2018•2019

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: nucleaire technologie

Masterthesis

Computer vision guided Monte Carlo dose calculations in interventional radiology

PROMOTOR :

Prof. dr. Brigitte RENIERS

PROMOTOR :

Prof. dr. ir. Frank VERHAEGEN

Nick Staut

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: nucleaire technologie, afstudeerrichting nucleaire technieken / medisch nucleaire technieken

Confidentieel

UHASSELT

KU LEUVEN

Preface

My whole life I have been fascinated by both medicine and technology which eventually brought me to the area of medical imaging as a radiographer. Once I started working in this field I gained an increasing interest in dosimetry and radiation physics, this led me to nuclear engineering technology at UHasselt-KULeuven.

First and foremost, I would like to thank my internal and external supervisors prof. Brigitte Reniers and prof. Frank Verhaegen for giving me the opportunity to work on such an exciting project and their guidance during the completion of this master's thesis.

This project could not have been completed without the financial support provided by Interreg Vlaanderen-Nederland under the form of the DoseGuard project. I would like to thank them and all the people who made the start of this project possible by providing their support: prof. Joachim Wildberger (Maastricht UMC+), Françoise Malchair (ZEPHYRA scrl) and Dr Emiliano D'Agostino (DoseVue).

I am especially grateful to Dr Blake Walters who made several adaptations to the source code of the Monte Carlo engine on our request and provided valuable feedback and detailed explanations regarding the use of the code.

Further thanks go to the people at the AZm radiology department specifically medical physicist Dr Cecile Jeukens and radiological lab technicians Pascal Aerts and Ellen Bouchoms for providing an insight in the inner workings of an interventional radiology procedure, providing example data, allowing access to their equipment and making time to operate the equipment during our measurements.

I would also like to thank all people at SmART Scientific Solutions, MAASTRO Clinic and the University of Hasselt who gave valuable feedback, guidance and helped with experiments, in particular: Stefan Van Hoof, Dr Gabriel Paiva Fonseca, Ana Vaniqui, Dr Mark Podesta, Murillo Bellezzo, Brent van der Heyden, José Baeza, and Gerben Bosmans.

In addition, I would also want to thank my ex-colleagues, especially the deputy head nurse Koen Van Caneghem and head nurse Dirk Van Vaerenbergh, at my previous employer AZ Nikolaas, for supporting me in my decision to change my career path and enabling me to start this education without losing my livelihood by adding flexibility to my working schedule.

Last but not least, I would like to thank my friends and family for the support and especially my partner Maxime Trienpont for motivating me to follow my dream and pursue this master's degree and for her patience and continued support during this entire journey.

Table of Contents

List of tables

List of figures

Abstract DoseGuard

The increasingly strict legislation regarding dose monitoring and high skin doses associated with interventional radiology dictate the need for accurate dose calculations. The current methods based on the radiation dose structured report (RDSR) data or indirect dose metrics have high inaccuracies.

An automated dose calculation system based on Monte Carlo simulations and computer vision using three RGB-D cameras was developed. The calibrated camera system determines the geometric relationship between the patient and X-ray source and combines this with the RSDR data to generate an input for a Monte Carlo simulation. The patients are modelled using a family of mathematical phantoms.

Skin dose calculations can be performed under 10 seconds within 10% uncertainty for all field sizes and within 5% uncertainty for all field sizes smaller than 15 cm x 15 cm. Organ dose calculations can also be executed offline with a longer calculation time. Initial tests with a limited number of beam positions were performed with GafChromic film and the simulations show differences below 5%, which is within the range of the measurement and simulation uncertainties. Visual evaluation shows good agreement over a large range of beam positions.

The created dose calculation system shows that the combination of Monte Carlo and computer vision has the potential to improve the accuracy of dose calculations in radiology, even providing near real time skin dose feedback during interventional procedures.

Abstract (Dutch)

Met de strengere wetgevingen betreffende stralingsdosisregistratie en de hoge huiddosissen bij interventionele radiologie is er nood aan nauwkeurige dosisberekeningen. De huidige methoden op basis van de RDSR-gegevens of indirecte metingen hebben hoge onnauwkeurigheden.

Een geautomatiseerd dosisberekeningssysteem werd ontwikkeld op basis van Monte Carlo simulaties en computervisie dat gebruik maakt van drie RGB-D camera's. Het gekalibreerde camerasysteem bepaalt de geometrische relatie tussen de patiënt en de stralingsbron en combineert dit met de gegevens uit het RDSR om input te genereren voor een aangepaste DOSXYZnrc code. De patiënten worden gemodelleerd door een familie van XCAT-fantomen.

Huiddosisberekeningen kunnen in minder dan 10 seconden uitgevoerd worden voor alle veldgroottes met een onzekerheid kleiner dan 10%, en minder dan 5% voor veldgroottes kleiner dan 15 cm x 15 cm. Orgaandosissen vereisen een langere rekentijd. Initiële testen bij een beperkt aantal buis posities werden uitgevoerd met GafChromic film en het verschil tussen meting en berekening was telkens kleiner dan 5%, wat binnen de onzekerheid valt. Visuele evaluatie van een groot aantal verschillende posities toonde goede overeenkomsten.

Het ontwikkelde dosisberekeningssysteem toont dat de combinatie van Monte Carlo met computervisie potentieel heeft om de nauwkeurigheid van dosisberekeningen in de radiologie te verbeteren. Het zou mogelijk zijn real time huiddosis informatie te geven tijdens interventionele procedures.

1. Introduction

X-rays are inseparable from the medical industry ever since Wilhelm Conrad Röntgen discovered them in 1895. Technological innovations over the course of decades introduced new applications such as radiography, fluoroscopy, mammography, angiography and computed tomography. Despite the development of imaging technologies without ionizing radiation such as ultrasound and magnetic resonance imaging, these X-ray applications are still irreplaceable and frequently used in today's routine clinical practice [1], [2]. The dangers associated with ionizing radiation such as X-rays became apparent in the early years following its discovery. Both deterministic effects such as dermatitis, cataract and cardiac problems and stochastic effects like the development of cancer were frequently observed in the population of early scientists working with ionizing radiation [3].

Development of complex surgical procedures and additional medical disciplines utilizing imaging in their patient care both attributed to the dramatic increase of radiation doses that patients received from fluoroscopic examinations [4]. Interventional procedures such as cardiovascular interventions can potentially introduce skin lesions due to high skin doses caused by prolonged exposure during fluoroscopy [5].

Radiation protection guidelines became increasingly restrictive over the last century. Most recent European Basic Safety Standards (EC directive 2013/59/EURATOM) emphasize the need for dose management. They state information related to patient exposure has to be part of the report of the medical radiological procedure, therefore an accurate measurement or calculation of the dose is required [6], [7].

This thesis approaches the dose calculation problem from another perspective by incorporating proven techniques from radiotherapy, such as Monte Carlo simulations, and not limiting the calculations to only skin dose. The goal is to enhance the accuracy of dose calculations in the field of radiology, focusing on interventional radiology. Monte Carlo dose calculations have been considered the gold standard in radiation therapy for decades [8]. We propose a method for automated skin and organ dose calculations in interventional radiology based on Monte Carlo methods and a color and depth (RGB-D) camera system.

2. Background information

2.1. Interventional radiology

Interventional radiology uses a range of medical imaging applications to guide the physician during a treatment. These procedures are often used as a less invasive alternative to surgery [9]. One of the most well-known applications is cardiovascular interventional radiology. During such an intervention a catheter is inserted into the blood vessels that is then used as both a diagnostic observational and a therapeutic tool. The first therapeutic procedure was performed in by Charles Dotter in 1964, 11 years after the first diagnostic catheterization method [10]. This field experienced tremendous growth over the past 50 years contributing to an increase in patients exposed to high radiation doses associated with complex procedures [4], [11]. Radiation induced skin lesions as a result of these interventions have been reported [5].

2.2. Skin dose monitoring

Skin dose measurements are most frequently performed using thermoluminescent detectors (TLD), MOSFETS or radiochromic film [12]. Measurements are rarely used in clinical practice due to the amount of work and difficulty in accurately processing the results, neither do they give online information regarding skin dose [13].

Initial attempts to provide real time skin dose feedback were made by searching for a correlation between the measured skin dose and dose area product (DAP). While DAP can be used to set dose reference levels to promote good practice, it is independent of the source to patient distance; therefore, it cannot be used as an indicator for local exposure [5], [13]. Other indirect metrics such as air kerma at a reference point can also be used. However, all these indirect dose metrics have high inaccuracies that can go up to 50% [14].

Recently some vendors offer dose calculation software that provides skin dose feedback. Offline vendor agnostic software solutions like DICOM dose (DiDo) and em.dose use the radiation dose structure report (RDSR) which provides a better result than resorting to indirect dose metrics. However not all C-arm manufacturers include information such as couch displacement to their RDSR, limiting the accuracy of these methods [12], [15]. Manufacturers started integrating real time dose mapping software with their newer machines. An example is the Dose Tracking System (Toshiba Medical), this software provides some improvements such as allowing the user to input a geometry resembling the patient. Skin dose is calculated on this geometry by applying an inverse square correction to each element of their graphic intersecting with an incident X-ray. The conversion factor for this method is determined using an ionization chamber [16]. These software solutions must resort to rough estimations of where the patient could be positioned introducing additional uncertainties or introducing errors in less commonly used tube positions. An example of such an error is the close correspondence found in most planes between CareGraph® (Siemens HealthCare) and GafChromic film except for a 78% underestimated dose in the lateral plane in [17]. Table 1 shows an overview of commercially available software solutions that provide skin dose information, note that this list contains most known commercial solutions but should not be considered exhaustive.

Table 1 – Overview of available software solutions for dose management and calculation in radiology

Software (company)	Vendor specific	Target objective	Skin dose	Organ dose
Dose (Qaelum) [18]	No	Dose management (post-examination)	Peak Skin Dose and dose map using RDSR information on unknown model	CT only: Monte Carlo by Virtual Dose® CT.
Teamplay (Siemens) [19], [20]	No	Dose management (post-examination)	Not specified	Not specified
DoseWatch (GE) [21], [22]	No	Dose management (post-examination)	Air Kerma in reference point (15 cm of isocenter)	CT only: now SSDE with conversion factor. Convolution with XCAT future release
DoseWise (Philips) [23], [24]	No	Dose management (post-examination)	Peak Skin Dose and Air Kerma in reference point	CT only: unknown calculation method
DoseTrack (Sectra) [25], [26]	No	Dose management (post-examination)	Fluoro skin dose at point of reporting	CT only: Monte Carlo by Virtual Dose® CT
Radimetrics (Bayer) [27], [28]	No	Dose management (post-examination)	Peak Skin Dose	Pre-calculated Monte Carlo on Christy phantoms.
DoseMonitor (PACS Health) [29], [30]	No	Dose management (post-examination)	Peak Skin Dose without specification of method used	CT only: Monte Carlo by Virtual Dose® CT
OpenREM (open source) [31]	No	Dose management (post-examination)	Skin entrance dose with backscatter correction based on kVp with no filtration on water cuboid with 2 semicylinders on each side	No
CARE+CLEAR and CareGraph (Siemens) [17]	Yes	Skin dose feedback (real-time)	Real time Air kerma in reference point and Peak Skin Dose based on exposure and geometric data from unit	No
Infinix-i Dose Tracking System (Toshiba) [16], [32]	Yes	Skin dose feedback (real-time)	Real time dose map base on distance corrections for a virtual phantom and correlated to calibrated ionization chamber measurements	No
em.dose (esprimed) [12]	No	Skin dose feedback (post-examination)	Dose map based on RDSR information	No

2.3. Monte Carlo for radiation dose calculations

The Monte Carlo method is a computational technique that uses random sampling to obtain numerical results for complex models. While stochastic sampling methods were used long before computers were invented, the computational technique was first invented by Ulam in the 40's. Ulam communicated this to von Neumann who was working alongside him on theoretical calculations related to the development of a nuclear fission bomb. The development required precise calculation of neutron transport and von Neumann suggested applying this stochastic sampling method to radiation transport calculations. In 1949 Ulam and von Neumann published their paper coined "The Monte Carlo Method" associating the name, "Monte Carlo" with stochastic sampling [33].

In [33] the Monte Carlo method is defined as: a numerical method to solve equations or to calculate integrals based on random number sampling. When applying this technique to radiation transport a single incident particle is tracked through a geometry. Each interaction has a probability of occurring and for each potential interaction a random number is generated determining whether a certain interaction should take place, this process is then repeated for a large number of so-called histories. The actual implementation is orders of magnitudes more complex but out of the scope of this discussion [34].

In medical physics the most commonly used Monte Carlo codes for particle transport are MCNP (Monte Carlo N-Particle), EGS (Electron Gamma Shower) and the more recently developed GEANT (GEometry ANd Tracking). These three codes are often considered to be the gold standard in dose calculations in radiation therapy. MCNP is a general-purpose code developed at Los Alamos National Laboratory that has state-of-the art neutron transport which makes it the first choice for many nuclear and radiological applications, electron transport was first added in 1990 after which it became popular in the medical physics field. GEANT was originally developed for high-energy physics at CERN and SLAC but supports many-particle transport over a wide range of energies and has made its impact in medical physics. EGS on the other hand specifically targeted the medical field but has been used in applications outside of medical physics [33], [34].

EGSnrc (NRCC, Canada) is a version of EGS by the National Research Council Canada, that was released as the successor for EGS4. It is designed for coupled electron-photon transport and supports particle energies ranging from 1 keV to several hundreds of GeV [35]. Two EGSnrc-based simulation codes dedicated to radiation treatment planning are available: BEAMnrc, a simulation system for modelling radiotherapy sources and DOSXYZnrc to perform dose distribution calculations in a rectilinear voxel phantom [36], [37].

2.4. Computer vision

Computer vision aims to build systems that can perform similar or better than the human vision system. Computer vision originates from a summers student's project in the sixties, making a computer see was perceived as a simple task at that time. More than fifty years later it is still a large challenging research field in computer science and has grown even more due to recent developments in artificial intelligence. Many other fields such as image processing, photogrammetry and computer graphics also contribute to development of computer vision applications [38], [39].

While being a research field with many challenges that have not yet been solved and mimicking human vision and improving on it is still out of reach. Many useful applications have already been developed using computer vision, this is done by using a goal driven approach. This implies that the vision system only needs to extract enough information from its surroundings to fulfil its tasks. A few successful examples are: industrial inspection applications which use 2D image processing and pattern recognition; automatic license plate reading; augmenting movies with virtual objects and computer games that allow real time detection of the players such as the Xbox Kinect [39], [40].

Perhaps the most important of a computer vision system are the sensors that provide the visual data. The camera type must be carefully selected to fit its task, having appropriate properties such as the correct resolution, working under the application's lighting conditions, etc. Manufacturing imperfections make every camera unique having slightly different lens imperfections or other parameters. Depending on the application careful calibration might be required to compensate for errors introduced due to such imperfections [41].

A camera maps the 3D world into a 2D image, an important and frequently used method of modelling how the camera maps the 3D world is the pinhole camera. This pinhole camera uses a perspective projection where all 3D points are mapped to an image plane [38].

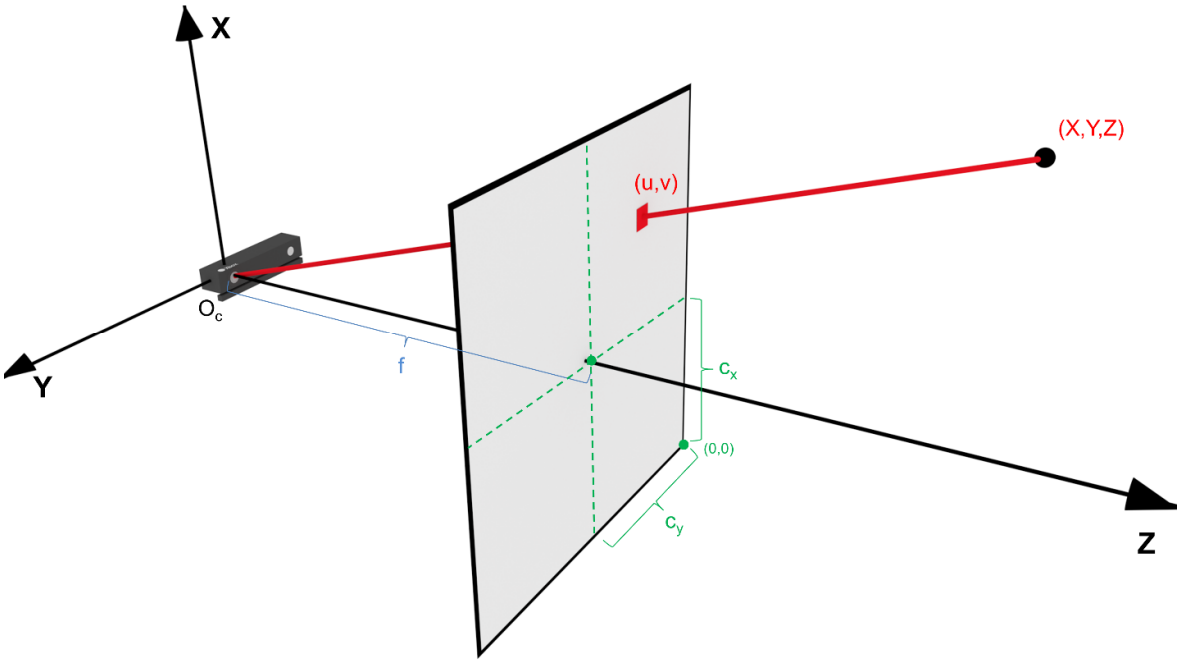

Figure 1 – Pinhole camera model where the coordinate system is defined with the Z-axis as principal axis

This pinhole camera model is shown in Figure 1. The optical center (O_c) of the camera is an infinitesimally small point in three-dimensional space where all rays, that are projected on top of the image plane, must cross. The principal point (c_x, c_y) is defined as the point of the image plane where the principal axis crosses. This principal axis is the line perpendicular to the image plane that crosses the optical center, in Figure 1 the Z-axis is the principal axis. The distance between the optical center and the intersection with the image plane is called the focal length f (f_x,f_y), in an ideal camera model the values of f_x and f_y are identical, but multiple factors can introduce differences, such as: flaws in the digital camera sensor, non-uniform scaling in postprocessing and several other factors [38]. From the schematic representation similar triangles can be used to find the relation between the pixel position and the 3D coordinate, this relationship is shown in equation (1).

$$u = \frac{X\, f_x}{Z} + c_x$$
$$v = \frac{Y\, f_y}{Z} + c_y \qquad (1)$$

This pinhole model assumes the camera is a perfect pinhole however in practice this is not the case and the projection of points is distorted by the camera's lens. This lens distortion is caused by different types of imperfections in design and assembly of lenses. Therefore, when calibrating a camera to determine the intrinsic camera parameters (focal length and principal point) corrections

for lens distortion must first be applied. While there are many forms of distortion the most commonly considered are radial and tangential distortion. A practical implementation of such lens distortion correction can be seen in Appendix A [42], [43].

Different calibration methods exist to determine these intrinsic parameters and distortion coefficients, most calibration methods can be classified as self-calibration and photogrammetric calibration methods. Self-calibration consists of moving the camera in a static scene where the correspondences between the images can be used to determine the parameters. This technique however is not mature, and the results are not always reliable. Photogrammetric calibration methods are very efficient but often require an expensive and elaborate setup. An alternative is Zhang's method, where a planar pattern is captured by the camera from different orientations by moving either the camera or the pattern [44].

3. Methods and materials

A dose calculation system using color and depth (RGB-D) cameras was developed in MATLAB (version 9.5, R2018b, The MathWorks Inc., Ma). Additional C++ libraries such as the open source Kinect v2 driver Libfreenect2 and the Open Source Computer Vision Library (OpenCV 3.4.0) were used for streaming and processing the image data [42], [45]. MexOpenCV is an open source development kit for MATLAB Mex functions for the OpenCV library, this package was used to interface between MATLAB and the C++ libraries. Several computationally intensive functions were also written in C++ and interfaced with MATLAB using the data type conversion template provided by MexOpenCV [46]. Absolute dose calculations are performed by interfacing with the Monte Carlo photon transport program DOSXYZnrc. DOSXYZnrc is a Monte Carlo simulation code based on EGSnrc that allows for dose distribution calculations in a rectilinear voxel phantom [37].

Figure 2 – Schematic representation of the dose calculation system setup in an interventional radiology room with the results displayed on the right side. A short animation video illustrating the setup can be found at https://youtu.be/yG9pjMmbv69

The dose calculation system was designed to perform fully automated absolute skin and organ dose calculations for an interventional radiology procedure. As seen in Figure 2 the patient and X-ray machine are tracked by a camera system. Several visual markers are attached to the X-ray device to provide fast and robust tracking of the source. Both marker-based and markerless tracking can be used to determine the patient's position and geometry.

While the system was designed specifically for interventional radiology, small modifications permit usage in CT or conventional radiology. Those modifications could be simplifications such as reducing the number of cameras as there would be less occlusion, or adapting the source model to match a fan or cone beam CT.

3.1. System flow

The prototype of the system provides offline dose calculations based on the structured reports generated after the interventional procedure is finished. However, with minor adaptations the system can be used to provide “real time” feedback about the skin dose received by the patient.

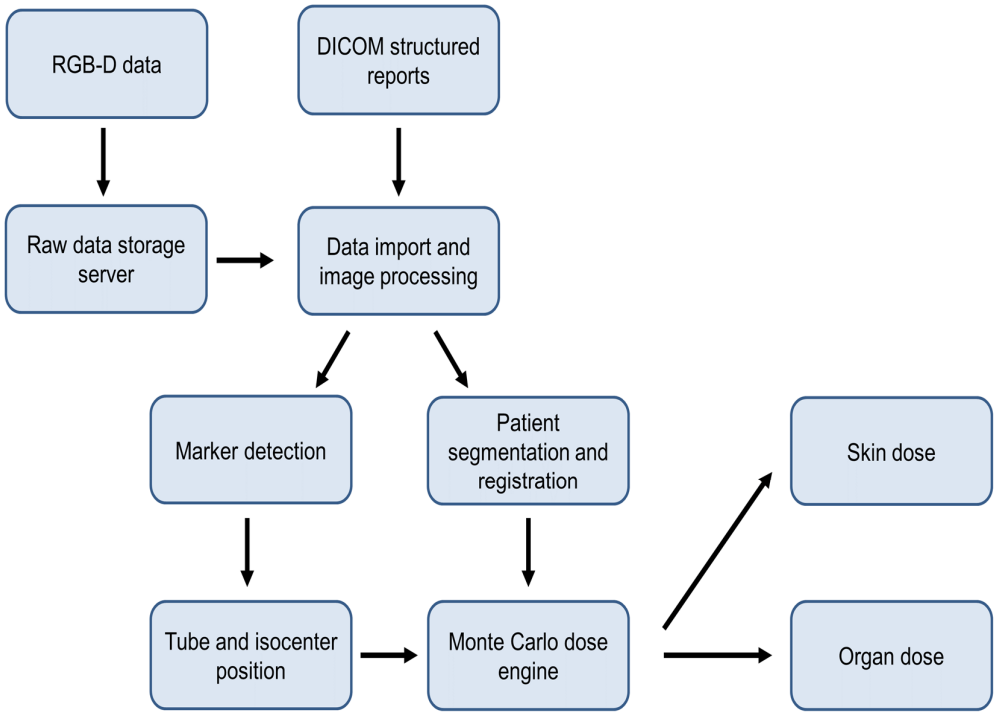

Figure 3 – System flow chart of developed prototype

Figure 3 shows the flow of the dose calculation system. The image data is collected on dedicated single board computers connected to the RGB-D cameras. These single board computers are connected to a server through ethernet where all raw image data is saved for later use. After the structured reports are available they can be imported together with the recorded image data of the examination and the system can perform processing. First the marker positions will be detected and combined with the structured report data the source and isocenter position will be assigned. The patient body parts will be segmented and registered with a predetermined mathematical phantom. Afterwards an EGSnrc input file will be generated and fed to the DOSXYZnrc executable while monitoring the progress. When the dose calculation is completed the binary 3ddose file is parsed and the dose information is visually and numerically represented for both skin and organ doses.

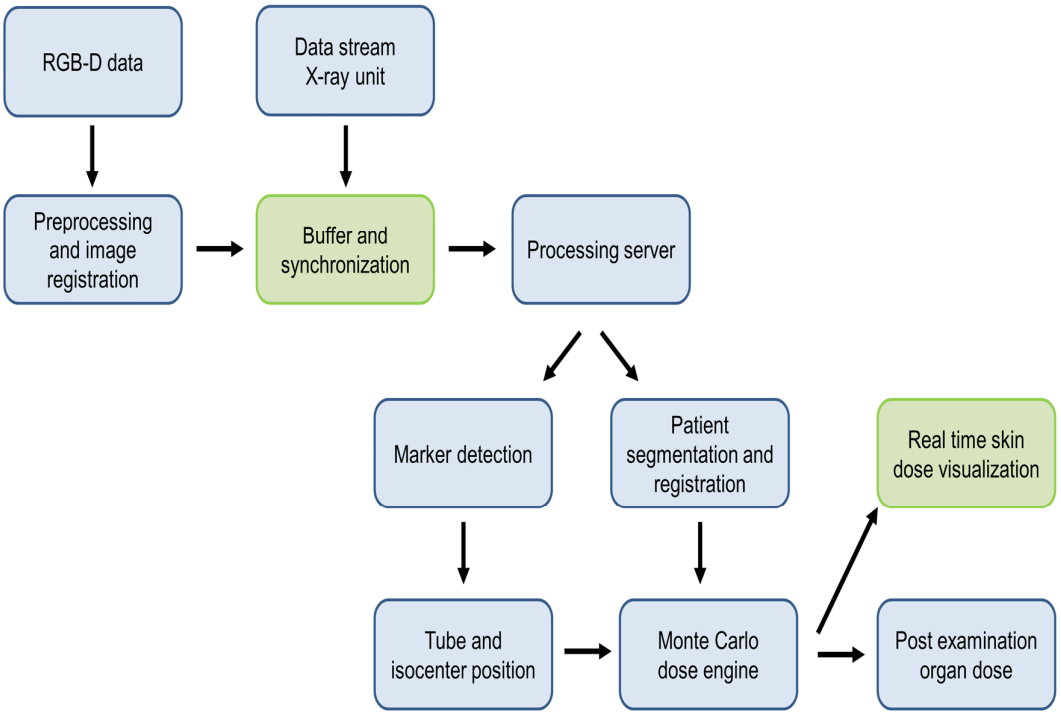

Figure 4 – Design suggestion online system

As this project was limited to a proof of concept it was not possible to directly interface with the X-ray unit. Nonetheless the architecture is designed with a real time feedback system in mind. As shown in Figure 4 the only modifications would be to integrate a data buffer to synchronize the data stream from the X-ray unit and the images before feeding it to the server where processing happens. To limit the required size of this buffer a sensitive sensor detecting radiation like a diode could be added to the single board computers of each camera to trigger the start of image acquisition.

The final design would only have near real time feedback for skin doses, visually presented to the physicians and physicists during the examination. This is possible through modifications in DOSXYZnrc to speed up skin dose calculation as this is of specific interest for interventional radiology. Organ dose calculations would be too time consuming to provide relevant and fast feedback.

3.2. Camera system

Microsoft Kinect v2 cameras were chosen for the camera system as they provide a reasonable resolution depth map (512x424) combined with a full HD color image (1920x1080) at 30 frames per second. The Kinect v2 uses a time of flight (TOF) principle as opposed to the structured light used by the Kinect v1 or Asus Xtion Pro. When using multiple cameras there is less interference and noise from other cameras using the TOF principle than when multiple structured light patterns are being projected and processed [47]. Another advantage of the TOF camera is the constant offset in distance that can easily be modelled compared to the distance dependent offset of the Kinect v1. The Kinect v2 camera has its own limitations such as: flying pixels, that occur near discontinuities of geometry; multipath interference, due to reflection of light from multiple surfaces often seen with concave objects; the depth value is influenced by color and the dependent on the camera temperature. Most of these limitations can be compensated by pre-processing the depth images [48], [49]. The Microsoft Kinect v2 also provides a skeleton tracking feature that detects the major joints of the human body and can do this for up to 6 persons. Although an interesting feature it was

not used to avoid dependency of one specific camera type, the system is designed to work with any camera or mixture of cameras that provide a color, infrared and depth image.

In the interventional radiology room three Microsoft Kinect v2 cameras are positioned with approximately 90 degrees rotation between each camera as shown in Figure 1. This configuration allows to cover most of the interventional radiology room and handle occlusion caused by staff. A fourth camera was considered unnecessary as this camera would be occluded by the C-arm most of the time. Each camera is connected to an Odroid XU4, a single board computer running Ubuntu 16.04 LTS, which in turn is connected to a switch with a Cat-6 ethernet cable and synchronized with the data server through the Network Time Protocol (NTP).

Each Odroid XU4 has a dedicated executable that streams and saves the image data from the Kinect to a Network File System (NFS) directory under the form of a binary file containing information such as a nanosecond precision timestamp, serial number taken from the Kinect, type of images stored, compression type and the image data: Color, depth and infrared. If calibration has been performed and intrinsic and extrinsic camera parameters are provided under the form of a YAML file the rectified color and registered depth image is calculated immediately instead of the raw depth, infrared and color. The processed images are larger to store than the raw data. Therefore, in the prototype it was chosen to not save the processed data.

The individual Kinects are all calibrated using Zhang's method with a custom-designed aluminum (Dibond®) panel with a 6 by 9 checkerboard pattern that has 90 mm by 90 mm squares. First the intrinsic camera parameters for the color camera and infrared camera were determined separately. Based on initial tests the Kinect cameras did not present any noticeable axes skew so only the following parameters were determined and used: focal length, principal point, radial distortion and tangential distortion. Each camera calibration step uses approximately 100 individually recorded images at different distances and angles to make sure every part of the field of view from the camera is taken in account for the model. After single camera calibration was performed the rotation and translation between the infrared and color camera is determined using a classic stereo camera calibration with the same checkerboard pattern. For the stereo calibration another 100 images are used but only in the field of view where both cameras can see the checkerboard pattern. The images recorded for calibration are then processed using MATLAB and OpenCV to determine all parameters.

An additional calibration step is performed after setting up the multi camera system to link all cameras to a predefined shared coordinate system. This is done by recording a stationary custom-made aluminum calibration cube that consists of several unique 45 mm by 45 mm checkerboard patterns. This calibration cube was precisely measured to know the exact geometric relation between all corners of the checkerboards. The origin of the shared coordinate system was assigned to one specific corner of the checkerboard at the top of the checkerboard.

3.2. Monte Carlo dose engine

Besides the camera system the second major component of the system is the Monte Carlo radiation transport code. EGSnrc was chosen as the Monte Carlo engine, more specifically the user code DOSXYZnrc. The choice was based on the in-house knowledge regarding DOSXYZnrc and its calculation speed in voxel dosimetry calculations compared to other codes such as MCNPX [50].

For this project several modifications were made to the source code of the Monte Carlo engine by Dr Blake Walters. A new source model was built and several methods to speed up calculation time for skin doses were introduced.

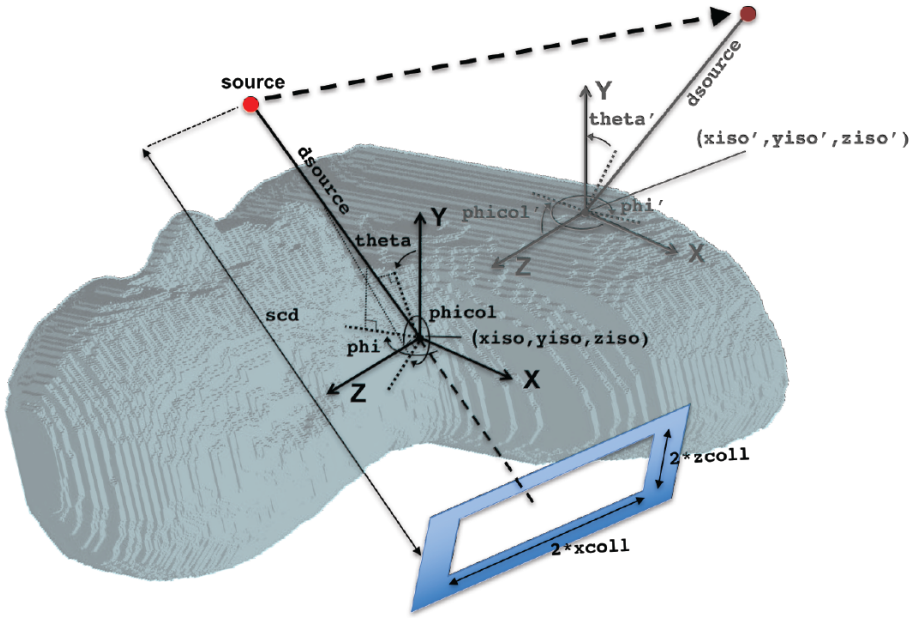

Figure 5 – Geometric relationships of the new source model, courtesy of Blake Walters

The alternative source type introduced is a dynamic point source represented by a spectrum. This point source can be blocked by a perfectly blocking mask to mimic the collimation of the beam, the collimation size is defined by a field size at a distance as shown in Figure 5. To allow modeling a moving tube this point source can also be assigned dynamic movement by giving several isocenter positions and rotation angles with a fixed collimation and source to isocenter distance. Each combination of isocenter coordinates and angles must be assigned a mu-index (μ) in ascending value with the first combination having an index 0 and the last 1. The simulated position for each history is then determined by a randomly generated number (r), between 0 and 1. The closest mu-index lower than the random number is chosen as the low position (μ_{i-1}) and the closest mu-index higher is chosen as the high position (μ_i). The exact position is than determined using equation (2).

$$P = P_{i-1} + \frac{P_i - P_{i-1}}{\mu_i - \mu_{i-1}} (r - \mu_{i-1}) \tag{2}$$

$$P_i = [x_i \quad y_i \quad z_i \quad \theta_{x,i} \quad \theta_{y,i} \quad \theta_{z,i}]$$

Providing real time skin dose information requires additional methods to improve the speed of calculation. The low photon energies used in radiology do not produce high energy electrons, therefore the kinetic energy released per unit mass (kerma) is equal to the dose for X-rays in the kV range. This allows dose calculations by scoring kerma, and electrons do not have to be tracked during the Monte Carlo simulation. First variance reduction techniques were evaluated, forcing photon interactions were compared to optimizing the number of photon splitting events, a more thorough

explanation is given in Appendix C. The optimal splitting count was different for skin dose calculations than for organ doses. Optimal splitting was preferred over forcing interactions. Further improving the efficiency of the skin dose calculation was done by allowing to specify a distance after which the particles and its descendants are no longer tracked. The particle or its descendant is rejected when it reaches a voxel of which the nearest surface voxel is further away than the specified distance. To avoid losing too much time transporting particles through air all phantoms are trimmed to have minimal air.

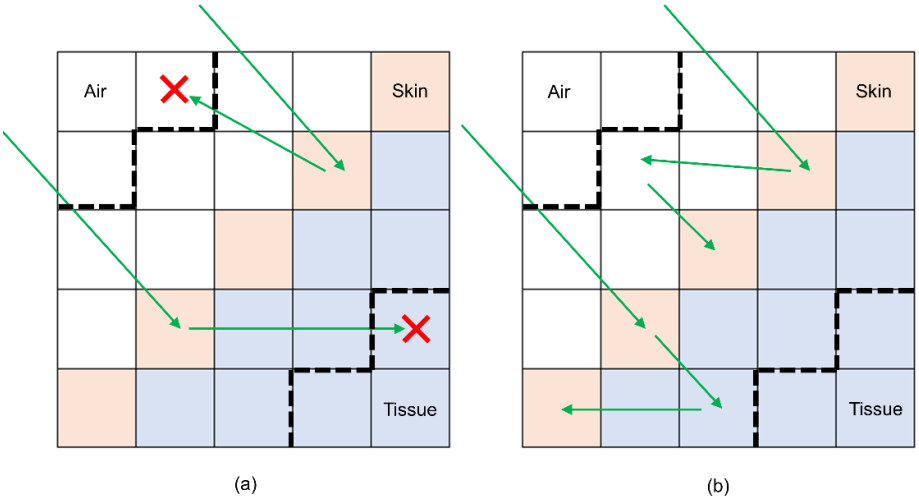

Figure 6 – Two-dimensional representation of the particle rejection in case of a rejection distance of 1 voxel. (a) shows particles travelling further than rejection distance and (b) shows particle backscatter inside the range

Figure 6 illustrates the method used to reject particles travelling too far from the skin after they have been transported through a surface voxel. The rejection distance is defined in voxel count so depending on the voxel size the distance should always be equal or slightly higher than the range in which backscatter is significant.

3.3. Obtaining a three-dimensional room model

The Microsoft Kinect v2 consists of two cameras integrated into one system, a color and infrared camera. As a result, there is no direct relation between raw depth and color image pixels. The images cannot simply be matched by correcting for the field of view, resizing and translating since they recorded the same scene from two different cameras with their own unique intrinsic camera matrix and they are not physically in the same location. Calibrating the cameras allowed to determine their intrinsic camera matrix as well as the translation and rotation between these two cameras.

Registration between the depth and color image is obtained through several steps: First the depth image is deformed to obtain the depth image as if it had captured the scene with an ideal pinhole camera with no lens distortion that has the same intrinsic camera matrix as the color camera. Second with the depth information all pixels are projected into a three-dimensional point cloud by combining this information with the intrinsic parameters of the color camera. Third this point cloud is now rigidly transformed with the translation and rotation determined by the stereo calibration, this point cloud now resembles how the scene was perceived if the infrared camera had been in the same physical position as the color camera. Step four involves projecting back this cloud of points using the color camera parameters and the pinhole model. After transformation some points might project to the same pixel resulting in an erroneous depth value, these errors are eliminated using the Z-buffer algorithm. Finally, the resulting image matches the color image, but the original color image is affected by lens distortion, so this must be corrected before a pixel to pixel relationship can be

attained. In Appendix A the entire process is explained in depth and a MATLAB implementation of the algorithm is provided.

After registration the color image can later be used to extract features and combined with the depth image to determine three-dimensional coordinate as the depth and camera parameters are known. For a single camera this coordinate could be used to determine the relationships between different objects in the room. However, the dose calculation system uses a multi-camera setup consisting of three Kinects and after individual calibration the origin of each camera's coordinate system is its optical center. Combining these three individual point clouds into a single three-dimensional model of the room requires an additional step. This was obtained using the calibration cube mentioned in Chapter 3.1. which has different checkerboard patterns and precisely known dimensions. One of the checkerboard coordinates is assigned as the origin of the coordinate system and all other checkerboard coordinates are generated using the measured relationships. This results in a ground truth, the transformation from the Kinects coordinate system to the ground truth is then determined with Horn's method for absolute orientation using unit quaternions [51]. The MATLAB implementation of this algorithm can be seen in Appendix B.

3.4. Source model

The radiation source model is one of the key components of the Monte Carlo input file, the radiation source is modelled as a point source with an X-ray spectrum in the adapted version of DOSXYZnrc. Spectra for a series of commonly used tube parameters were calculated using SpekCalc, creating a library of spectra for different parameters. SpekCalc is a software program designed to calculate X-ray spectra in the range of 40 to 300 kVp from tungsten anode X-ray tubes [52]. Each spectrum file was converted to a format supported by EGSnrc. Before writing the Monte Carlo input the structured report data is filtered for tube potential and filtration after which the most appropriate spectrum is chosen.

While some basic geometric information regarding the tube position is available in the RDSR the position needs to be translated to the coordinate system perceived by camera system. ArUco markers were chosen to perform marker-based tracking of the tube. ArUco is a library for the auto generation of highly reliable fiducial markers often used in augmented reality [53]. These markers were used for reliable and fast tracking of the X-ray tube motion. Eight unique markers were places on the X-ray tube, three on the detector, three on the tube, two on the suspension. The exact position of the source and isocenter in relation to the marker positions was determined using radiation measurements using an ionization chamber (Farmer, model…) while recording with the camera system.

Once the relationship between source, isocenter and markers has been established for calibration positions, these absolute positions are stored. During measurement when an event occurs the current marker positions are determined, and the structured report is parsed for data regarding changes in the relative positions of detector and source. If the relative positions are changed the calibration marker positions are changed accordingly, moving the detector markers as determined by previous calibration.

Once the calibration markers match the current situation of the C-arm the calibration positions are compared to the measured marker positions. Then the relative transformation is determined using Horn's method which returns a rotation matrix and translation vector. This calculated translation vector is used to translate the source and isocenter. The rotation matrix is decomposed into a rotation around the x,y and z-axis. The rotation angles indicate the rotation around the isocenter in the predefined camera coordinate system.

3.5. Patient model

A family of XCAT phantoms for different body types of each gender were generated to represent the patient geometry. XCAT are 3D humanoid phantoms with 4D capabilities [54].

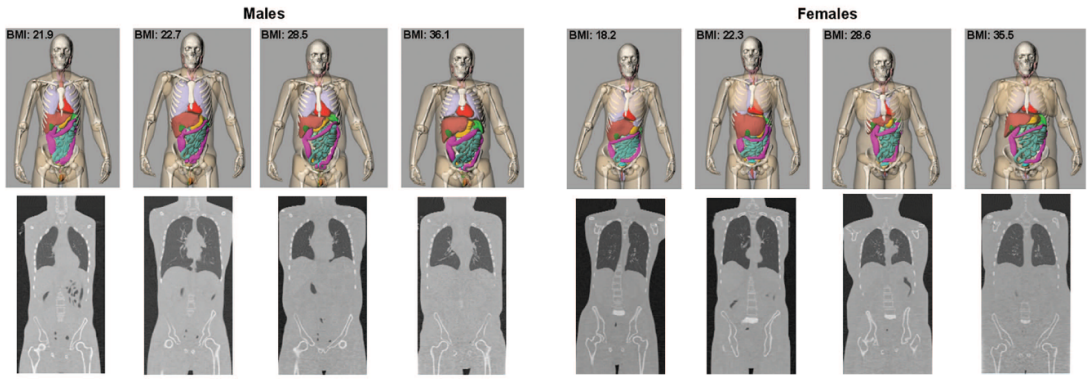

Figure 7 – Family of XCAT phantoms [55, p. 9]

The activity mode is used to produce an XCAT with different activity levels for each organ, these activities are used as labels for the tissue type when converting to an EGSPhant format. EGSPhant format is a text or binary file containing a list of tissues and their corresponding label, voxel boundaries and two three-dimensional matrices. The first matrix contains the tissue type label for each voxel and the second the density of that voxel. Conversion from an activity XCAT to EGSPhant was done using VOXSI [56]. An average skin thickness of 3 mm was selected and used for all XCAT models [57]. The user can enter the patient's gender, body mass index (BMI) and desired voxel size. After confirmation the structured report is parsed to extract the examined body region and then the most appropriate EGSPhant will be loaded into memory.

The EGSPhant is then filtered for skin voxels. The skin thickness of 3 mm results in some skin voxels inaccurately being labeled as adipose when creating a voxelized geometry. To achieve a correct three- dimensional logical array of skin voxels, the medium skin was defined as each skin or adipose voxel that has a neighboring voxel containing air. Al three-dimensional coordinates of the skin voxels are stored in a coordinate vector.

Segmentation of the examined body region is performed on each rectified color image. After this a mask is applied to the depth map, all pixels outside of the segmented region are set to 0 depth. The non-zero depth pixels are transformed to [x,y,z] coordinates and set to a shared coordinate system using the methods described in Chapter 3.3.

The coordinate vectors of both the EGSPhant and the segmented patient can now be further processed as point clouds. Initial coarse registration is done by using the center of mass of both point clouds and applying a translation to the segmentation point cloud to align both centers. The center of mass was determined for the surface points detected by the camera and for the skin voxels of the virtual phantom.

After the initial coarse registration another fine registration transformation matrix was determined using a fast KD-tree implementation of the iterative closest point algorithm [58], [59]. The determined transformation matrix and initial translation are saved and applied to the isocenter and source position to correctly represent the position of the beam in the same coordinate system as the phantom. The choice to apply the transformation to the beam instead of the phantom was done for performance reasons.

3.6. Result visualization

Reporting the calculated dose is done both visually and numerically. To visualize the calculated skin dose a mesh of the patient geometry is shown. Each mesh was created from the EGSPhant voxel geometry to preserve the coordinate system. A more visually pleasing mesh was attained by smoothing the geometry and reducing the number of faces in MATLAB. Meshes were then exported in Stereolithography (STL) format and loaded into Blender (Blender Foundation), an open source 3D modeling software, where the vertices subdivision was altered to allow for a more even distribution of vertices. The mesh was exported again as STL and compared to the original mesh to make sure no significant changes in coordinates were introduces.

Skin dose is represented by adding a colormap on top of the mesh. The colors are based on user defined thresholds, where blue is the visibility threshold, orange warning and dark red exceeds the maximum threshold. Alternatively, the option exists to show the location and value of the peak skin dose, this peak skin dose is the 0.5 cm^2 area on the skin with the highest dose.

Organ doses visualization is done numerically in the form of a table with the mean, median and maximum dose in Gray (Gy). The effective dose in Sievert (Sv) is also calculated using the calculated organ doses and weighting factors described in ICRP 103 [60]. Dose gradients inside the organs can be further investigated by a color-based dose map overlaid on top of the density image of the phantom. As the exact organ positions for individual patients are not known these values and gradients serve merely as an approximation for selected XCAT standard phantom.

3.7. Initial verification

Full validation of the system must be performed in a radiology room in service mode to have complete control over the tube parameters. This validation will be done using XR-RV3 or EBT3 film and these tests are scheduled but will not be finished before the end of this thesis. A preliminary test was performed using the standard clinical settings of the X-ray machine. Due to the clinical settings of the machine the tube parameters are chosen automatically, controlled by an Automatic Exposure Control (AEC) device. This AEC adjusts parameters based on the Incident Air Kerma Rate (IAKR) to avoid fluctuations in image brightness and noise. This holds the implication that the tube parameters will differ between each measurement even if the geometric relations remain the same, small fluctuations are still possible [61].

Film measurements were performed using XR-RV3 film (lot# 08311801) fixed on top of an anthropomorphic phantom: the Alderson RANDO phantom (Radiology support devices, Inc.). Converting the Monte Carlo calculated values to absolute dose requires a conversion factor for each kV of the X-ray tube, to transform the value from dose per photon ($\frac{D}{\gamma}$) to a more usable format that can be determined from tube parameters ($\frac{D}{mAs}$). This conversion factor is determined using in air measurement with a Farmer ionization chamber (2571, Thermo Electron Corporation) according to the AAPM TG-61 protocol [62]. This value is then correlated to a film calculation with a simple geometry and known geometric relations to the source. This simple geometry was a polymethyl methacrylate (PMMA) cube positioned with the surface in the isocenter and the tube above the film at a zero-degree angle, vertically pointing down.

Film calibration was done by irradiating a total of 33 films with a 40 mm^2 square field on a calibrated small animal irradiator (X-Rad 225Cx; Precision X-ray Inc., North Brandford, CT) using a 120 kV X-ray source with 2 mm Al filtration to closely resemble the diagnostic X-ray spectrum. The film irradiation at each set dose level was repeated 3 times to reduce the impact of noise and detect potentially erroneous exposures. The dose levels used in the calibration are shown in Table 2. All film processing was done using an in-house MATLAB routine. The reflective density of the red channel was used to determine the calibration curve [63], [64].

Table 2 – Irradiation settings for the film calibration, film pieces placed at surface of solid water and a calibrated dose rate factor of 1.452 mGy/mAs for the 120 kV and 2 mm Al filtration X-Rad 225Cx irradiator

Tube current (mA)	Beam on time (s)	Dose (mGy)
0.0	0	0.0
0.1	11	1.6
0.5	6	4.4
0.5	9	6.5
0.5	11	8.0
0.5	33	24.0
1	28	40.7
4	14	81.3
5.6	50	406.6
5.6	99	805.0
5.6	496	4,033.1

4. Results

The dose calculation system works on a per event basis, this means it will perform calculations only on the images that get captured around the time an irradiation takes place. Synchronization in this prototype is done using the timestamps of the RDSR and the individual cameras. To facilitate the use of this system a basic graphical user interface (GUI) was created to communicate with the calculation engine.

To minimize computation time some simplifications regarding synchronization are made for the different irradiation types. For static acquisitions only the frames corresponding to the starting time are processed. A rotational acquisition uses the frames at the start of the rotation and end of the rotation and interpolates all intermediate positions assuming a constant movement speed of the X-ray tube. During a fluoroscopy event the patient is assumed to remain immobile and markers on each frame are detected but only the tube movements with an angle larger than 3° or translation of more than 2 cm are processed further as new set points for the Monte Carlo model. These setpoints are then assigned a relative mu-index based on the time spent in each position. The minimal angle and translation thresholds were chosen to handle the noise that results from the uncertainty on the depth map obtained from the Kinect and the effect of rejecting these positions should be analysed at a later stage.

4.1. System flow

This section will show the entire trajectory of a single static event to illustrate how the system works. Due to not being able to fully integrate the prototype into a clinical system the process is interrupted by storing the data to be able to load it later instead of having a single flow.

The first part of the system consists of streaming, after triggering the program the three Odroids will run a dedicated executable that calls the libfreenect2 libraries to retrieve the information from the Kinect v2 camera it is connected to. Upon start-up the serial number from the Kinect is retrieved and a separate directory on a server with a shared NFS folder is created which is labelled after the starting time and serial number. After each combination of frames (Color, depth and infrared) is retrieved a nanosecond resolution timestamp is generated. The frames are then written to the separate directory as a binary file with the serial number, timestamp and additional information about the frames in its header. The rate at which the data is written is limited to 5 frames per second. This was done to conserve disk space in prolonged recordings, a problem that should not occur in a fully integrated system.

For the processing part the three directories created on the NFS directory, one by each camera, can be selected alongside their corresponding RDSR. After this a timeline is generated sorting all radiation events and matching them up with the corresponding image data based on the timestamp read from the header.

From now on the image data and its processing steps will be shown for an event with a RANDO phantom and the C-arm with the X-ray tube at a 30-degree angle above the table. The first step involves loading the images from the camera in to memory, as this is a static event there are a total of 9 images: 3 color, 3 depth, and 3 infrared images, for rotational events it is twice the amount and with fluoroscopy even more depending on the duration. The initial images are shown in Figure 8, these images are stored into a separate data class for each serial number.

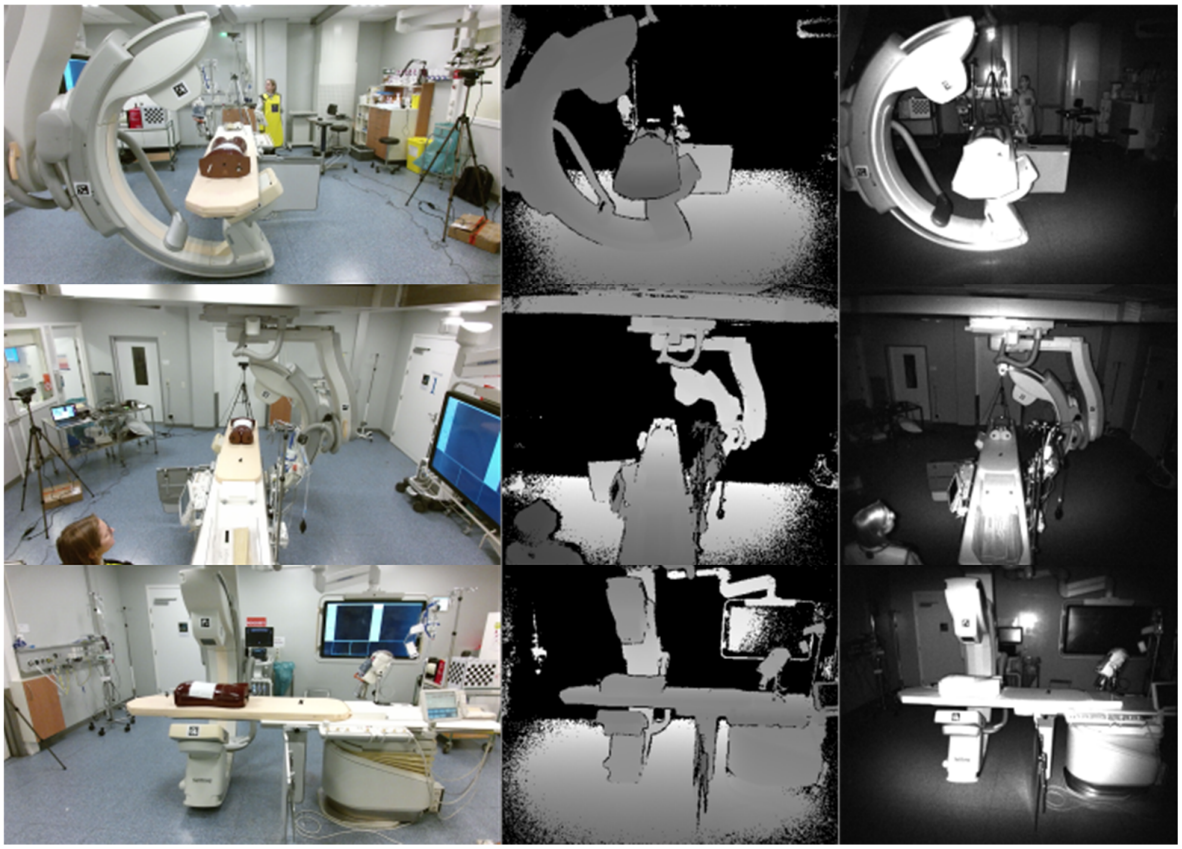

Figure 8 – Raw image data retrieved from the three Kinect v2 cameras, each row shows the data from a different camera. From left to right it shows the color, depth and infrared images

After importing the images, the data classes are assigned their corresponding calibration and transformation data according to their serial number. The calibration and transformation data are read from a YAML file as the calibration occurs before processing. First the calibration data is used to perform the rectification of the color image to remove lens distortion and remapping and registration of the depth image to correctly match the color pixels. The top row of Figure 9 shows the rectified color and registered depth image, the thin black curved lines on the edges of the rectified color image show that the color image had a small amount of barrel distortion. The bottom left shows the original depth and color image overlaid, illustrating the lack of relationship between pixel locations. Finally, the bottom right shows the remapped and registered depth superimposed on top of the color image, illustrating the correct relationship between the pixels of the two images essentially giving each color pixel depth information.

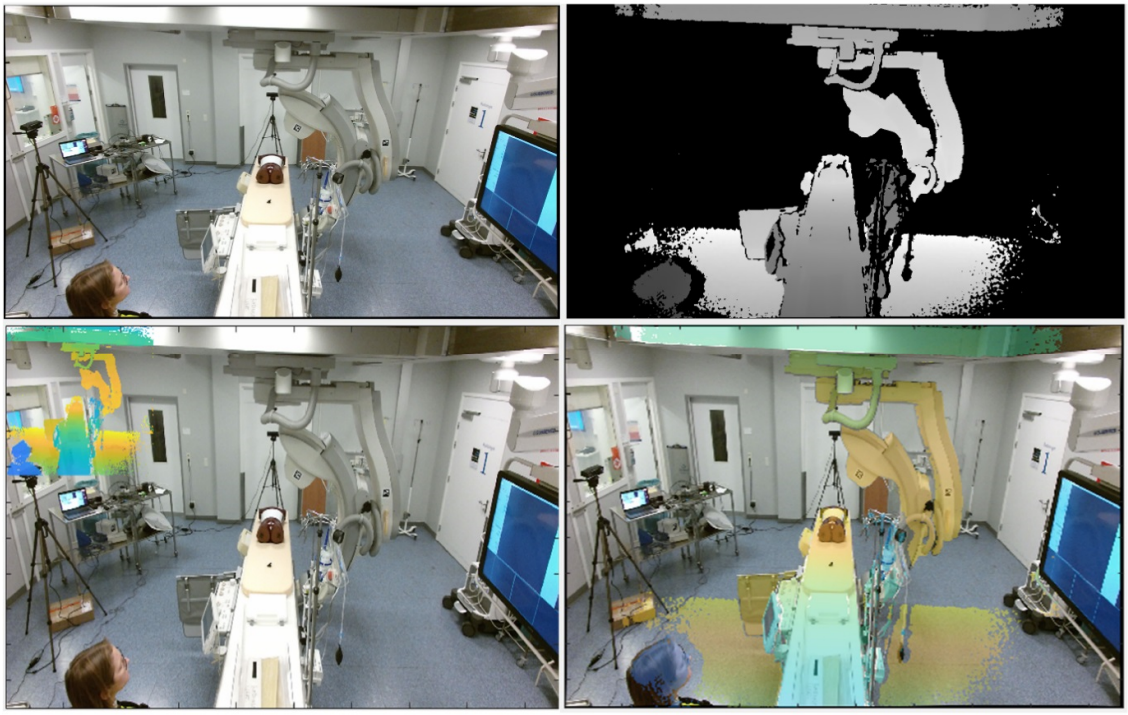

Figure 9 – The result of color and depth registration with the top row being the individual images after rectification and registration and the bottom row the non-registered superimposed image on the left and on the right the registered superimposed image

The registration of depth and color allows the creation of a point cloud from the view point of the color cameras. These point clouds are individually generated and then registered using the transformation matrices that were loaded into each image class from the transformation calibration. This results in a three-dimensional point cloud aligned to a reference coordinate system.

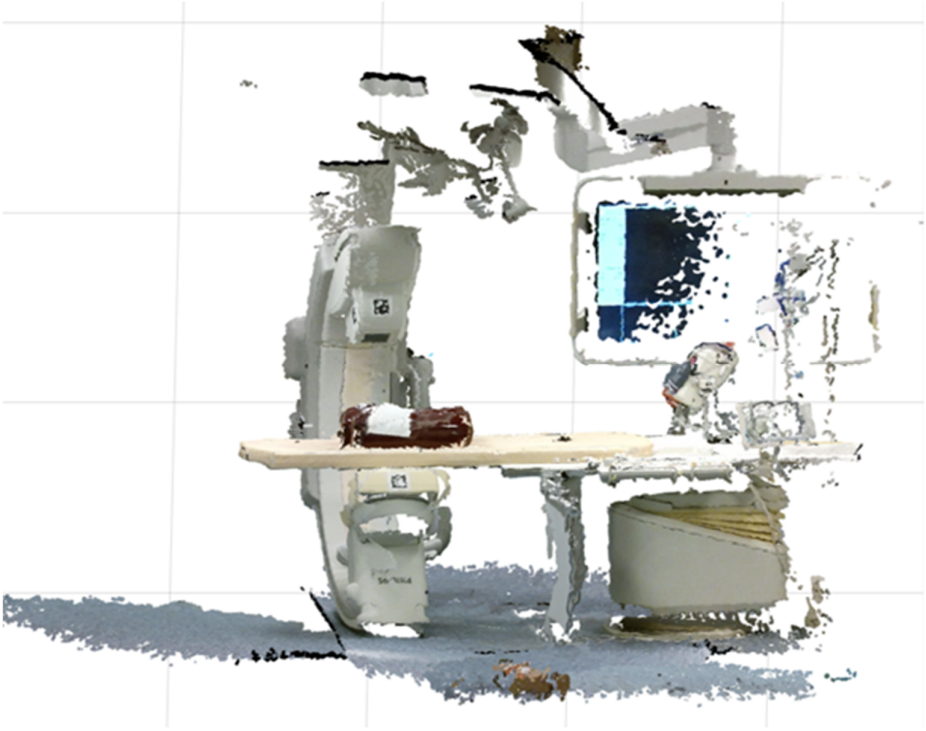

Figure 10 – Merged point cloud from three cameras

Determining the source and isocenter position is done by detecting the markers on the C-arm. Each marker has a unique pattern and is given an identifier corresponding to its location on the device. The marker positions are then compared to a calibrated position and their translation and rotation is determined. The results of the markers are compared to information about the tube position in the structured report to make sure the determined X-ray tube displacement and rotation are correct, providing a measure of redundancy and robustness. The isocenter should only be affected by the translation of tube. To verify the correct position of the isocenter it is recalculated based on the source to detector and source to isocenter distances found in the RDSR and the marker positions, this is then compared to the calibrated isocenter position based on the center of rotation to which the translation has been applied. The collimation of the source is also derived from the structured report and stored together with the source position.

On the individual rectified color images the phantom is segmented from the background image using image features specific to the RANDO phantom. From the segmented region a logical mask indicating which pixels belong to the RANDO phantom is created. This logical mask is used to eliminate all non-phantom pixels from the depth map. Which allows the creation of a point cloud containing only points belonging to the phantom. In Figure 11 the phantom segmentation is shown. The left image shows a missing region, this is the result of the radiochromic film which was not detected as part of the phantom for the camera system. The right image shows that an incomplete surface has no negative impact on the registration with between the patient or physical phantom (red) and the mathematical phantom (blue).

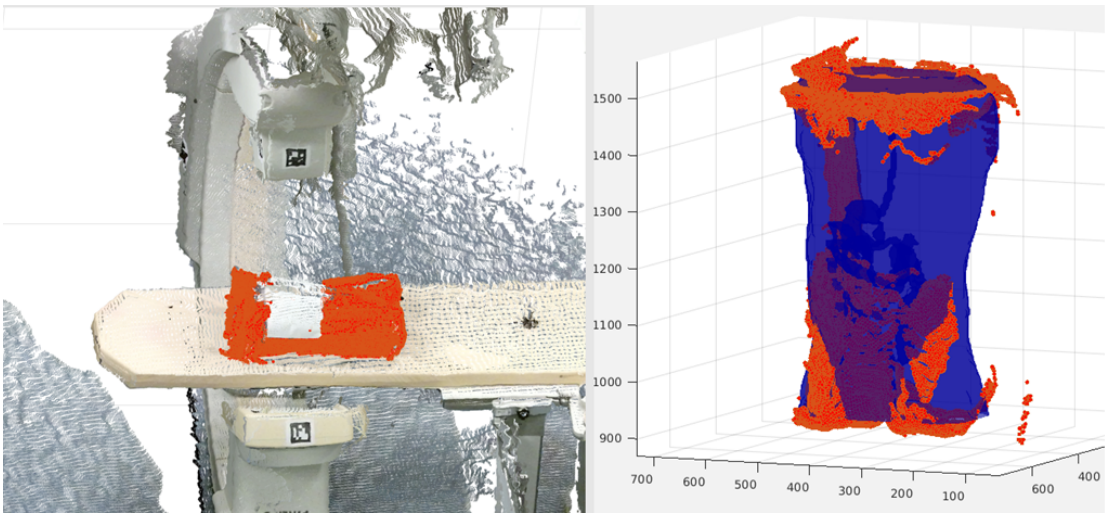

Figure 11 – Point cloud of segmented phantom displayed as red points. The left image shows the segmented patient in the original point cloud while the right shows how it was registered to the virtual phantom and shown on a 3D grid with the units set to millimeter

This point cloud of the patient is then matched to a point cloud created from the skin voxels of the virtual phantom using the iterative closest point algorithm. The virtual phantom model was chosen based on body mass index (BMI) and gender. The transformation matrix to convert the patient's position to the virtual phantom's is then applied to all other coordinates including the source, detector and isocenter positions derived from the markers and used to generate an input geometry for DOSXYZnrc. Figure 12 shows the visualization of this geometry.

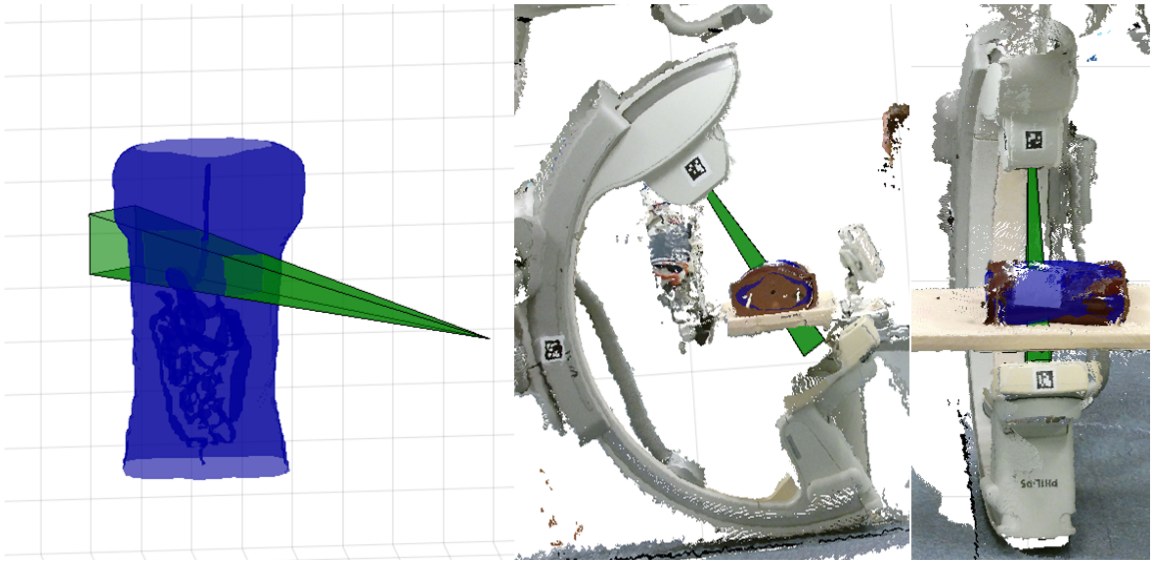

Figure 12 – Visualization of the Monte Carlo input geometry including the source position and beam collimation. The left image shows the pure Monte Carlo input geometry while the other images illustrate the correct registration and beam position by showing the geometry inside the captured point cloud

When the geometry is ready information about the beam such as the tube potential and filtration is extracted from structured report and the best matching spectrum file is chosen. This specific event had a tube potential of 80 kV with 1 mm aluminum filtration and 0.1 mm copper filtration. The spectrum shown in Figure 13 was selected as the source for the Monte Carlo simulation.

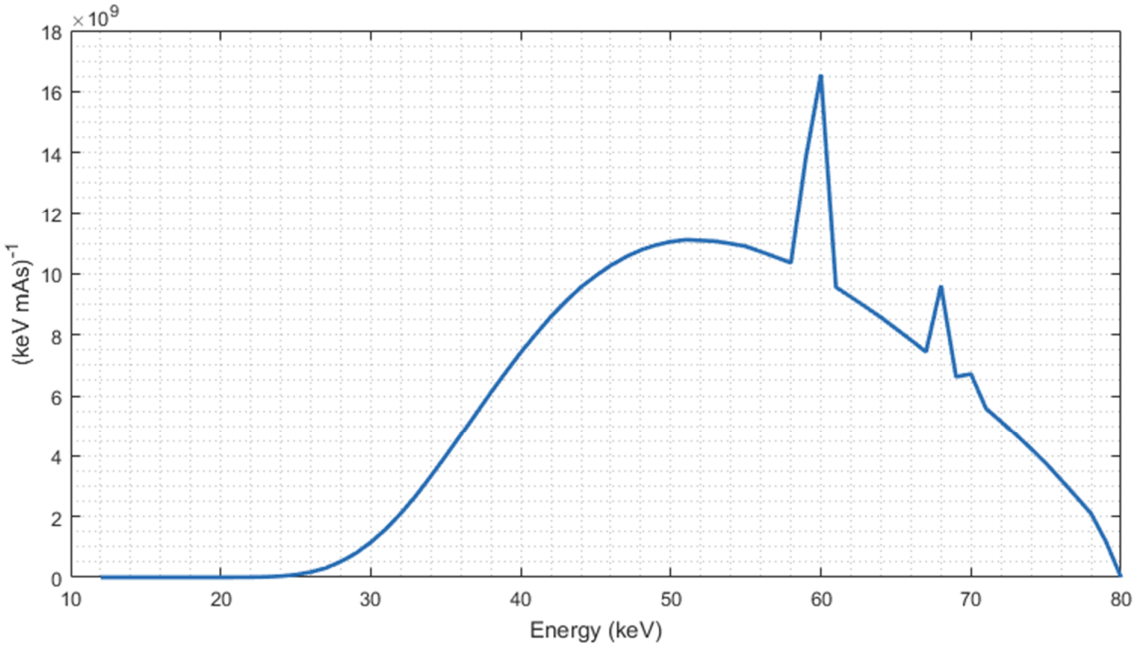

Figure 13 – Photon spectrum for 80 kV with 1 mm Al and 0.1 mm Cu filtration

Now the input file for the simulation can be generated using: the isocenter position, source distance, tube angles and spectrum from the beam; the target geometry and material data associated with the phantom. The number of histories required for the simulation is based on the field size of the beam. When the input file was generated the simulation is initialized by feeding the file to DOSXYZnrc. When the simulation is finished the .3ddose file containing the dose distributions in all voxels is imported into the program and used for dose analysis. These dose distributions are combined with a

calibration factor that transforms this dose per photon to a dose per mAs and together with the exposure time and tube current from the RDSR these distributions are converted to an absolute dose.

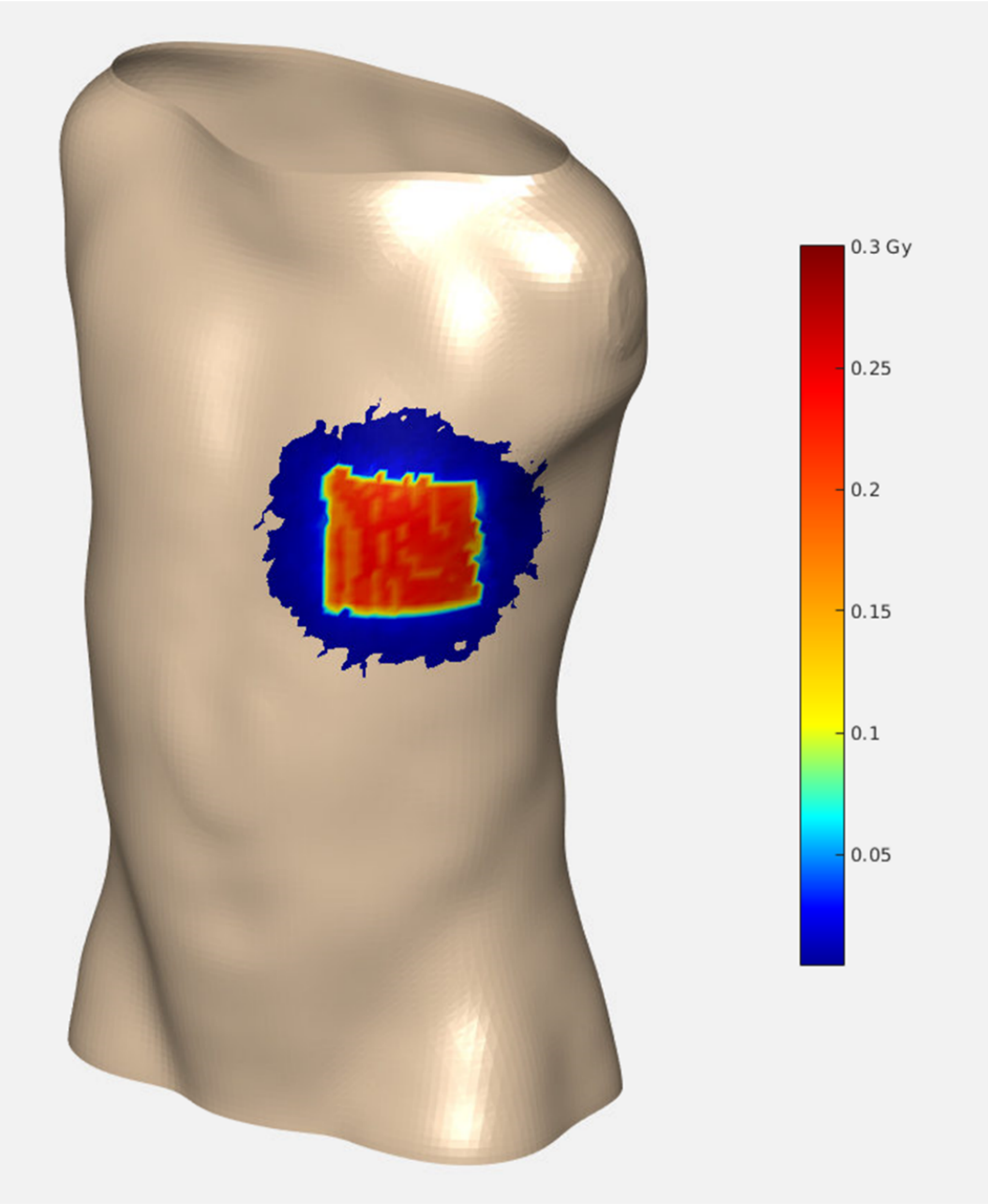

Figure 14 – Skin dose visualization

The default skin dose visualization is shown in Figure 14, a colormap based on user defined thresholds mapped to a mesh of the virtual phantom. Dose distributions inside the organs of this phantom can also be inspected by evaluating the mean, median or maximum dose in different tissue types as shown in Figure 15 or by looking at slices of the density image of the phantom with the dose values overlaid in a color map illustrated in Figure 16.

	Organ / Tissue	Dose (mGy)
1	**Gonads**	**0.00**
2	**Red Bone Marrow**	**0.60**
3	**Colon**	**0.59**
4	**Lung**	**8.45**
5	**Stomach**	**6.50**
6	**Breast**	**1.70**
7	**Bladder**	**0.00**
8	**Liver**	**17.02**
9	**Oesophagus**	**4.98**
10	**Thyroid**	**0.00**
11	**Skin**	**1.15**
12	**Bone**	**5.28**
13	**Salivary Glands**	**5.08**
14	**Brain**	**0.00**
15	**Remainder**	**1.47**

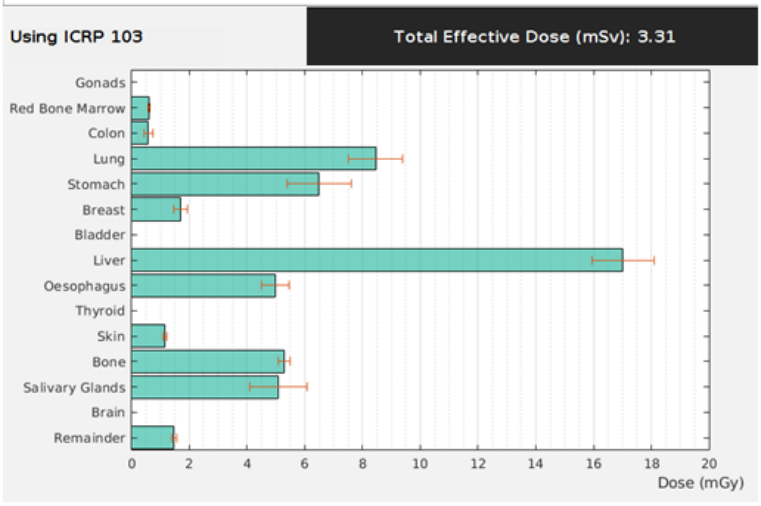

Figure 15 – Organ dose information, numeric evaluation

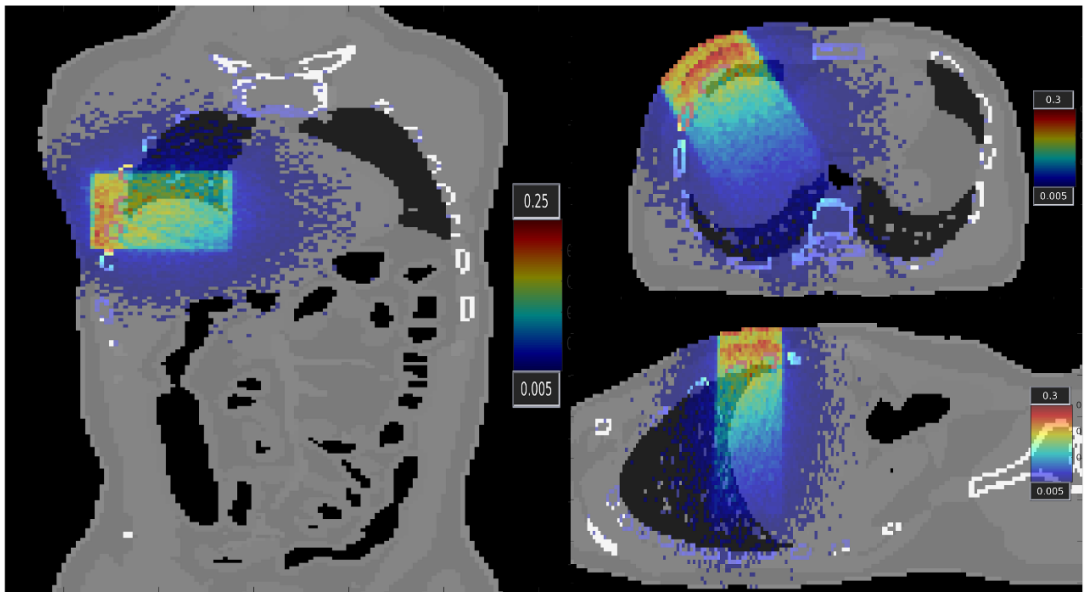

Figure 16 – Visual evaluation of dose gradients, a different color map scaling was used for the coronal image to attain the same contrast

This process is fully automated, no user interaction is required after initial calibration has been performed other than starting the image stream, selecting image directory, its calibration data and the corresponding structured report. This minimal manual intervention can easily be eliminated in an integrated system.

4.2. Graphical user interface

The graphical user interface was designed with two goals: to facilitate the use of the program by hiding the implementation details to start the stream and select the data behind some simple buttons; and to provide easy visualization of the calculated doses. To facilitate the calibration required and allow for evaluation and customization of the process at each step of the calculation process several additional modules were added to the GUI. Each of these modules will be discussed shortly. The only part not integrated into the GUI is the single camera calibration using Zhang's method, this was done using a custom MATLAB routine using the OpenCV library.

The streaming module as seen in Figure 17 is a user-friendly control panel that interfaces with the Odroids connected to the Kinects. It triggers and stops recording by calling underlying Bourne Again Shell (Bash) scripts that communicate through Secure Shell (SSH). Besides triggering the camera system, it also provides feedback about the recording, starting time and end time.

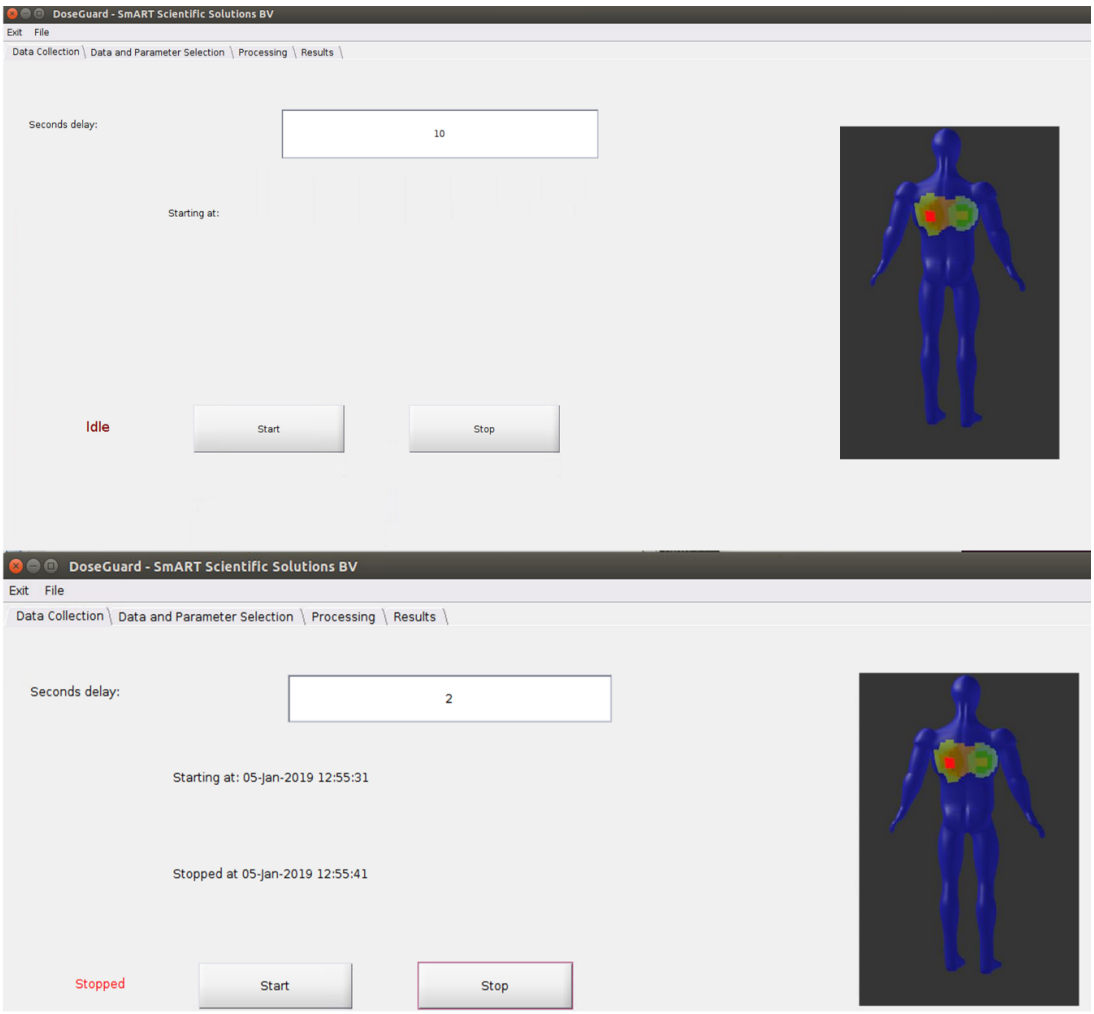

Figure 17 – Screenshot of the streaming module before and after a recorded stream

Importing recorded image data, calibration data and the corresponding RDSR is done in the data selection module shown in Figure 18. Here the directory containing the recorded image data is selected together with the corresponding directory containing the calibration data for that serial number, the data is quickly parsed to check if the serial numbers of the data from both directories match. The path to the RDSR can be entered as well as the possible time offset between the image data and the RDSR, in case the RDSR uses a different time zone than the images or in case the DICOM

server was synchronized to a time server with an offset compared to the NTP server used on the processing and recording server. This time offset is added to each event contained in the report.

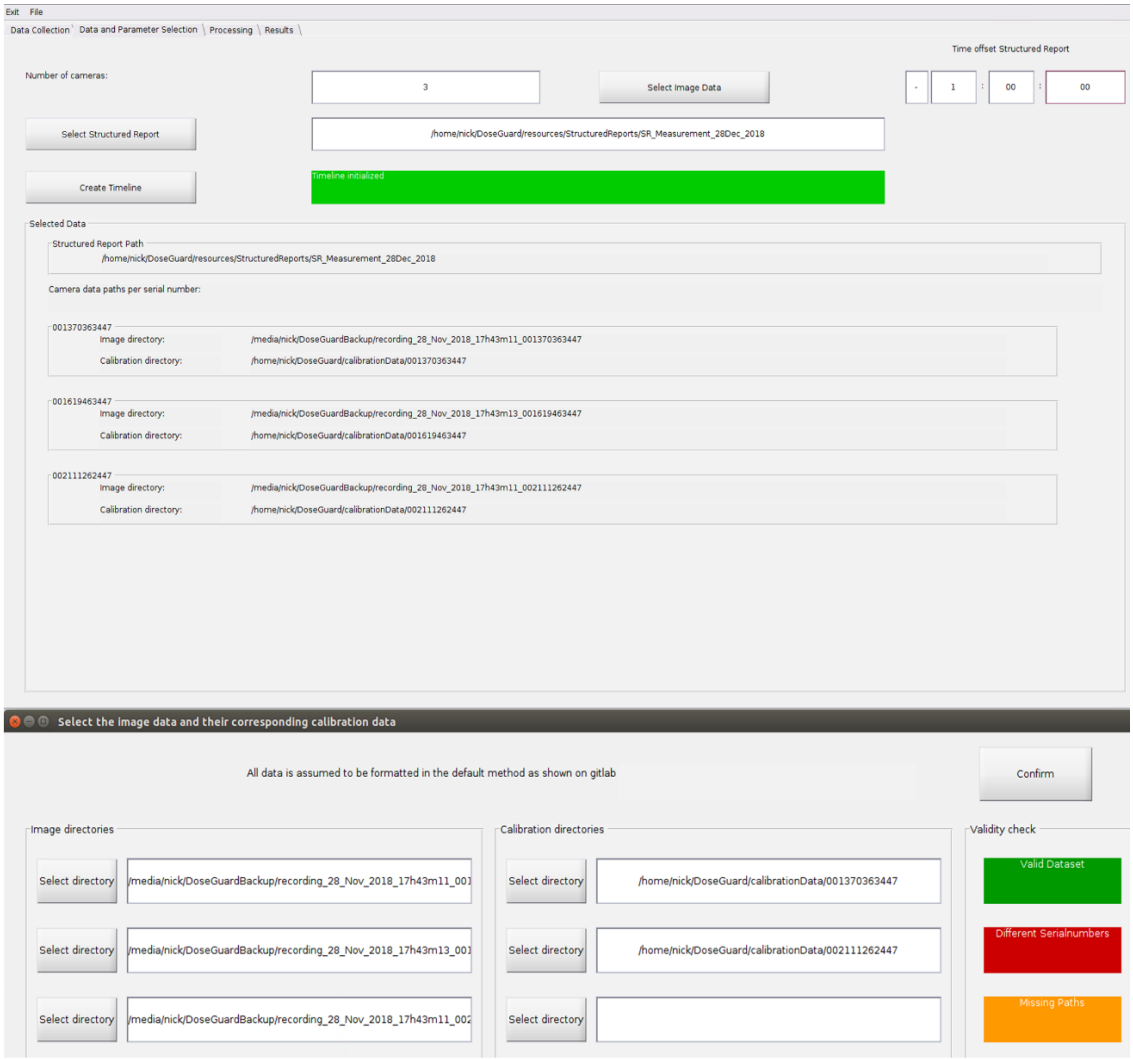

Figure 18 – Data selection module. Top image shows the normal layout of the selection module and the bottom part the popup to select image data

The processing module contains several separate submodules, the first four modules of which some are shown in figure 19 are used to allow for closer inspection of the image and RDSR data at different stages of processing: a basic viewer for visualization of the raw unprocessed data; a multi camera viewer for evaluation of the synchronization and rectification of the images; the point cloud viewer allows evaluation of the registered point cloud from all three cameras, this step however requires loading in data from the additional calibration module; finally the event viewer shows the extracted information from the RDSR for each recorded event.

Figure 19 – Two of the viewer modules, the point cloud viewer to inspect registration at the top and multi camera viewer to inspect the synchronization at the bottom, this is an optional feature for the advanced user

The two last submodules shown in figure 20 and 21 are required to load and perform additional calibrations. The additional calibrations module is used to get the transformation to the reference coordinate system for each camera. Two options are available, loading the transformation matrices from a YAML file or determining the transformation by loading a series of images with the calibration cube, which in turn can be exported in YAML format. This module is also used to find an initial estimate of the isocenter in relation to the source and detector positions. Finding the isocenter estimation is done by using a least squares approach in finding the center of rotation by detecting the marker positions on the C-arm at different known angles. Finally, the source calibration module is used to set the source and detector center positions in relation to the markers for one single calibrated position.

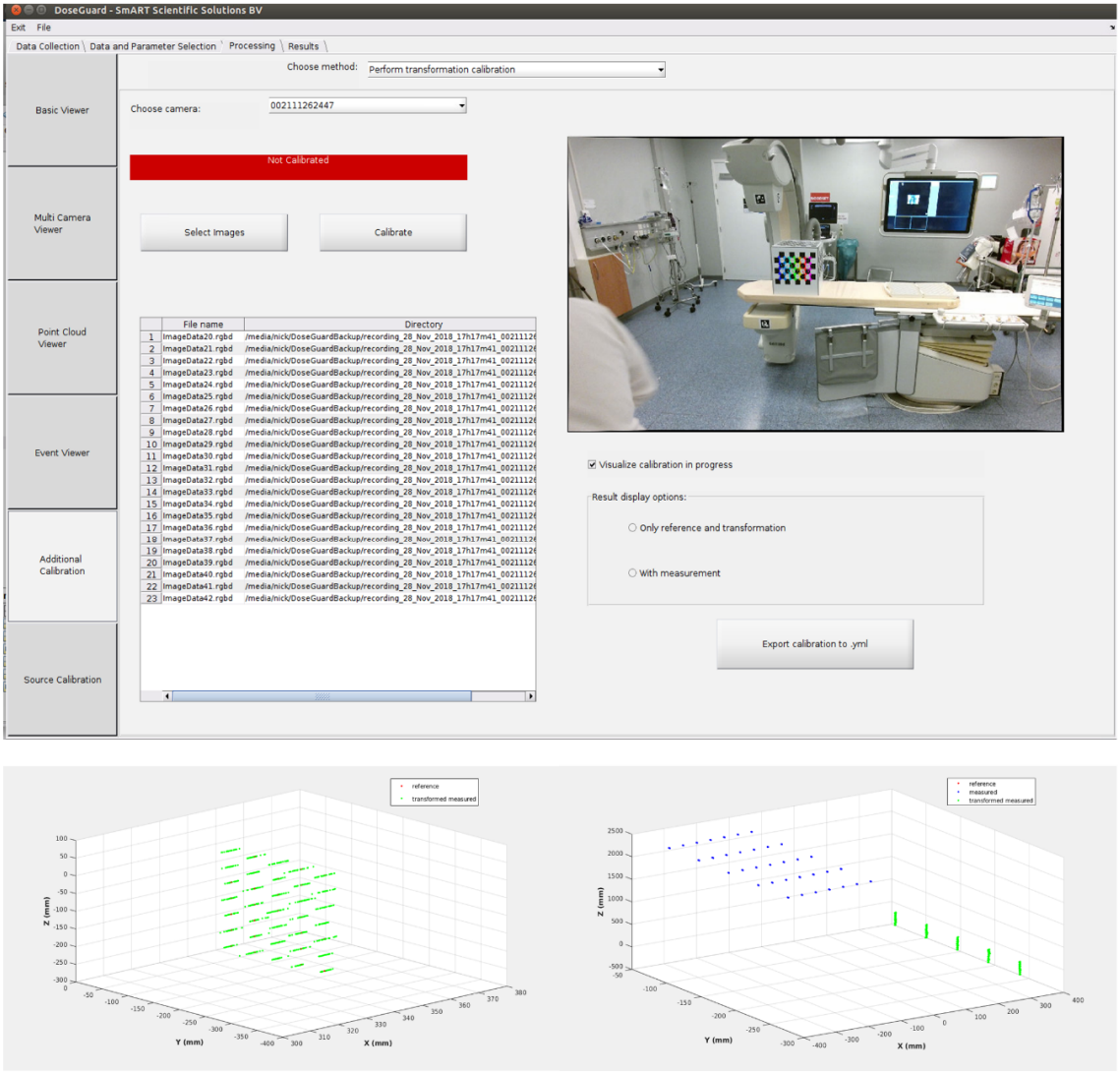

Figure 20 – Calibration sub module that calculates the transformation between the camera coordinate system and the shared reference coordinate system. The bottom images show the results display options with measured coordinates on the left image and only reference and transformed on the right side

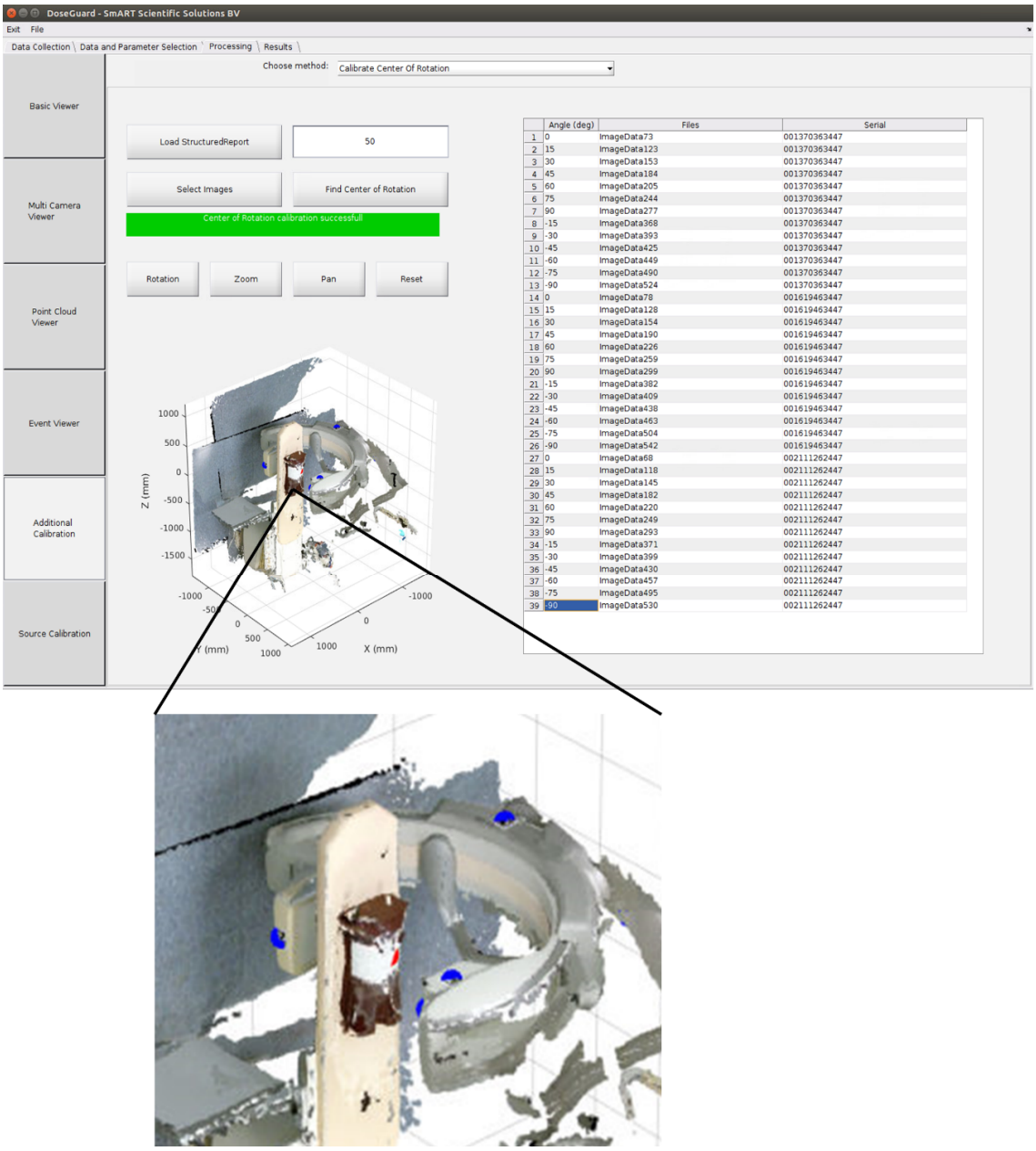

Figure 21 – Center of rotation calibration submodule, this module is used to get an estimate of the isocenter relation to the markers (blue) by finding the center of rotation (red)

The last main module is the result module, this module is used to initialize the calculation and analyse the results. It consists of three separate submodules calculation, skin dose and organ dose.

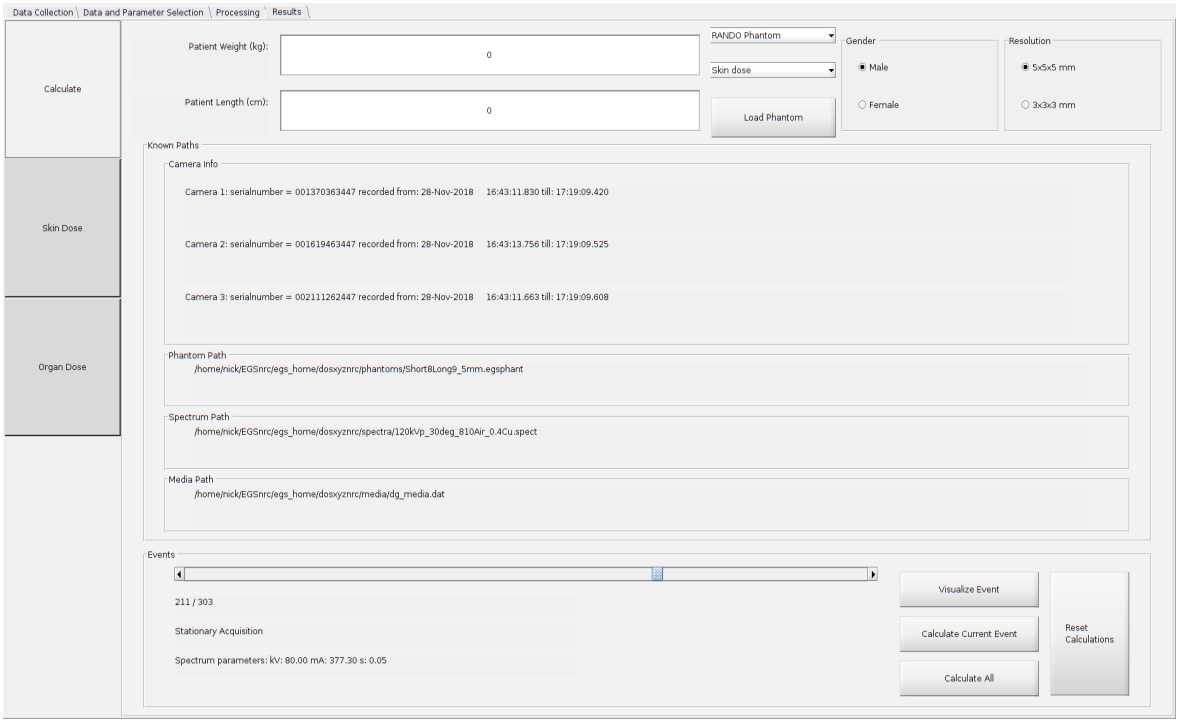

Figure 22 – Calculation module providing feedback about phantoms, spectra and media for user input. There are also options to calculate single events and visualize the Monte Carlo input of a specific event

The calculation submodule allows the user to input parameters required for an accurate calculation such as the patient length, weight and gender, the phantom resolution and the patient type: a physical phantom (for dosimetric measurements) or patient (clinical examination). The length and weight are used to calculate the BMI and together with the gender, patient type and requested resolution the most suitable phantom can be loaded. The patient type is important to select the method used to segment the patient and register its coordinates with the mathematical phantom. Each event can be calculated separately or a calculation of all events at once can be triggered. Besides triggering calculation an option is available to show the Monte Carlo input overlaid on the registered point cloud from the camera system. The visualization method can be used to determine possible sources of errors during the prototyping phase of the system. Additionally, the visualization allows verifying if the generated input is valid as seen in Figure 23.

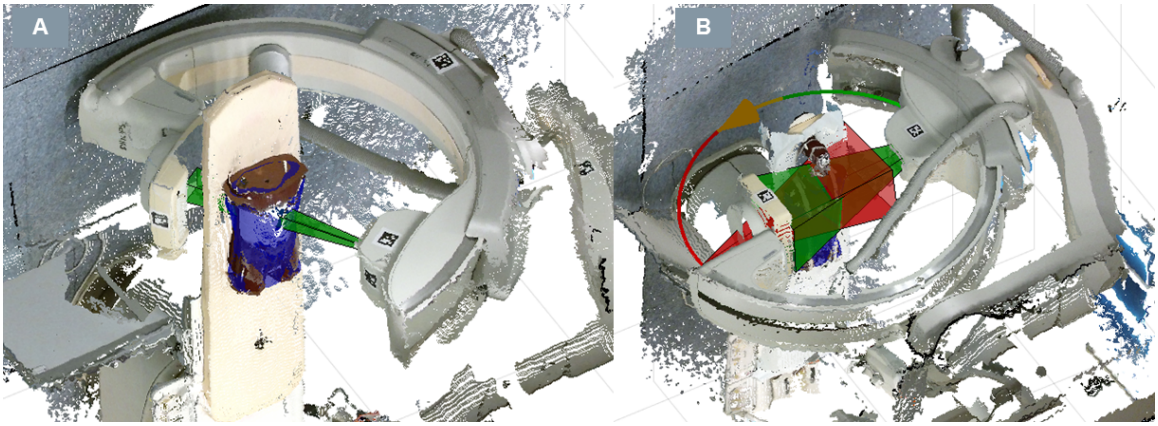

Figure 23 – Visualization of Monte Carlo input inside the point cloud generated from the camera system. A) shows a small 8 cm x 10 cm field at the detector with the tube at a 30-degree angle. B) shows a rotational acquisition with a 34 cm x 26 cm field, the green beam is the starting position and the red the end position. The point cloud is shown for the starting position

The skin dose module is a representation of what the physician would see inside the interventional room in case the system gets integrated and real-time feedback is provided. A mesh generated from the phantom used in the simulations is shown. On top of this mesh a color based dose map is projected, the color thresholds can be pre-set to change with a certain dose level as shown in Figure 24. The prototype is used offline and gives an additional option to show the location and value of the PSD as this is a commonly used value in evaluating risk in interventional radiology, this PSD could also be reported real time. Warnings when exceeding certain PSD values could easily be built into the system.

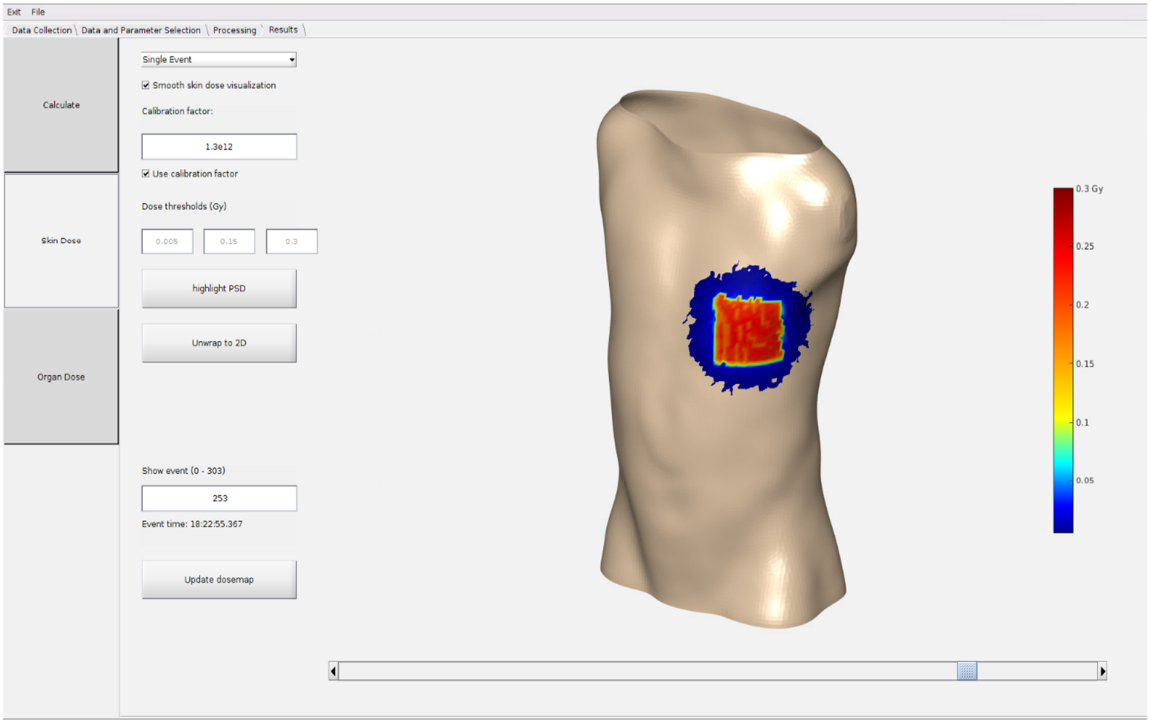

Figure 24 – Skin dose module showing the dose map from an event at a 30-degree angle with an arbitrary calibration factor

The organ dose module allows for a separate calculation of dose distributions where speed optimizations such as rejecting photons after a certain distance are not performed. Instead the dose in each voxel is calculated. This allows for an evaluation of the dose gradients inside the phantom which gives good approximation of the gradients inside the actual patient. The gradients can be evaluated for all voxels or separately per organ, for example isolating the skin or lungs. Besides evaluating dose gradients on the density image of the phantom the effective dose is calculated and histograms are available to evaluate the mean, median and maximum dose of the most important organs. An overview of the evaluation methods is given in Figure 25.

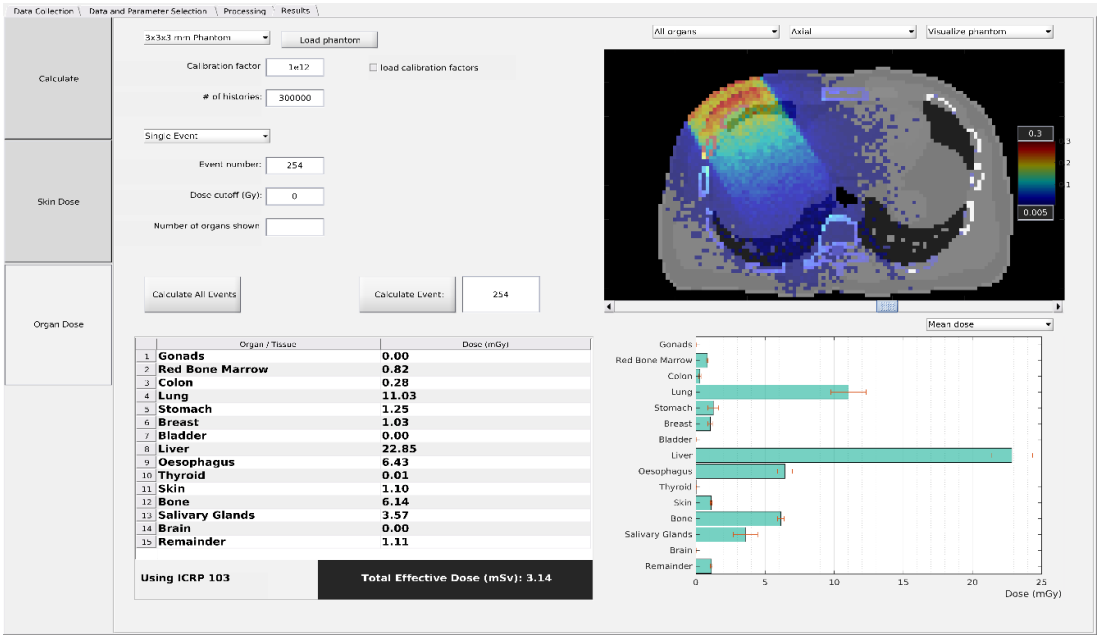

Figure 25 – Screenshot of organ dose viewer while evaluating a single event of a beam with a 30-degree rotation with an arbitrary calibration factor. The top left are user input parameters to customize the simulations and visualization. Top right shows the dose gradients in all voxels of a slice through the phantom. Bottom right shows the bar chart of the mean dose per organ, including the uncertainty from the simulation. Bottom left shows a table with the mean dose per organ and the effective dose for the selected ICRP guidelines, all tissue voxels not corresponding to one of the ICRP labels are grouped under the remainder tissues

4.3. Calculation time

Calculation time tests were performed on a high-end desktop with an Intel Core i9-7980XE CPU, GTX 1080Ti Nvidia GPU and 64 GB 3200MHz DDR4 memory. Table 3 shows the calculation times for each major step taken in the program excluding Organ dose visualization and Monte Carlo. Organ dose visualization calculation times were not determined as they do not belong to the time critical part of the system. However, no noticeable delay is present when evaluating different calculated events and charts in the organ dose module. The duration of the Monte Carlo simulation depends on many different parameters. Table 4 and 5 show the simulation times for the default settings, simulation times of different settings can be found in the tables of Appendix D.

Table 3 – Average calculation time for each major step of the program excluding Monte Carlo

Function group	Total time per event (s)	Relative time (%)
Import Images	0.019	0.55
Register Images	0.503	14.38
Detect markers	0.007	0.19
Segment and Register patient	0.805	23.00
Create point clouds	0.034	0.96
Read 3ddose	0.596	17.02
Interpolate colormap mesh	1.537	43.90
Monte Carlo input visualization	0.921	Optional

It is important to note that generating the input and performing the Monte Carlo simulation can occur in parallel. Meaning the input for the next event can be generated while the first is still being simulated. This parallel processing should have only a minor impact on the calculation time of both as the generation of input files is mostly single threaded with minor GPU acceleration leaving the remaining CPU threads available for EGSnrc. The interpolation of the colormap can also be optimized for speed but due to the offline nature of the prototype application this was not prioritized.

Table 4 – Calculation times for some different field sizes for skin dose calculations on a 5 mm x 5 mm x 5 mm upper body phantom with a 2.5 cm rejection distance. The uncertainty goal was ~5% on all voxels with a dose >50% of the max dose. Total time includes the time spent on file input and output by DOSXYZnrc

Field size (cm)	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories	Photons per cm^2
5x5	4.91	3.5	0.9	40,000	1,600
10 x 10	5.12	5.8	3.2	125,000	1,250
15 x 15	4.99	10.9	8.3	281,250	1,250
20 x 20	4.92	18.2	15.7	500,000	1,250
30 x 30	4.83	42.2	39.7	1,125,000	1,250

Table 5 – Calculation times for some different field sizes for organ dose calculations on a 3 mm x 3 mm x 3 mm upper body phantom. The uncertainty goal was ~5% on all voxels with a dose >50% of the max dose. Total time includes the time spent on file input and output by DOSXYZnrc

Field size (cm)	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories	Photons per cm^2
5x5	5.22	9.9	4.1	60,000	2,400
10x10	4.95	21.6	15.8	220,000	2,200
15x15	4.76	43.7	37.9	495,000	2,200
20x20	4.66	73.9	67.8	880,000	2,200
30x30	4.72%	153.9	148.1	1,980,000	2,200

The backscatter was found to be significant for depths up to 2.5 cm below the skin for simulations with ±5% average uncertainty for all doses above 50% of the max dose, after this skin dose will be accurate within uncertainty. This was determined using simulations with different backscatter distances compared a simulation with no backscatter rejection. Table 6 shows the difference in calculated dose rate versus backscatter distance and the time gain compared to a calculation with no photon rejection on a 5 mm x 5 mm x 5 mm phantom.

Table 6 – Relative CPU time speed increase and calculated dose rate for different backscatter rejection distances on a 5 mm x 5 mm x 5 mm phantom

Backscatter distance (cm)	Relative speed up (%)	Average dose rate (x10^{-15} Gy/Particle)
None	0	1.231 ± 5.84%
5.0	19.6	1.236 ± 5.82%
3.5	43.3	1.211 ± 5.89%
2.5	55.5	1.192 ± 5.94%
2.0	60.7	1.175 ± 5.99%
1.5	67.2	1.161 ± 6.04%
1.0	73.9	1.105 ± 6.20%
0.5	79.7	1.051 ± 6.37%
0	84.4	0.926 ± 6.84%

In Table 7 the calculation time for a 5% uncertainty is compared between organ dose and skin dose with 2.5 cm rejection distance for a 5 mm x 5 mm x 5 mm phantom. This illustrates the performance speed up from the adaptations to DOSXYZnrc.

Table 7 – CPU time for skin and organ dose calculations with 5% uncertainty on a 5 mm x 5 mm x 5 mm voxel phantom, a comparison in calculation speed. The performance increase column shows the relative calculation time gained for skin dose calculations

Field Size	CPU time skin (s)	CPU time organs (s)	Performance increase (%)
5x5	0.9	1.5	40.00
10x10	3.2	4.6	30.43
15x15	8.3	10.4	20.19
20x20	15.7	18	12.78
30x30	39.7	44.6	10.99

4.4. Film measurement

The relationship of the relative reflective density to dose was determined for each color channel. Figure 26 shows the difference in resolution for each color channel, especially at low doses.

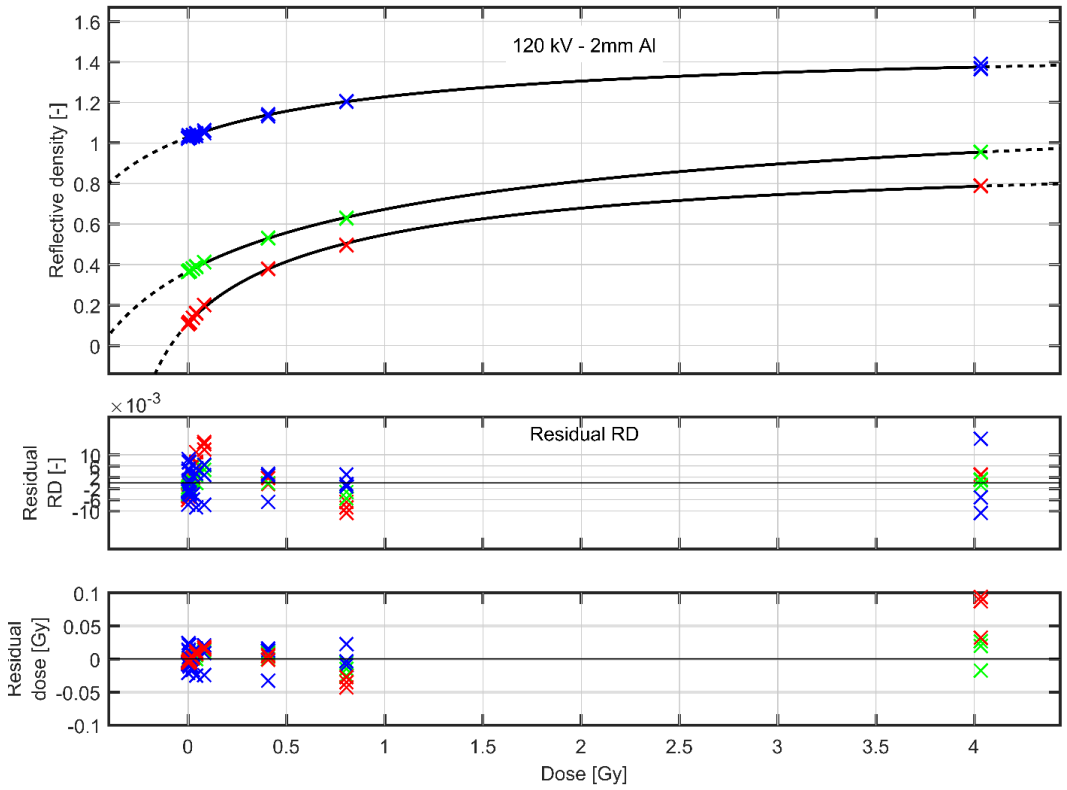

Figure 26 – Reflective density in relation to dose for the red, green and blue channels

The calibration fit was done using the relation between reflective density and dose according to equation (3).

$$RD = -\log_{10}\left(\frac{a + bD}{c + D}\right) \tag{3}$$

A least mean squares approach was used to determine the parameters for each color channel, the results can be seen in Table 8.

Table 8 – Parameters for film calibration fit

Parameter	Red	Green	Blue
a	0.274	0.316	0.074
b	0.110	0.053	0.032
c	0.354	0.736	0.787

The red channel shows a better dose sensitivity, especially at lower doses. Therefore, only the red channel was used for all further processing and as all measurements were in the low dose range the 4 Gy calibration point was removed to obtain the best possible fit. The new fit and the residuals are shown in Figure 27.

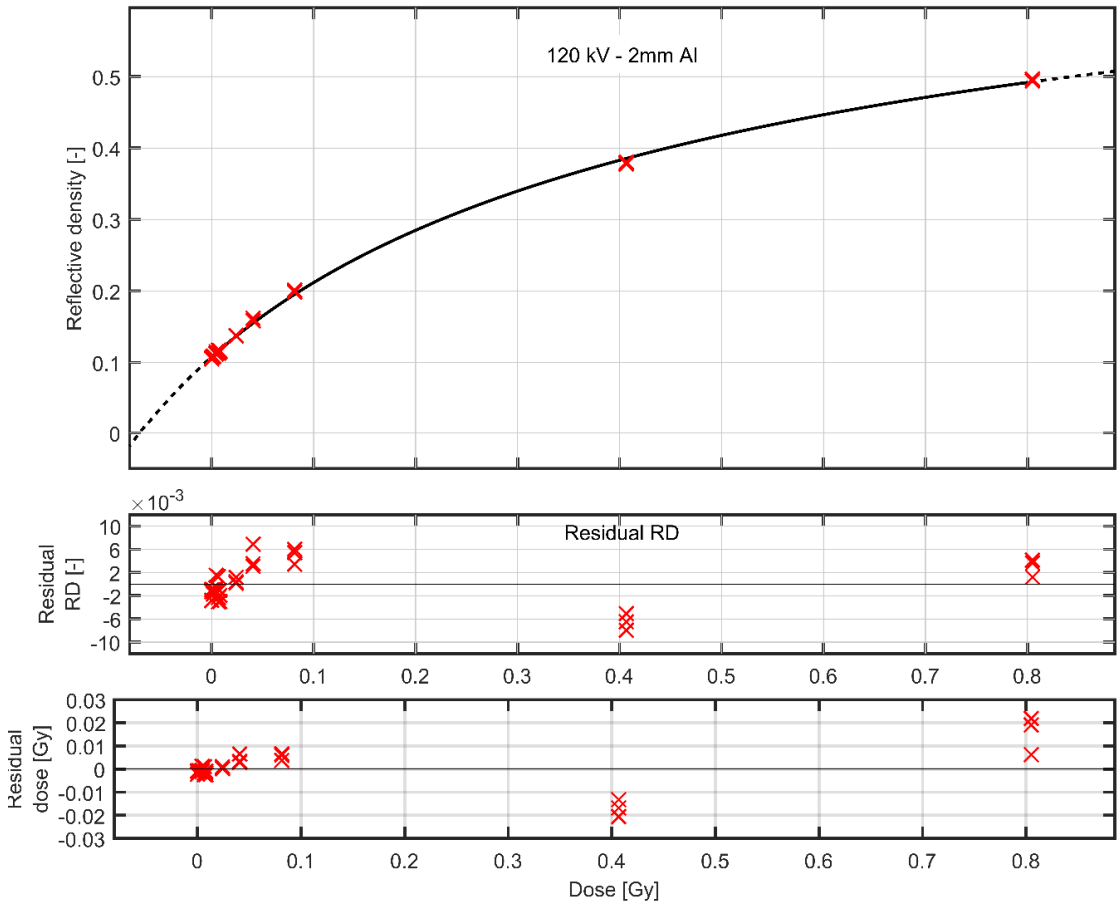

Figure 27 – Red channel calibration fit for low dose range

The films irradiated on the PMMA phantom could not be used due to underexposure as a consequence of the automatic exposure control. As an alternative the calibration factor was determined using the films from static anteroposterior (AP) projection of the RANDO phantom. A region of interest (ROI) was placed in the center part of the irradiated area. This was done on 2 different radiochromic films and the pixels inside all these regions of interest where averaged. A similar ROI was taken in the simulated skin dose multiplied by the tube current and exposure time from which we obtained a mean dose rate. Both regions of interests are shown for case 1 in Figure 28. The measured dose, simulated dose rates and the calculated conversion factor can be found in Table 9.

Table 9 – Calculation of the conversion factor for the 80 kV beam based on the RANDO phantom AP measurements

	Case 1	Case 2	Mean
Measured dose (Gy)	4.9×10^{-3}	5.5×10^{-3}	5.2×10^{-3}
Calculated dose rate (Gy mAs/particle)	7.2652×10^{-13}	8.9892×10^{-13}	8.1272×10^{-13}
		Conversion factor (particles/mAs)	6.3983×10^{9}

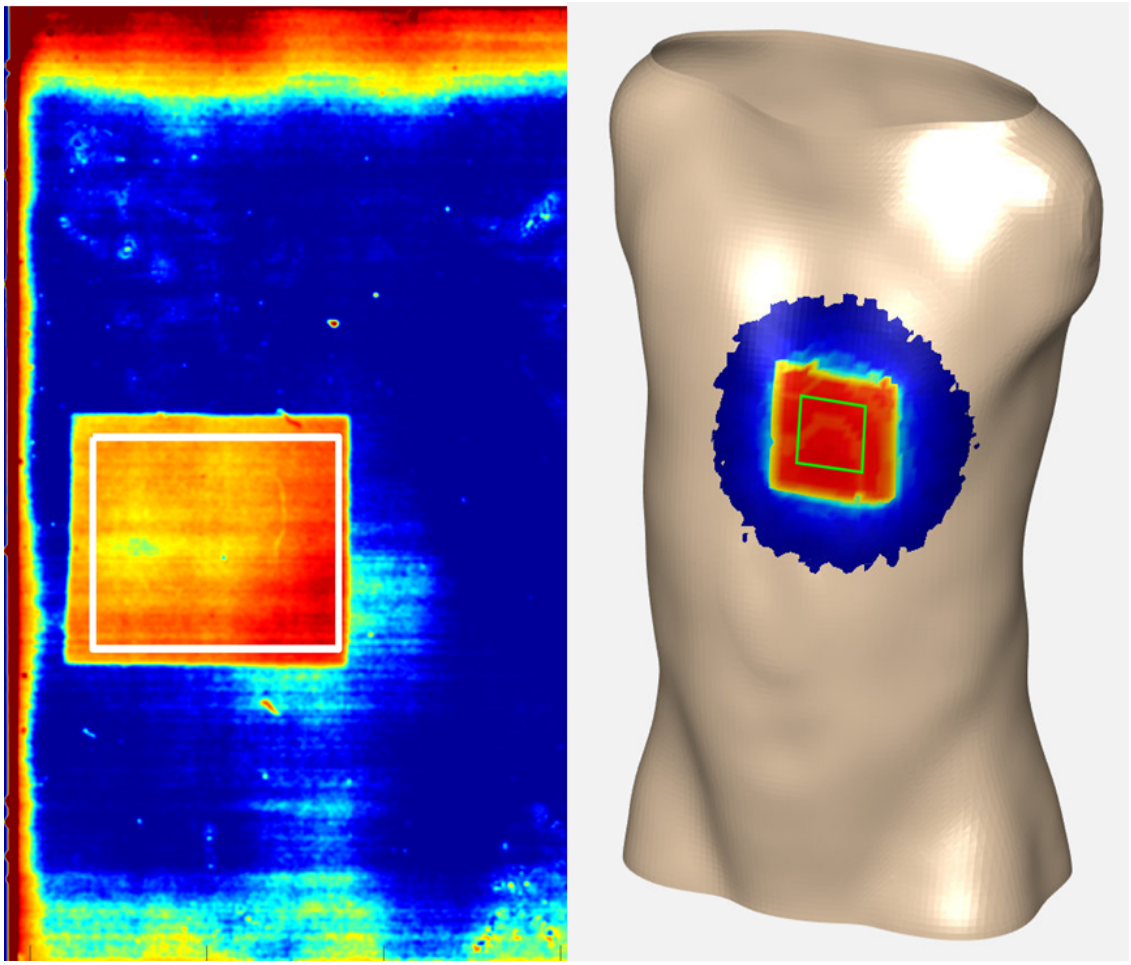

Figure 28 – Regions of interest used to determine the mean dose of the irradiated area. Left shows the ROI on the film scan and on the right the ROI on the simulated skin dose

The acquired calibration factor is then used on the other simulated series of exposures where 3 different tube angles were combined. The irradiation of two XR-RV3 films taped to a RANDO phantom was performed using 4 AP, 8 right anterior oblique (RAO) and 12 left anterior oblique (LAO) exposures. Both oblique exposures were performed at a 30-degree angle.

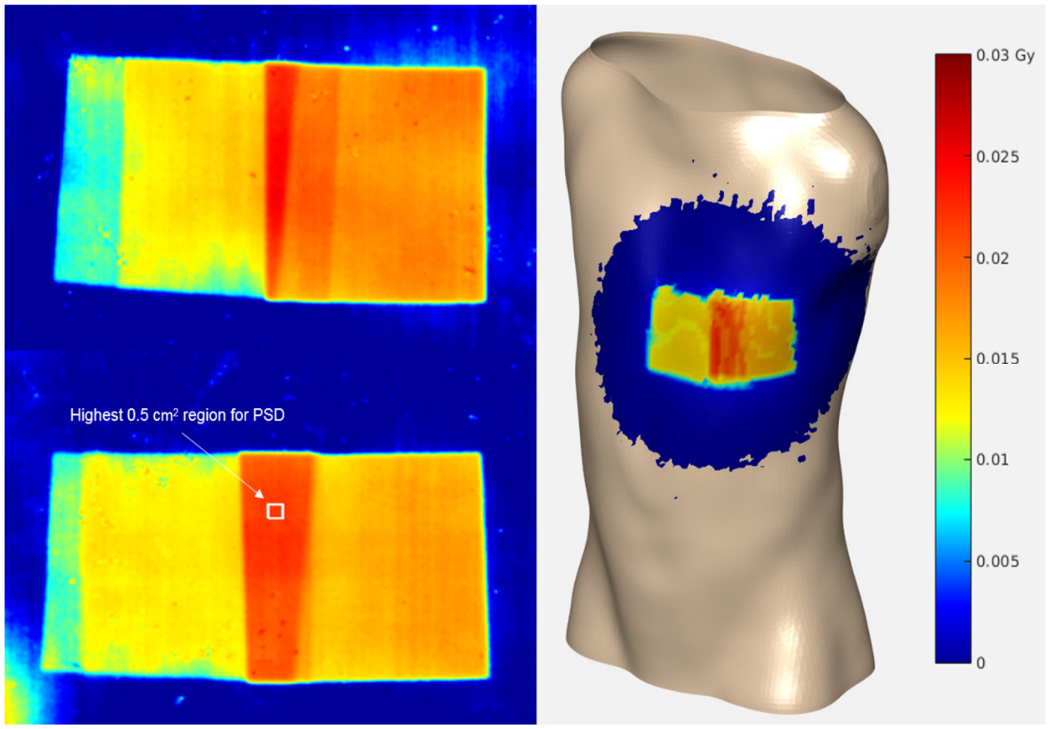

Figure 29 – On the left side the radiochromic film scans of case 1 and case 2 are displayed. The bottom left shows the region of interest used to determine the peak skin dose on the film scan of case 2. The left shows the simulated skin dose for case 2. All three images use the same color scale

Figure 29 shows both the radiochromic film scan and the calculated dose map for this multi beam irradiation. The peak skin dose in a 0.5 cm^2 area and the mean dose were determined for both film scans and the calculated data. The results are displayed in Table 10.

Table 10 – Calculated PSD and mean dose compared to the radiochromic film scan measurement for each case, difference is relative to the simulated dose of each individual case

	Case 1		**Case 2**	
	Simulation	Film scan	Simulation	Film scan
Mean dose (mGy)	12.8	12.8	12.3	12.5
PSD (mGy)	23.4	24.5	22.0	21.1
Difference mean (%)	0	0.2	0	1.6
Difference PSD (%)	0	4.8	0	-4.1

5. Discussion

The initial results show a difference of ±5% between the calculated and the measured PSD, the average only shows a ±2% difference. These initial results look promising but cannot be considered enough proof that the system functions appropriately. The film scans on the PMMA cube were not successful due to the automatic exposure control that prevented setting an appropriate dose. Due to this the AP projections on the RANDO phantom had to be used determine the Monte Carlo conversion factor, these simulations had a relatively similar positioning to the unknown cases creating a bias. Additionally, the films from the RANDO phantom are not ideal to perform the calibration of the conversion factor on due to its irregular shape and potential air gaps beneath the film.

Access to the clinical machine was limited in time. Therefore, only a few measurements could be performed in the period of this thesis limiting the number of test cases the system could be validated on. Additionally, the X-ray unit's parameters could not set to a fixed set of exposure parameters without collaboration of the machine's manufacture. This resulted in variations between each measurement. Due to these variations it is harder to detect possible measurement errors associated with the film scans as there is no reference. It also resulted in an underexposure of several film measurements.

The developed system selects a different spectrum depending on the tube parameters. It is not clear if the differences introduced by the spectrum changes are significant. But a quantitative evaluation of these differences is not possible without a conversion factor for the two different spectra. Initial tests were performed normalizing the calculated dose rates by their own maximum and these showed some differences. The negligible calculation time overhead associated with the different spectra led to keeping the spectrum selection part in the program, but its significance should be investigated.

The current patient geometry does not include the material for the table. This means the table is ignored during these dose calculations, introducing a potentially significant dose underestimation when the tube is below the table. The small correction for attenuation from the table will be implemented after the extensive validation measurements. Due to the increased computation time introduced by adding the table material to the geometry other options are explored first such as: choosing a different spectrum by adding the table material as a filter material; determining a different conversion factor for exposures below the table; or potentially even just using a correction factor for different angles below the table. More measurements are required to find the most optimal solution.

The organ dose module returns an effective dose in Sievert based on the ICRP 103 weighting factors. However, in the literature it is not clear if these weighting factors were calculated based on a dose calculation system that reports dose to water in water, dose to water in medium or dose to medium in medium. Most likely, it is the first option (dose to water in water). DOSXYZnrc always returns dose to medium in medium, adaptations have been reported to allow DOSXYZnrc to return dose to water in medium. But further investigation of this is advised and it might be required to apply different weighting factors to report a similar effective dose to be conform with ICRP 103.

Most Monte Carlo codes including DOSXYZnrc function through input and output files. This is an elegant solution as the time reading and writing these files is negligible compared to normal calculation times. However, the adaptations created to speed up the skin dose calculations greatly reduced the calculation time. These input/output actions now contribute to a significant increase in total time spent by the dose engine. Further development is planned to fully integrate the dose engine avoiding the time lost on reading and writing data to the hard drive.

The prototype was first aimed at successfully passing preclinical evaluation, which results in the patient detection and segmentation module being in an experimental phase. Several approaches are available: marker-based tracking being robust implementation for a human patient at this time; markerless tracking could be done by detecting joints similar to how the Kinect's skeleton tracking works and segmenting the body parts using the join locations. Or a body part segmentation using poselets or a convolutional neural network could be used. These methods have been investigated to assess the current state of technology but have not been fully implemented. The reason for not implementing these yet is due to the patient being partially covered by a surgical sheet during a clinical procedure. Observation of several interventions including some test data would be needed to construct a robust system. Overcoming the occlusion from the sheet could be done by adding markers to a swimcap that patients could wear during the examination, or if markerless detection is preferred adding one or more thermal cameras to the calibrated camera system could be a possible solution.

6. Conclusion

The initial tests were performed using the automatic exposure control on (there was no option to disable it), resulting in varying tube parameters. The tests were also too limited in number of measurement points and exposure types to fully validate the system. The largest deviation was a 4.75% underestimation of the peak skin dose and a 1.63% underestimation of the mean dose, which is within the 5% uncertainty of the simulation and the uncertainty of the XR-RV3 film. The visual representation of the Monte Carlo input geometry overlaid on the three-dimensional point cloud generated from the camera system showed good visual agreement over a large range of tube angles.

More extensive dosimetric measurements are needed to validate the system in a larger range of positions and exposure parameters. A more controlled determination of the Monte Carlo conversion factor should also be performed using a simple geometry. Additional investigation is needed to see how the patient detection algorithm could be best implemented in clinical practice with the presence of a surgical sheet.

The prototype system shows that combining computer vision with Monte Carlo simulations shows great potential in providing accurate and automated dose calculations in radiology improving on calculations based on the RDSR information alone. The limited calculation times required for skin dose calculations also show that real-time skin dose feedback is possible after integration with the X-ray unit. It is capable of reporting doses after several seconds delay depending on field sizes and requested uncertainties. This combined with the user defined thresholds could allow valuable feedback for the physician during the procedure, indicating when a change in the X-ray tube position is necessary to avoid skin lesions.

References

[1] W. G. Bradley, "History of Medical Imaging," *Proceedings of the American Philosophical Society,* vol. 152, no. 3, pp. 349-361, 2008.

[2] F. Van Gelderen, "A Brief History of Radiology," in *Understanding X-Rays*, Berlin Heidelberg, Springer-Verlag, 2005, pp. 597-602.

[3] W. Inkret, C. Meinhold and J. Taschner, "Radiation and Risk - A Hard Look At the Data," *Los Alamos Science,* vol. 23, pp. 116-123, 1995.

[4] P.-J. P. Lin, B. A. Scheuler, S. Balter, K. J. Strauss, K. A. Wunderle, T. M. LeFrance, D.-S. Kim, R. H. Behrman, J. S. Shepard and I. H. Bercha, "Accuracy and calibration of integrated radiation output indicators in diagnostic radiology: A report of the AAPM Imaging Physics Committee Task Group 190," *Med Phys,* vol. 42, no. 12, pp. 6815-6829, 2015.

[5] E. Bogaert, K. Bacher, K. Lemmens, M. Carlier, M. Desmet, X. De Wagter, D. Dijan, C. Hanet, G. Heyndrickx, V. Legrand, Y. Taeymans and H. Thierens, "A large-scale multicentre study of patient skin dosesin interventional cardiology: dose-area product action levels and dose reference levels," *The Britisch Journal of Radiology,* vol. 82, pp. 303-312, 2009.

[6] "Council Directive 2013/59/EURATOM: laying down basic safety standard for protection against the dangers arising from exposure to ionising radiation, and repealing Directives 89/618/Euratom, 90/641/Euratom, 96/29/Euratom, 97/43/Euratom and 2003/122/Euratom," 5 December 2013. [Online]. Available: https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2014:013:0001:0073:EN:PDF. [Accessed 6 October 2018].

[7] N. Fitousi, "Patient dose monitoring systems: A new way of managing patient dose and quality in the radiology department," *Physica Medica,* 2017.

[8] H. Wang, Y. Ma, G. Pratx and L. Xing, "Toward real-time Monte Carlo simulation using a commercial cloud computing infrastructure," *Phys Med Biol,* vol. 56, no. 17, pp. 175-181, 2011.

[9] D. Kessel, "What is Interventional Radiology?," British Society of Interventional Radiology, 2018. [Online]. Available: https://www.bsir.org/patients/what-is-interventional-radiology/. [Accessed 2018 12 16].

[10] J. Rösch and F. S. Keller, "Historical Account: Cardiovascular Interventional Radiology," in *Catheter-Based Cardiovascular Interventions*, Berlin Heidelberg, Springer-Verlag, 2013, pp. 15-26.

[11] T. Teo, B. Tan and K. Tay, "A brief history of interventional radiology in Singapore and its current status," *Biomedical Imaging and Intervention Journal,* vol. 7, no. 2, 2011.

[12] J. Greffier, C. Van Ngoc Ty, G. Bonniaud, G. Moliner, B. Ledermann, L. Schmutz, L. Cornillet, G. Cayla, J. Beregi and F. Pereira, "Assesment of peak skin dose in interventional cardiology: A comparison between Gafchromic film and dosimetric software em.dose," *Physica Medica,* vol. 38, pp. 16-22, 2017.

[13] S. Van De Putte, F. Verhaegen, Y. Taeymans and H. Thierens, “Correlation of patient skin doses in cardiac interventional radiology with dose-area product,” *The British Journal of Radiology,* vol. 73, pp. 504-513, 2000.

[14] A. Jonas, J. Ensor and A. Pasciak, “How accuratly can the peak skin dose in fluoroscopy be determined using indirect dose metrics?,” *Med Phys,* vol. 41, 2014.

[15] R. M. Sanchez, E. Vano, J. M. Fernandez Soto and J. I. Ten, “Validation of a quick skin dose map estimator for interventional cardiology procedures,” in *European Congress of Radiology*, Vienna, 2018.

[16] D. R. Bednarek, J. Barbarits, V. K. Rana, S. P. Nagaraja, M. S. Josan and S. Rudin, “Verification of the performance accuracy of a real-time skin-dose tracking system for interventional fluoroscopic procedures,” in *Proceedings of SPIE*, 2011.

[17] M. A. Ozeroglu, T. E. Johnson and S. A. Nemmers, “Verification of Caregraph Peak Skin Dose Data Using Radiochromic Film,” Department of Preventive Medicine and Miometrics, Bethesda, 2005.

[18] Qaelumn, “Qaelumn Announces Partnership with Virtual Phantoms, Inc to Add Organ Dose Data to Radiation Dose Monitoring Software,” Qaelumn NV, [Online]. Available: https://qaelum.com/view/blog/press-release-qaelum-announces-partnership-with-virtual-phantoms-inc-to-add-organ-dose-data-to-radiation-dose-monitoring-software. [Accessed 10 September 2018].

[19] Siemens, “teamplay: The departmental performance management solution for Radiology and Cardiology,” Siemens, [Online]. Available: https://www.healthcare.siemens.com/infrastructure-it/digital-ecosystem/teamplay#Home. [Accessed 10 September 2018].

[20] Siemens Healthineers, “teamplay data sheet,” Siemens, Erlangen, 2017.

[21] GE Healthcare, “DoseWatch: Radiation Dose in Interventional Radiology,” General Electric Company, 2013.

[22] D. Fornell, “GE Healthcare to License Duke University's CT Organ Dosimetry Technology,” Imaging Technology News, 31 July 2017. [Online]. Available: https://www.itnonline.com/content/ge-healthcare-license-duke-universitys-ct-organ-dosimetry-technology. [Accessed 10 February 2018].

[23] K. O'Reilly, “Philips announces DoseWise Portal 2.0 to manage radiation risk for both patients and clinicians at RSNA 2015,” Philips, 1 December 2015. [Online]. Available: https://www.usa.philips.com/a-w/about/news/archive/standard/news/press/2015/20151201-Philips-announces-DoseWise-Portal-2-0-to-manage-radiation-risk-for-both-patients-and-clinicians-at-RSNA-2015.html. [Accessed 7 February 2018].

[24] C. Martel and D. Siewko, “DoseWise: The role of dose tracking systems in radiation safety programs,” Philips, 2016.

[25] Sectra, “Sectra DoseTrack: Cloud-based solution for patient radiation dose monitoring,” Sectra, [Online]. Available: https://sectra.com/medical/product/sectra-dosetrack/. [Accessed 3 February 2018].

[26] Virtual Phantoms, Inc, “Sectra and Virtual Phantoms are Partners!,” 15 May 2015. [Online]. Available: http://www.virtualphantoms.com/sectra-and-virtual-phantoms/. [Accessed 8 February 2018].

[27] Bayer HealthCare, “Radiation Dose Calculations in Radimetric Enterprise Platform by Bayer,” Bayer, Whippany, 2014.

[28] Bayer HealthCare, “Size Specific Dose Estimate (SSDE) Calculations Including in Release 2.1 of Bayer Healthcare's Radimetrics Enterprise Platform,” Bayer, Whippany, 2014.

[29] National Institute for Health and Care Excellence, “Radiation dose monitoring software for medical imaging with ionising radiation,” 2017.

[30] “PACSHealth Partners With Virtual Phantoms to Improve Patient Radiation Dose Monitoring,” Imaging Technology News, 13 July 2015. [Online]. Available: https://www.itnonline.com/content/pacshealth-partners-virtual-phantoms-improve-patient-radiation-dose-monitoring. [Accessed 8 February 2018].

[31] E. McDonagh, “OpenREM Documentation Release 0.9.0b3,” 2018.

[32] Toshiba, “Visualize your way to better patient safety: Infinix Dose Tracking System,” 2014.

[33] J. Seco and F. Verhaegen, Monte Carlo Techniques in Radiation Therapy, CRC Press, 2016.

[34] A. Kajaria, N. Sharma, S. Sharma, S. Pradhan and L. M. Aggarwal, “Review of Monte Carlo Simulation in Radio Therapy Treatment Planning System for Modelling of Radiation Transport & Dose Calculation,” *VAICHARIKI,* vol. III, no. 4, 2013.

[35] I. Kawrakow, E. Mainagra-Hing, D. Rogers, F. Tessier and B. R. Walters, “The EGSnrc Code System: Monte Carlo Simulation of Electron and Photon Transport,” National Research Council of Canada, Ottawa, 2018.

[36] D. W. Rogers, B. Walters and I. Kawrakow, “BEAMnrc Users Manual,” National Research Council of Canada, Ottawa, 2018.

[37] B. Walters, I. Kawrakow and D. Rogers, “DOSXYZnrc User Manual,” National Research Council of Canada, Ottowa, 2018.

[38] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge: Cambridge University Press, 2004.

[39] T. S. Huang, “Computer Vision: Evolution and Promise,” *19th Cern Sch. Comput.,* pp. 21-25, 1996.

[40] K. Dawson-Howe, A Practical Introduction to Computer Vision with OpenCV, Wiley, 2014.

[41] B. Pinnamaneni, “Machine Vision Systems and Image Processing with Applications,” *Journal of Innovation in Computer Science and Engineering,* vol. 2, no. 2, 2013.

[42] G. Bradski, “The OpenCV Library,” *Dr. Dobb's Journal of Software Tools,* 2000.

[43] T. Zhongwei, “High precision camera calibration,” *General Mathematics,* 2011.

[44] Z. Zhang, “A Flexible New Technique for Camera Calibration,” Microsoft Research, Redmond, 1998.

[45] L. Xiang, F. Echtler, C. Kerl, T. Wiedemeyer, L. Hanyazou, R. Gordon, F. Facioni, R. Wareham, M. Goldhoorn, A. Gaborpapp, S. Fuchs, Imtatsch, J. Blake, Federico, H. Jungkurth, Y. Mingze, Vinouz, D. Coleman, B. Burns, R. Rawat, S. Mokhov, P. Reynolds, P. Viae, M. Fraissinet-Tachet, Ludique, J. Billingham and Alistair, “libfreenect2: Release 0.2,” *Zenodo,* 28 April 2016.

[46] K. Yamaguchi, “mexopencv v3.4.0,” [Online]. Available: https://github.com/kyamagu/mexopencv. [Accessed 10 April 2018].

[47] A. Maimone and H. Fuchs, “Reducing interference between multiple structured light depth sensors using motion,” in *VR '12 Proceedings of the 2012 IEEE Virtual Reality*, Washington, 2012.

[48] O. Wasenmüller and D. Stricker, “Comparison of the Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision,” in *Computer Vision - ACCV 2016 International Workshops*, 2016.

[49] Y. He, B. Liang, Y. Zou, J. He and J. Yang, “Depth Errors Analysis and correction for Time-of-Flight (ToF) Cameras,” *Sensors,* vol. 17, 2017.

[50] M. Hadad, M. Saeedi-Moghadam and B. Zeinali-Rafsanjani, “Voxel dosimetry: Comparison of MCNPX and DOSXYZnrc Monte Carlo codes in patient specific phantom calculations,” *Technology and Health Care,* vol. 25, pp. 29-35, 2017.

[51] B. K. P. Horn, “Closed-form solution of absolute orientation using unit quaternions,” *Journal of the Optical Society of America,* vol. 4, pp. 629-642, 1987.

[52] G. Poludniowski, G. Landry, F. DeBlois, P. Evans and F. Verhaegen, “SpekCalc: a program to caclulate photon spectra from tungsten anode X-ray tubes,” *Phys Med Biol,* vol. 54, no. 19, pp. 433-438, 2009.

[53] S. Garrido-Jurado, R. Munoz-Salinas, F. Madrid-Cuevas and M. Marın-Jimenez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition,* vol. 47, no. 6, pp. 2280-2292, 2014.

[54] P. W. Segars, G. Sturgeon, S. Mendonca, J. Grimes and W. B. Tsui, “4D XCAT phantom for multimodality imaging research,” *Med Phys,* vol. 37, no. 9, pp. 4902-4915, 2010.

[55] P. W. Segars, J. Bond, J. Frush, S. Hon and C. Eckersley, “Population of anatomically variable 4D XCAT adult phantoms for imaging research and optimization,” *Med Phys,* vol. 40, no. 4, p. 043701, 2013.

[56] B. Van Der Heyden, L. E. Schyns, M. Podesta, A. Vaniqui, I. P. Almeida, G. Landry and F. Verhaegen, “VOXSI: A voxelized single- and dual-energy CT scenario generator for quantitative imaging,” *Physics and Imaging in Radiation Oncology,* vol. 6, pp. 47-52, 2018.

[57] A. Laurent, F. Mistretta, D. Bottigioli, K. Dahel, C. Goujon, J.-F. Nicolas, A. Hennino and P. E. Laurent, “Echographic measurement of skin thickness in adults by high frequency ultrasound to asses the appropriate microneedle length for intradermal delivery of vaccines,” *Vaccine,* vol. 25, pp. 6423-6430, 2007.

[58] T. Jost and H. Hügli, “Fast ICP algorithms for shape registration,” in *Pattern Recognition (In: Lecture Notes in Computer Science) 2449*, 2002, pp. 91-99.

[59] H. M. Kjer and J. Wilm, *Evaluation of surface registation algorithms for PET motion correction,* Lyngby: Technical University of Denmark, 2010.

[60] ICRP, “The 2007 Recommendations of the International Commission on Radiological Protection,” *ICRP Publication 103,* Vols. Ann. ICRP 37 (2-4), 2007.

[61] A. Jones, “Chapter 8: Fluoroscopic Imaging Systems,” in *Diagnostic Radiology Physics*, IAEA, 2014, pp. 183-207.

[62] C.-M. Ma, C. Coffey, L. DeWerd, C. Liu, R. Nath, S. Seltzer and J. Seuntjens, “AAPM protocol for 40-300 kV x-ray beam dosimetry in radiotherapy and radiobiology,” *Medical Physics,* vol. 28, no. 6, pp. 868-893, 2001.

[63] B. P. McCabe, M. A. Speidel, T. L. Pike and M. S. Van Lysel, “Calibration of GafChromic XR-RV3 radiochromic film for skin dose measurements using a standardized x-ray spectra and a commercial flatbed scanner,” *Medical Physics,* vol. 38, pp. 1919-1930, 2011.

[64] J. Farah, A. Trianni, O. Ciraj-Bjelac, I. Clairand, C. De Angelis, S. Delle Canne, L. Hadid, C. Huet, H. Jarvinen, A. Negri, L. Novak, M. Pinto, T. Siiskonen, M. Waryn and Z. Knezevic, “Characterization of XR-RV3 GafChromic films in standard laboratory and in clinical conditions and means to evaluate uncertainties and reduce errors,” *Medical Physics,* vol. 42, pp. 4211-4226, 2015.

[65] T. Wiedemeyer, “IAI Kinect2,” Institute for Artifical Intelligence, University of Bremen, 2014. [Online]. Available: https://github.com/code-iai/iai_kinect2. [Accessed 20 Juli 2018].

[66] F. Dunn and I. Parberry, 3D Math Primer for Graphics and Game Development, Texas: Wordware Publishing, Inc, 2002.

Appendix A – Depth to color registration including source code sample

The source code sample below shows the class used to store image data and register color and depth. This algorithm was inspired by Wiedemeyer's approach in iai_kinectv2, a robotic operating system (ROS) bridge for Libfreenect2 [65]. Routinely the registration is performed using a C++ algorithm called through a mex file for a fast calculation (cv.doseguardRegisterDepth), but the initial MATLAB implementation of the algorithm is shorter and simpler to read in the MATLAB class provided below. Note that the "doseguard.data.LookUpTable" class code is not shown but this class generates a lookup table for each pixel that contains: the pixel position minus the principal point divided by the focal length. This is done to avoid unnecessary operations as the results of these operations stay the same for each frame from the same camera.

The registration code is started from the "registerDepth" method and follows several steps to attain a color and depth image with the point in three-dimensional mapped to the same pixel.

First the "cv.initUndistortRectifyMap" function is called with an identity matrix as transformation to bypass that step. This function follows a three step process: First the pixels (u,v) are converted to coordinates using the infrared intrinsic camera parameters, principal point (c') and focal length (f') as seen in equation (A.1).

$$x \leftarrow \frac{u - c'_x}{f'_x}$$
$$y \leftarrow \frac{v - c'_y}{f'_y} \tag{A.1}$$

Then the radial (k) and tangential distortion factor (p) are used to correct for lens distortion using equation (A.2).

$$x' = x\,(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x\,y + p_2(r^2 + 2x^2)$$
$$y' = y\,(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2x^2) + 2p_2 x\,y \tag{A.2}$$

As a final step the coordinates are projected back to pixel space using the color intrinsic camera parameters as shown in equation (A.3) [42].

$$map_x(u,v) \leftarrow x' f_x + c_x$$
$$map_y(u,v) \leftarrow y' f_y + c_y \tag{A.3}$$

These two new maps are arrays of the same size as the full HD color image. They hold the x or y pixel positions of the original depth image as if the image was originally taken by the infrared camera with the color camera's focal length and principal point.

Then these maps are used to perform a form of bilinear interpolation as shown in the method "interpolateDepth". A few additional checks were added to avoid reducing the sharpness of edges by adding a check for difference in depth value. The new resulting image is the depth image as if it had the intrinsic camera matrix of the color camera.

While the resolution and camera parameters are now the same for the color and depth image the scene is still captured from 2 cameras that are not physically on the same position, this is solved by

projecting the points using the pinhole mode and the depth value from each pixel as described by equation A.4 and the depth value is just equal to the depth value assigned to the pixel.

$$x = \frac{z\,(u - c_x)}{f_x}$$
$$y = \frac{z\,(v - c_y)}{f_y} \qquad \text{(A.1)}$$

These three-dimensional points are the rigidly transformed by the transformation matrix that was determined in the stereo calibration and projected back using the same pinhole model. The Z-buffer algorithm is used to handle multiple points being reprojected back onto the same pixel, if this occurs the all points but the closest one to the camera get rejected.

The final step is to remove lens distortion from the color [66] image as the registered depth has no distortion after reprojecting.

Source code sample

```
classdef KinectImageData < matlab.mixin.Copyable
    % package doseguard.data
    %
    %-------------------------------------------------------------------
    %**** updates / major changes ************************************
    % 04-04-2018 - class creation
    %
    %-------------------------------------------------------------------
    %**** Class description ******************************************
    %
    % // Summary
    % This class reads out the binary images written by the Odroid (or other
    % intermediate computer doing the recording)
    %
    % // Methods
    % obj.registerDepth uses the camera calibration parameters to change
    % the pixel coordinates of the depth map to match this of the color,
    % the resulting matched arrays are rectified color and registered depth
    % it tries to call the mex implementation first but uses the MATLAB
    % version if this is not compiled or the opencv library cannot be
    % called.
    %
    % obj.reprojectDepthMex uses a C++ implementation to speed up the
    % registration as it this was one of the bottlenecks of the
    % application, currently mex implementation runs at 0.1s, if a faster
    % registration is needed a GPU implementation could be made.
    %
    % obj.readBinaryFile reads in the image data from a binary file and is
    % automatically launched when a image data filepath is added or given
    % as input

    properties
        originalColor(:,:,:) uint8 = []
        infrared(:,:) single = []
        depth(:,:) single = []

        calibrationData(1,:) doseguard.data.CameraCalibrationData =
doseguard.data.CameraCalibrationData.empty
        transformationData(1,:) doseguard.analysis.TransformCoordinateSystems =
doseguard.analysis.TransformCoordinateSystems.empty;
```

```
        filePath(1,:) smart.io.Path = smart.io.Path.empty
        headerInfo(1,:) doseguard.io.ImportKinectData = doseguard.io.ImportKinectData.empty
        lookupTable(1,:) doseguard.data.LookUpTable = doseguard.data.LookUpTable.empty

        minimalDepth(1,1) = 0.3
        maximumDepth(1,1) = 5
    end

    properties (SetAccess = protected)
        rectifiedColor(:,:,:) uint8  = []
        registeredDepth(:,:) single = []
    end

    methods
        function newObj = KinectImageData(varargin)
            if nargin > 0
                for idx = 1:numel(varargin)
                    if isa(varargin{idx},'smart.io.Path') ||  ischar(varargin{idx})
                        assert(strcmpi(varargin{idx}.extension, '.rgbd'),'The file extension
is not .rgbd and thus the file does not contain the correct data');
                        if ischar(varargin{idx})
                            newObj.filePath = smart.io.Path(varargin{idx});
                        else
                            newObj.filePath = varargin{idx};
                        end
                        newObj.headerInfo = doseguard.io.ImportKinectData(newObj.filePath);
                        [newObj.infrared, newObj.depth, newObj.originalColor] = ...
                            newObj.headerInfo.readImagesFromRGBD;
                    elseif isa(varargin{idx},'doseguard.data.CameraCalibrationData')
                        newObj.calibrationData
                    elseif isa(varargin{idx}, 'doseguard.io.ImportKinectData')
                        newObj.headerInfo = varargin{idx};
                        newObj.filePath = varargin{idx}.filePath;
                        [newObj.infrared, newObj.depth, newObj.originalColor] = ...
                            newObj.headerInfo.readImagesFromRGBD;
                    elseif isa(varargin{idx}, 'doseguard.io.LookUpTable')
                        newObj.lookupTable = varargin{idx};
                    end
                end
            end
        end

        function registerDepthMex(this)
            [registeredDepth, this.rectifiedColor] = cv.doseguardRegisterDepth(...
                uint16(this.depth), this.originalColor,...
                this.calibrationData.colorMatrix,...
                this.calibrationData.infraredMatrix,...
                this.calibrationData.transformationMatrix,...
                this.calibrationData.infraredDistortionCoeffs,...
                this.calibrationData.colorDistortionCoeffs);

            this.registeredDepth = single(registeredDepth);
        end

        function registerDepth(this)
            try
                this.registerDepthMex;
                return
            catch
```

```
            warning('could not launch mexfile so running matlab code');
        end
        % initialize the same size depth image
        numberOfRows = size(this.originalColor,1);
        numberOfColumns = size(this.originalColor,2);

        scaledDepth = zeros(numberOfRows, numberOfColumns);

        % create the map of depth with the resolution and camera matrix
        % of the color image, opencv uses [width height] instead of
        % [rows columns]
        [mapX, mapY] = cv.initUndistortRectifyMap(...
            this.calibrationData.infraredMatrix,...
            this.calibrationData.infraredDistortionCoeffs,...
            [numberOfColumns numberOfRows], 'NewCameraMatrix',...
            this.calibrationData.colorMatrix, 'M1Type', 'single1');

        % interpolate the depthValue to the new resolution and camera
        % matrix
        for rowIdx = 1:numberOfRows
            for columnIdx = 1:numberOfColumns
                scaledDepth(rowIdx, columnIdx) = ...
                    this.interpolateDepth(mapX(rowIdx,columnIdx),...
                    mapY(rowIdx,columnIdx));
            end
        end

        if isempty(this.lookupTable)
            this.lookupTable = doseguard.data.LookUpTable(this);
            this.lookupTable.generateLookUp(this);
            lookupX = this.lookupTable.lookupTableX;
            lookupY = this.lookupTable.lookupTableY;
        else
            % get the lookups in memory to avoid having to grab it from
            % the handle every time (need to check if it is actually
            % faster later on
            lookupX = this.lookupTable.lookupTableX;
            lookupY = this.lookupTable.lookupTableY;
        end
        % preallocating
        registeredDepth = zeros(numberOfRows, numberOfColumns);

        scaledDepth = double(scaledDepth ./ 1000);
        fx = this.calibrationData.colorMatrix(1,1);
        fy = this.calibrationData.colorMatrix(2,2);
        cx = this.calibrationData.colorMatrix(1,3);
        cy = this.calibrationData.colorMatrix(2,3);
        for rowIdx = 1:numberOfRows
            for columnIdx = 1:numberOfColumns

                % conversion from mm to meter
                depthValue = scaledDepth(rowIdx, columnIdx);

                 % cut off extremely close and far values to avoid
                 % unreliable results
                 if depthValue < this.minimalDepth || depthValue > this.maximumDepth
                     continue
                 end

                % assign the original 3D coordinate using the pinhole
```

```
                    % camera model
                    originalCoordinates = [...
                        lookupX(columnIdx)*depthValue;...
                        lookupY(rowIdx)*depthValue;...
                        depthValue; 1];


                    % apply the rotation and translation to convert from
                    % the viewing frame from depth to color space
                    transformedPoints = this.calibrationData.transformationMatrix *...
                        originalCoordinates;

                    % project back to color image plane to get the depthmap
                    % in same pixel coordinates as the color image
                    inverseDepth = 1/transformedPoints(3);

                    newX = (fx * transformedPoints(1)) * inverseDepth + cx;
                    newY = (fy * transformedPoints(2)) * inverseDepth + cy;

                    % round of to integer value to use as pixel coordinates
                    newX = round(newX);
                    newY = round(newY);

                    % filter out pixel values outside of the image range
                    if (newX >= 1 && newX <=  numberOfColumns && newY >= 1 && newY <=
numberOfRows)
                        % return to millimeters
                        newDepth = single(transformedPoints(3) * 1000);

                        % use Z-buffer
                        previousRegistration = registeredDepth(newY, newX);

                        if previousRegistration == 0 || newDepth < previousRegistration
                            registeredDepth(newY, newX) = newDepth;
                        end
                    end
                end
            end

            this.registeredDepth = cv.medianBlur(single(registeredDepth), 'KSize', 5);

            % the registration requires a rectified color image
            % initialize undistortion maps
            [map1Color, map2Color] = ...
                cv.initUndistortRectifyMap(this.calibrationData.colorMatrix,...
                this.calibrationData.colorDistortionCoeffs, [numberOfColumns numberOfRows],...
                'NewCameraMatrix', this.calibrationData.colorMatrix,...
                'M1Type', 'int16');

            % create undistorted color image
            this.rectifiedColor = cv.remap(this.originalColor,...
                map1Color, map2Color, 'Interpolation', 'Lanczos4');

        end

        function depthValue = interpolateDepth(this, x, y)
            % initiate the values for bilineair interpolation
            xLow = floor(x);
            xHeight = ceil(x);
            yLow = floor(y);
```

```
            yHeight = ceil(y);

            % if the map value does not fit inside the original depth image
            % assign a zero value
            if xLow < 1 || yLow < 1 || xHeight > size(this.depth, 2) || yHeight >
size(this.depth, 1)
                depthValue = 0;
                return
            end

            % check if the depth image contains valid information or if the
            % value is 0 (outside of active depth range)
            depthLeftTop = this.depth(yLow, xLow);
            depthRightTop = this.depth(yLow, xHeight);
            depthLeftBottom = this.depth(yHeight, xLow);
            depthRightBottom = this.depth(yHeight, xHeight);

            validLeftTop = depthLeftTop > 0;
            validRightTop = depthRightTop > 0;
            validLeftBottom = depthLeftBottom > 0;
            validRightBottom = depthRightBottom > 0;
            numberOfValids = validLeftTop + validRightTop + validLeftBottom +
validRightBottom;

            % don't allow more than 2 bad pixels because interpolation is
            % not very reliable then
            if numberOfValids < 3
                depthValue = 0;
                return
            end

            % detect extreme edges to avoid unrealistic results
            averageDepth = (depthLeftTop + depthRightTop +...
                depthLeftBottom + depthRightBottom) / numberOfValids;
            thresholdValue = 0.01 * averageDepth;
            validLeftTop = abs(depthLeftTop - averageDepth) < thresholdValue;
            validRightTop = abs(depthRightTop - averageDepth) < thresholdValue;
            validLeftBottom = abs(depthLeftBottom - averageDepth) < thresholdValue;
            validRightBottom = abs(depthRightBottom - averageDepth) < thresholdValue;
            numberOfValids = validLeftTop + validRightTop + validLeftBottom +
validRightBottom;


            if(numberOfValids < 3)
                depthValue = 0;
                return
            end

            distanceXLow = x - xLow;
            distanceXHigh = 1.0 - distanceXLow;
            distanceYLow = y - yLow;
            distanceYHigh = 1.0 - distanceYLow;

            normalizationValue = sqrt(2.0);

            if validLeftTop
                finalLeftTop = normalizationValue - sqrt(distanceXLow + distanceYLow);
            else
                finalLeftTop = 0;
            end
```

```
            if validRightTop
                finalRightTop = normalizationValue - sqrt(distanceXHigh + distanceYLow);
            else
                finalRightTop = 0;
            end


            if validLeftBottom
                finalLeftBottom = normalizationValue - sqrt(distanceXLow + distanceYHigh);
            else
                finalLeftBottom = 0;
            end


            if validRightBottom
                finalRightBottom = normalizationValue - sqrt(distanceXHigh + distanceYHigh);
            else
                finalRightBottom = 0;
            end

            sumValues = finalLeftTop + finalRightTop + finalLeftBottom + finalRightBottom;

            depthValue = ((depthLeftTop * finalLeftTop +...
                depthRightTop * finalRightTop +...
                depthLeftBottom * finalLeftBottom +...
                depthRightBottom * finalRightBottom) / sumValues) + 0.5;

        end
    end % end of methods no attributes
end % end of class
```

Appendix B – MATLAB implementation Horn's method

Small part of the class "doseguard.analysis.TransformCoordinateSystems" to illustrate the implementation of Horn's method for absolute orientation using unit quaternions as described in [51].

```
classdef TransformCoordinateSystems < handle
    % package doseguard.analysis

    properties [...]

    properties (SetAccess = protected) [...]

    properties (Hidden) [...]

    methods [...]

    methods (Access = protected) [...]

    methods (Static) [...]
        function [rotationMatrix, translationVector] =
findTransformation(referenceCoordinates, measuredCoordinates)
            % this code uses horn's method for absolute orientation and
            % uses the same variable names as used in the paper
            assert(all(size(referenceCoordinates) == size(measuredCoordinates)),'Reference
coordinates and measured coordinates need to be the same size');

            % get centroids
            measuredCentroid = mean(measuredCoordinates,2);
            referenceCentroid = mean(referenceCoordinates,2);

            % deduct the centroids from the measured points so only the
            % rotation remains (we assume no scaling as all our markers are
            % rigid and should be the same size for all cameras.
            allMeasuredCentered = bsxfun(@minus,measuredCoordinates,measuredCentroid);
            allReferenceCentered = bsxfun(@minus,referenceCoordinates,referenceCentroid);

            % Sum of all products (M = [ Sxx Sxy Sxz; Syx Syy Syz; Szx Szy Szz])
            M = zeros(3,3);
            for idx = 1:length(allMeasuredCentered)
                intermedM = allMeasuredCentered(:,idx) * allReferenceCentered(:,idx).';
                M = M + intermedM;
            end

            % the sum and difference matrix (N)
            N = [(M(1,1)+M(2,2)+M(3,3)), (M(2,3)-M(3,2)), (M(3,1)-M(1,3)), (M(1,2)-M(2,1));...
                (M(2,3)-M(3,2)),(M(1,1)-M(2,2)-M(3,3)),(M(1,2)+M(2,1)),(M(3,1)+M(1,3));...
                (M(3,1)-M(1,3)),(M(1,2)+M(2,1)),(-M(1,1)+M(2,2)-M(3,3)),(M(2,3)+M(3,2));...
                (M(1,2)-M(2,1)),(M(3,1)+M(1,3)),(M(2,3)+M(3,2)),(-M(1,1)-M(2,2)+M(3,3))];

            % the unit quaternion that maximizes q'Nq is the eigenvector
            % corresponding to the most positive eigenvalue of matrix N so
            % we find the eigenvalue and its max
            [eigenVectors, eigenValues] = eig(N);
            % handle complex numbers
            eigenValues = real(eigenValues);
```

```
            % remove all the zeros as we only have the diagonal and get a
            % vector instead of a matrix
            eigenValues(eigenValues == 0) = [];
            [~,indexMaxEigenValue] = max(eigenValues);

            quaternion = eigenVectors(:,indexMaxEigenValue);

            % avoid complex numbers, remove imaginary parts
            quaternion = real(quaternion);

            % result of the root expression can be positive or negative so
            % we use this to make it so that the max of our quaternion is
            % positive
            [~,indexMax] = max(abs(quaternion));
            if quaternion(indexMax) < 0
                quaternion = -quaternion;
            end

            % convert to a unit quaternion
            quaternion = quaternion ./ norm(quaternion);

            % split the quaternions to make the rotation matrix conversion
            % more readable
            q0=quaternion(1);
            qx=quaternion(2);
            qy=quaternion(3);
            qz=quaternion(4);

            % construct the rotation matrix
            rotationMatrix = [...
                (q0^2 + qx^2 - qy^2 - qz^2), (2*(qx*qy - q0*qz)), (2*(qx*qz + q0*qy));...
                (2*(qy*qx + q0*qz)), (q0^2 - qx^2 + qy^2 - qz^2), (2*(qy*qz - q0*qx));...
                (2*(qz*qx - q0*qy)), (2*(qz*qy + q0*qx)), (q0^2 - qx^2 - qy^2 + qz^2)];

            % now find the translation using the previously determined
            % centroids (they should match after rotation)
            translationVector = ...
                referenceCentroid - rotationMatrix * measuredCentroid;
        end
    end
end
```

Appendix C – Effect of photon splitting number

Forcing photon interaction to improve dose statistics in surface voxels improved the efficiency compared to no variance reduction. However splitting photons resulted in a more significant increase in efficiency when the splitting number was carefully selected.

Photon splitting is an essential variance reduction technique to increase the efficiency of DOSXYZnrc dose calculations. The parameter used to control photon splitting is *nsplit*. When nsplit is higher than 1m each photon is forced to interact nsplit times along its path. Each split photon has its weight reduced by a factor of 1/nsplit.

Efficiency (Ɛ) of Monte Carlo simulations is calculated using equation (C.1).

$$\varepsilon = \frac{1}{\sigma^2 T} \tag{C.1}$$

Where σ is the uncertainty in the voxels of interest and T is the calculation time. In Figure 30 the voxels of interest are all these with doses higher than 50% of the maximum dose. The graph clearly indicates the difference in optimal nsplit for skin and organ dose calculations.

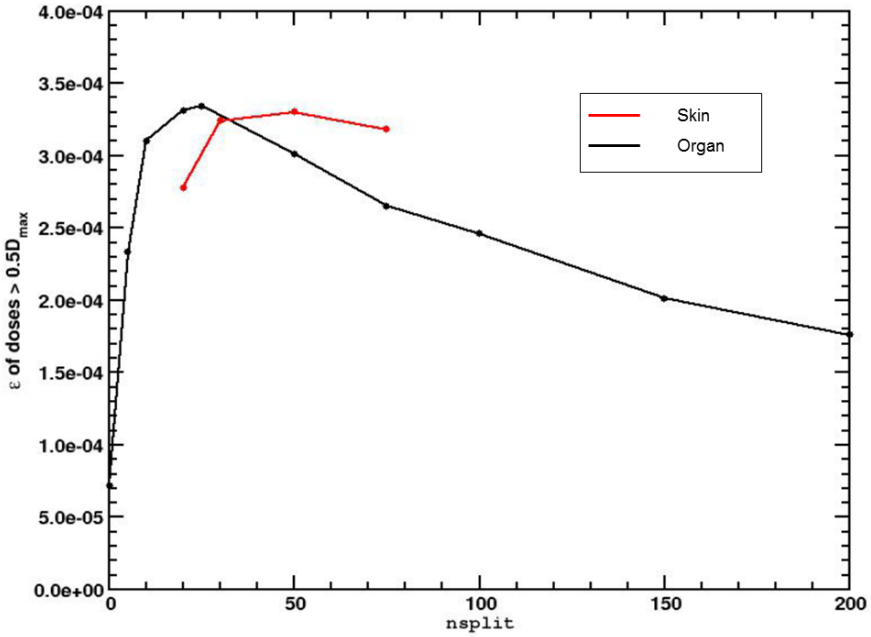

Figure 30 – Efficiency as a function of the photon splitting number in the case of 26 cm x 26 cm field on a thorax phantom for skin dose and organ dose

Appendix D – Simulation times for different field and voxel sizes

Table 11 – Calculation times for skin dose calculations with 5 voxels rejection distance on a 5 mm x 5 mm x 5 mm voxel size upper body phantom for different field sizes and uncertainties. Total time includes file input/output

Field Size	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories
5x5	2.47	6.2	3.6	160,000
	4.91	3.5	0.9	40,000
	9.63	2.8	0.2	10,000
10x10	2.56	14.9	12.4	500,000
	5.12	5.8	3.2	125,000
	10.21	3.4	0.8	31,250
15x15	2.50	36.3	33.7	1,125,000
	4.99	10.9	8.3	281,250
	9.94	4.6	2.1	70,000
20x20	2.46	65.4	62.9	2,000,000
	4.92	18.2	15.7	500,000
	9.74	6.5	3.9	125,000
30x30	2.43	157.6	155	4,500,000
	4.83	42.2	39.7	1,125,000
	9.63	12.3	9.7	281,250

Table 12 – Calculation times for skin dose calculations with 9 voxels rejection distance on a 3 mm x 3 mm x 3 mm voxel size upper body phantom for different field sizes and uncertainties. Total time includes file input/output

Field Size	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories
5x5	4.92	9.1	3.3	135,000
	9.84	6.8	0.8	33,750
10x10	4.96	18.5	12.4	480,000
	9.82	9.2	3.2	120,000
15x15	5.01	34.5	28.4	990,000
	9.95	13.0	7.1	247,500
20x20	4.93	62.4	56.4	1,760,000
	9.83	19.8	13.7	440,000
30x30	5.00	115.4	109.4	3,780,000
	9.92	41.9	36.0	945,000

Table 13 – Calculation times for organ dose calculations on a 5 mm x 5 mm x 5 mm voxel size upper body phantom for different field sizes and uncertainties. Total time includes file input/output

Field Size	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories
5x5	2.50	7.8	5.2	92,000
	5.00	4.2	1.5	23,000
	9.84	2.9	0.3	5,750
10x10	2.45	20.3	17.8	300,000
	4.98	7.2	4.6	75,000
	9.99	3.7	1.2	18,750
15x15	2.35	43.2	40.7	675,000
	5.00	13	10.4	168,750
	9.35	5.2	2.6	43,000
20x20	2.37	74.2	71.6	1,200,000
	4.70	20.5	18	300,000
	9.28	7	4.5	75,000

Table 14 – Calculation times for organ dose calculations on a 3 mm x 3 mm x 3 mm voxel size upper body phantom for different field sizes and uncertainties. Total time includes file input/output

Field Size	Uncertainty (%)	Total time (s)	CPU time (s)	Number of histories
5x5	5.22	9.9	4.1	60,000
	9.35	7.2	1.3	18,000
10x10	4.95	21.6	15.8	220,000
	9.89	9.7	4.0	55,000
15x15	4.76	43.7	37.9	495,000
	9.34	15.6	9.6	123,750
20x20	4.66	73.9	67.8	880,000
	9.21	21.5	15.8	220,000
30x30	4.72	153.9	148.1	1,980,000
	8.99	43.7	37.9	495,000