DOSIMETRY OF ^{90}Y LIQUID BRACHYTHERAPY IN A DOG WITH OSTEOSARCOMA USING PET/CT

A Thesis

by

JINGJIE ZHOU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2011

Major Subject: Health Physics

DOSIMETRY OF ^{90}Y LIQUID BRACHYTHERAPY IN A DOG WITH OSTEOSARCOMA USING PET/CT

A Thesis

by

JINGJIE ZHOU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Co-Chairs of Committee, Gamal Akabani
John W. Poston, Sr.
Committee Members, Mark W. Lenox
Head of Department, Raymond J. Juzaitis

May 2011

Major Subject: Health Physics

ABSTRACT

Dosimetry of ^{90}Y Liquid Brachytherapy in a Dog with Osteosarcoma Using PET/CT. (May 2011)

Jingjie Zhou, B.E., Tsinghua University

Co-Chairs of Advisory Committee: Dr. Gamal Akabani
Dr. John W. Poston, Sr.

A novel ^{90}Y liquid brachytherapy strategy is currently being studied for the treatment of osteosarcoma using a preclinical translational model in dogs to assess its potential efficacy and toxicity. In this study, dosimetry calculations are performed for ^{90}Y liquid brachytherapy in a dog with osteosarcoma using the Geant4 Monte Carlo code. A total of 611.83 MBq ^{90}Y radiopharmaceutical is administered via direct injections, and the *in vivo* distribution of ^{90}Y is assessed using a time-of-flight (TOF) PET/CT scanner. A patient-specific geometry is built using anatomical data obtained from CT images. The material properties of tumor and surrounding tissues are calculated based on a CT number - electron density calibration. The ^{90}Y distribution is sampled in Geant4 from PET images using a collapsing 3-D rejection technique to determine the decay sites. Dose distributions in the tumor bed and surrounding tissues are calculated demonstrating significant heterogeneity with multiple hot spots at the injection sites. Dose volume histograms show about 33.9% of bone and tumor and 70.2% of bone marrow and trabecular bone receive a total dose over 200 Gy; about 3.2% of bone and tumor and 31.0% of bone marrow and trabecular bone receive a total dose of over 1000 Gy. ^{90}Y liquid brachytherapy has the potential to be used as an adjuvant therapy or for palliation purposes. Future work includes evaluation of pharmacokinetics of the ^{90}Y radiopharmaceutical, calibration of PET/CT scanners for the direct quantitative assessment of ^{90}Y activity concentration, and assessment of efficacy of the ^{90}Y liquid brachytherapy strategy.

DEDICATION

I dedicate this dissertation to my family.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor and committee chair, Dr. Gamal Akabani for his patient guidance and enthusiastic support in my study. I would like to thank my committee co-chair, Dr. John W. Poston, Sr., and committee member, Dr. Mark W. Lenox, for their insightful comments and continuous help throughout my study and research.

I want to thank Dr. John Ford and Dr. Leslie A. Braby for teaching me about radiation biology, internal dosimetry, radiation physics and health physics instruments. I also want to thank other faculty and staff in the Department of Nuclear Engineering at Texas A&M University.

I would like to thank Dr. Theresa W. Fossum, director of the Texas A&M Institute for Preclinical Studies for her support of this project. My sincere thanks also go to Dr. Catherine Ruoff at the Texas A&M Institute for Preclinical Studies and Dr. R. Keith Frank at IsoTherapeutics Group, LLC for their kind assistance.

I am grateful to the authors and contributors of the Geant4 DICOM application, Louis Archambault, Luc Beaulieu, Vincent Hubert-Tremblay, Pedro Arce and Stephane Chauvie. I also want to thank Joseph Perl, Makoto Asai, Sebastien Incerti, Tatsumi Koi, and Dennis Wright from the SLAC Geant4 group.

I would like to thank my friends Fada Guan, Shaoyong Feng, Laura Rodriguez, Aundrie Blanchard and Vihar Bhakta for their generous help and support in my study and life. Special thanks go to my roommate, Wen Li, who always gives me the warmest encouragement.

Finally, deepest gratitude goes to all my family members. Without their love and support I wouldn't be able to complete this thesis.

NOMENCLATURE

1-D	One-dimensional
2-D	Two-dimensional
3-D	Three-dimensional
ASCII	American Standard Code for Information Interchange
CT	Computed Tomography
DCE-MRI	Dynamic Contrast-enhanced MRI
DICOM	Digital Imaging and Communications in Medicine
DVH	Dose Volume Histogram
^{18}F−FDG	^{18}F−Fluoro-2-deoxy-glucose
FOV	Field of View
FWHM	Full-width at Half Maximum
HU	Hounsfield Unit
LET	Linear Energy Transfer
MIP	Maximum Intensity Projection
MRI	Magnetic Resonance Imaging
mTOR	Mammalian Target of Rapamycin
PET	Positron Emission Tomography
SEER	Surveillance, Epidemiology and End Results
SPECT	Single Photon Emission Computed Tomography
SUV	Standard Uptake Value
TOF	Time-of-flight
TRT	Targeted Radionuclide Therapy
VEGF	Vascular Endothelial Growth Factor

TABLE OF CONTENTS

LIST OF TABLES

TABLE Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

Osteosarcoma is the most common primary malignant bone tumor, mainly affecting children, adolescents and young adults. The radiographic features, histological characteristics, clinical presentations and standard treatment of osteosarcoma have been described in many book chapters and review articles [1,2]. However, there has been little improvement in the survival and therapeutic strategies for osteosarcoma in the last 20 years. This leads to the investigation of novel treatment approaches and multi-modality strategies. This chapter reviews the general aspects of osteosarcoma, including epidemiology, histopathology, conventional treatment options, emerging therapies and preclinical animal studies.

1.1 Epidemiology of Osteoesarcoma

1.1.1 Incidence

The incidence rate of cancer increases with age. According to the analysis of the Surveillance, Epidemiology and End Results (SEER) Cancer Statistics Review of the National Cancer Institute 1975-2007, the overall incidence rate of cancer for all ages is 462.6 per 100,000 in the United States based on data from 2003 to 2007. For ages from 0 to 19, the incident rate of cancer is 16.7 per 100,000 during the same time period. The incidence rate of cancer for all ages has experienced an increase from 400.39 per 100,000 in 1975 to a peak as 510.73 per 100,000 in 1992 and then dropped to 472.68 per 100,000 in 2007. The incidence rate for ages from 0 to 19 shares the trend, exhibiting an increase in occurrence of childhood and adolescents cancer since 1970s [3]. The incident rate of cancer is higher in male than in female

This thesis follows the style of Nuclear Medicine and Biology.

for all cancer sites combined. Similarly, childhood and adolescents cancer affect boys more frequently than girls.

For all bone and joint cancers, the incidence rate is 0.9 per 100,000 for all ages, accounting for 0.19% of total cancer incidence for all ages but 5.39% for ages from 0-19. Among all malignant bone tumors for all ages, osteosarcoma is the most common type accounting for 35% of all cases, followed by chondrosarcoma (25%) and Ewing's sarcoma (16%), respectively. In children and adolescents, osteosarcoma represents about 56% of bone cancers and Ewing's sarcoma represents about 34% [2]. A primary peak in osteosarcoma incidence occurs in children and adolescents at ages 0-24, corresponding to the peak period of skeletal growth. A plateau of incidence is observed at ages 25-59, and there is a secondary peak of osteosarcoma incidence in the elderly (60-85+). The incidence rates of osteosarcoma is 0.44 per 100,000 for ages 0-24 and is 0.42 per 100,000 for ages 60-85+ [4]. The incidence of osteosarcoma has always been considered to be higher in males than in females, with a rate of 5.4 per million persons per year in males vs. 4.0 per million in females [2].

Osteosarcoma can occurs in any bone. However, the portions of skeleton with highest growth rate are most frequently affected. Osteosarcoma often occurs near the metaphyseal growth plates of the long bones of the extremities. The most common sites are the femur (42%), the tibia (19%), and the humerus (10%). Other likely locations are the skull or jaw, the vertebra, and the pelvis [2, 5].

1.1.2 Mortality and Survival

According to SEER Cancer Statistics Review 1975-2007, the mortality rate of all bone and joint cancers for all ages is 0.4 per 100,000, with a 5-year overall survival rate of 68.2%. For children at ages 0-14, the mortality rate is 0.1 per 100,000, and for children and adolescents at ages 0-19, the mortality rate is 0.2 per 100,000 [3]. Bone and joint tumor is the third cause of death in children, followed by leukemia and brain tumors [2].

The National Cancer Institute SEER Study for the years 1975-1995, reported that the overall 5-year survival rate for patients with osteosarcoma diagnosed between 1974 and 1994 was 63%. In another recent report based on data from 1973-2004, the relative 5-year survival rate was 61.6% for patients with ages 0-24, 58.7% for ages 25-59, and 24.2% for ages 60-85+ [4]. The strategy of using adjuvant chemotherapy with surgery has greatly improved the survival rate of osteosarcoma in the past decades. Survival increased significantly between 1973-83 and 1983-93, but there is little change subsequently [4]. Therefore, new treatment strategies need to be developed.

1.2 Histopathology

Osteosarcoma develops from propagation of malignant mesenchymal tumor cells which have the ability to produce osteoid extracellular matrix or immature bone. Conventional osteosarcoma can be divided into three categories according to the predominant matrix product of the tumor cells: 1) osteoblastic, 2) chondroblastic, and 3) fibroblastic. A mixture of production pattern is usual and small biopsies may not represent the overall histology of a tumor [1].

Generally, the tumor is a combination of bony and soft tissues showing significant heterogeneity, with areas varying in color, consistency, and degree of ossification. Osteosarcoma exhibits an invasive and bone-destroying pattern of growth. The borders between the tumor tissue and adjacent structures are indistinct and irregular. The small lesions are usually eccentrically located within the medullary cavity. The advanced lesions tend to invade the cortex, fill the medullary, progress toward the shaft and growth plate, and expand into soft tissue. Interactions between tumor osteoid and preexisting non-tumor bone produce large, densely ossified areas which may merges with adjacent cortex. Further growth of tumor produces an ball-like mass that overlies the large cortical defect and compresses surrounding muscles. Signs of periosteal reaction can often be seen at the periphery of the tumor lesion [1].

1.3 Imaging Assessment

Conventional radiography is usually used in the initial evaluation of the bone lesions. The radiographic presentation of osteosarcoma is usually a combination of osteolytic and sclerotic lesions with gradual transition to normal bones, which makes the borders of the lesions hard to define. The mineralization of tumor matrix exhibits cloudy opacities, which can be either uniformly distributed or clustered in one area. As previously described, the cortex is often involved and destroyed, resulting indistinct outer limits of the cortex in the area of penetration. As the tumor extends to soft tissue, its mineralization shadow can be seen overlying the area of penetration [1].

Magnetic resonance imaging (MRI) is employed in local staging because of its superb capabilities to evaluate the extent of intramedullary involvement and extension into soft tissue. Dynamic contrast-enhanced MRI (DCE-MRI) is used to evaluate the status of tumor angiogenesis for treatment planning [2].

Detailed examination of the lungs and bones is required to determine the presence of clinically detectable metastases. Computed tomography (CT) can be used as aid for conventional radiography in diagnosis of osteosarcoma in tubular and complex bones, but its primary role is to detect pulmonary metastases.

$^{18}\mathrm{F}-\mathrm{FDG}$ PET is often performed for the whole body to detect metastatic spread and local recurrence. ^{18}F-fluoro-2-deoxy-glucose is the most commonly used positron emission tracer in PET oncology. After administration, $^{18}\mathrm{F}-\mathrm{FDG}$ accumulates in locations with high glucose metabolism. Since tumors are hypermetabolic, the standard uptake value (SUV) of $^{18}\mathrm{F}-\mathrm{FDG}$ can be a sensitive indicator of malignancy [6]. Questions concerning the appropriate SUV cut-off value persist, but generally those lesions with an SUV greater than 2.0 are malignant (aggressive), and those below 2.0 are benign [7]. $^{18}\mathrm{F}-\mathrm{FDG}$ PET scans can also be used to assist in tumor staging and guide biopsy to the most biologically active region [2].

1.4 Metastasis and Recurrence

Osteosarcoma metastasizes through hematogenous spread [8]. At diagnosis, 80% of the cases are localized osteosarcoma while 20% of the patients are found with metastasis. Lungs are the most common metastatic sites, followed by other bones [9]. A retrospective study was carried out in Italy by Bacci et al. regarding 1,148 patients in 27 years period. 0.4% of patients who relapsed had local recurrence, 12% had local recurrence plus metastases, and 88% had metastases only [10]. The relapse rate for patients with inadequate surgical margins was significantly higher than those with adequate surgical margins. The first site of metastases was the lung with 89% of cases, 8% of patients had metastases in other bones and only 2% had metastases in other organs. The average time to relapse was 21.3 months [10].

1.5 Current Treatment Options

1.5.1 Surgery

Generally, surgery for osteosarcoma is a wide excision which removes the entire tumor and a cuff of adjacent normal tissue [11]. A more radical excision can be an amputation or disarticulation. Historically, most patients were treated with amputation. Limb-salvage surgery is now an alternative option, which spares more normal tissue and enables better reconstruction outcomes. The choice of surgery type depends on location and extent of disease and response of primary tumor to preoperative chemotherapy. For patients with metastases, especially pulmonary metastases, surgery to remove all sites of disease must be considered. Reconstruction of the limb is performed after surgery. Common options of reconstruction include osteoarticular allograft, prosthesis, or a combination of the two [11].

1.5.2 Chemotherapy

Advances in chemotherapy over the past 30 years have improved limb salvage outcome and led to higher survival rates. Surgery with adjuvant modern multi-drug chemotherapy has dramatically increased the 5-year disease-free survival rate of patients to 60-70% [9]. Chemotherapy has also been shown to reduce or delay the onset of pulmonary metastases. Agents commonly used in chemotherapy for osteosarcoma includes methotrexate (MTX), cisplatin, doxorubicin, and ifosfamide (IFO) [9].

For patients with non-metastatic osteosarcoma, current standard strategy is preoperative chemotherapy before surgery plus postoperative chemotherapy after surgery. Preoperative chemotherapy induces necrosis in the primary tumor which facilitates surgical resection and provides early treatment of micrometastatic diseases [12]. The postoperative chemotherapy or maintenance chemotherapy is carried out after the surgery. The postoperative chemotherapy should be chosen based on the degree of tumor necrosis induced by preoperative chemotherapy. The role of chemotherapy dose intensity in osteosarcoma has been widely debated. However, no relation between dose intensity and prognosis has been established so far [9].

For patients already diagnosed with metastases (synchronous metastatic osteosarcoma), the same chemotherapy strategy is used, but the prognosis of patients is poor. For patients with relapse, there is no standardized treatment so far. Resection of all metastatic sites is preferred, while the role of chemotherapy after relapse is still under debate. For patients not eligible for metastasectomy, chemotherapy is used for palliation purposes [9].

Recent improvements in chemotherapy have been achieved based on the understanding of the mechanism of resistance to agents in osteosarcoma cells. For example, the chemotherapy agent trimetrexate has been used in combination of high dose methotrexate in clinical trials. The rationale of this strategy is that trimetrexate works on transport-defective cells while methotrexate works on transport-normal cells. The non-overlapping toxicities of the two agents allow administrations at maximum dose intensity [11].

1.5.3 External Radiation Therapy

It is rare to use radiotherapy as a primary treatment for osteosarcoma. However, radiotherapy is an important option for patients with unresectable tumors and for palliation purposes. The most common modality of radiotherapy for osteosarcoma is external photon therapy, while experiences with proton, neutron and heavy ions radiotherapy for osteosarcoma have been very limited. An analysis was performed by Schwarz *et al.* based on patient data from the Cooperative Osteosarkomstudiengruppe (COSS)-Registry, which included 100 patients treated with radiotherapy for osteosarcoma [13]. A total of 94 patients received external photon therapy; 2 patients, proton therapy; 2 patients, neutron therapy; and 2 patients, intraoperative radiotherapy. The median dose for external RT was 55.8 Gy (ranging from 30 Gy to 120 Gy). The local control rates were calculated with a median follow-up of 1.5 years. Local control rates for combined surgery and radiotherapy were significantly better than those for radiotherapy alone, which demonstrates the benefit of combined treatment modalities.

1.5.4 Targeted Radionuclide Therapy

Targeted radionuclide therapy (TRT) uses unsealed radiolabeled compounds to deliver therapeutic doses of ionizing radiation to tumor sites [14]. Such radiopharmaceutical formulations may be solutions or colloidal suspensions and the administration can be either systemic or locoreginal. Radiopharmaceuticals used for TRT are designed to target widely disseminated lesions, such as secondary tumors or metastases [14]. Targeted radionuclide therapy has been used for the treatment of cancers for several decades, and current technological advances in molecular biology and biochemistry have enabled the production of radiopharmaceuticals with sophisticated molecular carriers with enhanced capability of targeting tumor cells [15].

Primary issues in the design of TRT agents include selection of radionuclides, efficiency of drug delivery, residence time of radioactivity at target sites and clearance time in normal tissues, and *in vivo* catabolism and metabolism of the drug. The choice of a radionuclides depends on its physical half-life, decay scheme, radiative properties, pharmacokinetics of the radiopharmaceutical, cost, and availability [15]. Radionuclides used in radiopharmaceutical therapy include β-particle emitters such as ^{90}Y and ^{188}Re, α-particle emitters such as ^{211}At, and Auger electron emitters, such as ^{125}I. Radionuclides with β-particle emissions are the most commonly used in radiopharmaceutical therapy. β-particles have a continuous energy spectrum ranging from zero to a maximum value. These electrons have relatively longer ranges compared to α-particles, so they are able to produce a spread out dose distribution in the tumor. On the other hand, α-particles have short ranges of a few cell diameters, so the dose deposition is highly localized, thus sparing surrounding normal tissues. With a relatively high LET about 100 keV/μm, α-particles also produce more direct damage to the tumor cells. Auger electrons deposit energy within the diameter of a cell. Therefore, the auger electron emitting radiopharmaceuticals need to be bound to the DNA in the tumor cell nucleus for efficient cell killing. However, the design of effective Auger-emitting targeting agents remains a great challenge [15].

A number of radiopharmaceuticals have been used in the treatment of human bone metastatic disease, including ^{32}P-sodium orthophosphate, $^{89}\mathrm{Sr}-\mathrm{chloride}$ and ^{153}Sm-EDTMP which have been approved by the FDA for rountine use in humans, $^{186}\mathrm{Re}-\mathrm{HEDP}$ which has been approved in Europe, and $^{188}\mathrm{Re}-\mathrm{HEDP}$ which has been undergoing clinical trials but not yet approved by the FDA. The administration of radiopharmaceuticals can be driven by site-directed molecular carriers or depend on specificity in organ uptake, such as iodide uptake in thyroid cells. For bone cancers, the accumulation of radioactivity in sites of growing and remodeling bone is determined by the chemical properties of the so-called "bone seekers" radionuclides. ^{32}P-sodium orthophosphate and $^{89}\mathrm{Sr} - \mathrm{chloride}$ are the first two agents used in the treatment of painful bone metastases, while the former is rarely used today. ^{153}Sm-EDTMP, $^{186}\mathrm{Re} - \mathrm{HEDP}$ and $^{188}\mathrm{Re} - \mathrm{HEDP}$ are newly introduced bone seeker agents which have been intensively studied in clinical trials. Even though the biophysical properties and pharmacokinetic profiles of these agents vary from each other, they have comparable efficacy in pain relief and similar acceptable hematological toxicity [16].

^{153}Sm-EDTMP has been used in the treatment of osteosarcoma as an addition to external beam therapy or chemotherapy, giving an extra dose to the tumor, while the benefit of the strategy is still under investigation [17]. In the analysis of COSS-Registry data by Schwarz *et al.*, 100 patients with osteosarcoma were treated with external radiation therapy and 17 of them received an extra dose of ^{153}Sm-EDTMP. Local control for patients given an addition of ^{153}Sm-EDTMP was worse than those who received external radiation therapy alone, though not statistically significant. The survival rate for patients with additionally administered ^{153}Sm-EDTMPF was not significantly different in comparison to that of other patients [13].

1.6 Emerging Therapies

1.6.1 Biological Target Therapies

Biological targeted therapy aims at blocking genes and signal transduction pathways responsible for the malignant behavior in osteosarcoma. A number of signal transduction pathways and receptors have been intensively studied as potential targets for the treatment of osteosarcoma, such as mTOR pathway, IGF-1 receptor, HER2/neu and VEGF receptor.

mTOR (mammalian target of rapamycin) pathway mediates cell growth and differentiation [11]. mTOR inhibitors have been shown to be effective against osteosarcoma cells in preclinical studies [9]. IGF-1 (insulin-like growth factor-1) is a major regulator of normal cell growth. It has been demonstrated that several key signal transduction pathways downstream of the IGF-1 receptor are activated in osteosarcoma cell lines. Several agents targeting on IGF-1 receptor are being tested in Phase I and II clinical trials [11]. HER2/neu is a member of the epidermal growth factor receptor family. The over expression of HER2/neu has been identified in a variety of carcinomas, including breast, ovarian, endometrial, prostate and lung. However, the prognostic significance of HER2/neu in osteosarcoma remains controversial. A Phase II clinical trial using the monoclonal antibody tratuzumab that targets the HER2/neu receptor, as an addition to standard chemotherapy in patients with HER2/neu expressing osteosarcoma is being conducted by the Children's Oncology Group [18]. The vascular endothelial growth factor (VEGF) is a naturally occurring protein that stimulates the development of tumor-associated blood vessels necessary for the tumor growth and metastases [9]. Inhibitors of the VEGF receptor are being studied for their efficacy of preventing tumor angiogenesis in various cancers, including osteosarcoma [11].

Therapies that focus on the tumor interaction with its microenvironments are also studied. For example, bisphosphonates have been studied with in *in vitro* osteosarcoma models, showing the capacity to inhibit tumor growth and to disrupt the tumor interactions with its surrounding microenvironment [11]. The adjuvant benefit of bisphosphonates administrated as an addition to chemotherapy has also been demonstrated both *in vitro* and *in vivo*. However, further clinical investigations are needed to assess the anti-tumor efficacy of bisphosphonates in human osteosarcoma [19].

1.6.2 α-particle Emitting ^{223}Ra Radionuclide Therapy

The α-particle emitting ^{223}Ra is a promising agent for targeted radionuclide therapy for bone cancer and bone metastases and it has been widely tested in clinical trials. The radionuclide ^{223}Ra produces a cascade of α-particles from decaying daughter radionuclides with a suitable half-life (11.34 d). The half-lives of its daughters are relatively short compared to other α-particle emitting radium radioisotopes, eliminating the translocalization problem of daughter products [20].

Preclinical studies have shown that cationic ^{223}Ra is an avid bone-seeker with high uptake in metabolically active osteoblastic bone and tumor lesions and little uptake in soft tissue. The daughter radionuclides can be well retained with minimal translocalization [20]. ^{223}RaCl$_2$ has been used in clinical trials for the treatment of skeletal metastases from breast and prostate cancers, resulting in impressive antitumor activity and improved overall survival [20]. Liposome-encapsulated ^{223}Ra has also been evaluated for the treatment of osteosarcoma [20]. The biodistribution and tumor uptake of ^{223}Ra was assessed in preclinical studies using mice with human osteosarcoma xenografts and dogs with spontaneous osteosarcoma and associated soft tissue metastases. Considerably higher uptake of ^{223}Ra was observed in dogs in both calcified and non-calcified tumor metastases in different organs than in normal soft tissues [20].

1.6.3 ^{90}Y Liquid Brachytherapy

The radionuclide ^{90}Y is a β-particle emitter that decays to stable daughter ^{90}Zr with a half-life of 64.2 hours and an average energy of 0.934 MeV. The maximum range of ^{90}Y β-particles in water is 1.1 cm. ^{90}Y is available in a convenient generator system and is able to be incorporated into many compounds, including complexes, chelates and monoclonal antibodies [21]. ^{90}Y labeled therapeutic microspheres have been approved for the routine use in human to treat primary and metastatic hepatic cancers. As one of the many bone seeker radioisotopes, ^{90}Y has also been considered for the treatment of bone cancers and metastases. A few attempts have been made to evaluate the pharmacokinetics of ^{90}Y labeled radiopharmaceutical. *In vivo* studies in mice showed desirable high uptake in bone and fast clearance in soft tissues and other organs, suggesting the feasibility of using ^{90}Y for the treatment of bone cancer and metastases [22, 23].

A novel ^{90}Y liquid brachytherapy agent is currently being studied for the treatment of osteosarcoma. Liquid brachytherapy is a new technique recently introduced to the field of radiopharmaceutical therapy [24]. Therapeutic radioisotopes are administrated via direct injections to tumor sites to minimize the activity loss and consequent hematological toxicity. In the case of osteosarcoma, since ^{90}Y is a bone seeker, we expect a high retention of radioactivity in the osteoblastic tumors and metastases and low uptake in soft tissues. As mentioned before, radiotherapy has been always an alternative for patients with unresectable tumors and for palliation purpose. For the disseminated tumor which is difficult to treat with external beam therapy, ^{90}Y agent should be able to deliver a localized high dose to the tumor and spare the surrounding normal tissue, considering the range of ^{90}Y β-particles.

Another advantage of ^{90}Y liquid brachytherapy is that the biodistribution of ^{90}Y can now be assessed non-invasively using a PET scanner. The decay of ^{90}Y contains a minor branch to the 0^+ first excited state of ^{90}Zr, followed by a transition of internal pair production [25]. The maximum energy of positrons was determined to be 800 keV. The branching ratio of positron emission was measured as $(31.8 \pm 0.47) \times 10^{-6}$ by Selwyn *et al.* [26]. Modern PET scanners using time-of-flight (TOF) reconstruction are able to assess the biodistribution of ^{90}Y radiopharmaceutical with good image quality. Lhommel *et al.* showed that TOF PET image of ^{90}Y better reflected the tumor heterogeneity assessed by FDG PET/CT than traditional bremsstrahlung ^{90}Y-SPECT [27]. PET imaging allows for the dosimetry of ^{90}Y liquid brachytherapy in the tumor and surrounding tissues.

1.7 Preclinical Studies Using Dogs with Osteosarcoma

1.7.1 Similarities Between Dog and Human Osteosarcoma

Animals have been widely used in the studies of a variety of diseases, including cancer, for a better understanding of the similar processes in humans. Spontaneously developing osteosarcoma in dogs is an excellent translational model for osteosarcoma in humans due to the similarities they share in many aspects, including male sex predilection, affected sites, pathology and treatment response.

Osteosarcoma in dogs is much common than in humans. Prevalence estimates suggest that over 8000 new cases occur each year in United States. Most affected dogs are large or giant purebreds. 90% of canine osteosarcoma patients are greater than 20 kg. The peak age of incidence of osteosarcoma in dogs is seven years. The incidence rate of osteosarcoma in dogs is higher in male than that in female, which is same as the situation in humans. For both humans and dogs, the most common site of osteosarcoma is the metaphyseal growth plates in long bones of the limbs. The majority of human tumors occurs in legs, while most of the canine cases occur in the front legs [28].

The most common metastatic site for both species is lungs, followed by other bones. Osteosarcoma in humans and dogs have similar histopathological features. Both species have a mixed matrix production pattern in the tumor and significant variation for the type and size of involved cells. More important, informative comparison of human and canine osteosarcoma demonstrated a strong similarity of the genomic expression pattern of the two [29].

Treatment options for dogs with osteosarcoma includes surgery, chemotherapy and radiotherapy. The response to treatment in dogs is also similar to that of humans. Both exhibits improved survival rate with adjuvant chemotherapy and high metastases rate with surgery or radiotherapy alone.

1.7.2 Translation of Therapies from Dog to Human

Preclinical studies in dogs have played an important role in the development of new cancer treatment strategies over the past decades. Compared to tumor mouse models, spontaneous dog models are more similar to human tumor in its growth time period, genomic expression, heterogeneity in tumor cells and tumor microenvironment [30]. Therefore, preclinical studies in dogs with osteosarcoma can assist in assessing the safety and efficacy of treatment strategies and predicting their response in human patients. As an example, dogs with osteosarcoma were used to assess the preoperative and postoperative treatment to optimize limb-sparing surgical procedures, which is currently used in the management of osteosarcoma in human patients [30].

Preclinical studies of ^{90}Y liquid brachytherapy are being performed using dogs with osteosarcoma. The results from these studies are expected to contribute to our understanding of the safety and efficacy of the ^{90}Y radiopharmaceutical. In this study, we performed three-dimensional (3D) dosimetry calculations of ^{90}Y liquid brachytherapy in dogs with osteosarcoma to assess its dose distribution in the tumor bed and surrounding tissues.

2. MATERIAL AND METHODS

2.1 ^{90}Y Liquid Brachytherapy Preclinical Protocol

2.1.1 Diagnostic ^{18}F-FDG PET/CT Scan

A diagnostic ^{18}F-FDG PET/CT scan is performed to detect the distribution of tumor using a Siemens Biograph mCT. The scanner is equipped with 128 slice molecular CT and high resolution TOF PET with extended field-of-view (FOV).

An example of the co-registered ^{18}F-FDG PET/CT 3-D maximum-intensity-projection (MIP) is shown in Figure 2.1. The tumor exhibits a group of heterogenous hot lesions with irregular borders. The average SUV in the volume of interest is 6.00 with a maximum of 36.23, suggesting a very high activity of glucose metabolism in the tumor.

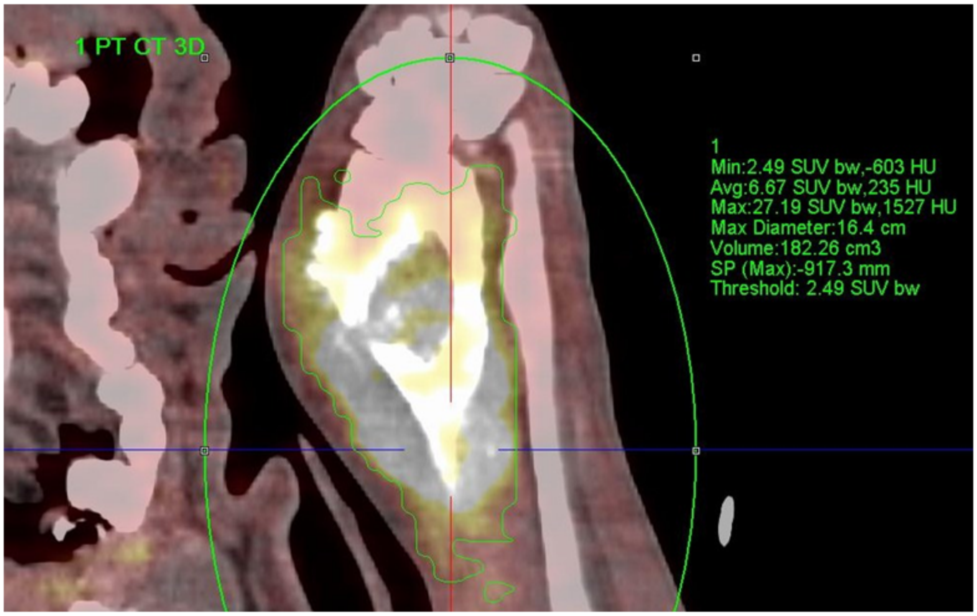


Fig. 2.1. ^{18}F-FDG PET/CT co-registered 3-D MIP view showing the tumor in a dog with osteosarcoma.

Figure 2.2 is a 3-D rendering image of ^{18}F-FDG PET/CT for a dog with osteosarcoma, which clearly shows the cortex has been penetrated and destroyed. The 3-D rendering view helps physicians to delineate the target treatment area on the dog's skin and predefine the locations of injection sites.

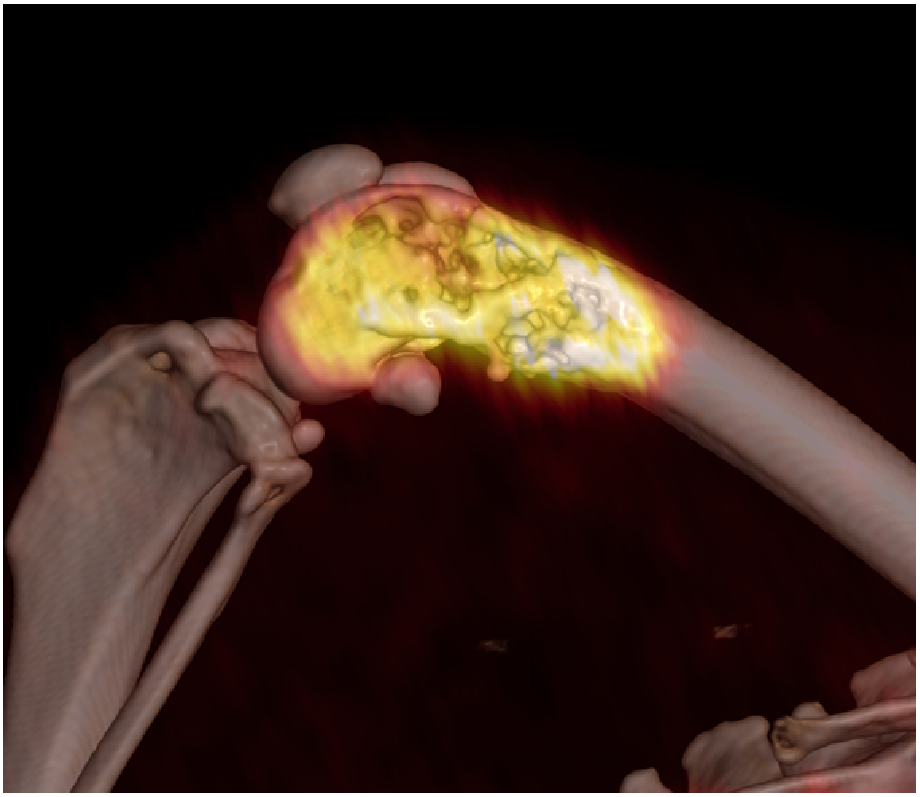


Fig. 2.2. ^{18}F-FDG PET/CT co-registered 3-D view showing the tumor in a dog with osteosarcoma.

2.1.2 Preparation of Injection Sites

The ^{90}Y radiopharmaceutical is administrated at multiple injection sites to cover as much as possible the total tumor volume. The injection sites are distributed uniformly in the delimitated treatment area, spaced approximately 1 cm from each other. After an injection site is chosen, a needle-like adaptor is inserted to touch the bone. The adaptor consists of three parts: a) an outer cannula, b) an inner cannula, and c) a cap. A specially designed miniature electric drill is used to drill on the osteoid tumor bed through the adapter to allow the direct injection of the radiopharmaceutical, as shown in Figure 2.3.

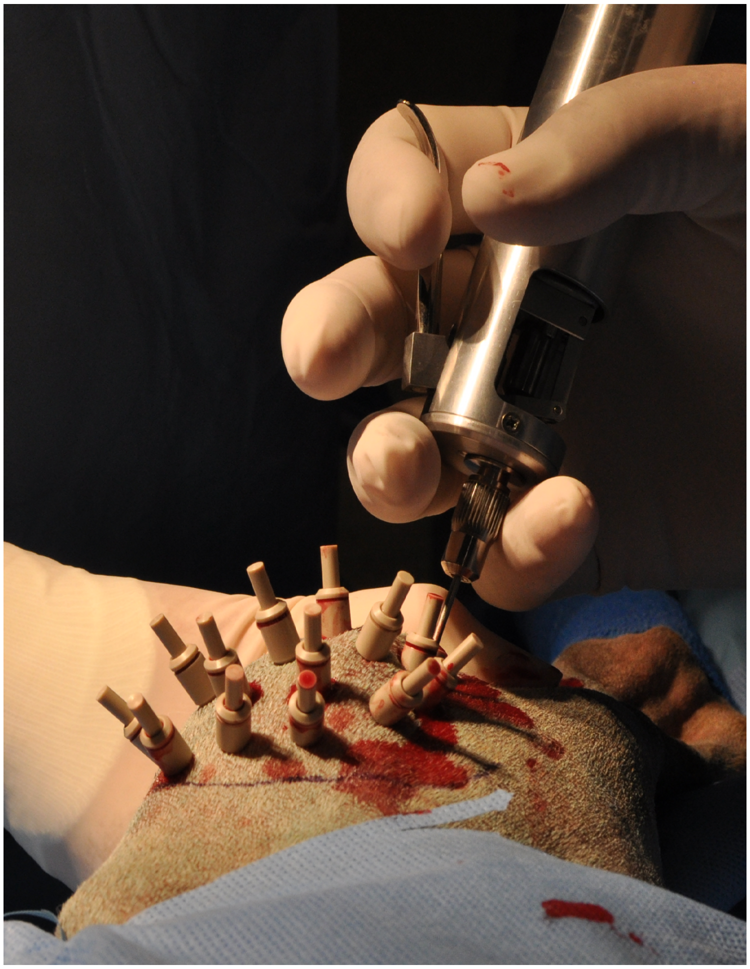


Fig. 2.3. Preparation of injection sites for ^{90}Y liquid brachytherapy in a dog with osteosarcoma. Before the administration of the radiopharmaceutical, adaptors are inserted at multiple injection sites distributed around the delimitated treatment area. A specially designed miniature electric drill is used to drill on the osteoid tumor bed through a adaptor.

After drilling, the adaptor is left in situ to stop blood from filling up the orifice and the cap is inserted to isolate the wound from air-borne contaminants.

After all the adaptors are inserted, a radiograph is taken to examine the placement and integrity of the adaptors to make sure the tumor volume is completely covered, as shown in Figure 2.4.

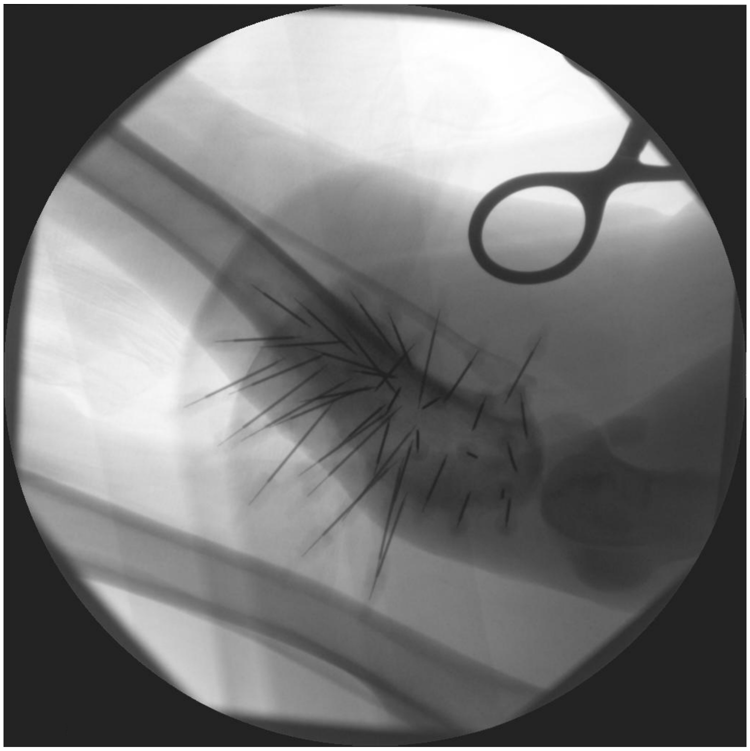


Fig. 2.4. Adaptor placement radiograph before injection of the ^{90}Y liquid brachytherapy in a dog with osteosarcoma.

2.1.3 ^{90}Y Radiopharmaceutical Injection

The ^{90}Y radiopharmaceutical formulation is prepared as solution with an activity concentration about 100 μCi/μl (3.7×10^4Bq/μl). The solution is usually dyed in blue so that it can be easily visualized in the syringe. A micro-needle syringe is used to perform the injection. For each adaptor, about 15 μl of ^{90}Y radiopharmaceutical is injected at 3 or 4 spots at different depths, spaced about 1 cm from each other. The adaptor is removed after the injection.

The volume of ^{90}Y solution in the syringe before and after each injection is recorded to determine the total volume injected. A picture of the ^{90}Y injection is shown in Figure 2.5.

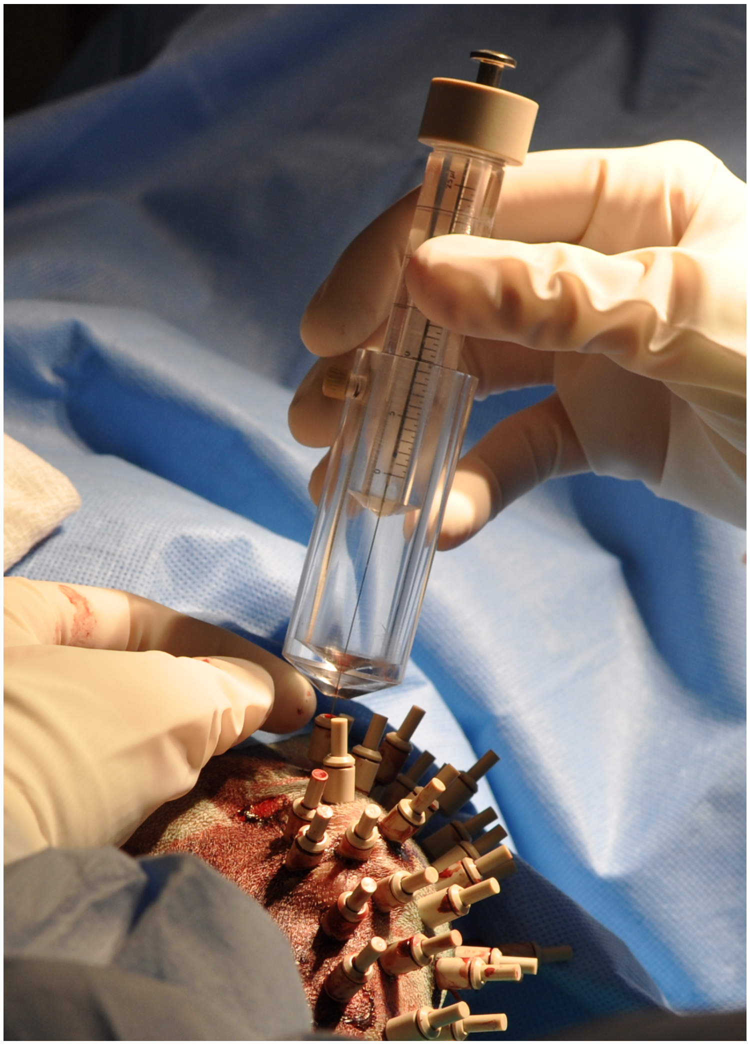


Fig. 2.5. The injection of ^{90}Y solution for ^{90}Y liquid brachytherapy in a dog with osteosarcoma. ^{90}Y solution was injected using a micro-needle syringe through a cannula.

2.1.4 ^{90}Y PET/CT Scan

A ^{90}Y PET/CT scan is performed with a Siemens Biograph mCT PET/CT scanner after the injection is finished to assess the *in vivo* activity distribution of ^{90}Y. The ^{90}Y PET scan usually lasts 1 hour. The PET image reconstruction method is ultraHD which uses TOF information.

2.2 The Conversion of CT Number to Material Properties

2.2.1 CT number - Electron Density Calibration

In this study, CT images in DICOM format are used to build a patient specific geometry for the Monte Carlo simulations. Anatomic information in the CT images is characterized by CT numbers (Hounsfield Unit, HU). However, mass density and elemental composition are required to define a material in most Monte Carlo codes. Thus, a method to convert CT numbers into material properties needs to be developed. The Hounsfield unit H assigned to a voxel is given by the equation:

$$H = (\frac{\overline{\mu}}{\overline{\mu}_{H_2O}} - 1)1000, \tag{2.1}$$

where $\overline{\mu}$ is the total attenuation coefficient of the tissue in that voxel averaged over the energy spectrum of the CT x-ray. It is assumed that the total attenuation has been corrected for the beam hardening effect, i.e. the energy spectrum is location independent. For a compound or mixture, the total attenuation is given by:

$$\overline{\mu} = \rho N_A \sum_{i=1}^{n} (\frac{\omega_i}{A_i}\overline{\sigma_i}), \tag{2.2}$$

where ρ is mass density of the material, N_A is the Avogadro constant, ω_i is the elemental weight of element i, A_i is the mass number of element i, and $\overline{\sigma_i}$ is the photon interaction cross section of element i averaged over the x-ray energy spectrum.

Because Compton effect is the predominant interaction process for photons at CT x-ray energies (from 50 keV to 200 keV), $\overline{\sigma_i}$ can be approximated as below [31]:

$$\overline{\sigma_i} \approx \overline{\sigma}_{incoh} = Z_i \overline{\sigma}_{KN}, \tag{2.3}$$

where $\overline{\sigma}_{incoh}$ is the average cross section of incoherent scattering (Compton scattering), Z_i is the atomic number of element i , $\overline{\sigma}_{KN}$ is the average Klein-Nishina cross section, which is a constant for a particular energy spectrum. Thus,

$$\overline{\mu} \approx \overline{\sigma}_{KN} \rho N_A \sum_{i=1}^{n} (\frac{\omega_i}{A_i} Z_i). \tag{2.4}$$

Note that $\rho N_A \sum_{i=1}^{n}(\frac{\omega_i}{A_i} Z_i)$ is the electron density of the material. Although equation 2.4 is a rough approximation, it is natural to relate CT number to electron density of the material.

Parker et al. suggested a linear relationship between CT number and electron density of human tissues with a flex point around HU = 65 [32]. Similar results were found by other authors [33–36]. Differences between authors were observed, especially in high CT numbers corresponding to bone materials, due to differences in scanners and tissue equivalent materials used for the calibrations. Therefore it is recommended that each CT scanner should be calibrated with an electron density phantom to accurately measure the relationship between CT number and electron density.

In this study, a CIRS (Computerized Imaging Reference Systems, Inc.) 062 electron density phantom is used for the CT number - electron density calibration. The phantom consists of a two-part body made of soft tissue equivalent epoxy resin, holding eight pairs of tissue equivalent inserts, and a solid water insert with known electron densities, as shown in Figure 2.6. For each pair of tissue equivalent inserts, one of the inserts is placed in the inner plate while the other in the outer ring. The solid water insert is placed in the center of the phantom.

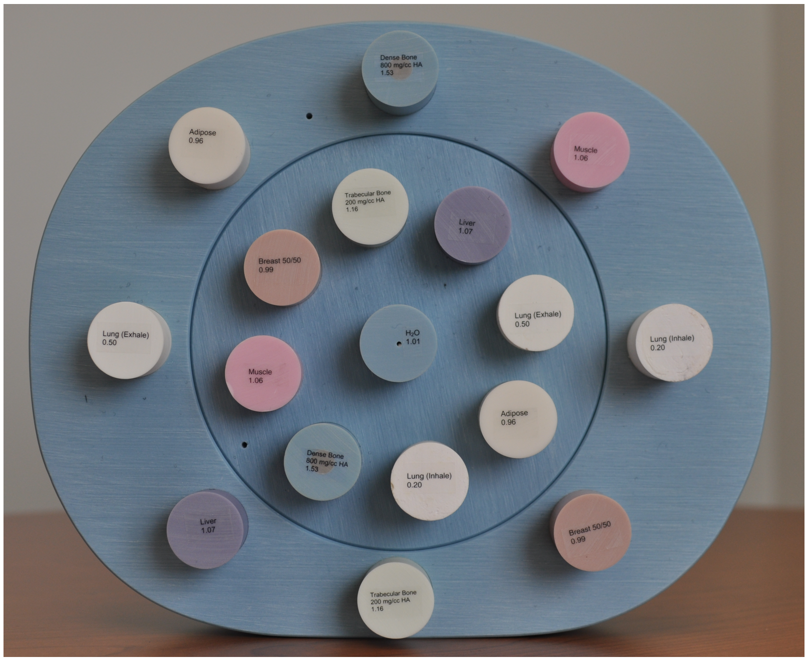


Fig. 2.6. CIRS 062 electron density phantom used for the CT number - electron density calibration. The phantom contains a two-part body, eight pairs of tissue equivalent inserts and a solid water insert. Tissue equivalent inserts include lung (exhale and inhale), adipose tissue, breast, liver, muscle, trabecular bone and dense bone.

The phantom is scanned using the Siemens Biograph PET/CT scanner. For each insert, the average CT number is determined by drawing a circular ROI with a diameter of approximately 2 cm on a central slice CT image. The CT number for each tissue equivalent material is the average of the inner plate and outer ring measurements. A calibration curve is obtained by plotting the average CT number against relative electron density (electron density normalized to that of water). Once the calibration is established, the relative electron density of each voxel in patient CT images can be calculated using linear interpolation.

2.2.2 Material Grouping and Density Bins

The relevant materials in this study are listed in Table 2.1. The material density, electron density, and elemental composition data for all the materials are taken from ICRU 46 [37].

Table 2.1
Material Groups for the Monte Carlo Simulations of ^{90}Y Liquid Brachytherapy in a Dog with Osteosarcoma based on ICRU 46

Group	Material	Density	Relative Electron	Relative Electron
		g/cm^3	Density[a]	Density Range
Air	**Air**	1.20×10^{-3}	1.08×10^{-3}	[0, 0.1)
Soft	**Soft Tissue**	1.00	0.991	[0.1, 1)
Tissue	Yellow Marrow	0.98	0.982	
	Red Marrow	1.03	1.02	[1.0, 1.1)
Red	Muscle	1.05	1.04	
Marrow	Blood	1.06	1.05	
	Cartilage	1.10	1.08	
Bone	Trabecular Bone	1.16	1.14	[1.1, +∞)
and Tumor	**Cortical Bone** [b]	1.92	1.78	

[a] The electron density of the material over that of water.

[b] The elemental composition of this group is determined by interpolation of cortical bone and blood.

The materials are divided into four groups according to their relative electron density: *Air*, *Soft Tissue*, *Red Marrow*, and *Bone and Tumor*. Voxels in the CT images can be sorted into these groups according to the relative electron density ρ_e^w calculated from CT numbers. The materials with their names in bold are specified as the reference materials. The elemental composition of voxels in each group is determined by the reference material of the group, except for those in the *Bone and*

Tumor group. *Bone and Tumor* group includes trabecular bone, tumor, and cortical bone, with the density of material varying in a wide range. Trabecular bone is a mixture of dense bone, bone marrow and blood; and the tumor of osteosarcoma, as mentioned before, is usually a combination of bony and soft tissue, with its electron density in the range of the *Bone and Tumor* group. Therefore, the material in the *Bone and Tumor* group is defined as a mixture of cortical bone (high density component) and blood (low density component) to better simulate the nature of the material in this group.

For voxels in *Air*, *Soft Tissue*, and *Red Marrow* groups, the elemental composition is fixed within each group while the mass density changes with a step size corresponding to 0.1 of relative electron density. For each bin, the relative electron density at the mid-point is used. The mass density ρ of the voxel is then calculated using equation 2.5:

$$\rho = \frac{\rho_e^w}{\rho_{e,ref}^w}\rho_{ref} \tag{2.5}$$

where ρ_e^w is the relative electron density of the voxel, $\rho_{e,ref}^w$ is the relative electron density of the reference material in that group and ρ_{ref} is the mass density of the reference material.

For voxels in the *Bone and Tumor* group, the elemental composition and mass density both change with step sizes corresponding to 0.1 of relative electron density. For each bin, the relative electron density at the mid-point is used. If $\rho_e^w \leq \rho_{e,c}^w$, where ρ_e^w is the relative electron density of the voxel and $\rho_{e,c}^w$ is the relative electron density of cortical bone, the fractions of cortical bone and blood for each voxel are calculated by interpolation, as shown in equation 2.6:

$$\{ \begin{array}{l} f_c\rho_{e,c}^w + f_b\rho_{e,b}^w = \rho_e^w \\ f_c + f_b = 1 \end{array} \tag{2.6}$$

where $\rho_{e,b}^w$ is the relative electron density of blood, f_c is the fraction of cortical bone and f_b is the fraction of blood. If $\rho_e^w > \rho_{e,c}^w$, the fraction of cortical bone is assumed to be 1 and the fraction of blood is 0. The elemental composition of the voxel can be expressed as equation 2.7:

$$\omega_i = f_c \omega_{c,i} + f_b \omega_{b,i} \tag{2.7}$$

where ω_i is the elemental weight of element i for that voxel, $\omega_{c,i}$ is the elemental weight of element i for cortical bone and $\omega_{b,i}$ is the elemental weight of element i for blood. Then the mass density is calculated using equation 2.8:

$$\rho = f_c \frac{\rho_e^w}{\rho_{e,c}^w} \rho_c + f_b \frac{\rho_e^w}{\rho_{e,b}^w} \rho_b \tag{2.8}$$

where ρ_c is the mass density of cortical bone and ρ_b is the mass density of blood.

The electron mass stopping power of the materials in the *Soft Tissue* and *Red Marrow* groups were calculated using the web-based program ESTAR [38], as shown in Figure 2.7. The total mass stopping power for all the materials are similar, with standard deviations less than 2%. The mass electron stopping power are barely affected by elemental composition. Therefore, it is proper to use the elemental composition of the reference material for the whole group.

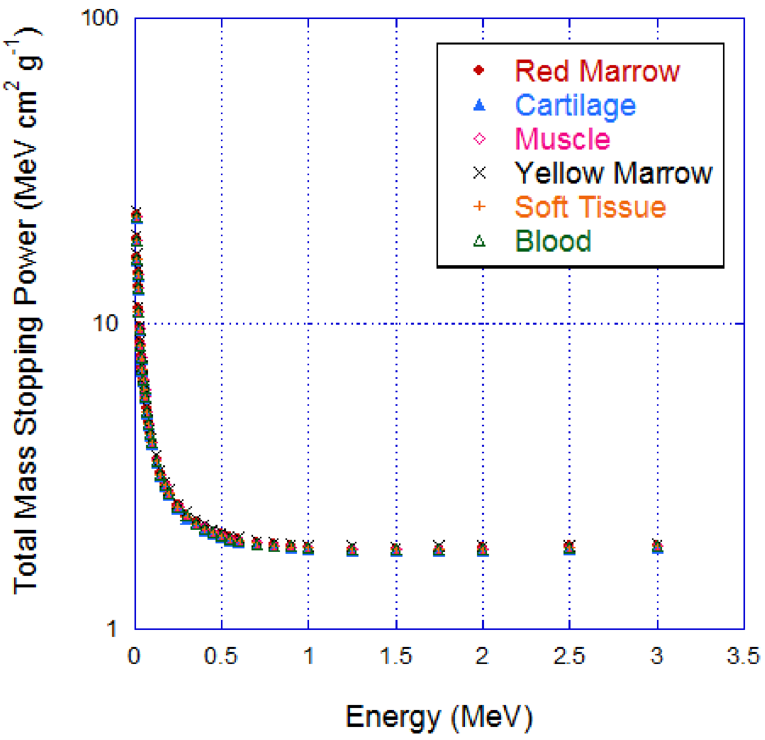


Fig. 2.7. Comparison of total mass stopping power of human tissue for electrons with energy ranging from 0 to 3 MeV.

2.3 PET Image Sampling

The activity distribution of ^{90}Y is sampled using PET images to determine the decay sites for Monte Carlo simulations. PET images in DICOM format are read as a 3-D array. A collapsing 3-D rejection technique is used to improve the efficiency of sampling [39]. To determine the indices (i, j, k) for one decay, the activity distribution array is firstly collapsed over the z-axis to sample k. Secondly, after k is chosen, the two-dimensional array on the corresponding plane k is collapsed over the y-axis to sample j. Finally, i is sampled from the column j at plane k. Then the decay site is randomized in the selected voxel. The coordinates (x, y, z) of the decay site are calculated from the indices (i, j, k) using the following equation:

$$\begin{aligned} \mathrm{x} &= (i + r_{\mathrm{x}})\Delta_{\mathrm{x}} + \mathrm{orig}_{\mathrm{x}}, \\ \mathrm{y} &= (j + r_{\mathrm{y}})\Delta_{\mathrm{y}} + \mathrm{orig}_{\mathrm{y}}, \\ \mathrm{z} &= (k + r_{\mathrm{z}})\Delta_{\mathrm{z}} + \mathrm{orig}_{\mathrm{z}}, \end{aligned} \tag{2.9}$$

where Δ_{x}, Δ_{y}, and Δ_{z} are the voxel sizes along x, y, z axis respectively, r_{x}, r_{y}, and r_{z} are random numbers between -0.5 and 0.5 (assume the position of primary particle is random in the voxel), and $(\mathrm{orig}_{\mathrm{x}}, \mathrm{orig}_{\mathrm{y}}, \mathrm{orig}_{\mathrm{z}})$ is the coordinates of the voxel with indices (0, 0, 0).

2.4 Geant4 Monte Carlo Simulation

Techniques described in 2.2 and 2.3 are implemented in the Monte Carlo code Geant4 for dosimetry calculations. Geant4 is a general purpose toolkit to simulate the passage of particles through matter which has been widely used in nuclear physics, accelerator design, space science and medical physics [40]. Geant4 includes a large variety of functionalities with high degree of flexibility. It also contains complete interaction cross section data files and physics processes over a wide energy range from eV to TeV.

In these simulations, a patient-specific geometry is built using anatomical data obtained from CT images. The material properties of tumor and surrounding tissues are calculated based on a CT number - electron density calibration. The ^{90}Y distribution is sampled from PET images using a collapsing 3-D rejection technique to determine the decay sites. The β^{-} spectrum of ^{90}Y is obtained from JANIS [41]. Then the energy deposition in each voxel is calculated. Dosimetry results are given as specific absorbed dose (S-values) in each voxel, with a unit of Gy/Bq-s. The following equation is used to convert S-value, S to absorbed dose, D:

$$D = S[\mathrm{Gy/Bq-s}] \times \mathrm{A[Bq]} \times \tau[\mathrm{s}], \tag{2.10}$$

where A is the total administrated activity, and τ is the residence time of ^{90}Y radiopharmaceutical. The residence time τ is given by the equation below:

$$\tau = \frac{1}{\lambda_p + \lambda_b} \tag{2.11}$$

where λ_p is the physical radioactive decay constant of ^{90}Y and λ_b is the biological clearance constant of the ^{90}Y radiopharmaceutical. Since the pharmacokinetics of the ^{90}Y radiopharmaceutical hasn't been assessed, τ is assumed to be the inverse of λ_p. τ equals 3.32×10^5 s.

2.5 3-D CT Segmentation

To evaluate the dose delivered to the bone, tumor and bone marrow separately, 3-D CT image segmentations are performed morphologically using a segmentation software ITK-SNAP [42]. A segmentation can be generated either manually by drawing regions of interest on the CT images, or automatically by specifying an CT number window and a smoothness filter for the structure of interest. The segmentation result is saved as a 3-D binary array. The dose volume histogram (DVH) of a structure of interest can be acquired by applying the segmentation array as a mask with the dose distribution array.

3. RESULTS AND DISCUSSION

3.1 CT Number - Electron Density Calibration

The CIRS 062 electron density phantom was scanned using the Siemens Biograph mCT. The average CT number for each tissue equivalent insert and its standard deviation were measured from the central slices of the reconstructed DICOM files and plotted against the relative electron density, as shown in Figure 3.1. The average standard deviation was 31.35 HU for all the materials. CT numbers from patient images were converted to relative electron densities by linear interpolation of data points and further converted into material properties as previous described.

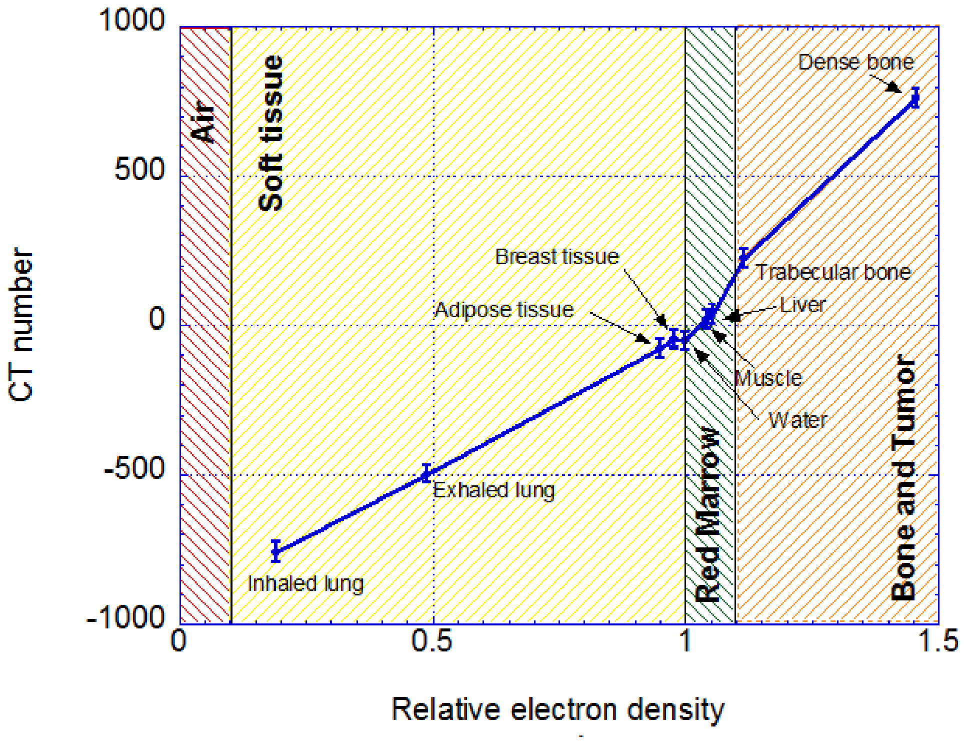


Fig. 3.1. CT numbers with standard deviation against relative electron density for all tissue equivalent inserts. Material groups are showed with corresponding relative electron density range.

3.2 Preclinical Results

A 5 years old male dog with osteosarcoma in left tibia was treated with ^{90}Y liquid brachytherapy. The activity concentration of ^{90}Y radiopharmaceutical solution was 78 μCi/μl (2.88 MBq/μl) and a total activity of 16.536 mCi (611.83 MBq) was administrated. After administration, a 60 min PET/CT scan was performed to assess the distribution of ^{90}Y radiopharmaceutical. The collected PET data were reconstructed using ultraHD (including TOF) with 3 iterations, 21 subsets and a post-smoothing filter of 5 mm. A 3-D co-registered ^{90}Y PET/CT image is shown in Figure 3.2

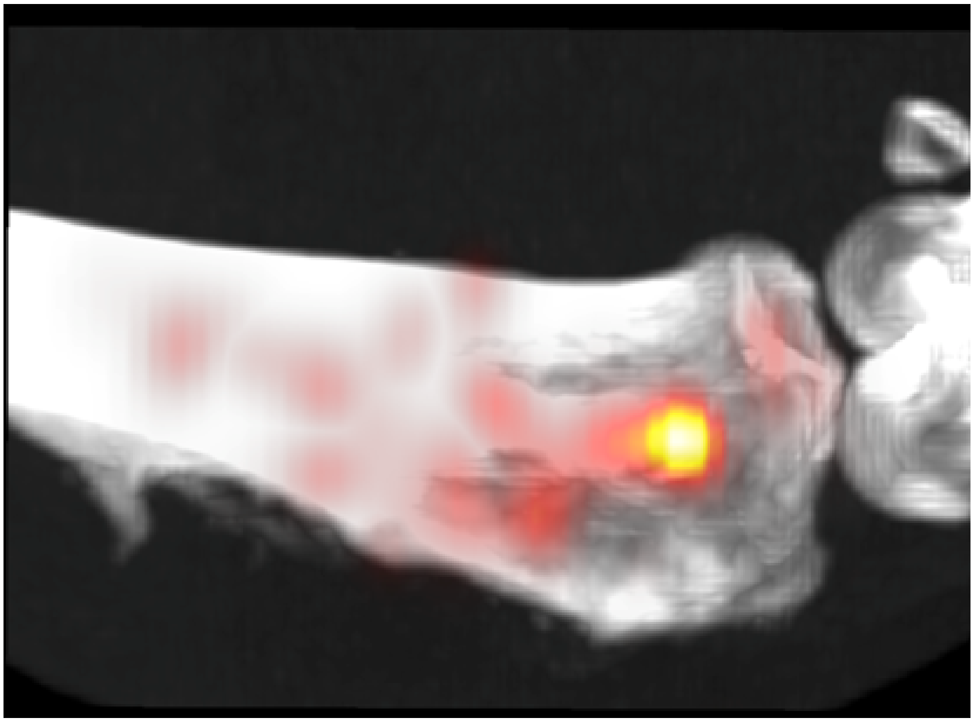


Fig. 3.2. A 3-D co-registered PET/CT image showing the distribution of ^{90}Y in a dog treated with ^{90}Y liquid brachytherapy for osteosarcoma. A high activity hot spot was found in the bone marrow and multiple isolated hotspots in the tumor bed, corresponding to the sites of administration.

3.3 Geant4 Monte Carlo Simulation

3.3.1 Geometry

The reconstructed PET DICOM images had 200 × 200 pixels in 100 slices (203.6 mm × 203.6 mm × 300 mm). The pixel size was 1.018 mm × 1.018 mm and the slice thickness was 3 mm. The reconstructed CT DICOM images were cropped to 209 × 209 pixels in 100 slices (204.0 mm × 204.0 mm × 300 mm) with a pixel size of 0.976 mm × 0.976 mm and a slice thickness of 3 mm. A voxelized cubic phantom was build in Geant4 geometry based on the cropped CT images. The phantom consisted of a box with 209 × 209 × 100 voxels. The voxel size was 0.976 mm × 0.976 mm × 3 mm. The material properties of each voxel were obtained from the CT number as previously described in section 2.2.

A basic description of the Geant4 code developed for ^{90}Y liquid brachytherapy dosimetry is given in Appendix A. Selected lines from class files for DICOM image handling and geometry setup are given in Appendix A.

3.3.2 Physics

In the Geant4 Monte Carlo simulations, the position of each primary particle was determined by sampling the ^{90}Y distribution from PET images, as described in 2.3. The momentum direction of primary particles was assumed to be isotropic. The energy of each primary particle was determined by sampling the β-particle energy spectrum of ^{90}Y obtained from JANIS, as plotted in Figure 3.3. The average energy per decay was 0.934 MeV and the maximum energy was 2.275 MeV. The cut value of Monte Carlo transport for all particles was set as 0.01 mm, i.e. particles with a range smaller than 0.01 mm would not be tracked and were assumed to deposit all the remaining energy locally. The codes for primary particle generation and PET image sampling are shown in Appendix A.

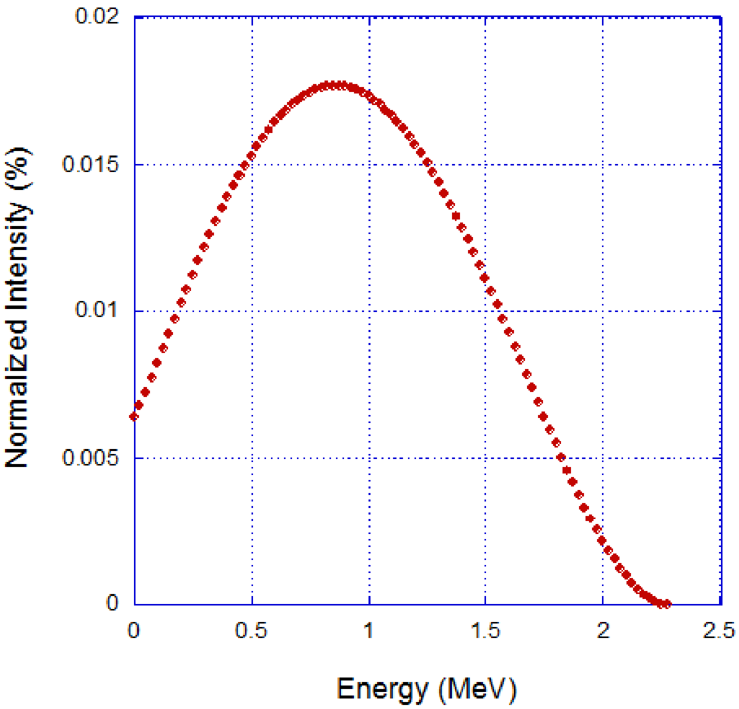


Fig. 3.3. ^{90}Y β-particle emission energy spectrum calculated by JANIS. The average energy was 0.934 MeV and the maximum energy was 2.275 MeV.

3.3.3 Dosimetry Calculations

The Geant4 Monte Carlo simulation was performed in 10 runs to assess the uncertainties of all calculations with a total number of 10^8 particles transported. The S-value in each voxel was recorded in binary files with unit in Gy/Bq-S . The absorbed dose was calculated using equation 2.10.

The output files were handled using MATLAB (MATLAB 7.7.0, The MathWorks Inc., Natick, MA, 2008.) The relevant commands are listed in Appendix B.

3.4 ^{90}Y Liquid Brachytherapy Dosimetry

3.4.1 3-D Dose Distribution and Isodose Curves

A 3-D volume rendering of CT images fused with the dose distribution is shown in Figure 3.4. The resulting dose distribution consisted of multiple hot-spots corresponding to the injection sites.

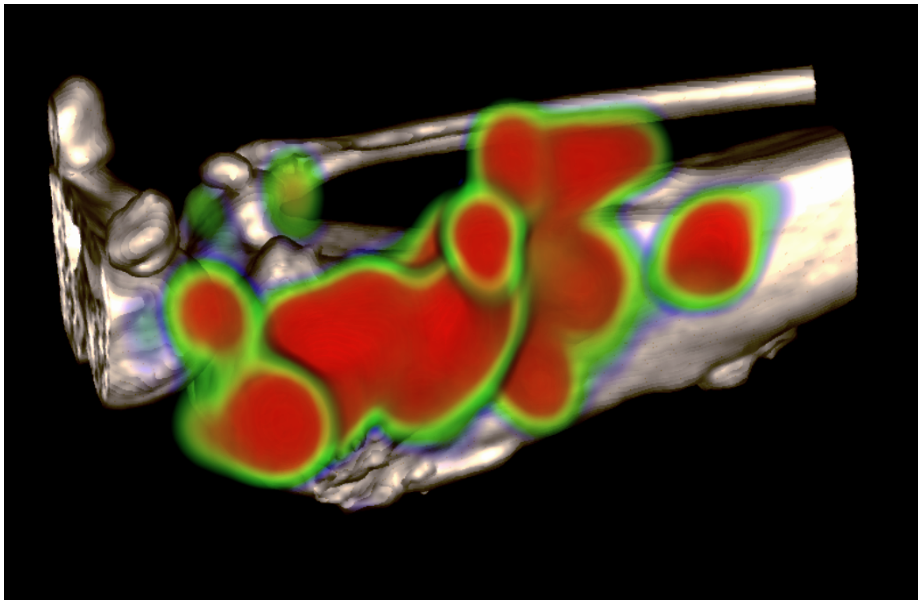


Fig. 3.4. 3-D volume rendering of CT fused with dose distribution of ^{90}Y liquid brachytherapy in a dog with osteosarcoma in tibia.

Isodose curves were plotted on an axial plane, as shown in Figure 3.5. The osteolytic lesions were covered in a total dose over 200 Gy with a central hot-spot of over 1200 Gy. However, the bone marrow was also irradiated with a hotspot of over 1000 Gy.

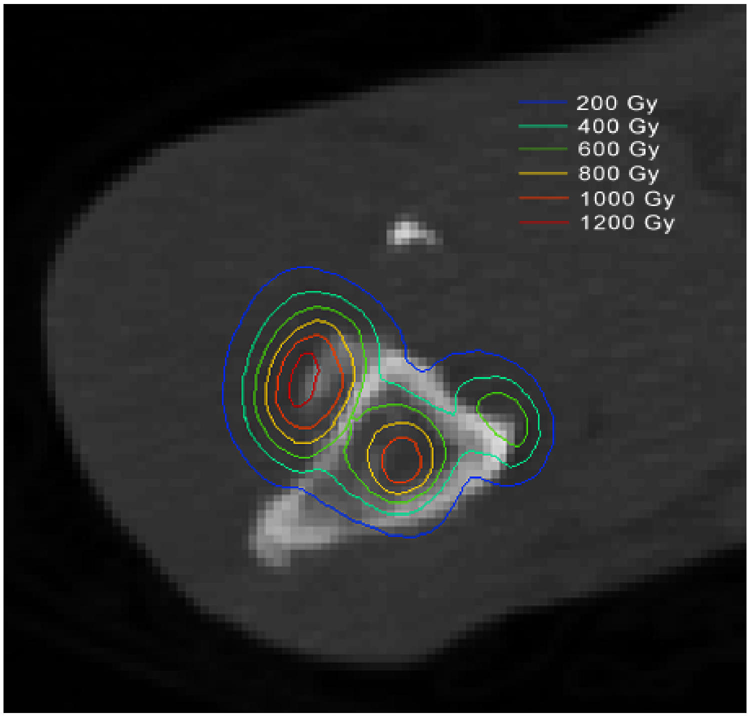


Fig. 3.5. Isodose curves on an axial plane, showing the dose distribution in the tumor bed, adjacent bone and bone marrow.

3.4.2 Dose Volume Histograms

Two 3-D CT segmentations were generated to evaluate the dose delivered to the tumor and surrounding tissues: 1) bone and tumor, and 2) bone marrow (including trabecular bone). Because the tumor of osteosarcoma was mainly bony tissue with intensity close to that of bone, it was difficult to separate tumor from adjacent bone. The dose delivered to the tumor and bone was evaluated as an entity with segmentation 1). Segmentation 1) was generated with a lower CT number threshold of 400 HU and a smoothness filter of 2 mm. segmentation 2) was generated by manually selecting the low density volume in the bone, including the red marrow, trabecular bone, and yellow marrow in the medullary cavity. The two segmentations are shown in Figure 3.6.

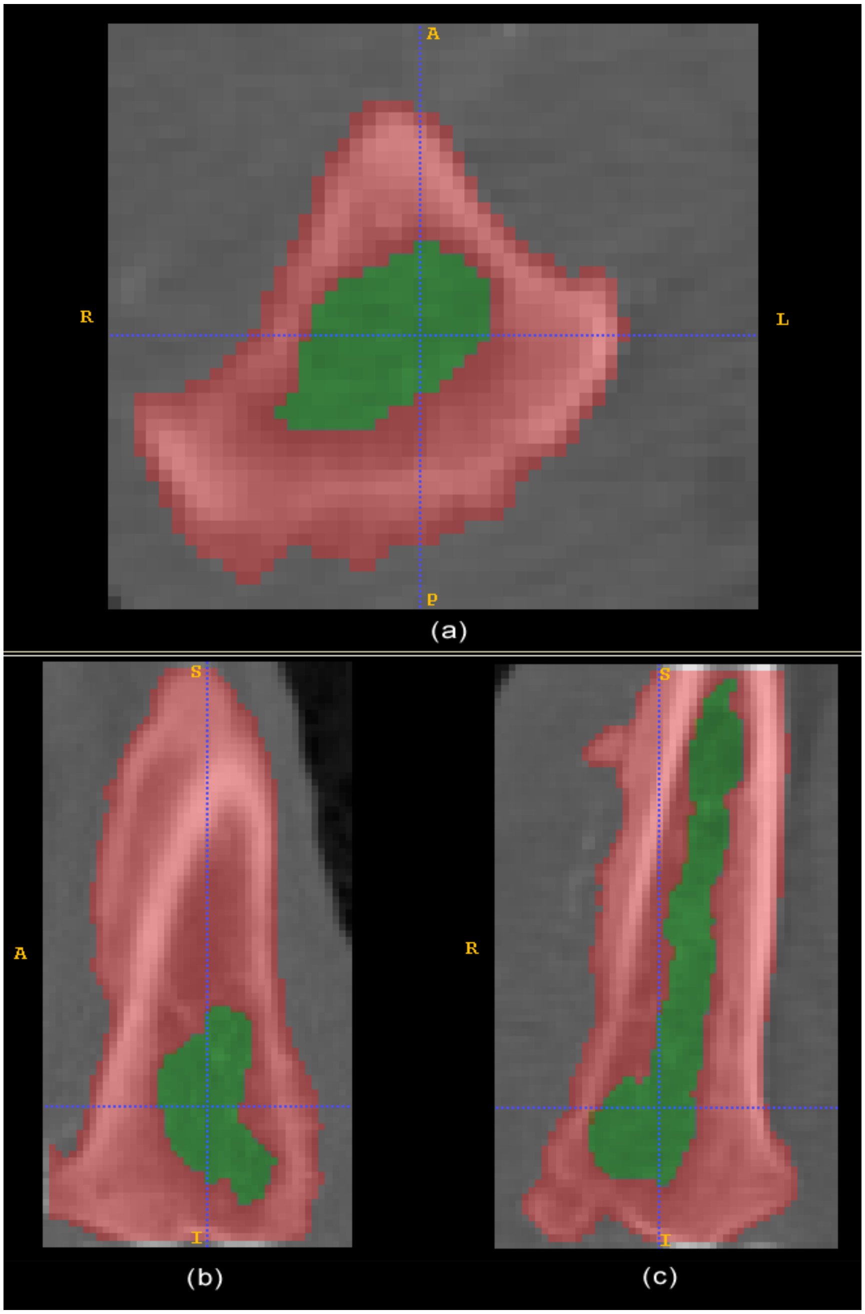


Fig. 3.6. 3-D CT segmentations in the (a) axial, (b) sagittal and (c) coronal planes. Segmentation 1) bone + tumor is marked in red and 2) bone marrow (including trabecular bone) is marked in green.

The CT number histograms for these two segmentations are shown in Figure 3.7. Corresponding material groups are marked in the figure. Segmentation 1) as bone and tumor is in the range of *Muscle* and *Bone and Tumor* group. Segmentation 2) as bone marrow (including trabecular bone) is in the range of *Soft Tissue* and *Muscle* group.

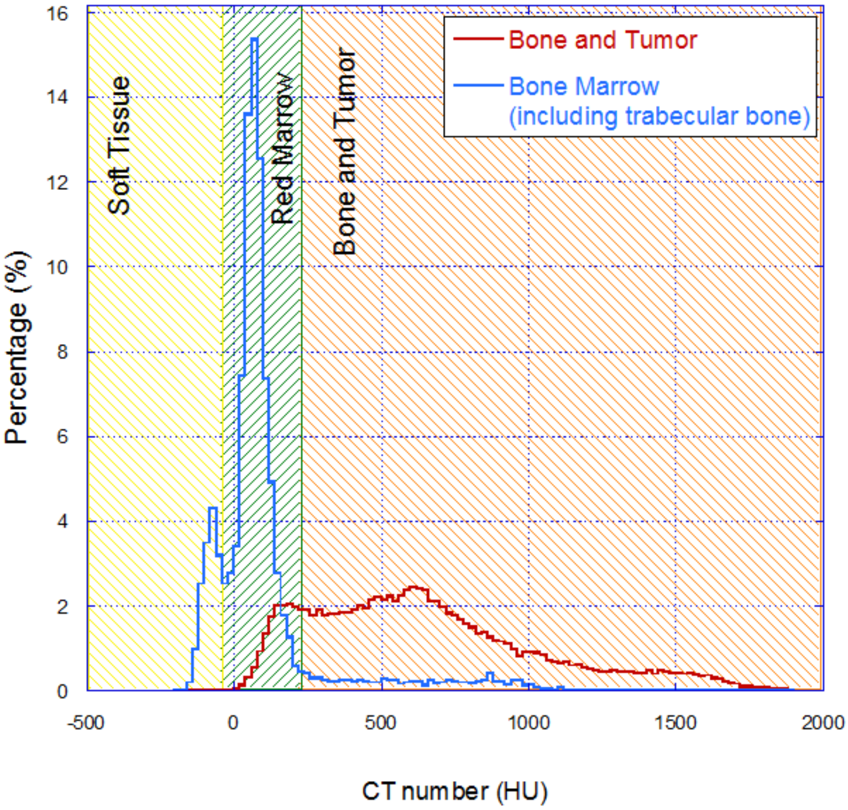
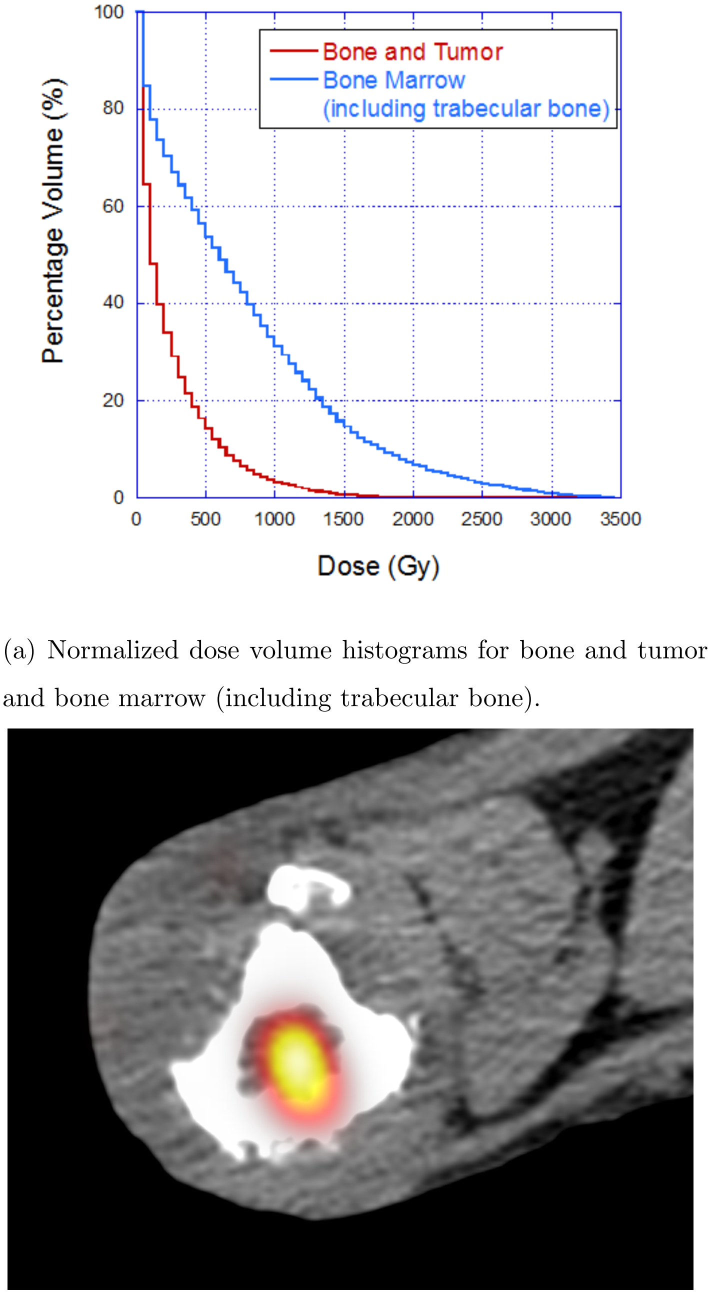


Fig. 3.7. CT number histograms for segmentation 1) as bone and tumor, and 2) as bone marrow (including trabecular bone).



(a) Normalized dose volume histograms for bone and tumor and bone marrow (including trabecular bone).

(b) Co-registered ^{90}Y PET/CT image showing a hot spot in the medullary cavity, resulting high dose to the bone marrow and trabecular bone.

Fig. 3.8. Normalized dose volume histograms for bone and tumor and bone marrow (including trabecular bone), and corresponding ^{90}Y activity distribution in a co-registered PET/CT image.

Normalized dose volume histograms were calculated by applying the segmentation masks to the 3-D dose distribution and shown in Figure 3.8(a). Relevant MATLAB commands are given in Appendix B. The dose distribution in the tumor and other tissues was extremely heterogenous. From the DVH data, about 33.9% of bone and tumor and 70.2% of bone marrow and trabecular bone received a total dose over 200 Gy; about 3.2% of bone and tumor and 31.0% of bone marrow and trabecular bone received a total dose of over 1000 Gy. The ^{90}Y hot spot in the center of the bone as shown in Figure 3.8(b) explained the high dose to the bone marrow and trabecular bone. The uncertainties in absorbed dose to bone and tumor in each voxel were also calculated. The average percentage standard deviation among all voxel was 13.4%, with a range from 0% to 250%.

3.5 Discussions

3.5.1 Safety Concerns

During the injection of ^{90}Y radiopharmaceutical, a small fraction of activity entered the blood stream directly, causing hematological toxicity. Because ^{90}Y is a bone seeker, this activity loss resulted in a dose to the bone surfaces. This activity loss should evaluated for safety concerns, which requires quantitative assessment of ^{90}Y activity concentration from PET images. PET scanners should be calibrated with ^{90}Y in future to measure the ^{90}Y activity concentration directly.

3.5.2 Limitations in the Current Dosimetry Calculations

There are two major limitations in the current dosimetry calculations. Firstly, the activity loss during the injection should be excluded from the total administrated activity. The improvement depends on the quantitative assessment of ^{90}Y activity concentration from PET images. Secondly, the pharmacokinetics of the radiopharmaceutical should be investigated in animals and the residence time τ should be calculated with both physical radioactive decay and biological decay, as in equation 2.11. Then the absorbed dose should be re-calculated. Notice that the clearance rates of ^{90}Y radiopharmaceutical in the bone and tumor and in the soft tissue could be different. Current dosimetry calculations were performed based on ^{90}Y distribution from PET/CT scan immediately after the injection. In the future, the changes in ^{90}Y activity distribution with time should be considered for accurate dosimetry.

3.5.3 Dose Distribution of ^{90}Y Liquid Brachytherapy

The heterogenous dose distribution of ^{90}Y liquid brachytherapy mainly resulted from the injection pattern: for each adaptor, the ^{90}Y radiopharmaceutical was injected at 3 or 4 spots spaced about 1 cm from each other. An alternative was to administrated the radiopharmaceutical as a continuous line along the drilling hole for a more uniform dose distribution. However, since the injection was performed manually, it was extremely difficult to achieve.

Problems related with heterogenous dose distribution include inadequate dose in tumor and high dose in normal tissues. In the ^{90}Y liquid brachytherapy for osteosarcoma, the tumor usually merges with the adjacent cortex with a indistinct and irregular border, it is difficult to separate the tumor from adjacent bone and estimate the dose delivered to the normal bone.

In the meanwhile, it is also hard to evaluate the dose received by the red marrow in the treated bone. Low density volume in the bone represents a mixture of red marrow, trabecular bone, and yellow marrow in the medullary cavity. It is difficult to separate these tissue from each other. The dose to the red marrow should be carefully estimated according to the bone type and the tumor location in the bone.

In this study, all segmentations were performed using CT images morphologically. MRI with supreme tissue differentiation ability should be considered as an alternative for the definition of target treatment volume and segmentations.

3.5.4 Efficacy and Potential Utility

The evaluation of the efficacy of ^{90}Y liquid brachytherapy cannot be established without acquiring large amount of preclinical data, including longitudinal data for individual dogs. Similar to ^{153}Sm-EDTMP targeted radionuclide therapy, it is not likely to use ^{90}Y liquid brachytherapy as an primary treatment method for osteosarcoma. The potential utility of ^{90}Y liquid brachytherapy include being used as an addition to chemotherapy or adjuvant therapy after surgery, being used for the treatment of unresectable tumors or metastases, and for the pain palliation purposes.

4. CONCLUSION

A novel ^{90}Y liquid brachytherapy radiopharmaceutical is currently being studied for the treatment of osteosarcoma using a preclinical translational model in dogs to assess its potential efficacy and toxicity. In this study, dosimetry calculations were carried out for the ^{90}Y liquid brachytherapy in a dog with osteosarcoma using Geant4. A patient-specific geometry was build using anatomy data obtained from CT images. The material properties of tumor and surrounding tissues were obtained from CT numbers based on an electron density phantom calibration. The *in vivo* distribution of ^{90}Y radiopharmaceutical was assessed using TOF PET scan, and then sampled in Geant4 for Monte Carlo simulations.

The 3-D dose distribution of ^{90}Y liquid brachytherapy showed significant heterogeneity with multiple hot spots corresponding to the injection sites. Dose volume histograms were calculated for 1) bone and tumor, and 2) bone marrow (including trabecular bone). From the DVH data, about 33.9% of bone and tumor and 70.2% of bone marrow and trabecular bone received a total dose over 200 Gy; about 3.2% of bone and tumor and 31.0% of bone marrow and trabecular bone received a total dose of over 1000 Gy. The average percentage standard deviation of the absorbed dose to bone and tumor was 13.4%, with a range from 0% to 250%.

^{90}Y liquid brachytherapy has the potential to be used as an adjuvant therapy or for palliation purposes. As for future work, dosimetry calculations should be performed for more cases, and comparison should be made for osteosarcoma at different sites, with varied tumor volume, treated with different activity concentration and total activity. PET scanners should be calibrated to establish the direct quantitative assessment of ^{90}Y activity concentration. The pharmacokinetics of the ^{90}Y radiopharmaceutical should also be studied. These studies allow the assessment of hematological toxicity of ^{90}Y liquid brachytherapy and accurate dosimetry calculations. Furthermore, the efficacy of the ^{90}Y liquid brachytherapy alone or combined with other modalities in dogs with osteosarcoma should be evaluated with adequate preclinical data.

REFERENCES

[1] Dorfman HD, Czerniak B. Bone tumors. St. Louis, MO: Mosby 1998.

[2] Jaffe N, Bruland OS, Bielack S. Pediatric and adolescent osteosarcoma. Vol. 152 of Cancer Treatment and Research Boston, MA: Springer US 2010.

[3] Altekruse SF, Kosary CL, Krapcho M, Neyman N, Aminou R, Waldron W, Ruhl J, Howlader N, Tatalovich Z, Cho H, Mariotto A, Eisner MP, Lewis DR, Cronin K, Chen HS, Feuer EJ, Stinchcomb DG, Edwards BK (eds.). SEER cancer statistics review, 1975-2007. Bethesda, MD: National Cancer Institute 2010. Accessed on April 2011: http://seer.cancer.gov/csr/1975-2007/index.html

[4] Mirabello L, Troisi RJ, Savage SA. Osteosarcoma incidence and survival rates from 1973 to 2004: Data from the Surveillance, Epidemiology, and End Results program. Cancer 2009; 115(7): 1531–18.

[5] Gurney JG, Bulterys M. Malignant bone and soft tissue tumors in childhood: SEER pediatric monograph. Bethesda, MD: National Cancer Institute. 2006.

[6] Dresel S. PET in oncology. New York: SpringerLink 2007.

[7] Pappo A. Pediatric bone and soft tissue sarcomas. New York: Springer 2006.

[8] Hawkins DS, Arndt CAS. Pattern of disease recurrence and prognostic factors in patients with osteosarcoma treated with contemporary chemotherapy. Cancer 2003; 98(11): 2447–56.

[9] Ta HT, Dass CR, Choong PFM, Dunstan DE. Osteosarcoma treatment: State of the art. Cancer Metast Rev 2009; 28(1-2): 247–63.

[10] Bacci G, Longhi A, Fagioli F, Briccoli A, Versari M, Picci P. Adjuvant and neoadjuvant chemotherapy for osteosarcoma of the extremities: 27 year experience at Rizzoli Institute, Italy. Eur J Cancer 2005; 41(18): 2836–45.

[11] Chou AJ, Geller DS, Gorlick R. Therapy for osteosarcoma: Where do we go from here? Paediatr Drugs 2008; 10(5): 315–27.

[12] Wittig J, et al. . Osteosarcoma: A multidisciplinary approach to diagnosis and treatment. Am Fam Physician 2002; 65(6): 1123–32.

[13] Schwarz R, Bruland OS, Cassoni A, Schomberg P, Bielack S. The role of radiotherapy in oseosarcoma. Cancer Treat Res 2010; 152 147–64.

[14] NCRP . Management of radionuclide therapy patients: (NCRP Report 155). Bethesda, MD: National Council on Radiation Protection and Measurements. 2007.

[15] Volkert WA, Hoffman TJ. Therapeutic radiopharmaceuticals. Chem Rev 1999; 99(9): 2269–92.

[16] Paes FM, Serafini AN. Systemic metabolic radiopharmaceutical therapy in the treatment of metastatic bone pain. Semin Nucl Med 2010; 40(2): 89–104.

[17] Bruland OS, Skretting A, Solheim OP, Aas M. Targeted radiotherapy of osteosarcoma using Sm-153-EDTMP: A new promising approach. Acta Oncol 1996; 35(3): 381–84.

[18] Geller DS, Gorlick R. HER-2 targeted treatment of osteosarcoma: The challenges of developing targeted therapy and prognostic factors for rare malignancies. Expert Opin Pharmacother 2010; 11(1): 51–61.

[19] Moriceau G, Ory B, Verrecchia F, Gouin F, Blanchard F, Redini F, Heymann D. Therapeutic approach of primary bone tumours by bisphosphonates. Curr Pharm Des 2010; 16(27): 2981–87.

[20] Bruland OS, Jonasdottir TJ, Fisher DR, Larsen RH. Radium-223: From radiochemical development to clinical applications in targeted cancer therapy. Curr Radiopharm 2010; 1(3): 203–8.

[21] Herzog H, Rosch F, Stocklin G, Lueders C, Qaim SM, Feinendegen LE. Measurement of pharmacokinetics of yttrium-86 radiopharmaceuticals with PET and radiation dose calculation radiotherapeutics. J Nucl Med 1993; 34(12): 2222–6.

[22] Djokić DD, Janković DL, Nikolić NS. Labeling, characterization, and in vivo localization of a new 90Y-based phosphonate chelate 2,3-dicarboxypropane-1,1-diphosphonic acid for the treatment of bone metastases: Comparison with 99mTc-DPD complex. Bioorg Med Chem 2008; 16(8): 4457–65.

[23] Ogawa K, Kawashima H, Shiba K, Washiyama K, Yoshimoto M, Kiyono Y, Ueda M, Mori H, Saji H. Development of [(90)Y]DOTA-conjugated bisphosphonate for treatment of painful bone metastases. Nucl Med Biol 2009; 36(2): 129–35.

[24] Grant RL, Crowder ML, Ibbott GS, Simon J, Frank RK, Rogers J. Three-dimensional dosimetry of a beta-emitting raidonuclide using PRESAGE dosimeters. J Phys Conf Ser 2010; 250(1): 1–7.

[25] Nickles RJ, Roberts AD, Nye JA, Converse AK, Barnhart TE, Avila-Rodriguez MA, Sundaresan R, Dick DW, Hammas RJ, Thomadsen BR Assaying and pet imaging of ytrrium-90. In: IEEE Nuclear Science Symposium Conference Record. Vol. 6, pp. 3412–4.

[26] Selwyn RG, Nickles RJ, Thomadsen BR, DeWerd L, Micka J. A new internal pair production branching ratio of 90Y: The development of a non-destructive assay for 90Y and 90Sr. Appl Radiat Isot 2007; 65(3): 318–27.

[27] Lhommel R, Goffette P, Van den Eynde M, Jamar F, Pauwels S, Bilbao JI, Walrand S. Yttrium-90 TOF PET scan demonstrates high-resolution biodistribution after liver SIRT. Eur J Nucl Med Mol I 2009; 36(10): 1696.

[28] Withrow SJ, Powers BE, Straw RC, Wilkins RM. Comparative aspects of osteosarcoma: Dog versus man. Clin Orthop Relat Res 1991; (270): 159–68.

[29] Withrow SJ, Wilkins RM. Cross talk from pets to people: Translational osteosarcoma treatments. ILAR J 2010; 51(3): 208–13.

[30] Paoloni M, Khanna C. Translation of new cancer treatments from pet dogs to humans. Nat Rev Cancer 2008; 8(2): 147–56.

[31] Hawkes DJ, Jackson DF. An accurate parametrisation of the x-ray attenuation coefficient. Phys Med Biol 1980; 25(6): 1167–71.

[32] Parker RP, Hobday PA, Cassell KJ. The direct use of CT numbers in radiotherapy dosage calculations for inhomogeneous media Phys Med Biol 1979; 24(4): 802–9.

[33] McCullough EC, Holmes TW. Acceptance testing computerized radiation therapy treatment planning systems: Direct utilization of CT scan data. Med Phys 1985; 12(2): 237–42.

[34] DeMarco JJ, Solberg TD, Smathers JB. CT-based Monte Carlo simulation tool for dosimetry planning and analysis. Med Phys 1998; 25(1): 1–11.

[35] du Plessis FCP, Willemse CA, Lotter MG, Goedhals L. The indirect use of CT numbers to establish material properties needed for Monte Carlo calculation of dose distributions in patients. Med Phys 1998; 25(7): 1195–201.

[36] Thomas SJ. Relative electron density calibration of CT scanners for radiotherapy treatment planning. Br J Radiol 1999; 72(860): 781–6.

[37] ICRU . Photon, electron, proton and neutron interaction data for body tissues (ICRU Report 46). Bethesda, MD: International Commission on Radiation Units and Measurements. 1992.

[38] Berger MJ, Coursey JS, Zucker MA, Chang J. Stopping-power and range tables for electrons, protons, and helium ions. Gaithersburg, MD: The National Institute of Standards and Technology (NIST) 1998.

[39] Akabani G, Hawkins WG, Eckblade MB, Leichner PK. Patient-specific dosimetry using quantitative SPECT imaging and three-dimensional discrete Fourier transform convolution. J Nucl Med 1997; 38(2): 308–14.

[40] Agostinelli S, Allison J, Amako K, Apostolakis J, Araujo H. Geant4-a simulation toolkit. Nucl Instrum Meth A 2003; 506(3): 250–303.

[41] Java-based Nuclear Information Server (JANIS). France: OECD Nuclear Energy Agency and Aquitaine Electronique Informatique. Accessed on April 2011: http://www.nea.fr/janis

[42] Yushkevich PA, Piven J, Cody Hazlett H, Gimpel Smith R, Ho S, Gee JC, Gerig G. User-guided 3D active contour segmentation of anatomical structures: significantly improved efficiency and reliability. Neuroimage 2006; 31(3): 1116–28.

APPENDIX A

GEANT4 PET/CT APPLICATION

A.1. Introduction

The Geant4 application used for the ^{90}Y liquid brachytherapy dosimetry (named PETCT) was developed from the DICOM application, which is one of the extended example in Geant4.9.3.p01 package, under the directory \examples\extended\medical \DICOM. The original DICOM example serves first to convert a CT DICOM file to a simple ASCII file (.g4dcm file), where the CT numbers are converted to material composition and densities to create a Geant4 geometry.

The PET/CT application performs the following major functios:

- PET DICOM file handling (implemented in `PETHandler.cc`): read a PET DICOM image sequence is read as a 3-D array and store the array in memory; collapse the array along z-axis direction to a 1-D array, collapse each slice in the 3-D array along y-axis direction, and the collapsed results are stored in memory; read the image position and sizes from the headers.

- CT DICOM file handling (implemented in `CTHandler.cc`): crop CT DICOM images to match the PET image size; convert CT DICOM files to .g4dcm files, which contain the material ID and relative electron density of converted from the CT number of each voxel.

- Geometry construction (implemented in `PETCTDetectorConstruction.cc`): build a voxelized cubic phantom was build in Geant4 geometry using information in .g4dcm files.

- Sampling PET images (implemented in `PETCTPrimaryGeneratorAction.cc`): sample the collapsed PET image arrays to determine the position of each primary particle.

- Dosimetry calculation (implemented in PETCTRunAction.cc): score the dose deposition in each voxel; write the scoring results of each run to a binary file, where the unit of dose is Gy/Bq-s.

Supporting files of PET/CT application include:

- CT.dat, containing the total number and filenames of CT DICOM files.
- PET.dat, containing the total number and filenames of PET DICOM files.
- CT2REDensity.dat, containing the calibration data points to convert CT number to relative electron density.
- run.mac, containing the definition of primary particle energy spectrum.

Selected lines from PET/CT application class files are shown in A and A.

A.2. DICOM Image Handling and Geometry Setup

From PETHandler.cc:

```
...
void PETHandler::CheckFileFormat()
{
    //Check existence of PET.dat
        //if not exists, exit.
        std::ifstream checkData("PET.dat");
    char * oneLine = new char[FILENAMESIZE];
    if(!(checkData.is\_open())) {
        G4cout << "PET.dat␣does␣not␣exist!";
        exit(0);
    }

    //Read dicom file names from PET.dat
    // read the number of PET dicom files
    checkData >> nFiles;
    checkData.getline(oneLine,100);
        char * name= new char[FILENAMESIZE];
        FILE * dicom;
        for(int rep = 0; rep < nFiles; rep++)  //begin loop on
            filenames
        {
      checkData.getline(oneLine,100);
          std::sprintf(name,"%s.dcm",oneLine);  //get the
             filename of a dicom
          std::printf("␣Opening␣%s␣and␣reading␣:\n",name);
          dicom = std::fopen(name,"rb");   //open a dicom
          if( dicom != 0 )
          { ReadFile(dicom,rep); }   //read the dicom
          else
          { G4cout << "\nError␣opening␣file␣:␣" << name <<
             G4endl;}
          std::fclose(dicom);   //close the dicom
         }
        checkData.close();    //close "PET.dat"

        //Normalized PETsum
        for(int aa=0;aa<nFiles;aa++)
        {PETsum[aa]=PETsum[aa]/maxPETsum;}

        delete [] name;
        delete [] oneLine;
}

//Read PET dicom files and store the image in arrays
```

```
void PETHandler::ReadFile(FILE *dicom,int rep)
{
    char * buffer = new char[LINEBUFFSIZE];
    implicitEndian = false;
    littleEndian = true;

    std::fread( buffer, 1, 128, dicom ); // The first 128
       bytes are not important

        // Reads the "DICOM" letters
    std::fread( buffer, 1, 4, dicom );
    if(std::strncmp("DICM", buffer, 4) != 0) { //if there is
       no "DICM" letters
        std::fseek(dicom, 0, SEEK_SET);        //go back to
           the beginning of the file
        implicitEndian = true;
    }

    short readGroupId;    // identify the kind of input data
    short readElementId;  // identify a particular type
       information
    short elementLength2; // deal with element length in 2
       bytes
    G4int elementLength4; // deal with element length in 4
       bytes

    char * data = new char[DATABUFFSIZE];

    // Read the header
        //Get information with tags
        //If find the beginning of pixel data, break the loop
    while(true) {

        //Reading groups and elements :
        readGroupId = 0;
        readElementId = 0;
        // group ID
        std::fread(buffer, 2, 1, dicom);
        GetValue(buffer, readGroupId);
        // element ID
        std::fread(buffer, 2, 1, dicom);
        GetValue(buffer, readElementId);

        // Creating a tag to be identified afterward
        G4int tagDictionary = readGroupId*0x10000 +
           readElementId;

      // VR or element length
        std::fread(buffer,2,1,dicom);
        GetValue(buffer, elementLength2);

        // If value representation (VR) is OB, OW, SQ, UN, OF
           or UT, the next length is 32 bits
        if((elementLength2 == 0x424f ||  // "OB"
            elementLength2 == 0x574f ||  // "OW"
            elementLength2 == 0x464f ||  // "OF"
            elementLength2 == 0x5455 ||  // "UT"
            elementLength2 == 0x5153 || //  "SQ"
            elementLength2 == 0x4e55) && // "UN"
           !implicitEndian ) {           // explicit VR
```

```
        std::fread(buffer, 2, 1, dicom); // Skip 2
           reserved bytes

        // element length
        std::fread(buffer, 4, 1, dicom);
        GetValue(buffer, elementLength4);

        // If find the beginning of the pixels, then break
        if(tagDictionary == 0x7FE00010) break;

        if(elementLength4 == 0xFFFFFFFF) G4cerr << "Too␣
           difficult␣for␣me!" << G4endl; // VR = SQ with
           undefined length

        // Reading data which follows the tag
        std::fread(data, elementLength4,1,dicom);

    } else {

            if(!implicitEndian || readGroupId == 2) {  //
                explicit with VR different than previous
               ones
              // element length (2 bytes)
              std::fread(buffer, 2, 1, dicom);
              GetValue(buffer, elementLength2);
              elementLength4 = elementLength2;

              if(tagDictionary == 0x7FE00010) break; //
                 beginning of the pixels
              std::fread(data, elementLength4, 1, dicom);

            } else {                                 //
               Implicit VR

              // element length (4 bytes)
              if(std::fseek(dicom, -2, SEEK_CUR) != 0) {
                  G4cerr << "[PETHandler]␣fseek␣failed" <<
                      G4endl;
                  exit(-10);}

              std::fread(buffer, 4, 1, dicom);
              GetValue(buffer, elementLength4);

              if(elementLength4 == 0xFFFFFFFF) {// VR = SQ
                  with undefined length
          short momreadGroupId;
              short momreadElementId;
              unsigned int momelementLength4 ; // deal
                 with element length in 4 bytes
              while(true){
                    std::fread(buffer, 2, 1, dicom);
                    GetValue(buffer, momreadGroupId);
                    std::fread(buffer, 2, 1, dicom);
                    GetValue(buffer, momreadElementId);
                    std::fread(buffer, 4, 1, dicom);
                    GetValue(buffer, momelementLength4);
```

```
                    if(momelementLength4 == 0x00000000 ||
                       momelementLength4 == 0xFFFFFFFF )
                       break;
        std::fread(buffer, momelementLength4, 1, dicom);
                    }
              } else {
              if(tagDictionary == 0x7FE00010) break; //
                 beginning of the pixels
              std::fread(data, elementLength4, 1, dicom);
             }
       }
    }

    // Add a NULL termination
    data[elementLength4] = '\0';

    // analyzing data of each tag
    GetInformation(tagDictionary, data);

}//end of the while loop

//--- Store Image Position :coordinates of the upper left
   hand corner (first pixel) of the image
    origX.push_back(ImagePositionX);
    origY.push_back(ImagePositionY);
    origZ.push_back(ImagePositionZ);

G4double sum=0; //sum of all pixels
    G4double maxY=0; //maximum pixel value in a row
```

```
 if(bitAllocated == 8) {
     // Case of 8 bits :
    char ch;
    for(G4int j = 0; j < rows; j++) {
        for(G4int i = 0; i < columns; i++) {
    sumY[rep][j]=0;
            std::fread(&ch, 1, 1, dicom);
            image[rep][j][i] = ch;//store pixel values in
               image
            sumY[rep][j] += ch;//sum up in one row
            sum +=image[rep][j][i] ;//sum up the whole
               slice
        }
    if(sumY[rep][j]>=maxY){maxY=sumY[rep][j];} //Get the
       maximum of sumY
    }
} else {
    //  case from 12 to 16 bits :

    // Set the maxPixelValue to 0
    maxPixelValue=0;

    char sbuff[2];
    short pixel;
    for( G4int j = 0; j < rows; j++) {
        sumY[rep][j]=0;
        for( G4int i = 0; i < columns; i++) {
            std::fread(sbuff, 2, 1, dicom);
            GetValue(sbuff, pixel);
```

```
                   //Get the maxPixelValue in this slice
           if(pixel>maxPixelValue)
           {maxPixelValue=pixel;}

           image[rep][j][i] = pixel;//store pixel values
              in image
           sumY[rep][j]+=pixel;////sum up in one row
           sum +=image[rep][j][i] ;//sum up the whole
              slice
       }
           if(sumY[rep][j]>=maxY){maxY=sumY[rep][j];} //
              Get the maximum of sumY
       }
}

   //Normalize sumY
   for(G4int yy=0;yy< rows;yy++)
   {
   sumY[rep][yy]=sumY[rep][yy]/maxY;
   }

   //Normalize the slice
   for(int j=0;j<rows;j++)
   {
     for(int i=0;i<columns;i++)
     {image[rep][j][i]=image[rep][j][i]/G4double(
        maxPixelValue);}
   }

   //Store the sum of this slice in the vector PETsum
```

```
        PETsum.push_back(sum);
        if(sum>=maxPETsum)//Get the maximun of the sums of
           slices
        {maxPETsum=sum;}

    delete [] buffer;
    delete [] data;
}

void PETHandler::GetInformation(G4int & tagDictionary, char *
   data) {
    if(tagDictionary == 0x00280010 ) { // Number of Rows
        GetValue(data, rows);
        std::printf("[0x00280010]␣Rows␣->␣%i\n",rows);

    } else if(tagDictionary == 0x00280011 ) { // Number of
       columns
        GetValue(data, columns);
        std::printf("[0x00280011]␣Columns␣->␣%i\n",columns);

    } else if(tagDictionary == 0x00280100 ) { // Bits
       allocated: This is how much computer memory will be
       used to store a pixel sample.
        GetValue(data, bitAllocated);
        std::printf("[0x00280100]␣Bits␣allocated␣->␣%i\n",
           bitAllocated);

    } else if(tagDictionary == 0x00280107 ) { //  Max. pixel
       value
        GetValue(data, maxPixelValue);
        std::printf("[0x00280107]␣Max.␣pixel␣value␣->␣%i\n",
           maxPixelValue);

    } else if(tagDictionary == 0x00080006 ) { //  Modality
        std::printf("[0x00080006]␣Modality␣->␣%s\n", data);

    } else if(tagDictionary == 0x00280030 ) { // Pixel Spacing
      G4String datas(data);
      int iss = datas.find('\\');
      pixelSpacingX = atof( datas.substr(0,iss).c_str() );
          pixelSpacingY = atof(datas.substr(iss+1,datas.length
             ()).c_str());

    } else if(tagDictionary == 0x00200032 ) { // Image
       Position
        G4String datas(data);
        int iss = datas.find('\\');
        ImagePositionX=atof( datas.substr(0,iss).c_str() );
        G4String datas2=datas.substr(iss+1,datas.length()).
           c_str();
          iss = datas2.find('\\');
      ImagePositionY = atof( datas2.substr(0,iss).c_str() );
        ImagePositionZ=atof(datas2.substr(iss+1,datas.length()
           ).c_str());
        std::printf("[0x00200032]␣Image␣Position␣(Patient,mm)␣
           ->␣%s\n", data);

    } else if(tagDictionary == 0x00180050 ) { // Slice
       Thickness
        sliceThickness = atof(data);
```

```
        std::printf("[0x00180050]␣Slice␣Thickness␣(mm)␣->␣%f\n
           ", sliceThickness);

    } else if(tagDictionary == 0x00020010) { // Endian
        if(strcmp(data, "1.2.840.10008.1.2") == 0)
            implicitEndian = true;
        else if(strncmp(data, "1.2.840.10008.1.2.2", 19) == 0)
            littleEndian = false;
    }
}
...
```

From CTHandler.cc:

```
...
void CTHandler::CheckFileFormat()
{

 petImagePositionX=PETHandler::origX[0];
 petImagePositionY=PETHandler::origY[0];
 petrows=PETHandler::rows;
 petcolumns=PETHandler::columns;
 petpixelSpacingX=PETHandler::pixelSpacingX;
 petpixelSpacingY=PETHandler::pixelSpacingY;

//Check existance of CT.dat
std::ifstream checkData("CT.dat");
char * oneLine = new char[128];
    if(!(checkData.is_open())) {
        G4cout << "CT.dat␣does␣not␣exist!"<<G4endl;
        exit(0);
    }

//Read the number of CT dicom files
    checkData >> nFiles;
    G4String oneName;
    checkData.getline(oneLine,100);

 std::ifstream testExistence;
    G4bool existAlready = true;

for(G4int rep = 0; rep < nFiles; rep++) {
      checkData.getline(oneLine,100);
      oneName = oneLine;
      oneName += ".g4dcm"; // .g4dcm file name
    //  G4cout <<  "Check the existence of  " << oneName <<
       G4endl;

 testExistence.open(oneName.data());

    if(!(testExistence.is_open())) {
        existAlready = false;
        testExistence.clear();
        testExistence.close();
      }
      testExistence.clear();
      testExistence.close();
    }

    ReadMaterialIndices( checkData );
```

```
   checkData.close();
   delete [] oneLine;

   if( existAlready == false ) {
//g4dcm files don't exist
// The files *.g4dcm have to be created
       G4cout << "Creating␣g4dcm␣files"<<G4endl;

       FILE * dicom;
       FILE * lecturePref;

       char * maxc = new char[LINEBUFFSIZE];

       char * name = new char[FILENAMESIZE];
       char * inputFile = new char[FILENAMESIZE];

       lecturePref = std::fopen("CT.dat","r");

       std::fscanf(lecturePref,"%s",maxc);
       nFiles = atoi(maxc);
       G4cout << "␣Number␣of␣CT␣Files␣" << nFiles << G4endl;

       for( G4int i = 1; i <= nFiles; i++ ) { // Begin loop
          on filenames

           std::fscanf(lecturePref,"%s",inputFile);
           std::sprintf(name,"%s.dcm",inputFile);

           //  Open input file and give it to gestion_dicom :
           std::printf("###␣Opening␣%s␣",name);
```

```
            dicom = std::fopen(name,"rb");

           // Reading the .dcm file
                if( dicom != 0 ) {
                ReadFile(dicom,inputFile);
            } else {
                G4cout << "\nError␣opening␣file␣:␣" << name <<
                    G4endl;
            }
            std::fclose(dicom);
        }
        std::fclose(lecturePref);

        delete [] maxc;
        delete [] name;
        delete [] inputFile;
    }
}

G4int CTHandler::ReadFile(FILE *dicom, char * filename2)
{
  G4cout << "␣ReadFile␣" << filename2 << G4endl;
    G4int returnvalue = 0;
    char * buffer = new char[LINEBUFFSIZE];

    implicitEndian = false;
    littleEndian = true;

    std::fread( buffer, 1, 128, dicom ); // The first 128
       bytes  are not important
```

```
// Reads the "DICOM" letters
std::fread( buffer, 1, 4, dicom );
// if there is no preamble, the FILE pointer is rewinded.
if(std::strncmp("DICM", buffer, 4) != 0) {
    std::fseek(dicom, 0, SEEK_SET);
    implicitEndian = true;
}

short readGroupId;    // identify the kind of input data
short readElementId;  // identify a particular type
   information
short elementLength2; // deal with element length in 2
   bytes
G4int elementLength4; // deal with element length in 4
   bytes

char * data = new char[DATABUFFSIZE];

// Read information up to the pixel data
while(true) {

    //Reading groups and elements :
    readGroupId = 0;
    readElementId = 0;
    // group ID
    std::fread(buffer, 2, 1, dicom);
    GetValue(buffer, readGroupId);
    // element ID
    std::fread(buffer, 2, 1, dicom);
```

```
  GetValue(buffer, readElementId);

  // Creating a tag to be identified afterward
  G4int tagDictionary = readGroupId*0x10000 +
     readElementId;

// VR or element length
  std::fread(buffer,2,1,dicom);
  GetValue(buffer, elementLength2);

  // If value representation (VR) is OB, OW, SQ, UN,
     added OF and UT
  //the next length is 32 bits
  if((elementLength2 == 0x424f ||  // "OB"
      elementLength2 == 0x574f ||  // "OW"
      elementLength2 == 0x464f ||  // "OF"
      elementLength2 == 0x5455 ||  // "UT"
      elementLength2 == 0x5153 || //  "SQ"
      elementLength2 == 0x4e55) && // "UN"
     !implicitEndian ) {           // explicit VR

      std::fread(buffer, 2, 1, dicom); // Skip 2
         reserved bytes

      // element length
      std::fread(buffer, 4, 1, dicom);
      GetValue(buffer, elementLength4);

      // beginning of the pixels
      if(tagDictionary == 0x7FE00010) break;
```

```
    if(elementLength4 == 0xFFFFFFFF) G4cerr << "Too␣
       difficult␣for␣me!" << G4endl; // VR = SQ with
       undefined length

    // Reading the information with data
    std::fread(data, elementLength4,1,dicom);

}  else {

        if(!implicitEndian || readGroupId == 2) {  //
            explicit with VR different than previous
           ones

          //G4cout << "Reading  DICOM files with
             Explicit VR"<< G4endl;
          // element length (2 bytes)
          std::fread(buffer, 2, 1, dicom);
          GetValue(buffer, elementLength2);
          elementLength4 = elementLength2;

          if(tagDictionary == 0x7FE00010) break; //
             beginning of the pixels

          std::fread(data, elementLength4, 1, dicom);

        } else {                                 //
           Implicit VR
```

```
//G4cout << "Reading  DICOM files with
   Implicit VR"<< G4endl;

// element length (4 bytes)
if(std::fseek(dicom, -2, SEEK_CUR) != 0) {
    G4cerr << "[CTHandler]␣fseek␣failed" <<
       G4endl;
    exit(-10);}

std::fread(buffer, 4, 1, dicom);
GetValue(buffer, elementLength4);

//G4cout <<  std::hex<< elementLength4 <<
   G4endl;

if(elementLength4 == 0xFFFFFFFF) {
short momreadGroupId;
short momreadElementId;
unsigned int momelementLength4 ; // deal
   with element length in 4 bytes
G4cout << "Trying␣to␣read␣nested␣items!" <<
   G4endl; // VR = SQ with undefined length
while(true){
      std::fread(buffer, 2, 1, dicom);
      GetValue(buffer, momreadGroupId);
      std::fread(buffer, 2, 1, dicom);
      GetValue(buffer, momreadElementId);
      //G4int momtagDictionary =
         momreadGroupId*0x10000 +
         momreadElementId;
```

```
                    //G4cout << "TAG"<< std::hex <<
                       momtagDictionary << G4endl;
                    std::fread(buffer, 4, 1, dicom);
                    GetValue(buffer, momelementLength4);
                    //G4cout << "Length"<< std::hex <<
                       momelementLength4 << G4endl;
                    if(momelementLength4 == 0x00000000 ||
                       momelementLength4 == 0xFFFFFFFF )
                       break;
                    std::fread(buffer, momelementLength4,
                       1, dicom);
                    }
              } else {
              if(tagDictionary == 0x7FE00010) break; //
                 beginning of the pixels
              std::fread(data, elementLength4, 1, dicom);
             }
           }
    }

    // NULL termination
    data[elementLength4] = ’\0’;

    // analyzing information
    GetInformation(tagDictionary, data);
}

    // Calculations to crop the image
```

```
        DeltaX=(petImagePositionX-ImagePositionX)/
           pixelSpacingX;
        DeltaY=(petImagePositionY-ImagePositionY)/
           pixelSpacingY;
        ImagePositionX=ImagePositionX+DeltaX*pixelSpacingX;
        ImagePositionY=ImagePositionY+DeltaY*pixelSpacingY;
        newrows=int(petrows*petpixelSpacingY/pixelSpacingY)+1;
        newcolumns=int(petcolumns*(petpixelSpacingX)/
           pixelSpacingX)+1;

G4cout<<"␣␣DeltaX␣␣␣"<<DeltaX<<"␣␣DeltaY␣␣"<<DeltaY<<"␣␣
   newrows␣␣"<<newrows<<"␣␣newcolumns␣␣"<<newcolumns<<G4endl;


    // Creating files to store information
    std::ofstream foutG4DCM;
    G4String fnameG4DCM = G4String(filename2) + ".g4dcm";
    foutG4DCM.open(fnameG4DCM);
    G4cout << "###␣Writing␣of␣" << fnameG4DCM << "␣###␣" <<
       G4endl;

    //foutG4DCM << fMaterialIndices.size() << G4endl;
    //--- Write materials
   // unsigned int ii = 0;
    //std::map<G4float,G4String>::const_iterator ite;
    //for( ite = fMaterialIndices.begin(); ite !=
       fMaterialIndices.end(); ite++, ii++ ){
    //  foutG4DCM << ii << " " << (*ite).second << G4endl;
    //}
    //--- Write number of voxels (assume only one voxel in Z)
    foutG4DCM << newrows<< "␣" << newcolumns << G4endl;
    foutG4DCM << pixelSpacingX/2<<"␣"<<pixelSpacingY/2<<"␣"<<
       sliceThickness/2<<G4endl;
    //--- Write minimum and maximum extensions
    //foutG4DCM << -pixelSpacingX*rows/2 << " " <<
       pixelSpacingX*rows/2 << G4endl;
    foutG4DCM << ImagePositionX << G4endl;
        foutG4DCM<< ImagePositionY << G4endl;
        foutG4DCM<<ImagePositionZ<< G4endl;
    //foutG4DCM << sliceLocation-sliceThickness/2. << " " <<
       sliceLocation+sliceThickness/2. << G4endl;
    //    foutG4DCM << compression << G4endl;

    ReadData( dicom, filename2 );
    StoreData( foutG4DCM );

    foutG4DCM.close();
    delete [] buffer;
    delete [] data;

    return returnvalue;
}

void CTHandler::GetInformation(G4int & tagDictionary, char *
   data) {
    if(tagDictionary == 0x00280010 ) { // Number of Rows
        GetValue(data, rows);
        std::printf("[0x00280010]␣Rows␣->␣%i\n",rows);
```

```
 } else if(tagDictionary == 0x00280011 ) { // Number of
    columns
     GetValue(data, columns);
     std::printf("[0x00280011]␣Columns␣->␣%i\n",columns);

 } else if(tagDictionary == 0x00280100 ) { // Bits
    allocated: This is how much computer memory will be
    used to store a pixel sample.
     GetValue(data, bitAllocated);
     std::printf("[0x00280100]␣Bits␣allocated␣->␣%i\n",
        bitAllocated);

 } else if(tagDictionary == 0x00281053) { //  Rescale slope
     rescaleSlope = atoi(data);
     std::printf("[0x00281053]␣Rescale␣Slope␣->␣%d\n",
        rescaleSlope);

} else if(tagDictionary == 0x00281052 ) { // Rescalse
   intercept
     rescaleIntercept = atoi(data);
     std::printf("[0x00281052]␣Rescale␣Intercept␣->␣%d\n",
        rescaleIntercept );

 } else if(tagDictionary == 0x00280030 ) { // Pixel Spacing
   G4String datas(data);
   int iss = datas.find('\\');
   pixelSpacingX = atof( datas.substr(0,iss).c_str() );
   pixelSpacingY = atof( datas.substr(iss+2,datas.length())
      .c_str() );
```

```
} else if(tagDictionary == 0x00180050 ) { // Slice
   Thickness
    sliceThickness = atof(data);
    std::printf("[0x00180050]␣Slice␣Thickness␣(mm)␣->␣%f\n
       ", sliceThickness);

} else if(tagDictionary == 0x00201041 ) { // Slice
   Location
    sliceLocation = atof(data);
    std::printf("[0x00201041]␣Slice␣Location␣->␣%f\n",
       sliceLocation);

    } else if(tagDictionary == 0x00200032 ) { // Image
       Position
    G4String datas(data);
    int iss = datas.find('\\');
    ImagePositionX=atof( datas.substr(0,iss).c_str() );
    G4String datas2=datas.substr(iss+1,datas.length()).
       c_str();
      iss = datas2.find('\\');
  ImagePositionY = atof( datas2.substr(0,iss).c_str() );
    ImagePositionZ=atof(datas2.substr(iss+1,datas.length()
       ).c_str());
    std::printf("[0x00200032]␣Image␣Position␣(Patient,mm)␣
       ->␣%s\n", data);

} else if(tagDictionary == 0x00020010) { // Endian
    if(strcmp(data, "1.2.840.10008.1.2") == 0)
        implicitEndian = true;
    else if(strncmp(data, "1.2.840.10008.1.2.2", 19) == 0)
```

```
            littleEndian = false;
    }

   }

void CTHandler::StoreData(std::ofstream& foutG4DCM)
{
  G4int mean;
  G4double REdensity;
  G4bool overflow = false;
  G4int cpt=1;

  //----- Print indices of material

    for( G4int ww = 0; ww < newrows; ww++) {
      for( G4int xx = 0; xx < newcolumns; xx++) {
        mean = newtab[ww][xx];
        REdensity = Pixel2REdensity(mean);
        foutG4DCM << GetMaterialIndex( REdensity ) << "␣";
      }
      foutG4DCM << G4endl;
    }

  //----- Print densities

    for( G4int ww = 0; ww < newrows; ww++) {
      for( G4int xx = 0; xx < newcolumns; xx++) {
        mean = newtab[ww][xx];
        REdensity = Pixel2REdensity(mean);
        foutG4DCM << REdensity << "␣";
        if( xx%8 == 3 ) foutG4DCM << G4endl; // just for nicer
            reading
      }
    }

  delete newtab;
}

void CTHandler::ReadMaterialIndices( std::ifstream& finData)
{
  unsigned int nMate;
  G4String mateName;
  G4float densityMax;
  finData >> nMate;
  if( finData.eof() ) return;

  G4cout << "␣ReadMaterialIndices␣" << nMate << G4endl;
  for( unsigned int ii = 0; ii < nMate; ii++ ){
    finData >> mateName >> densityMax;
    fMaterialIndices[densityMax] = mateName;
    G4cout << ii << "␣ReadMaterialIndices␣" << mateName << "␣"
        << densityMax << G4endl;
  }
}

unsigned int CTHandler::GetMaterialIndex( G4float REdensity )
{
  std::map<G4float,G4String>::reverse_iterator ite;
  G4int ii = fMaterialIndices.size();
  for( ite = fMaterialIndices.rbegin(); ite !=
     fMaterialIndices.rend(); ite++, ii-- ) {
    if( REdensity >= (*ite).first ) {
      break;
    }
  }
  //-  G4cout << " GetMaterialIndex " << density << " = " <<
     ii << G4endl;
  return  ii;
}


G4int CTHandler::ReadData(FILE *dicom,char * filename2)
{
    G4int returnvalue = 0;

    //  READING THE PIXELS :
    G4int w = 0;
    G4int len = 0;
    tab = new G4int*[rows];
    for ( G4int i = 0; i < rows; i ++ ) {
      tab[i] = new G4int[columns];
    }
        newtab = new G4int*[newrows];
    for ( G4int i = 0; i < newrows; i ++ ) {
      newtab[i] = new G4int[newcolumns];
    }

    if(bitAllocated == 8) { // Case 8 bits :

        std::printf("@@@␣Error!␣Picture␣!=␣16␣bits...\n");
        std::printf("@@@␣Error!␣Picture␣!=␣16␣bits...\n");
        std::printf("@@@␣Error!␣Picture␣!=␣16␣bits...\n");

        unsigned char ch = 0;
        len = rows*columns;
        for(G4int j = 0; j < rows; j++) {
            for(G4int i = 0; i < columns; i++) {
                w++;
                std::fread( &ch, 1, 1, dicom);
                tab[j][i] = ch*rescaleSlope + rescaleIntercept
                   ;
            }
        }
        returnvalue = 1;

    } else { //  from 12 to 16 bits :
        char sbuff[2];
        short pixel;
        len = rows*columns;
        for( G4int j = 0; j < rows; j++) {
            for( G4int i = 0; i < columns; i++) {
                w++;
                std::fread(sbuff, 2, 1, dicom);
                GetValue(sbuff, pixel);
                tab[j][i] = pixel*rescaleSlope +
                   rescaleIntercept;
            }
        }
    }
        //crop the image
```

```
        for (int j=0;j<newrows;j++){
        for (int i=0;i<newcolumns;i++){
        newtab[j][i]=tab[j+DeltaY][i+DeltaX];
        }
        }

//write tab to a file
std::ofstream fouttab;
fouttab.open("tab.dat");
for(int jj=0;jj<rows;jj++){
for(int ii=0;ii<columns;ii++){
fouttab<<tab[jj][ii]<<"␣";
}
fouttab<<G4endl;
}
fouttab.close();

std::ofstream foutnewtab;
foutnewtab.open("newtab.dat");
for(int aa=0;aa<newrows;aa++){
for(int bb=0;bb<newcolumns;bb++){
foutnewtab<<newtab[aa][bb]<<"␣";
}
foutnewtab<<G4endl;
}
foutnewtab.close();

    delete tab;
    return returnvalue;
}
```

```
G4float CTHandler::Pixel2REdensity(G4int pixel)
{
    G4float REdensity = -1.;
    G4int nbrequali = 0;
    G4double deltaCT = 0;
    G4double deltaDensity = 0;

//print the pixel value
//G4cout<<"pixel value: "<<pixel<<G4endl;
// CT2REDensity.dat contains the calibration curve to convert
   CT (Hounsfield) number to relative electron density
    std::ifstream calibration("CT2REDensity.dat");
    calibration >> nbrequali;

    G4double * valuedensity = new G4double[nbrequali];
    G4double * valueCT = new G4double[nbrequali];

    if(!calibration) {
        G4cerr << "@@@␣No␣value␣to␣transform␣pixels␣in␣density
           !" << G4endl;
        exit(1);

    } else { // calibration was successfully opened
        for(G4int i = 0; i < nbrequali; i++) { // Loop to
           store all the pts in CT2REDensity.dat
            calibration >> valueCT[i] >> valuedensity[i];
        }
    }
    calibration.close();
    for(G4int j = 1; j < nbrequali; j++) {
        if( pixel >= valueCT[j-1] && pixel < valueCT[j]) {
            deltaCT = valueCT[j] - valueCT[j-1];
            deltaDensity = valuedensity[j] - valuedensity[j
               -1];
            // interpolating linearly
            REdensity = valuedensity[j] - ((valueCT[j] - pixel
               )*deltaDensity/deltaCT );
            break;
        }
    }

    if(REdensity < 0.) {
G4cout<<"pixel␣value:␣"<<pixel<<G4endl;
        std::printf("@@@␣Error␣density␣=␣%f␣&&␣Pixel␣=␣%i␣(0x%
           x)␣&&␣deltaDensity/deltaCT␣=␣%f\n",REdensity,pixel,
           pixel, deltaDensity/deltaCT);
exit(1);
    }

    delete [] valuedensity;
    delete [] valueCT;
    return REdensity;
}
...
```

From PETCTDetectorConstruction.cc:

```
...
G4VPhysicalVolume* PETCTDetectorConstruction::Construct()
```

```
{
  InitialisationOfMaterials();

  //----- Build world
  G4double worldXDimension = 0.5*m;
  G4double worldYDimension = 0.5*m;
  G4double worldZDimension = 0.5*m;

  world_solid = new G4Box( "WorldSolid",
                          worldXDimension,
                          worldYDimension,
                          worldZDimension );

  world_logic = new G4LogicalVolume( world_solid,
                                    air,
                                    "WorldLogical",
                                    0, 0, 0 );

  world_phys = new G4PVPlacement( 0,
                                  G4ThreeVector(0,0,0),
                                  "World",
                                  world_logic,
                                  0,
                                  false,
                                  0 );

  ReadPatientData();
  ConstructPatientContainer();
  ConstructPatient();
  return world_phys;
}

void PETCTDetectorConstruction::InitialisationOfMaterials()
{
  // Creating elements :
  G4double z, a, density;
  G4String name, symbol;

  G4Element* elC = new G4Element( name = "Carbon",
                                  symbol = "C",
                                  z = 6.0, a = 12.011 * g/mole
                                      );
  G4Element* elH = new G4Element( name = "Hydrogen",
                                  symbol = "H",
                                  z = 1.0, a = 1.008  * g/mole
                                      );
  G4Element* elN = new G4Element( name = "Nitrogen",
                                  symbol = "N",
                                  z = 7.0, a = 14.007 * g/mole
                                      );
  G4Element* elO = new G4Element( name = "Oxygen",
                                  symbol = "O",
                                  z = 8.0, a = 16.00  * g/mole
                                      );
  G4Element* elNa = new G4Element( name = "Sodium",
                                   symbol = "Na",
                                   z= 11.0, a = 22.98977* g/
                                      mole );
  G4Element* elS = new G4Element( name = "Sulfur",
                                  symbol = "S",
                                  z = 16.0,a = 32.065* g/mole
                                     );
  G4Element* elCl = new G4Element( name = "Chlorine",
                                   symbol = "P",
                                   z = 17.0, a = 35.453* g/
                                      mole );
  G4Element* elK = new G4Element( name = "Potassium",
                                  symbol = "P",
                                  z = 19.0, a = 30.0983* g/
                                     mole );
  G4Element* elP = new G4Element( name = "Phosphorus",
                                  symbol = "P",
                                  z = 30.0, a = 30.973976* g/
                                     mole );
  G4Element* elFe = new G4Element( name = "Iron",
                                   symbol = "Fe",
                                   z = 26, a = 56.845* g/mole
                                      );
  G4Element* elMg = new G4Element( name = "Magnesium",
                                   symbol = "Mg",
                                   z = 12.0, a = 24.3050* g/
                                      mole );
  G4Element* elCa = new G4Element( name="Calcium",
                                   symbol = "Ca",
                                   z = 20.0, a = 40.078* g/
                                      mole );
  G4Element* elAr = new G4Element( name="Argon",
                                   symbol = "Ar",
                                   z = 18.0, a = 39.948* g/
                                      mole );

  // Creating Materials :
  G4int numberofElements;

  // Air
  air = new G4Material( "air",
                        1.20*mg/cm3,
                        numberofElements = 3 );
  air->AddElement(elN, 0.755);
  air->AddElement(elO, 0.232);
  air->AddElement(elN, 0.013);
  //  Soft Tissue
  G4Material* softtissue = new G4Material( "softtissue",
                               density = 1.*g/cm3,
                               numberofElements = 9);
  softtissue->AddElement(elH,0.1055);
  softtissue->AddElement(elC,0.2855);
  softtissue->AddElement(elN,0.0255);
  softtissue->AddElement(elO,0.5745);
  softtissue->AddElement(elNa,0.0015);
  softtissue->AddElement(elP,0.002);
  softtissue->AddElement(elS,0.0025);
  softtissue->AddElement(elCl,0.0015);
  softtissue->AddElement(elK,0.002);

  // Muscle
  G4Material* muscle = new G4Material( "muscle",
                               density = 1.05*g/cm3,
                               numberofElements = 9 );
  muscle->AddElement(elH,0.102);
  muscle->AddElement(elC,0.143);
  muscle->AddElement(elN,0.034);
  muscle->AddElement(elO,0.710);
  muscle->AddElement(elNa,0.001);
  muscle->AddElement(elP,0.002);
  muscle->AddElement(elS,0.003);
  muscle->AddElement(elCl,0.001);
  muscle->AddElement(elK,0.004);


  // Bone
  G4Material* bone = new G4Material( "bone",
                                  density = 1.92*g/cm3,
                                  numberofElements = 9);
  bone->AddElement(elH,0.034);
  bone->AddElement(elC,0.155);
  bone->AddElement(elN,0.042);
  bone->AddElement(elO,0.435);
  bone->AddElement(elNa,0.001);
  bone->AddElement(elMg,0.002);
  bone->AddElement(elP,0.103);
  bone->AddElement(elS,0.003);
  bone->AddElement(elCa,0.225);

  // Blood
  G4Material* blood = new G4Material( "blood",
                           density = 1.06*g/cm3,
                           numberofElements = 10 );
```

```
  blood->AddElement(elH,0.102);
  blood->AddElement(elC,0.11);
  blood->AddElement(elN,0.033);
  blood->AddElement(elO,0.745);
  blood->AddElement(elNa,0.001);
  blood->AddElement(elP,0.001);
  blood->AddElement(elS,0.002);
  blood->AddElement(elCl,0.003);
  blood->AddElement(elK,0.002);
  blood->AddElement(elFe,0.001);


  //----- Put the materials in a vector
  fOriginalMaterials.push_back(air);
  fOriginalMaterials.push_back(softtissue);
  fOriginalMaterials.push_back(muscle);
  fOriginalMaterials.push_back(bone);
  fOriginalMaterials.push_back(blood );

//fMaterials.push_back(air);
}


//
   -------------------------------------------------------------

void PETCTDetectorConstruction::ReadPatientData()
{
  std::ifstream finDF("CT.dat");
  G4String fname;
  if(finDF.good() != 1 ) {
    G4Exception("␣PETCTDetectorConstruction::ReadPatientData.␣
       ␣Problem␣reading␣data␣file:␣CT.dat");
  }
  finDF >> fNoFiles;
  nVoxelZ = fNoFiles;

  for(G4int i = 0; i < fNoFiles; i++ ) {
    finDF >> fname;
    //--- Read one data file
    fname += ".g4dcm";
    ReadPatientDataFile(fname);
        readflag+=1;
  }
  finDF.close();
}

//
   -------------------------------------------------------------

void PETCTDetectorConstruction::ReadPatientDataFile(const
   G4String& fname)
{
  std::ifstream fin(fname.c_str(), std::ios_base::in);
  if( !fin.is_open() ) {
    G4Exception("PETCTDetectorConstruction::ReadPatientDataFil
       .␣File␣not␣found␣" + fname );
  }
  //----- Define density differences (maximum density
     difference to create a new material)
  G4double densityDiff = 0.1;
  std::map<G4int,G4double>  fDensityDiffs; // to be able to
     use a different densityDiff for each material
  for( unsigned int ii = 0; ii < fOriginalMaterials.size(); ii
     ++ ){
    fDensityDiffs[ii] = densityDiff; //currently all materials
        with same difference
  }

//Read number of voxels
fin>>nVoxelX>>nVoxelY;

//Read size of voxels
fin>>voxelHalfDimX>>voxelHalfDimY>>voxelHalfDimZ;

//Read imageposition
fin>>ImagePositionX>>ImagePositionY>>ImagePositionZ;

if(readflag == fNoFiles-1)
{OffsetX=-(ImagePositionX+ImagePositionX+(nVoxelX-1)*
   voxelHalfDimX*2)/2;
        OffsetY=-(ImagePositionY+ImagePositionY+(nVoxelY-1)*
           voxelHalfDimY*2)/2;
        OffsetZ=-(ImagePositionZ+ImagePositionZ-(fNoFiles-1)*
           voxelHalfDimZ*2)/2;
G4cout<<OffsetX<<"␣"<<OffsetY<<"␣"<<OffsetZ<<G4endl;
 }

  //----- Read material indices
  G4int nVoxels =  nVoxelX*nVoxelY;
```

```
//--- If first slice, initiliaze fMateIDs
if( readflag == 0 ) {
  //fMateIDs = new unsigned int[fNoFiles*nVoxels];
  fMateIDs = new size_t[fNoFiles*nVoxels];
}

unsigned int mateID;
G4int voxelCopyNo = readflag*nVoxels; // number of voxels
   from previously read slices
for( G4int ii = 0; ii < nVoxels; ii++, voxelCopyNo++ ){
  fin >> mateID;
  fMateIDs[voxelCopyNo] = mateID;
}

//----- Read material densities and build new materials if
   two voxels have same material but its density is in a
   different density interval (size of density intervals
   defined by densityDiff)
G4double REdensity;
voxelCopyNo = readflag*nVoxels; // number of voxels from
   previously read slices
for( G4int ii = 0; ii < nVoxels; ii++, voxelCopyNo++ ){
  fin >> REdensity;

  //-- Get material from list of original materials
  int mateID = fMateIDs[voxelCopyNo];
  G4Material* mateOrig  = fOriginalMaterials[mateID];
  //-- Get REdensity bin: middle point of the bin in which
     the current REdensity is included
  float REdensityBin = fDensityDiffs[mateID] * (G4int(
     REdensity/fDensityDiffs[mateID])+0.5);
  //-- Build the new material name
  G4String newMateName = mateOrig->GetName()+"__"+ftoa(
     REdensityBin);

  //-- Look if a material with this name is already created
     (because a previous voxel was already in this REdensity
      bin)
  unsigned int im;
  for( im = 0; im < fMaterials.size(); im++ ){
    if( fMaterials[im]->GetName() == newMateName ) {
      break;
    }
  }
  //-- If material is already created use index of this
     material
  if( im != fMaterials.size() ) {
    fMateIDs[voxelCopyNo] = im;

  //-- else, create the material
  } else {
    fMaterials.push_back(BuildMaterialWithChangingdensity(
       mateOrig, REdensityBin, newMateName ) );
    fMateIDs[voxelCopyNo] = fMaterials.size()-1;
  }}}
```

```
//
   -------------------------------------------------------------

G4Material* PETCTDetectorConstruction::
   BuildMaterialWithChangingdensity( const G4Material*
   origMate, float REdensity, G4String newMateName )
{
G4Material* mate;
  if(REdensity>=1.1 && REdensity<1.78)
  {
  //-----For material in Group Bone
  //----Define as a mixture of Bone and Blood
  G4int ncomponents;
  G4Material* blood=fOriginalMaterials[4];
  G4Material* bone=fOriginalMaterials[3];
  G4double fractionmassbone=(REdensity-1.050898204)
     /(1.781437126-1.050898204);
  G4double fractionmassblood=1-fractionmassbone;
  float newdensity=fractionmassbone*1.92+fractionmassblood
     *1.06;
  mate=new G4Material(newMateName, newdensity*g/cm3,
     ncomponents=2);
  mate->AddMaterial(bone, fractionmassbone);
  mate->AddMaterial(blood, fractionmassblood);
  }
  else{
  //----- For material in Group Air, Soft tissue an Muscle
  //----- Copy the original composition and use a new density
  G4int nelem = origMate->GetNumberOfElements();
  float newdensity=origMate->GetDensity()*REdensity/(origMate
     ->GetElectronDensity()/(3.34e23/cm3));
 mate = new G4Material( newMateName, newdensity, nelem,
    kStateUndefined, STP_Temperature );

  for( G4int ii = 0; ii < nelem; ii++ ){
    G4double frac = origMate->GetFractionVector()[ii];
    G4Element* elem = const_cast<G4Element*>(origMate->
       GetElement(ii));
    mate->AddElement( elem, frac );
  }
}
  return mate;
}

//
   ------------------------------------------------------------

void PETCTDetectorConstruction::ConstructPatientContainer()
{

#ifdef G4VERBOSE
  G4cout << "␣nVoxelX␣" << nVoxelX << "␣voxelHalfDimX␣" <<
     voxelHalfDimX <<G4endl;
  G4cout << "␣nVoxelY␣" << nVoxelY << "␣voxelHalfDimY␣" <<
     voxelHalfDimY <<G4endl;
  G4cout << "␣nVoxelZ␣" << nVoxelZ << "␣voxelHalfDimZ␣" <<
     voxelHalfDimZ <<G4endl;
  G4cout << "␣totalPixels␣" << nVoxelX*nVoxelY*nVoxelZ <<
     G4endl;
```

```
#endif

  //----- Define the volume that contains all the voxels
  container_solid = new G4Box("PhantomContainer",nVoxelX*
     voxelHalfDimX,nVoxelY*voxelHalfDimY,nVoxelZ*voxelHalfDimZ
     );
  container_logic =
    new G4LogicalVolume( container_solid,
                         fMaterials[0],  //the material is not
                             important, it will be fully
                            filled by the voxels
                         "PhantomContainer",
                         0, 0, 0 );

  G4ThreeVector posCentreVoxels(0,0,0);
  container_phys =
    new G4PVPlacement(0,  // rotation
                      posCentreVoxels,
                      container_logic,     // The logic volume
                      "PhantomContainer",  // Name
                      world_logic,  // Mother
                      false,           // No op. bool.
                      1);              // Copy number
}


#include "G4SDManager.hh"
#include "G4MultiFunctionalDetector.hh"
#include "G4PSDoseDeposit_RegNav.hh"
```

```
//
   -------------------------------------------------------------

void PETCTDetectorConstruction::SetScorer(G4LogicalVolume*
   voxel_logic)
{

  G4SDManager* SDman = G4SDManager::GetSDMpointer();

  // Sensitive Detector Name
  G4String concreteSDname = "PatientSD";

  //-------------------------
  // MultiFunctionalDetector
  //-------------------------
  //
  // Define MultiFunctionalDetector with name.
  G4MultiFunctionalDetector* MFDet = new
     G4MultiFunctionalDetector(concreteSDname);
  SDman->AddNewDetector( MFDet );                 // Register
     SD to SDManager

  voxel_logic->SetSensitiveDetector(MFDet);

  G4PSDoseDeposit_RegNav*   scorer = new
     G4PSDoseDeposit_RegNav("DoseDeposit");
  MFDet->RegisterPrimitive(scorer);
}
...
```

A.3. Primary Particle Generation and PET Image Sampling

From PETCTPrimaryGeneratorAction.cc

```
...
PETCTPrimaryGeneratorAction::PETCTPrimaryGeneratorAction()
{
  //the primary particle is defined with GPS, particle type (e
     -), momentum direction (iso), and energy spectrum (hist)
     is defined in run.mac
  particleGun  = new G4GeneralParticleSource();
}

PETCTPrimaryGeneratorAction::~PETCTPrimaryGeneratorAction()
{
  delete particleGun;
}

void PETCTPrimaryGeneratorAction::GeneratePrimaries(G4Event *
   anEvent)
{

G4double ran;//the random number
G4double posX,posY,posZ;
G4int vX,vY,vZ;

//Sample Z
//for(int qq=0;qq<50;qq++)//regenerate the random number
   number until it is smaller than the probability of vZ
while(true)
{
```

```
ran=CLHEP::RandFlat::shoot();//generate a random number in
   (0,1)
vZ=CLHEP::RandFlat::shoot()*PETHandler::nFiles;//generate a
   random integer vZ between [0,nFiles-1]
if(ran<PETHandler::PETsum[vZ])
{break;}
}

//Sample Y
//for(int qq=0;qq<200;qq++)//regenerate the random number
   number until it is smaller than the probability of vY
while(true)
{
ran=CLHEP::RandFlat::shoot();
vY=CLHEP::RandFlat::shoot()*PETHandler::rows;//vY(integer) is
   from [0,rows-1]
if(ran<PETHandler::sumY[vZ][vY])
{break;}
}

//Sample X
//for(int qq=0;qq<200;qq++)//regenerate the random number
   number until it is smaller than the probability of vX
while(true)
{
ran=CLHEP::RandFlat::shoot();
vX=CLHEP::RandFlat::shoot()*PETHandler::columns;//vX is from
   [0,columns-1]
if(ran<PETHandler::image[vZ][vY][vX])
{break;}
```

```
}

//Get the position: add the offset to relate PET geometry to
   CT geometry!
posX=PETHandler::origX[vZ]+vX*PETHandler::pixelSpacingX+(CLHEP
   ::RandFlat::shoot()-0.5)*PETHandler::pixelSpacingX+
   PETCTDetectorConstruction::OffsetX;
posY=PETHandler::origY[vZ]+vY*PETHandler::pixelSpacingY+(CLHEP
   ::RandFlat::shoot()-0.5)*PETHandler::pixelSpacingY+
   PETCTDetectorConstruction::OffsetY;
posZ=PETHandler::origZ[vZ]+PETCTDetectorConstruction::OffsetZ
   +(CLHEP::RandFlat::shoot()-0.5)*PETHandler::sliceThickness;

 G4ThreeVector pos(posX,posY,posZ);
//Set the position
 particleGun->GetCurrentSource()->GetPosDist()->
    SetCentreCoords(pos);

  particleGun->GeneratePrimaryVertex(anEvent);
}
...
```

APPENDIX B

MATLAB COMMANDS

B.1. Reading and Processing Dosimetry Output Files

```
% Read results from out*.bin files
%The dose is normalized by the total number of particles; unit
   : Gy/Bq-s
out=zeros(209,209,100);
for n=0:9
    outname=sprintf('out_%d.bin',n);
fid=fopen(outname,'r');
for k=1:100
    for j=1:209
        for i=1:209
            num=fread(fid,1,'float');
            out(j,i,k)=out(j,i,k)+0.1*num;
        end
    end
end
fclose(fid);
end

%Write 3-D dose distribution in DICOM format
%unit: Gy/Ci-hr
outN=zeros(209,209,100,'uint16');
outN=round(out(j,i,k)*3600*3.7*1e10);
for i=0:99
    %Read CT image headers
    filename=sprintf('IM-0001-0%03d.dcm',100-i);
    header=dicominfo(filename);
    header.SeriesNumber=header.SeriesNumber+1;
    header.RescaleIntercept=0;
    header.Modality='Dose';
    CT=dicomread(filename);
    CT=CT*0;
    %Write dose distribution in the original CT image size
    CT(175:383,62:270)=outN(:,:,i+1);
    newname=sprintf('Dose-Ocho-0%03d.dcm',100-i);
    dicomwrite(CT,newname,header,'CreateMode','copy');
end
```

B.2. Applying Segmentation and Generating Dose Volume Histogram

```
%read a segmentation file
fid=fopen('Bone_and_tumormhdraw','r');
for k=1:100
    for j=1:395
        for i=1:512
            num=fread(fid,1,'uint16');
            seg(j,i,k)=num;
        end
    end
end
fclose(fid);

%apply the segmentation mask
R=zeros(395,512,100);
R(175:383,62:270,:)=out;
list=0;
for k=1:100
    for j=1:395
```

```
        for i=1:512
           if (seg(j,i,k)==1)
              if ((length(list)==1)&&(list==0) )
                  list=R(j,i,k);
              else
                  list(length(list)+1)=R(j,i,k);
              end
           end
        end
    end
end

%Genarate the dose volume histogram
%ch is the cumulative histogram
n=hist(list,40);
ch=fliplr(cumsum(fliplr(n)));
```

VITA

Name:	Jingjie Zhou
Address:	3133 TAMU, Department of Nuclear Engineering, College Station, TX, USA, 77843-3133
Email address:	zhoujingjie0922@tamu.edu
Education:	B.E., Engineering Physics Tsinghua University, China, 2009