" deformation. The transformations are intensity-based, as the algorithm is designed for mono-modality registration. Both transformations were tested on multiple patients to determine which transformation performed better for this registration problem. The normalized deformation proved to be appropriate for CT-to-CBCT registration, but the DIR module provided no solution to the truncated FOV issue. However, MIM allows for the development of automated workflows with a wide range of capabilities, and a workflow was created which could stitch together the smaller dCT with the larger pCT. The workflow took the pCT and replaced all image data inside the FOV of the CBCT with the dCT image data. Unfortunately, the joining of the two image sets was anatomically unrealistic and left large gaps in the shoulders (Fig. 3.1A). These gaps could be manually masked out and filled with soft tissue-equivalent HU values for adequate dose calculation, and investigation was undertaken to streamline or automate that process. Ultimately, however, the inflexibilities of this method resulted in the pursuit of a more adaptable method.

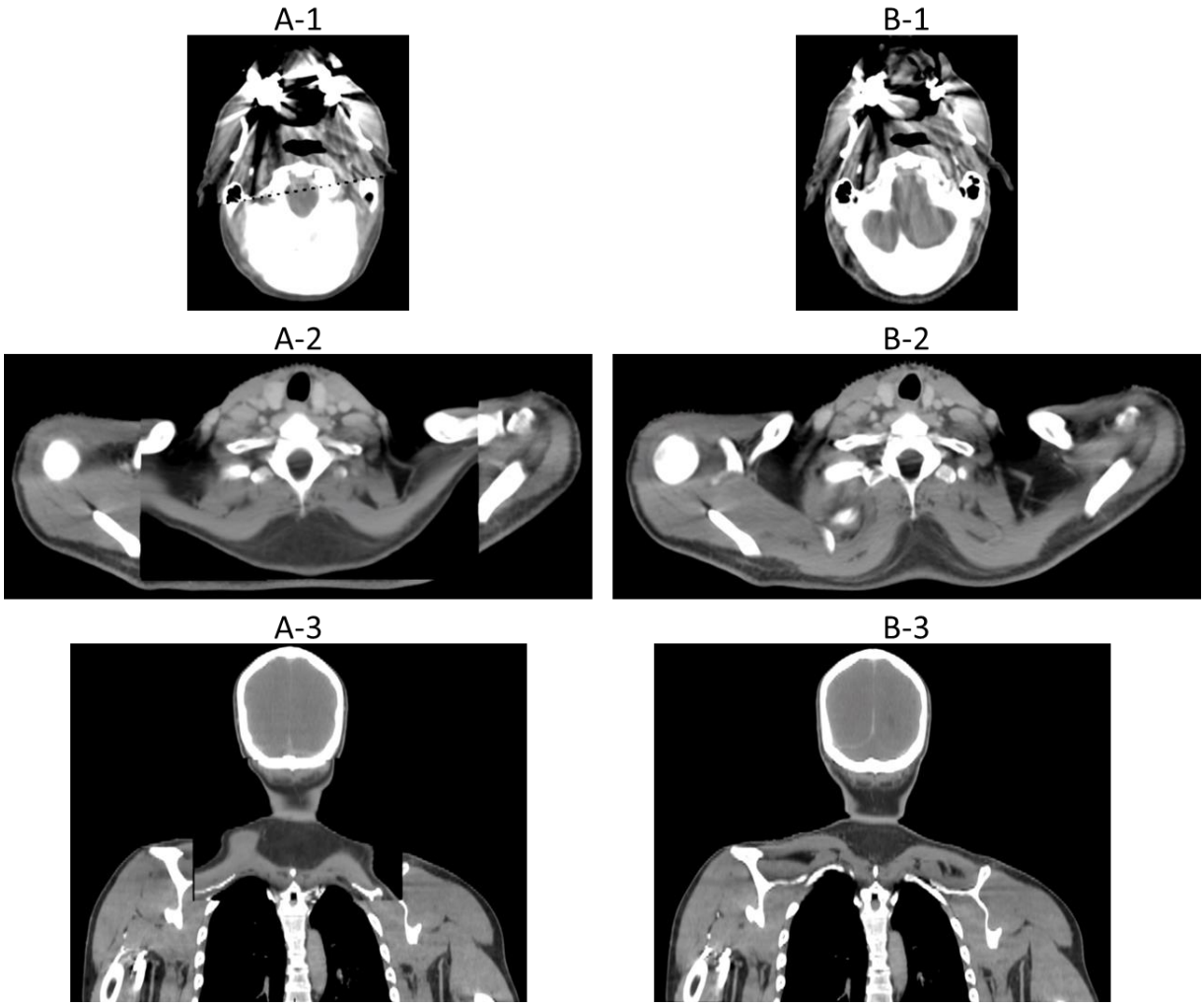


Fig. 3.1 A: pCT-to-CBCT deformation and stitching as performed in MIM. B: The dCT produced using elastix for the same image set. 1: axial view at the superior edge of the CBCT FOV, 2: axial view near the inferior edge of the CBCT FOV, 3: coronal view.

3.2 Development of `elastix`-based method

The open-source DIR algorithm `elastix` was investigated as a potential solution considering the drawbacks found with using a commercially available software. `elastix` is a parametric registration software package which allows for maximum flexibility in choosing parameters to create the best possible set of transform parameters for a given dataset[63]. While investigating parameters for CT-to-CBCT registration, it was discovered that the FOV of the pCT

could be preserved in the dCT by expanding the array size of the CBCT prior to registration. The resulting dCTs were smooth, exhibiting none of the discontinuities or gaps of the image replacement method used in MIM (Fig. 3.1B).

Fifteen HN patients with one pCT and three CBCTs each were retrospectively selected to develop the overall method and specific `elastix` parameters (Table 3.1). Patient selection was limited to those treated on Varian TrueBeam linacs (Varian Medical Systems, Palo Alto, CA), due to the superior image quality of the onboard imager (OBI) relative to other image guided radiotherapy (IGRT) systems available at our institution. Only patients with anatomic changes visible on CBCT were chosen for development of the method (as shown in Fig. 1.4). Parameter files were developed from an in-house registration parameter file and a previously published deformable registration parameter file[72].

Table 3.1 Patient Demographics for DIR Development

Pt #	Sex	Age	Disease Site	Stage
1	M	51	Larynx	T2N3
2	M	58	Oropharynx	T2N1
3	M	50	Oropharynx	T2N1
4	M	47	Oropharynx	T1N3
5	M	64	Oropharynx	T2Nx
6	M	63	Larynx	T3N2b
7	M	64	Neck	T0N2c
8	M	57	Oropharynx	T2N1M0
9	M	68	Nasopharynx	T4N2M0
10	M	71	Oropharynx	T3N2
11	M	57	Oral cavity	T3N1
12	M	62	Oropharynx	T2N0M0
13	M	54	Oropharynx	T3N2
14	F	47	Oropharynx	T3N2c
15	M	64	Oropharynx	T4a

The developed method used a combination of MATLAB (Mathworks, Natick, MA) and `elastix`, an open-source DIR algorithm. The pCT and CBCT were exported from MIM in mhd format. MATLAB scripts resampled and padded the CBCTs with air to the resolution and approximate size of the corresponding pCTs (Fig. 3.2). Padding was performed symmetrically in the anterior-posterior, left-right, and superior-inferior directions. Binary masks were also created corresponding to the FOV of the CBCT to ensure the algorithm only uses information inside the FOV for calculating the rigid and deformable registration. The mask was drawn slice-by-slice, creating circles with radius equal to half the width of the image, which were then stacked to generate a cylindrical mask. The most superior and inferior 3 slices were omitted from the mask entirely to avoid the decreasing FOV at the scan edges. Each registration was completed in `elastix` which included a rigid transform followed by a B-spline transform.

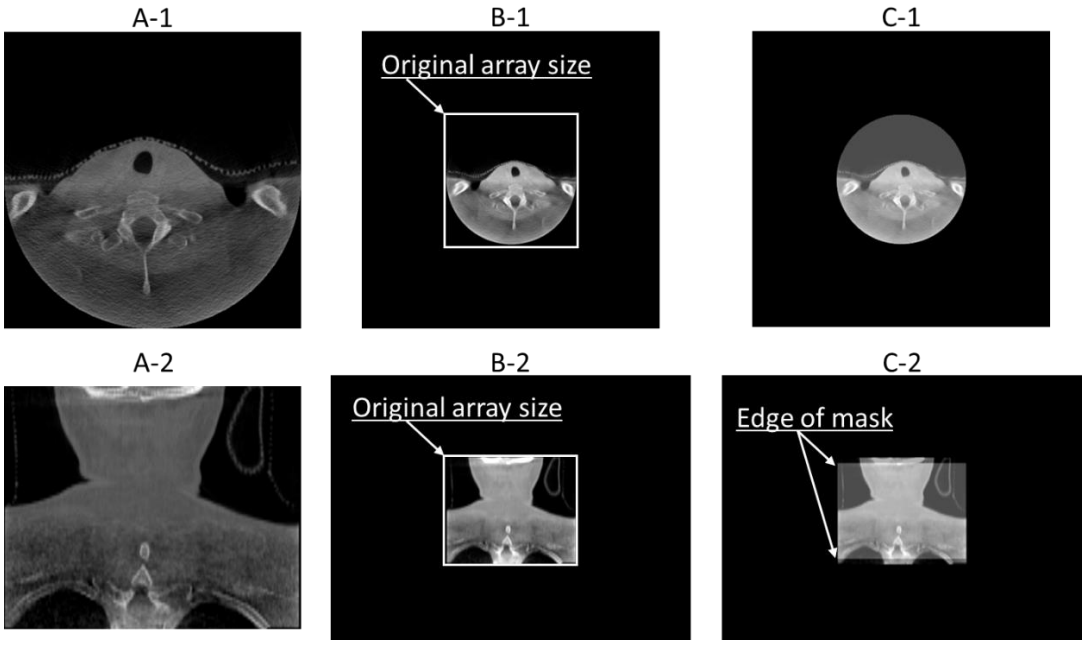


Fig. 3.2 A: Original CBCT. B: Resampled and padded CBCT showing the expanded array. C: Binary mask overlaid on resampled and padded CBCT. 1: Axial view. 2: Coronal view.

Deformable registration was performed using a B-spline transformation basis, defined by a uniform grid of control points overlaid on the fixed image (CBCT). Since a mask was used to indicate the CBCT FOV as the region of interest, the cost function was only calculated from sample points within the mask, and the deformation field smoothly decreased to zero across the boundary. This meant the anatomical information outside of the CBCT FOV in the dCT was not deformed but simply rigidly registered. The registration was performed with an anisotropic, multi-resolution image pyramid, such that in each of the 3 resolutions used, the superior-inferior dimension had less downsampling to account for the larger spacing inherent to the images. The deformable portion of the registration algorithm used a combination of mutual information and a bending energy penalty term for the similarity metric and regularization term, respectively. The bending energy penalty term penalizes abrupt changes in the transformation, such as large compression abutting large expansion. This may prevent the deformation from becoming irregular or folding in on itself. Weights for the two metrics were optimized to improve performance on this dataset. During development, the registration quality was assessed using the Jacobian map of the deformation vector field and visual inspection. The determinant of the Jacobian can be used to determine whether a physically possible deformation has occurred, while visual inspection ensures that bony anatomy and patient surfaces are reasonably aligned. Negative values for the determinant of the Jacobian indicate the deformation has folded in on itself (invalid); values between 1 and 0 indicate reduction in volume, while values greater than one indicate expansion.

Investigation of the Jacobian map revealed that mutual information alone was not sufficient to generate acceptable deformations, as the resulting transformations exhibited inaccurate and excessive deformation, including folding. The final parameters are listed in table 3.2.

Table 3.2 Rigid and Deformable Image Registration Parameters

	Rigid	Deformable
Image Pyramid	Gaussian Pyramid	Gaussian Pyramid
Sampler	Random	Random Coordinate
Interpolator	B-Spline	B-Spline
Optimizer	Adaptive Stochastic Gradient Descent	Adaptive Stochastic Gradient Descent
Transform	Euler Transform	B-Spline
Similarity Metric	Normalized Correlation Coefficient	Mutual Information
Regularization Term		Bending Energy Penalty

Following a successful registration, a MATLAB script converted the dCT from the mhd image format used by elastix to DICOM for importing back to the treatment planning system (TPS).

3.3 Validation of the DIR Method

Several methods were considered for validating the DIR method. Target registration error (TRE), which uses pre-defined point pairs on registered images to measure the distance between paired points after registration, is the gold standard for assessment of DIR. However, this process is time consuming and labor-intensive, requiring an expert to spend hours designating points which cover the entire registration volume. The next best option, which is widely used to assess DIR accuracy, is a quantitative comparison between contours drawn on

the fixed image and the deformed moving contours. Metrics were selected based on a literature search to determine commonly used statistics, and Dice similarity coefficient (DSC), Jaccard index (JI), and Hausdorff distance (HD) were chosen for validation[66,67,69,73–76].

The most frequently used contour analysis statistic, DSC calculates the spatial overlap between two segmentations; if A is a contour drawn on CBCT and B is the corresponding deformed contour, then

$$DSC = \frac{2(A \cap B)}{A + B} \tag{3.1}$$

where ∩ is the intersection. Perfect overlap results in a DSC of 1, and no overlap at all results in a DSC of 0. TG-132, a guidance document for the use of image registration in radiotherapy, detailed the current approaches to registration and recommended guidelines for quality assurance, including recommended DSC scores[77]. While TG-132 sets a goal of achieving DSC of 0.8-0.9 for validating image registration methods, this is not always achievable for every anatomical structure. Commercial algorithms have been found to fall short, specifically in HN, with an average DSC of 0.73 for HN OARs in one study directly assessing TG-132 recommendations[74].

JI is a stricter measure of the spatial overlap of two segmentations, calculated by

$$JI = \frac{|A \cap B|}{|A \cup B|} \tag{3.2}$$

where U is the union of A and B. JI also ranges from 0 to 1, and while DSC and JI are equivalent at 0 and 1, JI is always less than DSC between the two extremes.

HD calculates the greatest distance between a point in one set and the closest point in a second set. Given point a within A, the single-sided HD (HDS) is calculated as follows:

$$HDS_{max}(A,B) = max_a \in_A \{d_{min}(a,B)\} \quad (3.3)$$

where d_{min}(a, B) indicates the minimal distance between point a and any point in B. This can then be implemented to calculate symmetric HD as follows:

$$HD_{max}(A,B) = \max\{HDS_{max}(A,B), HDS_{max}(B,A)\} \quad (3.4)$$

A higher HD indicates dissimilarity between the two sets of points, while an HD of zero indicates that A and B are identical sets of points. Voxels within the contour take the place of points for HD when assessing segmentations.

Under the guidance of an expert physician, the mandible, cord, and parotids were selected for evaluation structures, due to their importance in treatment planning and outcomes for the patients of this work. Ten new HN cases were retrospectively chosen for validation (Table 3.3). Patient selection was again limited to patients treated on the TrueBeam, and patients with visible anatomic change were preferred. Structures were delineated by an expert physician on CBCT, and the corresponding pCT contours were deformed according to the transform parameters for the respective dCTs. MATLAB scripts then calculated DSC, JI, and HD for each pair of structures.

Table 3.3 Patient Demographics for DIR Validation				
Pt #	**Sex**	**Age**	**Disease Site**	**Stage**
1	M	55	Oral cavity	T4aN3b
2	F	19	Nasopharynx	T2N1
3	M	49	Neck	T0N2aM0
4	M	68	Oropharynx	T4aN0
5	M	66	Larynx	T3N0M0 (III)
6	M	64	Larynx	T3N2cM0 (IVA)
7	M	57	Larynx	IVA
8	F	48	Nasal cavity	T4N0M0
9	M	49	Oropharynx	T2N2b
10	F	64	Oral cavity	T3N0

3.4 Results

3.4.1 DIR Validation

Registrations appeared reasonable upon visual inspection. Quantitative results based on contours are summarized in Table 3.4. The mean DSC values were 0.78 and 0.71 for bony anatomy and soft tissue, respectively. The mean JI values were 0.65 and 0.56, respectively. The mean HD were 4.4mm and 3.6mm for bony anatomy and soft tissue, respectively.

Table 3.4 Contour Evaluation Scores for Validation Patients												
	Mandible			**Cord**			**Left Parotid**			**Right Parotid**		
	DSC	JI	HD	DSC	JI	HD	DSC	JI	HD	DSC	JI	HD
Pt1	0.70	0.55	5.1	0.83	0.71	2.4	0.71	0.56	4.1	N/A	N/A	N/A
Pt2	0.85	0.73	5.8	0.71	0.55	3.9	0.62	0.45	3.1	0.50	0.33	3.3
Pt3	0.82	0.69	7.5	0.69	0.53	2.9	0.78	0.64	3.3	0.81	0.67	3.5
Pt4	0.78	0.65	4.9	0.75	0.59	2.4	0.67	0.50	3.5	0.39	0.24	4.1
Pt5	0.87	0.77	6.1	0.74	0.59	2.6	0.77	0.63	3.1	0.82	0.69	3.7
Pt6	0.88	0.78	4.4	0.73	0.57	4.6	0.75	0.60	3.9	0.77	0.63	3.5
Pt7	0.90	0.82	4.1	0.71	0.56	2.6	0.79	0.66	2.6	0.80	0.66	3.7
Pt8	0.86	0.76	6.1	0.77	0.62	3.1	0.83	0.72	3.7	0.78	0.64	3.7
Pt9	0.80	0.67	7.3	0.67	0.51	2.9	0.68	0.51	3.5	0.72	0.56	4.3
Pt10	0.75	0.61	6.9	0.81	0.68	2.6	0.55	0.38	3.1	0.65	0.48	4.1
Mean	0.82	0.70	5.8	0.74	0.59	3.0	0.72	0.56	3.4	0.69	0.55	3.8
Stdev	0.06	0.08	1.1	0.05	0.06	0.7	0.08	0.10	0.4	0.14	0.15	0.3
Min	0.71	0.55	4.1	0.67	0.51	2.4	0.55	0.38	2.6	0.39	0.24	3.3
Max	0.90	0.82	7.5	0.83	0.82	4.6	0.83	0.72	4.1	0.82	0.69	4.3

The DSC and JI values for the mandible significantly outperformed that for the cord. Notably, the right parotid of Pt 4 had remarkably low DSC and JI values. This likely stems from the difficulty of contouring soft tissue on CBCT, as the other structures for Pt 4 perform reasonably well, and the artifacts and registration of the images do not differ significantly from the other validation patients. Fig 3.3A shows the difference between the original contours and the CBCT contours for Pt 4, which may explain the poor contour-based metrics. The SI extent of the right parotid structure in particular was much smaller in the CBCT than in the pCT. On the other hand, for Pt 7, the CBCT contours look much more similar to the original contours drawn on the pCT (Fig. 3.3B). This points to interobserver variability strongly affecting contour evaluation.

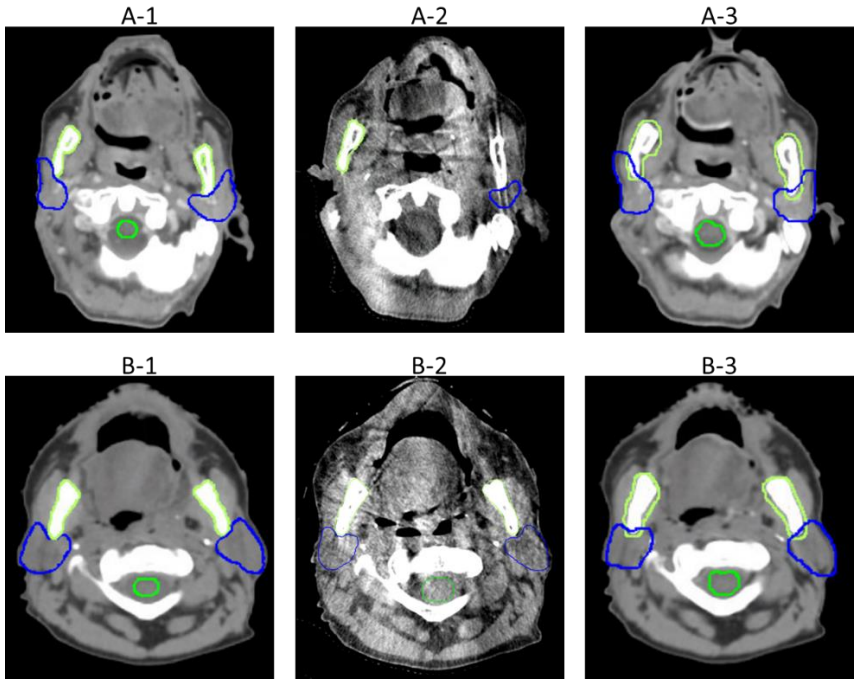


Fig. 3.3 A: Patient 4. B: Patient 7. 1: pCT with original contours. 2: CBCT with expert-drawn contours for comparison to deformed contours. 3: dCT with deformed contours. Mandible in lime, parotids in blue, and cord in green.

As previously stated, inaccurate HU values from CBCT images is a primary barrier to direct dose calculation. Therefore, HU values of the dCT were also evaluated. Histograms of the HU values for the whole body structure as well as all four evaluation structures were generated for the pCT, CBCT, and dCT (Fig. 3.4-5). Bin size was set to 10, and the histograms were overlaid for analysis. The desired outcome was for the pCT and dCT histograms to demonstrate reasonable agreement. This was generally achieved for all ten patients and all structures, apart from the mandible exhibiting a larger peak towards the lower end of the histogram. Visual inspection of the mandible contour revealed that the deformed structure included soft tissue outside of the mandible, which is likely what causes this peak (Fig. 3.6). The original pCT mandible contour included a small margin of soft tissue, and it expanded during deformation. The CBCT contour, however, stayed within the bone, another difference between observers.

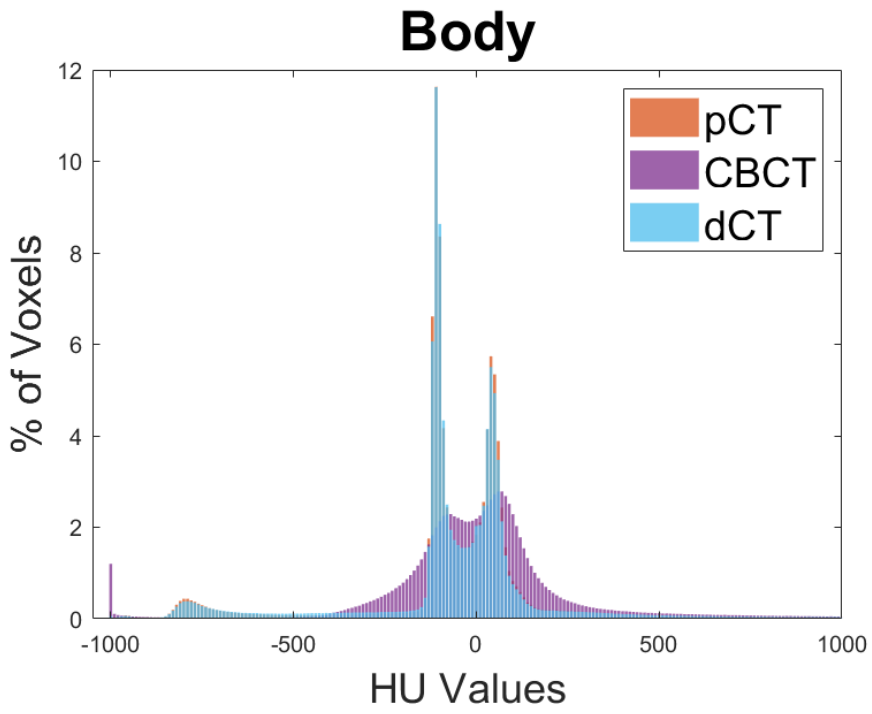


Fig 3.4 Histogram of HU values for the body structure for Pt 10. Bin size is 10 HU.

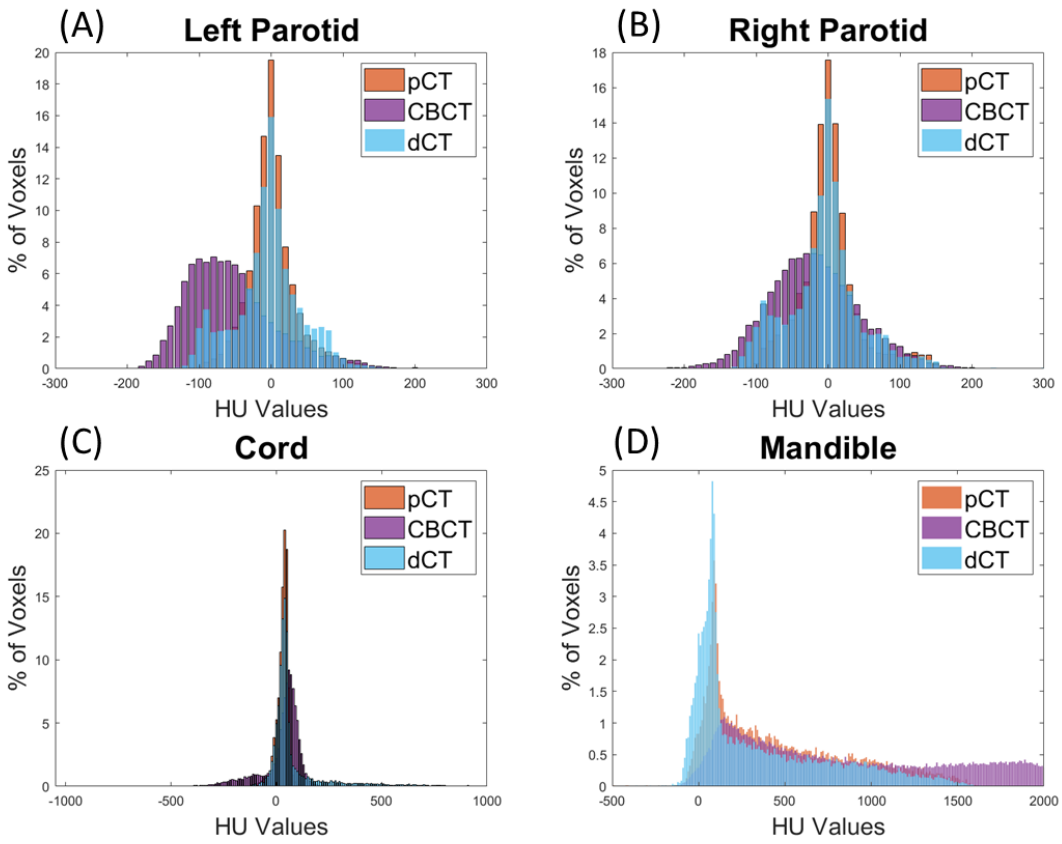


Fig 3.5 Histogram of HU values for (A) mandible, (B) spinal cord, (C) left parotid, and (D) right parotid for Pt 10. Bin size is 10 HU.

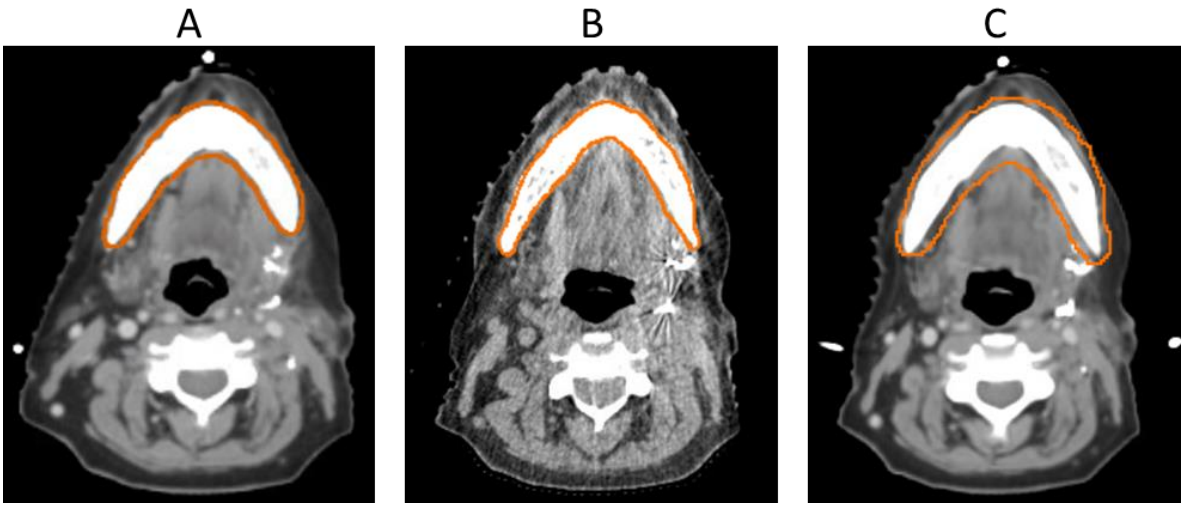
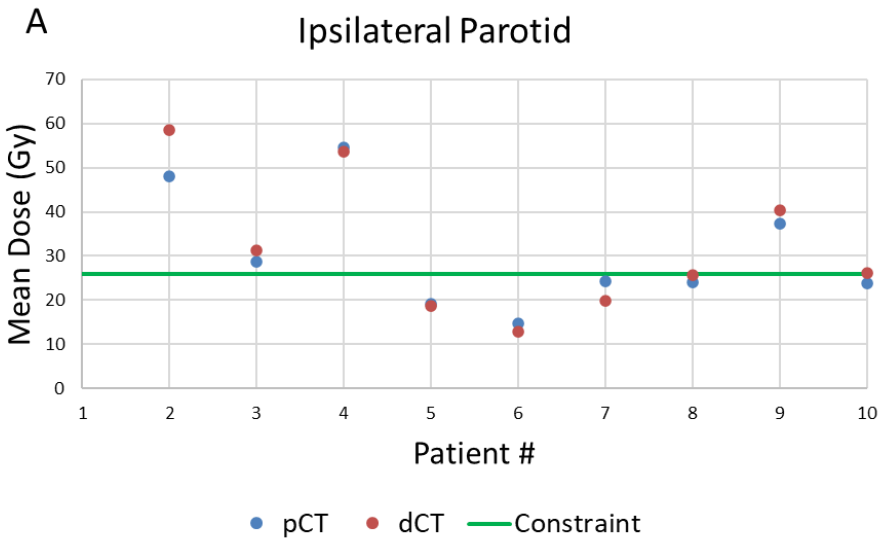


Fig 3.6 A: pCT, B: CBCT, C: dCT, with respective mandible contours in orange.

3.4.2 Dose Calculation on dCTs

Following validation of the DIR method, the resulting dCTs were used for dose calculation. The original treatment plan was transferred to the dCT and recalculated in Eclipse. For the ten patients used for DIR validation, the pCT dose was compared to the dCT dose in terms of typical OAR constraints and OAR DVHs.

The parotid gland dose was inconsistent across the ten patients, but the spinal cord and mandible dose were likely to be higher on the dCT than the pCT (Fig. 3.7). For patients with minimal anatomic change, DVHs for OARs appeared similar between the pCT and dCT (Fig. 3.8). However, patients with larger anatomic change displayed a marked difference in their DVHs (Fig. 3.9).



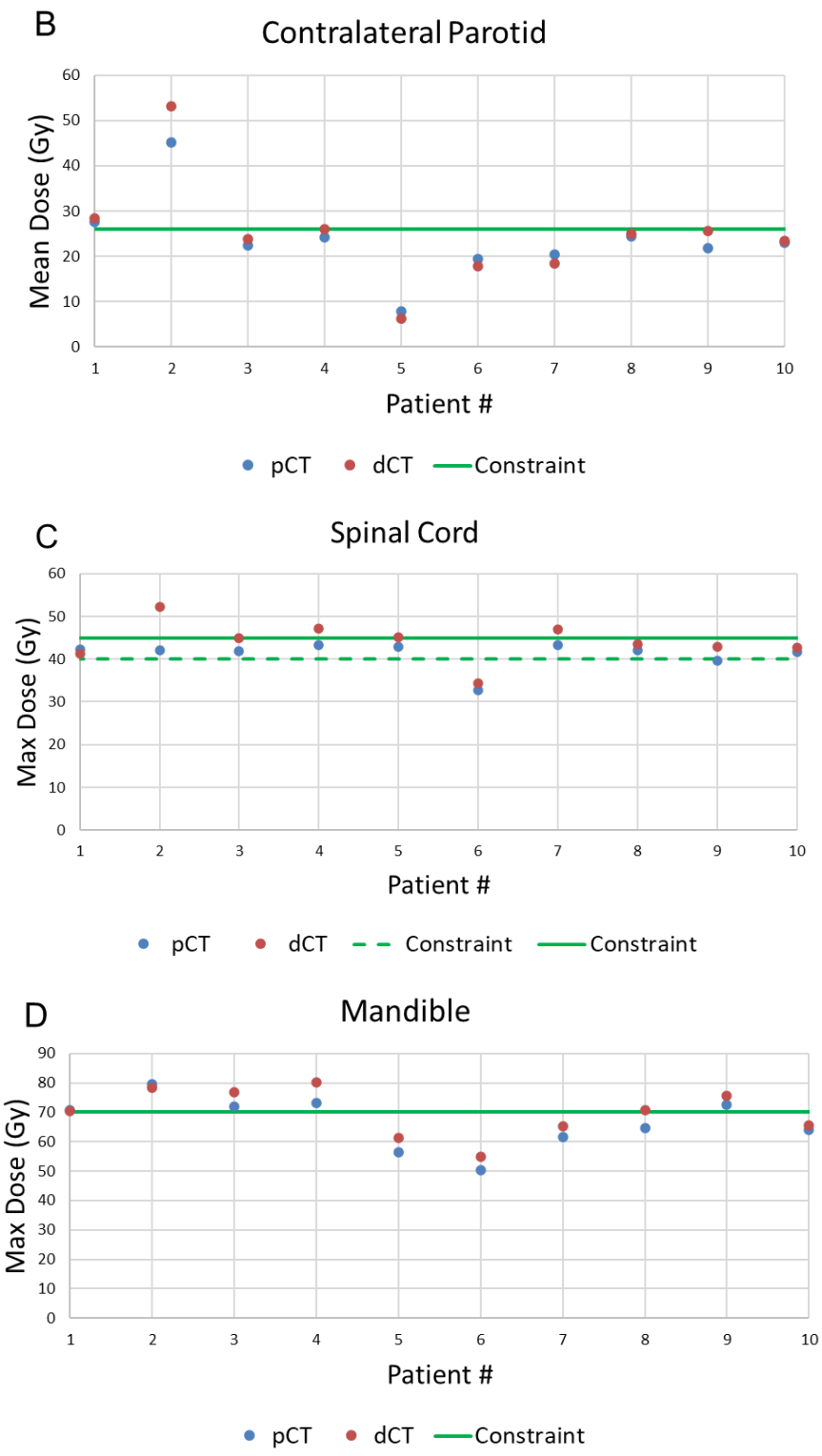


Fig. 3.7 OAR doses on pCT and dCT for each of 10 HN patients. A, B: Parotid gland dose; planning constraint of $D_{mean} < 26$ Gy indicated by the green line. C: Cord dose; planning constraint of $D_{max} < 45$ Gy (< 40 Gy if achievable) D: Mandible dose; planning constraint of $D_{max} < 70$ Gy.

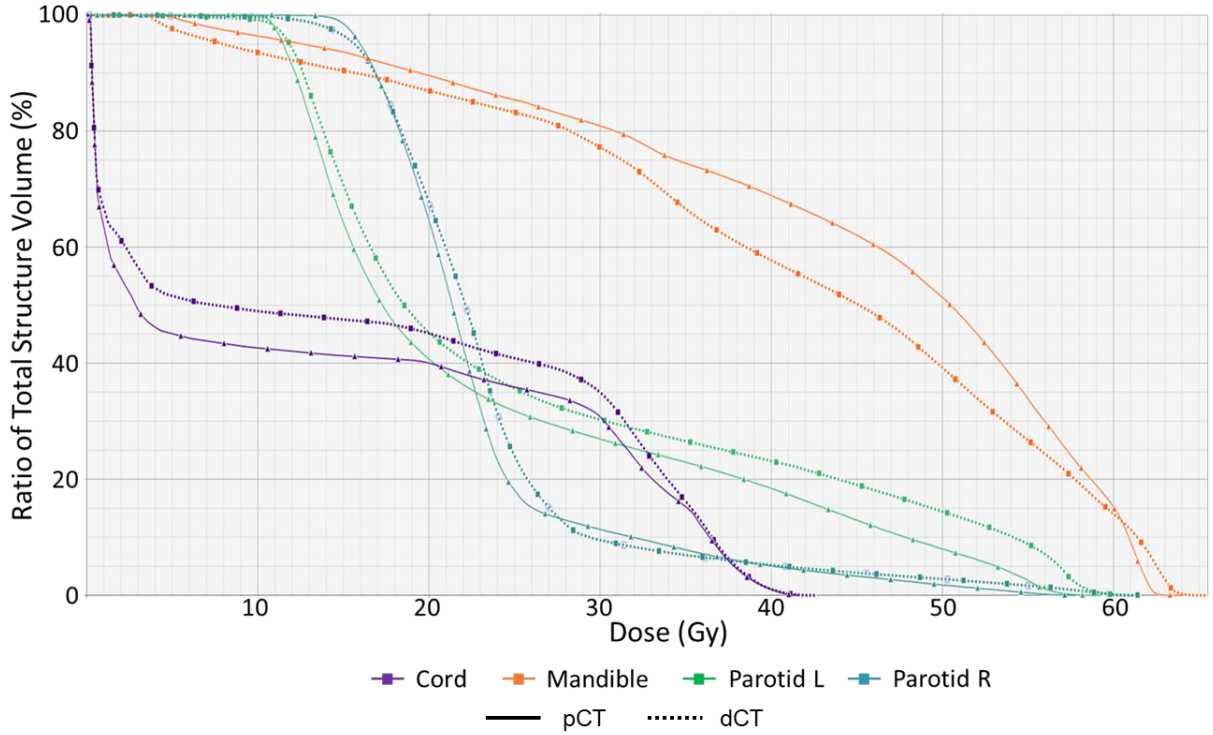


Fig. 3.8 DVH for a patient with little anatomic change. Solid lines are pCT, dashed is dCT.

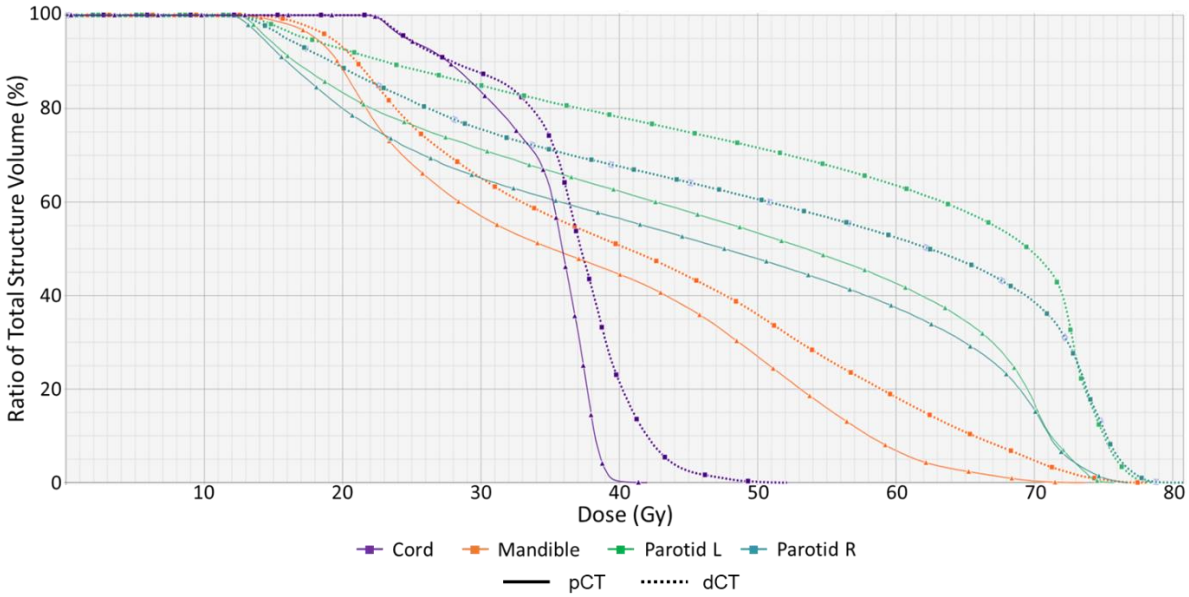


Fig. 3.9 DVH for a patient with significant anatomic change. Solid lines are pCT, dashed is dCT.

Additionally, D_{max} was calculated for the pCT and three dCTs from three time points during the treatment course for a larger cohort of 47 patients. A slight trend of increase in D_{max} as treatment progressed was observed (Fig. 3.10).

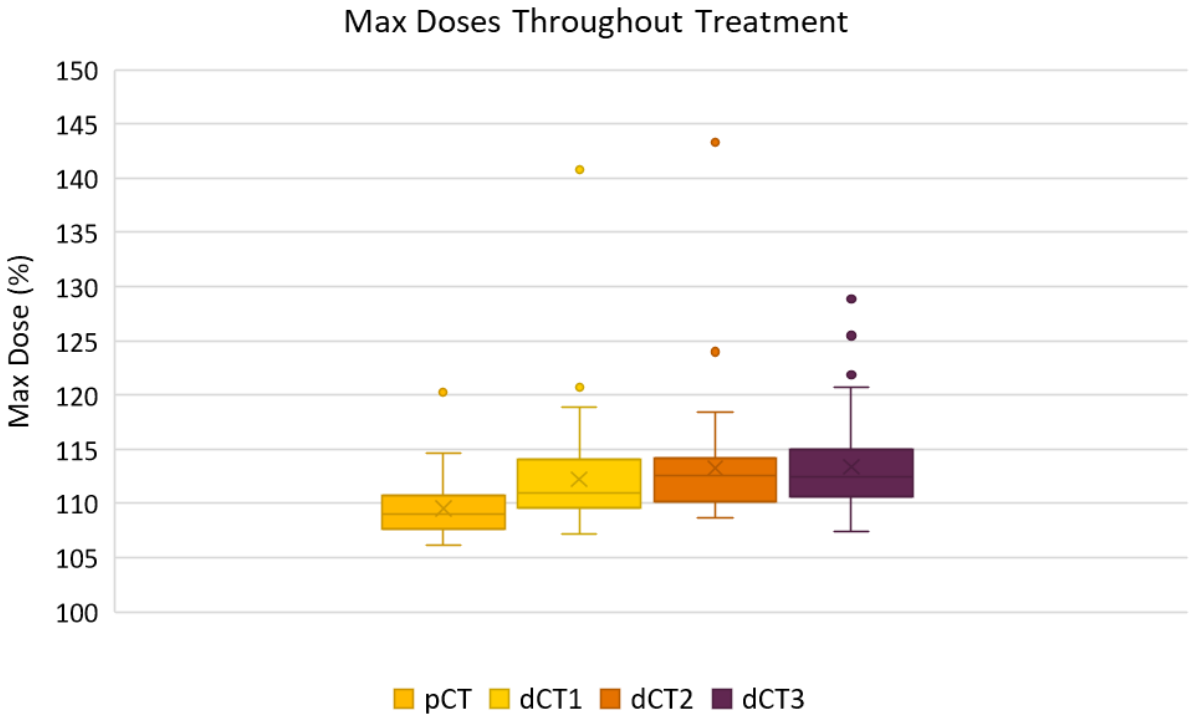


Fig. 3.10 D_{max} relative to the prescription dose for a cohort of 47 patients, showing a slight increase over the treatment course.

3.5 Discussion

The developed method for deformable registration of CT to CBCT solved several common issues encountered for this task, including limited FOV and inaccurate Hounsfield Unit values for dose calculation. The method was developed using fifteen HN patients who exhibited anatomic changes visible on CBCT and involved expanding the CBCT array to allow the resulting dCT to have a larger FOV. The registration method was validated using ten additional HN patients by assessing the DSC, JI, and HD between contours drawn on the CBCT and those deformed from the pCT.

The results for all contours are comparable to previous publications. Zukauskaite et al evaluated DIR performance for registration of planning CT and recurrence CT images for HN patients[78]. Their open-source algorithm achieved DSCs of 0.83, 0.79, 0.74, and 0.72 for the mandible, spinal cord, and left and right parotid, respectively. Their study involved CT-CT registration, as opposed to CT-CBCT registration, and a single observer contouring both image sets. This may explain why their spinal cord value differs somewhat from ours, as they had no interobserver variation and a larger extent of spinal cord included in their images. CT-CBCT DIR was performed by Veiga for the purpose of dose of the day calculation[69]. Using an open-source algorithm, they achieved a DSC of 0.85 for bony anatomy and 0.79 for soft tissue, where bony anatomy included the C1, C4, and C7 vertebrae and soft tissue included the external body contour and the left and right sternocleidomastoid muscles. The use of vertebrae for contour comparison, though not a typical OAR structure, avoids the innate ambiguity of delineating the spinal cord. Additionally, the soft tissue structures used in their study are more easily visualized on CBCT than parotids, as well as being larger in volume, which makes higher DSCs more easily achievable. Jiang et al performed a HN CT-to-CBCT study to compare two DIR methods which achieved HD ranging from 2.00 to 5.19 mm over soft tissue and bony anatomy using an open-source algorithm[66].

A study investigating interobserver variability on HN CT found average DSC values of 0.90, 0.78, 0.82, and 0.83 for mandible, spinal cord, and left and right parotid, respectively, with a large majority of the observers following the same guidelines for all contours apart from the spinal cord. This indicates that our results are comparable to those of previous publications,

despite limitations of our study, including interobserver variability and difficulty delineating soft tissue structures on CBCT.

During validation, a limitation of our algorithm was identified, as the algorithm was unable to generate acceptable registrations when presented with extreme image artifacts caused by high density implants. Seven of the original ten patients selected had dental implants, resulting in streaking and shadow artifacts on both the pCT and CBCT (Fig. 3.11). For four of those patients, the artifacts severely affected the deformable registration, with significant expansion of high intensity regions around the artifact. An attempt to mask out the artifact so the registration would ignore it was unsuccessful (Fig. 3.11). This may be due to the nature of the registration algorithm. The mask is applied to the reference image, but during transformation, data from within the related area on the secondary image (pCT) will still be visible and usable by the algorithm, and the pCT also exhibits artifacts. It may be possible to improve this issue using a metal artifact reduction reconstruction algorithm on the pCT in conjunction with masking out the CBCT artifact during registration. Since those four registrations were unsuccessful, four additional patients without artifacts were selected to replace them for the quantitative validation.

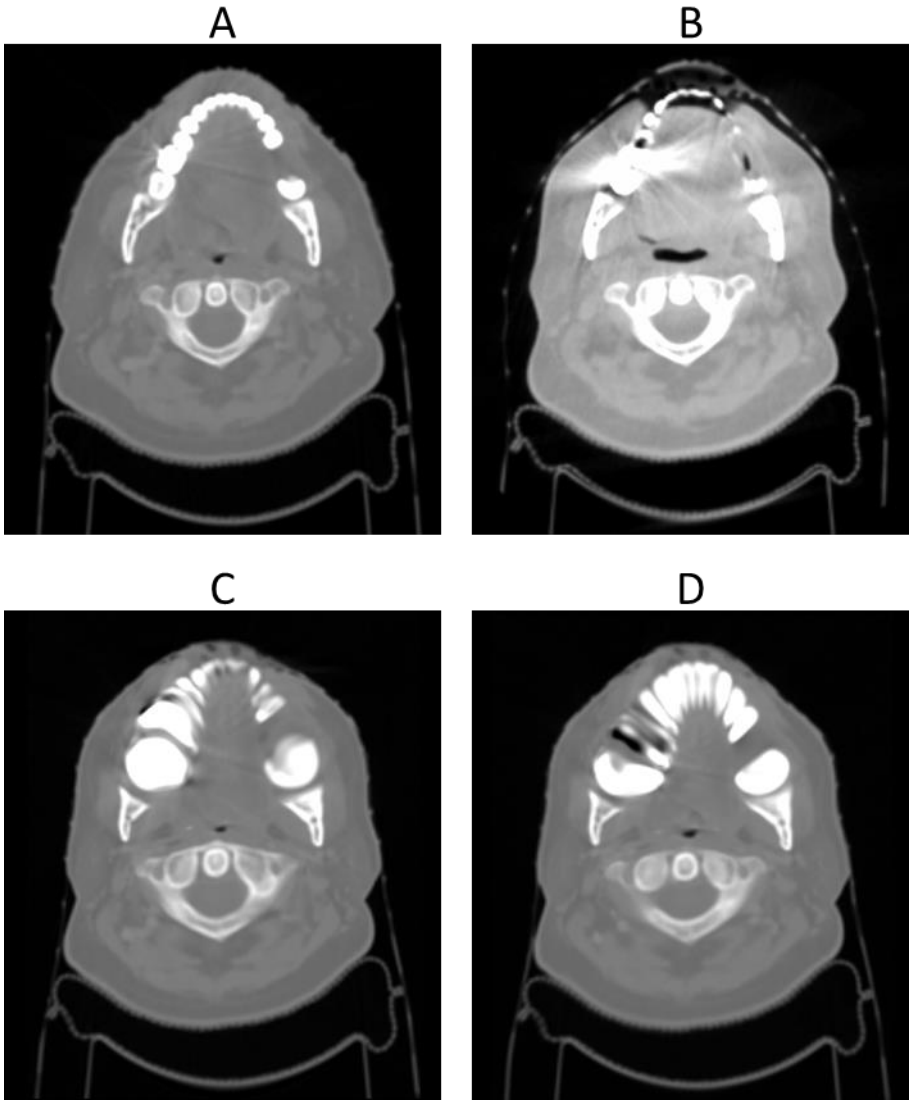


Fig. 3.11 A patient whose artifact caused severe deformation errors. A: pCT, B: CBCT, C: dCT, D: dCT with mask used to ignore artifact region.

A more minor CBCT artifact which affected the dCT quality was also detected. One patient's CBCT exhibited significant shadows at the posterior and inferior extreme of the scan, which resulted in an indentation at that point in the back on the dCT (Fig. 3.12). The registration mask could be altered for similar cases to not include the shadowed area. However, that indentation is not expected to be significant enough to affect dose calculation negatively.

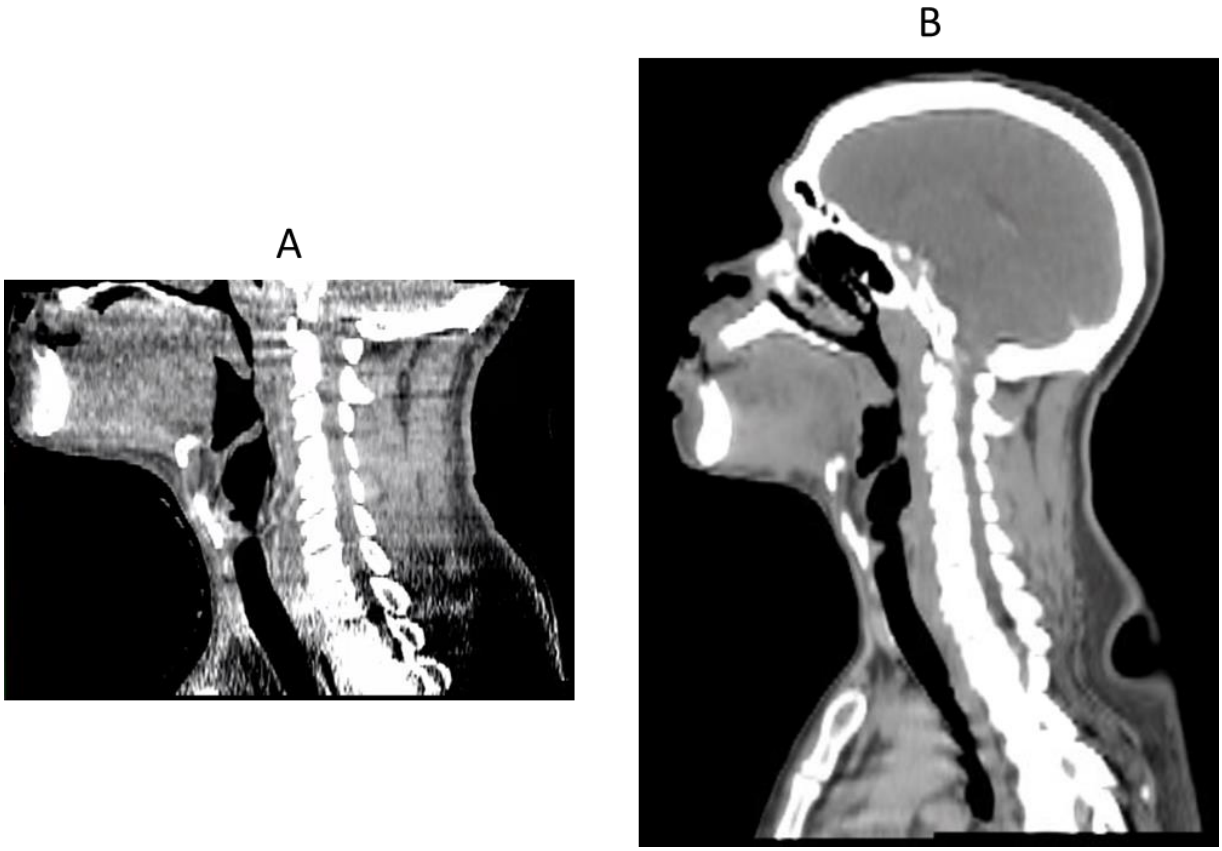


Fig. 3.12 A patient with shadowing at the inferior edge of the CBCT (A), causing an indent in the corresponding location in the dCT (B).

Due to the sensitivity of the registration method to artifacts, it did not perform as well on an older OBI system with inherently poorer imaging quality. However, further work could focus on making the method more robust to artifacts by testing different reconstruction algorithms and masking methods.

3.6 Conclusion

A CT-CBCT DIR method was developed for the purpose of performing dose of the day calculations. Using elastix, an open-source software package for image registration, the method was able to generate dCTs reflecting the anatomy of the CBCT, with the HU and FOV of the pCT. The method was validated on 10 HN patients, selected for anatomical change visible on CBCT,

using contour evaluation methods. The developed DIR method was found to be reasonably accurate for both bony anatomy and soft tissue. Since the external and bony anatomy are well-matched in the dCT, it may be used for dose calculation, which is not very sensitive to small differences in tissue density.

Chapter 4 Development of Neural Network

This chapter summarizes the work performed to achieve specific aim 2 as outlined in Chapter 1.5.

4.1 Machine Learning and Neural Networks

The field of machine learning is concerned with developing computational programs which can "learn" in a similar way to humans, through exposure to repeated examples, in order to improve performance in a specified task. This is in contrast to "hard-coding" programs to perform a specific task. For example, a classification program could either be implemented by making a database filled with examples of an object, or by feeding a machine learning algorithm labeled examples, through which it can determine the hallmarks of the object itself. A benefit of machine learning is that a computer program may find important attributes through this process that the average human observer may miss.

There are many different approaches to machine learning, including but not limited to linear regression, decision trees, and deep learning. Deep learning algorithms, or deep neural networks (DNN), are comprised of several layers of artificial neurons[79]. These artificial neurons are meant to emulate human neurons. Input data passes through weighted nodes, and the outputs of the nodes are then summed and passed through an activation function to generate the outputs of the layer. One layer feeds into another layer, and so on, until the output layer is reached. Layers between the input layer and the output layer are referred to as hidden layers, as they are unseen by the user, and DNNs can have many hidden layers. These layers perform feature extraction and transformation, abstracting the data in ways that humans cannot, to find patterns and structure.

The training of a neural network may be supervised, unsupervised, or semi-supervised. In supervised learning, the input data has labels or targets, and the network will compare its prediction to the provided label. This comparison is performed via calculation of a loss function, the exact form chosen based on the type of task being performed. The result of the loss function is then propagated backwards through the network to update the weights of each node, using a backpropagation gradient algorithm. The most popular algorithm for this purpose is stochastic gradient descent. In unsupervised learning, there are no labels; the algorithm clusters data and discovers patterns on its own. Semi-supervised learning falls between the two. For the task of dose prediction outlined in this work, supervised learning was chosen, since there was a known target to predict.

4.1.1 Convolutional Neural Networks

One of the most popular types of deep learning algorithms in radiotherapy specifically is convolutional neural networks (CNN). CNNs have found great popularity in image analysis tasks, as their architecture takes advantage of the spatial structure of image data, particularly in medical images[80]. Rather than having each node in one layer connected to every single node of the next layer, as in a standard DNN, which is computationally expensive, CNNs employ convolutional blocks to process the data, which preserves the spatial nature of the data. Convolutional filters (also known as kernels) take the place of the nodes, where the values of the filter are the weights, and the pixel values of the image take the place of the input data. A simplified example of an image being processed by a convolutional filter is presented in Fig. 4.1. In this case, the image is a 5x5 array of values which can take either -1, 0, or 1. The filter's values have been arbitrarily assigned. The filter starts in the upper left corner of the image.

Each (x,y) pixel value is multiplied by the corresponding (x,y) pixel value in the filter, and the results are summed and then divided by the number of contributing pixels (in this case, 9). The output of this operation is entered into pixel (2,2) of the output array, referred to as a feature map. The filter then slides one column to the right, which is referred to as stride. Stride can take any value and indicates the number of columns or rows the filter shifts over between operations. Once the filter reaches the last column of the array, it moves one row down (for stride=1) and continues on the next row. A convolutional neural network consists of many convolutional filters with many weights, which are updated through backpropagation during training.

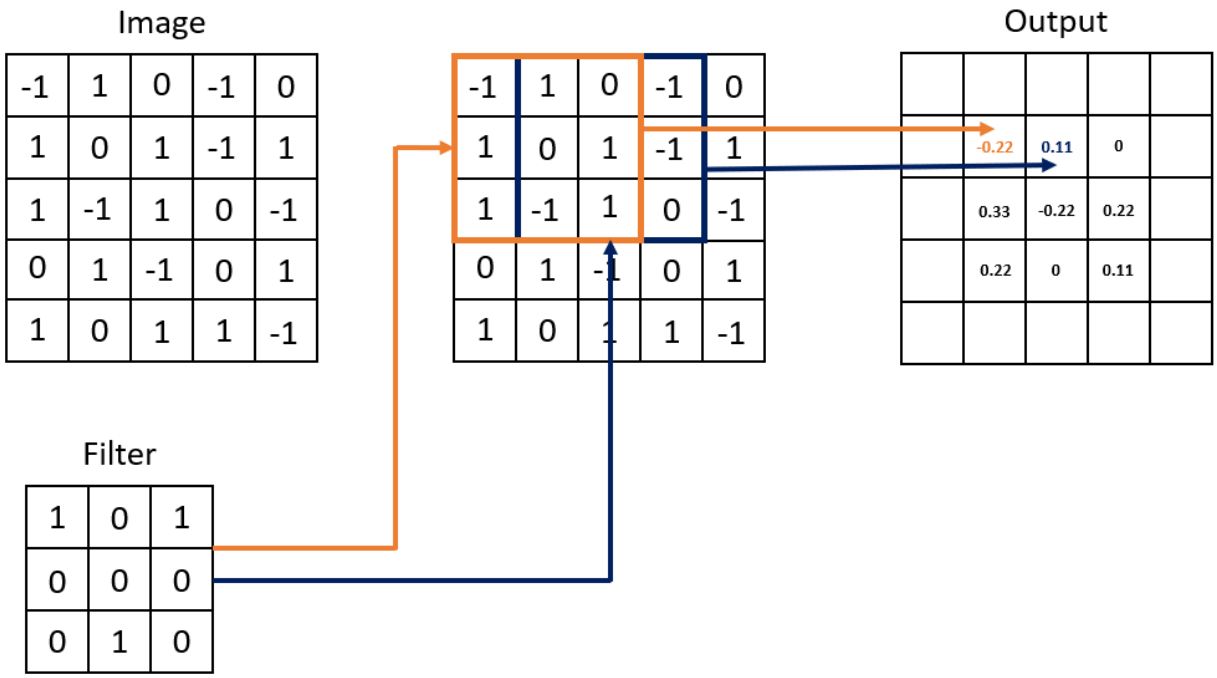


Fig. 4.1 Convolution operation on an image.

A challenge when using convolutional layers is the size of feature maps produced by the convolutional layers, which can often be quite large and computationally expensive to carry through the whole network. To reduce this load, CNNs often feature pooling layers between

convolutional layers. Max pooling is a common approach: a filter (different from the convolutional filter) sweeps across the image in a similar fashion and, for each operation, keeps only the maximum pixel value (Fig. 4.2). Depending on the size of the filter and the stride chosen, this can greatly reduce the size of the feature map passed to the next layer.

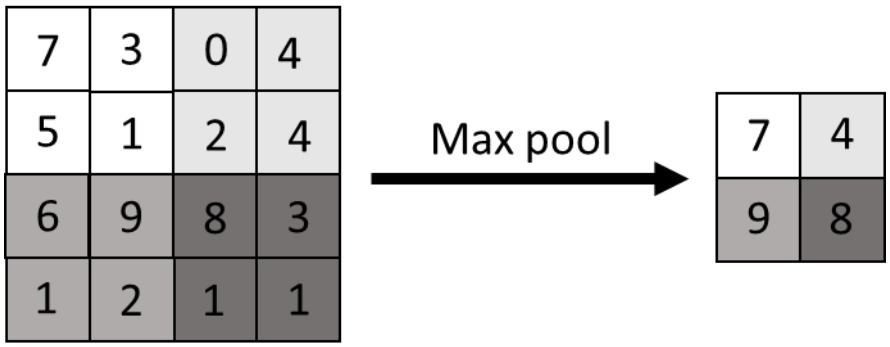


Fig. 4.2 Max pooling operation on an image.

4.1.2 U-Net

A popular version of CNN architecture is the U-Net, originally developed by Ronneberger et. al in 2015 for segmentation tasks[81]. The U-Net gets its name from the trademark shape of the network architecture when illustrated, achieved by using a series of convolutional, pooling, and rectified linear unit (ReLU) layers followed by a series of deconvolutional layers (Fig. 4.3). ReLU is the standard activation function for many neural networks, including CNNs, and the output of the ReLU is the same as the input unless the input is negative, in which case the output is zero. Deconvolutional layers are transposed convolutional layers which upsample the image. The first half of the U is a contracting path, while the second half of the U is an expanding path bringing the image back to the original resolution. The initial contracting path decreases the resolution of the image while increasing extracted features, and the subsequent expanding path increases the resolution of the image while factoring in high-resolution features from the contracting path. Skipped connections, illustrated in Fig. 4.3 as “copy and crop”

arrows, allow for the propagation of features from earlier in the network to later layers, skipping layers along the way. This architecture allows the network to learn from global and local features of image sets, and it has been shown to work well with smaller image sets. It has since been used for dose prediction as well as image segmentation in the radiotherapy field, with many variations of it published in the literature[82–84]. One such variation added several convolutional layers to the end of the U-Net to achieve better precision in prediction of three-dimensional dose maps[82]. Another variation by the same group combined characteristics of DenseNet, a network architecture valued for its efficiency in feature propagation and reusing parameters by connecting all layers directly to each other, with U-Net to create a more efficient network that would still be effective at capturing both local and global features[83]. Kearney et al used residual blocks to propagate information from earlier in the network to deeper portions of the network[84]. Each variation outperformed the original architectures in accuracy of the predicted dose maps and proved to be a potential tool for predicting 3D dose using just patient contours. Due to a limited dataset and the goal of predicting 3D dose distributions, the U-Net was chosen for this project.

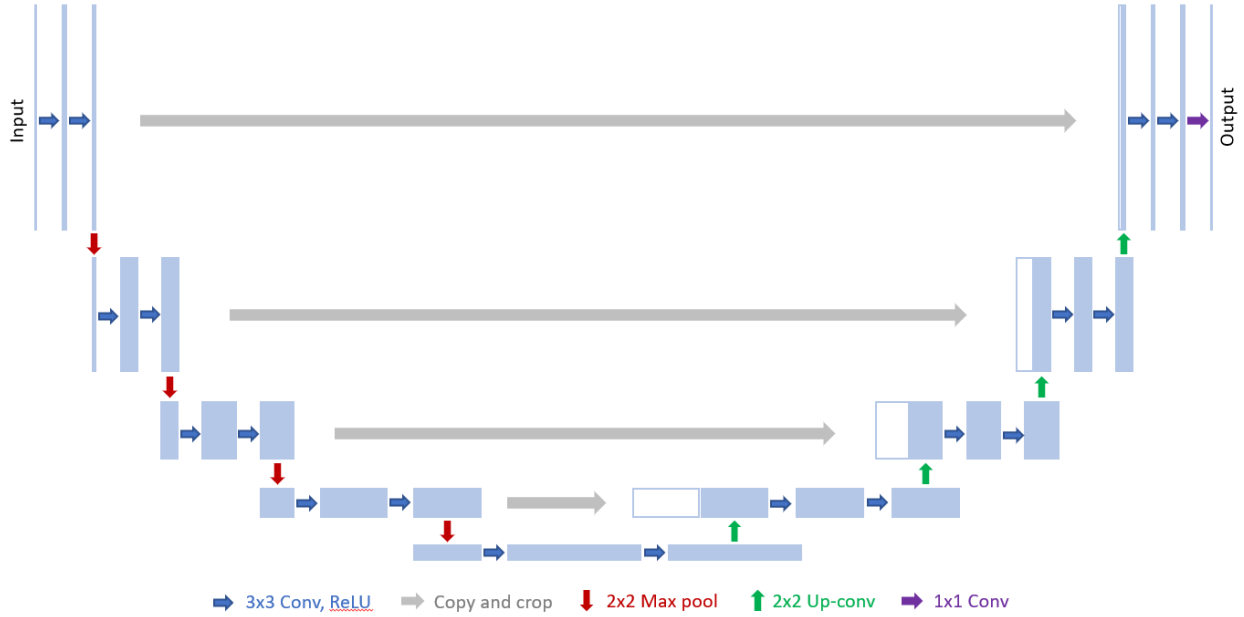


Fig. 4.3 The original U-Net architecture, with blue boxes corresponding to multi-channel feature maps and arrows denoting the operations performed at each step[81].

4.2 Development of Neural Network

4.2.1 Patient Data

Patients were retrospectively selected for this study from those treated for HNC on Varian TrueBeam linacs (Varian Medical Systems, Palo Alto, CA). The goal was to recruit 50 patients, and 47 patients were ultimately chosen over a three year period (Table 4.1). Patients with noticeable anatomic change by the end of treatment were sought out, but many of the included patients exhibited little change. Patients with severe high density artifacts which affected deformation quality were excluded. Patient selection was not limited to a specific disease origin; the majority of patients were oropharynx (OPX) cases. For those patients whose weights were recorded (40), there was an average of 16 pounds (8.5%) lost over the course of treatment.

Table 4.1 Patient Demographics for Neural Network Development		
Sex	Female	6
	Male	41
Median age		59 (range 19-86)
Median weight before treatment (lbs)		177 (range 113-295)
Median weight after treatment (lbs)		163 (range 104-265)
Disease site	Oropharynx	20
	Larynx	11
	Nasal cavity	3
	Nasopharynx	2
	Oral cavity	5
	Hypopharynx	1
	Other	5

For each patient, one planning CT (pCT) and 3 cone-beam CTs (CBCT) were selected. CBCTs were selected as close to the tenth, twentieth, and thirtieth fraction as possible to have them evenly spread over the 30-33 fraction treatment course (Table 4.2). Deformed CTs (dCT) were generated for each pCT-CBCT pair according to the method outlined in Chapter 3.

Table 4.2 Fraction numbers for each CBCT	
	Median fraction number
First CBCT	11 (range 7-16)
Second CBCT	21 (range 16-23)
Third CBCT	30 (range 24-33)

The dCTs were transferred back to the original TPS (Eclipse, Varian Medical Systems, Palo Alto, CA) and registered with the pCT (Fig. 4.4). The original plan was then copied onto each of the dCTs and the dose recalculated (Fig.4.5), using the Acuros calculation model with a calculation resolution of 2.5 mm and the heterogeneity correction enabled. Dose maps were exported in DICOM file type. The body contour for each pCT and dCT was refined and the posterior half of the structure was removed to mimic what the AlignRT surface imaging system

would see on the treatment table (Fig. 4.6). This “surface” structure was exported to the AlignRT system to generate the AlignRT surface file, which uses a 3D obj file type. The surface is composed of many triangles, and the obj file type stores this information through the coordinates of the vertices and faces of the triangles.

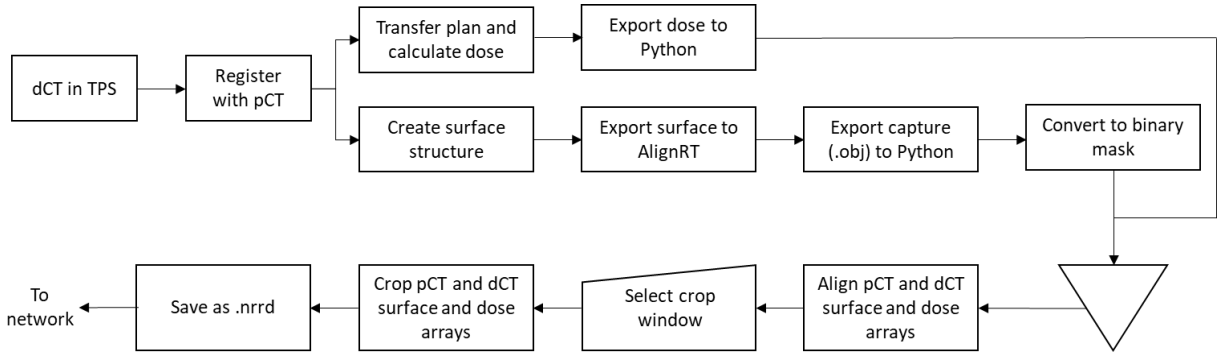


Fig. 4.4 Flowchart depicting pre-processing steps for dose and surface data generation.

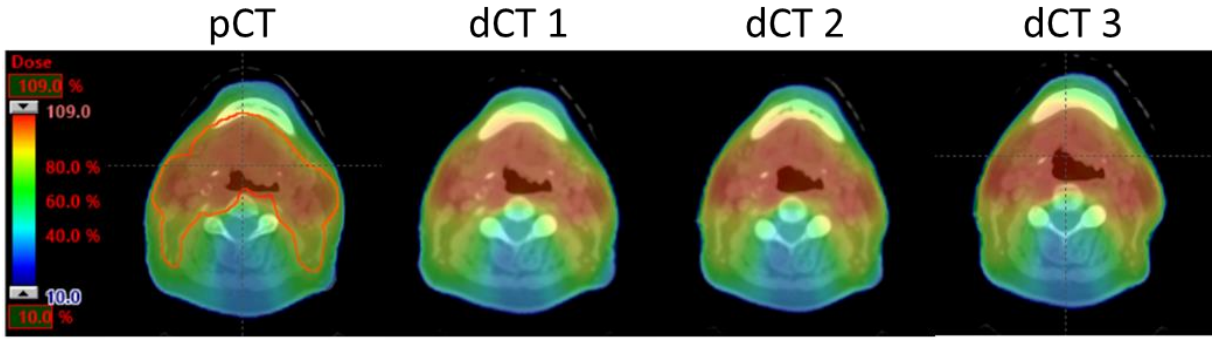


Fig. 4.5 Original treatment plan calculated on the pCT and the first, second, and third dCT. PTV is contoured in red on the pCT.

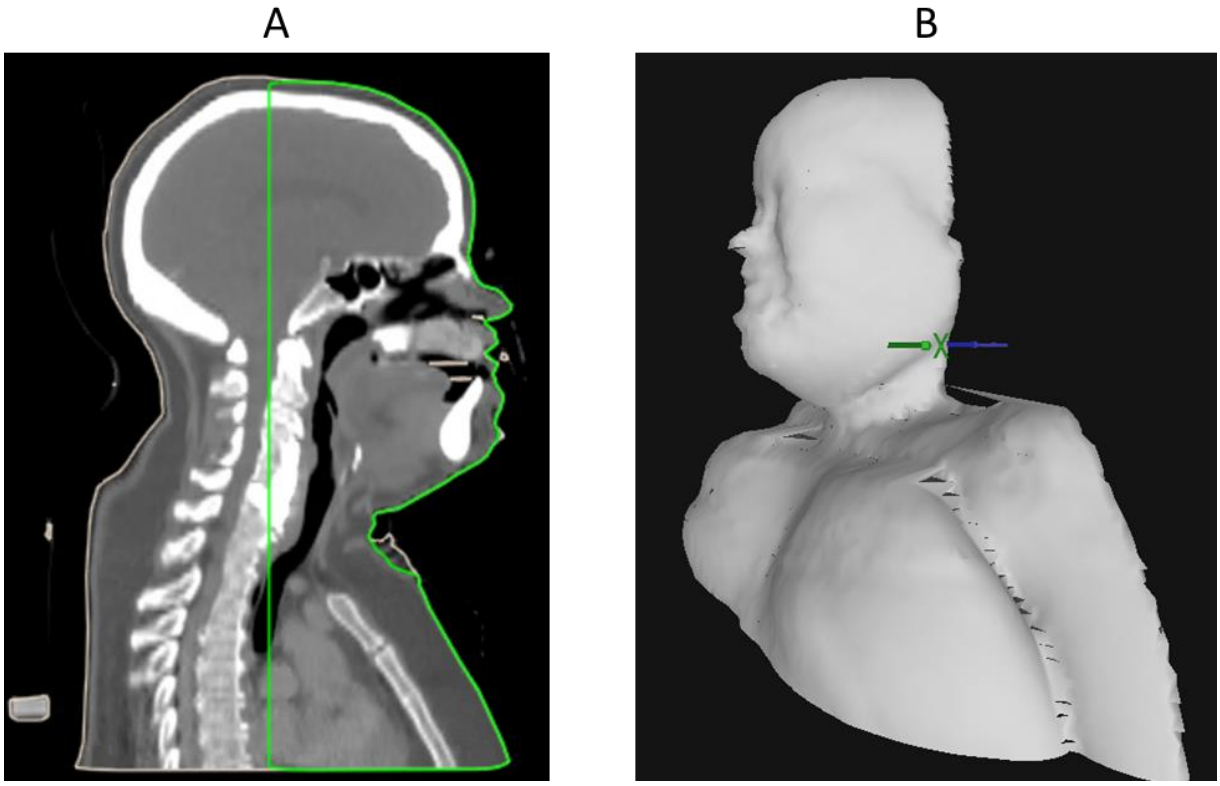


Fig. 4.6 Surface structure for an example patient. A: dCT with body contour in tan and surface contour in green. B: AlignRT visualization of the same surface structure.

To generate image sets for the network, the exported surfaces and dose maps needed to be converted to the same file type, in simple three-dimensional array format. The dCT surface-dose map pairs also needed to be shifted to align with the original pCT surface-dose map pair. A python script was written which converted the list of vertex coordinates in the surface files to a three-dimensional binary array representing the surface, then shifted all arrays so they were aligned with the original pCT. All arrays were also resampled to 3 mm/pixel isotropically; the surfaces had varying resolutions but were generally 1 mm/pixel in all directions. Dose maps were generally 1.17 x 1.17 x 3 mm^3. Arrays were then cropped to 64x64x64 pixels to reduce computational expense while running the network. Since the field of view for each patient's scan was not consistent both in terms of size and relative position of

patient anatomy, the crop window was manually chosen on the pCT dose map to extend approximately from the bottom of the neck to the tip of the nose in the superior-inferior direction and from the front of the face to the back of the head in the anterior-posterior direction. The crop extent in the lateral direction was centered on the array as the images were found to be centered laterally within the array. It was not always possible to include the entire anterior-posterior extent of the head in the crop window; in this case, the inclusion of the face was favored over the back of the head, as cheeks are known to exhibit some anatomic change (Fig. 4.7).

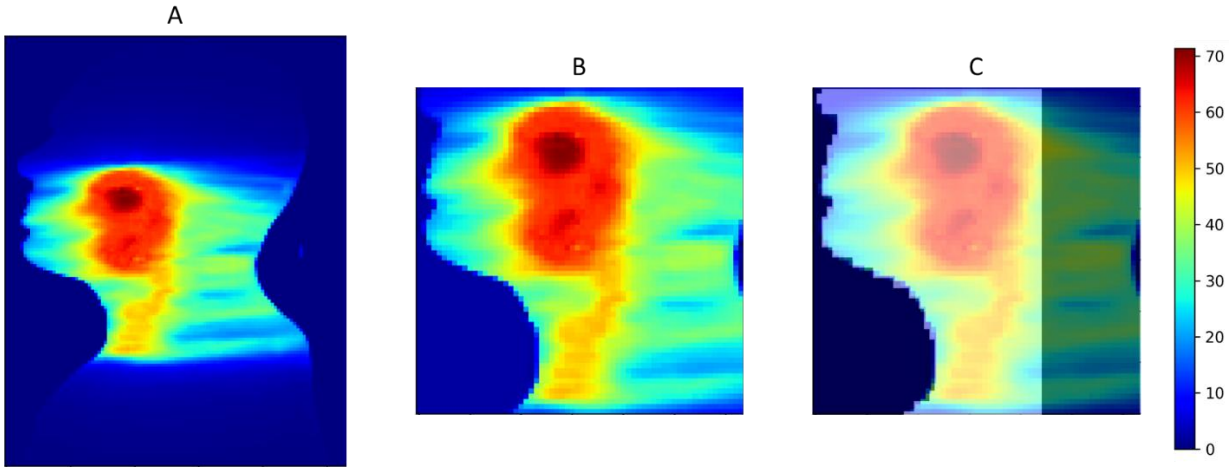


Fig. 4.7 Example of cropping of dose array. A: Original dose distribution. B: Resampled and cropped dose array. C: Surface overlay on dose array.

4.2.2 Network Architecture

The input to the network was a stack of volumes including the original pCT dose distribution, the original pCT surface, and the "surface of the day", the surface from a given dCT. The target was the "dose of the day", the dose from that same dCT calculated within the treatment planning system. Since each of the 47 patients in the study had 3 dCTs, this resulted in 141 input-target image sets. The original U-Net architecture was modified for three-dimensional image sets and the size and shape of the input and target in this study. While

training the network, the bottleneck at the base of the U was also extended by two convolutional layers to improve performance (Fig. 4.8). The final convolutional layer originally used a 3x3x3 filter but was updated to a 1x1x1 filter after referencing the original U-Net and observing a slight improvement in performance with the 1x1x1 filter. The network was implemented using Keras, a high-level API for machine learning implementation, in Google Colab, a cloud-based Jupyter notebook with options for graphics processing unit (GPU) and tensor processing unit (TPU) allocation for faster computing. TPUs are hardware specifically developed for ML workload acceleration.

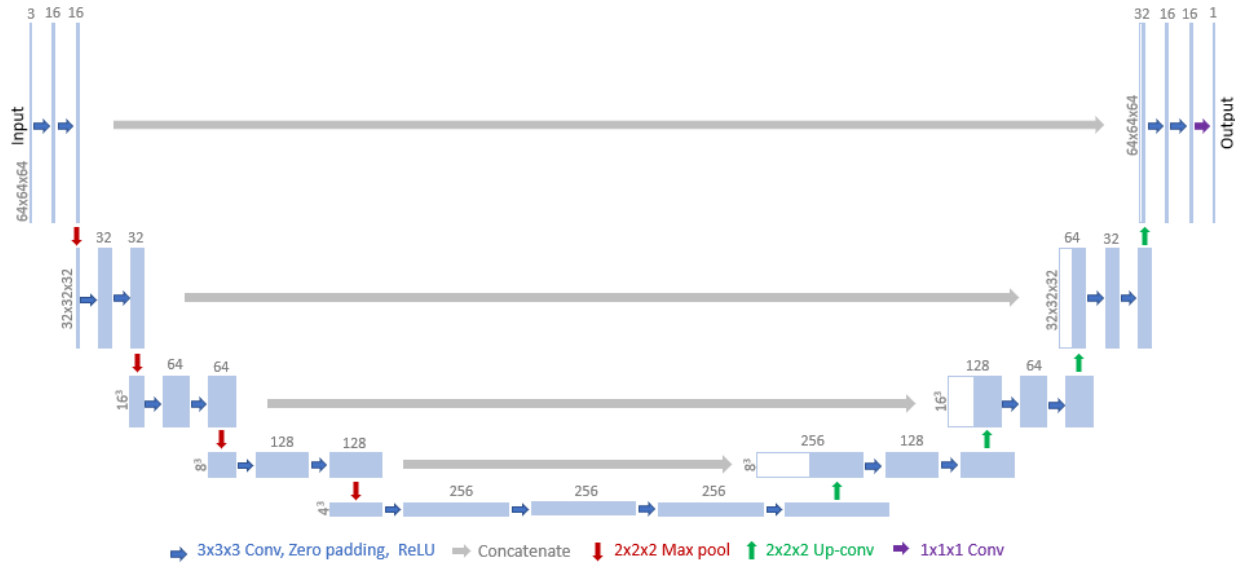


Fig. 4.8 U-Net architecture used for this study, with sizes for various layers and arrows indicating different types of operations.

The loss function for training was mean squared error (MSE) between the predicted and target dose distribution,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y'_i)^2 \qquad (4.1)$$

which is the sum of squared differences between each pixel value in the target and the corresponding pixel value in the predicted dose distributions. This was chosen since the dose prediction task is a regression problem and after studying other dose prediction neural networks[83–87]. Root mean squared error (RMSE) and mean absolute error (MAE) were also used as metrics to evaluate model performance during and after training:

$$RMSE = \sqrt{MSE} \tag{4.2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - y'_i|^2 \tag{4.3}$$

RMSE and MAE are useful metrics since they give error in the same units as the variable of interest, in this case dose. Both RMSE and MSE are more strongly affected by large outliers than MAE, so one poorly performing prediction out of many well-performing predictions is likely to result in a high MSE and RMSE but a lower MAE. Training with MAE as the loss metric was performed, and while it resulted in lower MAE for training and validation sets, the MSE and RMSE were higher. Given that the loss function is calculated over the whole array, including a large volume of air, where errors are frequently small, MSE and RMSE were favored over MAE, since MSE and RMSE would be less forgiving to larger deviations. Therefore, MSE was ultimately kept as the loss function for training.

The Adaptive Moment Estimation (Adam) gradient descent algorithm was selected as the optimizer after consulting literature[84,86–89]. Rather than having just one learning rate for updating all weights, Adam assigns independent learning rates to each network weight, which is beneficial for data where some features occur with very different frequencies than others and is preferred for computer vision problems. The default values for the parameters of Adam were

found to be suitable after training with other values and finding poor performance and a lack of convergence.

Typically, neural networks include normalization to ensure all input data is on the same scale and is therefore equally weighted. However, normalization becomes difficult when using dose maps as both an input and the prediction target. Initially, dose distributions were input to the network “as is”, in units of Gray (Gy). However, after seeing suboptimal performance, and RMSE and MAE for both validation and training sets on the order of tens of Gy, the magnitude of the input and target dose distributions was reduced. It was theorized that the network was not taking into account the surface information, which was presented in 0s and 1s, since dose values ranged between 0 and 70 Gy or more. The network was trained with the dose maps divided by 10 and by 100, and the largest improvement was seen with dose divided by 100 (Fig. 4.9). This fits with expectation, as the dose divided by 100 was on the same scale as the surface data, with all values falling between 0 and 1. Further reducing the magnitude of the dose did not improve performance.

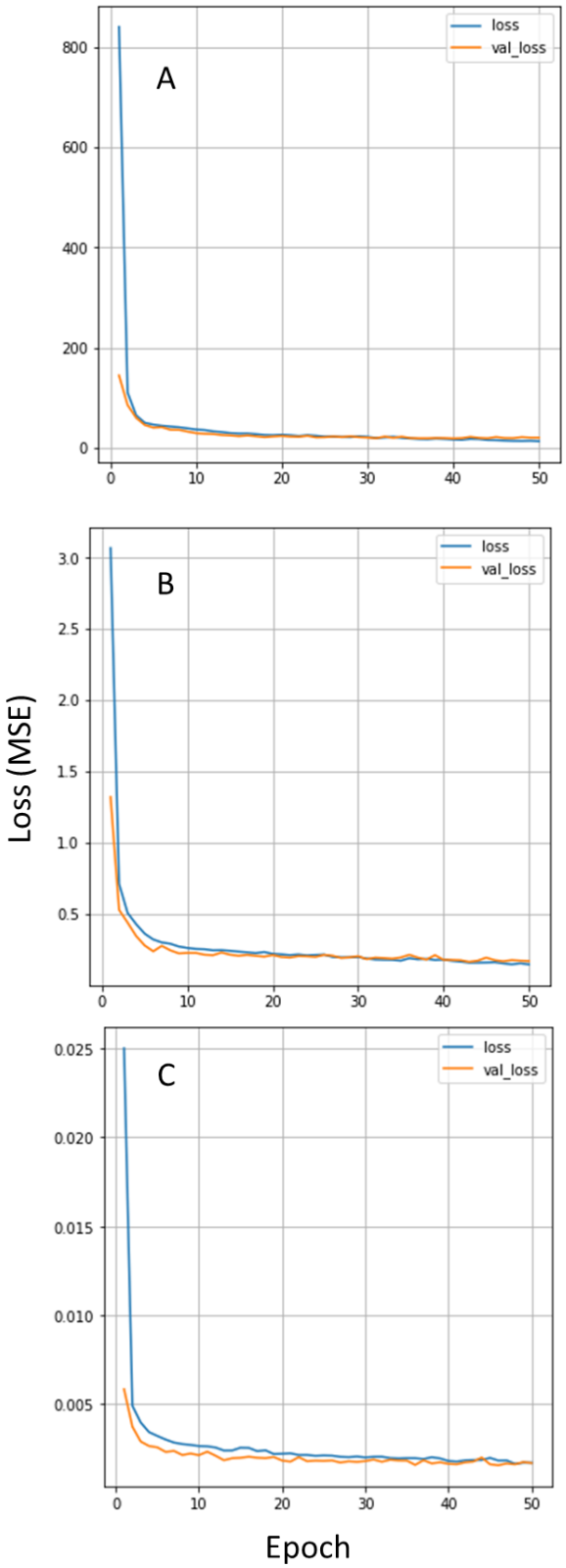


Fig. 4.9 Loss plots for training data with (A) no division of dose arrays, (B) division by 10, and (C) division by 100.

4.2.3 Network Training

Six datasets were set aside for testing, from six different patients. Each of these datasets included the pCT dose distribution, the pCT surface, the third dCT surface, and the third dCT dose distribution. Since the other datasets from these patients included additional input information and a different target, it was determined that it was permissible to have the other datasets from those patients in the training or validation sets. The six test patients were randomly selected from the ten patients used for DIR validation, with the goal of using the contours from those patients for network evaluation. This left 135 image sets for training. In the initial training and development phase, 119 sets were allotted to training and 16 to validation to ensure no overfitting. This is approximately a 90/10 split between training and validation and was manually executed by use of a random number generator. While 60/40 and 80/20 splits are often seen in machine learning development, a 90/10 split was preferred for this project, given the small number of datasets. Image sets were randomly placed into either training or validation, and the image sets were shuffled for training. After testing with various batch sizes, a batch size of 7 was selected for the best performance. Fifty epochs were used for training, but the model weights were saved when the validation loss was lowest, which may be before reaching fifty epochs. It was found that training for more epochs resulted in overfitting to the training data (Fig. 4.10).

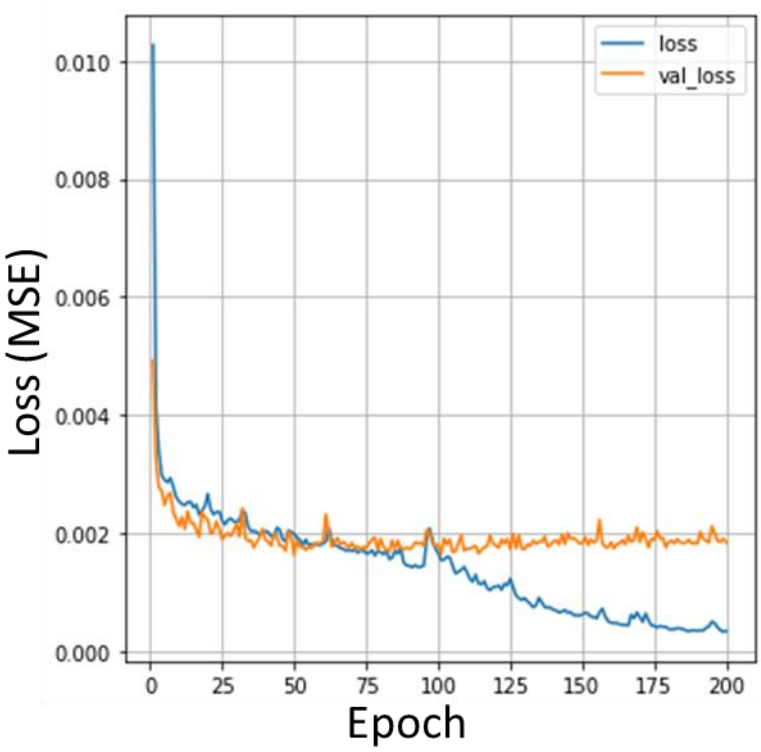


Fig. 4.10 Loss plot for 200 epochs using training data with dose values divided by 100.

K-fold cross-validation was also employed to assess the model, given the small number of samples. This is done by splitting the dataset k ways. For example, if there are 100 samples and 5-fold cross-validation is used, there will be 5 splits, each with 80 samples in the training set and 20 samples in the validation set. Each sample then gets a chance to be in the training and the validation sets. The model is then trained on each of the 5 splits, with new weights for each split. Performance metrics are then averaged over all the splits. Five and ten are common values for k, due to the computational expense of doing many splits; ten has been found to result in generally low bias and small variance in results. Nine was chosen for k for this dataset, as it divided evenly into the 135 image sets, meaning each split would have 120 training samples and 15 validation samples. A batch size of 5 was used for the cross-validation, and

training ran for 50 epochs. The best weights for each split were saved and evaluated on the test set as well.

After choosing the best model using k-fold cross-validation, the final model was trained using all 135 image sets, with a batch size of 5 and 45 epochs. The weights which achieved the lowest loss were saved to use for test set prediction. Training took 9 minutes on the standard GPU runtime available with Colab. Prediction of the test set of 6 samples took 1 second.

4.3 Network Evaluation

The predicted dose maps were multiplied by 100 to return to units of Gy for evaluation. The true and predicted max and mean doses were calculated, as well as the RMSE and MAE for each pair. Additionally, dose volumes were generated. In the absence of target contours, these dose volumes were generated for the entire body. The true and predicted V100, V95, V90, V70, V50, V20, and V10 were created. For each of these dose volumes, the DSC between the true and predicted was calculated as a way to evaluate the similarity of the dose distributions. Dose volume histograms (DVHs) for dose volumes were also generated. The volumes of the true and predicted V100 and V90 were also calculated.

Table 4.3 shows the RMSE, MAE, difference in D_{max} as a percentage of prescription dose, and difference in D_{mean} in Gy; a negative value indicates the prediction was lower than the true value. Overall, the predictions were “colder” than the true dose distributions. Of note is the large D_{max} difference shown in Pt 1; the reason for this is unknown, although the true D_{max} for Pt 1 was much higher than that of other patients.

Table 4.3 Network performance on test set						
	RMSE	MAE	True D_{max} (Gy)	Pred D_{max} (Gy)	D_{max} diff (%)	D_{mean} diff (Gy)
Pt 1	5.18	2.24	84.42	74.24	-14.6	-2.95
Pt 2	3.21	1.41	69.16	67.06	-3.3	-3.14
Pt 3	7.00	4.13	73.45	70.77	-4.1	0.20
Pt 4	4.04	1.94	77.18	76.68	-0.72	-0.89
Pt 5	3.06	1.57	79.38	75.41	-5.7	-0.04
Pt 6	3.03	1.63	77.62	75.69	-2.8	-1.15

When looking at the dose volumes, Pt. 3 predicts much larger dose volumes than the true volume, whereas the other patients exhibit the opposite phenomenon (Table 4.4). Pt 3 is also the only patient where the predicted D_{max} did not fall within the true V100. The reason for this is unclear; however, Pt 3 did have a large dental implant resulting in artifacts on his scans, which may affect dose calculation and the behavior the model expected.

Table 4.4 Differences in true and predicted isodose volumes as a percentage of true volume		
	ΔV100(%)	ΔV90(%)
Pt 1	-31.23	-9.55
Pt 2	-65.44	-19.72
Pt 3	563.53	26.98
Pt 4	-24.10	-7.27
Pt 5	-34.32	-16.03
Pt 6	-45.57	-13.41

DSCs for isodose volumes were lowest at V100 and rapidly increased (Fig. 4.11). DVHs for dose volumes looked similar overall, albeit shifted towards lower doses from the true DVHs, as the predictions were colder overall (Fig. 4.12). Fig 4.13 shows the true and predicted dose distributions for Pt 4, the absolute difference map in Gy, and the true and predicted V100 and V90 overlaid on the true and predicted dose distributions, as an example of the average patient's predicted dose.

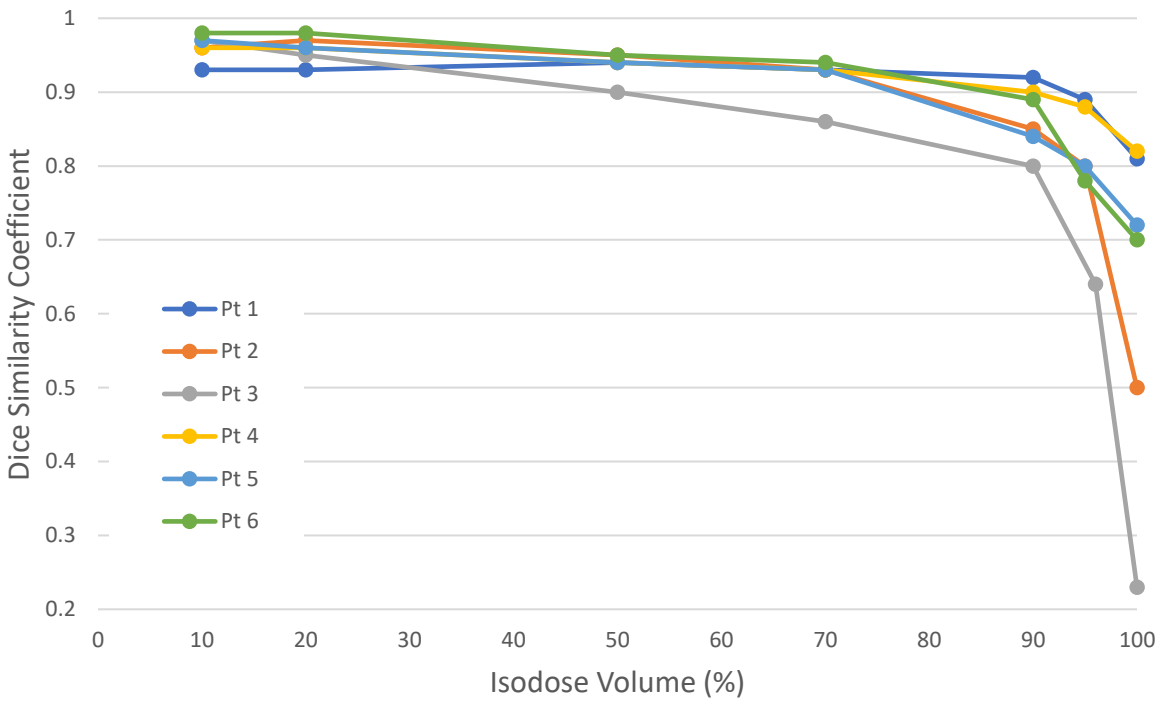


Fig. 4.11 DSC for true versus predicted isodose volumes

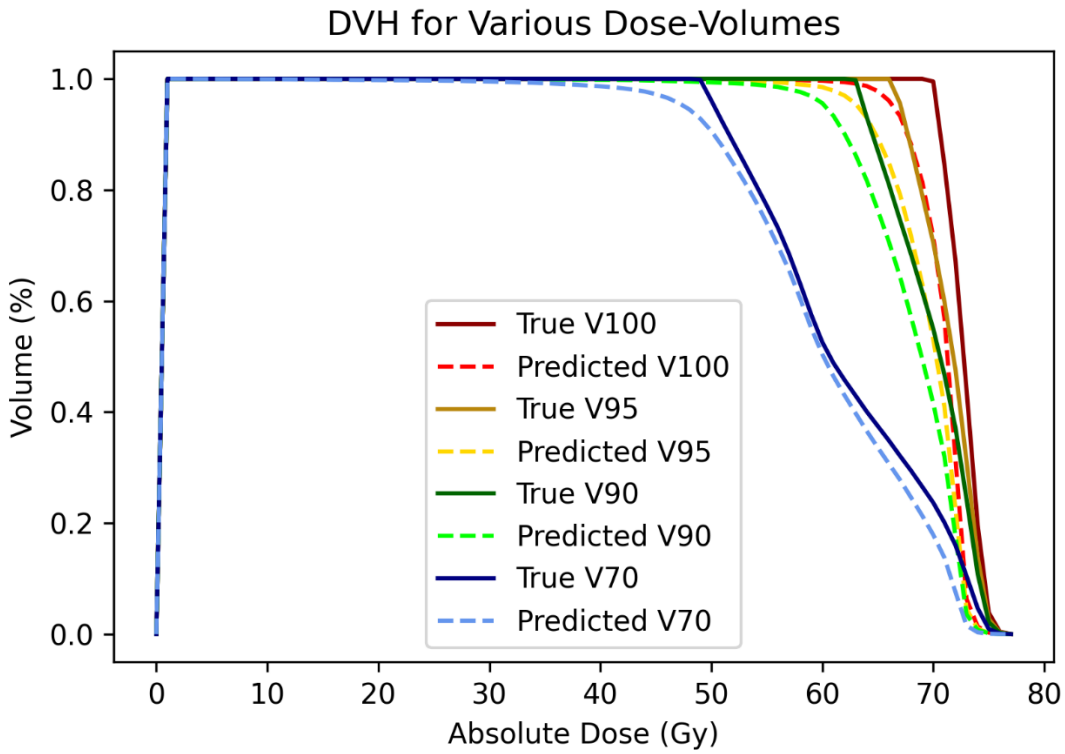


Fig. 4.12 Dose-volume histogram for true and predicted isodose volumes for Pt. 4.

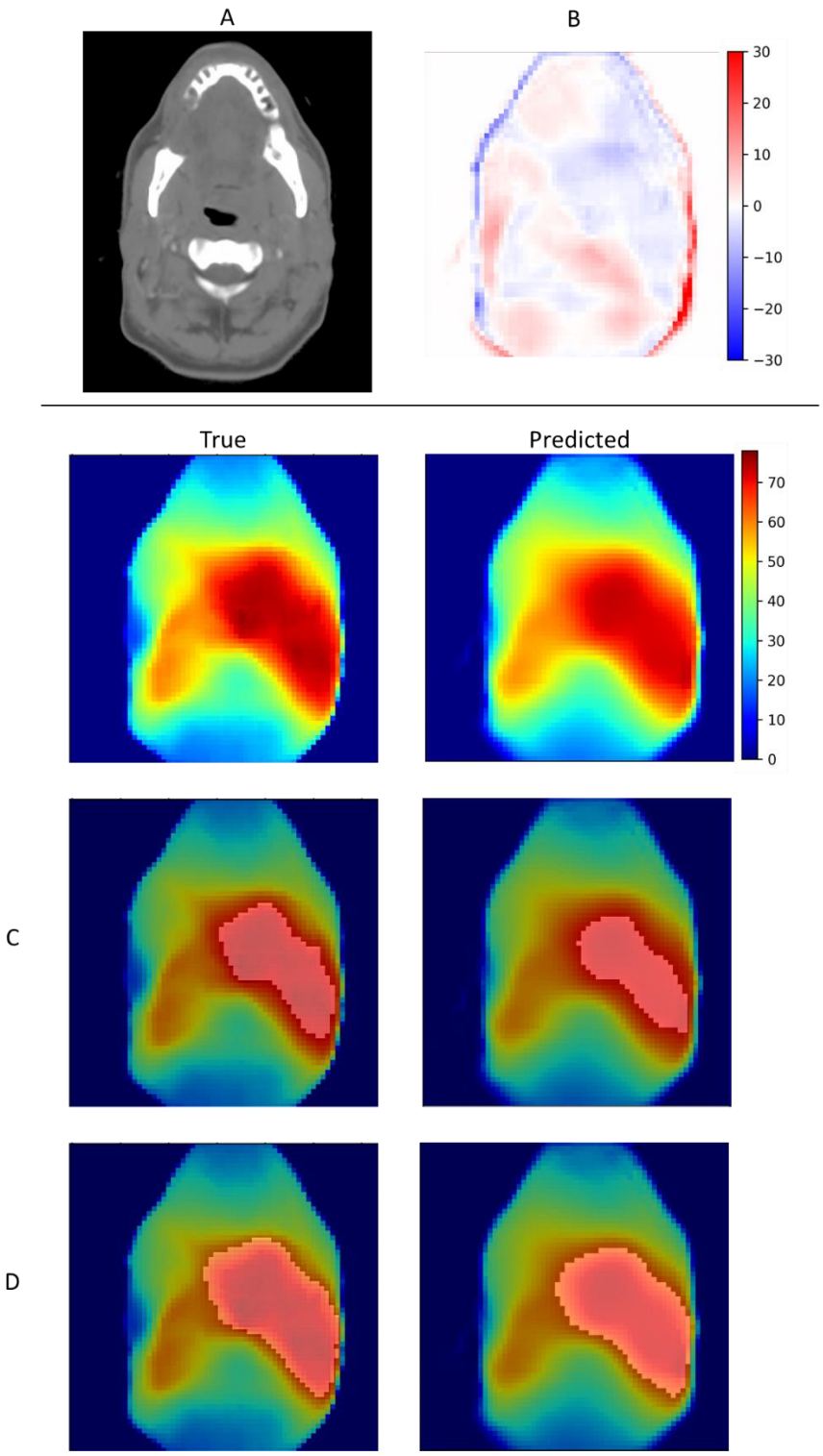


Fig. 4.13 Example predictions for Pt. 4. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

4.4 Discussion

A proof-of-concept neural network was developed which takes original dose distribution, original surface, and treatment surface as input and delivers the predicted treatment dose distribution as output. The model predictions prove that a neural network can predict 3D dose distributions based on inputs of two surfaces and one dose distribution, which has not been previously reported in the literature. Additionally, the prediction is carried out very quickly on a standard GPU runtime. The current predictions mimic real dose distributions quite closely. Interestingly, the edges are sharp where the network has surface data indicating the edges of the body; the posterior surface of the patient, where the network does not know the edge of the body, is fuzzy in the predictions.

However, the accuracy of the model predictions is inconsistent and indicates need for further improvement. It is of interest that the network generally predicts colder than the true values; investigating the cause of this is a priority. This could indicate that the network is starting with the original dose distribution and essentially weighting individual pixels to get to the treatment day dose distribution, since the treatment day dose distributions are observed to have higher max doses than the original. The network may not be altering the distribution enough due to limitations of the training. Two avenues for improvement have been identified: further patient accrual and a custom loss function. More data is needed to further train the network, and a custom loss function may improve training. The current loss function calculates over the entire volume, which includes a large volume of air, an area that is easy to predict and not of concern and therefore drives the loss down artificially. Finding a way to calculate the loss solely on the body, or part of the body, could more accurately represent model performance

while training and result in better learning. This could potentially be done by writing a custom MSE function with a threshold to ignore low dose pixels. Additionally, the model architecture could be modified to add different types of layers or additional inputs. For instance, a principal component analysis operation could be carried out on surface changes before input to the U-Net, and features extracted from that could be added as inputs to the U-Net. Another common method for improving dose prediction models which could be explored is adding beam geometry as an input.

The poorest performing patients, Pt 1 and Pt 3, may be outliers for this dataset, which may explain the poor prediction. Pt 1 had a prescription of 69.96 Gy, but the true max on the deformed CT was 84.42 Gy, which is higher than most of the patients included in this study. Pt 3 had a major dental implant which may affect the behavior of dose accumulation (Fig. 4.14). While other patients with implants were included in the study, none were to the extent of Pt 3. After patient recruitment had been ongoing for some time, a decision to exclude patients with artifacts from dental implants was made due to the unreliability of the DIR method in handling artifacts. However, patients which already had produced good registrations even with artifacts were kept for training. There may simply not be enough patients with dental implants in the network for it to predict those patients accurately. These results may indicate that different models need to be trained for different dose levels or patients with artifacts. Predictions may also improve by training separate models for different disease sites. Stratifying the data this way would require a larger amount of data in order to train effectively.

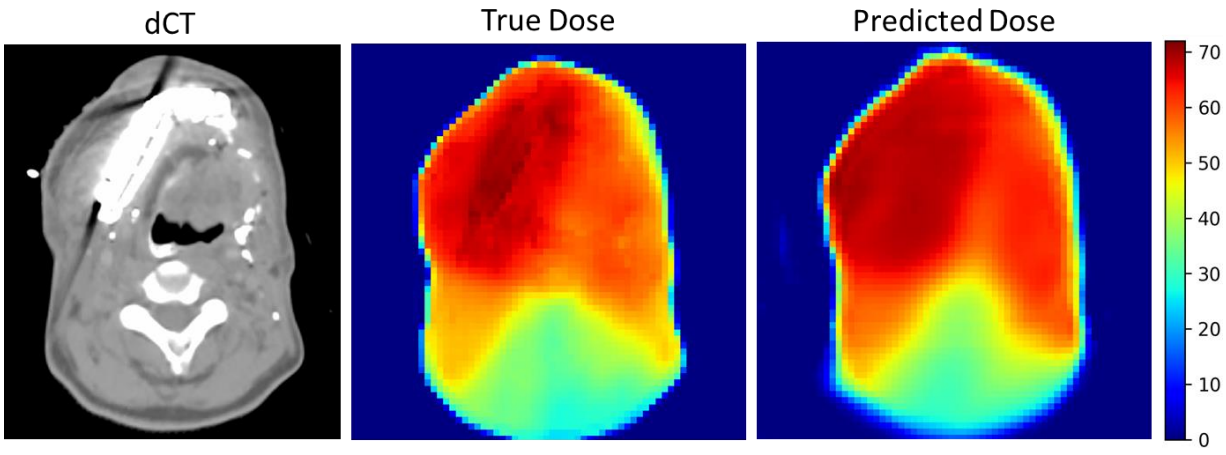


Fig. 4.14 Pt 3 dCT displaying high density artifact from dental implant, along with the true and predicted dose for the same region.

Dose prediction neural networks have largely been developed for the purpose of automating treatment planning, rather than predicting changes to delivered dose. However, the metrics used to compare predicted dose distributions or DVHs to ground truth in automated planning studies are still relevant in this context. D_{max} and D_{mean} are commonly reported, as well as DVHs and standard dose metrics for PTVs and OARs. Additionally, studies frequently report DSC between isodose volumes. Average errors in predicted D_{max} have been reported as high as 10%, though more commonly in the range of 1-2%[83,86]. Errors in D_{mean} are commonly reported in the 1-5% range[86,87]. DSCs for isodose volumes are commonly reported to be greater than 0.9, though they have been reported in the 0.8-0.9 range for dose thresholds below 40% of the prescription dose. The range in error in D_{max} for the small test set of this study was 0.7-14%, with 5 of the 6 patients having errors in excess of 2%, and DSCs for V90 to V100 isodose ranges were commonly below 0.9. The networks reported for comparison here used a larger number of patients for their training and validation, while this study was limited in the number of patients it was able to acquire, which limits training performance. The results

indicate that there is great need for improvement in this model, which may be achieved through the methods described above.

For this study, retrospective data was utilized. A prospective use of the network would involve exporting the treatment plan dose to a workstation with the pre-processing scripts and network code after planning is complete; the body contour would be exported to AlignRT to generate a reference surface, and a surface capture would be obtained after setting up the patient but before placement of the thermoplastic mask at every fraction (Fig. 4.15). At regular intervals in the treatment course, both the reference surface and that day's capture would then be exported from AlignRT and input to the pre-processing scripts alongside the treatment dose, and the pre-processing would feed directly into the network without manual intervention. The output would be a 3D dose distribution reflecting the treatment day delivered dose.

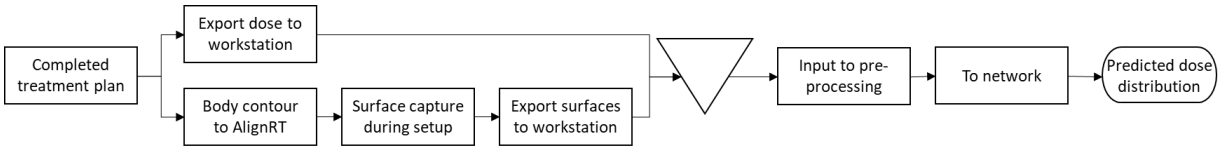


Fig. 4.15 Flowchart depicting ideal workflow for clinical use of dose prediction network.

4.5 Conclusion

This project was able to develop a neural network as proof-of-concept for predicting 3D dose of the day using the original dose distribution, the original surface, and the surface of the day as inputs. The model architecture was based on the original U-Net, with modifications for this dataset. The network was trained on 135 image sets and tested on 6 new image sets. The results presented here are promising indicators of the ability of this type of model to serve as a tool for making adaptive therapy decisions. However, further work is needed to improve the accuracy and consistency of the predictions for it to be clinically viable.

Chapter 5 Summary and Conclusions

The final chapter of this thesis summarizes the work presented here and discusses limitations of the work and possible improvements.

5.1 Summary

The work presented in this dissertation demonstrates the feasibility of using a neural network to predict changes to delivered dose in HN cases using surface data. This research sought to accomplish two specific aims, with the purpose of improving HN radiotherapy: the development and evaluation of a DIR method for CT-to-CBCT registration which delivered deformed CTs with the field of view of planning CTs and the development and initial evaluation of a neural network to predict changes in HN dose distributions based on surface data.

Fulfillment of the first specific aim was completed by the development and validation of the DIR method described in Chapter 3. Using a combination of MATLAB and elastix, an open-source software package for image registration, a process was designed to register HN planning CTs to CBCTs. The method was specifically designed to result in dCTs with the FOV of pCTs for the purpose of more accurate dose calculation. Once validation of the DIR method was completed and the method was determined to be adequate for dose calculation, work on the second specific aim began. With the ability to generate dCTs for dose of the day calculation, patients were recruited for neural network training. Treatment plans were recalculated on dCTs and surfaces were generated, resulting in the data needed to train the network. Once pre-processing was completed, the network was developed and trained. The network was tested on a small set of new patients to evaluate accuracy and verify proof-of-concept. The results

demonstrated that the concept was feasible, but that the network needed improvements and more training data.

5.2 Limitations and Future Work

Several limitations of the work and methods described here were discovered. The major limitation of this study is the limited patient cohort. This stems partially from a limitation of the developed DIR method. While some high density artifacts did not affect the quality of registrations, the frequency of poor registrations due to high density artifacts caused by dental implants led to an exclusion of most patients with high density artifacts from the network training. This also restricted patient recruitment to only those treated on Varian TrueBeams (Varian Medical Systems, Palo Alto, CA), as the poorer image quality of the on-board imagers on other systems increased the severity and frequency of artifacts. Additionally, patients who had replans performed mid-course were excluded, as the aim was to collect CBCTs throughout the entire treatment course that would be registered with the original pCT. In the future, these patients could be used to validate the dose prediction with dose calculated on the repeat CT, provided the setup is consistent. On the other hand, recruitment was largely limited to patients who exhibited visible anatomic change on CBCT, as those would be most likely to exhibit significant dosimetric change. Recruitment was eventually extended to patients with little to no anatomic change due to the low patient numbers, but this composed only a small subset of the patient cohort. Further training with a larger dataset is necessary to improve the network performance and move towards viability for use as a clinical tool. A larger dataset would also permit stratification based on dose metrics or site, which could further improve model performance.

There are several avenues to pursue for improving model performance. While a larger dataset will likely have the most impact, pre-processing is also critical to training. The current method for generating surface masks from the AlignRT files is known to have errors in certain sections of the array, most notably the superior-inferior transition from shoulders to neck. The method for drawing the polygons based on coordinates makes errors in determining where the interior of the polygon should be. The errors are restricted to a small section of the array and do not occur on all patients; however, the dose prediction is less accurate in that region due to the inaccuracy of the array. A solution for this has not yet been found but is an active point of focus. The network could be trained using the structure directly from the TPS, which would eliminate this error. However, the intent is for the network and pre-processing steps to use surfaces from AlignRT for prediction, since a surface structure would not be available in the TPS for each new fraction, so a solution must be found eventually.

The neural network may additionally benefit from alterations to the architecture, which may come in the form of additional convolutional layers, fully connected layers, residual blocks, or others. Benefits of these adaptations to the architecture could be seen more clearly with a larger dataset for training and a full set of patients for testing. One clear modification to the network would be a custom loss function. The current loss function calculates on the entire array, which includes a large volume of air, where small errors are frequent, and therefore the loss is artificially low. A custom loss function which calculates only on the body may improve learning drastically. This may require an additional input of the original body structure to the network. Attempts to create a custom loss function thus far have been unsuccessful.

Once the network is evaluated to be clinically viable, it will be necessary to determine and implement useful metrics which could be reported to physicians for the purposes of making clinical decisions about whether patients need resimulation and replanning. These metrics would need to be independent of contours and could include reporting the change in D_{max} or isodose volumes. While contour-based metrics would be ideal, the process of obtaining those metrics independent of DIR or CBCT is not trivial. The simplest method would be to evaluate the dose distribution using the pCT contours; however, this ignores the effect of any anatomic change on the structures themselves. To generate DVHs or new contours independent of DIR or CBCT that accurately reflect the geometry of the contours at each treatment fraction would likely require the development of an additional neural network separate from the one presented here. After acceptable metrics are chosen, it would then be ideal to perform a prospective study with the clinical tool implemented.

5.3 General conclusions

This thesis investigated the development of a CT-to-CBCT DIR method and a 3D dose prediction neural network for HN patients treated with RT. This research was driven by two specific aims.

The first, to develop a CT-to-CBCT DIR method, was necessary due to the lack of commercially available DIR packages which preserve the field of view of the pCT when generating the dCT. Without an image with a full field of view and correct HU, accurate dose calculation is impossible. Therefore, an in-house method was developed which addressed these challenges and was validated on ten patients. Validation demonstrated that the method was comparable to published in-house and commercial algorithms. The increased FOV of the dCT

relative to commercially-generated dCTs allowed for accurate dose-of-the-day calculation and surface generation, necessary for the second aim.

The second aim included the development and evaluation of a proof-of-concept neural network in the pursuit of creating a clinical tool for informing clinical decisions about replanning during a HN patient's treatment course. The developed neural network uses the original dose distribution and surface information to predict how the dose distribution has changed. The predictions for test patients indicate the need for improvement in accuracy, but they are promising indicators of feasibility. Given a larger patient cohort and the modifications to the pre-processing methods and network proposed above, it is expected that the model could be developed into a useful clinical tool.

Both the first and second specific aims were successfully achieved, and a novel method for predicting dose changes during HN RT was presented which shows promise for clinical use and improving adaptive therapy.

References

1. Nevens D, Duprez F, Daisne JF, et al. Reduction of the dose of radiotherapy to the elective neck in head and neck squamous cell carcinoma; a randomized clinical trial. Effect on late toxicity and tumor control. *Radiother Oncol*. 2017;122(2):171-177. doi:10.1016/j.radonc.2016.08.009

2. Dirix P, Abbeel S, Vanstraelen B, Hermans R, Nuyts S. Dysphagia After Chemoradiotherapy for Head-and-Neck Squamous Cell Carcinoma: Dose–Effect Relationships for the Swallowing Structures. *Int J Radiat Oncol*. 2009;75(2):385-392. doi:10.1016/j.ijrobp.2008.11.041

3. Levendag PC, Teguh DN, Voet P, et al. Dysphagia disorders in patients with cancer of the oropharynx are significantly affected by the radiation therapy dose to the superior and middle constrictor muscle: A dose-effect relationship. *Radiother Oncol*. 2007;85(1):64-73. doi:10.1016/j.radonc.2007.07.009

4. Duprez F, Madani I, De Potter B, Boterberg T, De Neve W. Systematic Review of Dose–Volume Correlates for Structures Related to Late Swallowing Disturbances After Radiotherapy for Head and Neck Cancer. *Dysphagia*. 2013;28(3):337-349. doi:10.1007/s00455-013-9452-2

5. Chow LQM. Head and Neck Cancer. *N Engl J Med*. 2020;382(1):60-72. doi:10.1056/NEJMra1715715

6. Cancer deaths by type. Our World in Data. Accessed June 29, 2022. https://ourworldindata.org/grapher/total-cancer-deaths-by-type

7. Hashim D, Genden E, Posner M, Hashibe M, Boffetta P. Head and neck cancer prevention: from primary prevention to impact of clinicians on reducing burden. *Ann Oncol*. 2019;30(5):744-756. doi:10.1093/annonc/mdz084

8. Mock U, Georg D, Bogner J, Auberger T, Pötter R. Treatment planning comparison of conventional, 3D conformal, and intensity-modulated photon (IMRT) and proton therapy for paranasal sinus carcinoma. *Int J Radiat Oncol*. 2004;58(1):147-154. doi:10.1016/S0360-3016(03)01452-4

9. Cozzi L, Fogliata A, Bolsi A, Nicolini G, Bernier J. Three-dimensional conformal vs. intensity-modulated radiotherapy in head-and-neck cancer patients: comparative analysis of dosimetric and technical parameters. *Int J Radiat Oncol*. 2004;58(2):617-624. doi:10.1016/j.ijrobp.2003.09.059

10. Kouloulias V, Thalassinou S, Platoni K, et al. The Treatment Outcome and Radiation-Induced Toxicity for Patients with Head and Neck Carcinoma in the IMRT Era: A Systematic Review with Dosimetric and Clinical Parameters. *BioMed Res Int*. 2013;2013:401261. doi:10.1155/2013/401261

11. Gomez–Millan Barrachina J, Jerez Sainz I, Perez Rozos A, et al. Potential advantages of volumetric arc therapy in head and neck cancer. *Head Neck*. 2015;37(6):909-914. doi:10.1002/hed.23685

12. Orlandi E, Palazzi M, Pignoli E, Fallai C, Giostra A, Olmi P. Radiobiological basis and clinical results of the simultaneous integrated boost (SIB) in intensity modulated radiotherapy (IMRT) for head and neck cancer: A review. *Crit Rev Oncol Hematol*. 2010;73(2):111-125. doi:10.1016/j.critrevonc.2009.03.003

13. Dogan N, King S, Emami B, et al. Assessment of different IMRT boost delivery methods on target coverage and normal-tissue sparing. *Int J Radiat Oncol*. 2003;57(5):1480-1491. doi:10.1016/S0360-3016(03)01569-4

14. Margalit DN, Sacco AG, Cooper JS, et al. Systematic review of postoperative therapy for resected squamous cell carcinoma of the head and neck: Executive summary of the American Radium Society appropriate use criteria. *Head Neck*. 2021;43(1):367-391. doi:10.1002/hed.26490

15. Castadot P, Geets X, Lee JA, Christian N, Grégoire V. Assessment by a deformable registration method of the volumetric and positional changes of target volumes and organs at risk in pharyngo-laryngeal tumors treated with concomitant chemo-radiation. *Radiother Oncol*. 2010;95(2):209-217. doi:10.1016/j.radonc.2010.03.007

16. Ahn PH, Chen CC, Ahn AI, et al. Adaptive Planning in Intensity-Modulated Radiation Therapy for Head and Neck Cancers: Single-Institution Experience and Clinical Implications. *Int J Radiat Oncol*. 2011;80(3):677-685. doi:10.1016/j.ijrobp.2010.03.014

17. Castadot P, Lee JA, Geets X, Grégoire V. Adaptive Radiotherapy of Head and Neck Cancer. *Semin Radiat Oncol*. 2010;20(2):84-93. doi:10.1016/j.semradonc.2009.11.002

18. Kataria T, Gupta D, Goyal S, et al. Clinical outcomes of adaptive radiotherapy in head and neck cancers. *Br J Radiol*. 89(1062). doi:10.1259/bjr.20160085

19. Yip C, Thomas C, Michaelidou A, et al. Co-registration of cone beam CT and planning CT in head and neck IMRT dose estimation: a feasible adaptive radiotherapy strategy. *Br J Radiol*. 2014;87(1034). doi:10.1259/bjr.20130532

20. Barker JL, Garden AS, Ang KK, et al. Quantification of volumetric and geometric changes occurring during fractionated radiotherapy for head-and-neck cancer using an integrated CT/linear accelerator system. *Int J Radiat Oncol*. 2004;59(4):960-970. doi:10.1016/j.ijrobp.2003.12.024

21. Height R, Khoo V, Lawford C, et al. The dosimetric consequences of anatomic changes in head and neck radiotherapy patients. *J Med Imaging Radiat Oncol*. 2010;54(5):497-504. doi:10.1111/j.1754-9485.2010.02209.x

22. Lee C, Langen KM, Lu W, et al. Evaluation of geometric changes of parotid glands during head and neck cancer radiotherapy using daily MVCT and automatic deformable registration. *Radiother Oncol*. 2008;89(1):81-88. doi:10.1016/j.radonc.2008.07.006

23. Jensen AD, Nill S, Huber PE, Bendl R, Debus J, Münter MW. A Clinical Concept for Interfractional Adaptive Radiation Therapy in the Treatment of Head and Neck Cancer. *Int J Radiat Oncol*. 2012;82(2):590-596. doi:10.1016/j.ijrobp.2010.10.072

24. Castelli J, Simon A, Lafond C, et al. Adaptive radiotherapy for head and neck cancer. *Acta Oncol*. 2018;57(10):1284-1292. doi:10.1080/0284186X.2018.1505053

25. Lee C, Langen KM, Lu W, et al. Assessment of Parotid Gland Dose Changes During Head and Neck Cancer Radiotherapy Using Daily Megavoltage Computed Tomography and Deformable Image Registration. *Int J Radiat Oncol*. 2008;71(5):1563-1571. doi:10.1016/j.ijrobp.2008.04.013

26. Beltran M, Ramos M, Rovira JJ, et al. Dose variations in tumor volumes and organs at risk during IMRT for head-and-neck cancer. *J Appl Clin Med Phys*. 2012;13(6):101-111. doi:10.1120/jacmp.v13i6.3723

27. Zhao L, Wan Q, Zhou Y, Deng X, Xie C, Wu S. The role of replanning in fractionated intensity modulated radiotherapy for nasopharyngeal carcinoma. *Radiother Oncol*. 2011;98(1):23-27. doi:10.1016/j.radonc.2010.10.009

28. Yoon SW, Lin H, Alonso-Basanta M, et al. Initial Evaluation of a Novel Cone-Beam CT-Based Semi-Automated Online Adaptive Radiotherapy System for Head and Neck Cancer Treatment – A Timing and Automation Quality Study. *Cureus*. 12(8):e9660. doi:10.7759/cureus.9660

29. Sibolt P, Andersson LM, Calmels L, et al. Clinical implementation of artificial intelligence-driven cone-beam computed tomography-guided online adaptive radiotherapy in the pelvic region. *Phys Imaging Radiat Oncol*. 2021;17:1-7. doi:10.1016/j.phro.2020.12.004

30. Muelas-Soria R, García-Mollá R, Morillo-Macías V, et al. The Usefulness of Adaptative Radiotherapy in Prostate Cancer: How, When, and Who? *Biomedicines*. 2022;10(6):1401. doi:10.3390/biomedicines10061401

31. Witt JS, Rosenberg SA, Bassetti MF. MRI-guided adaptive radiotherapy for liver tumours: visualising the future. *Lancet Oncol*. 2020;21(2):e74-e82. doi:10.1016/S1470-2045(20)30034-6

32. Piperdi H, Portal D, Neibart SS, Yue NJ, Jabbour SK, Reyhan M. Adaptive Radiation Therapy in the Treatment of Lung Cancer: An Overview of the Current State of the Field. *Front Oncol*. 2021;11. Accessed December 15, 2022. https://www.frontiersin.org/articles/10.3389/fonc.2021.770382

33. Guidi G, Maffei N, Meduri B, et al. A machine learning tool for re-planning and adaptive RT: A multicenter cohort investigation. *Phys Med*. 2016;32(12):1659-1666. doi:10.1016/j.ejmp.2016.10.005

34. Figen M, Çolpan Öksüz D, Duman E, et al. Radiotherapy for Head and Neck Cancer: Evaluation of Triggered Adaptive Replanning in Routine Practice. *Front Oncol*. 2020;10:579917. doi:10.3389/fonc.2020.579917

35. Morgan HE, Sher DJ. Adaptive radiotherapy for head and neck cancer. *Cancers Head Neck*. 2020;5:1. doi:10.1186/s41199-019-0046-z

36. Castelli J, Simon A, Rigaud B, et al. A Nomogram to predict parotid gland overdose in head and neck IMRT. *Radiat Oncol Lond Engl*. 2016;11. doi:10.1186/s13014-016-0650-6

37. Brouwer CL, Steenbakkers RJHM, van der Schaaf A, et al. Selection of head and neck cancer patients for adaptive radiotherapy to decrease xerostomia. *Radiother Oncol*. 2016;120(1):36-40. doi:10.1016/j.radonc.2016.05.025

38. Surucu M, Shah KK, Mescioglu I, et al. Decision Trees Predicting Tumor Shrinkage for Head and Neck Cancer: Implications for Adaptive Radiotherapy. *Technol Cancer Res Treat*. 2016;15(1):139-145. doi:10.1177/1533034615572638

39. Brown E, Owen R, Harden F, et al. Predicting the need for adaptive radiotherapy in head and neck cancer. *Radiother Oncol*. 2015;116(1):57-63. doi:10.1016/j.radonc.2015.06.025

40. Brown E, Owen R, Harden F, et al. Head and neck adaptive radiotherapy: Predicting the time to replan. *Asia Pac J Clin Oncol*. 2016;12(4):460-467. doi:10.1111/ajco.12516

41. Vickress JR, Battista J, Barnett R, Yartsev S. Online daily assessment of dose change in head and neck radiotherapy without dose-recalculation. *J Appl Clin Med Phys*. 2018;19(5):659-665. doi:10.1002/acm2.12432

42. Capelle L, Mackenzie M, Field C, Parliament M, Ghosh S, Scrimger R. Adaptive Radiotherapy Using Helical Tomotherapy for Head and Neck Cancer in Definitive and Postoperative Settings: Initial Results. *Clin Oncol*. 2012;24(3):208-215. doi:10.1016/j.clon.2011.11.005

43. Brown E, Porceddu S, Owen R, Harden F. Developing an Adaptive Radiotherapy Technique for Virally Mediated Head and Neck Cancer. *J Med Imaging Radiat Sci*. 2013;44(3):134-140. doi:10.1016/j.jmir.2013.04.001

44. Chitapanarux I, Chomprasert K, Nobnaop W, et al. A dosimetric comparison of two-phase adaptive intensity-modulated radiotherapy for locally advanced nasopharyngeal cancer. *J Radiat Res (Tokyo)*. 2015;56(3):529-538. doi:10.1093/jrr/rru119

45. Schwartz DL, Garden AS, Shah SJ, et al. Adaptive radiotherapy for head and neck cancer—Dosimetric results from a prospective clinical trial. *Radiother Oncol*. 2013;106(1):80-84. doi:10.1016/j.radonc.2012.10.010

46. Surucu M, Shah KK, Roeske JC, Choi M, Small W, Emami B. Adaptive Radiotherapy for Head and Neck Cancer: Implications for Clinical and Dosimetry Outcomes. *Technol Cancer Res Treat*. 2017;16(2):218-223. doi:10.1177/1533034616662165

47. Schwartz DL, Garden AS, Thomas J, et al. Adaptive Radiotherapy for Head and Neck Cancer: Initial Clinical Outcomes from a Prospective Trial. *Int J Radiat Oncol Biol Phys*. 2012;83(3):986-993. doi:10.1016/j.ijrobp.2011.08.017

48. Chen AM, Daly ME, Cui J, Mathai M, Benedict S, Purdy JA. Clinical outcomes among patients with head and neck cancer treated by intensity-modulated radiotherapy with and without adaptive replanning. *Head Neck*. 2014;36(11):1541-1546. doi:10.1002/hed.23477

49. Simone CB, Ly D, Dan TD, et al. Comparison of Intensity-modulated Radiotherapy, Adaptive Radiotherapy, Proton Radiotherapy, and Adaptive Proton Radiotherapy for Treatment of Locally Advanced Head and Neck Cancer. *Radiother Oncol J Eur Soc Ther Radiol Oncol*. 2011;101(3):376-382. doi:10.1016/j.radonc.2011.05.028

50. Chen AM, Yoshizaki T, Hsu S, Mikaeilian A, Cao M. Image-guided adaptive radiotherapy improves acute toxicity during intensity-modulated radiation therapy for head and neck cancer. *J Radiat Oncol*. 2018;7(2):139-145. doi:10.1007/s13566-017-0336-1

51. Castelli J, Simon A, Louvel G, et al. Impact of head and neck cancer adaptive radiotherapy to spare the parotid glands and decrease the risk of xerostomia. *Radiat Oncol Lond Engl*. 2015;10. doi:10.1186/s13014-014-0318-z

52. Al-Hallaq HA, Cerviño L, Gutierrez AN, et al. AAPM task group report 302: Surface-guided radiotherapy. *Med Phys*. 2022;49(4):e82-e112. doi:10.1002/mp.15532

53. Willoughby T, Lehmann J, Bencomo JA, et al. Quality assurance for nonradiographic radiotherapy localization and positioning systems: Report of Task Group 147. *Med Phys*. 2012;39(4):1728-1747. doi:10.1118/1.3681967

54. Bert C, Metheany KG, Doppke K, Chen GTY. A phantom evaluation of a stereo-vision surface imaging system for radiotherapy patient setup. *Med Phys*. 2005;32(9):2753-2762. doi:10.1118/1.1984263

55. Hoisak JDP, Pawlicki T. The Role of Optical Surface Imaging Systems in Radiation Therapy. *Semin Radiat Oncol*. 2018;28(3):185-193. doi:10.1016/j.semradonc.2018.02.003

56. Gopan O, Wu Q. Evaluation of the Accuracy of a 3D Surface Imaging System for Patient Setup in Head and Neck Cancer Radiotherapy. *Int J Radiat Oncol*. 2012;84(2):547-552. doi:10.1016/j.ijrobp.2011.12.004

57. Padilla L, Havnen-Smith A, Cerviño L, Al-Hallaq HA. A survey of surface imaging use in radiation oncology in the United States. *J Appl Clin Med Phys*. 2019;20(12):70-77. doi:10.1002/acm2.12762

58. Cerviño LI, Detorie N, Taylor M, et al. Initial clinical experience with a frameless and maskless stereotactic radiosurgery treatment. *Pract Radiat Oncol*. 2012;2(1):54-62. doi:10.1016/j.prro.2011.04.005

59. Wiant D, Squire S, Liu H, Maurer J, Lane Hayes T, Sintay B. A prospective evaluation of open face masks for head and neck radiation therapy. *Pract Radiat Oncol*. 2016;6(6):e259-e267. doi:10.1016/j.prro.2016.02.003

60. Lee SK, Huang S, Zhang L, et al. Accuracy of surface-guided patient setup for conventional radiotherapy of brain and nasopharynx cancer. *J Appl Clin Med Phys*. 2021;22(5):48-57. doi:10.1002/acm2.13241

61. Zhao B, Maquilan G, Jiang S, Schwartz DL. Minimal mask immobilization with optical surface guidance for head and neck radiotherapy. *J Appl Clin Med Phys*. 19(1):17-24. doi:10.1002/acm2.12211

62. 8. Image Registration - Knovel. Accessed November 23, 2022. https://app-knovel-com.proxy.library.vcu.edu/web/view/khtml/show.v/rcid:kpHMIVMIP7/cid:kt0087DEA5/viewerType:khtml/root_slug:handbook-medical-imaging/url_slug:image-registration?&b-toc-cid=kpHMIVMIP7&b-toc-root-slug=handbook-medical-imaging&b-toc-title=Handbook%20of%20Medical%20Imaging%2C%20Volume%202%20-%20Medical%20Image%20Processing%20and%20Analysis&b-toc-url-slug=statistical-image-reconstruction&kpromoter=federation&page=1&view=collapsed&zoom=1

63. Klein S, Staring M, Murphy K, Viergever MA, Pluim JPW. elastix: A Toolbox for Intensity-Based Medical Image Registration. *IEEE Trans Med Imaging*. 2010;29(1):196-205. doi:10.1109/TMI.2009.2035616

64. Robbins H, Monro S. A Stochastic Approximation Method. *Ann Math Stat*. 1951;22(3):400-407.

65. Klein S, Staring M, Pluim JPW. Evaluation of Optimization Methods for Nonrigid Medical Image Registration Using Mutual Information and B-Splines. *IEEE Trans Image Process*. 2007;16(12):2879-2890. doi:10.1109/TIP.2007.909412

66. Jiang C, Huang Y, Ding S, et al. Comparison of an in-house hybrid DIR method to NiftyReg on CBCT and CT images for head and neck cancer. *J Appl Clin Med Phys*. 2022;23(3):e13540. doi:10.1002/acm2.13540

67. Hou J, Guerrero M, Chen W, D'Souza WD. Deformable planning CT to cone-beam CT image registration in head-and-neck cancer. *Med Phys*. 2011;38(4):2088-2094. doi:10.1118/1.3554647

68. Landry G, Nijhuis R, Dedes G, et al. Investigating CT to CBCT image registration for head and neck proton therapy as a tool for daily dose recalculation. *Med Phys*. 2015;42(3):1354-1366. doi:10.1118/1.4908223

69. Veiga C, McClelland J, Moinuddin S, et al. Toward adaptive radiotherapy for head and neck patients: Feasibility study on using CT-to-CBCT deformable registration for "dose of the day" calculations. *Med Phys*. 2014;41(3):031703. doi:10.1118/1.4864240

70. Takayama Y, Kadoya N, Yamamoto T, et al. Evaluation of the performance of deformable image registration between planning CT and CBCT images for the pelvic region: comparison between hybrid and intensity-based DIR. *J Radiat Res (Tokyo)*. 2017;58(4):567-571. doi:10.1093/jrr/rrw123

71. Kim J, Kumar S, Liu C, et al. A novel approach for establishing benchmark CBCT/CT deformable image registrations in prostate cancer radiotherapy. *Phys Med Biol*. 2013;58(22):8077-8097. doi:10.1088/0031-9155/58/22/8077

72. Leibfarth S, Mönnich D, Welz S, et al. A strategy for multimodal deformable image registration to integrate PET/MR into radiotherapy treatment planning. *Acta Oncol*. 2013;52(7):1353-1359. doi:10.3109/0284186X.2013.813964

73. Hammers JE, Pirozzi S, Lindsay D, et al. Evaluation of a commercial DIR platform for contour propagation in prostate cancer patients treated with IMRT/VMAT. *J Appl Clin Med Phys*. 2020;21(2):14-25. doi:10.1002/acm2.12787

74. Latifi K, Caudell J, Zhang G, Hunt D, Moros EG, Feygelman V. Practical quantification of image registration accuracy following the AAPM TG-132 report framework. *J Appl Clin Med Phys*. 2018;19(4):125-133. doi:10.1002/acm2.12348

75. Loi G, Fusella M, Lanzi E, et al. Performance of commercially available deformable image registration platforms for contour propagation using patient-based computational phantoms: A multi-institutional study. *Med Phys*. 2018;45(2):748-757. doi:10.1002/mp.12737

76. Riegel AC, Antone JG, Zhang H, et al. Deformable image registration and interobserver variation in contour propagation for radiation therapy planning. *J Appl Clin Med Phys*. 2016;17(3):347-357. doi:10.1120/jacmp.v17i3.6110

77. Brock KK, Mutic S, McNutt TR, Li H, Kessler ML. Use of image registration and fusion algorithms and techniques in radiotherapy: Report of the AAPM Radiation Therapy Committee Task Group No. 132. *Med Phys*. 2017;44(7):e43-e76. doi:10.1002/mp.12256

78. Zukauskaite R, Brink C, Hansen CR, et al. Open source deformable image registration system for treatment planning and recurrence CT scans: Validation in the head and neck region. *Strahlenther Onkol*. 2016;192(8):545-551. doi:http://dx.doi.org/10.1007/s00066-016-0998-4

79. Meyer P, Noblet V, Mazzara C, Lallement A. Survey on deep learning for radiotherapy. *Comput Biol Med*. 2018;98:126-146. doi:10.1016/j.compbiomed.2018.05.018

80. Huang G, Liu Z, van der Maaten L, Weinberger KQ. Densely Connected Convolutional Networks. *ArXiv160806993 Cs*. Published online January 28, 2018. Accessed July 20, 2020. http://arxiv.org/abs/1608.06993

81. Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *ArXiv150504597 Cs*. Published online May 18, 2015. Accessed July 20, 2020. http://arxiv.org/abs/1505.04597

82. Nguyen D, Long T, Jia X, et al. A feasibility study for predicting optimal radiation therapy dose distributions of prostate cancer patients from patient anatomy using deep learning. *Sci Rep*. 2019;9(1). doi:10.1038/s41598-018-37741-x

83. Nguyen D, Jia X, Sher D, et al. 3D radiotherapy dose prediction on head and neck cancer patients with a hierarchically densely connected U-net deep learning architecture. *Phys Med Biol*. 2019;64(6):065020. doi:10.1088/1361-6560/ab039b

84. Kearney V, Chan JW, Haaf S, Descovich M, Solberg TD. DoseNet: a volumetric dose prediction algorithm using 3D fully-convolutional neural networks. *Phys Med Biol*. 2018;63(23):235022. doi:10.1088/1361-6560/aaef74

85. Thomas MA, Fu Y, Yang D. Development and evaluation of machine learning models for voxel dose predictions in online adaptive magnetic resonance guided radiation therapy. *J Appl Clin Med Phys*. n/a(n/a). doi:10.1002/acm2.12884

86. Ahn SH, Kim E, Kim C, et al. Deep learning method for prediction of patient-specific dose distribution in breast cancer. *Radiat Oncol Lond Engl*. 2021;16(1):154. doi:10.1186/s13014-021-01864-9

87. Barragán-Montero AM, Nguyen D, Lu W, et al. Three-dimensional dose prediction for lung IMRT patients with deep neural networks: robust learning from heterogeneous beam configurations. *Med Phys*. 2019;46(8):3679-3691. doi:10.1002/mp.13597

88. Malone C, Fennell L, Folliard T, Kelly C. Using a neural network to predict deviations in mean heart dose during the treatment of left-sided deep inspiration breath hold patients. *Phys Med*. 2019;65:137-142. doi:10.1016/j.ejmp.2019.08.014

89. Murakami Y, Magome T, Matsumoto K, Sato T, Yoshioka Y, Oguchi M. Fully automated dose prediction using generative adversarial networks in prostate cancer patients. *PLOS ONE*. 2020;15(5):e0232697. doi:10.1371/journal.pone.0232697

Appendix A Additional Figures

This appendix includes patient-specific figures for the DIR validation, dose calculation on the DIR validation patients, and neural network test set.

A.1 HU Histograms

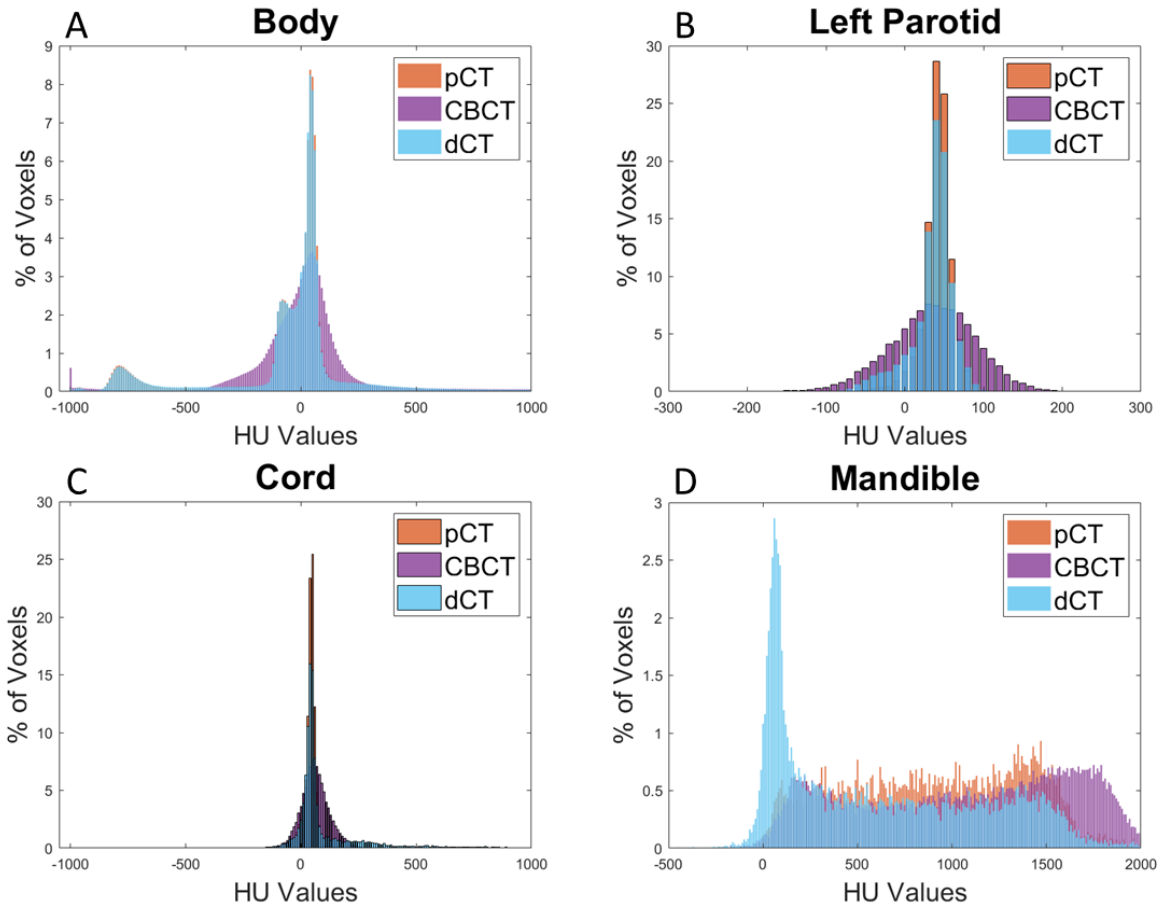


Fig A.1 Histograms of HU values for pCT, CBCT, and dCT for Pt. 1. A: Body, B: Left parotid, C: Cord, D: Mandible. Note: Pt.1 did not have a right parotid.

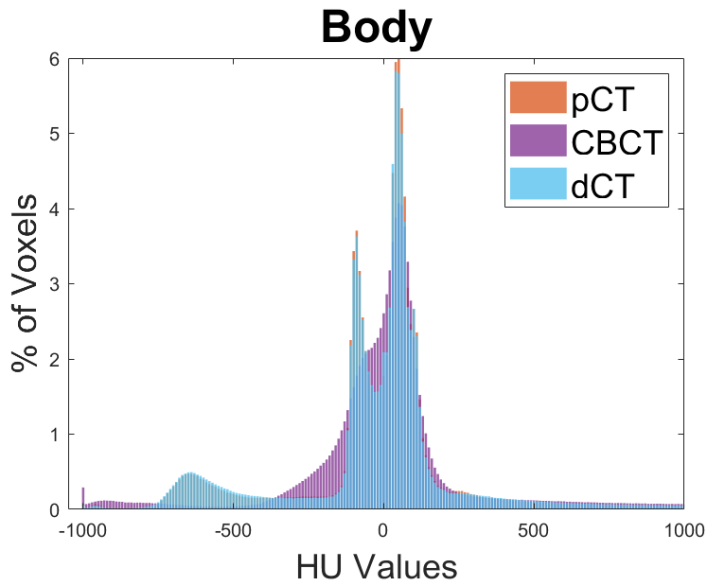


Fig A.2 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 2.

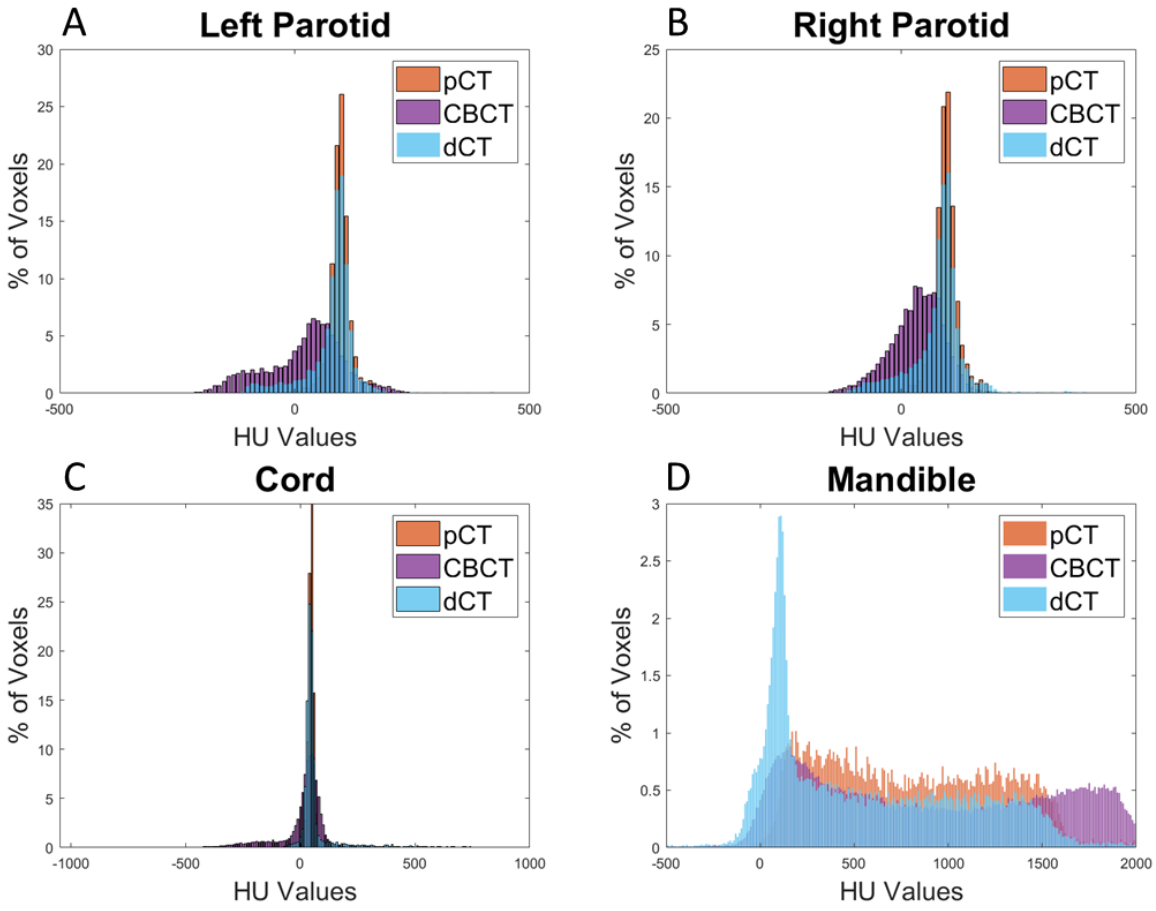


Fig A.3 Histograms of HU values for pCT, CBCT, and dCT for Pt. 2. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

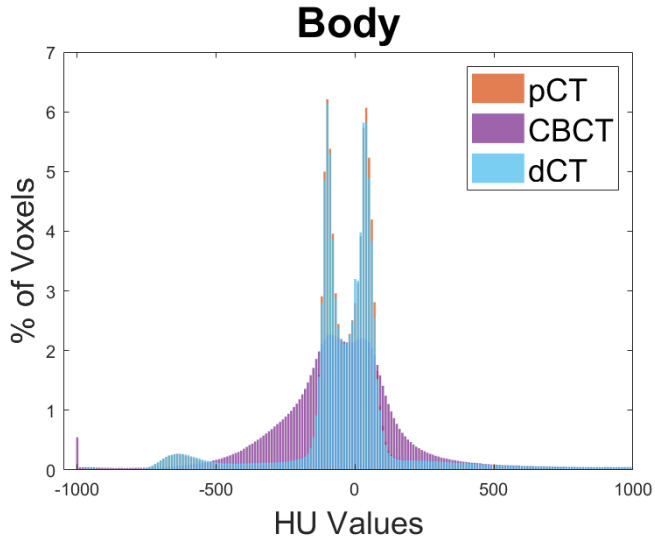


Fig A.4 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 3.

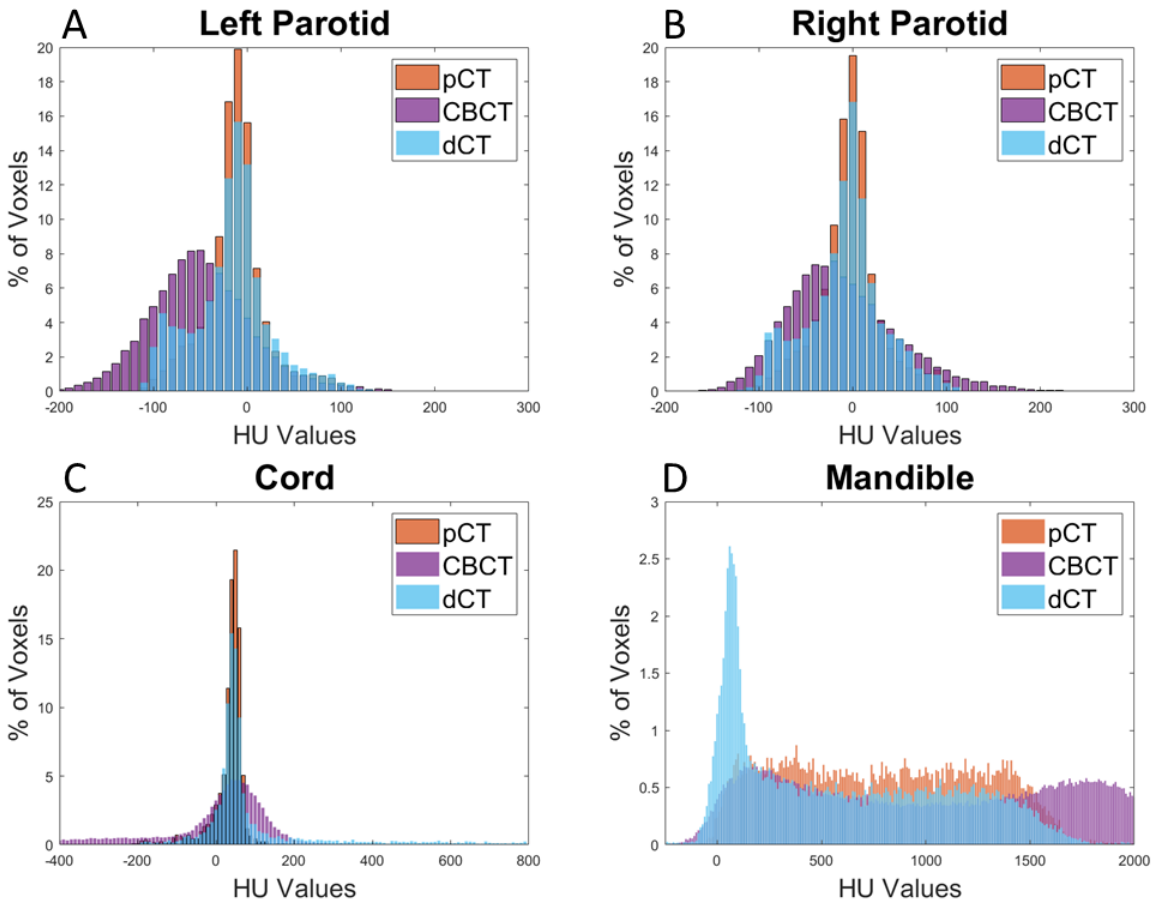


Fig A.5 Histograms of HU values for pCT, CBCT, and dCT for Pt. 3. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

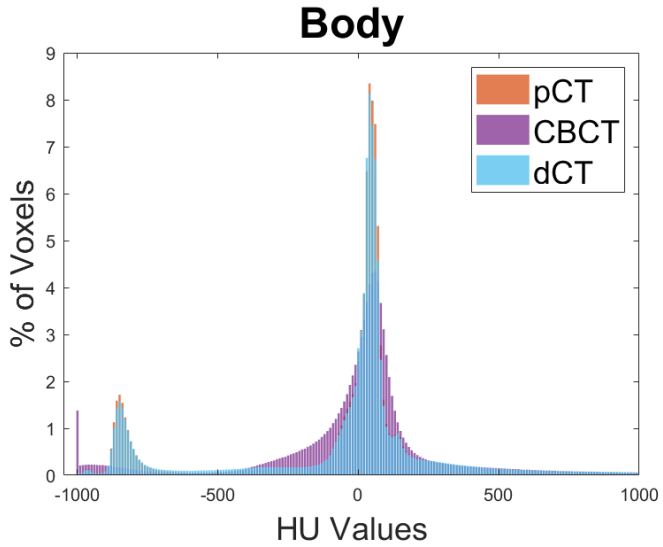


Fig A.6 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 4.

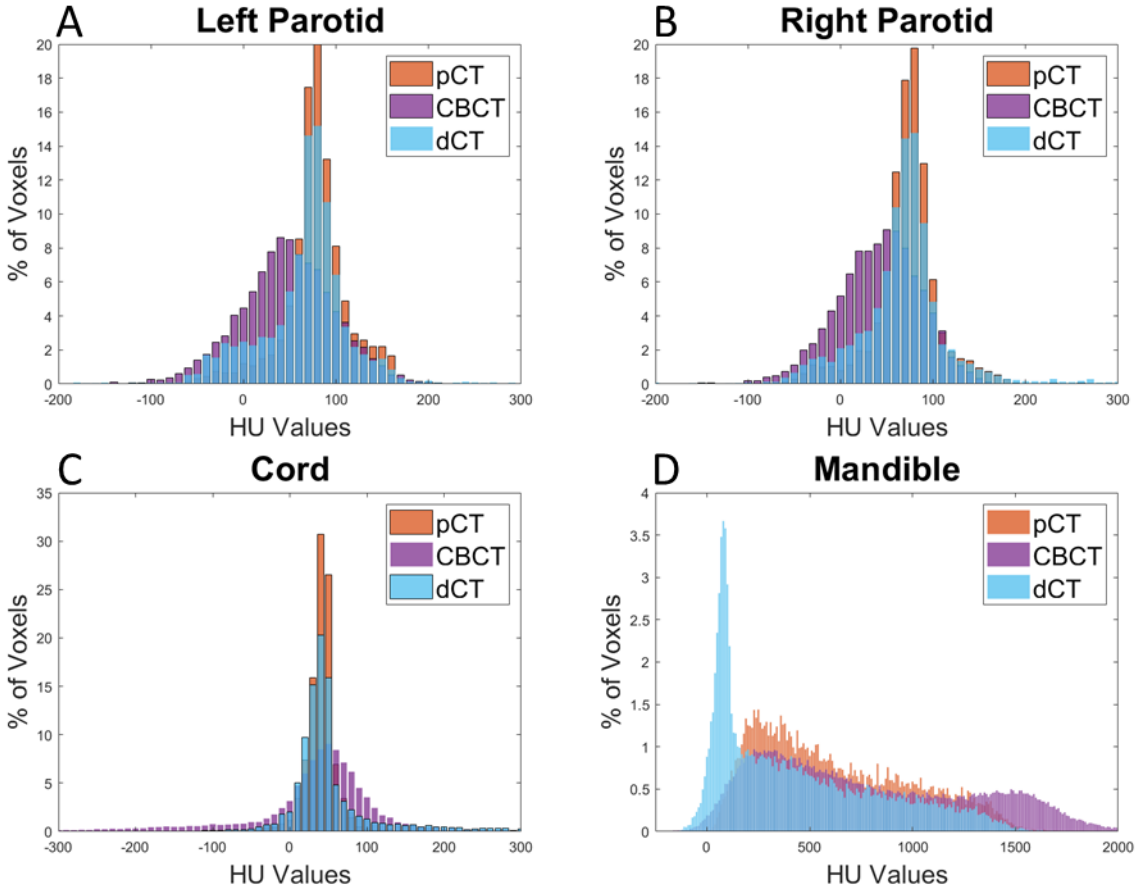


Fig A.7 Histograms of HU values for pCT, CBCT, and dCT for Pt. 4. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

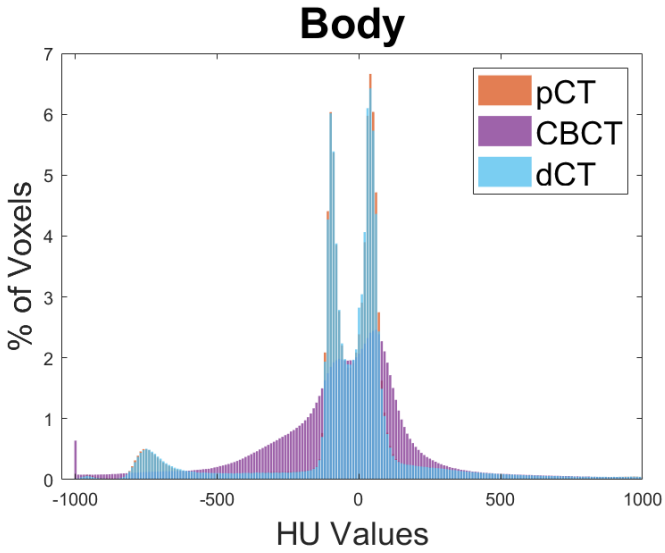


Fig A.8 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 5.

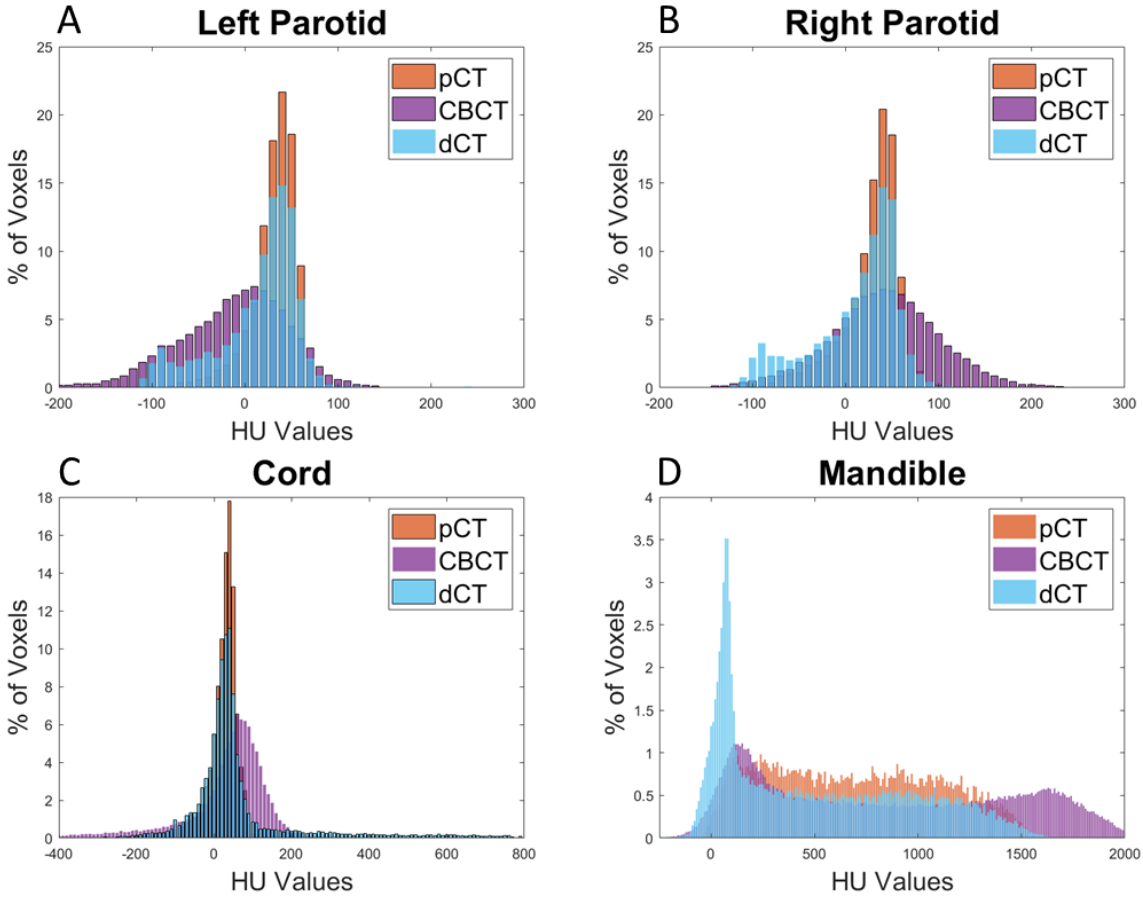


Fig A.9 Histograms of HU values for pCT, CBCT, and dCT for Pt. 5. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

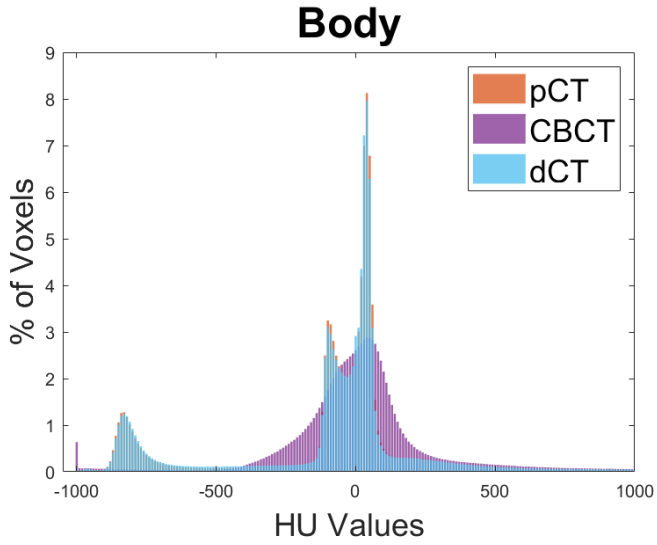


Fig A.10 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 6.

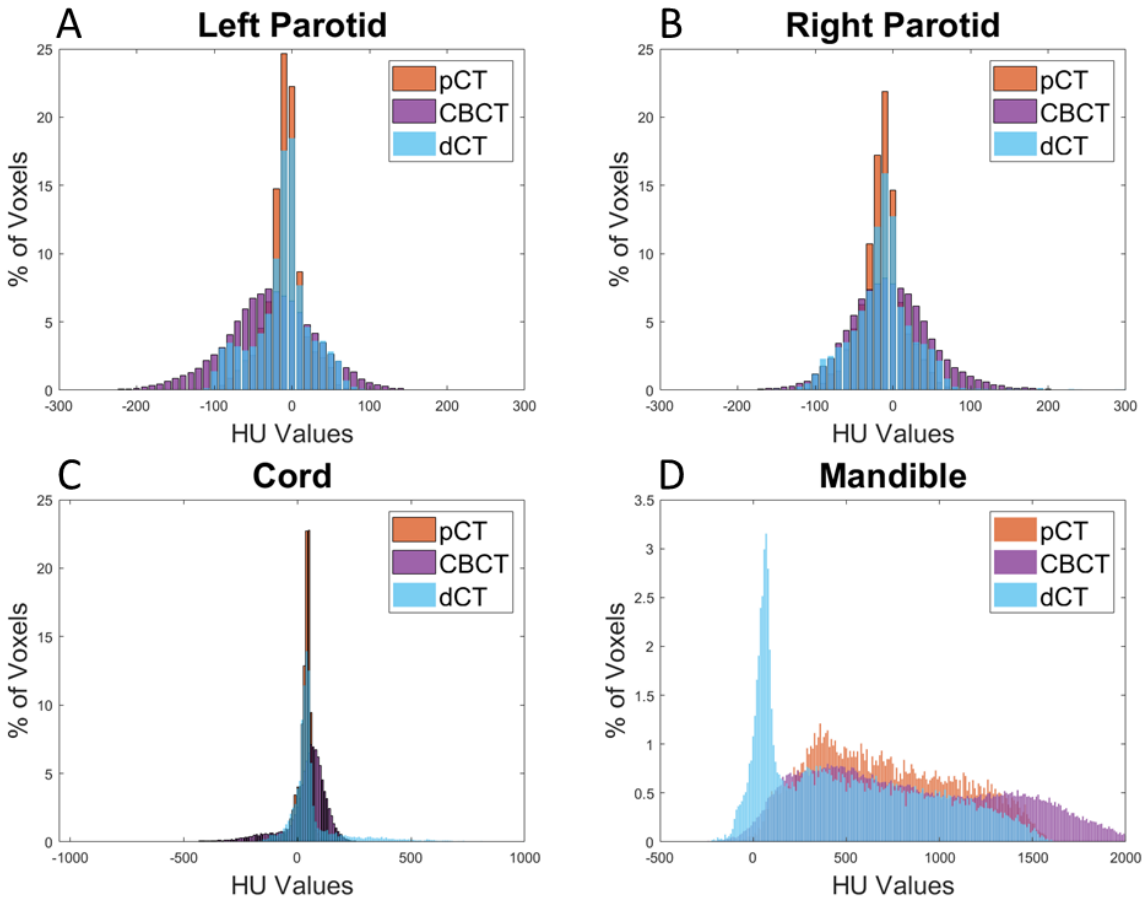


Fig A.11 Histograms of HU values for pCT, CBCT, and dCT for Pt. 6. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

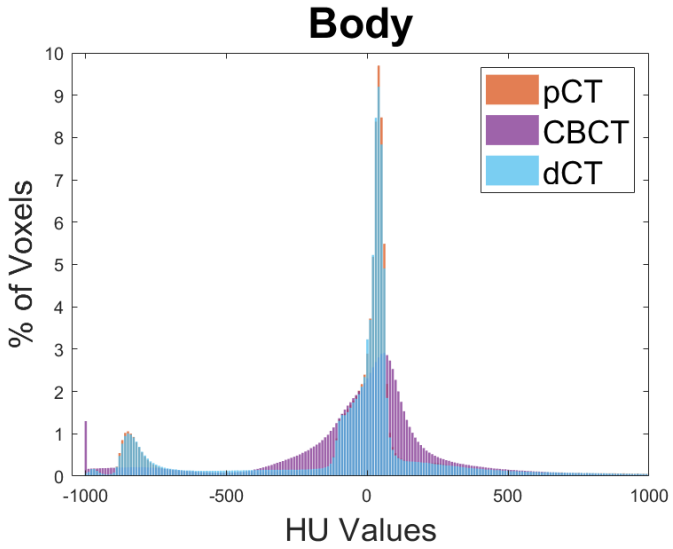


Fig A.12 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 7.

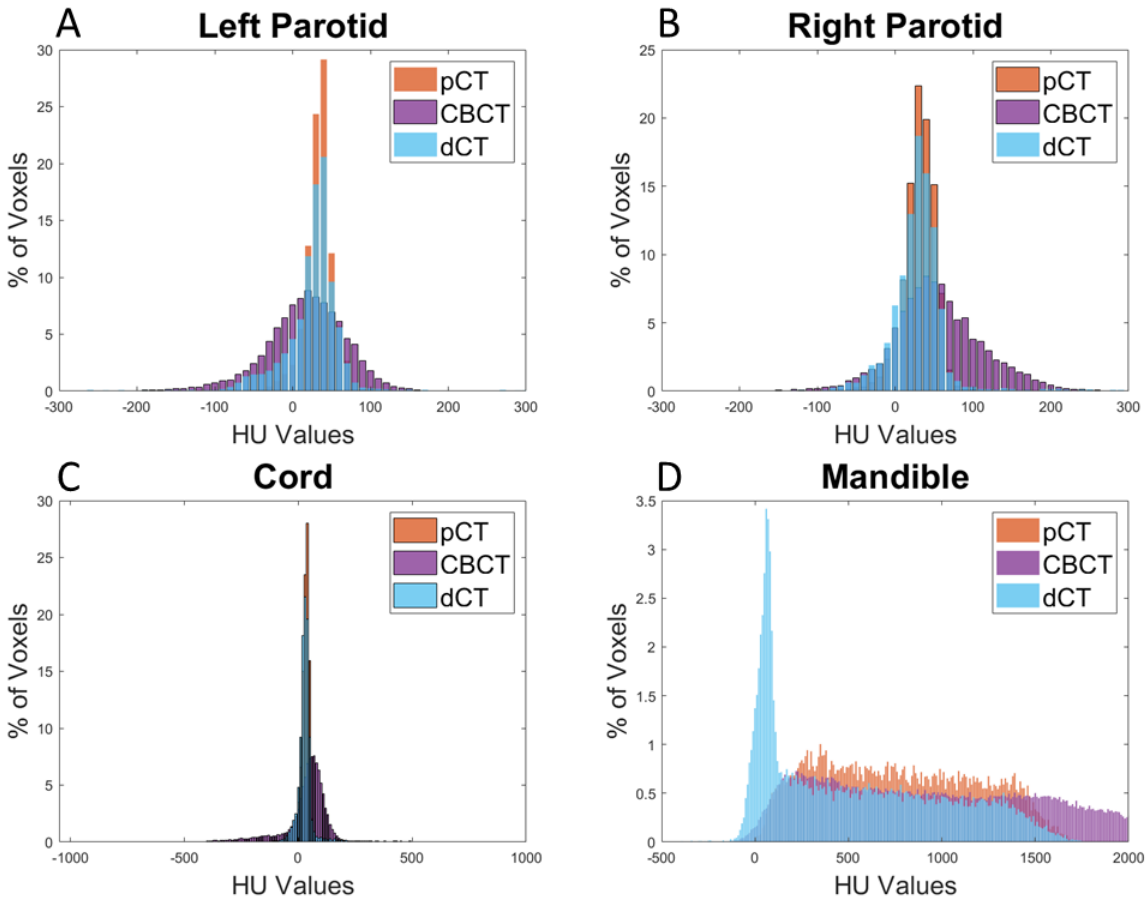


Fig A.13 Histograms of HU values for pCT, CBCT, and dCT for Pt. 7. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

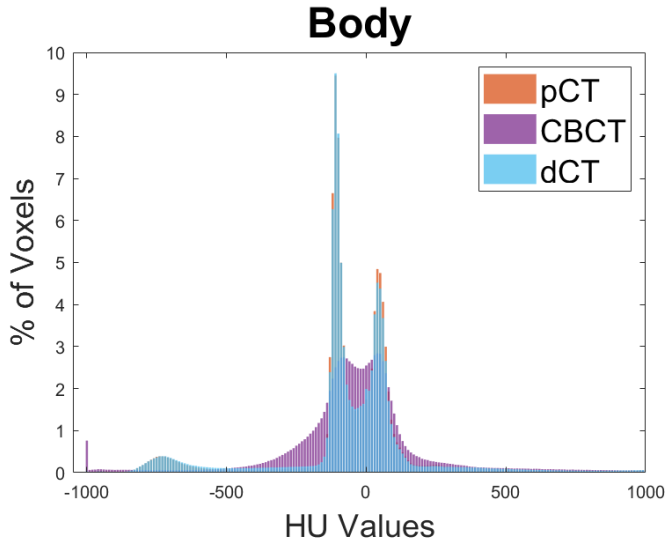


Fig A.14 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 8.

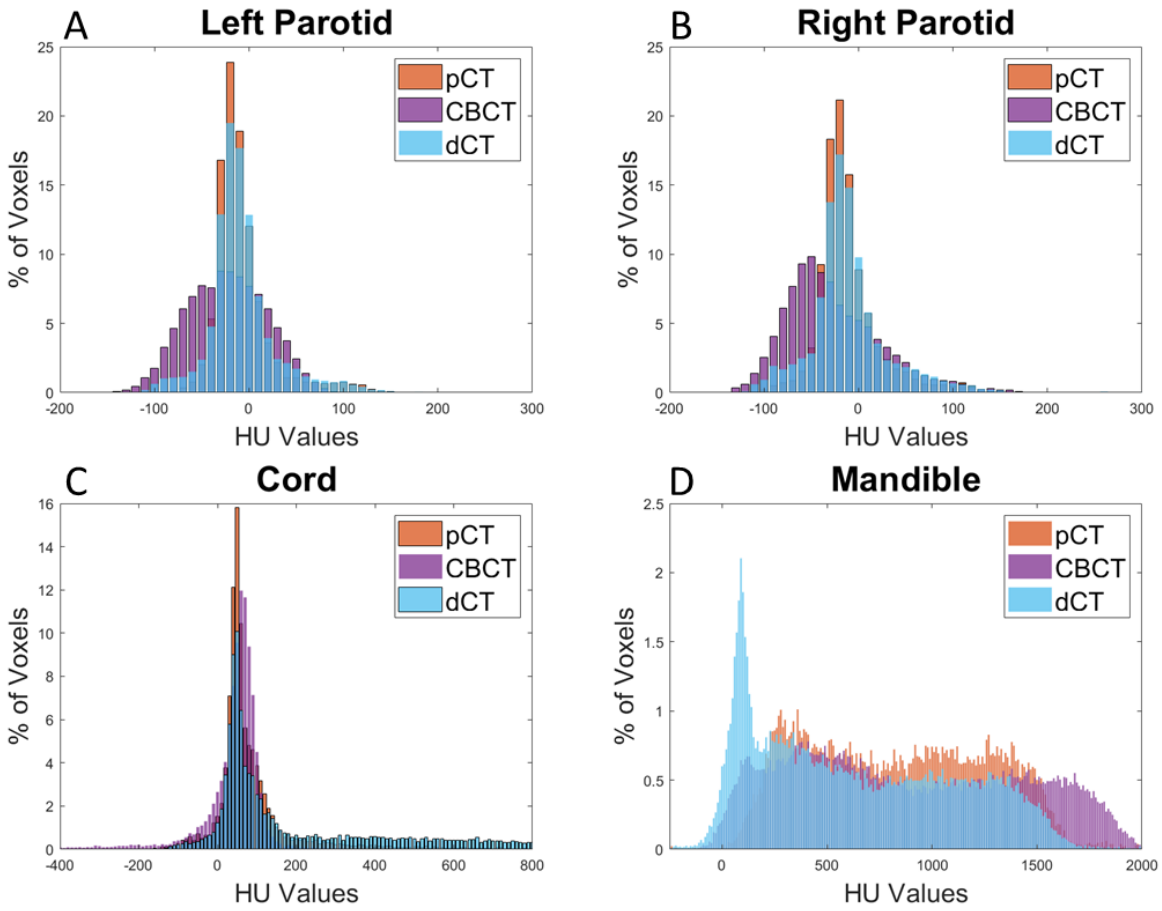


Fig A.15 Histograms of HU values for pCT, CBCT, and dCT for Pt. 8. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

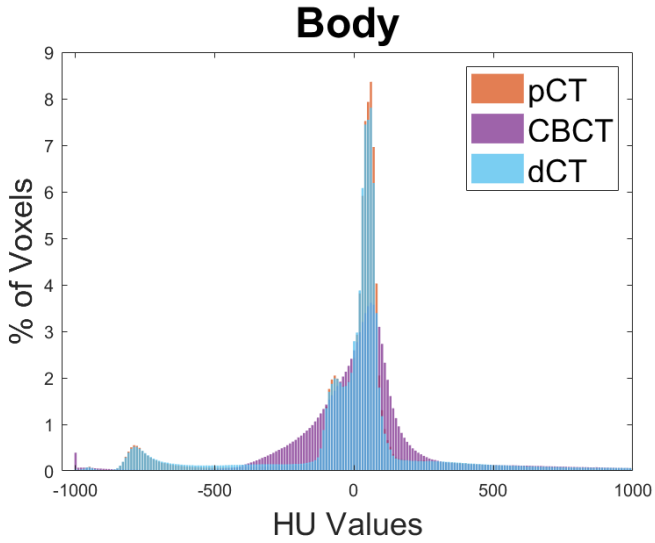


Fig A.16 Histogram of HU values of the body for pCT, CBCT, and dCT for Pt. 9.

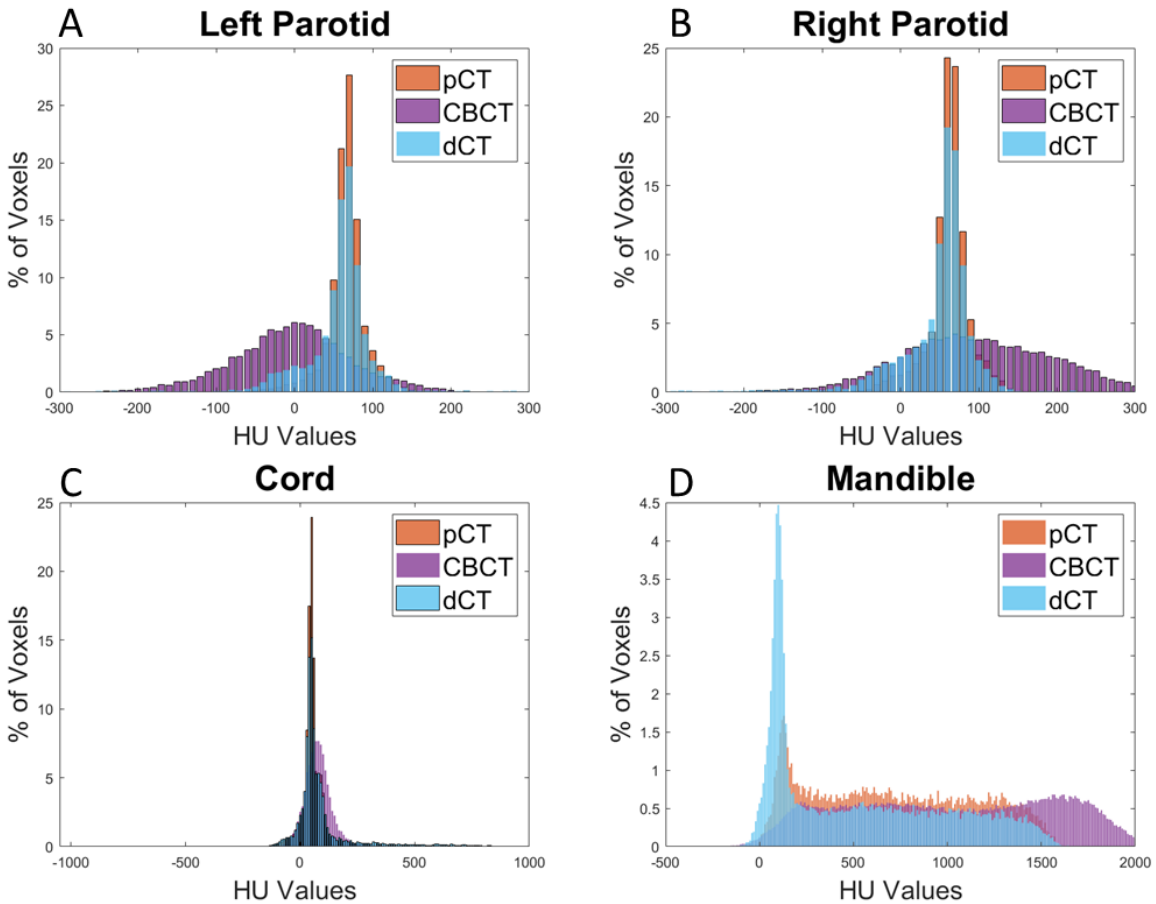


Fig A.17 Histograms of HU values for pCT, CBCT, and dCT for Pt. 9. A: Left parotid, B: Right parotid, C: Cord, D: Mandible.

A.2 OAR DVHs

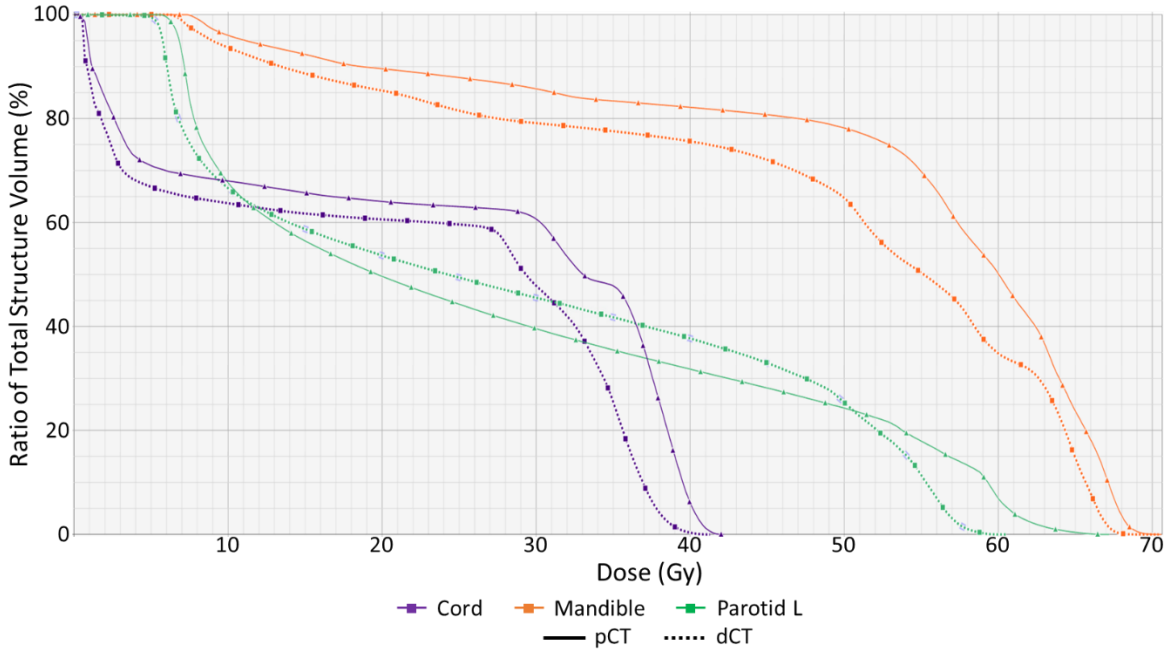


Fig A.18 DVHs for selected OARs for Pt. 1 comparing dose on pCT (solid) to dose on dCT (dashed). Note: Pt. 1 did not have a right parotid.

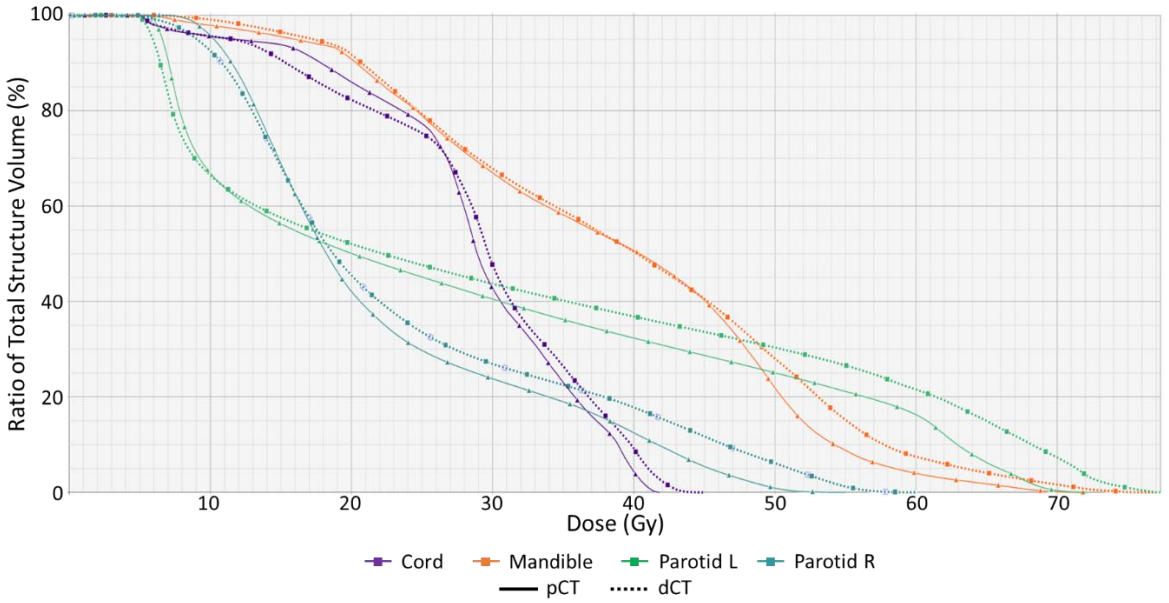


Fig A.19 DVHs for selected OARs for Pt. 3 comparing dose on pCT (solid) to dose on dCT (dashed).

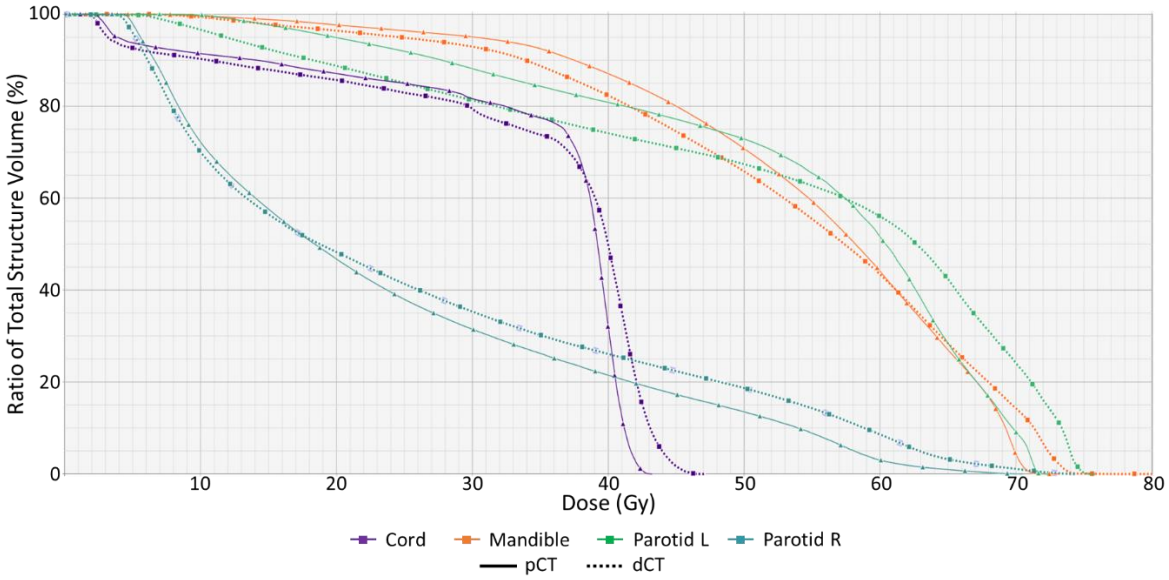


Fig A.20 DVHs for selected OARs for Pt. 4 comparing dose on pCT (solid) to dose on dCT (dashed).

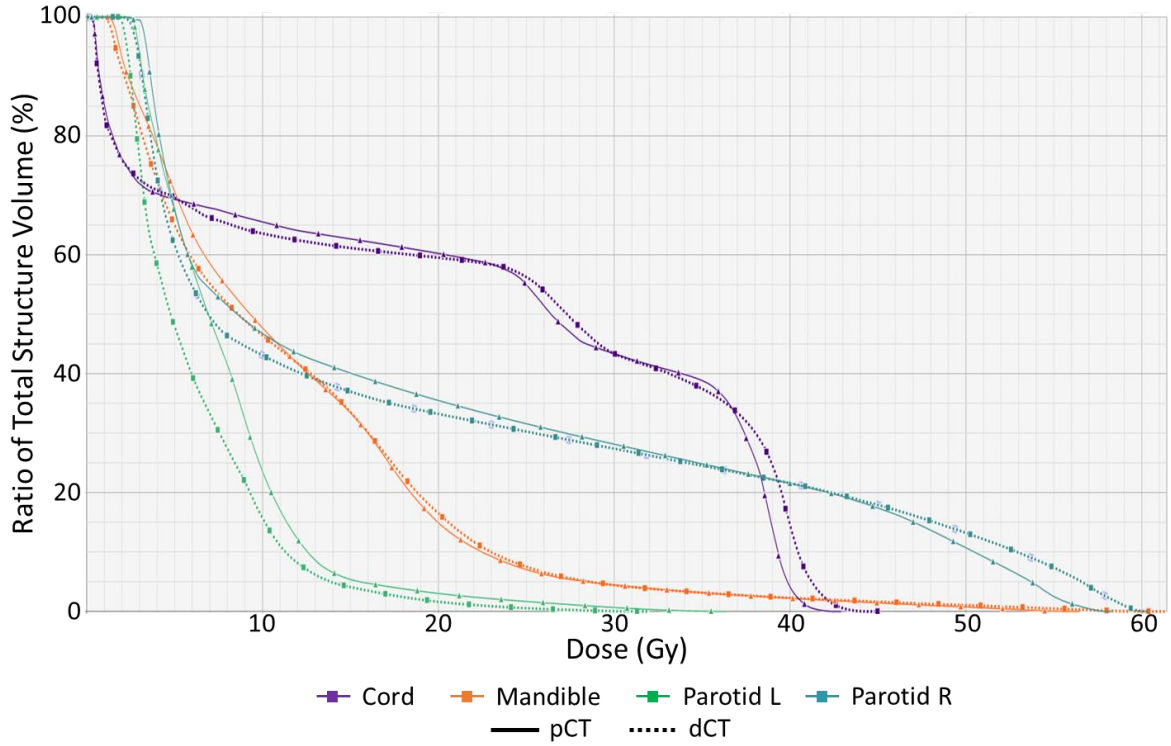


Fig A.21 DVHs for selected OARs for Pt. 5 comparing dose on pCT (solid) to dose on dCT (dashed).

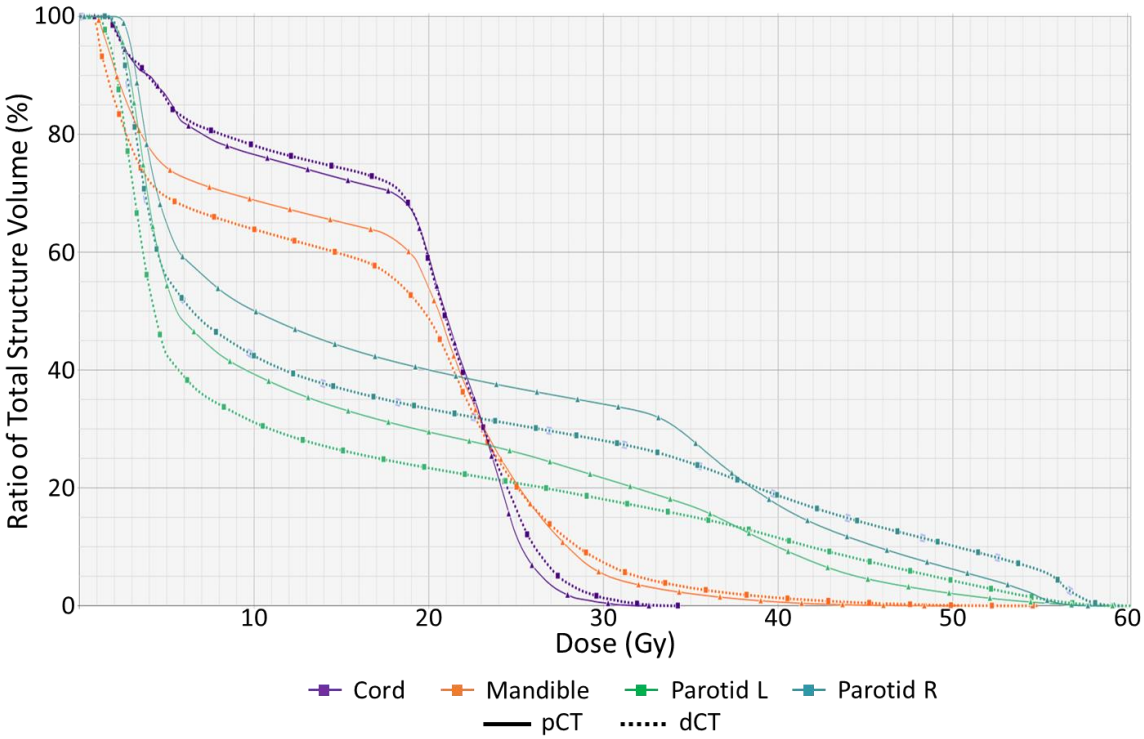


Fig A.22 DVHs for selected OARs for Pt. 6 comparing dose on pCT (solid) to dose on dCT (dashed).

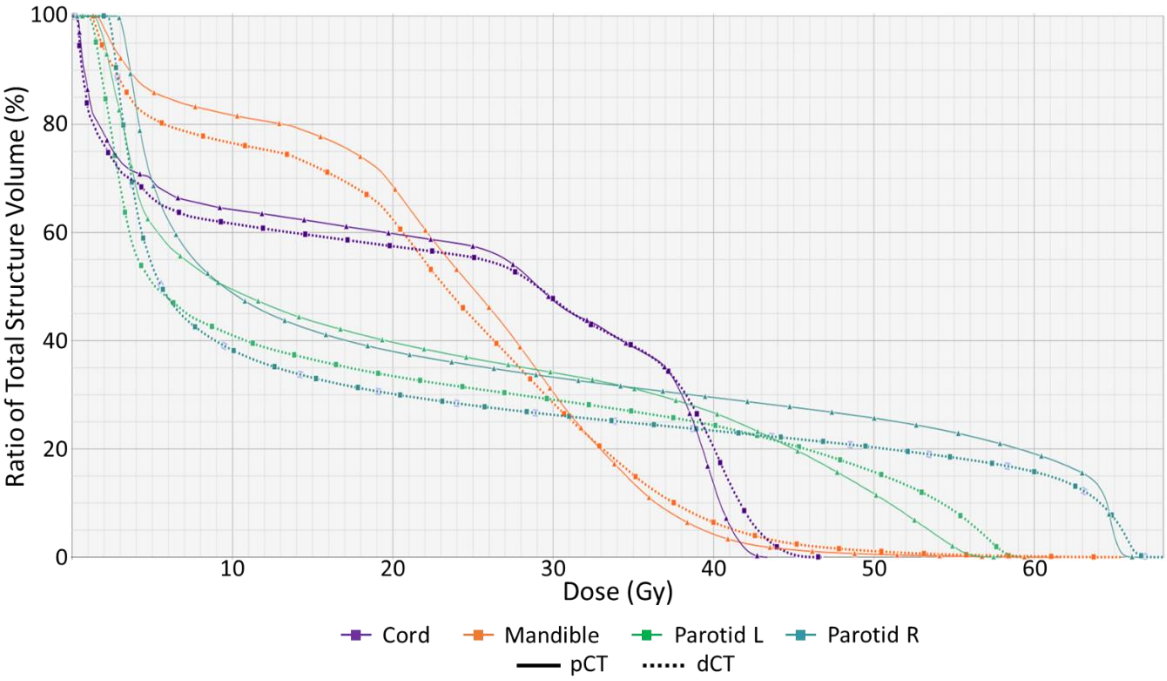


Fig A.23 DVHs for selected OARs for Pt. 7 comparing dose on pCT (solid) to dose on dCT (dashed).

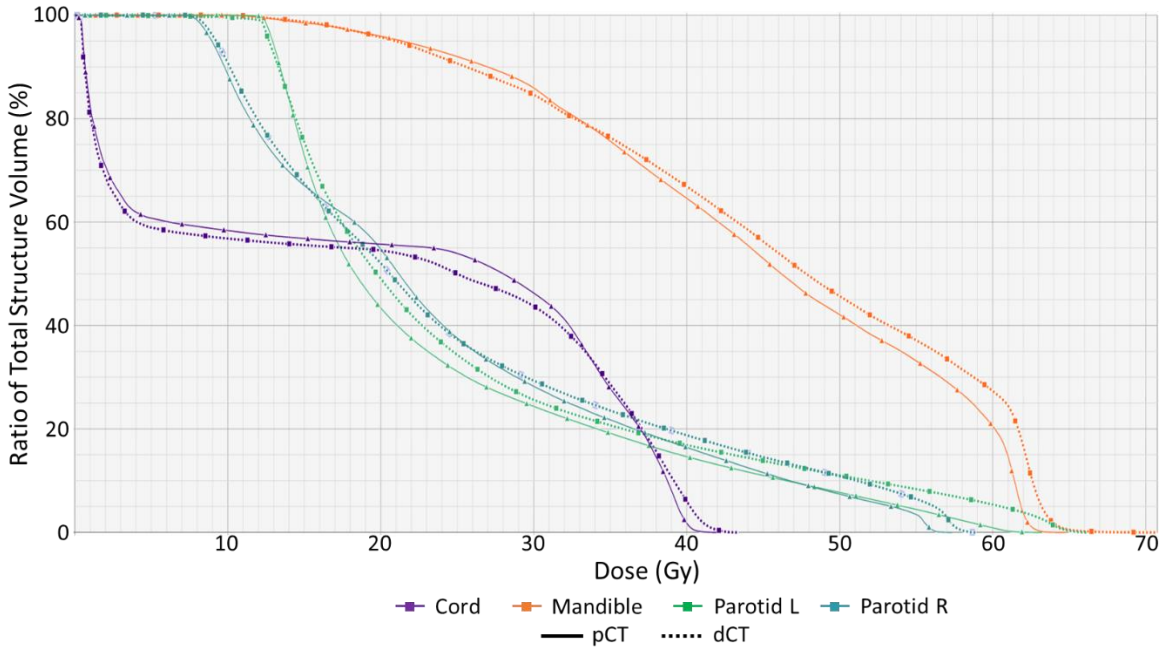


Fig A.24 DVHs for selected OARs for Pt. 8 comparing dose on pCT (solid) to dose on dCT (dashed).

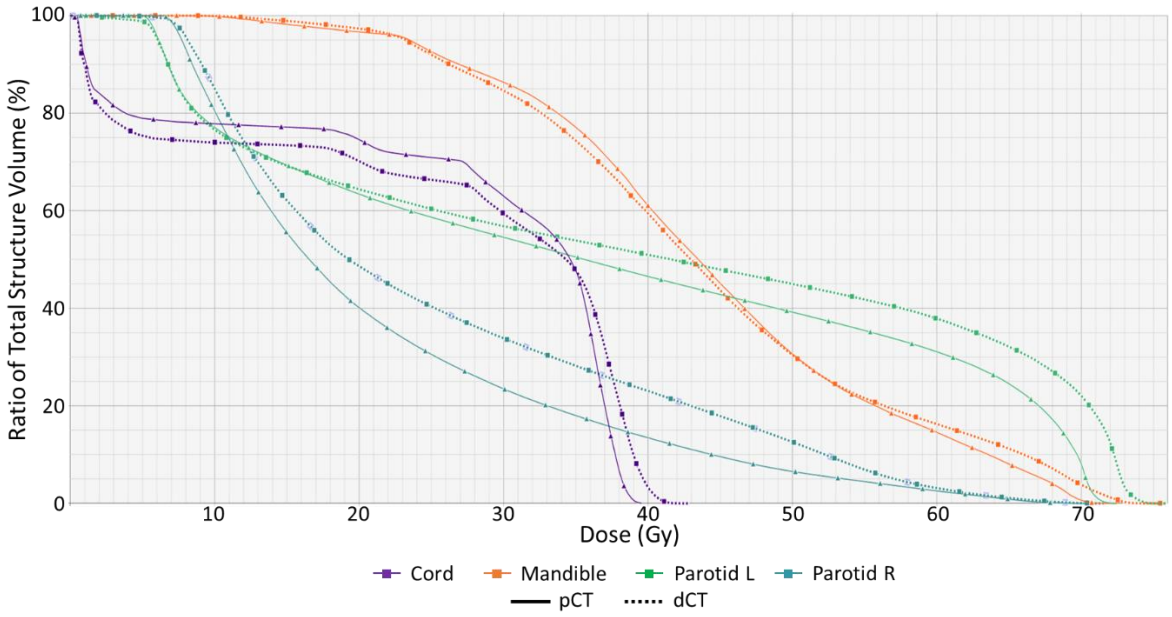


Fig A.25 DVHs for selected OARs for Pt. 9 comparing dose on pCT (solid) to dose on dCT (dashed).

A.3 Neural Network Predictions

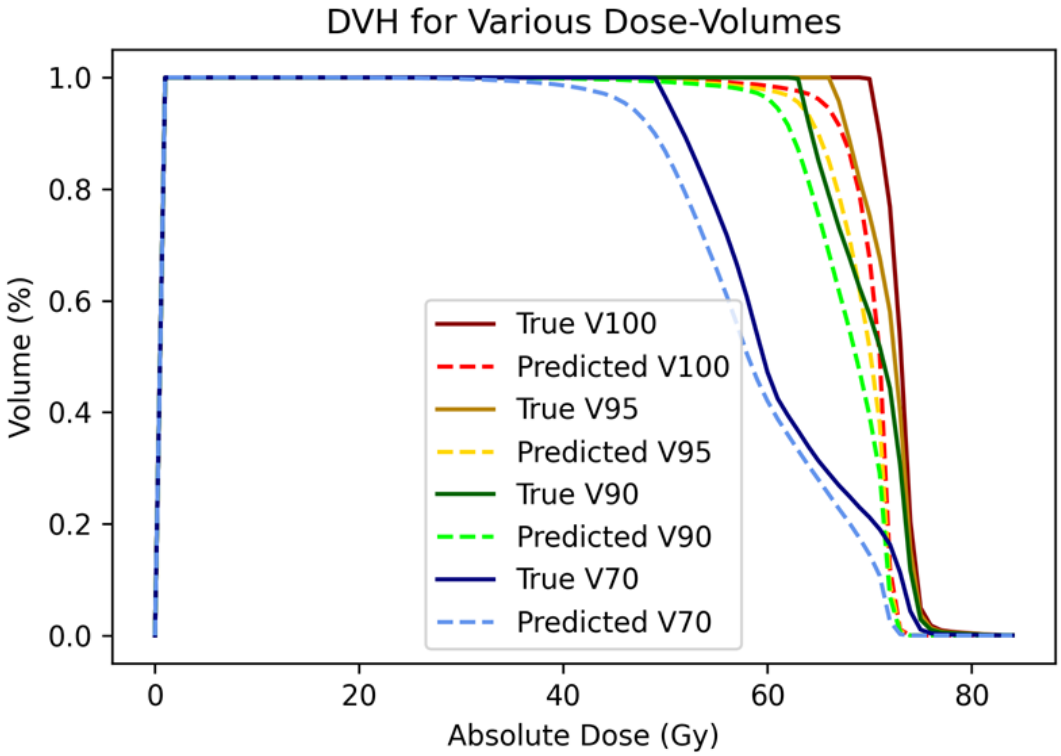


Fig. A.26 DVH for true and predicted isodose volumes for Pt. 1.

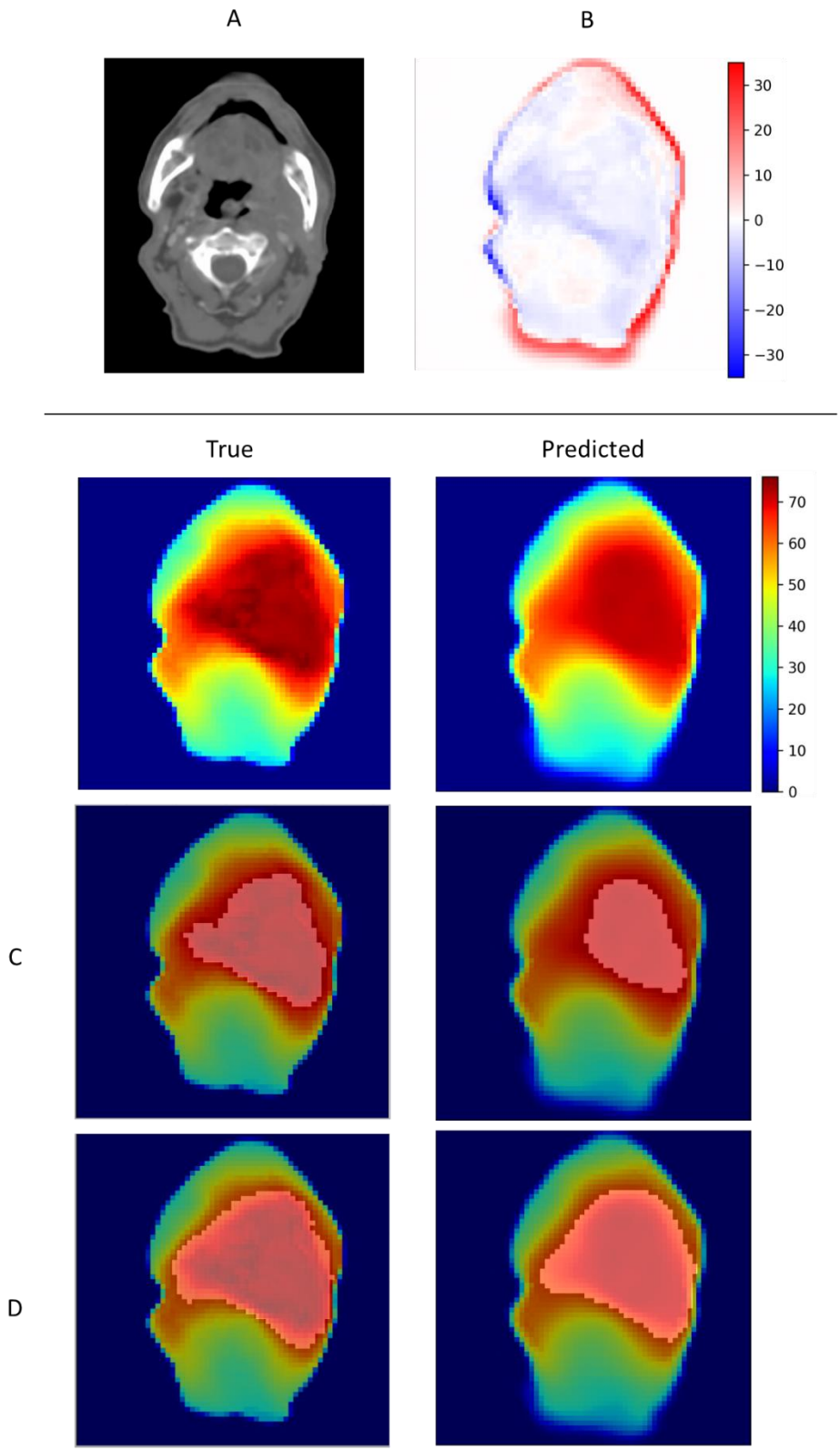


Fig. A.27 Example predictions for Pt. 1. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

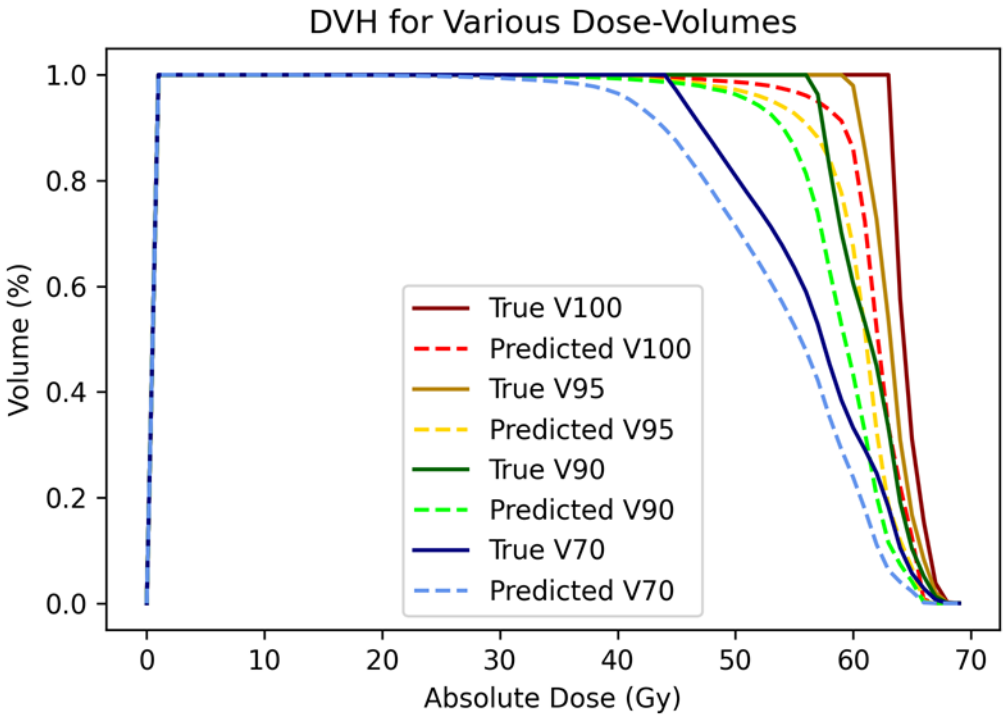


Fig. A.28 DVH for true and predicted isodose volumes for Pt. 2.

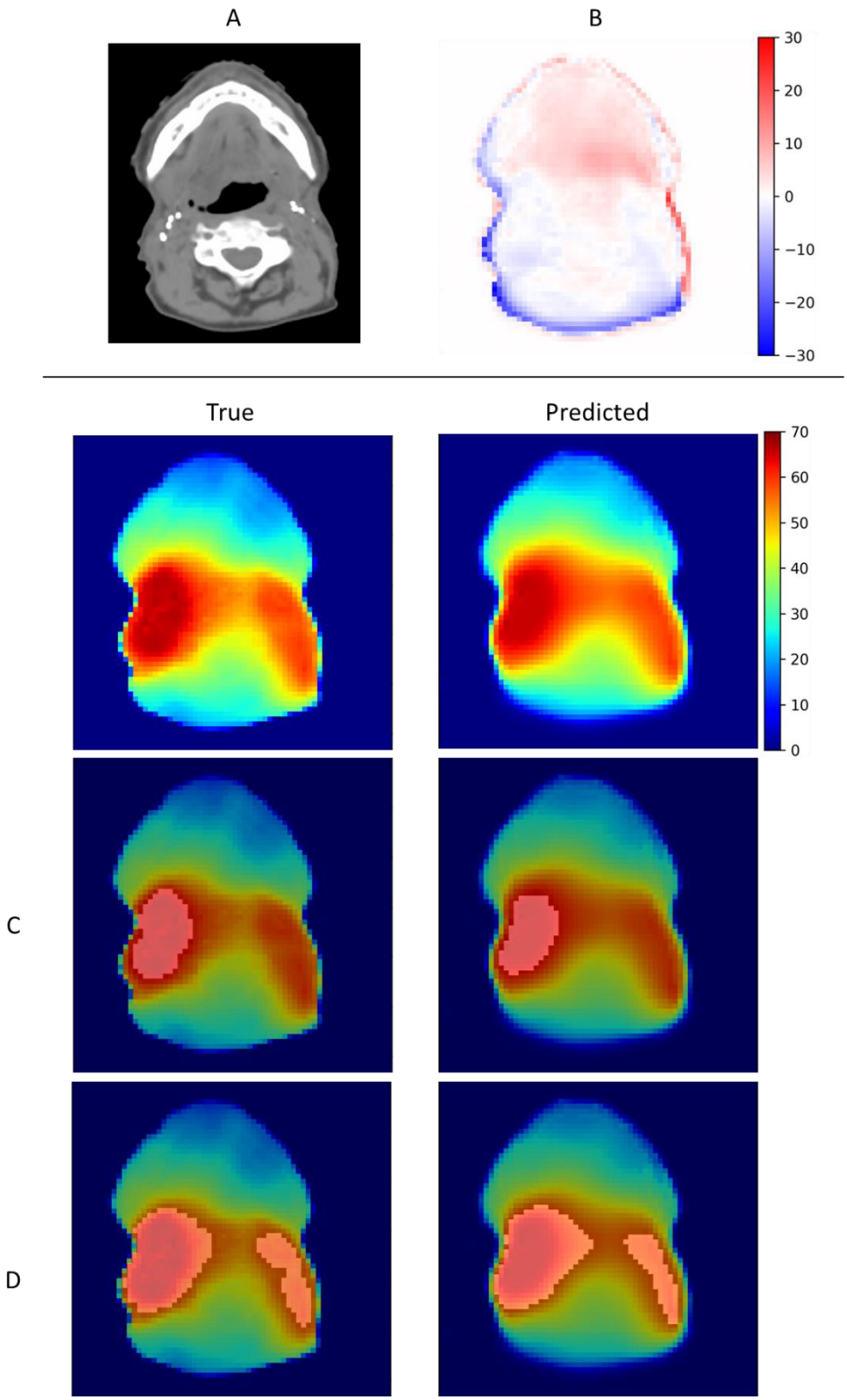


Fig. A.29 Example predictions for Pt. 2. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

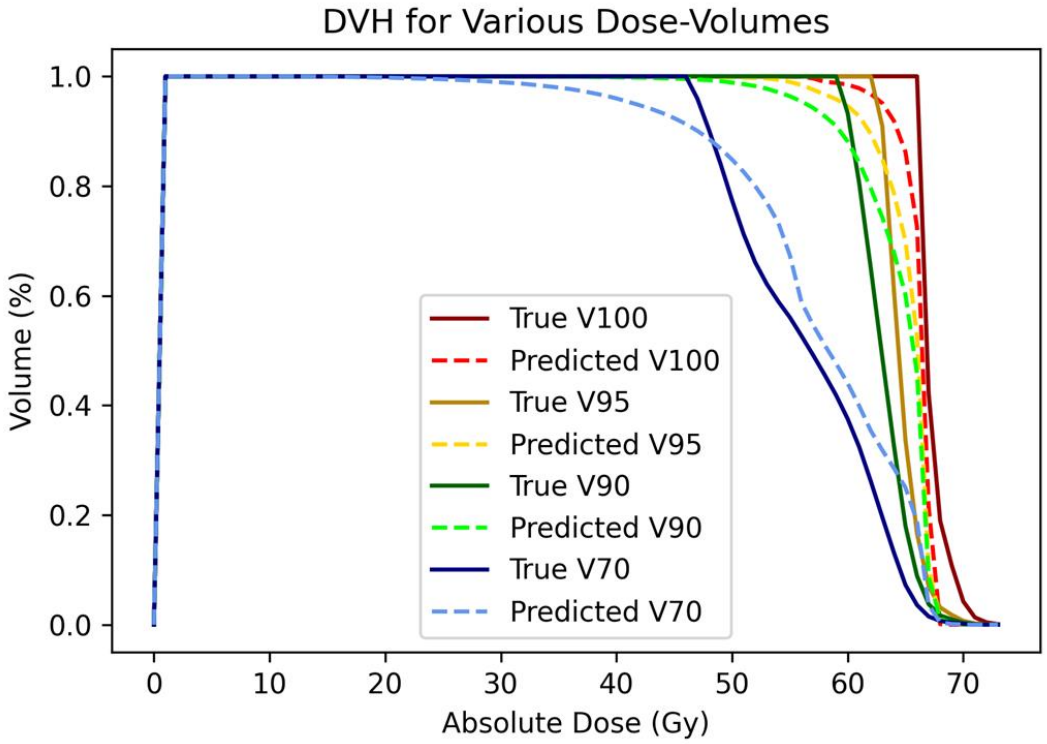


Fig. A.30 DVH for true and predicted isodose volumes for Pt. 3.

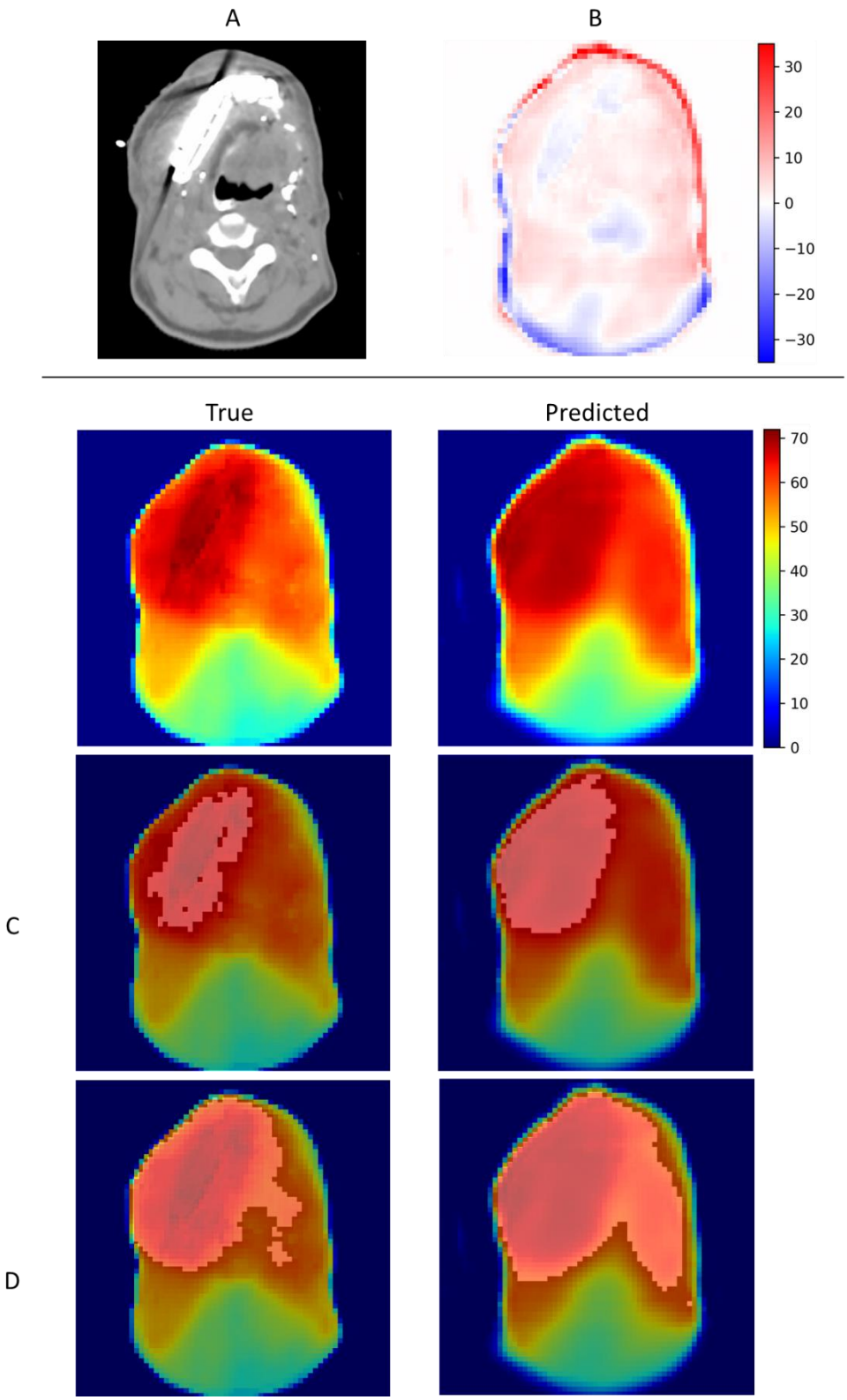


Fig. A.31 Example predictions for Pt. 3. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

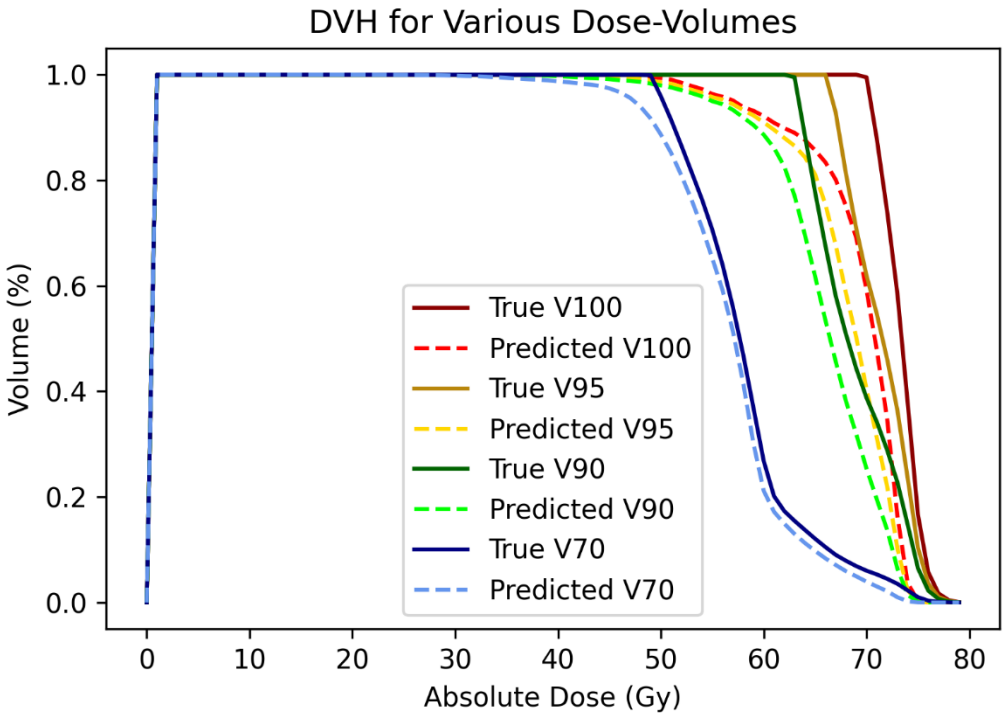


Fig. A.32 DVH for true and predicted isodose volumes for Pt. 5.

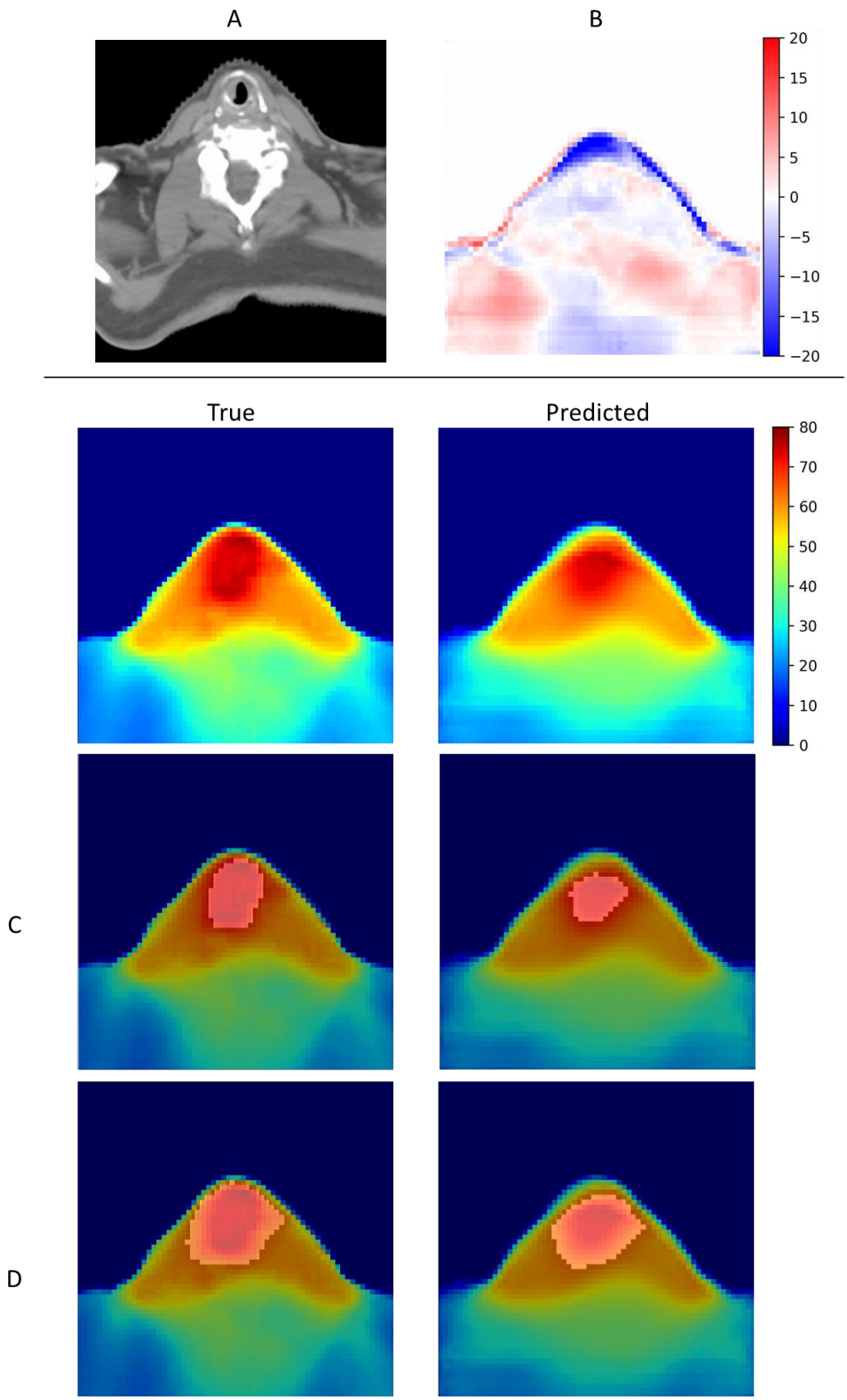


Fig. A.33 Example predictions for Pt. 5. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

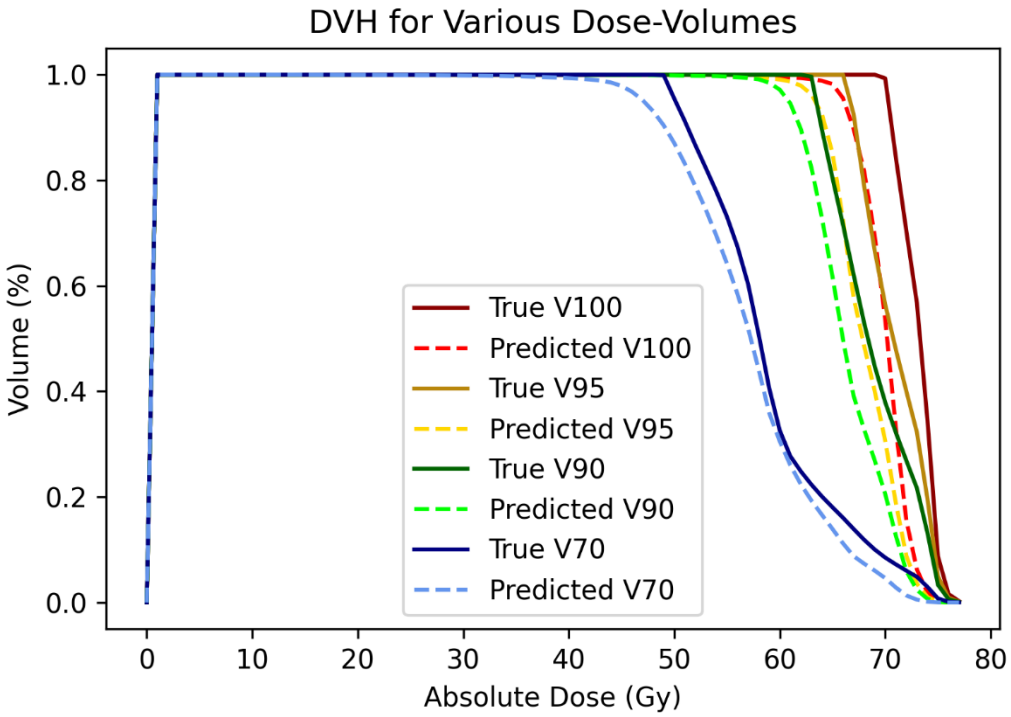


Fig. A.34 DVH for true and predicted isodose volumes for Pt. 6.

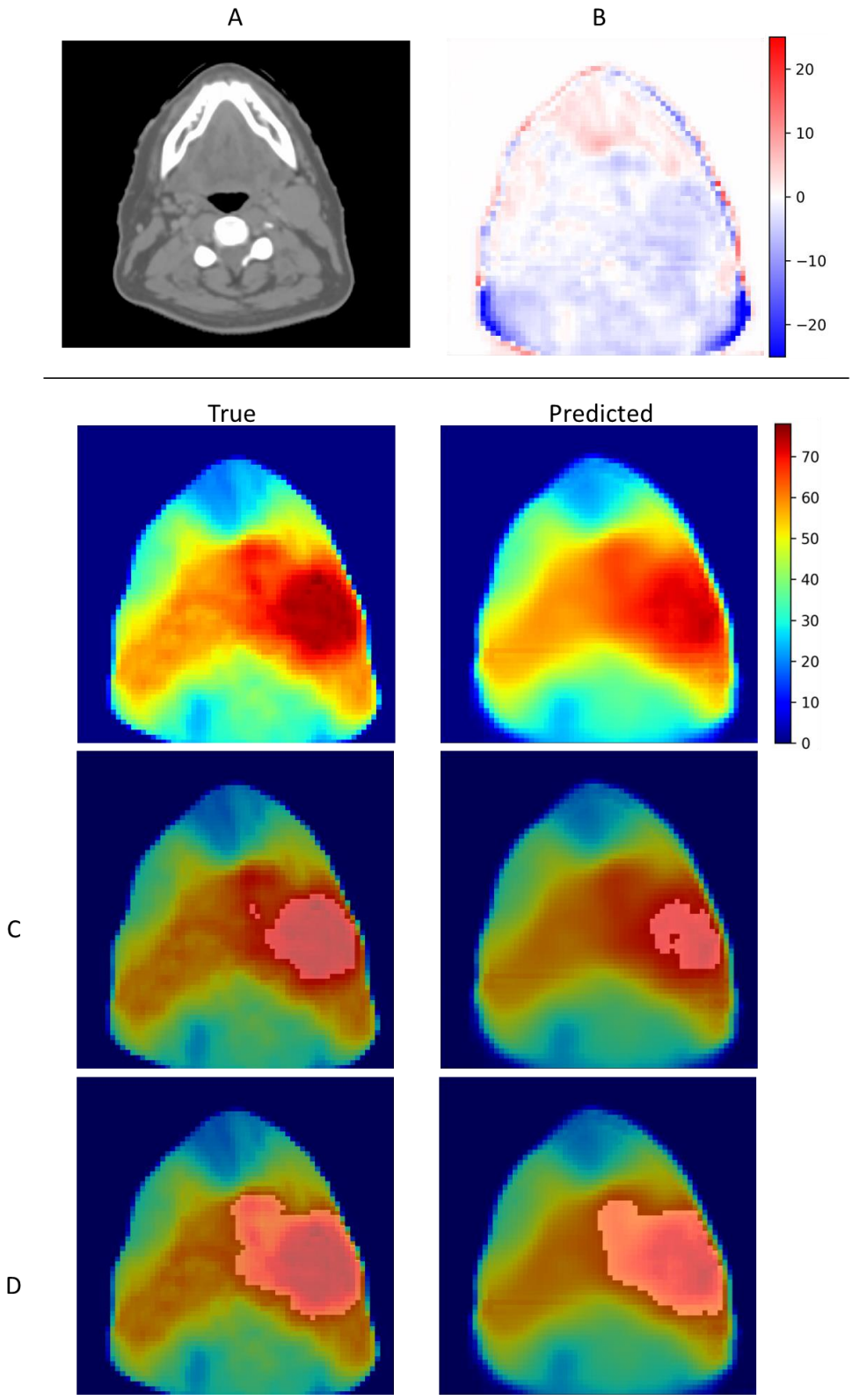


Fig. A.35 Example predictions for Pt. 6. A: Corresponding dCT slice, B: predicted minus true dose difference map (Gy), C: true and predicted V100 overlay, D: true and predicted V90 overlay.

Appendix B Code

This appendix contains code written for this project.

B.1 Image registration parameter files

The following is the parameter file used for rigid registration:

```
// Example parameter file for rotation registration
// C-style comments: //

// The internal pixel type, used for internal computations
// Leave to float in general.
// NB: this is not the type of the input images! The pixel
// type of the input images is automatically read from the
// images themselves.
// This setting can be changed to "short" to save some memory
// in case of very large 3D images.
(FixedInternalImagePixelType "float")
(MovingInternalImagePixelType "float")


// Specify whether you want to take into account the so-called
// direction cosines of the images. Recommended: true.
// In some cases, the direction cosines of the image are corrupt,
// due to image format conversions for example. In that case, you
// may want to set this option to "false".
(UseDirectionCosines "true")

// *************** Main Components **************************

// The following components should usually be left as they are:
(Registration "MultiResolutionRegistration")
(Interpolator "BSplineInterpolator")
(ResampleInterpolator "FinalBSplineInterpolator")
(Resampler "DefaultResampler")

// These may be changed to Fixed/MovingSmoothingImagePyramid.
// See the manual.
(FixedImagePyramid "FixedRecursiveImagePyramid")
(MovingImagePyramid "MovingRecursiveImagePyramid")

// The following components are most important:
// The optimizer AdaptiveStochasticGradientDescent (ASGD) works
// quite ok in general. The Transform and Metric are important
// and need to be chosen careful for each application. See manual.
(Optimizer "AdaptiveStochasticGradientDescent")
(Transform "EulerTransform")
(Metric "AdvancedNormalizedCorrelation")
//(Metric "AdvancedMattesMutualInformation")

//***********Mask**********
(ErodeMask "true")

// ******Similarity Measure*********
```

```
// Number of grey level bins in each resolution level,
// for the mutual information. 16 or 32 usually works fine.
// You could also employ a hierarchical strategy:
//(NumberOfHistogramBins 16 32 32 64)
(NumberOfHistogramBins 32)

// ***************** Transformation **************************

// Scales the rotations compared to the translations, to make
// sure they are in the same range. In general, it's best to
// use automatic scales estimation:
(AutomaticScalesEstimation "true")

// Automatically guess an initial translation by aligning the
// geometric centers of the fixed and moving.
(AutomaticTransformInitialization "true")

// Whether transforms are combined by composition or by addition.
// In generally, Compose is the best option in most cases.
// It does not influence the results very much.
(HowToCombineTransforms "Compose")

// ******************** Multiresolution **********************

// The number of resolutions. 1 Is only enough if the expected
// deformations are small. 3 or 4 mostly works fine. For large
// images and large deformations, 5 or 6 may even be useful.
(NumberOfResolutions 4)

// The downsampling/blurring factors for the image pyramids.
// By default, the images are downsampled by a factor of 2
// compared to the next resolution.
// So, in 2D, with 4 resolutions, the following schedule is used:
//(ImagePyramidSchedule 8 8  4 4  2 2  1 1 )
// And in 3D:
//(ImagePyramidSchedule 8 8 8  4 4 4  2 2 2  1 1 1 )
// You can specify any schedule, for example:
//(ImagePyramidSchedule 4 4  4 3  2 1  1 1 )
// Make sure that the number of elements equals the number
// of resolutions times the image dimension.

// ****************** Optimizer ****************************

// Maximum number of iterations in each resolution level:
// 200-500 works usually fine for rigid registration.
// For more robustness, you may increase this to 1000-2000.
(MaximumNumberOfIterations 300)

// The step size of the optimizer, in mm. By default the voxel size is
used.
// which usually works well. In case of unusual high-resolution images
// (eg histology) it is necessary to increase this value a bit, to the
size
// of the "smallest visible structure" in the image:
```

```
//(MaximumStepLength 1.0)

// **************** Image sampling **********************

// Number of spatial samples used to compute the mutual
// information (and its derivative) in each iteration.
// With an AdaptiveStochasticGradientDescent optimizer,
// in combination with the two options below, around 2000
// samples may already suffice.
(NumberOfSpatialSamples 3000)

// Refresh these spatial samples in every iteration, and select
// them randomly. See the manual for information on other sampling
// strategies.
(NewSamplesEveryIteration "true")
(ImageSampler "RandomCoordinate")

// ************* Interpolation and Resampling ***************

// Order of B-Spline interpolation used during registration/optimisation.
// It may improve accuracy if you set this to 3. Never use 0.
// An order of 1 gives linear interpolation. This is in most
// applications a good choice.
(BSplineInterpolationOrder 1)

// Order of B-Spline interpolation used for applying the final
// deformation.
// 3 gives good accuracy; recommended in most cases.
// 1 gives worse accuracy (linear interpolation)
// 0 gives worst accuracy, but is appropriate for binary images
// (masks, segmentations); equivalent to nearest neighbor interpolation.
(FinalBSplineInterpolationOrder 3)

//Default pixel value for pixels that come from outside the picture:
(DefaultPixelValue -1000)

// Choose whether to generate the deformed moving image.
// You can save some time by setting this to false, if you are
// only interested in the final (nonrigidly) deformed moving image
// for example.
//(WriteResultImageAfterEachResolution "true")
(WriteResultImage "true")
//(ShowExactMetricValue "true")

// The pixel type and format of the resulting deformed moving image
(ResultImagePixelType "short")
(ResultImageFormat "mhd")
```

The following is the parameter file used for deformable registration:

```
(FixedInternalImagePixelType "float")
(MovingInternalImagePixelType "float")
(FixedImageDimension 3)
(MovingImageDimension 3)
(UseDirectionCosines "true")

// *************** Main Components **************************

(Registration "MultiMetricMultiResolutionRegistration")
//(Registration "MultiResolutionRegistration")
(Interpolator "BSplineInterpolator")
(ResampleInterpolator "FinalBSplineInterpolator")
(Resampler "DefaultResampler")

//(FixedImagePyramid "FixedSmoothingImagePyramid")
//(MovingImagePyramid "MovingSmoothingImagePyramid")
(FixedImagePyramid "FixedRecursiveImagePyramid")
(MovingImagePyramid "MovingRecursiveImagePyramid")


(Optimizer "AdaptiveStochasticGradientDescent")
(Transform "BSplineTransform")
(Metric "AdvancedMattesMutualInformation" "TransformBendingEnergyPenalty")
//(Metric "AdvancedMattesMutualInformation")
(Metric0Weight 1)
(Metric1Weight 25) // OPTIMIZED BY QUANTITATIVE MEASURES

//********Mask*******
(ErodeMask "true")

// ***************** Transformation **************************

//(GridSpacingSchedule 4 2 1)
//(FinalGridSpacingInPhysicalUnits 15)
(HowToCombineTransforms "Compose")

// ****************** Similarity measure *********************

(NumberOfHistogramBins 32) // OPTIMIZED BY QUANTITATIVE MEASURES

// ******************** Multiresolution **********************

(NumberOfResolutions 3)
(ImagePyramidSchedule 8 8 2  4 4 1  1 1 1) // ACCOUNTING FOR ANISOTROPIC
RESOLUTION

// ****************** Optimizer ****************************

(MaximumNumberOfIterations 2500) // COULD PROBABLY BE LOWERED, ESPECIALLY FOR
THE FIRST LEVELS OF THE IMAGE PYRAMID
(ASGDParameterEstimationMethod "DisplacementDistribution")

// *************** Image sampling **********************
```

```
(NumberOfSpatialSamples 3000) // COULD PROBABLY BE LOWERED, ESPECIALLY FOR
THE FIRST LEVELS OF THE IMAGE PYRAMID
(NewSamplesEveryIteration "true")
(ImageSampler "RandomCoordinate")
//(SampleRegionSize 40)  // OPTIMIZED BY QUANTITATIVE MEASURES
//(UseRandomSampleRegion "true")
//(MaximumNumberOfSamplingAttempts 5)
//(RequiredRatioOfValidSamples 0.05)

// ************* Interpolation and Resampling ****************

(BSplineInterpolationOrder 1)
(FinalBSplineInterpolationOrder 3)

(ShowExactMetricValue "false")
(WriteTransformParametersEachResolution "true")
//(WriteResultImageAfterEachResolution "true")
//(WritePyramidImagesAfterEachResolution "true")

(DefaultPixelValue -1000)
(WriteResultImage "true")
(ResultImagePixelType "short")
(ResultImageFormat "mhd")
```

B.2 MATLAB Code

The following are scripts used for processing the images before and after image registration, as well as the script for contour comparison.

B.2.1 Preparing for registration

This script creates the resampled and padded CBCT image and the mask based on the input CBCT and pCT.

```
function[createImages] = createCBCTandMask(CBCTimage, Simimage, CBCTbkg, Maskbkg,
ResampledName, MaskName, PadName, MaskPadName)

RescaleMetaImage(CBCTimage, Simimage, ResampledName);

[CBimg, CBhdr] = OpenMetaImageSH(strcat(ResampledName, '.mhd'));
mask = CylinderMask(CBimg);
WriteMetaImageSH(MaskName, mask, CBhdr);

PadMetaImage(strcat(ResampledName, '.mhd'), Simimage, CBCTbkg, PadName);

PadMetaImage(strcat(MaskName, '.mhd'), Simimage, Maskbkg, MaskPadName);

end
```

This script resamples the CBCT to the same resolution as the pCT.

```
function[resizedImg] = RescaleMetaImage(ImgPath, SimPath, ResizedImgName)

[CBimg, CBhdr] = OpenMetaImageSH(ImgPath);
[Simimg, Simhdr] = OpenMetaImageSH(SimPath);

newx = round(size(CBimg,1)*CBhdr.x_pixdim/Simhdr.x_pixdim);
newy = round(size(CBimg,2)*CBhdr.y_pixdim/Simhdr.y_pixdim);
newz = round(size(CBimg,3)*CBhdr.z_pixdim/Simhdr.z_pixdim);

resizedImg = imresize3(CBimg, [newx, newy, newz], 'linear');

CBhdr.x_pixdim = Simhdr.x_pixdim;
CBhdr.y_pixdim = Simhdr.y_pixdim;
CBhdr.z_pixdim = Simhdr.z_pixdim;

CBhdr.x_dim = newx;
CBhdr.y_dim = newy;
CBhdr.z_dim = newz;

WriteMetaImageSH( ResizedImgName, resizedImg, CBhdr);

end
```

This script pads the resampled CBCT to approximately the same array size as the pCT.

```
function[paddedImg] = PadMetaImage(ImgPath, SimPath, BackgroundNumber, PaddedImgName)

[CBimg, CBhdr] = OpenMetaImageSH(ImgPath);
[Simimg, Simhdr] = OpenMetaImageSH(SimPath);

paddedImg = BackgroundNumber*ones(size(Simimg));

CBx = size(CBimg, 1);
CBy = size(CBimg, 2);
CBz = size(CBimg, 3);

xdiff = size(Simimg,1) - size(CBimg,1);
ydiff = size(Simimg,2) - size(CBimg,2);
zdiff = size(Simimg,3) - size(CBimg,3);

if mod(xdiff, 2) == 1  %is odd
    paddedImg = paddedImg(1:end-1,1:end-1,:);
    xdiff = xdiff - 1;
    ydiff = ydiff - 1;  %should separate out y from x in case of non-square array
end

if mod(zdiff, 2) == 1  %is odd
    paddedImg = paddedImg(:,:,1:end-1);
    zdiff = zdiff - 1;
end

paddedImg(xdiff/2+1:(CBx+xdiff/2), ydiff/2+1:(CBy+ydiff/2), zdiff/2+1:(CBz+zdiff/2)) =
CBimg(:,:,:);

CBhdr.x_dim = size(paddedImg,1);
CBhdr.y_dim = size(paddedImg,2);
```

```
CBhdr.z_dim = size(paddedImg,3);

WriteMetaImageSH( PaddedImgName, paddedImg, CBhdr);

end
```

This script creates the cylindrical mask for the FOV of the CBCT.

```
function[mask] = CylinderMask(img)

[x,y,z] =size(img);
mask = zeros(x,y,z);
for i = 4:(z-3);  %shortened mask on 05/11/20 from 2:(z-1) to avoid truncation effects
    imshow(img(:,:,i))
%    if i == 1 | i == z
%        r = (x-72)/2;
%        circ = drawcircle('Center',[x/2,y/2],'Radius',r);
%    else
    circ = drawcircle('Center',[x/2,y/2],'Radius',x/2);
%    end
    mask(:,:,i) = createMask(circ);
end

end
```

This script resamples the cylindrical mask to the resolution of the pCT.

```
function[resizedImg] = RescaleMetaImage(ImgPath, SimPath, ResizedImgName)

[CBimg, CBhdr] = OpenMetaImageSH(ImgPath);
[Simimg, Simhdr] = OpenMetaImageSH(SimPath);

newx = round(size(CBimg,1)*CBhdr.x_pixdim/Simhdr.x_pixdim);
newy = round(size(CBimg,2)*CBhdr.y_pixdim/Simhdr.y_pixdim);
newz = round(size(CBimg,3)*CBhdr.z_pixdim/Simhdr.z_pixdim);

resizedImg = imresize3(CBimg, [newx, newy, newz], 'nearest');

CBhdr.x_pixdim = Simhdr.x_pixdim;
CBhdr.y_pixdim = Simhdr.y_pixdim;
CBhdr.z_pixdim = Simhdr.z_pixdim;

CBhdr.x_dim = newx;
CBhdr.y_dim = newy;
CBhdr.z_dim = newz;

WriteMetaImageSH( ResizedImgName, resizedImg, CBhdr);

end
```

B.2.2 Additional MATLAB Scripts

This script converts the dCT from mhd format to DICOM format for import to TPS, using info from the pCT DICOM tags.

```
function[newDicom] = convertMHDtoDICOM(dCT, pCTdicom, CBCTdate, CBCTdateDec,
dicomFolder, dicomName)

%dCT 'filename.mhd'
%pCTdicom 'pathname/slicename.dcm'
%CBCTdate 'yyyymmdd'
%CBCTdateDec 'mm.dd.yyyy'
%dicomFolder - name of dicom folder to be created
%ex:'ANON11111_CBCT#_DICOM'
%dicomName - name of dicom slices ex:'ANON11111_mmddyyyy_CBCT_'

[dCTimg, dCThdr] = OpenMetaImageSH(dCT);
pCTinfo = dicominfo(pCTdicom);

studyuid = dicomuid;
seriesuid = dicomuid;
instanceuid = dicomuid;
refuid = dicomuid;

origPos = pCTinfo.ImagePositionPatient;
origSlice = origPos(3);

pCTinfo.FileModDate = string(datetime('now'));
pCTinfo.FileSize = '';
pCTinfo.Width = dCThdr.x_dim;
pCTinfo.Height = dCThdr.y_dim;
pCTinfo.MediaStorageSOPInstanceUID = instanceuid;
pCTinfo.ImplementationVersionName = '';
pCTinfo.SpecificCharacterSet = '';
pCTinfo.InstanceCreationDate = '';
pCTinfo.InstanceCreationTime = '';
pCTinfo.SOPInstanceUID = instanceuid;
pCTinfo.StudyDate = CBCTdate;
pCTinfo.SeriesDate = CBCTdate;
pCTinfo.AcquisitionDate = CBCTdate;
pCTinfo.ContentDate = CBCTdate;
pCTinfo.SeriesDescription = strcat('CT_HN.', CBCTdateDec);
pCTinfo.StudyInstanceUID = studyuid;
pCTinfo.SeriesInstanceUID = seriesuid;
pCTinfo.StudyID = strcat('DIR ', CBCTdate);
pCTinfo.SeriesNumber = [];
pCTinfo.FrameOfReferenceUID = refuid;
pCTinfo.Rows = dCThdr.x_dim;
pCTinfo.Columns = dCThdr.y_dim;
pCTinfo.RescaleIntercept = 0;
pCTinfo.RescaleSlope = 1;

mkdir(dicomFolder);

for i = 1:size(dCTimg, 3)
    filename = strcat(dicomName, sprintf('%04d', i));
    pCTinfo.Filename = filename;
    pCTinfo.InstanceNumber = i;
    pCTinfo.ImagePositionPatient = origPos - [0; 0; pCTinfo.SliceThickness*(i-1)];
    pCTinfo.SliceLocation = origSlice - pCTinfo.SliceThickness*(i-1);
    dicomwrite(dCTimg(:,:,i), strcat(dicomFolder, '\', filename, '.dcm'), pCTinfo);
end
```

```
end
```

This script calculates DSC, JI, and HD between two contours.

```
function[compareContours] = contourCompare(BQContourPath, simPath,
resampleContourPath, padContourPath, defContourPath)

%Rescale and pad CBCT contour
RescaleMetaImage(BQContourPath, simPath, resampleContourPath);
PadMetaImage(strcat(resampleContourPath, '.mhd'), simPath, 0, padContourPath);

%Open both masks
maskBQ = OpenMetaImageSH(strcat(padContourPath, '.mhd'));
maskDef = OpenMetaImageSH(defContourPath);

%Binarize both masks
bmaskBQ = imbinarize(maskBQ);
bmaskDef = logical(maskDef);

%Find extent of CBCT mask
slices = size(bmaskBQ, 3);
ind_vector = zeros(slices);

for i = 1:slices
    slice = bmaskBQ(:,:,i);
    if max(slice(:)) == 1
        ind_vector(i) = 1;
    end
end

index1 = find(ind_vector, 1);
index2 = find(ind_vector, 1, 'last');
len = index2-index1 + 1;

dsc = dice(bmaskBQ(:,:,index1:index2), bmaskDef(:,:,index1:index2));
hdarray = zeros(1,len);
jac = jaccard(bmaskBQ(:,:,index1:index2), bmaskDef(:,:,index1:index2));
j = 1;
for i = index1:index2
    [hd D] = HausdorffDist(bmaskBQ(:,:,i), bmaskDef(:,:,i));
    hdarray(j) = hd;
    j = j + 1;
end

hdavg = max(hdarray(:));
disp(['The DSC is ', num2str(dsc), '. The HD is ', num2str(hdavg), '. The Jaccard
Index is ', num2str(jac), '.'])

end
```

B.3 Python Code

The following code includes programs to process data for input to the neural network, the neural network itself, and scripts for analysis of predictions.

B.3.1 Pre-processing

The following code performs conversion of surface obj files and dose files, resamples all arrays to isotropic resolution, and aligns and crops all arrays.

```
def convertSurfandDose(ptid, ptnum):
  surf0Path = '/content/drive/MyDrive/Colab/HNSI/Source/SimSurface/' + pti
d + '/Sim/capture.obj'
  surf1Path = '/content/drive/MyDrive/Colab/HNSI/Source/CBCTSurface/' + pt
id + '/CBCT1/capture.obj'
  surf2Path = '/content/drive/MyDrive/Colab/HNSI/Source/CBCTSurface/' + pt
id + '/CBCT2/capture.obj'
  surf3Path = '/content/drive/MyDrive/Colab/HNSI/Source/CBCTSurface/' + pt
id + '/CBCT3/capture.obj'
  dose0Path = '/content/drive/MyDrive/Colab/HNSI/Source/SimDose/' + ptid +
 '/Sim/RD.' + ptid + '.ANON.dcm'
  dose1Path = '/content/drive/MyDrive/Colab/HNSI/Target/CBCTDose/' + ptid
+ '/CBCT1/RD.' + ptid + '.CBCT1.dcm'
  dose2Path = '/content/drive/MyDrive/Colab/HNSI/Target/CBCTDose/' + ptid
+ '/CBCT2/RD.' + ptid + '.CBCT2.dcm'
  dose3Path = '/content/drive/MyDrive/Colab/HNSI/Target/CBCTDose/' + ptid
+ '/CBCT3/RD.' + ptid + '.CBCT3.dcm'
  surf0nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Surface/0/' +
ptnum + '.nrrd'
  surf1nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Surface/1/' +
ptnum + '.nrrd'
  surf2nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Surface/2/' +
ptnum + '.nrrd'
  surf3nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Surface/3/' +
ptnum + '.nrrd'
  dose0nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/0/' + ptn
um + '.nrrd'
  dose1nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/1/' + ptn
um + '.nrrd'
  dose2nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/2/' + ptn
um + '.nrrd'
  dose3nrrd = '/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/3/' + ptn
um + '.nrrd'
  #convert all the surfaces to xarrays
  surf0 = convertOBJ(surf0Path)
  surf1 = convertOBJ(surf1Path)
  surf2 = convertOBJ(surf2Path)
```

```
  surf3 = convertOBJ(surf3Path)
  #import dose distributions and obtain xarray, numpy array, and coords
  xdose0, d0, d0xcoords, d0ycoords, d0zcoords = importDose(dose0Path)
  xdose1, d1, d1xcoords, d1ycoords, d1zcoords = importDose(dose1Path)
  xdose2, d2, d2xcoords, d2ycoords, d2zcoords = importDose(dose2Path)
  xdose3, d3, d3xcoords, d3ycoords, d3zcoords = importDose(dose3Path)

  #separate numpy array from coords of surfaces
  nsurf0, s0xcoords, s0ycoords, s0zcoords = xyz_and_numpy_from_xarray(surf
0)
  nsurf1, s1xcoords, s1ycoords, s1zcoords = xyz_and_numpy_from_xarray(surf
1)
  nsurf2, s2xcoords, s2ycoords, s2zcoords = xyz_and_numpy_from_xarray(surf
2)
  nsurf3, s3xcoords, s3ycoords, s3zcoords = xyz_and_numpy_from_xarray(surf
3)

  #get resolutions of surfaces and dose
  s0xres, s0yres, s0zres = getxyzRes(s0xcoords, s0ycoords, s0zcoords)
  s1xres, s1yres, s1zres = getxyzRes(s1xcoords, s1ycoords, s1zcoords)
  s2xres, s2yres, s2zres = getxyzRes(s2xcoords, s2ycoords, s2zcoords)
  s3xres, s3yres, s3zres = getxyzRes(s3xcoords, s3ycoords, s3zcoords)
  d0xres, d0yres, d0zres = getxyzRes(d0xcoords, d0ycoords, d0zcoords)
  d1xres, d1yres, d1zres = getxyzRes(d1xcoords, d1ycoords, d1zcoords)
  d2xres, d2yres, d2zres = getxyzRes(d2xcoords, d2ycoords, d2zcoords)
  d3xres, d3yres, d3zres = getxyzRes(d3xcoords, d3ycoords, d3zcoords)

  #getting to the desired resolution (matching sim dose z res)
  desres = 3.0
  s0xdiff = round(desres/s0xres, 1)
  s0ydiff = round(desres/s0yres, 1)
  s0zdiff = round(desres/s0zres, 1)
  s1xdiff = round(desres/s1xres, 1)
  s1ydiff = round(desres/s1yres, 1)
  s1zdiff = round(desres/s1zres, 1)
  s2xdiff = round(desres/s2xres, 1)
  s2ydiff = round(desres/s2yres, 1)
  s2zdiff = round(desres/s2zres, 1)
  s3xdiff = round(desres/s3xres, 1)
  s3ydiff = round(desres/s3yres, 1)
  s3zdiff = round(desres/s3zres, 1)


  s0newx = int(round(len(s0xcoords)/(desres/s0xres), 0))
  s0newy = int(round(len(s0ycoords)/(desres/s0yres), 0))
```

```
  s0newz = int(round(len(s0zcoords)/(desres/s0zres), 0))
  s1newx = int(round(len(s1xcoords)/(desres/s1xres), 0))
  s1newy = int(round(len(s1ycoords)/(desres/s1yres), 0))
  s1newz = int(round(len(s1zcoords)/(desres/s1zres), 0))
  s2newx = int(round(len(s2xcoords)/(desres/s2xres), 0))
  s2newy = int(round(len(s2ycoords)/(desres/s2yres), 0))
  s2newz = int(round(len(s2zcoords)/(desres/s2zres), 0))
  s3newx = int(round(len(s3xcoords)/(desres/s3xres), 0))
  s3newy = int(round(len(s3ycoords)/(desres/s3yres), 0))
  s3newz = int(round(len(s3zcoords)/(desres/s3zres), 0))

  #resampling surface and dose
  s0res, s0resx, s0resy, s0resz = resampSurface(nsurf0, s0newx, s0newy, s0
newz, s0xcoords, s0ycoords, s0zcoords, desres)
  s1res, s1resx, s1resy, s1resz = resampSurface(nsurf1, s1newx, s1newy, s1
newz, s1xcoords, s1ycoords, s1zcoords, desres)
  s2res, s2resx, s2resy, s2resz = resampSurface(nsurf2, s2newx, s2newy, s2
newz, s2xcoords, s2ycoords, s2zcoords, desres)
  s3res, s3resx, s3resy, s3resz = resampSurface(nsurf3, s3newx, s3newy, s3
newz, s3xcoords, s3ycoords, s3zcoords, desres)

  d0res = zoom(d0, (d0xres/desres, d0yres/desres, d0zres/desres))
  d0resx = zoom(d0xcoords, d0xres/desres)
  d0resy = zoom(d0ycoords, d0yres/desres)
  d0resz = zoom(d0zcoords, d0zres/desres)
  d1res = zoom(d1, (d1xres/desres, d1yres/desres, d1zres/desres))
  d1resx = zoom(d1xcoords, d1xres/desres)
  d1resy = zoom(d1ycoords, d1yres/desres)
  d1resz = zoom(d1zcoords, d1zres/desres)
  d2res = zoom(d2, (d2xres/desres, d2yres/desres, d2zres/desres))
  d2resx = zoom(d2xcoords, d2xres/desres)
  d2resy = zoom(d2ycoords, d2yres/desres)
  d2resz = zoom(d2zcoords, d2zres/desres)
  d3res = zoom(d3, (d3xres/desres, d3yres/desres, d3zres/desres))
  d3resx = zoom(d3xcoords, d3xres/desres)
  d3resy = zoom(d3ycoords, d3yres/desres)
  d3resz = zoom(d3zcoords, d3zres/desres)

  #cut off last entry of resampled coordinates since it doesn't extend int
erpolation properly
  d0resx = d0resx[0:-1]
  d0resy = d0resy[0:-1]
  d0resz = d0resz[0:-1]
  d1resx = d1resx[0:-1]
  d1resy = d1resy[0:-1]
```

```
  d1resz = d1resz[0:-1]
  d2resx = d2resx[0:-1]
  d2resy = d2resy[0:-1]
  d2resz = d2resz[0:-1]
  d3resx = d3resx[0:-1]
  d3resy = d3resy[0:-1]
  d3resz = d3resz[0:-1]

  #shifting coordinates so 1,2,3 align with sim
  thresh = 350
  if np.count_nonzero(s1res[:,:,-4]) > thresh:
    s1xcshifted, s1ycshifted, s1zcshifted, d1xcshifted, d1ycshifted, d1zcs
hifted = findScalp2(s0res, s1res, s0resx, s0resy, s0resz, s1resx, s1resy,
s1resz, d1resx, d1resy, d1resz)
  else:
    s0x1, s0x2, s0y1, s0y2, s0z = findScalp(s0res, s0resx, s0resy, s0resz)
    s1x1, s1x2, s1y1, s1y2, s1z = findScalp(s1res, s1resx, s1resy, s1resz)
    s1xcshifted, s1ycshifted, s1zcshifted, d1xcshifted, d1ycshifted, d1zcs
hifted = shiftCoords(s0x1, s0x2, s0y1, s0y2, s0z, s1x1, s1x2, s1y1, s1y2,
s1z, s1resx, s1resy, s1resz, d1resx, d1resy, d1resz)
  if np.count_nonzero(s2res[:,:,-4]) > thresh:
    s2xcshifted, s2ycshifted, s2zcshifted, d2xcshifted, d2ycshifted, d2zcs
hifted = findScalp2(s0res, s2res, s0resx, s0resy, s0resz, s2resx, s2resy,
s2resz, d2resx, d2resy, d2resz)
  else:
    s0x1, s0x2, s0y1, s0y2, s0z = findScalp(s0res, s0resx, s0resy, s0resz)
    s2x1, s2x2, s2y1, s2y2, s2z = findScalp(s2res, s2resx, s2resy, s2resz)
    s2xcshifted, s2ycshifted, s2zcshifted, d2xcshifted, d2ycshifted, d2zcs
hifted = shiftCoords(s0x1, s0x2, s0y1, s0y2, s0z, s2x1, s2x2, s2y1, s2y2,
s2z, s2resx, s2resy, s2resz, d2resx, d2resy, d2resz)
  if np.count_nonzero(s3res[:,:,-4]) > thresh:
    s3xcshifted, s3ycshifted, s3zcshifted, d3xcshifted, d3ycshifted, d3zcs
hifted = findScalp2(s0res, s3res, s0resx, s0resy, s0resz, s3resx, s3resy,
s3resz, d3resx, d3resy, d3resz)
  else:
    s0x1, s0x2, s0y1, s0y2, s0z = findScalp(s0res, s0resx, s0resy, s0resz)
    s3x1, s3x2, s3y1, s3y2, s3z = findScalp(s3res, s3resx, s3resy, s3resz)
    s3xcshifted, s3ycshifted, s3zcshifted, d3xcshifted, d3ycshifted, d3zcs
hifted = shiftCoords(s0x1, s0x2, s0y1, s0y2, s0z, s3x1, s3x2, s3y1, s3y2,
s3z, s3resx, s3resy, s3resz, d3resx, d3resy, d3resz)

  midx = d0res.shape[0]//2
  fig = px.imshow(d0res[midx,:,:])
  fig.show()
  breakpoint()
```

```
  d0startx = midx-32
  d0stopx = midx+32
  y1 = input('starty index?')
  y2 = input('stopy index?')
  z1 = input('startz index?')
  z2 = input('stopz index?')
  d0starty = int(y1)
  d0stopy = int(y2)
  d0startz = int(z1)
  d0stopz = int(z2)
  d0startxcoord = d0resx[d0startx]
  d0stopxcoord = d0resx[d0stopx]
  d0startycoord = d0resy[d0starty]
  d0stopycoord = d0resy[d0stopy]
  d0startzcoord = d0resz[d0startz]
  d0stopzcoord = d0resz[d0stopz]
  d0xcrop = d0resx[d0startx:d0stopx]
  d0ycrop = d0resy[d0starty:d0stopy]
  d0zcrop = d0resz[d0startz:d0stopz]
  d0crop = d0res[d0startx:d0stopx, d0starty:d0stopy, d0startz:d0stopz]

  #crop surfaces and remaining dose maps based on coordinates from sim dos
e crop
  s0crop = cropSurface(s0res, s0resx, s0resy, s0resz, d0startxcoord, d0sto
pxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stopzcoord, desres,
 'surface')
  s1crop = cropSurface(s1res, s1xcshifted, s1ycshifted, s1zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'surface')
  s2crop = cropSurface(s2res, s2xcshifted, s2ycshifted, s2zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'surface')
  s3crop = cropSurface(s3res, s3xcshifted, s3ycshifted, s3zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'surface')


  d1crop = cropSurface(d1res, d1xcshifted, d1ycshifted, d1zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'dose')
  d2crop = cropSurface(d2res, d2xcshifted, d2ycshifted, d2zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'dose')
```

```
  d3crop = cropSurface(d3res, d3xcshifted, d3ycshifted, d3zcshifted, d0sta
rtxcoord, d0stopxcoord, d0startycoord, d0stopycoord, d0startzcoord, d0stop
zcoord, desres, 'dose')

  #write to nrrd file
  print(s0crop.shape, s1crop.shape, s2crop.shape, s3crop.shape)
  print(d0crop.shape, d1crop.shape, d2crop.shape, d3crop.shape)
  nrrd.write(surf0nrrd,s0crop)
  nrrd.write(surf1nrrd,s1crop)
  nrrd.write(surf2nrrd,s2crop)
  nrrd.write(surf3nrrd,s3crop)

  nrrd.write(dose0nrrd,d0crop)
  nrrd.write(dose1nrrd,d1crop)
  nrrd.write(dose2nrrd,d2crop)
  nrrd.write(dose3nrrd,d3crop)

  return
```

The following code converts OBJ files to binary masks.

```
def convertOBJ(surface):
  #import surface
  surf = pywavefront.Wavefront(surface)
  #put vertices into a numpy array
  v = np.array(surf.vertices)
  #see xv below for explanation
  zv = np.unique(v, axis = 0)
  zv = zv[:, 0]
  #print(np.unique(zv))
  diffz = np.round_(zv[1:]-zv[0:-1], decimals = 0)
  diffz = max(diffz)

  #switch x and y?
  v[:,[2,0]] = v[:,[0,2]]
  #do the same for y as below with x
  yv = np.unique(v, axis = 0)
  yv = yv[:,0]
  yv = np.unique(yv, axis=0)
  diffy = np.round_(yv[1:]-yv[0:-1], decimals = 0)
  if max(diffy) == 0:
    diffy = 1
  else:
    diffy = min(diffy[diffy != 0])
```

```
  #switch axes again
  v[:,[1,0]] = v[:,[0,1]]
  #sorts v according to unique elements of x?
  xv = np.unique(v, axis = 0)

  #take just x elements for array/vector
  xv = xv[:,0]
  #round the difference between each element in x and the previous element
 in x
  diffx = np.round_(xv[1:]-xv[0:-1], decimals = 0)
  #find the minimum non-zero difference?
  diffx = min(diffx[diffx != 0])

  #get max x, y, z, values
  xmax = max(v[:,0])
  xmin = min(v[:,0])
  ymax = max(v[:,1])
  ymin = min(v[:,1])
  zmax = max(v[:,2])
  zmin = min(v[:,2])
  #get range of x,y,z values
  xspan = (xmax-xmin)/diffx
  xspan = xspan.astype(int)
  yspan = (ymax-ymin)/diffy
  yspan = yspan.astype(int)
  zspan = (zmax-zmin)/diffz
  zspan = zspan.astype(int)


  #create regular coordinate grid
  x_coords = np.arange(xmin, xmax, diffx)
  y_coords = np.arange(ymin, ymax, diffy)
  z_coords = np.arange(zmin, zmax+1, diffz)
  pixel_data = np.ones((len(x_coords), len(y_coords), len(z_coords)))
  coords = {'x': x_coords, 'y': y_coords, 'z': z_coords}
  coordGrid = xr.DataArray(pixel_data, dims=('x','y','z'), coords = coords
)
  #print(coords)


  vlist = v.tolist()


  #generate mask
  vol = create_mask2(coordGrid, vlist);
```

```
  return vol
```

The following code creates binary masks from a list of coordinates.

```
def create_mask2(array, contour_list):
    'Takes in an Array we want to mask, and a contour'
    'coordinate nested list.'
    #surf = pywavefront.Wavefront(obj)
    #creates array of zeros with same size and shape as array
    #mask_array = xr.zeros_like(array)
    slice_dict = {}
    mask_dict = {}
    z = 0

    #automatic iteration through contour_list. i is iterating value (0, 1,
 2...), coord is the value at that index?
    #puts x,y coordinates into slice_dict (essentially reorganizes coordin
ates)
    for i, coord in enumerate(contour_list):
        'Each list inside contour_list is a list of (x,y,z)'
        'coordinates defining the contour on each slice. For each'
        'sequence, the z coord is constant. This for-loop builds'
        'a numpy array for each series of (x,y) coordinates and'
        'stores them in a dict where the corresponding key is the'
        'z-coordinate.'
        #
        x_start = float(array.x[0].values)
        y_start = float(array.y[0].values)
        spacing = np.round_(float(array.x[1].values) - float(array.x[0].va
lues), decimals = 1)
        resized = np.resize(coord, (int(np.size(coord) / 3), 3))
        #print(resized)
        #print(resized[:,:2])
        rtstruct_pixelArray = (resized[:,:2] - [x_start, y_start]) / spaci
ng
        rtstruct_pixelArray = np.rint(rtstruct_pixelArray)
        #print(rtstruct_pixelArray[0])
        #print(coord[2])
        #print(type(rtstruct_pixelArray))
        #print(type(coord[2]))
        if coord[2] != z:
          slice_dict[coord[2]] = rtstruct_pixelArray
          #print(z)
        else:
```

```
      #print(slice_dict[coord[2]])
      prev = slice_dict[coord[2]]
      #print(prev)
      #print(rtstruct_pixelArray)
      new = np.append(prev, rtstruct_pixelArray, axis = 0)
      #print(new)
      slice_dict[coord[2]] = new
      #breakpoint()
    #print(slice_dict)
    z = coord[2]

#print(slice_dict.keys())

for z_slc in slice_dict.keys():
    'For each z-slice, we need to draw a polygon defined by'
    'the points we pulled above. We can do this with ImageDraw.'
    polygon = slice_dict[z_slc].flatten().tolist()
    #print(polygon)
    img = Image.new('1', (array.shape[0], array.shape[1]))
    #matplotlib.pyplot.imshow(img)
    ImageDraw.Draw(img).polygon(polygon, outline=1, fill=1)
    #matplotlib.pyplot.imshow(img)
    #breakpoint()
    mask = np.array(img)
    #print(z_slc, np.sum(mask))
    mask_dict[z_slc] = mask

i = -1
mask_vol = xr.zeros_like(array)
mask_dict = dict(sorted(mask_dict.items()))
#print(mask_dict.keys())
#breakpoint()
for z_slc in mask_dict.keys():
    'We can reassign values in mask_array with indexing.'
    'We rely on sel to give us the slice we need. This works'
    'because xarray does not copy the data with sel.'
    i += 1
    if i > mask_vol.shape[2]-1:
      break
    #print(i)
    #mask_slice = mask_array.sel(z=z_slc, method=None)
    #print(mask_slice)
    #print(mask_dict)
    a = np.swapaxes(mask_dict[z_slc], 0, 1)
    #mask_slice[:,:] = mask_dict[z_slc]
```

```
            mask_slice = a
            mask_vol[:,:,i] = mask_slice
            #print(mask_slice)

        #final = mask_vol[:,:,0:i]
        #print(mask_vol.coords)
        #print(mask_vol.shape)

        return mask_vol
```

The following code converts radiation dose files to numpy arrays.

```
def importDose(dosePath):
  #imports dose from rd file and returns xarray as xdose and separated num
py array of dose map and coordinates
  ref = pydicom.read_file(dosePath)
  coords, dose = pymedphys.dicom.zyx_and_dose_from_dataset(ref)
  new_dose = np.swapaxes(dose, 0,2)

  xcoords = np.around(np.asarray(coords[2]), decimals = 1)
  ycoords = np.around(np.asarray(coords[1]), decimals = 1)
  zcoords = np.around(np.asarray(coords[0]), decimals = 1)
  dosecoords = {'x': xcoords, 'y': ycoords, 'z': zcoords}

  xdose = xr.DataArray(new_dose, dims=('x','y','z'), coords = dosecoords)

  return xdose, new_dose, xcoords, ycoords, zcoords
```

The following code separates coordinates and data from Python data arrays.

```
def xyz_and_numpy_from_xarray(xarray):
  #separates numpy array and coordinates out of xarray
  xcoord = xr.DataArray.to_numpy(xarray.coords['x'])
  ycoord = xr.DataArray.to_numpy(xarray.coords['y'])
  zcoord = xr.DataArray.to_numpy(xarray.coords['z'])
  array = xr.DataArray.to_numpy(xarray)
  return array, xcoord, ycoord, zcoord
```

The following code obtains the resolution of coordinate lists.

```
def getxyzRes(xcoord, ycoord, zcoord):
  #gets resolution of x,y,z coordinates
  xres = round((xcoord[-1] - xcoord[0])/len(xcoord), 1)
  yres = round((ycoord[-1] - ycoord[0])/len(ycoord), 1)
  zres = round((zcoord[-1] - zcoord[0])/len(zcoord), 0)
  return xres, yres, zres
```

The following code resamples the binary surface mask.

```
def resampSurface(surfArray, newx, newy, newz, xcoords, ycoords, zcoords,
dzres):
  #resamples the array and coordinates to dzres
  resampsxyVol = np.zeros((newx, newy, len(zcoords)))
  for z in range(surfArray.shape[2]):
    resampsxyVol[:,:,z] = cv2.resize(surfArray[:,:,z], (newy, newx), inter
polation=cv2.INTER_NEAREST)
  resampsVol = np.zeros((newx, newy, newz))
  if dzres ==3:
    for z in range(surfArray.shape[2]):
      resampsVol[:,:,2*z] = resampsxyVol[:,:,z]
      resampsVol[:,:,2*z+1] = resampsxyVol[:,:,z]
    rszcoords = np.around(np.arange(zcoords[0],zcoords[-
1]+2*dzres,dzres), decimals=1)
  elif dzres ==2:
    for z in range(surfArray.shape[2]):
      resampsVol[:,:,3*z] = resampsxyVol[:,:,z]
      resampsVol[:,:,3*z+1] = resampsxyVol[:,:,z]
      resampsVol[:,:,3*z+2] = resampsxyVol[:,:,z]
    rszcoords = np.around(np.arange(zcoords[0],zcoords[-
1]+3*dzres,dzres), decimals=1)
  rsxcoords = np.around(np.arange(xcoords[0],xcoords[-
1]+dzres,dzres), decimals=1)
  rsycoords = np.around(np.arange(ycoords[0],ycoords[-
1]+dzres,dzres), decimals=1)

  return resampsVol, rsxcoords, rsycoords, rszcoords
```

The following code finds the superior extent of the anatomy in a surface array, if the top of the skull is included in the array.

```
def findScalp(array, xcoords, ycoords, zcoords):
  #finds the scalp in a surface array if the top of the surface includes t
he scalp
  for z in reversed(range(array.shape[2])):
    if np.sum(array[:,:,z]) > 0:
      nz = np.nonzero(array[:,:,z])
      x1 = min(nz[0])
      x2 = max(nz[0])
      y1 = min(nz[1])
      y2 = max(nz[1])
      zin = z
      break
  startx = xcoords[x1]
  stopx = xcoords[x2]
```

```
  starty = ycoords[y1]
  stopy = ycoords[y2]
  zc = zcoords[zin]
  return startx, stopx, starty, stopy, zc
```

The following code matches two surface arrays if the top of the skull is not included in the surface array.

```
def findScalp2(array0, array1, x0coords, y0coords, z0coords, sx1coords, sy
1coords, sz1coords, dx1coords, dy1coords, dz1coords):
  #shifts coordinates for a given surface (1,2,3) to align with surface 0
by trying to find slices where the size of the head matches
  #print(np.count_nonzero(array1[:,:,-1]))
  nz = np.nonzero(array1[:,:,-4])
  x1 = min(nz[0])
  x2 = max(nz[0])
  y1 = min(nz[1])
  y2 = max(nz[1])
  width1 = x2-x1
  #print(width1)
  startx1 = sx1coords[x1]
  stopx1 = sx1coords[x2]
  starty1 = sy1coords[y1]
  stopy1 = sy1coords[y2]
  zc1 = sz1coords[-1]
  for z in reversed(range(array0.shape[2])):
    nz0 = np.nonzero(array0[:,:,z])
    width0 = max(nz0[0])-min(nz0[0])
    if width0 > width1:
      if z == array0.shape[2]-1:
        zin = z
        x01 = min(nz0[0])
        x02 = max(nz0[0])
        y01 = min(nz0[1])
        y02 = max(nz0[1])
      #print(np.nonzero(array0[:,:,z]))
      #print(np.nonzero(array0[:,:,z+1]))
      else:
        nzprev = np.nonzero(array0[:,:,z+1])
        width0prev = max(nzprev[0])-min(nzprev[0])
        #print(z)
        if min(abs(width0-width1), abs(width0prev-width1)) == abs(width0-
width1):
          zin = z
          x01 = min(nz0[0])
          x02 = max(nz0[0])
```

```
            y01 = min(nz0[1])
            y02 = max(nz0[1])
          else:
            zin = z+1
            x01 = min(nzprev[0])
            x02 = max(nzprev[0])
            y01 = min(nzprev[1])
            y02 = max(nzprev[1])
        break
  startx0 = x0coords[x01]
  stopx0 = x0coords[x02]
  starty0 = y0coords[y01]
  stopy0 = y0coords[y02]
  zc0 = z0coords[zin]
  xshift = ((startx0-startx1)+(stopx0-stopx1))/2
  yshift = ((starty0-starty1)+(stopy0-stopy1))/2
  zshift = zc0-zc1
  shifteddxc = dx1coords + xshift
  shifteddyc = dy1coords + yshift
  shifteddzc = dz1coords + zshift
  shiftedsxc = sx1coords + xshift
  shiftedsyc = sy1coords + yshift
  shiftedszc = sz1coords + zshift

  return shiftedsxc, shiftedsyc, shiftedszc, shifteddxc, shifteddyc, shift
eddzc, xshift, yshift, zshift
```

The following code shifts surface and dose coordinates to ensure alignment.

```
def shiftCoords(s0x1, s0x2, s0y1, s0y2, s0z, s1x1, s1x2, s1y1, s1y2, s1z,
sxcoords, sycoords, szcoords, dxcoords, dycoords, dzcoords):
  #shifts coordinates if the top of the head is included in the scan to al
ign the surface and dose with the sim
  xshift = ((s0x1-s1x1)+(s0x2-s1x2))/2
  yshift = ((s0y1-s1y1)+(s0y2-s1y2))/2
  zshift = s0z-s1z
  shifteddxc = dxcoords + xshift
  shifteddyc = dycoords + yshift
  shifteddzc = dzcoords + zshift
  shiftedsxc = sxcoords + xshift
  shiftedsyc = sycoords + yshift
  shiftedszc = szcoords + zshift
  return shiftedsxc, shiftedsyc, shiftedszc, shifteddxc, shifteddyc, shift
eddzc, xshift, yshift, zshift
```

The following code crops the surface and dose arrays

```
def cropSurface(surfArray, xcoords, ycoords, zcoords, startxcoord, stopxco
ord, startycoord, stopycoord, startzcoord, stopzcoord, dzres, type):
  #crops surface (or dose) based on matching coordinates to the dose crop
  xstartdiff = np.absolute(xcoords-startxcoord)
  surfxstartindex = xstartdiff.argmin()
  xstopdiff = np.absolute(xcoords-stopxcoord)
  surfxstopindex = xstopdiff.argmin()
  ystartdiff = np.absolute(ycoords-startycoord)
  surfystartindex = ystartdiff.argmin()
  if stopycoord > ycoords[-1] +2.5:
    ystopdiff = np.absolute(stopycoord-ycoords[-1])
    surfystopindex = int(round(ystopdiff/dzres, 0) + len(ycoords)-1)
  else:
    ystopdiff = np.absolute(ycoords - stopycoord)
    surfystopindex = ystopdiff.argmin()
  zstartdiff = np.absolute(zcoords-startzcoord)
  surfzstartindex = zstartdiff.argmin()
  zstopdiff = np.absolute(zcoords-stopzcoord)
  surfzstopindex = zstopdiff.argmin()
  #print(surfxstartindex, surfxstopindex, surfystartindex, surfystopindex,
 surfzstartindex, surfzstopindex)
  if surfxstopindex - surfxstartindex < 64:
    if surfxstopindex - surfxstartindex == 63:
      if surfxstartindex == 0:
        surfxstopindex += 1
      elif abs(xcoords[surfxstartindex]-xcoords[surfxstartindex-
1]) < abs(xcoords[surfxstopindex]+xcoords[surfxstopindex+1]):
        surfxstartindex = surfxstartindex - 1
      elif abs(xcoords[surfxstartindex]-xcoords[surfxstartindex-
1]) > abs(xcoords[surfxstopindex]+xcoords[surfxstopindex+1]):
        surfxstopindex += 1
    elif surfxstopindex - surfxstartindex == 62 and surfxstartindex != 0:
      surfxstartindex = surfxstartindex - 1
      surfxstopindex = surfxstopindex + 1
    elif surfxstopindex - surfxstartindex == 62 and surfxstartindex == 0:
      surfxstopindex = surfxstopindex + 2
  if surfxstopindex - surfxstartindex > 64:
    if xstopdiff.argmin() > xstartdiff.argmin():
      surfxstartindex += 1
    else:
      surfxstopindex += -1

  if surfystopindex - surfystartindex < 64:
```

```
    if surfystopindex - surfystartindex == 63:
      if surfystopindex > len(ycoords)-1:
        surfystopindex += 1
      else:
        if surfystartindex == 0:
          surfystopindex += 1
        elif abs(ycoords[surfystartindex]-ycoords[surfystartindex-
1]) < abs(ycoords[surfystopindex]+ycoords[surfystopindex+1]):
          surfystartindex = surfystartindex - 1
        elif abs(ycoords[surfystartindex]-ycoords[surfystartindex-
1]) > abs(ycoords[surfystopindex]+ycoords[surfystopindex+1]):
          surfystopindex += 1
    elif surfystopindex - surfystartindex == 62 and surfystartindex != 0:
      surfystartindex = surfystartindex - 1
      surfystopindex = surfystopindex + 1
    elif surfystopindex - surfystartindex == 62 and surfystartindex == 0:
      surfystopindex = surfystopindex + 2
    elif surfystopindex - surfystartindex == 61 and surfystartindex != 0:
      surfystartindex = surfystartindex - 1
      surfystopindex = surfystopindex + 2
    elif surfystopindex - surfystartindex == 61 and surfystartindex == 0:
      surfystopindex = surfystopindex + 3
  if surfystopindex - surfystartindex > 64:
    if ystopdiff.argmin() > ystartdiff.argmin():
      surfystartindex += 1
    else:
      surfystopindex = surfystopindex - 1

  if surfzstopindex - surfzstartindex < 64:
    if surfzstopindex - surfzstartindex == 63:
      if surfzstartindex == 0:
        surfzstopindex += 1
      elif abs(zcoords[surfzstartindex]-zcoords[surfzstartindex-
1]) < abs(zcoords[surfzstopindex]+zcoords[surfzstopindex+1]):
        surfzstartindex = surfzstartindex - 1
      elif abs(zcoords[surfzstartindex]-zcoords[surfzstartindex-
1]) > abs(zcoords[surfzstopindex]+zcoords[surfzstopindex+1]):
        surfzstopindex += 1
    elif surfzstopindex - surfzstartindex == 62 and surfzstartindex != 0:
      surfzstartindex = surfzstartindex - 1
      surfzstopindex = surfzstopindex + 1
    elif surfzstopindex - surfzstartindex == 62 and surfzstartindex == 0:
      surfzstopindex = surfzstopindex + 2
  elif surfzstopindex - surfzstartindex > 64:
    if zstopdiff.argmin() > zstartdiff.argmin():
```

```
      surfzstartindex += 1
    else:
      surfzstopindex = surfzstopindex - 1

  if type == 'surface':
    surfcrop = np.zeros((surfxstopindex-surfxstartindex, surfystopindex-
surfystartindex, surfzstopindex-surfzstartindex))
    #print(surfArray.shape)
    for y in range(surfArray.shape[1]-surfystartindex):
      if y == surfcrop.shape[1]:
        break
      else:
        surfcrop[:,y,:] = surfArray[surfxstartindex:surfxstopindex, surfys
tartindex+y, surfzstartindex:surfzstopindex]
  else:
    surfcrop = surfArray[surfxstartindex:surfxstopindex, surfystartindex:s
urfystopindex, surfzstartindex:surfzstopindex]

  return surfcrop, surfxstartindex, surfystartindex, surfzstartindex
```

B.3.2 Neural Network

The following code prepares the data for input to the network.

```
class surfaceDoseData(keras.utils.Sequence):
  def __init__(self, batch_size, img_size, input_img_paths, target_img_pat
hs):
    self.batch_size = batch_size
    self.img_size = img_size
    self.input_img_paths = input_img_paths
    self.target_img_paths = target_img_paths

  def __len__(self):
    return len(self.target_img_paths) // self.batch_size

  def __getitem__(self, idx):
    i = idx * self.batch_size
    batch_input_img_paths = self.input_img_paths[i:i+self.batch_size]
    batch_target_img_paths = self.target_img_paths[i:i+self.batch_size]
    x = np.zeros((self.batch_size, )+ self.img_size + (3,))
    for j, path in enumerate(batch_input_img_paths):
      #print(path)
      surface0, header = nrrd.read('/content/drive/MyDrive/Colab/HNSI/Cent
erCrop/Surface/' + '0/' + path.split('/')[1] + '.nrrd')
      surfacex, header = nrrd.read('/content/drive/MyDrive/Colab/HNSI/Cent
erCrop/Surface/' + path + '.nrrd')
```

```
      dose0, header = nrrd.read('/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/' + '0/' + path.split('/')[1] + '.nrrd')

      dose0norm = dose0/100


      x[j] = np.stack((surface0, surfacex, dose0norm), axis=-1)
    y = np.zeros((self.batch_size,) + self.img_size)
    for j, path in enumerate(batch_target_img_paths):
      dosex, header = nrrd.read('/content/drive/MyDrive/Colab/HNSI/CenterCrop/Dose/' + path + '.nrrd')

      dosexnorm = dosex/100


      y[j] = dosexnorm

    return x, y
```

The following code is the network.

```
def unetconv1(sz = (64, 64, 64, 3)):
  x = tf.keras.Input(sz)
  inputs = x
  ### [First half of the network: downsampling inputs] ###
  layers = []
  # Entry block
  x = tf.keras.layers.Conv3D(16, (3, 3, 3), padding = 'same', activation='relu', input_shape=(64, 64, 64, 3))(x)
  x = tf.keras.layers.Conv3D(16, (3, 3, 3), padding = 'same', activation='relu')(x)
  layers.append(x)
  #print(x.shape)
  x = tf.keras.layers.MaxPooling3D()(x)


  # Blocks 1, 2, 3 are identical apart from the feature depth.
  for filters in [32, 64, 128]:
    x = tf.keras.layers.Conv3D(filters, (3, 3, 3), padding = 'same', activation='relu')(x)

    x = tf.keras.layers.Conv3D(filters, (3, 3, 3), padding = 'same', activation='relu')(x)

    layers.append(x)
    print(x.shape)
```

```
    x = tf.keras.layers.MaxPooling3D()(x)

  j = len(layers) - 1
  x = tf.keras.layers.Conv3D(256, (3, 3, 3), padding = 'same', activation=
'relu')(x)
  x = tf.keras.layers.Conv3D(256, (3, 3, 3), padding = 'same', activation=
'relu')(x)
  x = tf.keras.layers.Conv3D(256, (3, 3, 3), padding = 'same', activation=
'relu')(x)
  x = tf.keras.layers.Conv3D(256, (3, 3, 3), padding = 'same', activation=
'relu')(x)
  #print(x.shape)
  x = tf.keras.layers.Conv3DTranspose(128, 2, strides = 2, activation='rel
u')(x)
  #print(x.shape)
  x = tf.keras.layers.Concatenate(axis=4)([x, layers[j]])
  #print('after 1 concat', x.shape)
  j = j -1
  ### [Second half of the network: upsampling inputs] ###

  for filters in [128, 64, 32]:
    x = tf.keras.layers.Conv3D(filters, (3, 3, 3), padding = 'same', activ
ation='relu')(x)
    x = tf.keras.layers.Conv3D(filters, (3, 3, 3), padding = 'same', activ
ation='relu')(x)
    x = tf.keras.layers.Conv3DTranspose(filters/2, 2, strides = 2, activat
ion='relu')(x)
    #print(x.shape)
    x = tf.keras.layers.Concatenate(axis=4)([x, layers[j]])
    #print(x.shape)
    j = j-1

  x = tf.keras.layers.Conv3D(16, (3, 3, 3), padding = 'same', activation='
relu')(x)
  x = tf.keras.layers.Conv3D(16, (3, 3, 3), padding = 'same', activation='
relu')(x)
  outputs = tf.keras.layers.Conv3D(1, (1,1,1), padding = 'same', activatio
n='relu')(x)
  print(outputs.shape)

  # Add a per-pixel classification layer
  #outputs = layers.Conv2D(num_classes, 3, activation="softmax", padding="
same")(x)

  # Define the model
```

```
  model = tf.keras.Model(inputs, outputs)

  return model
```

B.3.3 Dose Prediction Analysis

The following script was used for evaluation of the network.

```
truearray, headert = nrrd.read('/content/drive/MyDrive/Colab/HNSI/CenterCr
op/Dose/3/1.nrrd')
predarray, headerp = nrrd.read('/content/drive/MyDrive/Colab/HNSI/AllTrain
Predictions/3-1/Pt1CBCT3Pred.nrrd')

rx = 69.96
predarray = predarray*100
maxpred = np.amax(predarray)
maxtrue = np.amax(truearray)
print('True max is', maxtrue, 'Gy. Predicted max is', maxpred, 'Gy.')
truearray001 = truearray*(truearray >= 0.001)
predarray001 = predarray*(predarray >= 0.001)
meanpred = np.sum(predarray)/np.count_nonzero(predarray)
meantrue = np.sum(truearray001)/np.count_nonzero(truearray >= 0.001)
#meanpred = np.mean(predarray)
#meantrue = np.mean(truearray)
print('True mean is', meantrue, 'Gy. Predicted mean is', meanpred, 'Gy.')
rmse = np.sqrt(((predarray - truearray)**2).mean())
mae = np.mean(np.abs(truearray - predarray))
print('RMSE is', rmse, '. MAE is', mae)

#V100 DSC
v100true = np.copy(truearray)
v100pred = np.copy(predarray)

v100true[v100true < rx] = 0
v100true[v100true >= rx] = 1
v100pred[v100pred < rx] = 0
v100pred[v100pred >= rx] = 1

v100true.astype(bool)
v100pred.astype(bool)
intersection = np.logical_and(v100true, v100pred)
dsc100 = 2. * intersection.sum() / (v100true.sum() + v100pred.sum())
print('The DSC between the true V100 and the predicted V100 is', dsc100)
vol100true = np.count_nonzero(v100true)*27/1000
vol100pred = np.count_nonzero(v100pred)*27/1000
```

```
print('The volume of the true V100 is', vol100true, 'cc. The volume of the
 predicted V100 is', vol100pred, 'cc.')

#V95 DSC
v95true = np.copy(truearray)
v95pred = np.copy(predarray)
thresh = 0.95*rx

v95true[v95true < thresh] = 0
v95true[v95true >= thresh] = 1
v95pred[v95pred < thresh] = 0
v95pred[v95pred >= thresh] = 1

v95true.astype(bool)
v95pred.astype(bool)
intersection = np.logical_and(v95true, v95pred)
dsc95 = 2. * intersection.sum() / (v95true.sum() + v95pred.sum())
print('The DSC between the true V95 and the predicted V95 is', dsc95)

#V90 DSC
v90true = np.copy(truearray)
v90pred = np.copy(predarray)
thresh = 0.90*rx

v90true[v90true < thresh] = 0
v90true[v90true >= thresh] = 1
v90pred[v90pred < thresh] = 0
v90pred[v90pred >= thresh] = 1

v90true.astype(bool)
v90pred.astype(bool)
intersection = np.logical_and(v90true, v90pred)
dsc90 = 2. * intersection.sum() / (v90true.sum() + v90pred.sum())
print('The DSC between the true V90 and the predicted V90 is', dsc90)
vol90true = np.count_nonzero(v90true)*27/1000
vol90pred = np.count_nonzero(v90pred)*27/1000
print('The volume of the true V90 is', vol90true, 'cc. The volume of the p
redicted V90 is', vol90pred, 'cc.')

#V70 DSC
v70true = np.copy(truearray)
v70pred = np.copy(predarray)
thresh = 0.70*rx

v70true[v70true < thresh] = 0
```

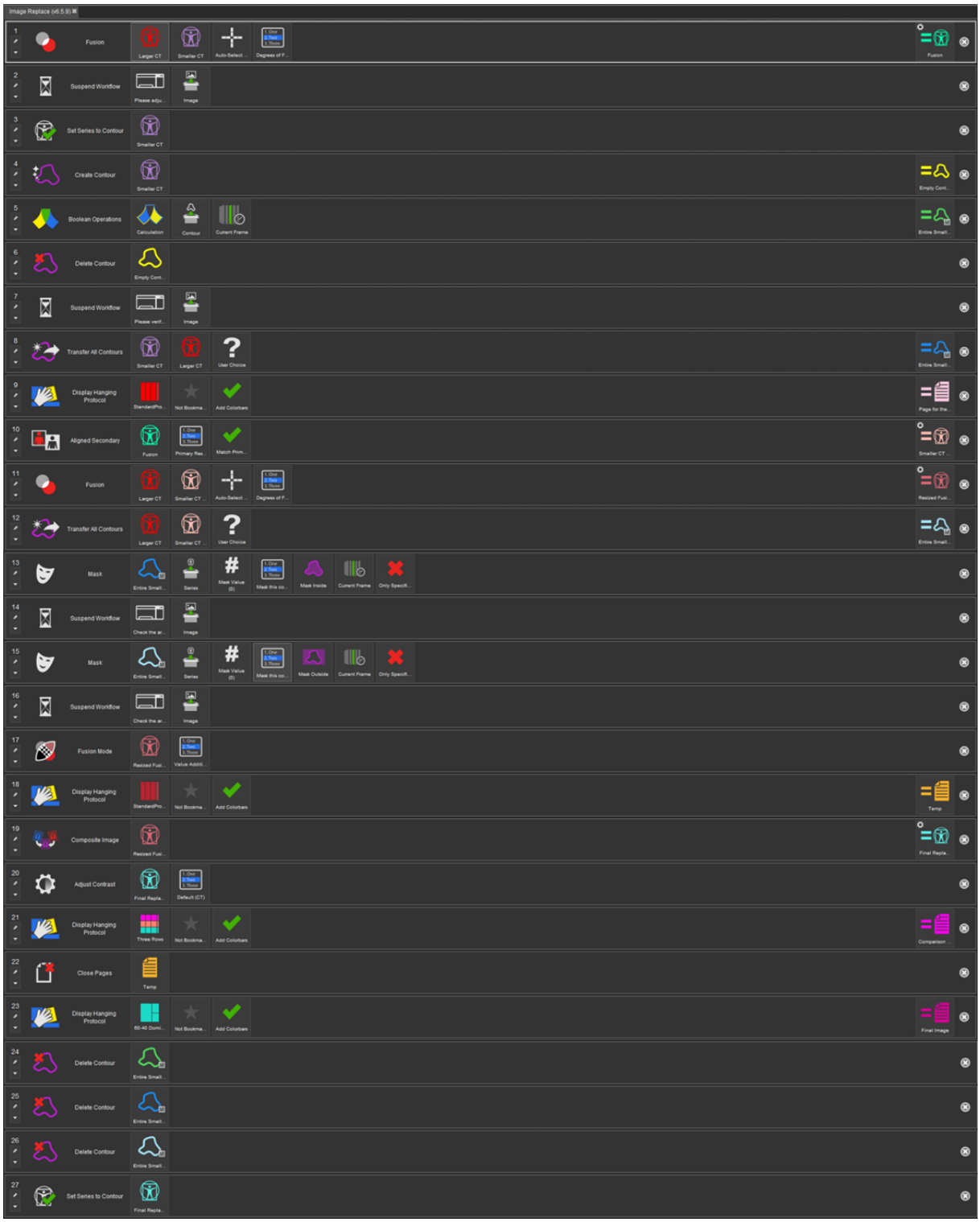
```
v70true[v70true >= thresh] = 1
v70pred[v70pred < thresh] = 0
v70pred[v70pred >= thresh] = 1

v70true.astype(bool)
v70pred.astype(bool)
intersection = np.logical_and(v70true, v70pred)
dsc70 = 2. * intersection.sum() / (v70true.sum() + v70pred.sum())
print('The DSC between the true V70 and the predicted V70 is', dsc70)

#V50 DSC
v50true = np.copy(truearray)
v50pred = np.copy(predarray)
thresh = 0.50*rx

v50true[v50true < thresh] = 0
v50true[v50true >= thresh] = 1
v50pred[v50pred < thresh] = 0
v50pred[v50pred >= thresh] = 1

v50true.astype(bool)
v50pred.astype(bool)
intersection = np.logical_and(v50true, v50pred)
dsc50 = 2. * intersection.sum() / (v50true.sum() + v50pred.sum())
print('The DSC between the true V50 and the predicted V50 is', dsc50)

#V20 DSC
v20true = np.copy(truearray)
v20pred = np.copy(predarray)
thresh = 0.20*rx

v20true[v20true < thresh] = 0
v20true[v20true >= thresh] = 1
v20pred[v20pred < thresh] = 0
v20pred[v20pred >= thresh] = 1

v20true.astype(bool)
v20pred.astype(bool)
intersection = np.logical_and(v20true, v20pred)
dsc20 = 2. * intersection.sum() / (v20true.sum() + v20pred.sum())
print('The DSC between the true V20 and the predicted V20 is', dsc20)

#V10 DSC
v10true = np.copy(truearray)
v10pred = np.copy(predarray)
```

```
thresh = 0.10*rx

v10true[v10true < thresh] = 0
v10true[v10true >= thresh] = 1
v10pred[v10pred < thresh] = 0
v10pred[v10pred >= thresh] = 1

v10true.astype(bool)
v10pred.astype(bool)
intersection = np.logical_and(v10true, v10pred)
dsc10 = 2. * intersection.sum() / (v10true.sum() + v10pred.sum())
print('The DSC between the true V10 and the predicted V10 is', dsc10)

#True V100 overlaid on predicted dose map: mean and is the max inside
v100td = v100true*truearray
v100pd = v100true*predarray
v100tnonzero = np.count_nonzero(v100td)
v100pnonzero = np.count_nonzero(v100pd)
v100tmean = np.sum(v100td)/v100tnonzero
v100pmean = np.sum(v100pd)/v100pnonzero
print('The mean of the true V100 is', v100tmean, 'Gy. The mean of the pred
icted V100 is', v100pmean, 'Gy.')
print('The predicted max is within the true V100 volume:', np.amax(v100pd)
 == maxpred)

#True V95 overlaid on predicted dose map: mean
v95td = v95true*truearray
v95pd = v95true*predarray
v95tnonzero = np.count_nonzero(v95td)
v95pnonzero = np.count_nonzero(v95pd)
v95tmean = np.sum(v95td)/v95tnonzero
v95pmean = np.sum(v95pd)/v95pnonzero
print('The mean of the true V95 is', v95tmean, 'Gy. The mean of the predic
ted V95 is', v95pmean, 'Gy.')

#True V90 overlaid on predicted dose map: mean
v90td = v90true*truearray
v90pd = v90true*predarray
v90tnonzero = np.count_nonzero(v90td)
v90pnonzero = np.count_nonzero(v90pd)
v90tmean = np.sum(v90td)/v90tnonzero
v90pmean = np.sum(v90pd)/v90pnonzero
print('The mean of the true V90 is', v90tmean, 'Gy. The mean of the predic
ted V90 is', v90pmean, 'Gy.')
```

```
#True V70 overlaid on predicted dose map: mean
v70td = v70true*truearray
v70pd = v70true*predarray
v70tnonzero = np.count_nonzero(v70td)
v70pnonzero = np.count_nonzero(v70pd)
v70tmean = np.sum(v70td)/v70tnonzero
v70pmean = np.sum(v70pd)/v70pnonzero
print('The mean of the true V70 is', v70tmean, 'Gy. The mean of the predic
ted V70 is', v70pmean, 'Gy.')


bins = range(1, math.ceil(maxtrue))
truehist100 = np.zeros(math.ceil(maxtrue))
predhist100 = np.zeros(math.ceil(maxtrue))
truehist95 = np.zeros(math.ceil(maxtrue))
predhist95 = np.zeros(math.ceil(maxtrue))
truehist90 = np.zeros(math.ceil(maxtrue))
predhist90 = np.zeros(math.ceil(maxtrue))
truehist70 = np.zeros(math.ceil(maxtrue))
predhist70 = np.zeros(math.ceil(maxtrue))
for i in bins:
  truehist100[i] = np.count_nonzero(v100td >= i)/np.count_nonzero(v100td)
  predhist100[i] = np.count_nonzero(v100pd >= i)/np.count_nonzero(v100pd)
  truehist95[i] = np.count_nonzero(v95td >= i)/np.count_nonzero(v95td)
  predhist95[i] = np.count_nonzero(v95pd >= i)/np.count_nonzero(v95pd)
  truehist90[i] = np.count_nonzero(v90td >= i)/np.count_nonzero(v90td)
  predhist90[i] = np.count_nonzero(v90pd >= i)/np.count_nonzero(v90pd)
  truehist70[i] = np.count_nonzero(v70td >= i)/np.count_nonzero(v70td)
  predhist70[i] = np.count_nonzero(v70pd >= i)/np.count_nonzero(v70pd)

plt.plot(truehist100, color = 'darkred', label = 'True V100')
plt.plot(predhist100, color = 'red', linestyle = 'dashed', label = 'Predic
ted V100')
plt.plot(truehist95, color = 'darkgoldenrod', label = 'True V95')
plt.plot(predhist95, color = 'gold', linestyle = 'dashed', label = 'Predic
ted V95')
plt.plot(truehist90, color = 'darkgreen', label = 'True V90')
plt.plot(predhist90, color = 'lime', linestyle = 'dashed', label = 'Predic
ted V90')
plt.plot(truehist70, color = 'navy', label = 'True V70')
plt.plot(predhist70, color = 'cornflowerblue', linestyle = 'dashed', label
 = 'Predicted V70')

plt.xlim(60, 85)
plt.title('DVH for Various Dose-Volumes')
```

```
plt.xlabel('Absolute Dose (Gy)')
plt.ylabel('Volume (%)')
plt.legend(loc = 'best')
plt.savefig('/content/drive/MyDrive/Colab/HNSI/AllTrainPredictions/3-
1/DoseVolDVHTail.png', dpi = 300, format = 'png')
#plt.show()
#plt.imshow(predarray[:,:,30], cmap = 'jet', vmin=0, vmax=78)
#plt.colorbar()
#plt.imshow(v90pred[:,:,40], cmap = 'gray')
#plt.savefig('/content/drive/MyDrive/Colab/HNSI/AllTrainPredictions/3-
46/PredDoseMap.png', dpi = 300, format = 'png')
#plt.imshow(predarray[:,:,30]-truearray[:,:,30], cmap = 'bwr', vmin = -
25, vmax = 25)
#plt.colorbar()
#plt.savefig('/content/drive/MyDrive/Colab/HNSI/AllTrainPredictions/3-
46/DoseDiff.png', dpi = 300, format = 'png')
```

Appendix C MIM Workflow for DIR



Appendix D DIR Technical Note

Technical Note: Validation of an Open-Source Deformable Image Registration Algorithm for CT-to-CBCT Registration of HN Patients

Sarah Holler, Bridget Quinn, Laura Padilla[1], Chris Guy
Department of Radiation Oncology, Virginia Commonwealth University, Richmond, Virginia

Corresponding author: Sarah Holler
hollerse@vcu.edu
401 College Street
Box 980058
Richmond, VA 23298

Running Title: Validating DIR for CT-to-CBCT Registration

[1] Department of Radiation Medicine and Applied Sciences, University of California San Diego, San Diego, California

Abstract

Background: Growing in popularity, adaptive radiotherapy therapy (ART) has the potential to improve outcomes and reduce side effects for patients. However, determining if/when a patient may benefit from ART remains difficult. Deformable image registration (DIR) between the planning CT and daily CBCT images can enable estimation of delivered dose and aid in this decision-making process. Yet, this task is complicated by inaccurate Hounsfield Units (HUs) and limited field-of-view (FOV) of CBCT.

Purpose: This work developed and validated an open-source DIR algorithm for CT-to-CBCT registration which addressed the primary challenges facing this registration problem.

Methods: Twenty-five retrospectively selected head and neck (HN) patients, exhibiting various levels of weight loss during their treatment course, were chosen for the study. Several important CBCT preprocessing steps aided in overcoming the key challenges for the registration. Fifteen patients were used to develop the algorithm based on the open-source software `Elastix.` Parameterization of the algorithm relied on qualitative inspection of registrations. The remaining ten patients were used for validation of the algorithm which was based on Dice similarity coefficient (DSC), Jaccard index (JI), and Hausdorff distance (HD) for mandible, spinal cord, and parotids.

Results: Across all ten validation patients, the DSC was 0.78 and 0.71 for bony anatomy and soft tissue, respectively. The JI was 0.65 and 0.56, and the HD was 4.4mm and 3.6mm, for bony anatomy and soft tissue, respectively.

Conclusions: The developed method resulted in sufficient accuracy for the task of delivered dose calculation to inform ART decision making.

Introduction

Image guidance is a key tool in radiotherapy, improving geometric accuracy and allowing for advanced treatment techniques, such as adaptive radiotherapy (ART).

In head and neck (HN) treatments in particular, ART has been explored to address the significant anatomic changes commonly encountered over the course of treatment. Apart from intended tumor volume shrinkage, studies have reported marked reduction in parotid glands size[1,2], and overall weight loss from side effects like xerostomia, dysphagia, and loss of taste[3]. These anatomic changes have been linked to differences between delivered and planned doses, including increased hot spots, reduced target coverage, and increased dose to organs at risk (OARs)[4–7]. This has prompted research to determine what patients benefit most from ART. Some studies have used pretreatment indicators as predictors, with some success[8–11]. One study used deformable image registration (DIR) between the pCT and the CBCT to recalculate the dose at each fraction, which led to no missed necessary replans and fewer unnecessary ones[12]. Using DIR to deform the pCT to the CBCT is complicated, amongst other factors, by the limited CBCT FOV. In this study, we developed and validated a method for deforming a pCT to a CBCT without sacrificing the pCT field of view using an open-source DIR algorithm.

Methods

Patient Data

Fifteen HN patients with one pCT and three CBCTs each were retrospectively selected to develop the registration method. Patients were treated on a Truebeam Linac (**Varian Medical Systems, Palo Alto, CA**). Only patients with anatomic changes visible on CBCT were chosen.

Ten additional HN patients with the same criteria were retrospectively chosen to evaluate the registration method. A single CBCT from each patient with a large anatomic change relative to their pCT was chosen.

DIR Method

The developed method was implemented using a combination of MATLAB (**Mathworks, Natick, MA**) and `elastix`, an open-source DIR algorithm[13]. The pCT and CBCT were exported from MIM in mhd format. MATLAB scripts resampled and padded the CBCTs with air to the resolution and approximate size of the corresponding pCTs. Binary masks corresponding to the FOV of the CBCT were created to ensure the algorithm only used

information inside the FOV for calculating the rigid and deformable registration. Each registration (rigid transform followed by a B-spline transform) was completed in `elastix`.

	Rigid	Deformable
Image Pyramid	Gaussian Pyramid	Gaussian Pyramid
Sampler	Random	Random Coordinate
Interpolator	B-Spline	B-Spline
Optimizer	Adaptive Stochastic Gradient Descent	Adaptive Stochastic Gradient Descent
Transform	Euler Transform	B-Spline
Similarity Metric	Normalized Correlation Coefficient	Mutual Information
Regularization Term		Bending Energy Penalty

Table 1 Parameters used for rigid and deformable registration in elastix. The weights assigned to the mutual information metric and bending energy penalty were 1 and 25, respectively.

Registration parameters are shown in Table 1 and were based on those used in another study[14]. Deformable registration was performed using a B-spline transformation basis, defined by a uniform grid of control points overlaid on the fixed image (CBCT). Since a mask is equipped to indicate the CBCT FOV as the region of interest, the B-spline grid extends just outside of the mask, where the calculated transformation quickly goes to zero. Thus the anatomical information outside of the CBCT FOV in the dCT is not deformed but rigidly registered. The registration was performed with an anisotropic, multi-resolution image pyramid, such that in each of the three resolutions used, the superior-inferior dimension had less downsampling to account for the larger spacing inherent to the images. The deformable portion of the registration algorithm used a combination of local mutual information (similarity metric) and a bending energy penalty (regularization) term. The bending energy penalty term penalizes abrupt changes in the transformation to prevent the deformation from becoming irregular or folding in on itself. Weights for the two metrics were updated to improve performance on this dataset, and it was concluded that the two metrics in combination were superior to mutual information alone, after investigations with a single parameter resulted in inaccurate and excessive deformation.

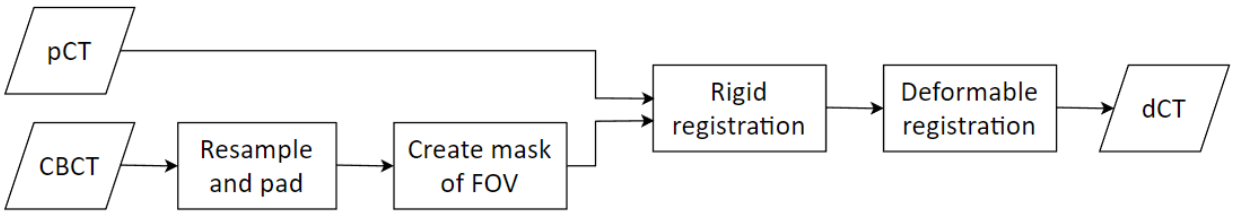


Fig 1: DIR flowchart

During development, the registration quality was assessed using the Jacobian map of the deformation vector field and visual inspection. Following a successful registration, a MATLAB script converted the dCT from the mhd image format used by `elastix` to DICOM for importing back to the treatment planning system (TPS).

Validating the DIR Method

The deformation method was evaluated through contour analysis. Contours were drawn on the CBCT by expert physicians. Parotids, mandible, and spinal cord were chosen due to their importance in treatment planning and outcomes. Planning CT contours were deformed according to their respective dCT transform parameters. Dice similarity coefficient (DSC), Jaccard Index (JI), and Hausdorff distance (HD) were used to quantitatively compare the deformed contours to the CBCT contours.

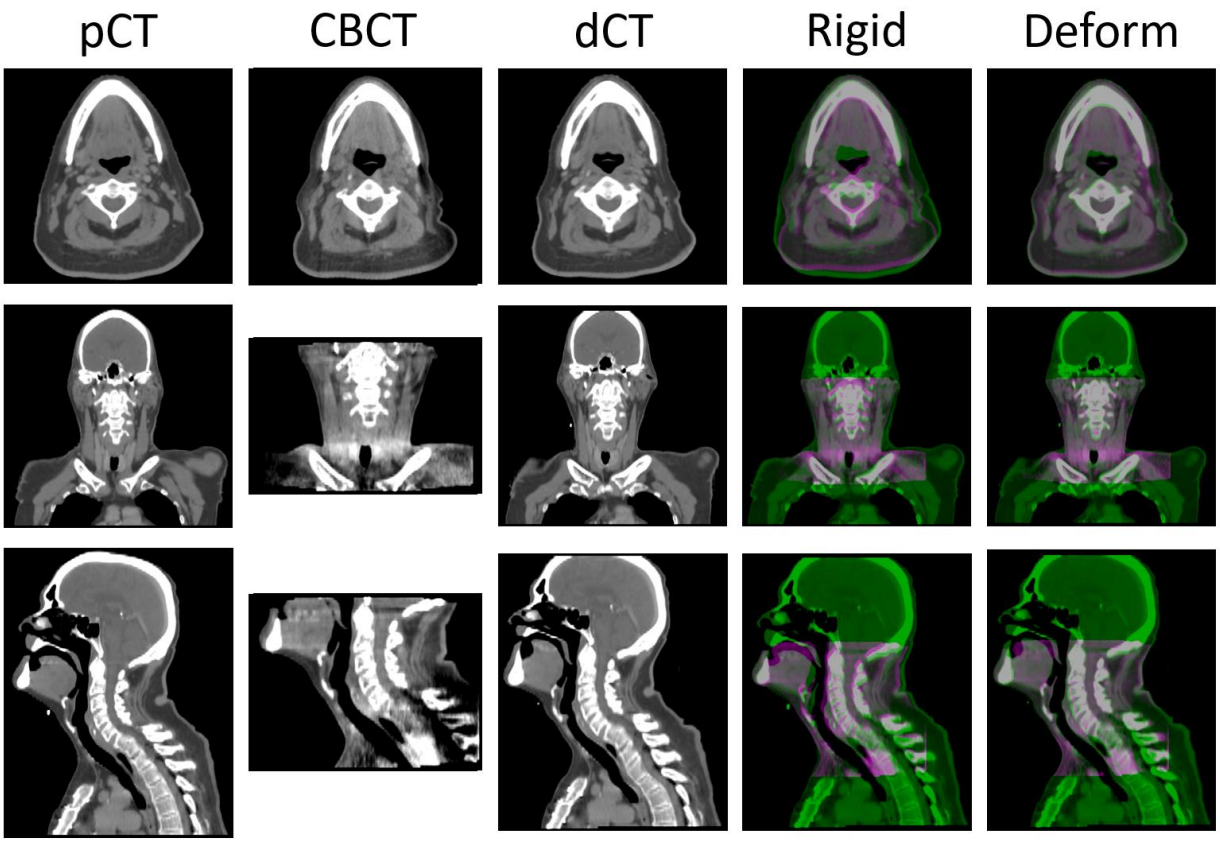


Fig 2: Axial, coronal, and sagittal views of the pCT, CBCT, dCT, rigid registration of the pCT to the CBCT, and rigid registration of the dCT to the CBCT.

Results

Upon visual inspection, the registrations and resultant dCTs appeared reasonable (see Figure 2). The DSC, JI, and HD results are summarized in Table 2.

The DSC values are 0.78 and 0.71 for bony anatomy and soft tissue, respectively. The JI values are 0.65 and 0.56, respectively. The DSC and JI for the mandible significantly ($p=0.002$, Student's t-test) outperformed that for the cord. The HD are 4.4mm and 3.6mm for bony anatomy and soft tissue, respectively.

	Mandible			Cord			Left Parotid			Right Parotid		
	DSC	JI	HD	DSC	JI	HD	DSC	JI	HD	DSC	JI	HD
Pt1	0.70	0.55	5.1	0.83	0.71	2.4	0.71	0.56	4.1	N/A	N/A	N/A
Pt2	0.85	0.73	5.8	0.71	0.55	3.9	0.62	0.45	3.1	0.50	0.33	3.3
Pt3	0.82	0.69	7.5	0.69	0.53	2.9	0.78	0.64	3.3	0.81	0.67	3.5
Pt4	0.78	0.65	4.9	0.75	0.59	2.4	0.67	0.50	3.5	0.39	0.24	4.1
Pt5	0.87	0.77	6.1	0.74	0.59	2.6	0.77	0.63	3.1	0.82	0.69	3.7
Pt6	0.88	0.78	4.4	0.73	0.57	4.6	0.75	0.60	3.9	0.77	0.63	3.5
Pt7	0.90	0.82	4.1	0.71	0.56	2.6	0.79	0.66	2.6	0.80	0.66	3.7
Pt8	0.86	0.76	6.1	0.77	0.62	3.1	0.83	0.72	3.7	0.78	0.64	3.7
Pt9	0.80	0.67	7.3	0.67	0.51	2.9	0.68	0.51	3.5	0.72	0.56	4.3
Pt10	0.75	0.61	6.9	0.81	0.68	2.6	0.55	0.38	3.1	0.65	0.48	4.1
Mean	0.82	0.70	5.8	0.74	0.59	3.0	0.72	0.56	3.4	0.69	0.55	3.8
Stdev	0.06	0.08	1.1	0.05	0.06	0.7	0.08	0.10	0.4	0.14	0.15	0.3
Min	0.71	0.55	4.1	0.67	0.51	2.4	0.55	0.38	2.6	0.39	0.24	3.3
Max	0.90	0.82	7.5	0.83	0.82	4.6	0.83	0.72	4.1	0.82	0.69	4.3

Table 2 Dice similarity coefficient (DSC), Jaccard index (JI), and Hausdorff distance (HD) resultsfor CBCT and deformed contours from the pCT for 10 patients. HD is reported in mm.

Discussion

We developed a deformable registration method to register a pCT to a CBCT which solved several common issues encountered for this task, including limited FOV and inaccurate Hounsfield Unit values.

The results for all contours are comparable to previous publications. While TG-132 sets a goal of achieving DSC of 0.8-0.9 for validating image registration methods, this is not achievable for all structures. Commercial algorithms have been found to fall short, specifically in HN, with an average DSC of 0.73 for HN OARs in one study directly assessing TG-132 recommendations[15]. In-house algorithms have achieved similar results, with reported DSC values of 0.85 for bony anatomy and 0.79 for soft tissue, or 0.83, 0.79, 0.74, and 0.72 for the mandible, spinal cord, and left and right parotid, specifically, and reported HD values ranging from 2.00 to 5.19 mm over soft tissue and bony anatomy[16–18]. However, these studies had crucial differences from this study including imaging modality, number of observers, and contour selection. A study investigating interobserver variability on HN CT found average DSC values of 0.90, 0.78, 0.82, and 0.83 for mandible, spinal cord, and left and right parotid, respectively, with a large majority of the observers following the same guidelines for all contours apart from the spinal

cord[19]. This indicates that our results are comparable to previous publications, despite limitations such as interobserver variability and difficulty delineating soft tissue structures on CBCT.

During validation, a limitation of our algorithm was identified, as the algorithm was unable to generate acceptable registrations when presented with extreme image artifacts caused by high density implants. An attempt to mask out the artifact so the registration would ignore it was unsuccessful. This may be due to the nature of the registration algorithm. Further work could focus on making the method more robust to artifacts by testing different reconstruction algorithms and masking methods. In its current state, the registration method is a viable option for dose calculation mid-treatment course to inform resimulating and replanning decisions. Patient with artifacts were excluded from this study.

Conclusions

DIR can be a useful tool for ART and related research. However, commercially available DIR software is expensive and may not offer the flexibility needed for specific tasks. This study used an open-source algorithm to address problems deforming pCTs to CBCTs. The resulting deformations are useful for dose calculation at multiple timepoints over the course of a patient's treatment.

References

1. Chen C, Lin X, Pan J, Fei Z, Chen L, Bai P. Is it necessary to repeat CT imaging and replanning during the course of intensity-modulated radiation therapy for locoregionally advanced nasopharyngeal carcinoma? *Jpn J Radiol*. 2013;31(9):593-599. doi:10.1007/s11604-013-0225-5

2. Lee C, Langen KM, Lu W, et al. Evaluation of geometric changes of parotid glands during head and neck cancer radiotherapy using daily MVCT and automatic deformable registration. *Radiother Oncol*. 2008;89(1):81-88. doi:10.1016/j.radonc.2008.07.006

3. Van Cutsem E, Arends J. The causes and consequences of cancer-associated malnutrition. *Eur J Oncol Nurs*. 2005;9:S51-S63. doi:10.1016/j.ejon.2005.09.007

4. Moallim M, Maungwe P, Chamunyonga C. CBCT-guided radiotherapy for locally advanced head and cancer: dosimetric impact of weight loss on VMAT and IMRT plans for selected organs at risk structures (OARs). *J Radiother Pract*. 2015;14(04):336-342. doi:10.1017/S1460396915000291

5. Ahn PH, Chen CC, Ahn AI, et al. Adaptive Planning in Intensity-Modulated Radiation Therapy for Head and Neck Cancers: Single-Institution Experience and Clinical Implications. *Int J Radiat Oncol*. 2011;80(3):677-685. doi:10.1016/j.ijrobp.2010.03.014

6. Hansen EK, Bucci MK, Quivey JM, Weinberg V, Xia P. Repeat CT imaging and replanning during the course of IMRT for head-and-neck cancer. *Int J Radiat Oncol*. 2006;64(2):355-362. doi:10.1016/j.ijrobp.2005.07.957

7. Nishi T, Nishimura Y, Shibata T, Tamura M, Nishigaito N, Okumura M. Volume and dosimetric changes and initial clinical experience of a two-step adaptive intensity modulated radiation therapy (IMRT) scheme for head and neck cancer. *Radiother Oncol*. 2013;106(1):85-89. doi:10.1016/j.radonc.2012.11.005

8. Brouwer CL, Steenbakkers RJHM, van der Schaaf A, et al. Selection of head and neck cancer patients for adaptive radiotherapy to decrease xerostomia. *Radiother Oncol*. 2016;120(1):36-40. doi:10.1016/j.radonc.2016.05.025

9. Surucu M, Shah KK, Mescioglu I, et al. Decision Trees Predicting Tumor Shrinkage for Head and Neck Cancer: Implications for Adaptive Radiotherapy. *Technol Cancer Res Treat*. 2016;15(1):139-145. doi:10.1177/1533034615572638

10. Brown E, Owen R, Harden F, et al. Predicting the need for adaptive radiotherapy in head and neck cancer. *Radiother Oncol*. 2015;116(1):57-63. doi:10.1016/j.radonc.2015.06.025

11. Brown E, Owen R, Harden F, et al. Head and neck adaptive radiotherapy: Predicting the time to replan. *Asia Pac J Clin Oncol*. 2016;12(4):460-467. doi:10.1111/ajco.12516

12. Vickress JR, Battista J, Barnett R, Yartsev S. Online daily assessment of dose change in head and neck radiotherapy without dose-recalculation. *J Appl Clin Med Phys*. 2018;19(5):659-665. doi:10.1002/acm2.12432

13. Klein S, Staring M, Murphy K, Viergever MA, Pluim JPW. elastix: A Toolbox for Intensity-Based Medical Image Registration. *IEEE Trans Med Imaging*. 2010;29(1):196-205. doi:10.1109/TMI.2009.2035616

14. Leibfarth S, Mönnich D, Welz S, et al. A strategy for multimodal deformable image registration to integrate PET/MR into radiotherapy treatment planning. *Acta Oncol*. 2013;52(7):1353-1359. doi:10.3109/0284186X.2013.813964

15. Latifi K, Caudell J, Zhang G, Hunt D, Moros EG, Feygelman V. Practical quantification of image registration accuracy following the AAPM TG-132 report framework. *J Appl Clin Med Phys*. 2018;19(4):125-133. doi:10.1002/acm2.12348

16. Veiga C, McClelland J, Moinuddin S, et al. Toward adaptive radiotherapy for head and neck patients: Feasibility study on using CT-to-CBCT deformable registration for "dose of the day" calculations. *Med Phys*. 2014;41(3):031703. doi:10.1118/1.4864240

17. Zukauskaite R, Brink C, Hansen CR, et al. Open source deformable image registration system for treatment planning and recurrence CT scans: Validation in the head and neck region. *Strahlenther Onkol*. 2016;192(8):545-551. doi:http://dx.doi.org/10.1007/s00066-016-0998-4

18. Jiang C, Huang Y, Ding S, et al. Comparison of an in-house hybrid DIR method to NiftyReg on CBCT and CT images for head and neck cancer. *J Appl Clin Med Phys*. 2022;23(3):e13540. doi:10.1002/acm2.13540

19. van der Veen J, Gulyban A, Willems S, Maes F, Nuyts S. Interobserver variability in organ at risk delineation in head and neck cancer. *Radiat Oncol Lond Engl*. 2021;16:120. doi:10.1186/s13014-020-01677-2

Vita

Sarah Elizabeth Holler was born on December 27, 1993 in Allentown, Pennsylvania. She graduated from Salisbury High School, Allentown, Pennsylvania in 2012. She received her Bachelor of Science in Physics from Bucknell University in Lewisburg, Pennsylvania, in May 2016. In August 2016, she entered the Medical Physics Graduate Program at Virginia Commonwealth University. While completing her degree at VCU, Sarah was co-author on two peer-reviewed research articles and the primary author on one oral presentation and three poster presentations at the national meeting of AAPM.