PHASM201 Masters Project Final Report

Modelling the Clatterbridge Proton Therapy Beamline with Geant4

Department of Physics & Astromony

University College London

Jordan Silverman

supervised by

Dr. Simon Jolly

Prof. Ruben Saakyan

25th September 2017 - 26th March 2018

UCL DEPARTMENT OF PHYSICS AND ASTRONOMY

Submission of coursework for Physics and Astronomy course PHASM201/PHAS3400

Please sign, date and return this form with your coursework by the **specified deadline** to the Departmental Office.

DECLARATION OF OWNERSHIP

I confirm that I have read and understood the guidelines on plagiarism, that I understand the meaning of plagiarism and that I may be penalised for submitting work that has been plagiarised.

I confirm that all work will also be submitted electronically and that this can be checked using the JISC detection service, Turnitin®.

I understand that the work cannot be assessed unless both hard copy and electronic versions of the work are handed in.

I declare that all material presented in the accompanying work is entirely my own work except where explicitly and individually indicated and that all sources used in its preparation and all quotations are clearly cited.

Should this statement prove to be untrue, I recognise the right of the Board of Examiners to recommend what action should be taken in line with UCL's regulations.

Signed Jordan Silverman

Print Name JORDAN SILVERMAN

Dated 26th MARCH 2018

Title	Date Received	Examiner	Examiner's Signature	MARK

University College London, Gower Street, London WC1E 6BT
Tel: +44 (0)20 7679 2000
email@ucl.ac.uk
www.ucl.ac.uk

Abstract: A project is currently underway to aid in the improvement of a general purpose simulation of the Clatterbridge Cancer Center's Proton Beam Therapy (PBT) beamline. The project summarised within this document aimed to improve the usability this simulation by adapting it to allow for a beamline geometry to be imported from an external file created with a Computer Aided Design (CAD) package. While previous iterations of the beamline behave as expected (although with a slightly higher output energy than expected of 60.1 MeV verses the expected 60.0 MeV), the simulation did not behave as expected after being adapted for use with imported geometries. Reasons for and corrections to this behaviour are suggested.

Contents

1 Introduction

In 2016, 28.6% of all deaths recorded in the UK by the Office for national Statistics were caused by Cancer and Cancer related issues [1]. With a reported age standardised rate of 1078.9 per 100000, about 1% of the population was diagnosed with an incidence of cancer or received a malignant neoplasm related diagnosis [2]. In response to the continuously high prevalence of deaths caused by or related to cancer, a substantial amount of money is spent on cancer research and treatment each year by the UK government.

Cancer treatment aims to completely remove or destroy cancerous cells from a patient whilst reducing the negative or adverse effects to the rest of their body. Treatments currently include, but are not limited to, surgery, chemotherapy, and radiotherapy, and at present most successful treatment plans consist of some combination of all three [3].

1.1 Surgery

There are two main types of surgical biopsy in the treatment of cancer. Incisional biopsy aims to remove a portion of the suspicious tissue for means of diagnosis, and Excisional biopsy aims to remove the diagnosed and surrounding tissues completely from the body [4]. While Excisional biopsy is often the first treatment type considered for easy to remove cancers, it has a few drawbacks. For example, if a cancer is located within sensitive tissue, such as around the brain or spine, then surgery may be too damaging to be a feasible treatment. Also, recent studies have shown that surgical tumour removal can alter the growth of Minimal Residual Disease (i.e. the growth of residual malignant cells), leading to perioperative tumour growth [5]. Furthermore, if the cancer is aggressively spreading or has spread over a wide area, surgery may not be a feasible treatment due to the possibility of not removing all cancerous tissues. In scenarios such as this, a more non local treatment may be advised, such

as Chemotherapy.

1.2 Chemotherapy

Chemotherapy is a treatment whereby the patient is given doses of cytotoxic drugs that aim to target and kill cancerous cells [6]. While this is a relatively non-invasive treatment, the cytotoxic drugs have many adverse side effects. A balance between the likely benefits of the treatment and the acceptable level of toxicity of the drugs must be tailored to the patient during treatment, as the toxicity of the drugs can also adversely affect healthy cells. These negative effects can be minimised by applying small doses in a process similar to dose fractionation (discussed in section 1.3), but, due to the non-local nature of the treatment (i.e. because the drugs are metabolised [7] across the entire body), the side affects cannot be removed altogether. Furthermore, cancer cells have been shown to develop resistance to chemotherapy [8]. In the scenario that the cancerous tissues become immune to chemotherapy, a radiotherapy treatment may be advised.

1.3 Radiotherapy

Radiotherapy typically consists of bombarding the tumour and surrounding cells with ionising radiation (for example, X-ray photons) [9]. Generally, these X-rays will be somewhere between 1-25 MeV (megavoltage X-rays, for treating the majority of deep seated cancers), although skin cancers have been treated with *Orthovoltage X-rays*, with energies between 100-500 keV, since the 1920's [10, 11]. X-rays travelling through tissue occasionally interact with the matter forming the tissue. These interactions can come in the following ways.

1. Compton scattering whereby the X-rays interact and impart momentum to unbound electrons in the matter;

2. The Photoelectric effect, whereby bound electrons are given enough energy after absorbing a photon to leave their bound state around an atom;

3. Pair production, where the photon interacts with the electric field of the atoms or electrons in the matter to create a matter anti-matter pair of particles.

These interactions can cause *ionising events*, where the matter is stripped of its electrons. On top of the fact that ionised DNA can split [12, 13], these ionising events create free radicals, which themselves can cause DNA to further split up [14]. As DNA becomes more damaged, apoptosis is more likely to be triggered, even in cancerous cells [15]. This process does not discriminate between healthy and cancerous cells, and while the cell has a high chance of being killed if the DNA within it is ionised, there is a chance that instead a healthy cell undergoes carcinogenesis [16]. Therefore, it is preferable to minimise the dose to healthy cells, while still delivering a high enough dose to effectively treat the tumour or cancerous cells.

One method by which this can be achieved is a process called dose fractionation. If a total radiation dose is fractionated into smaller doses, applied then separated by periods of rest, fewer healthy cells die. This is because healthy cells have more active self-repair mechanisms than cancerous cells which repair the damage to their DNA in these rest periods [17]. Modern fractionation treatment plans come in two forms; Hypofractionation, where higher doses are applied in fewer visits to combat the effects of accelerated or otherwise rapid tumour growth often occurring during the latter stages of a standard radiotherapy treatment [18], and Hyperfractionation (or superfractionation), where a typical total dose is divided up and treatments of the smaller doses are applied. Treatments are applied more than once a day for the same period of time as a typical radiotherapy plan [19].

Photon beams deposit ionising energy along their path. Due to the profile of the photons dose deposition curve, seen in figure 1, cells both before and after the tumour along the path of the photon will be damaged. While this is relatively

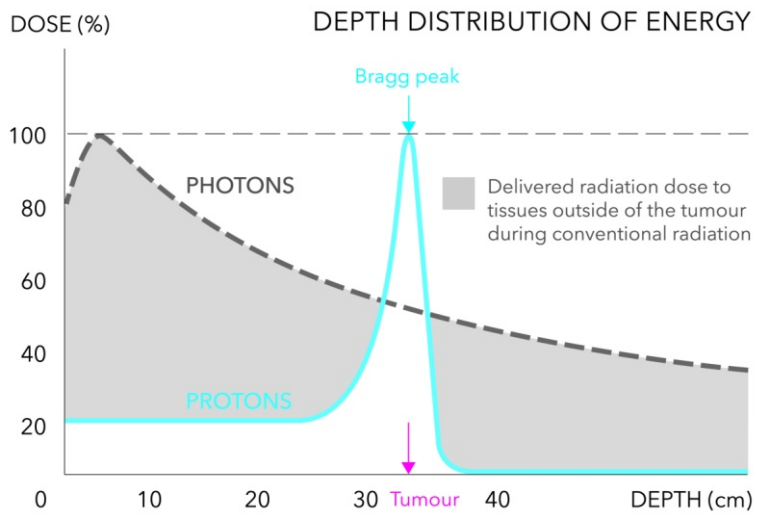


Figure 1: Dose deposition curves comparing photon (black, dashed) energy dose vs proton (blue) energy dose over depth. The proton curve is known as a Bragg Curve [20]

unimportant for cancerous cells located in regions of the body that do not house other sensitive tissues or vital organs, it is a serious complication when the cells are located within parts of the body that do. For example, cancers located deep within the brain of infants treated with photon radiotherapy can cause damage to the infants brain, causing growth complications and other severe events in survivors. For more information, see appendix B.1.

In order to minimise these side effects, modern methods of application utilise techniques such as Intensity Modulated Radiation Therapy (IMRT), which use 3D CT or MRI scans in combination with a multi-leaf collimator to create a dose plan from multiple points of entry [21]. This spreads the dose not incident on the tumour over a greater volume, and therefore decreases the maximum dose not incident on the tumour when compared to a treatment using only a single point of entry. However, as seen in figures 1 and 2, these techniques do not remove the dose to the tissue surrounding the cancerous cells, only decrease it. Therefore, when a tumour is situated in a particularly sensitive or delicate area of the body, a treatment plan comprising of proton beam therapy may be recommended.

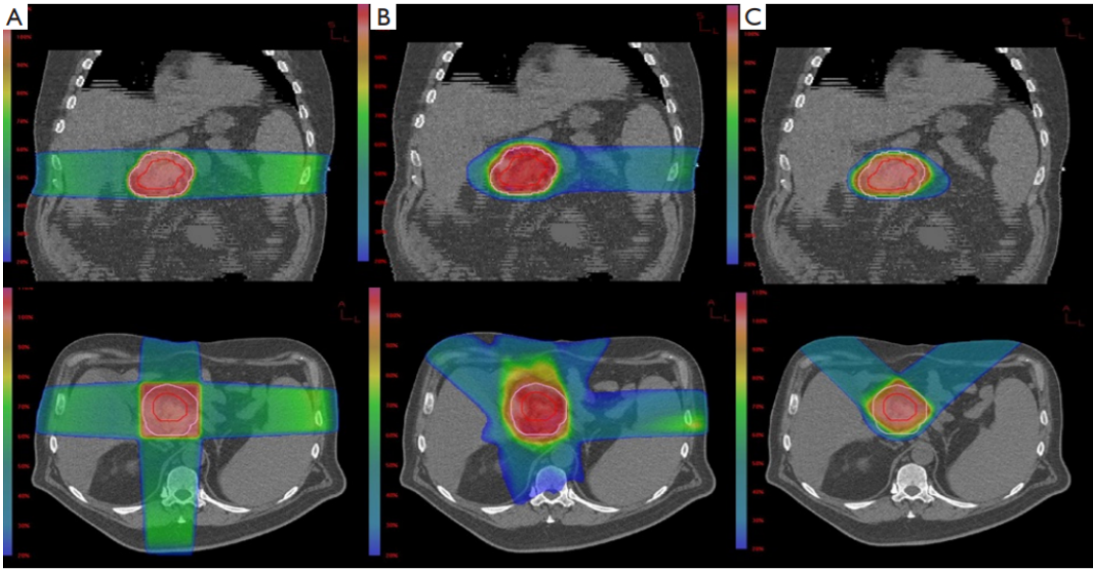


Figure 2: Transverse and coronal images of the (A) Conventional Radiotherapy; (B) IMRT; and (C) Proton Beam Therapy. [22] Conventional and Intensity Modulated Radiotherapy are discussed in section 1.3, and Proton Beam Therapy is discussed in section 2

2 Proton Beam Therapy and Theory

The capabilities for use of ionising particles as medical treatments was first postulated by Robert Wilson in his 1946 paper "Radiological use of fast protons" [23]. This set the groundwork for what has become known as Proton Beam Therapy (PBT). The first treatments were performed in laboratories in the 1950s, and eventually became implemented in hospitals in 1989 with the opening of the Clatterbridge Centre for Oncology in the UK [11]. Proton therapy is a particularly attractive treatment option for cancers located in sensitive or hard to reach areas of the body due to the nature of interactions between protons and matter.

2.1 Proton Interactions

Compared to the interactions of X-rays with matter, protons interact with matter very differently. The regime of interactions that occur is dependent on the energy of the incident proton; at high energies, pair production dominates the methods by which protons lose energy, whereas at low energies the photon electric effect

dominates, and between these two regimes Compton scattering is the most likely interaction causing energy loss. At very high energies, protons can interact with nuclei via processes described by quantum chromodynamics. These processes are accurately described as many body inelastic scattering interactions. However, the energies required to reach these types of interactions are much higher than those used in PBT. For theraputic energies, protons can lose energy to matter via Coulomb interactions whereby energy is directly given to electrons and charged nuclei. These interactions are described in equation 2.1 by the Bethe formula.

2.2 The Bethe Formula

The mean rate of energy loss of moderately relativistic, charged, heavy particles is given by the Bethe Equation [24]

$$\left\langle -\frac{dE}{dx} \right\rangle = K z^2 \frac{Z}{A} \frac{1}{\beta^2} \left[\frac{1}{2} \ln \left(\frac{2 m_e c^2 \beta^2 \gamma^2 W_{max}}{I^2} \right) - \beta^2 - \frac{\delta(\beta\gamma)}{2} \right] \tag{2.1a}$$

where E is the energy of the particle, x is the depth into the target, z is the charge number of the incident particle, Z is the atomic number of the absorber, A is the atomic mass of the absorber, c is the speed of light, β is the particles velocity (v) as a fraction of the speed of light ($\beta = \frac{v}{c}$), m_e is the electron rest mass, γ is the Lorentz factor ($\gamma = \frac{1}{\sqrt{1-\beta^2}}$), I is the mean excitation potential, and with the K coefficient defined as

$$K = 4\pi N_A r_e^2 m_e c^2 \tag{2.1b}$$

where N_A is Avagadro's number, r_e is the classical electron radius. The maximum energy transfer in a single collision for a particle with mass M, W_{max} is defined as

$$W_{max} = \frac{2 m_e c^2 \beta^2 \gamma^2}{1 + 2\gamma m_e / M + (m_e/M)^2} \tag{2.1c}$$

and where the density effect correction to ionisation energy loss, $\delta(\beta\gamma)$, computed from Sternheimer's parameterisation [25], is defined as

$$\delta(\beta\gamma) = \begin{cases} 2(\ln 10)x - \bar{C} & \text{if } x \geq x_1 \\ 2(\ln 10)x - \bar{C} + a(x_1 - x)^k & \text{if } x_0 \leq x < x_0 \\ 0 & \text{if } x < x_0 \text{ (for nonconductors)} \\ \delta_0 10^{2(x-x_0)} & \text{if } x < x_0 \text{ (for conductors)} \end{cases} \tag{2.1d}$$

Here, the value $x = \log_{10}\eta = \log_{10}(p/Mc)$ (where p is the momentum of the particle), and k is the bremsstrahlung photon energy. The quantities a, x_0, and x_1 are constants which must be evaluated for each material [26] and $\bar{C}$ is obtained by equating the high energy case of (2.1d) with the limit given in (2.1e) [24].

$$\delta/2 \to ln(\hbar\omega_p/I) + ln(\beta\gamma) - 1/2 \tag{2.1e}$$

where $\hbar\omega_p$ is defined as the plasma energy

$$\hbar\omega_p = \sqrt{4\pi N_e r_e^3} m_e c^2/\alpha \tag{2.1f}$$

where N_e is the electron density and α is the fine structure constant.

2.3 The Bragg Peak

Equation 2.1 shows a dependence of the magnitude mean energy loss per depth on the inverse of the square of the velocity. This gives rise to the characteristic Bragg curve profile (shown in figure 1) because at lower velocities, such as at the end of the path of the particle, the particle loses all of its energy. Furthermore, as the right hand side of equation 2.1 only depends on the energy of the incoming particle and properties of the material the particle is travelling through, it is possible to select

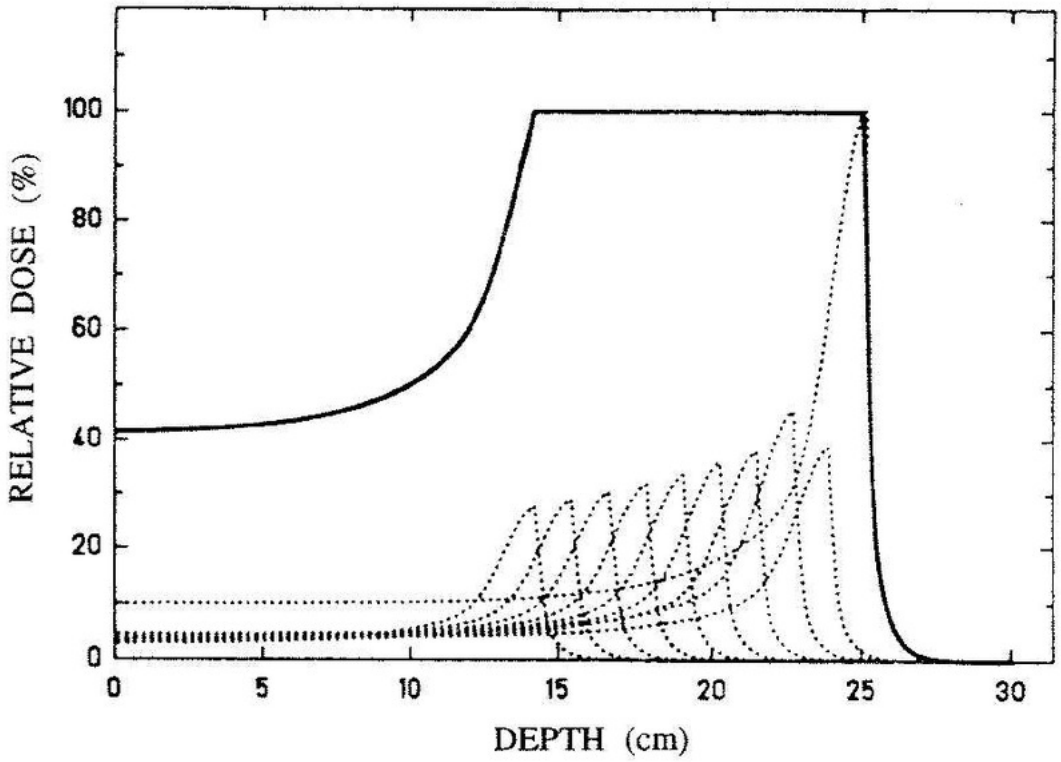


Figure 3: A spread-out bragg peak formed of many bragg curves.[27]

a depth of energy deposition (i.e. the depth of the maxima of the Bragg curve) by preparing the particle with a specific energy.

This allows for a proton dose deposition much more localised than that possible to attain with conventional X-ray based radiotherapy. If several proton beams are selected with different energies, and superimposed, a "spread-out Bragg peak" (as shown in figure 3) can be achieved. This allows for a maximal dose deposition within a specified volume, corresponding to the energies of the selected proton beams. By choosing these energies to match the depth and width of a tumour, PBT can be personalised to the patient. It is therefore because of the inherently precise nature of proton beam dose deposition that the longer term negative effects associated with conventional radiotherapy can be avoided or reduced.

2.4 Advantages and Disadvantages of Proton Beam Therapy

The major advantage of PBT over conventional radiotherapy is due to its ability to localise a dose to a specific volume. This however is a double edged sword, due to the fact that PBT relies on the ability to correctly specify the depth of a dose. If there is uncertainty in this, then the dose will be applied very precisely to an incorrect tissue volume, and as such the treatment will fail. Therefore, reducing the uncertainty in dose depth is a task of high priority when considering PBT facilities. One tool by which this can be achieved is through use of Monte-Carlo based simulations to trial different beam strengths and dose distributions, and GEANT4 is an example a simulation tool suited to this purpose.

3 GEANT4

GEANT4 is a simulation package that utilises Monte-Carlo methods to model the passage of individual particles through a defined geometry. Written in C++, the toolkit is used in many different fields of physics such as High Energy Physics (HEP), Medical physics, accelerator physics, astrophysics, and radiation protection [28]. It is used to model many parts of the LHC beamline and its detectors, such as ATLAS [29], CMS [30], ALICE [31] and GAUSS (LHCb) [32], to study their beam characteristics and for use with proposals potential changes to the detectors. This section introduces GEANT4 and describes its functionality[1].

3.1 How GEANT4 carries out a simulation

A simulation starts by instantiating a *run manager* (represented by the `G4RunManager` class). This handles all the processes required by a *run*, described by an instance of the `G4Run` class. The run manager performs a number of prerequisite tasks;

[1] Although work has recently started in migrating over to versions 10.3 (and more recently 10.4), this project uses version 10.2.1

1. First, the *geometry* of the system is generated by instantiating the `G4UserDetectorGenerator` class. Contrary to its name, this class does not just define the detector, but also the beamline, as well as defining their prequisites (i.e. the materials used in both).

2. Then the run manager initialises the `G4UserPhysicsList` class (see section 3.4 for more details). This class represents the physics and *physical processes* to be utilised when an interaction occurs during the simulation.

3. The `G4UserPhysicsList` class creates definitions of *particles* (i.e. their mass, lifetimes, charges etc.) by instantiating a `G4ParticleDefinition` class, and physical processes (cross sections, energy losses under interactions etc.) by instantiating a `G4Process` class.

A process can describe a number of different things depending on the type of interaction that it is handling. As such, it can take on one or more of the following actions [33]:

Along step An `AlongStepDoIt()` function that describes continuously changing processes during the step;

Post step A `PostStepDoIt()` function called after the step has been completed.

At rest An `AtRestDoIt()` function called if the particle has zero kinetic energy;

Each of these `DoIt` methods have a corresponding `GetPhysicalInteractionLength` (GPIL) method that are used to indicate the step length to be taken (this is put into context during discussion of the stepping algorithm later in this section). Many processes can be called along step, as these processes are continuous and continuous processes can be handles simultaneously (i.e. a particle can lose energy and produce

a secondary particle at the same time). However, only one process can be handled post step due to the discrete nature of post step processes (i.e. a particle can only scatter either elastically or inelastically).

After this, the run manager uses the `G4UserPrimaryGeneratorAction` class to generate the *primary particles* (or primaries). This defines initial properties of the particles such as their spacial and angular distribution at the source, their energy distribution and their direction of travel.

Now that the prerequisites have been completed, the run manager instantiates a `G4Run` object. A run consists of a predefined number of *events* controlled by a `G4EventManager` object, and where each event is described by an instance of the `G4Event` class. Each event[2] contains a `G4PrimaryVertex` object describing the spacial and temporal coordinates of a primary, and a `G4PrimaryParticle` object defining the particles elementary characteristics (such as charge, rest mass etc.).

The event manager handles events sequentially until the primary the local event is describing is "killed". A primary is "killed" if it exits the world volume, comes to rest (i.e. its kinetic energy is set within a threshold of zero), or undergoes a decay process.

3.2 GEANT4s Stepping Algorithm

The properties of a particle are accessed by a *track* described by an instance of the `G4Track` class. This stores useful information about the particle (i.e. its position, definition, the ID of its parent particle etc.) as well as hosting the particles specific `G4Step` object. The *step* is the smallest unit of simulation in GEANT4, and it is managed by the `G4SteppingManager`. The `Stepping()` method of this class updates and steers the stepping of each particle. The algorithm used to carry out a single step is given below [34].

[2]It should be noted that in general, any predefined number of particles can be defined by an event, but in this simulation a single event corresponds to a single primary.

1. If the particle is stationary (i.e. it has zero kinetic energy), each active `AtRest` process calls its GPIL method and proposes a step length corresponding to its described interaction. The process that returned the shortest step length is invoked.

2. If the particle has kinetic energy above zero, each active process calls its GPIL method and proposes a step length, and the shortest of these step lengths is taken.

3. The distance to the next boundary is calculated by the geometry navigator, and compared to the minimum step length from the processes. If this distance is larger than minimum step length, this minimum step length is selected to be used as the next step length. Given this condition is met, methods calculating distances to further boundaries are halted for the rest of the step (as no more boundaries are going to be crossed this step).

4. If the minimum step length given by the GPIL methods is longer than distance to the next boundary, the distance to the next boundary is re-calculated.

5. The shorter of the GPIL calculated minimum step length and the distance to the nearest boundary is taken.

6. All active continuous processes are performed. After all invoked processes are completed, the particle's kinetic energy is updated. The change to the particles kinetic energy is calculated by summing the contributions from each process.

7. Prior to any discrete processes being invoked, the particles properties (i.e. kinetic energy, position) are updated. While doing this, any secondary particles created by the processes are calculated and stored.

8. If the kinetic energy of the particle has been updated to zero by a continuous process, any `AtRest` processes will be applied at the next step (if applicable).

9. The particle specific discrete process is called by the `PostStepDoIt()` method. Afterwards, the energy, position and time of the particle are updated, and, as with the continuous processes, any secondaries generated are stored.

10. The track is checked to see if it has been terminated by the discrete process.

11. The distance to the nearest boundary is recalculated and updated.

12. If the step was confined by a volume boundary, the particle is pushed into the next volume. I.e. if the particle hits a volume boundary, push the particle into the next volume.

13. Information regarding scorer hits is processed.

14. The user defined `G4UserSteppingAction` class is invoked to perform tasks unique to the specific simulation.

15. Data is saved to a `G4Trajectory` object used for visualisation purposes (see section 3.6).

16. The mean free paths of each the of discrete processes are updated.

17. If the parent particle is still alive, the maximum interaction length of the discrete process called by the `PostStepDoIt()` method is reset.

18. The step terminates.

`G4Track` stores the current local values relating to the information of the particle, such as its momentum, position and energy. It also stores the static local information of the particle such as its mass and charge. It is updated after all `AlongStepDoIt` methods have completed, and after each `PostStepDoIt` has finished. The `G4Step` object stores the current local values relating to each step, such as the coordinates of the start and the end of the step. It also stores the change in track values between these

coordinates, and these change in property variables are updated by the processes instantiated along it.

3.3 Macros

GEANT4 can be run in two different modes; *interactive mode* and *batch mode*. In interactive mode, user defined settings are entered into a GUI. This can be done using various interfaces such as `G4UIterminal` and `G4UIQt`. To include functionality for these interfaces, their respective header file (`G4UIxxx.hh` where `xxx` = `terminal`, `Qt` etc.) needs to be included in the simulation. However, this project used the batch mode to specify some inputs.

In batch mode, macro files (such as `proton.mac` required to run the simulation or `visualisation.mac` used to create visual renderings of the simulation) are created that specify commands for the simulation. These files are read in at the beginning of the simulation and they set respective parameters specified within them.

3.4 Simulated Interactions and the particle source

A physics list (discussed briefly in section 3.1) defines the particles and physics used in a simulation. These can be customised to meet the needs of the user with the macro command `/protonB/phy/addPhysics` from the user defined `AddPhysicsList()` method of the `PhysicsList` class.

The physics list is questioned whenever an interaction takes place. However, when the simulation is started, it is the `G4VUserPrimaryGeneratorAction` that defines the particles. This simulation used a `G4GeneralParticleSource` and its characteristics were controlled via macros. The position of the source was defined by placing creating a cylindrical volume in the `DetectorConstuctor` class and placing it at coordinates such that it coincided with the start of the beamline. The source was then confined to this volume by using the `ConfineSourceToVolume()` method

in the `PrimaryGenerator` class. This was done to enforce a normally distributed beam profile at the source.

3.5 Defining Geometry and Materials

The geometry of the beamline, and the material components of this geometry can be created directly using GEANT4 within the `G4VUserDetectorConstruction` class. Materials are defined within the user defined `DefinedMaterials()`. To define a material, its constituent elements much first be defined using a `G4Element` object. This is done by specifying a label, its atomic and, its molar mass. The material, defined using a `G4Material` object, is then created by first specifying its label, density and charge, and then by specifying its contents using either a fractional mass method or by directly stating its atomic composition. Materials can also be instantiated by creating a `G4Material` directly from the NIST database (probed using an instance of the `G4NistManager` class).

After the materials have been defined, the geometry can be defined component-wise within the `ConstructVolume()` method.Each component volume is specified by three layers;

1. A `G4VSolid` that defines the shape (and its dimensions) that the component will be constructed from, such as a tube or a box;

2. A `G4LogicalVolume` that gives the volume its material;

3. A `G4PhysicalVolume` that assigns the logical volume to the solid volume and assigns the component within a mother component.

There are many types of `G4VSolid`, such as `G4Box` that creates a rectangular parallelepiped, or `G4Tubs` that defines a cylinder. Solids can be subtracted from one another do define 'holes', which allows definitions of slightly more complex structures

such as hollow tubes or boxes, appropriate for defining components of the beamline. The logical volumes can be assigned colour attributes with a `G4VisAttributes` pointer. For example, the source volume is set to be invisible in the simulation:

```
G4VisAttributes* invisibleVisAtt = new G4VisAttributes( false );
...
lSource->SetVisAttributes( invisibleVisAtt );
```

such that when displayed the visualisation can be more easily understood.

3.6 Outputs and Visualisations

GEANT4 can visualise the simulations it produces. There are a few methods to create these simulations, and a few different renderers to use from. This simulation uses a macro, `visualisation.mac`, to specify all the settings for these visualisation. This macro is run from the `proton.mac` macro with the command `/control/execute visualisation.mac`.

The visualisation macro switches on rendering with the `/vis/open DAWNFILE` command. This creates a `.prim` file that can be opened with the DAWN renderer. DAWN is very suited to drawing visualisations with large amounts of detail, to the level of rendering each segment of a track when rendering the trajectories of particles. However, DAWN is a relatively deprecated visualiser, with its last major update being rolled out in 2010 [35]. As such, its user interface is relatively clunky, with visualisation parameters having to be set before a render is produced, meaning that to rotate a render of the simulation the current render must be closed, the visualisation parameters changed, and the renderer then must be run again.

More user friendly renderers are available, and an example of one such visualiser is `OpenGL`. This visualiser has a click-and-drag interface, allowing renders to be easily and more intuitively manipulated. This renderer was not used for the main simulation, however, a proof of concept was developed to show that GDML files (discussed

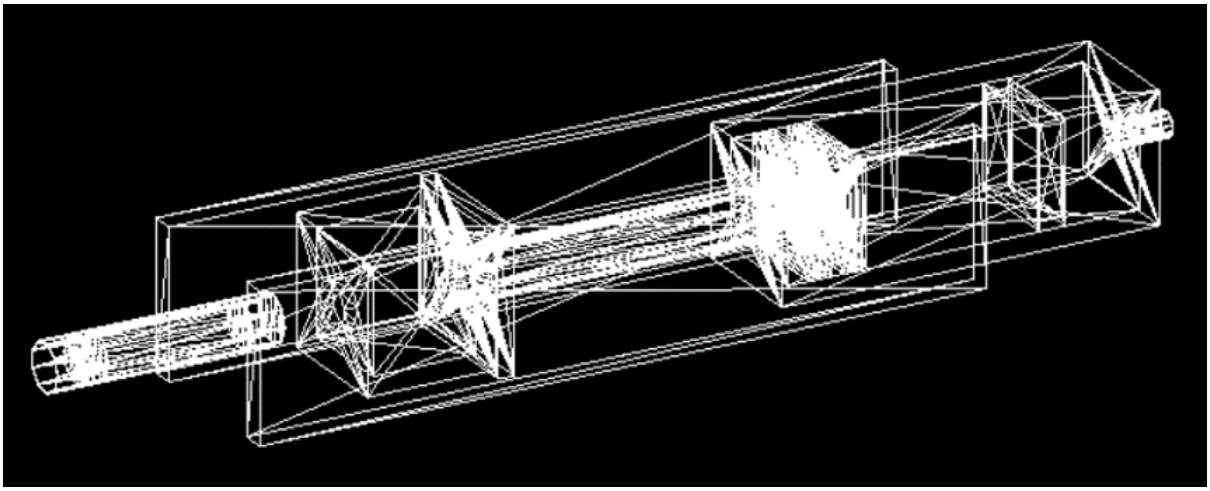


Figure 4: Mesh view of the beamline after being imported into GEANT4. This render was created using `OpenGL`. One feature of this figure that should be noted is the truangular facet structure of the GDML files, having been created by conversion from STL files, can be seen. This is discussed in section 5 in detail.

in section 5.2) could be imported into a GEANT4 simulation. The render produced is shown in figure 4

4 The Clatterbridge model

The Clatterbridge Cancer eye treatment facility is the oldest hospital based proton beam therapy centre in the world [36], and began treating patients in 1989 [37, 38]. It is also currently the only active proton therapy centre in the UK, and as such there is a high demand for its medical use. This means that finding time to collect valuable data from the beamline *in situ* is a difficult task. Therefore, it is valuable to have a model of the Clatterbridge beamline that can be used to gather data and perform experiments with in the stead visiting the centre itself. One of the main aims of this project was to update current simulations to make them more easily run and modifiable by the layman. This project aimed to achieve this goal by creating a protocol by which new, or improved beamlines could be added to the simulation; however, it is still necessary to understand how other parts of the simulation are implemented to understand the protocol. This section described the parts of the

simulation necessary to understand, before moving on to how the simulation was modified in section 5.

4.1 Partial Geometry definition

The geometry of the beamline is partially defined within the simulation. This is done within the `DetectorConstruction` class (shown in appendix 7.2). Part of creating the geometry of the beamline consists of defining the detector, and this can in a number of ways by specifying certain characteristics in the Macro settings as discussed in section 3.3. First, the material that the detector is made from needs to be defined. In the following example, the detector is chosen to be made of polyvinyltoluene (PVT), a commonly used plastic scintillator [39]. To define PVT, first, the elements that it comprises of need to be defined.

```
G4Element* H = new G4Element("Hydrogen", "H", z=1, a=1.008 * g/mole);
G4Element* C = new G4Element("Carbon"   , "C", z=6, a=12.01 * g/mole);
```

Next, the material must have its density and number of components defined, and then can be constructed from its constituent elements (or other, previously defined materials) by specifying either their fractional mass or the number of atoms of each element as shown below:

```
fScintillatorPVT = new G4Material("Scintillator_PVT",
                                  density=1.023 * g/cm3,
                                  ncomponents=2);
fScintillatorPVT->AddElement(C, natoms=9  );
fScintillatorPVT->AddElement(H, natoms=10 );
```

Is it also possible to pull an element or material from the NIST database, however none were used in this simulation.

Once the materials used for the detector have been defined, it is possible to define its geometry. The geometry of the beamline that is described using GEANT4 contains the Detector and the particle source. The source is described using the

`G4Tubs` solid, set to contain a vacuum in its volume, and given its position of z = −8400 mm by the following lines;

```
sSource = new G4Tubs( "Source", 0., 30*CLHEP::mm, 1*CLHEP::mm/2, 0, 2.*
    CLHEP::pi );
lSource = new G4LogicalVolume( sSource, fVacuum, "Source" );
pSource = new G4PVPlacement( 0, G4ThreeVector(0., 0., -4200*CLHEP::mm),
     lSource,
                              "Source", lWorld, false, 0, checkOverlaps );
```

Recalling that solid is defined by a `G4VSolid`, the material is set by the `G4LogicalVolume`, and placed at its coordinate location within the beamline geometry with a `G4PhysicalVolume`. It should be noted that GEANT4 uses half lengths, so instead of being placed at z = −8400 mm it is placed at z = −4200 mm. Elsewhere in the code these lengths are specified as `length`/2 for legibility purposes.

The majority of the beamline, however, is defined within an imported GDML file, discussed in section 5.

4.2 Using the simulation

To get some idea of the performance of the simulation before the GDML models were imported, the simulation was run in parallel mode with 100 sub-simulations each calculating the trajectories of 100000 primaries for a total of 10^7. The kinetic energy of the beam, as well as its lateral profile and emittance, was measured at different points along the beamline. While this simulation was run with a high enough amount of particles for its results to be statistically significant, the main purpose of the run to get a general understanding of how the beam evolved through the geometry for comparisons to the simulation that imported its geometry from GDML files. This comparison is done in the results section. The geometry of the beamline and the terminology used in this section are discussed and explained fully in section 3.5.

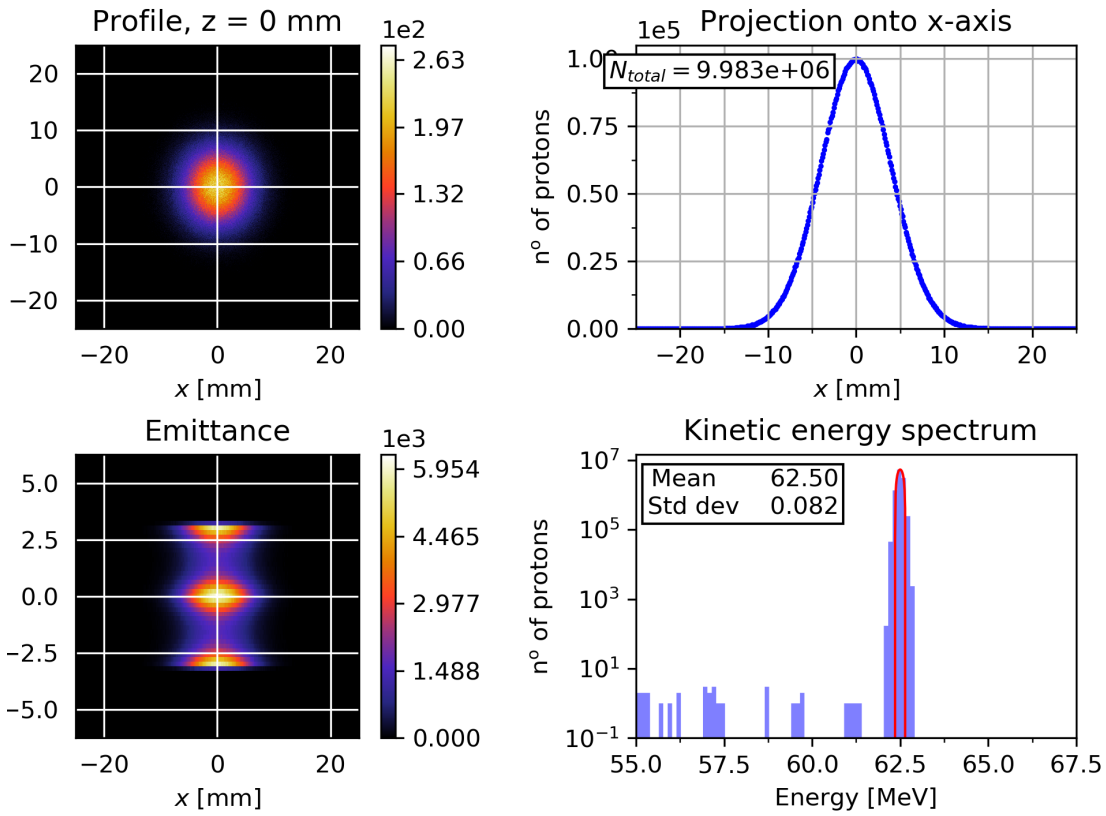


Figure 5: The beam at the source.

The first measurements of the simulation were made at the source, shown in figure 5. The number of protons at each slice are counted by integrating their projection into the x-axis. The emittance is showing slightly odd behaviour with the egg timer shaped plot, but the rest of the figures are as expected; the mean energy is 62.50 MeV with a standard deviation of 0.082. The beam is relatively spread out as it has not passed through the collimator.

The second set of measurements were made after the brass collimator, shown in figure 6. The profile of the beam has been cut down as expected by approximately 75 %, and the energy spread has increased to a standard deviation of 0.219 MeV, although the mean energy has not changed by much (0.01 MeV). The halo of particles seen in the profile plot of figure 5 has been cut away from the beam due to the particles on the peripheries of the beam hitting the brass collimator.

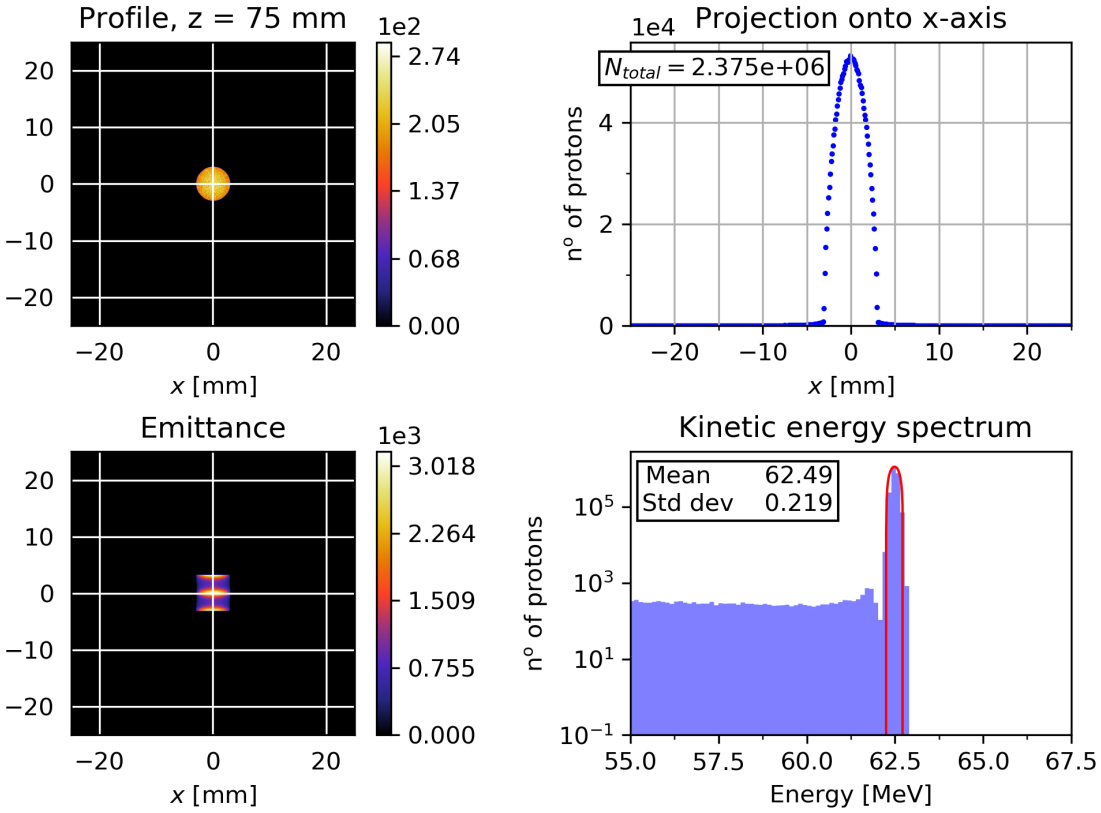


Figure 6: The beam after passing through the collimator.

The next set of measurements were made after the beam entered the first aluminium box, shown in figure 7. As the beam has passed the beam stopper, the beam has has a large number of central particles removed, as shown by the dip in the projection plot. The profile has started to disperse at this point., and the mean energy has fallen to 61.77 MeV with a standard deviation of 0.268.

The final set of measurements were made at the end of the beamline, before the brass nozzle. Even when running the simulation with 10^7 particles, not enough reached the end for a descriptive graphic to be produced..

5 CAD Models

There are two ways of defining a beamline geometry within GEANT4. The first is described in section 4.1, whereby the materials are first described, then the geometry

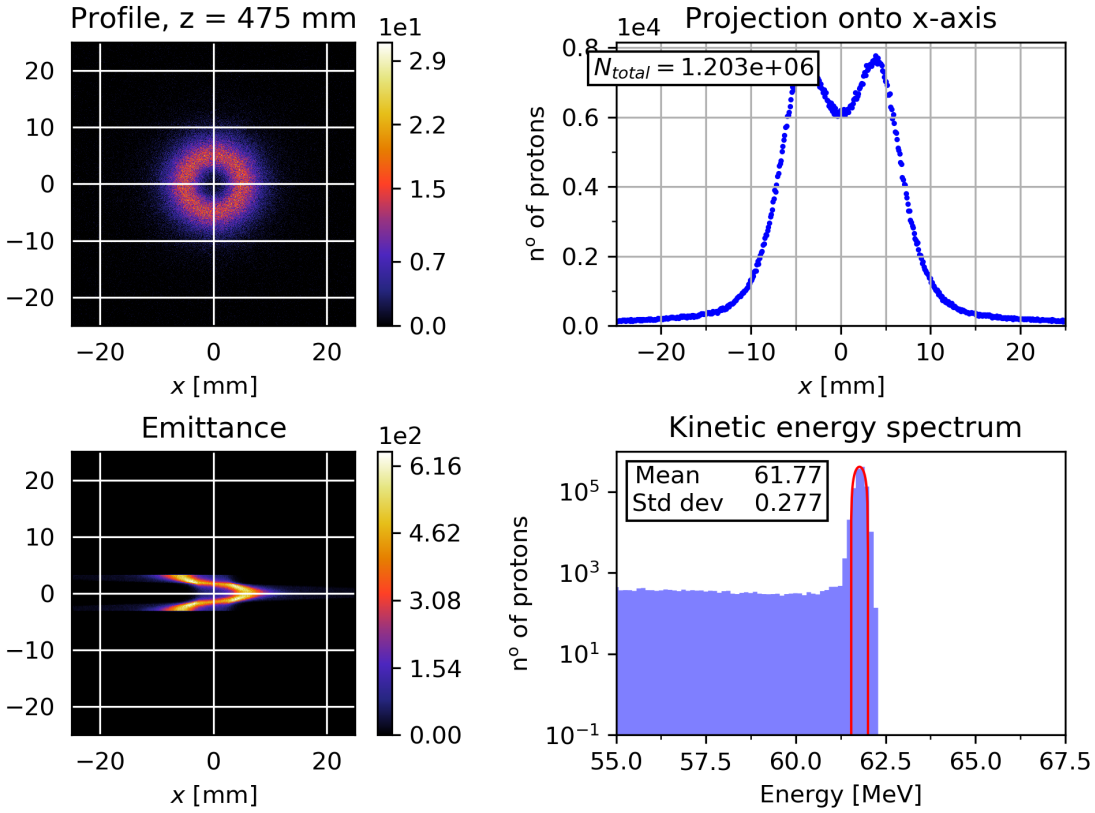


Figure 7: The beam passing into the first aluminium box.

is created out of these materials. This is all done within the `DetectorConstruction` class, and is the way that the simulation generated its geometry prior to the work of this project.

A second approach to geometry design can be implemented whereby a beamline geometry designed with a CAD software package can be imported into the simulation. Due to the complex nature of having to fully understand the inner workings of GEANT4 before being able to add to a simulation via the direct method discussed above, this importing approach makes the simulation more user friendly, and allows for faster prototyping of new or potential geometries.

A protocol was created whereby a model created in a CAD package could be converted and imported to the GEANT4 simulation. This section outlines the protocol, as well as the knowledge required to understand it.

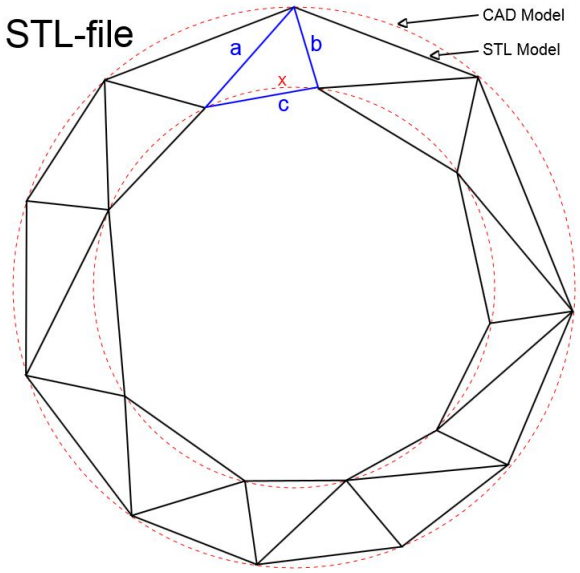


Figure 8: Top down view of a torus represented by a CAD model versus an STL model. This shows how the CAD model is approximated by a mesh of many triangles.

5.1 STL

Invented in 1987 by the Albert Consulting Group for 3D Systems [40], Stereolithography (STL) is a file-type that describes a triangulated surface by the unit normal and vertices of triangles. It describes only the surface geometry of a three-dimensional object in a very basic and rudimentary way; it does not store any other auxiliary information such as colour or scale information.

STL files come in two types; binary (STLB) and ASCII (STLA). The binary type is much more compact and so therefore more widely used, although the conversion process detailed in section 5.5 requires any STL file used to be in STLA format. The STLA format starts first by defining the name of the solid with the line

```
solid name
```

Note that here *name* is in fact optional, and if left blank the file will operate as expected (as long as a space is still left after the word solid). The file then continues by listing a number of triangles as follows:

```
facet normal n_i n_j n_k
```

```
        outer loop
            vertex v1_x v1_y v1_z
            vertex v2_x v2_y v2_z
            vertex v3_x v3_y v3_z
        endloop
    endfacet
```

where each n, v are floating point numbers in exponential format (i.e. "5.322000e+2"). The file then concludes with the line

```
    endsolid name
```

While these files are able to be generated by AutoDesk Inventor, the GEANT4 simulation requires the CAD files to be of the GDML format to be imported.

5.2 GDML

The Geometry Description Markup Language (GDML) is a file format designed to be used in describing the geometry of detectors. Created by CERN, it is an application independent format based in XML. It is structured in a way to accommodate a *mother* and *daughter* geometry tree, corresponding to a hierarchy of geometric volumes. The structure of a GDML file can be separated into five parts, each determined by an XML style block. This structure is detailed below [41, 42].

The `<define> ... </define>` block holds variable values, such as position vectors but also including ideas such as rotation and scaling matrices.

The `<materials> ... </materials>` block defines the materials used in the various parts of the geometry. Materials can be defined as basic as elements, and fractional combinations of elements can be defined to create more complex materials. Furthermore, once these more complex materials have been defined, they too can be used to create even more complex materials by combining them with other previously defined materials and elements.

The `<solids> ... </solids>` block defines all the solids in a given geometry. Specific structural descriptions are supported by the GDML schema; for example simple geometric objects such as `<box>` structures, but also more complex structural descriptions such as Tessellated solids, defined by a combination of triangular or quadrangular facets. Solids can be combined with Boolean operations such as intersection or subtraction.

The `<structure> ... </structure>` block defines the implementation of the geometry hierarchy tree. This includes references to daughter volumes defined in external GDML files. The hierarchy is determined by specifying the position of the daughter volumes within a given mother volume.

The `<setup> ... </setup>` block defines the uppermost volume (often referred to as the *world volume*). This block can be multiply defined, that is, there can be more than one setup block within a single file, allowing for trialing and experimentation on the hierarchy or component sub-parts used in the geometry without changing the GDML file.

5.3 Materials

Before materials can be defined within a GDML file, the materials components must be defined. The most simple of these components is an element, defined in an `<element> ... </element>` block. Elements can be defined in multiple ways. One way is via specifying their charge and atomic value, as shown below;

```
<element Z="1" formula="H" name"Hydrogen">
   <atom value="1" />
 </element>
```

Another is by first defining an Isotope of an element;

```
<isotope name="U235" Z="92" N="235">
   <atom type="A" value="235.01">
 </isotope>
```

```
    <isotope name="U238" Z="92" N="238">
       <atom type="A" value="235.03">
     </isotope>
```

where Z is its atomic number and N is the number of nucleons in the isotope, and then by specifying an element's isotopic fractional composition;

```
    <element name="enrichedUranium" >
       <fraction ref="U235" n="0.9" />
       <fraction ref="U238" n="0.1" />
     </element>
```

Outside of elements and isotopes (and compositions of isotopes), more complex materials can be defined. This is done in one of three different ways. The first is by declaring a material directly from an element;

```
    <material name="Copper" state="solid">
          <D value="8.960" unit="g/cm3"/>
          <fraction n="1." ref="copper"/>
    </material>
```

where the D value and unit are the variables corresponding to the density of the material. The second way to define a basic material is by specifying the number of atoms of each of its constituent elements;

```
    <material name="Water">
          <D value="1." unit="g/cm3"/>
          <composite n="2" ref="hydrogen" />
          <composite n="1" ref="oxygen" />
          <MEE unit="eV" value="78.0"/>
    </material>
```

In the example given above, the minimum excitation energy is also specified. The final way to define a basic material is by specifying its fractional composition.

```
    <material name="Kapton" state="solid">
```

```
        <D value="1.42" unit="g/cm3"/>
        <fraction n="0.0273" ref="hydrogen"/>
        <fraction n="0.7213" ref="carbon"/>
        <fraction n="0.0765" ref="nitrogen"/>
        <fraction n="0.1749" ref="oxygen"/>
    </material>
```

Furthermore, it is possible to reference other materials as components for more complex ones. While this can be done by either compositional or fractional methods described above, it is common practice to use the fractional methods. An example of a fractional approach to defining a complex material can be seen below;

```
    <material name="DC3140">
        <D value="1.2" unit="g/cm3"/>
        <fraction n="0.60" ref="dimethylsiloxane_hydroxy_terminated"/>
        <fraction n="0.30" ref="trimethylated_silica"/>
        <fraction n="0.10" ref="methyltrimethoxysilane"/>
     </material>
```

Once the required materials have been defined, the geometry of the system can be created.

5.4 Geometry

The geometry within a GDML file is built in two stages. First, the daughter components are defined, usually without a materials block, in their own GDML file. Then, the geometry hierarchy tree is constructed in the master file.

The geometry of a daughter file is created out of *solids*. These are created by creating a block respective to the type of solid being created. For example, the simplest type of solid definable is a `GDML box`, created with a `<box/>` block. A box is defined by specifying its name, side lengths, and their accompanying units;

```
<box name = "box1" x = "100" y = "200" z = "300" units = "cm"/>
```

This line would create a solid named `box1` with a volume of 6 m^3.

While there are many ways of constructing solid volumes by hand, this project used a conversion script to convert an STL file to a GDML file. As STL files contain a list of triangular facets, the simplest solid type to convert to is a tessellated solid. The way this type of solid is defined follows the following blueprint;

```
<tessellated aunit="deg" lunit="cm" name="MylarComponents-SOL">
    <triangular vertex1="MylarComponents_v0" vertex2="
    MylarComponents_v1" vertex3="MylarComponents_v2"/>
    <triangular vertex1="MylarComponents_v2" vertex2="
    MylarComponents_v1" vertex3="MylarComponents_v3"/>
    <triangular vertex1="MylarComponents_v4" vertex2="
    MylarComponents_v0" vertex3="MylarComponents_v5"/>
    ...
    </tessellated>
</solids>
```

Where each of the vertices are vectors that have been predefined in the `<define> ... </define>` block as follows;

```
<define>
    <position name="MylarComponents_v0" unit="cm" x="-0.668208" y="
    7.00343" z="-302.719"/>
</define>
```

The script also automatically creates the structure block and setup blocks so the world volume of the component GDML file is defined as follows;

```
<structure>
    <volume name="MylarComponents">
        <materialref ref="Mylar"/>
        <solidref ref="MylarComponents-SOL"/>
    </volume>
</structure>

```

```
<setup name="Default" version="1.0">
    <world ref="MylarComponents"/>
</setup>
```

where the material type given in the structure block is parsed from the STL filename by the conversion script, which is discussed in more detail in section 5.5. After creating the daughter components, the master file is created. This contains the details of all the materials used in the geometry, as well as defining the world volume. The world volume is defined as a box with sidelengths specified by the used in the conversion script.

After each component file has been defined, the master GDML file is created to bring the whole geometry together. The majority of this is done in the `<structure> ... </structure>` block of the master file, although the world solid is defined before this first. In the master GDML file used in this simulation, the world solid is defined as a box with lengths x = 9450.0mm y = 4450.0mm z = 9450.0mm in the `<solids> ... </solids>` block as follows;

```
    <solids>
        <box lunit="mm" name="world_solid" x="9450.0" y="4450.0" z="
    9450.0" />
    </solids>
```

After the world solid is defined, the component files are included in the `<structure> ... </structure>` block as follows;

```
    <structure>
        <volume name="world_volume">
            <materialref ref="Air"/>
            <solidref ref="world_solid"/>

            <physvol>
                <file name="AluminiumComponents.gdml"/>
             </physvol>
```

```
9                ...
10           </volume>
11       </structure>
```

5.4.1 The Clatterbridge Beamline Layout

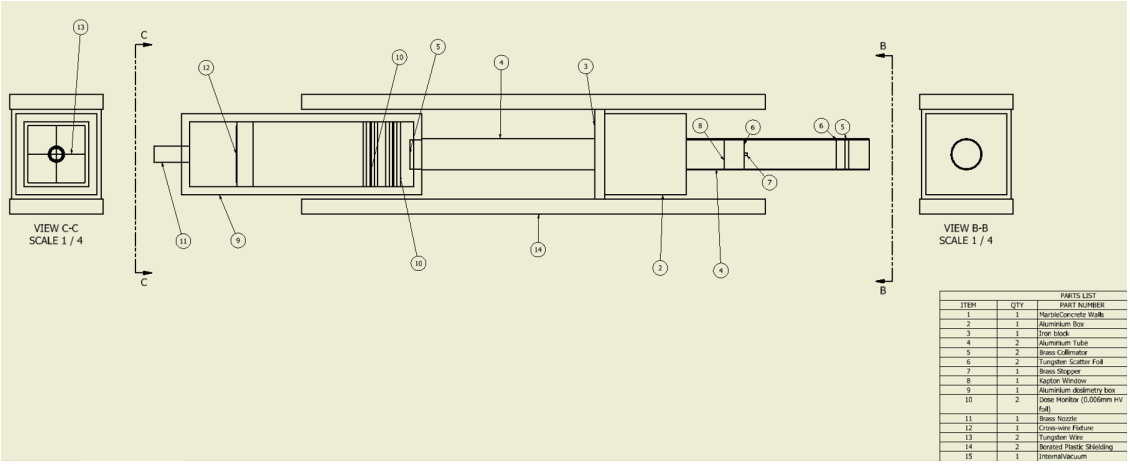


Figure 9: Schematic view of the entire beamline. The first aluminium tube and the second box housing the dose monitors have been made transparent so their internal components are visible. This schematic was created using AutoDesk Inventor's schematic functionality after the beamline had been defined.

The complete beamline is shown in figures 9, 10, and 11. The beamline consists of multiple components, detailed in the sections below. The beamline was created using AutoDesk Inventor Professional 2018. Each component was added to an assembly file, and the total beamline was converted and then imported to the GEANT4 simulation. The design of the beamline is construed in such a way as to deliver as uniform a lateral dose profile as possible to a patient (as the energy deposited in a patient should not be dependent on the relative lateral position of the beam), and as such utilises a *dual scattering* tube to passively scatter the proton beam. Borated plastic shields surround the beamline to absorb any scattered particles (i.e. secondary neutrons) and shield the rest of the treatment room. [3]

[3]It should be noted that the marble concrete room surrounding the beamline was also modeled using AutoDesk Inventor and added to the assembly file, although this is not shown in any of the images of the beam as this would lower their clarity and ultimately usefulness.

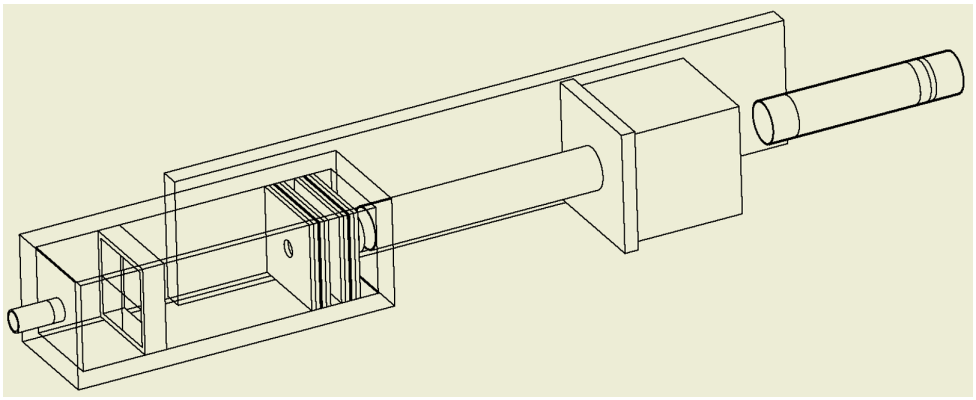


Figure 10: Side wireframe view of the entire beamline. The front facing shielding block, first aluminium tube, and the second box housing the dose monitors have been made transparent so their internal components are visible.

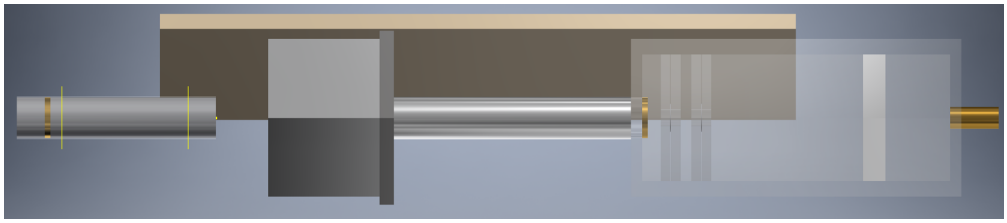


Figure 11: Side view of the entire beamline as designed in AutoDesk Inventor. The front facing shielding block, first aluminium tube, and the second box housing the dose monitors have been made transparent so their internal components are visible.

5.4.2 First tube

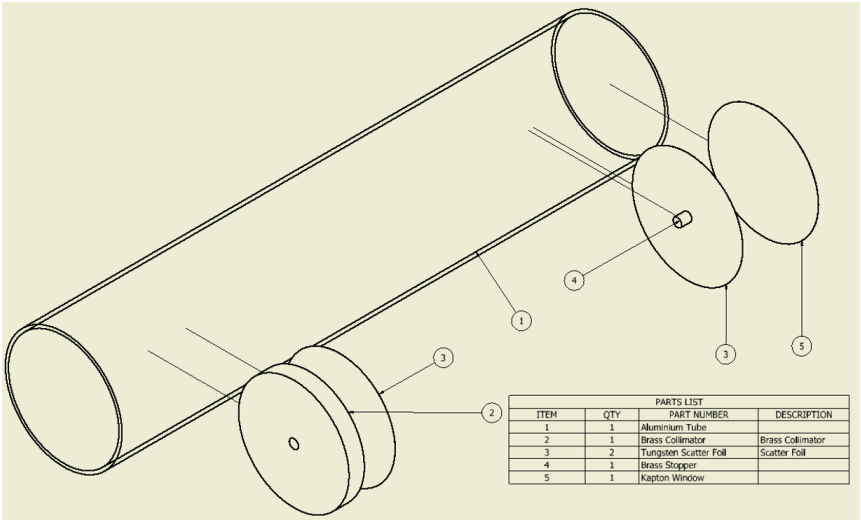


Figure 12: Side wireframe view of the dual scattering tube. The model is shown in an 'exploded' view such that the internal components are visible.

The first 356 mm aluminium tube in the beamline houses the first two 0.025 mm tungsten scattering foils, a 6 mm brass collimator, a 5.71 mm brass stopper, before ending at a 0.05mm Kapton window. The tube is kept under vacuum conditions. This is known as a dual scattering tube, as protons are passively scattered by both the tungsten foils, spreading them out until they are uniformly distributed in the lateral plane of the beamline. This effect is more noticeable after the second foil, as the first acts to scatter the angular distribution of the particles, whereas the second foil acts on the beam after the stopper which has removed a large portion of the central distribution of the beam. Tungsten is used in both the scattering foils due to its high density due to its high atomic number, causing it to be a highly scattering material. The protons exit the tube via a Kapton window. Kapton is chosen for its high resistance to radiation damage and high structural stability under vacuum. After leaving the dual scattering tube, the protons enter the first aluminium box.

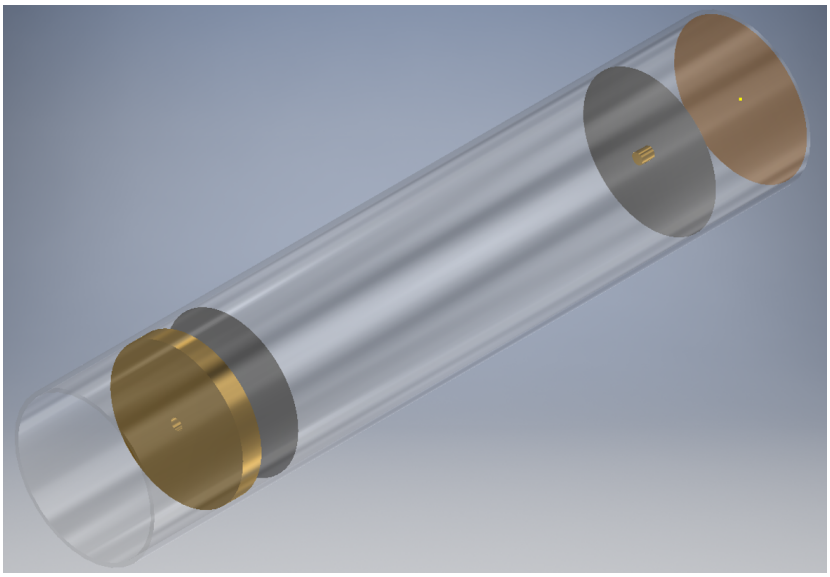


Figure 13: Side view of the dual scattering tube as designed in AutoDesk Inventor. The aluminium tube has been made transparent such that the internal components are visible.

5.4.3 First box and second tube

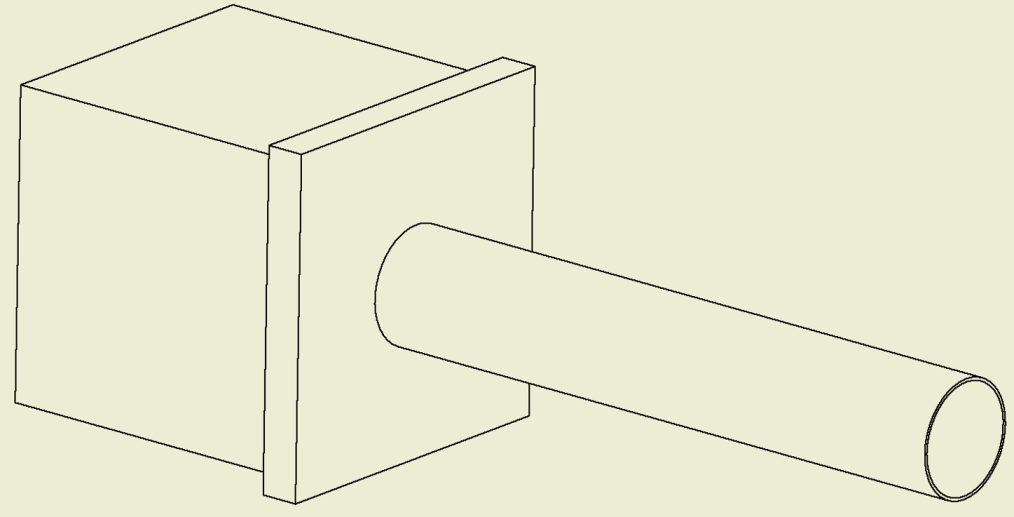


Figure 14: Side wireframe view of the first box and second tube.

This box (shown in figures 14 and 15), while empty in the simulation, usually houses the range shifter and a rotating propeller shaped modulator wheel (also made of PMMA). The range shifter is a block of PMMA used to reduce the energy of the beam by a known value, specified by the shifters water equivalent thickness or WET. The propeller shaped modulator wheel is used to create a beam that constantly varies in energy, such as to produce a spread out Bragg peak (see figure 3). This can be easily understood by considering that the incident thickness of the modulator wheel to the oncoming beam varies as a saw-tooth wave. The second tube connects the first aluminium box to the second, and is joined to the first by a block of iron that holds the tube in place and off the treatment table.

5.4.4 Dosimetry box

The second aluminium boss houses another collimator (of inner radius 10 mm, the dose monitors (shown in more detail in section 5.4.5), and an aluminium fixture suspending two perpendicular tungsten wires of diameter 0.4 mm. The nozzle by which the particles leave this box creates a sharp edge to the resolution of the proton

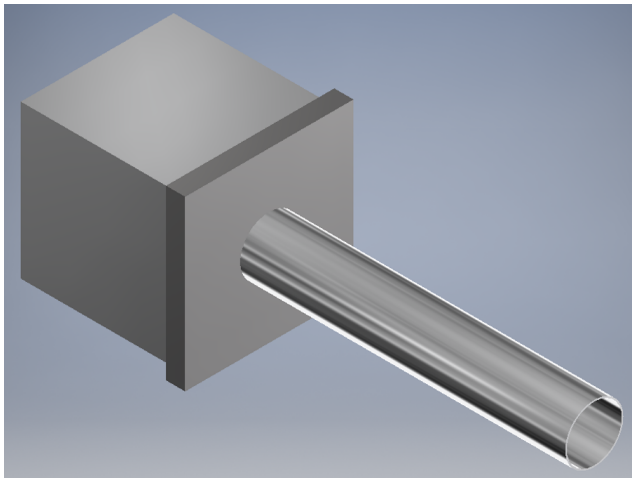


Figure 15: Side view of the first box and second tube as designed in AutoDesk Inventor.

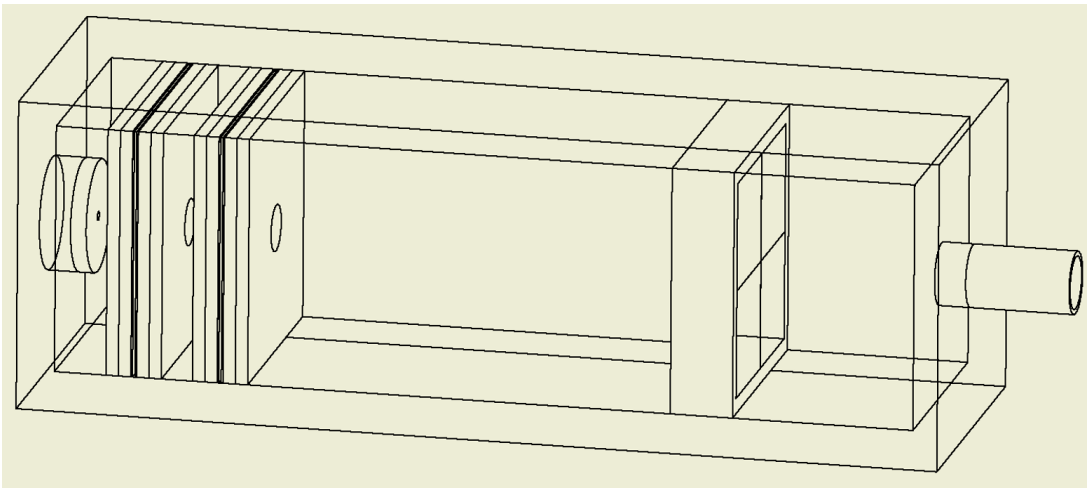


Figure 16: Side wireframe view of the dosimetry box. The aluminium housing has been made transparent.

distribution. The tungsten wires are used to measure the profile of the beam. This is done by measuring the electrical current from both wires, and corresponding this to a horizontal or vertical position along the wire. It should be noted that the wires are designed to cross along the axis of travel of the beam (i.e. the transverse origin of the proton beam).

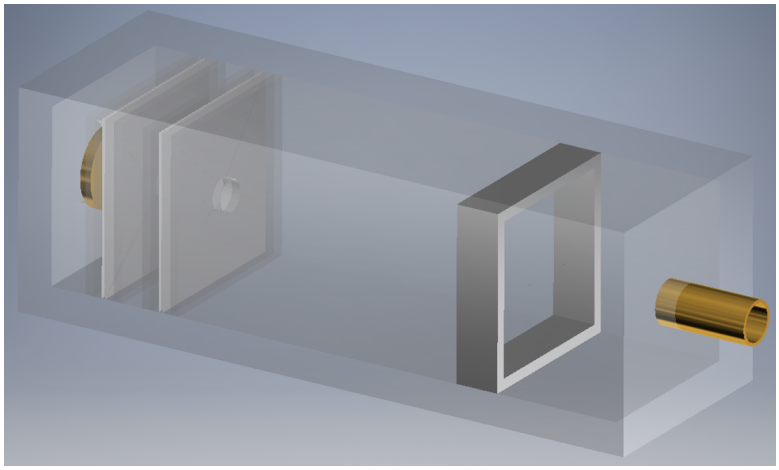
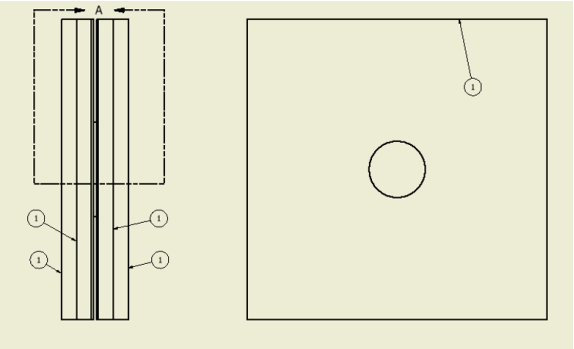


Figure 17: Side view of the dosimetry box as designed in AutoDesk Inventor. The aluminium housing has been made transparent.



(a) Side-on schematic view of the dose monitors. View (b) is shown in figure 19

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	4	Outer Perspex (PMMA) Layer	
2	2	HV Foil (Mylar (PET) Layer)	
3	2	HV Foil (Aluminium Layer)	
4	2	Inner Perspex (PMMA) Layer	
5	1	Guard Ring (Material Unknown)	

(b) Parts list and key for the schematic view of the dose monitors

Figure 18: Schematic diagram and parts list for the dose monitors as used in simulation.

5.4.5 Dose monitors

The dose monitors (shown in figures 18 and 19) consist of two 8 mm sheets of perspex, a high voltage foil that consists of 0.005 mm of Mylar (a thin PET sheet) aluminised by 0.001 mm of aluminium, an inner 1 mm sheet of perspex, a guard ring of width 1.6 mm and outer radius 50 mm, followed by another 1 mm sheet of perspex, the second HV foil (as before, the aluminium side facing towards the centre of the dose monitor), and finally two more 8 mm outer perspex sheets. All the holes in the dose monitor have radius 30 mm. When a proton strikes the HV foil, the volume of air

that separates them acts like an ionisation chamber. These dose monitors are an approximation initially used in the simulation, designed in a way that should affect the virtual beam properties only slightly differently from how the real dose monitors do to the real beam. However, now that the components of the virtual beamline can be fabricated using CAD software packages, creating a much more true to life beamline is possible.

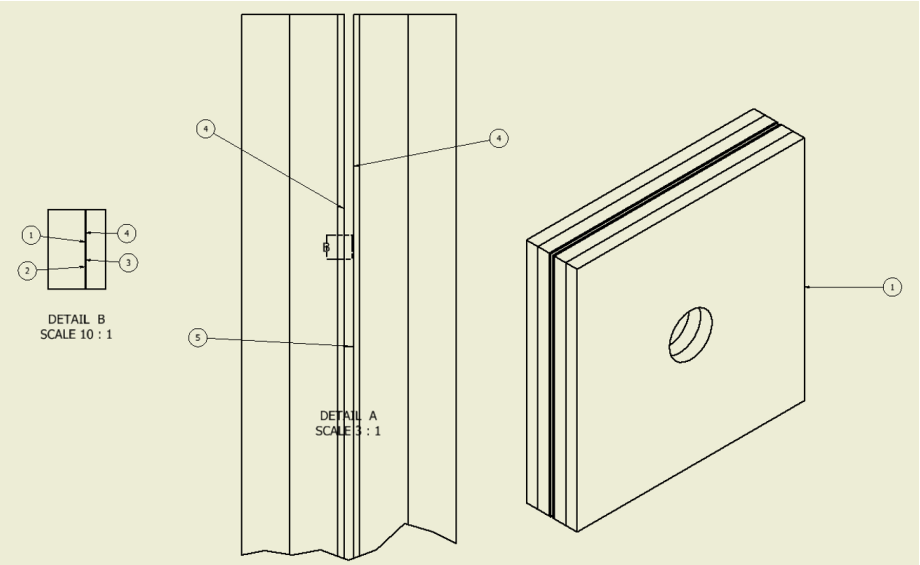


Figure 19: Zoomed in view of the dose monitors, showing the high voltage sheets. See figure 18b for a key.

Realistic dose monitor

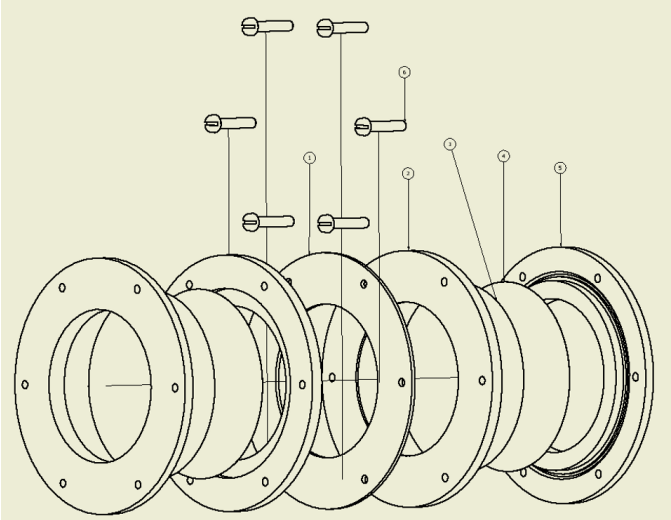


Figure 20: Side-on, wireframe, exploded view of the realistic dose monitor.

The dose monitors in figures 20, 21, and 22 are based off design blueprints for the dose monitors used in the Clatterbridge beamline. As such, they are much more detailed than the approximations discussed earlier. However, these were never implemented into the main simulation geometry due to the fact that the perspex carrier blocks detailed in the blueprints were lacking measurements.

PARTS LIST			
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	Guard Ring	
2	2	Inner Foil Clamp	
3	2	HV Foil (Aluminium 0.001mm)	
4	2	HV Foil (Mylar 0.005mm)	
5	2	Outer Foil Clamp	
6	6	DIN 84 - M4 x 20	Slotted Cheese Head Screw

Figure 21: Parts list for the realistic dose monitor.

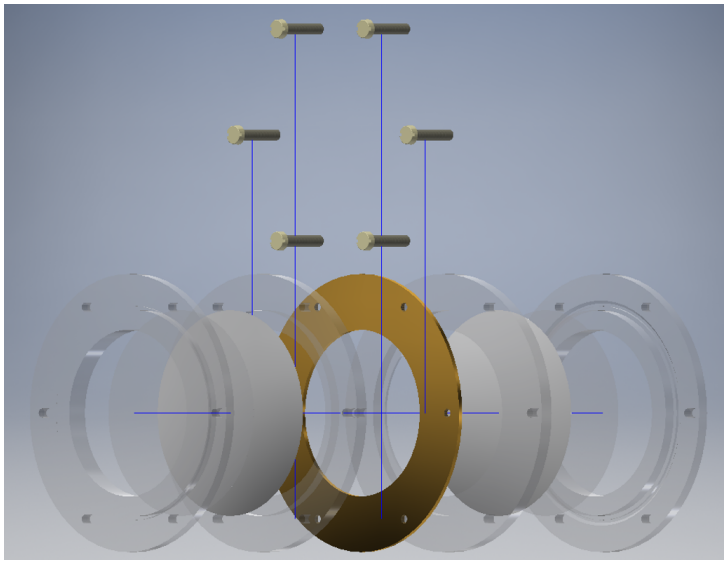


Figure 22: Side-on, exploded view of the realistic dose monitor as designed in AutoDesk Inventor.

5.5 The Conversion Process

After the beamline was drawn up in a CAD software package, it had to be converted to a filetype that GEANT4 can handle. This conversion process is detailed in this subsection.

First, the beamline was drawn up in AutoDesk Inventor. This process was split into a few stages; first, each component of the beamline was modelled independently in a `part` file. These files contained the basic geometry of the individual components, as well as the physical properties of the component (if known). Examples of these `part` files are shown in section 5.4. After all individual components were drawn up, an `assembly` file was created. This file allowed for the relative positions of the components to be defined, and provided a visualisation of the complete beamline. `Assembly` files were drawn up for more complex components, for example the dose monitors, and these component `assembly` files added to the total beamline `assembly` file.

Once the complete beamline geometry was assembled in the master beamline `assembly` file, the beamline was split up into files containing all components of a

single material. This process allowed all components of a specific material to be exported to an STL file (which does *not* preserve material type), and subsequently converted to a GDML file (which *does* preserve material type) all the while conserving the relative positions of the beamline components. It should be noted that AutoDesk Inventor can only export files to an STLB format, so an online file-type converter was used to unpack and convert the STLB files to a readable STLA format.

After exporting each material file to an STLB format, a python script was used to convert the file to a GEANT4 readable GDML file. This script is based off [43]; it was modified to include more materials, as well as a geometry tree file structure, and other quality of life changes such as the ability to define the world volume (as opposed to the world volume being a staticly defined box with side length 10000 mm). As the materials used in previous iterations of the beamline geometry had been written in GEANT4 natively, a script was developed to parse the deprecated GEANT4 materials code and output a string representative of the GDML formatting required. This string was then added to the conversion script such that these new materials were defined upon conversion. The modifier script can be seen in appendix A.1.

When the files were initially converted, they appeared much smaller than anticipated. This was due to the faulty assumption that the component STL files would conserve their units from the original CAD files, and so when read in to the master GDML file were assumed to be in units of mm. However, upon inspection, while the internal relative lengths were conserved, the file had a factor of 10 missing from all lengths i.e. it had been converted to units of cm. Once this was noticed, the conversion script was changed once again to reflect this.

One issue encountered in reading in the component files to the master file was an issue regarding sub-directories housing the component files. If the component GDML files were stored in a directory lower than the directory that the python script was called from, the local names of the daughter files within the mother file would contain

the directory name i.e. `GDMLfiles/AluminiumComponents.gdml`. This became an issue for two separate reasons. Firstly, the script was run on a windows machine, and as directory names on windows systems use a backslash to denote a lower directory, this meant that the GDML files generated had to be manually changed when being used on the scientific Linux server. However, it became apparent that this was more than just an operating system issue, as upon further investigation the "name" attribute of a GDML file is identified in the schema as being of type *xs:ID*, defined as an *NCName* value. As NCNames are only allowed to contain alphanumeric characters and underscores, the slashes required to house daughter GDML files would not be allowed in the file names. While this clutters up the source code for any simulation (i.e. All the GDML files must be present in the same directory), it may be possible to extend the schema to allow for these characters in the file-names (see section 7.2 for a more thorough discussion of this).

After the geometry files were converted to a GDML format, they were imported into the GEANT4 simulation.

5.6 Importing GDML models into GEANT4

GEANT4 allows importing of GDML files by way of a *parser* defined by the `G4GDMLParser` object. To use this, necessary header files must be included and the parser object instantiated;

```
#include "G4GDMLParser.hh"
G4GDMLParser parser;
```

The parser must then be run on the master GDML file;

```
parser.Read("master.gdml");
```

By default, the volume names contained within the GDML file are stripped of whitespaces that can potentially cause errors when they are imported. This can be disabled by calling the `SetStripFlag(False)` method before reading the file.

Also by default, the GDML file read in is compared to the standard GDML schema. This comparison can be switched off by instead calling

```
parser.Read("master.gdml", False);
```

to read in the file.

After the file is read in, the world volume can be extracted

```
G4VphysicalVolume* pworld = parser.GetWorldVolume();
```

and it can be manipulated as any other `G4VphysicalVolume` pointer (i.e. it can be set to be invisible, which is often done as the world volume is often not meant to be visualised).

6 Results

After importing the GDML files, a run consisting of 10^5 primaries was performed. This number was kept relatively low, as the aim of the run was only to compare how well the new geometry fared when compared to the version of the beamline created directly in GEANT4.

The first comparison was made at $z = 0$ mm. The beam at the source is shown in figure 23. This plot hints at the simulation containing errors, as the emittance spectrum is not behaving as expected. Here, the emittance is expected to be elliptic in shape, but the shape seen in the simulation is more akin to an egg timer. However, compared to figure 5, this behaviour is similar to the previous version of the simulation, and so this behaviour may be due to an error in the analysis script instead.

The second comparison was made before and after the first brass collimator. This is the point where the simulations results increasingly diverge (i.e. when compared to figure 6. The expected results at this point is that approximately 70-90% of the protons are lost between $z = 50$ mm and $z = 75$ mm, due to them being absorbed by

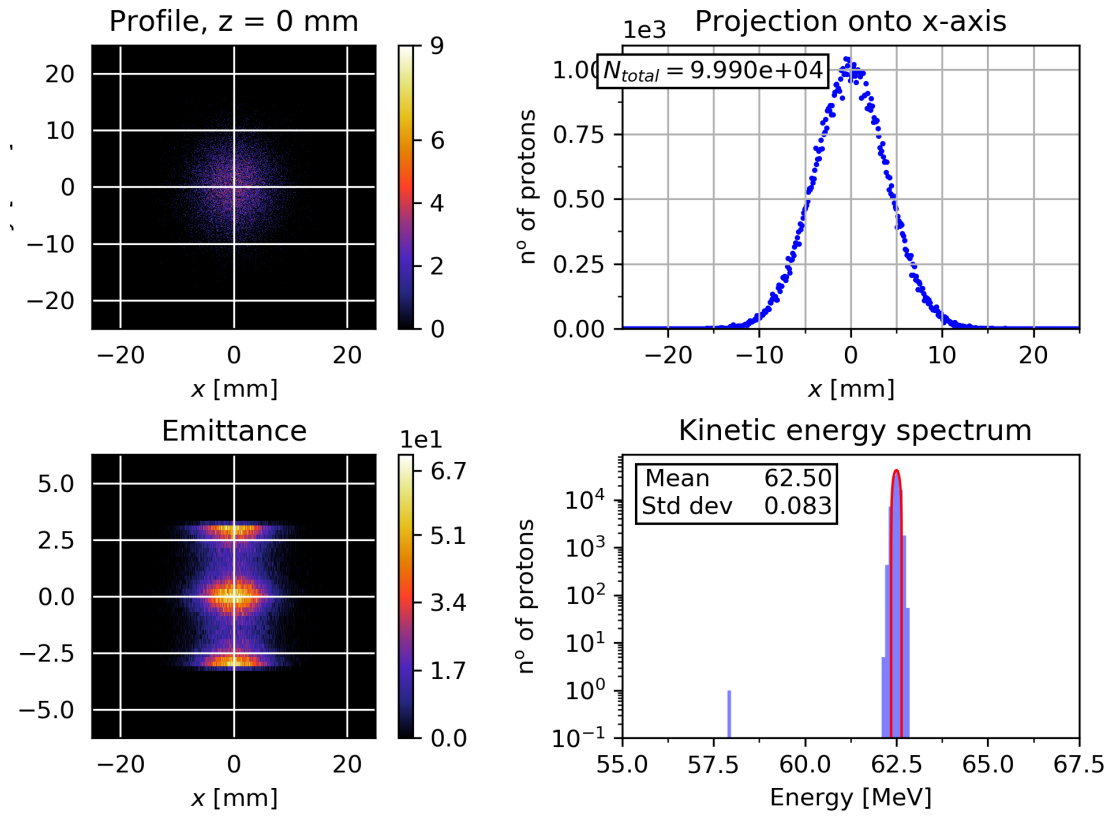


Figure 23: The beam at the source. While the profile plot and the energy spectrum are as expected, the emittance is not behaving as expected.

the collimator that only has an internal diameter of 6 mm compared to the sources diameter of 30 mm. However, as seen by figures 24 and 25, the simulation using the GDML files does not act this way. In fact, the protons pass through the volume containing the collimator as if it is not there. This hints at a few mistakes in the simulation.

One possible cause of this is that the material is incorrectly defined. If the density of brass was defined incorrectly in the GDML file, this could cause the effects seen.

A second possibility could be that due to the nature of only splitting up the geometry by its materials, and not its components, the simulation may not be able to determine when primaries leave certain solid volumes. This could be corrected by changing the protocol to not include a single GDML file per material in the beamline, but rather to include a single GDML file per beamline component. This would define

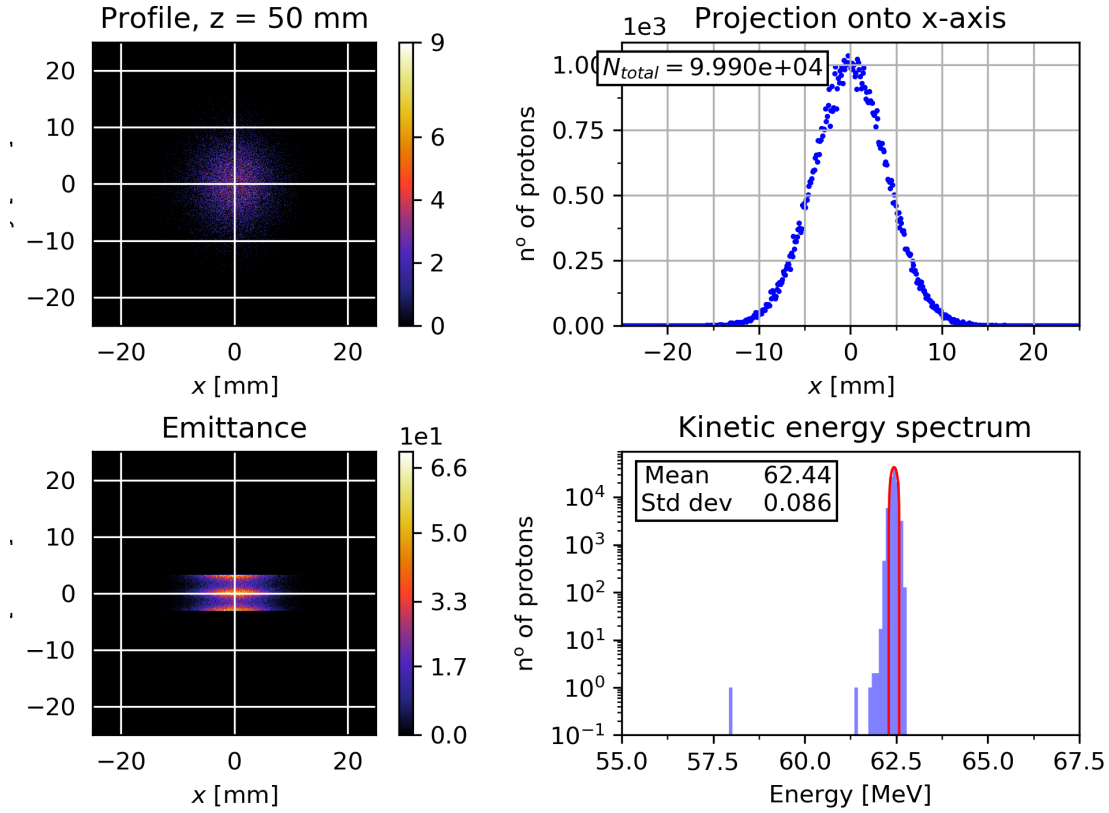


Figure 24: The beam before passing the first collimator.

the beamline at a slightly higher resolution, and may correct the behaviour seen in the simulation.

Another possibility is that the origin of the beamline geometry in the GDML file differs from the origin created when the previous version of the simulation was created. This would cause the entire beamline to be at a different location within the GDML file, and so when setting the source position the protons would behave unpredictably further down the beamline.

A final comparison was made at the start of the first aluminium box. At this point, the number of primaries that are still active in the simulation was expected to not change by a significant amount. However, the simulation using GDML files showed unexpected behaviour here again when compared to figure 7. Here, almost all (99.99%) of the primaries were killed off, as shown in figures 26 and 27. This is

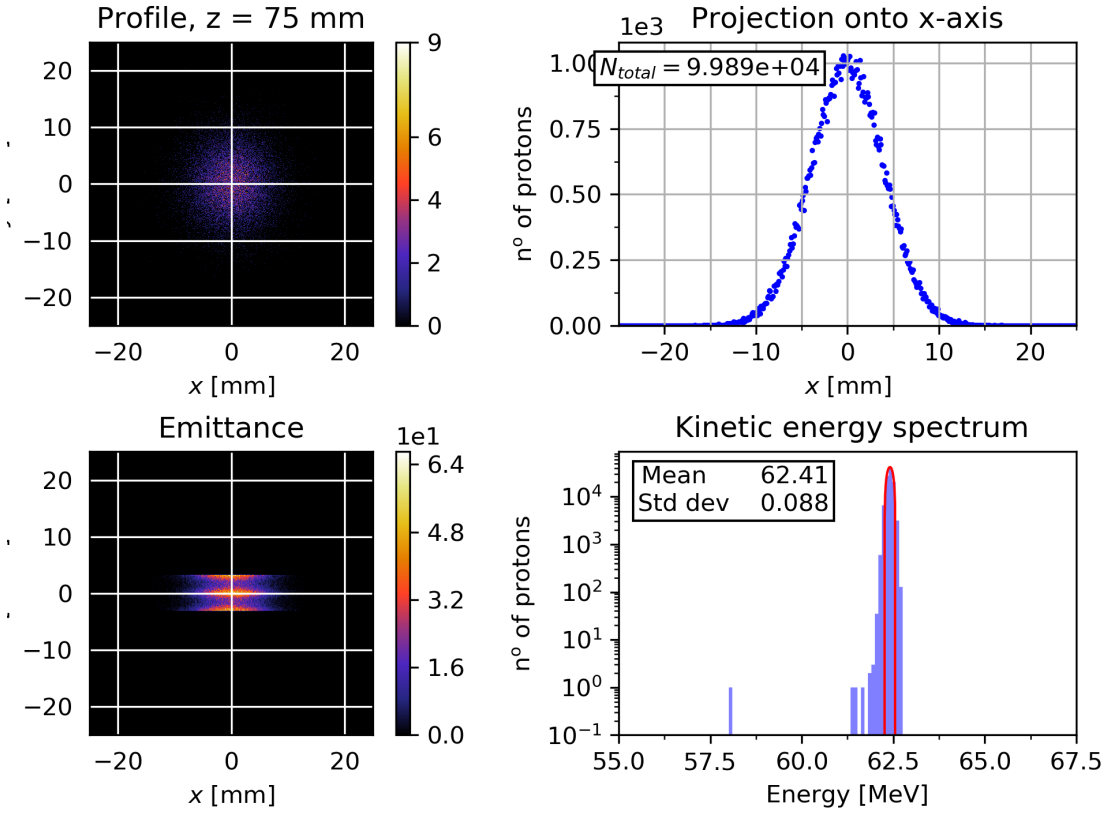


Figure 25: The beam after passing the first collimator.

too high a number to suggest that this is the true position of the first brass collimator however, so alternate reasons for this behaviour are suggested.

One possible reason for this behaviour is that this is the true z position of the concrete wall housing the beamline. As this is made of marble concrete, the drop off in number of protons would be expected to be this high if this was the case.

A second possible reason is that the hole into the aluminium box is incorrectly defined within the GDML file. If the hole did not appear, then the protons would be hitting a wall of aluminium, culling their population.

Due to the large number of primaries killed off here, even when run with 10^7 primaries, no protons made it to $z = 1825$.

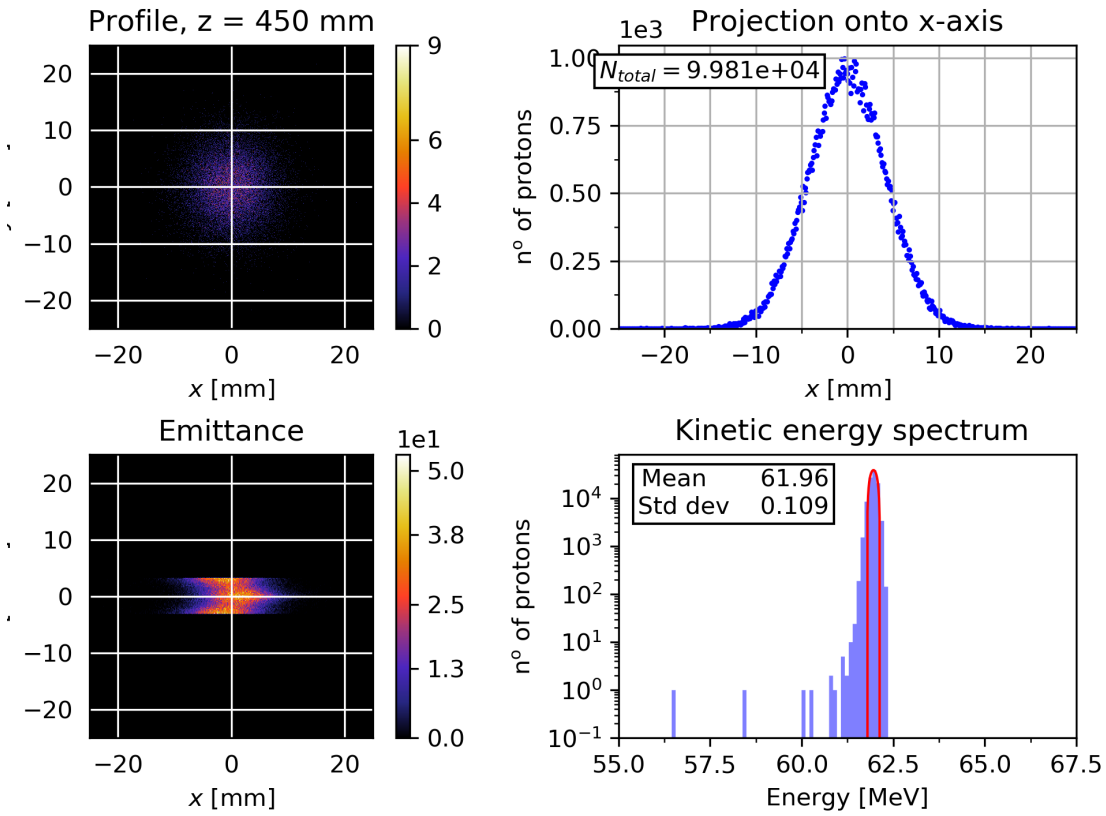


Figure 26: The beam before entering the first aluminium box.

7 Conclusions

Without importing the GDML file, the simulation behaves as expected to, barring the fact that the output energy is slightly too high (60.1 MeV versus the expected 60.0 MeV). One possible reason behind this is that the input parameters of the beamline are slightly incorrect, as they have not been confirmed by the Clatterbridge facility. A second reason could be due to the incompleteness in the description of the beamline. While the second aluminium box is empty in the simulation, in the real beamline this box houses a range shifter and a rotating propeller shaped modulator wheel. These two devices are designed to reduce the energy of the beam, and these reductions in the beam energy are not accounted for in the simulation.

The simulation, after being converted to accept a GDML file containing its geometry, did not run as expected. The emittance at the source was not as previous

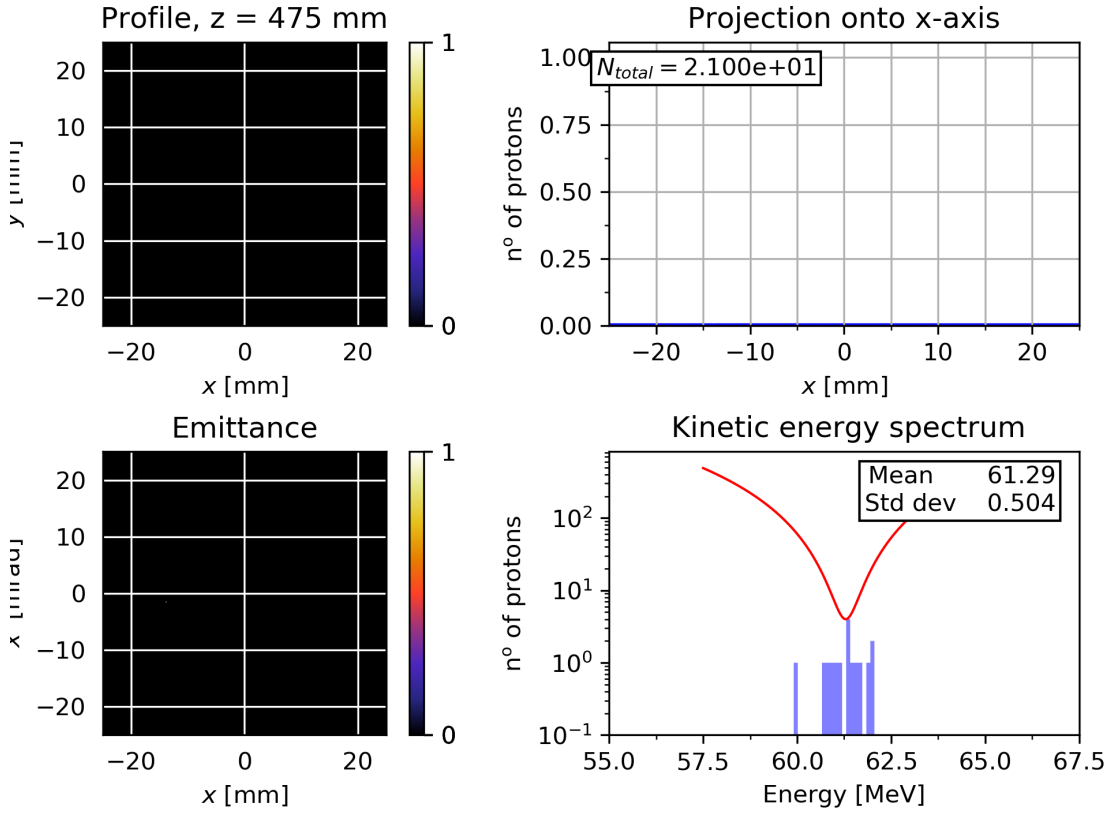


Figure 27: The beam after entering the first aluminium box.

versions had it described, and the primaries did not interact with the geometry correctly.

7.1 Improvements

One improvement that could have been made was the splitting up of the CAD models into its materials components. The method employed to do this in the project was to first assemble the entire beamline, then create a copy of the assembly, delete components such that only part files of a particular material file were left, export this as an STL file, and then undo the deletion of components until the beamline was completed again. This was to preserve the relative geometry of components of the beamline. While it accomplished its task and the relative geometry of the material components was preserved, it created a lot of errors in the part files due to the dependencies of the part files on the total assembly file.

This has the potential to solve the error in the simulations discussed in the results section due to it giving the simulation a larger distinction between material volumes

A better approach to this could have been to specify the origin of the system within each part file (by creating each component via a sketch in a plane offset from the local file origin), and export these part files directly to STL. While this would add time and complexity to the creation of each part file, this would reduce the potential for accidental deletion of component files when adjusting the assembly file. Furthermore, this would allow for a more defined beamline when importing the system to GDML as each part of the system would be held in a separate GDML file.

7.2 Next steps

The next steps to be taken involve modifying the GDML files to be more realistic, and to include more components of the beamline, such as including upstream components. An example of this would be to include the realistic dose monitors into the beamline geometry. However, this would either require updated schematics from the Clatterbridge centre or for measurements of the dose monitors to be made by hand, and so may not be feasible, meaning that approximations to the dose monitor housings would be required instead.

One possible extension that would improve the visualisation of the output would be to extend the GDML schema to include a colouring for various beamline components. A second quality of life change would be to change the name variable type of GDML files in the schema to allow for GDML component files to be kept in a subdirectory.

A further quality of life improvement would be changes to the CMakeLists file. These changes would cause the GDML files and possible GDML subdirectory to be carried over to the build folder upon compiliation.

The most important follow up to this project is to find the reasons as to why the GDML described geometry did not behave as expected. The reasons for this

could include the origin of the imported beamline being incorrect, the densities of defined materials being incorrect, and the beamline not being correctly defined post conversion from STL.

Appendix

A Scripts

This appendix contains the various scripts written to complete various tasks during the project.

A.1 Geant4 to Python string exporting script

```
# -*- coding: utf-8 -*-
"""
Created on Tue Mar  6 17:58:58 2018

@author: Jordan
"""

import re

G4List = """
  G4Element* H  = new G4Element( "Hydrogen" , "H" , z=1 , a=1.008    *
   CLHEP::g/CLHEP::mole );
  G4Element* Li = new G4Element( "Lithium"  , "Li", z=3 , a=6.941    *
   CLHEP::g/CLHEP::mole );
  G4Element* B  = new G4Element( "Boron"    , "B" , z=5 , a=10.81    *
   CLHEP::g/CLHEP::mole );
  G4Element* C  = new G4Element( "Carbon"   , "C" , z=6 , a=12.01    *
   CLHEP::g/CLHEP::mole );
  G4Element* N  = new G4Element( "Nitrogen" , "N" , z=7 , a=14.01    *
   CLHEP::g/CLHEP::mole );
  G4Element* O  = new G4Element( "Oxygen"   , "O" , z=8 , a=16.00    *
   CLHEP::g/CLHEP::mole );
  G4Element* Na = new G4Element( "Sodium"   , "Na", z=11, a=22.99    *
   CLHEP::g/CLHEP::mole );
```

```
  G4Element* Mg = new G4Element( "Magnesium", "Mg", z=12, a=24.305  *
   CLHEP::g/CLHEP::mole );
  G4Element* Al = new G4Element( "Aluminium", "Al", z=13, a=26.98   *
   CLHEP::g/CLHEP::mole );
  G4Element* Si = new G4Element( "Silicon"  , "Si", z=14, a=28.09   *
   CLHEP::g/CLHEP::mole );
  G4Element* Ar = new G4Element( "Argon"    , "Ar", z=18, a=39.948  *
   CLHEP::g/CLHEP::mole );
  G4Element* Ca = new G4Element( "Calcium"  , "Ca", z=20, a=40.08   *
   CLHEP::g/CLHEP::mole );
  G4Element* Fe = new G4Element( "Iron"     , "Fe", z=26, a=55.85   *
   CLHEP::g/CLHEP::mole );
  G4Element* Co = new G4Element( "Cobalt"   , "Co", z=27, a=58.933  *
   CLHEP::g/CLHEP::mole );
  G4Element* Cu = new G4Element( "Copper"   , "Cu", z=29, a=63.546  *
   CLHEP::g/CLHEP::mole );
  G4Element* Zn = new G4Element( "Zinc"     , "Zn", z=30, a=65.38   *
   CLHEP::g/CLHEP::mole );
  G4Element* Eu = new G4Element( "Europium" , "Eu", z=63, a=151.964 *
   CLHEP::g/CLHEP::mole );
  G4Element* Cs = new G4Element( "Caesium"  , "Cs", z=55, a=132.905 *
   CLHEP::g/CLHEP::mole );
  G4Element* W  = new G4Element( "Tungsten" , "W" , z=74, a=183.84  *
   CLHEP::g/CLHEP::mole );"""

G4List = G4List.split(";")

bigstring = ""
element = {}

for line in G4List:
    if line.split("(") == ['']:
        break
```

```
    line = line.split("(")[1][:-23].split(",")
    name = ''.join([i for i in line[0] if i != " "])
    formula = ''.join([i for i in line[1] if i != " "])
    z = ''.join([i for i in line[2][3:] if i != " "])
    a = ''.join([i for i in line[3][3:] if i != " "])
    element[formula[1:-1]] = [name[1:-1], z, a]
    form = '<element name='+name+'    formula='+formula+'    Z="'+z+'">
     <atom value="'+a+'"/>    </element>\n'
    bigstring+=form

print(bigstring)

G4matbit = """  fAir = new G4Material( "Air", density=1.290*CLHEP::mg/
   CLHEP::cm3, ncomponents=3 );
  fAir->AddElement( N, fractionmass=0.7810 );         // 78.10%
  fAir->AddElement( O,  fractionmass=0.2096 );         // 20.96%
  fAir->AddElement( Ar, fractionmass=0.0094 );         // 0.94%

  fWater = new G4Material( "Water", density=1.*CLHEP::g/CLHEP::cm3,
   ncomponents=2 );
  fWater->AddElement( H, natoms=2 );
  fWater->AddElement( O, natoms=1 );
  fWater->GetIonisation()->SetMeanExcitationEnergy( 78.0*eV );

  fScintillatorPVT = new G4Material( "Scintillator_PVT", density=1.023*
   CLHEP::g/CLHEP::cm3, ncomponents=2 );
  fScintillatorPVT->AddElement( C, natoms=9 );
  fScintillatorPVT->AddElement( H, natoms=10 );

  fMarbleConcrete = new G4Material( "MarbleConcrete", density=2.7*CLHEP
   ::g/CLHEP::cm3, ncomponents=12 );
  fMarbleConcrete->AddElement( Ca, fractionmass=0.473942 );     //
   47.3942%
```

```
66  fMarbleConcrete->AddElement( O,  fractionmass=0.375    );    // 37.5%
67  fMarbleConcrete->AddElement( C,  fractionmass=0.105    );    // 10.5%
68  fMarbleConcrete->AddElement( Si, fractionmass=0.024    );    // 2.4%
69  fMarbleConcrete->AddElement( Fe, fractionmass=0.01     );    // 1%
70  fMarbleConcrete->AddElement( Al, fractionmass=0.007    );    // 0.7%
71  fMarbleConcrete->AddElement( Mg, fractionmass=0.003    );    // 0.3%
72  fMarbleConcrete->AddElement( Na, fractionmass=0.002    );    // 0.2%
73  fMarbleConcrete->AddElement( Li, fractionmass=0.000037 );    // 37
     ppm
74  fMarbleConcrete->AddElement( Co, fractionmass=0.000018 );    // 18
     ppm
75  fMarbleConcrete->AddElement( Cs, fractionmass=0.000002 );    // 2 ppm
76  fMarbleConcrete->AddElement( Eu, fractionmass=0.000001 );    // 1 ppm
77
78  fAluminium = new G4Material( "Aluminium", density=2.7*CLHEP::g/CLHEP
      ::cm3, ncomponents=1 );
79  fAluminium->AddElement( Al, fractionmass=1 );
80
81  fBrass = new G4Material( "Brass", density=8.75*CLHEP::g/CLHEP::cm3,
     ncomponents=2 );
82  fBrass->AddElement( Cu, fractionmass=0.7 );       // 70%
83  fBrass->AddElement( Zn, fractionmass=0.3 );       // 30%
84
85  fTungsten = new G4Material( "Tungsten", density=19.25*CLHEP::g/CLHEP
      ::cm3, ncomponents=1 );
86  fTungsten->AddElement( W, fractionmass=1. );
87
88  fKapton = new G4Material( "Kapton", density=1.42*CLHEP::g/CLHEP::cm3,
       ncomponents=4 );
89  fKapton->AddElement( H, fractionmass=0.027 );     // 2.7%
90  fKapton->AddElement( C, fractionmass=0.691 );     // 69.1%
91  fKapton->AddElement( N, fractionmass=0.073 );     // 7.3%
92  fKapton->AddElement( O, fractionmass=0.209 );     // 20.9%
```

```
93
94   fIron = new G4Material( "Iron", density=7.874*CLHEP::g/CLHEP::cm3,
       ncomponents=1 );
95   fIron->AddElement( Fe, fractionmass=1. );
96
97   fPMMA = new G4Material( "PMMA", density=1.18*CLHEP::g/CLHEP::cm3,
       ncomponents=3 );
98   fPMMA->AddElement( C, natoms=5 );
99   fPMMA->AddElement( O, natoms=2 );
100  fPMMA->AddElement( H, natoms=8 );
101
102  fMylar = new G4Material( "Mylar", density=1.397*CLHEP::g/CLHEP::cm3,
       ncomponents=3 );
103  fMylar->AddElement( C, natoms=10 );
104  fMylar->AddElement( H, natoms=8  );
105  fMylar->AddElement( O, natoms=4  );
106
107  fBoratedPlastic = new G4Material( "BoratedPlastic", density=1.04*
       CLHEP::g/CLHEP::cm3, ncomponents=3 );// 5% borated polyethylene
108  fBoratedPlastic->AddElement( B, fractionmass=0.05  );
109  fBoratedPlastic->AddElement( C, fractionmass=0.317 );
110  fBoratedPlastic->AddElement( H, fractionmass=0.633 );"""
111
112 #format
113 """
114       <!-- BGO -->
115       <material name="BGO" formula="Bi4Ge3O12" >
116          <D value="7.13" unit="g/cm3" />
117          <composite n="4" ref="bismuth" />
118          <composite n="3" ref="germanium" />
119          <composite n="12" ref="oxygen" />
120       </material>
121 """
```

```
122
123 for matStr in G4matbit.split("\n\n"):
124     elementData = matStr.split("\n")
125     matName = elementData[0].split("(")[1].split(",")[0][1:]
126     dVal = elementData[0].split("(")[1].split(",")[1][9:].split("*")[0]
127     dUnit = len(elementData[0].split("(")[1].split(",")[1][9:].split("*
    ")[1])
128     newline = '      <material name='+matName+'>\n'
129     newline += '         <D value="'+dVal+'" unit="'+("g/cm3","mg/cm3")[
    dUnit == 20]+'"/>\n'
130     #print('  {"name" : '+matName+',')
131     isComposite = None
132     for ele in elementData[1:]:
133         args = ele.split("(")[1].split(")")[0].split(",")
134         locID = args[0][1:]
135         #print(args)
136         if isComposite is None: #only need to check first line
137             if args[1][1] == "f":
138                 isComposite = False
139             elif args[1][1] == "n":
140                 isComposite = True
141             else:
142                 print(args[1])
143         if isComposite and (args != ['']):
144             currN = int(args[1][8:])
145             newline += '         <composite n="'+str(currN)+'" ref="'+
    element[locID][0].lower()+'" />\n'
146         elif (not isComposite) and (args != ['']):
147             currF = re.findall(r"[-+]?\d*\.\d+|\d+",args[1])[0]
148             newline += '         <fraction n="'+str(currF)+'" ref="'+
    element[locID][0].lower()+'"/>\n'
149         elif args == ['']:
```

```
150             MME = re.findall(r"[-+]?\d*\.\d+|\d+",ele.split("(")[-1])
    [0]
151             newline += '          <MEE unit="eV" value="'+MME+'"/>\n'
152
153     newline += '      </material>\n'
154     print(newline)
```

A.2 DetectorConstruction.cc

```
// M Hentz, 2016
// Updated by J. Silverman, 2018

#include "DetectorConstruction.hh"
#include "DetectorMessenger.hh"

#include "G4Material.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"
#include "G4UnionSolid.hh"
#include "G4SubtractionSolid.hh"
#include "G4Transform3D.hh"
#include "G4VisAttributes.hh"

#include "G4PhysicalVolumeStore.hh"
#include "G4LogicalVolumeStore.hh"
#include "G4SolidStore.hh"
#include "G4GeometryManager.hh"
#include "G4TransportationManager.hh"
#include "G4RunManager.hh"
#include "G4NistManager.hh"

#include "G4UnitsTable.hh"
#include "G4PhysicalConstants.hh"
#include "G4SystemOfUnits.hh"

#include "G4GDMLParser.hh"

```

```
32 DetectorConstruction::DetectorConstruction()
33     :G4VUserDetectorConstruction(),
34      fWorldMaterial(0),
35      fAbsorMaterial(0),
36      lAbsor(0),
37      fDetectorMessenger(0)
38 {
39   fWorldSizeX = fWorldSizeZ = 9450.*CLHEP::mm;
40   fWorldSizeY = 4450.*CLHEP::mm;
41   fWorldMaterial = 0;
42
43   fRoomSizeX = 8000.*CLHEP::mm;
44   fRoomSizeY = 3400.*CLHEP::mm;
45   fRoomSizeZ = 8400.*CLHEP::mm;
46
47
48   fAbsorSizeXY = 40.*CLHEP::mm;
49   fAbsorSizeZ  = 40.*CLHEP::mm;
50   fAbsorMaterial = 0;
51   lAbsor = 0;
52
53   detSizeXY    = fAbsorSizeXY + 5.*CLHEP::mm;
54   detSizeZ     = fAbsorSizeZ  + 5.*CLHEP::mm;
55
56   fLayerNumber = 0;
57   fLayerMass = 0;
58   fLayerSizeXY = 20.*CLHEP::mm;
59   fLayerSizeZ = 0.*CLHEP::mm;
60
61   DefineMaterials();
62
63   // Create commands for interactive definition of the detector
64   fDetectorMessenger = new DetectorMessenger( this );
```

```
65 }
66
67
68 DetectorConstruction::~DetectorConstruction()
69 { delete fDetectorMessenger; }
70
71
72 G4VPhysicalVolume* DetectorConstruction::Construct()
73 { return ConstructVolumes(); }
74
75
76 void DetectorConstruction::DefineMaterials()
77 {
78   // Use G4NistManager if material from NIST database is needed
79   // G4NistManager* nistMan = G4NistManager::Instance();
80
81   //
     ------------------------------------------------------------------------------
82   // Define elements
83   //
     ------------------------------------------------------------------------------
84   G4double z, a;
85
86   G4Element* H  = new G4Element( "Hydrogen" , "H" , z=1 , a=1.008    *
      CLHEP::g/CLHEP::mole );
87   G4Element* Li = new G4Element( "Lithium"  , "Li", z=3 , a=6.941    *
      CLHEP::g/CLHEP::mole );
88   G4Element* B  = new G4Element( "Boron"    , "B" , z=5 , a=10.81    *
      CLHEP::g/CLHEP::mole );
89   G4Element* C  = new G4Element( "Carbon"   , "C" , z=6 , a=12.01    *
      CLHEP::g/CLHEP::mole );
```

```
90  G4Element* N  = new G4Element( "Nitrogen" , "N"  , z=7  , a=14.01    *
     CLHEP::g/CLHEP::mole );
91  G4Element* O  = new G4Element( "Oxygen"   , "O"  , z=8  , a=16.00    *
     CLHEP::g/CLHEP::mole );
92  G4Element* Na = new G4Element( "Sodium"   , "Na", z=11, a=22.99    *
     CLHEP::g/CLHEP::mole );
93  G4Element* Mg = new G4Element( "Magnesium", "Mg", z=12, a=24.305   *
     CLHEP::g/CLHEP::mole );
94  G4Element* Al = new G4Element( "Aluminium", "Al", z=13, a=26.98    *
     CLHEP::g/CLHEP::mole );
95  G4Element* Si = new G4Element( "Silicon"  , "Si", z=14, a=28.09    *
     CLHEP::g/CLHEP::mole );
96  G4Element* Ar = new G4Element( "Argon"    , "Ar", z=18, a=39.948   *
     CLHEP::g/CLHEP::mole );
97  G4Element* Ca = new G4Element( "Calcium"  , "Ca", z=20, a=40.08    *
     CLHEP::g/CLHEP::mole );
98  G4Element* Fe = new G4Element( "Iron"     , "Fe", z=26, a=55.85    *
     CLHEP::g/CLHEP::mole );
99  G4Element* Co = new G4Element( "Cobalt"   , "Co", z=27, a=58.933   *
     CLHEP::g/CLHEP::mole );
100 G4Element* Cu = new G4Element( "Copper"   , "Cu", z=29, a=63.546   *
     CLHEP::g/CLHEP::mole );
101 G4Element* Zn = new G4Element( "Zinc"     , "Zn", z=30, a=65.38    *
     CLHEP::g/CLHEP::mole );
102 G4Element* Eu = new G4Element( "Europium" , "Eu", z=63, a=151.964 *
     CLHEP::g/CLHEP::mole );
103 G4Element* Cs = new G4Element( "Caesium"  , "Cs", z=55, a=132.905 *
     CLHEP::g/CLHEP::mole );
104 G4Element* W  = new G4Element( "Tungsten" , "W"  , z=74, a=183.84   *
     CLHEP::g/CLHEP::mole );
105
106
107 //
```

```
108  // Define materials
109  //
     ------------------------------------------------------------------------

110  G4double density, temperature, pressure, fractionmass;
111  G4int natoms, ncomponents;
112
113  // Vacuum
114  density     = universe_mean_density;    // from PhysicalConstants.h
115  pressure    = 3.e-18*pascal;
116  temperature = 2.73*kelvin;
117  fVacuum = new G4Material( "Vacuum", z=1, a=1.008*g/mole, density,
      kStateGas, temperature, pressure );
118
119  fAir = new G4Material( "Air", density=1.290*CLHEP::mg/CLHEP::cm3,
      ncomponents=3 );
120  fAir->AddElement( N,  fractionmass=0.7810 );        // 78.10%
121  fAir->AddElement( O,  fractionmass=0.2096 );        // 20.96%
122  fAir->AddElement( Ar, fractionmass=0.0094 );        // 0.94%
123
124  fWater = new G4Material( "Water", density=1.*CLHEP::g/CLHEP::cm3,
      ncomponents=2 );
125  fWater->AddElement( H, natoms=2 );
126  fWater->AddElement( O, natoms=1 );
127  fWater->GetIonisation()->SetMeanExcitationEnergy( 78.0*eV );
128
129  fScintillatorPVT = new G4Material( "Scintillator_PVT", density=1.023*
      CLHEP::g/CLHEP::cm3, ncomponents=2 );
130  fScintillatorPVT->AddElement( C, natoms=9 );
131  fScintillatorPVT->AddElement( H, natoms=10 );
132
```

```
133  fMarbleConcrete = new G4Material( "MarbleConcrete", density=2.7*CLHEP
       ::g/CLHEP::cm3, ncomponents=12 );
134  fMarbleConcrete->AddElement( Ca, fractionmass=0.473942 );     //
       47.3942%
135  fMarbleConcrete->AddElement( O,  fractionmass=0.375    );     // 37.5%
136  fMarbleConcrete->AddElement( C,  fractionmass=0.105    );     // 10.5%
137  fMarbleConcrete->AddElement( Si, fractionmass=0.024    );     // 2.4%
138  fMarbleConcrete->AddElement( Fe, fractionmass=0.01     );     // 1%
139  fMarbleConcrete->AddElement( Al, fractionmass=0.007    );     // 0.7%
140  fMarbleConcrete->AddElement( Mg, fractionmass=0.003    );     // 0.3%
141  fMarbleConcrete->AddElement( Na, fractionmass=0.002    );     // 0.2%
142  fMarbleConcrete->AddElement( Li, fractionmass=0.000037 );     // 37
       ppm
143  fMarbleConcrete->AddElement( Co, fractionmass=0.000018 );     // 18
       ppm
144  fMarbleConcrete->AddElement( Cs, fractionmass=0.000002 );     // 2 ppm
145  fMarbleConcrete->AddElement( Eu, fractionmass=0.000001 );     // 1 ppm
146
147  fAluminium = new G4Material( "Aluminium", density=2.7*CLHEP::g/CLHEP
       ::cm3, ncomponents=1 );
148  fAluminium->AddElement( Al, fractionmass=1 );
149
150  fBrass = new G4Material( "Brass", density=8.75*CLHEP::g/CLHEP::cm3,
       ncomponents=2 );
151  fBrass->AddElement( Cu, fractionmass=0.7 );       // 70%
152  fBrass->AddElement( Zn, fractionmass=0.3 );       // 30%
153
154  fTungsten = new G4Material( "Tungsten", density=19.25*CLHEP::g/CLHEP
       ::cm3, ncomponents=1 );
155  fTungsten->AddElement( W, fractionmass=1. );
156
157  fKapton = new G4Material( "Kapton", density=1.42*CLHEP::g/CLHEP::cm3,
       ncomponents=4 );
```

```
158  fKapton->AddElement( H, fractionmass=0.027 );      // 2.7%
159  fKapton->AddElement( C, fractionmass=0.691 );      // 69.1%
160  fKapton->AddElement( N, fractionmass=0.073 );      // 7.3%
161  fKapton->AddElement( O, fractionmass=0.209 );      // 20.9%
162
163  fIron = new G4Material( "Iron", density=7.874*CLHEP::g/CLHEP::cm3,
       ncomponents=1 );
164  fIron->AddElement( Fe, fractionmass=1. );
165
166  fPMMA = new G4Material( "PMMA", density=1.18*CLHEP::g/CLHEP::cm3,
       ncomponents=3 );
167  fPMMA->AddElement( C, natoms=5 );
168  fPMMA->AddElement( O, natoms=2 );
169  fPMMA->AddElement( H, natoms=8 );
170
171  fMylar = new G4Material( "Mylar", density=1.397*CLHEP::g/CLHEP::cm3,
       ncomponents=3 );
172  fMylar->AddElement( C, natoms=10 );
173  fMylar->AddElement( H, natoms=8  );
174  fMylar->AddElement( O, natoms=4  );
175
176  // 5% borated polyethylene
177  fBoratedPlastic = new G4Material( "BoratedPlastic", density=1.04*
       CLHEP::g/CLHEP::cm3, ncomponents=3 );
178  fBoratedPlastic->AddElement( B, fractionmass=0.05  );
179  fBoratedPlastic->AddElement( C, fractionmass=0.317 );
180  fBoratedPlastic->AddElement( H, fractionmass=0.633 );
181
182
183  //
       ------------------------------------------------------------------------------
184  // Default materials
```

```
185   //
      ------------------------------------------------------------------------------

186   fWorldMaterial = fAir;
187   fAbsorMaterial = fScintillatorPVT;
188 }
189
190
191 G4VPhysicalVolume* DetectorConstruction::ConstructVolumes()
192 {
193   G4GeometryManager::GetInstance()->OpenGeometry();
194   G4PhysicalVolumeStore::GetInstance()->Clean();
195   G4LogicalVolumeStore::GetInstance()->Clean();
196   G4SolidStore::GetInstance()->Clean();
197
198   G4bool checkOverlaps = false;
199
200
201   // Set visAttributes for various components
202   G4VisAttributes* invisibleVisAtt = new G4VisAttributes( false );
203   G4VisAttributes* wireframeVisAtt = new G4VisAttributes();
204   wireframeVisAtt->SetForceWireframe( true );
205
206   // %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %
207   // ---->>> WORLD <<<----
208   // %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %
209
210   fWorldSizeX = fWorldSizeZ = 9450.*CLHEP::mm;
211   fWorldSizeY = 4450.*CLHEP::mm;
212   fWorldMaterial = 0;
213
```

```
fRoomSizeX = 8000.*CLHEP::mm;
fRoomSizeY = 3400.*CLHEP::mm;
fRoomSizeZ = 8400.*CLHEP::mm;

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// ---->>> BEAM LINE <<<----
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


G4GDMLParser parser;
parser.Read("masterGDMLfile.gdml", true);
G4VPhysicalVolume* pWorld = parser.GetWorldVolume();
pWorld->GetLogicalVolume()->SetVisAttributes(wireframeVisAtt);
G4LogicalVolume* lWorld = pWorld->GetLogicalVolume();
//lWorld->SetVisAttributes( invisibleVisAtt );


// ------------------------------------------------------------------------------------------
// Particle source
// Position relative to world: z = -4200.0*mm
// - The GerealParticleSource is confined to this volume in PrimaryGeneratorAction.cc
// ------------------------------------------------------------------------------------------

sSource = new G4Tubs( "Source", 0., 30*CLHEP::mm, 1*CLHEP::mm/2, 0, 2.*CLHEP::pi );
lSource = new G4LogicalVolume( sSource, fVacuum, "Source" );
```

```
239  pSource = new G4PVPlacement( 0, G4ThreeVector(0., 0., -4199*CLHEP::mm
      ), lSource,
240                               "Source", lWorld, false, 0,
      checkOverlaps );
241  lSource->SetVisAttributes( invisibleVisAtt );
242
243
244  //
      ------------------------------------------------------------------------

245  // Detector
246  // Position set in macro through DetectorMessenger
247  // - Padded volume to allow for wrapping around absorber etc.
248  //
      ------------------------------------------------------------------------

249
250  sDetector = new G4Box( "Detector", detSizeXY/2, detSizeXY/2, detSizeZ
      /2 );
251  lDetector = new G4LogicalVolume( sDetector, fAir, "Detector" );
252  pDetector = new G4PVPlacement( 0, fDetPosition, lDetector, "Detector"
      , lWorld, false, 0, checkOverlaps );
253  lDetector->SetVisAttributes( invisibleVisAtt );
254
255  // Absorber
256  // Placed in centre of detector volume
257  sAbsor = new G4Box( "Absorber", fAbsorSizeXY/2, fAbsorSizeXY/2,
      fAbsorSizeZ/2 );
258  lAbsor = new G4LogicalVolume( sAbsor, fWater, "Absorber" );
259  pAbsor = new G4PVPlacement( 0, G4ThreeVector(0., 0., 0.), lAbsor, "
      Absorber", lDetector, false, 0, checkOverlaps );
260  lAbsor->SetVisAttributes( invisibleVisAtt );
261
```

```
262   // Layers in absorber for segmented detector
263   if (fLayerNumber > 0) { // fLayerNumber set in macro
264     fLayerSizeZ = fAbsorSizeZ/fLayerNumber;
265
266     G4Box* sLayer = new G4Box( "Layer", fLayerSizeXY/2, fLayerSizeXY/2,
       fLayerSizeZ/2 );
267     lLayer = new G4LogicalVolume( sLayer, fWater, "Layer" );
268     pLayer = new G4PVReplica( "Layer", lLayer, lAbsor, kZAxis,
      fLayerNumber, fLayerSizeZ );
269
270     G4double layerVolume = fLayerSizeXY*fLayerSizeXY*fLayerSizeZ;
271     G4double layerDensity = fAbsorMaterial->GetDensity();
272     fLayerMass = layerVolume*layerDensity;
273   }
274
275   PrintParameters();
276
277   // Always return the World volume
278   return pWorld;
279 }
280
281 void DetectorConstruction::PrintParameters()
282 {
283   // Print parameters when the detector is constructed
284
285   G4cout << *( G4Material::GetMaterialTable() ) << G4endl;
286   G4cout << "\n
     ----------\n";
287   G4cout << "---> The Absorber is " << G4BestUnit(fAbsorSizeZ,"Length")
288          << " of " << fAbsorMaterial->GetName() << G4endl;
289   G4cout << "\n
     ----------\n";
290 }
```

```
291
292 void DetectorConstruction::SetMaterial( G4String materialChoice )
293 {
294   // Used by DetectorMessenger to set absorber material to that chosen
      in macro
295
296   G4Material* material = G4NistManager::Instance()->FindOrBuildMaterial
      ( materialChoice );
297
298   if ( material ) {
299     fAbsorMaterial = material;
300
301     if ( lAbsor ) {
302       lAbsor->SetMaterial( fAbsorMaterial );
303       lLayer->SetMaterial( fAbsorMaterial );
304
305       G4RunManager::GetRunManager()->PhysicsHasBeenModified();
306     }
307
308   }
309 }
310
311 void DetectorConstruction::SetSizeZ( G4double value )
312 {
313   fAbsorSizeZ = value;
314   G4RunManager::GetRunManager()->GeometryHasBeenModified();
315 }
316
317 void DetectorConstruction::SetSizeXY( G4double value )
318 {
319   fAbsorSizeXY = value;
320   G4RunManager::GetRunManager()->GeometryHasBeenModified();
321 }
```

```
322
323 void DetectorConstruction::SetLayerSizeXY( G4double value )
324 {
325   fLayerSizeXY = value;
326   G4RunManager::GetRunManager()->GeometryHasBeenModified();
327 }
328
329 void DetectorConstruction::SetLayerNumber( G4int value )
330 {
331   fLayerNumber = value;
332   G4RunManager::GetRunManager()->GeometryHasBeenModified();
333 }
334
335 void DetectorConstruction::UpdateGeometry()
336 {
337   G4RunManager::GetRunManager()->PhysicsHasBeenModified();
338   G4RunManager::GetRunManager()->DefineWorldVolume( ConstructVolumes()
      );
339 }
```

A.3 Master GDML file

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gdml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://service-spi.web.cern.ch/
    service-spi/app/releases/GDML/schema/gdml.xsd">

    <materials>
        <!--           -->
  <!-- elements -->
        <!--           -->
        <!-- http://www-cdf.fnal.gov/~kirby/lbne_geo_tests/
    lbne_10kT_Materials.gdml -->
        <element name="videRef"     formula="VACUUM" Z="1"> <atom value=
    "1."/>        </element>
        <element name="hydrogen"    formula="H"    Z="1">    <atom value=
    "1.0079"/>    </element>
        <element name="carbon"      formula="C"    Z="6">    <atom value=
    "12.0107"/>  </element>
        <element name="nitrogen"    formula="N"    Z="7">    <atom value=
    "14.0067"/>  </element>
        <element name="oxygen"      formula="O"    Z="8">    <atom value=
    "15.999"/>   </element>
        <element name="sodium"      formula="Na"   Z="11">   <atom value=
    "22.99"/>    </element>
        <element name="magnesium"   formula="Mg"   Z="12">   <atom value=
    "24.305"/>   </element>
        <element name="aluminium"    formula="Al"  Z="13">   <atom value
    ="26.9815"/>  </element>
        <element name="silicon"     formula="Si"   Z="14">   <atom value=
    "28.0855"/>  </element>
        <element name="phosphorus" formula="P"     Z="15">   <atom value=
    "30.973"/>   </element>
```

```
19      <element name="sulphur"    formula="S"   Z="16">  <atom value=
    "32.065"/>  </element>
20      <element name="argon"      formula="Ar"  Z="18">  <atom value=
    "39.9480"/> </element>
21      <element name="potassium"  formula="K"   Z="19">  <atom value=
    "39.0983"/> </element>
22      <element name="calcium"    formula="Ca"  Z="20">  <atom value=
    "40.078"/>  </element>
23      <element name="titanium"   formula="Ti"  Z="22">  <atom value=
    "47.867"/>  </element>
24      <element name="chromium"   formula="Cr"  Z="24">  <atom value=
    "51.9961"/> </element>
25      <element name="iron"       formula="Fe"  Z="26">  <atom value=
    "55.8450"/> </element>
26      <element name="nickel"     formula="Ni"  Z="28">  <atom value=
    "58.6934"/> </element>
27      <element name="copper"     formula="Cu"  Z="29">  <atom value=
    "63.55"/>   </element>
28      <element name="germanium"  formula="Ge"  Z="32">  <atom value=
    "72.63"/>   </element>
29      <element name="bromine"    formula="Br"  Z="35">  <atom value=
    "79.904"/>  </element>
30      <element name="aurum"      formula="Au"  Z="79">  <atom value=
    "196.97"/>  </element>
31      <element name="tungsten"   formula="W"   Z="74">  <atom value=
    "183.84"/>  </element>
32      <element name="bismuth"    formula="Bi"  Z="83">  <atom value=
    "208.980"/> </element>
33      <element name="lithium"    formula="Li"  Z="3">   <atom value=
    "6.941"/>   </element>
34      <element name="boron"      formula="B"   Z="5">   <atom value=
    "10.81"/>   </element>
35      <element name="cobalt"     formula="Co"  Z="27">  <atom value=
```

```
"58.933"/>   </element>
36          <element name="zinc"        formula="Zn"  Z="30">   <atom value=
"65.38"/>    </element>
37          <element name="europium"    formula="Eu"  Z="63">   <atom value=
"151.964"/>  </element>
38          <element name="caesium"     formula="Cs"  Z="55">   <atom value=
"132.905"/>  </element>
39
40
41          <!--                    -->
42  <!-- composite elements -->
43          <!--                    -->
44
45          <isotope name="B10"  N="5" Z="5">   <atom unit="g/mole" value="
10.0129369"/>  </isotope>
46          <isotope name="B11"  N="6" Z="5">   <atom unit="g/mole" value="
11.0093054"/>  </isotope>
47          <element name="B">
48              <fraction n="0.199" ref="B10"/>
49              <fraction n="0.801" ref="B11"/>
50          </element>
51
52
53
54
55          <!--          -->
56          <!-- vacuum   -->
57          <!--          -->
58          <material formula=" " name="Vacuum">
59              <D value="1.e-25" unit="g/cm3" />
60              <fraction n="1.0" ref="videRef" />
61          </material>
62
```

```
63
64
65          <!--            -->
66  <!-- materials -->
67          <!--            -->
68
69          <!-- FR4 submaterials -->
70
71          <material name="EpoxyResin" formula="C38H40O6Br4">
72              <D value="1.1250" unit="g/cm3"/>
73              <composite n="38" ref="carbon"/>
74              <composite n="40" ref="hydrogen"/>
75              <composite n="6" ref="oxygen"/>
76              <composite n="4" ref="bromine"/>
77          </material>
78
79          <material name="SiO2" formula="SiO2">
80              <D value="2.2" unit="g/cm3"/>
81              <composite n="1" ref="silicon"/>
82              <composite n="2" ref="oxygen"/>
83          </material>
84
85          <material name="Al2O3" formula="Al2O3">
86              <D value="3.97" unit="g/cm3"/>
87              <composite n="2" ref="aluminium"/>
88              <composite n="3" ref="oxygen"/>
89          </material>
90
91          <material name="Fe2O3" formula="Fe2O3">
92              <D value="5.24" unit="g/cm3"/>
93              <composite n="2" ref="iron"/>
94              <composite n="3" ref="oxygen"/>
95          </material>
```

```
96
97      <material name="CaO" formula="CaO">
98          <D value="3.35" unit="g/cm3"/>
99          <composite n="1" ref="calcium"/>
100         <composite n="1" ref="oxygen"/>
101     </material>
102
103     <material name="MgO" formula="MgO">
104         <D value="3.58" unit="g/cm3"/>
105         <composite n="1" ref="magnesium"/>
106         <composite n="1" ref="oxygen"/>
107     </material>
108
109     <material name="Na2O" formula="Na2O">
110         <D value="2.27" unit="g/cm3"/>
111         <composite n="2" ref="sodium"/>
112         <composite n="1" ref="oxygen"/>
113     </material>
114
115     <material name="TiO2" formula="TiO2">
116         <D value="4.23" unit="g/cm3"/>
117         <composite n="1" ref="titanium"/>
118         <composite n="2" ref="oxygen"/>
119     </material>
120
121     <material name="FibrousGlass">
122         <D value="2.74351" unit="g/cm3"/>
123         <fraction n="0.600" ref="SiO2"/>
124         <fraction n="0.118" ref="Al2O3"/>
125         <fraction n="0.001" ref="Fe2O3"/>
126         <fraction n="0.224" ref="CaO"/>
127         <fraction n="0.034" ref="MgO"/>
128         <fraction n="0.010" ref="Na2O"/>
```

```
129             <fraction n="0.013" ref="TiO2"/>
130         </material>
131 
132         <material name="FR4">
133             <D value="1.98281" unit="g/cm3"/>
134             <fraction n="0.47" ref="EpoxyResin"/>
135             <fraction n="0.53" ref="FibrousGlass"/>
136         </material>
137 
138 
139         <!-- Glue (DC3140, Dow Corning) sub materials -->
140 
141         <material name="dimethylsiloxane_hydroxy_terminated" formula="
    HOSiCH3CH3OH">
142             <D value="0.98" unit="g/cm3"/>
143             <composite n="2" ref="oxygen"/>
144             <composite n="8" ref="hydrogen"/>
145             <composite n="2" ref="carbon"/>
146             <composite n="1" ref="silicon"/>
147         </material>
148 
149         <material name="trimethylated_silica" formula="O2Si">
150             <D value="2.6" unit="g/cm3"/>
151             <composite n="2" ref="oxygen"/>
152             <composite n="1" ref="silicon"/>
153         </material>
154 
155         <material name="methyltrimethoxysilane" formula="C4H12O3Si">
156             <D value="0.955" unit="g/cm3"/>
157             <composite n="3"  ref="oxygen"/>
158             <composite n="12" ref="hydrogen"/>
159             <composite n="4"  ref="carbon"/>
160             <composite n="1"  ref="silicon"/>
```

```
161       </material>
162
163       <material name="DC3140">
164           <D value="1.2" unit="g/cm3"/>
165           <fraction n="0.60" ref="dimethylsiloxane_hydroxy_terminated"
    />
166           <fraction n="0.30" ref="trimethylated_silica"/>
167           <fraction n="0.10" ref="methyltrimethoxysilane"/>
168       </material>
169
170
171       <!-- Conductive materials for PCB -->
172
173       <material name="Copper" state="solid">
174           <D value="8.960" unit="g/cm3"/>
175           <fraction n="1." ref="copper"/>
176       </material>
177
178       <material name="Gold" state="solid">
179           <D value="19.32" unit="g/cm3"/>
180           <fraction n="1." ref="aurum"/>
181       </material>
182
183
184       <material name="Nickel" state="solid">
185           <D value="8.96" unit="g/cm3"/>
186           <fraction n="1." ref="nickel"/>
187       </material>
188
189
190       <!-- Kapton -->
191
192       <material name="Kapton" state="solid">
```

```
193         <D value="1.42" unit="g/cm3"/>
194         <fraction n="0.0273" ref="hydrogen"/>
195         <fraction n="0.7213" ref="carbon"/>
196         <fraction n="0.0765" ref="nitrogen"/>
197         <fraction n="0.1749" ref="oxygen"/>
198     </material>
199
200
201
202     <!-- aluminium, Hineycomb, Carbon fibre, etc. (simple materials)
    -->
203
204     <material formula="Al" name="Aluminium" state="solid">
205         <D value="2.700" unit="g/cm3"/>
206         <fraction n="1." ref="aluminium"/>
207     </material>
208
209     <material name="CarbonFibre" state="solid">
210         <D unit="g/cm3" value="0.145"/>
211         <fraction n="1.0" ref="carbon"/>
212     </material>
213
214     <material name="Honeycomb" state="solid">
215         <D value="0.030" unit="g/cm3"/>
216         <fraction n="1." ref="aluminium"/>
217     </material>
218
219     <!-- Silicon -->
220     <material name="Silicon" state="solid">
221         <D value="2.333" unit="g/cm3"/>
222         <fraction n="1." ref="silicon"/>
223     </material>
224
```

```
225     <!-- Tungsten -->
226     <material name="Tungsten" state="solid">
227         <D value="19.25" unit="g/cm3"/>
228         <fraction n="1." ref="tungsten"/>
229     </material>
230 
231     <!-- PMT -->
232     <!--sylgard 170, Silicon Rubber Polydimethylsiloxane(PDMS)-->
233     <material name="PDMS" formula="SiOC2H6" >
234         <D value="1.34" unit="g/cm3" />
235         <composite n="1" ref="silicon" />
236         <composite n="1" ref="oxygen" />
237         <composite n="2" ref="carbon" />
238         <composite n="6" ref="hydrogen" />
239     </material>
240 
241     <material name="Glass" formula="SiO2" >
242         <D value="2.5" unit="g/cm3" />
243         <composite n="1" ref="silicon" />
244         <composite n="2" ref="oxygen" />
245     </material>
246 
247     <material name="PMT" state="solid">
248        <D value="2.524" unit="g/cm3"/>
249        <fraction n="0.7" ref="aluminium"/>
250        <fraction n="0.2" ref="Glass"/>
251        <fraction n="0.1" ref="PDMS"/>
252     </material>
253 
254     <!-- BGO -->
255     <material name="BGO" formula="Bi4Ge3O12" >
256        <D value="7.13" unit="g/cm3" />
257        <composite n="4" ref="bismuth" />
```

```
258         <composite n="3" ref="germanium" />
259         <composite n="12" ref="oxygen" />
260       </material>
261
262
263     <!-- BC254 (neutron detector) -->
264     <material name="BC254" state="solid">
265         <MEE unit="eV" value="173"/>
266         <D unit="g/cm3" value="1.026"/>
267         <fraction n="0.2492" ref="carbon"/>
268         <fraction n="0.7475" ref="hydrogen"/>
269         <fraction n="0.0033" ref="B"/>
270     </material>
271
272     <!-- M. Hentz Material definitions parsed from Geant4 code -->
273
274     <material name="Air">
275         <D value="1.290" unit="mg/cm3"/>
276         <fraction n="0.7810" ref="nitrogen"/>
277         <fraction n="0.2096" ref="oxygen"/>
278         <fraction n="0.0094" ref="argon"/>
279     </material>
280
281     <material name="Water">
282         <D value="1." unit="g/cm3"/>
283         <composite n="2" ref="hydrogen" />
284         <composite n="1" ref="oxygen" />
285         <MEE unit="eV" value="78.0"/>
286     </material>
287
288     <material name="Scintillator_PVT">
289         <D value="1.023" unit="g/cm3"/>
290         <composite n="9" ref="carbon" />
```

```
291         <composite n="10" ref="hydrogen" />
292     </material>
293 
294     <material name="MarbleConcrete">
295         <D value="2.7" unit="g/cm3"/>
296         <fraction n="0.473942" ref="calcium"/>
297         <fraction n="0.375" ref="oxygen"/>
298         <fraction n="0.105" ref="carbon"/>
299         <fraction n="0.024" ref="silicon"/>
300         <fraction n="0.01" ref="iron"/>
301         <fraction n="0.007" ref="aluminium"/>
302         <fraction n="0.003" ref="magnesium"/>
303         <fraction n="0.002" ref="sodium"/>
304         <fraction n="0.000037" ref="lithium"/>
305         <fraction n="0.000018" ref="cobalt"/>
306         <fraction n="0.000002" ref="caesium"/>
307         <fraction n="0.000001" ref="europium"/>
308     </material>
309 
310     <material name="Brass">
311         <D value="8.75" unit="g/cm3"/>
312         <fraction n="0.7" ref="copper"/>
313         <fraction n="0.3" ref="zinc"/>
314     </material>
315 
316     <material name="KaptonMH">
317         <D value="1.42" unit="g/cm3"/>
318         <fraction n="0.027" ref="hydrogen"/>
319         <fraction n="0.691" ref="carbon"/>
320         <fraction n="0.073" ref="nitrogen"/>
321         <fraction n="0.209" ref="oxygen"/>
322     </material>
323 
```

```
324     <material name="Iron">
325         <D value="7.874" unit="g/cm3"/>
326         <fraction n="1" ref="iron"/>
327     </material>
328 
329     <material name="PMMA">
330         <D value="1.18" unit="g/cm3"/>
331         <composite n="5" ref="carbon" />
332         <composite n="2" ref="oxygen" />
333         <composite n="8" ref="hydrogen" />
334     </material>
335 
336     <material name="Mylar">
337         <D value="1.397" unit="g/cm3"/>
338         <composite n="10" ref="carbon" />
339         <composite n="8" ref="hydrogen" />
340         <composite n="4" ref="oxygen" />
341     </material>
342 
343     <material name="BoratedPlastic">
344         <D value="1.04" unit="g/cm3"/>
345         <fraction n="0.05" ref="boron"/>
346         <fraction n="0.317" ref="carbon"/>
347         <fraction n="0.633" ref="hydrogen"/>
348     </material>
349 
350   </materials>
351 
352   <solids>
353       <box lunit="mm" name="world_solid" x="9450.0" y="4450.0" z="
    9450.0" />
354   </solids>
355 
```

```
356     <structure>
357         <volume name="world_volume">
358             <materialref ref="Air"/>
359             <solidref ref="world_solid"/>
360 
361             <physvol>
362                 <file name="AluminiumComponents.gdml"/>
363              </physvol>
364             <physvol>
365                 <file name="BoratedPlasticComponents.gdml"/>
366              </physvol>
367             <physvol>
368                 <file name="BrassComponents.gdml"/>
369              </physvol>
370             <physvol>
371                 <file name="IronComponents.gdml"/>
372              </physvol>
373             <physvol>
374                 <file name="KaptonMHComponents.gdml"/>
375              </physvol>
376             <physvol>
377                 <file name="MarbleConcreteComponents.gdml"/>
378              </physvol>
379             <physvol>
380                 <file name="MylarComponents.gdml"/>
381              </physvol>
382             <physvol>
383                 <file name="PMMAComponents.gdml"/>
384              </physvol>
385             <physvol>
386                 <file name="TungstenComponents.gdml"/>
387              </physvol>
388             <physvol>
```

```
389                 <file name="VacuumComponents.gdml"/>
390             </physvol>
391
392
393         </volume>
394     </structure>
395
396     <setup name="Default" version="1.0">
397         <world ref="world_volume"/>
398     </setup>
399 </gdml>
```

B Additional and oversized figures

B.1 Aging and Risk of Severe, Disabling, Life-Threatening, and Fatal Events in the Childhood Cancer Survivors

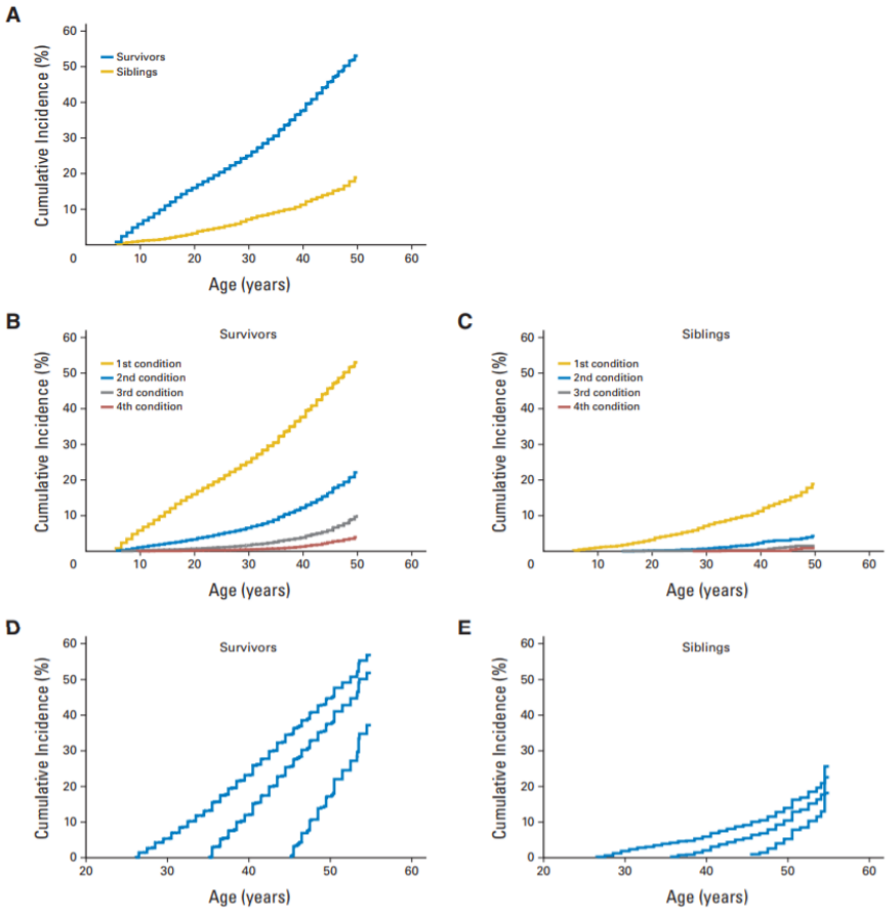


Figure 28: Cumulative incidence of chronic health conditions for (A) grades 3 to 5 chronic health conditions, (B) multiple grade 3 to 5 conditions in survivors, (C) multiple grade 3 to 5 conditions in siblings, (D) conditioned based on no previous grade 3 to 5 conditions among survivors by ages 25, 35, or 45, and (E) conditioned based on no previous grade 3 to 5 conditions among siblings by ages 25, 35, or 45. From [44].

Acknowledgments

I would like to first and foremost give my warmest thanks to my supervisor, Dr. Simon Jolly, for his patience, support, and mentoring throughout the project. Thank you to the entirety of the PBT group for helping me through the project and providing direction and guidance when it was most needed. Thank you to D. Walker for his support in programming, and motivational talks without which I would have not had the determination to complete the project. Finally, I would like to thank both D. Walker and A. Morris for their unrivalled friendship and the late night melee that got me through this more than anything else.

References

[1] Vasita Patel. *Deaths registered in England and Wales (series DR) [Online].* 2017. URL: https://www.ons.gov.uk/file?uri=/peoplepopulationandcommunity/birthsdeathsandmarriages/deaths/datasets/deathsregisteredinenglandandwalesseriesdrreferencetables/2016/drtables16.xls (visited on 20/10/2017).

[2] Vasita Patel. Jan. 2018. URL: https://www.ons.gov.uk/file?uri=/peoplepopulationandcommunity/healthandsocialcare/conditionsanddiseases/datasets/cancerregistrationstatisticscancerregistrationstatisticsengland/2016/2016cancerregistrationsreferencetablesfinal.xls (visited on 14/03/2018).

[3] Damian R. Beil and Lawrence M. Wein. "Analysis and comparison of multimodal cancer treatments". In: *Mathematical Medicine and Biology: A Journal of the IMA* 18.4 (2001), pp. 343–376. DOI: 10.1093/imammb/18.4.343. eprint: /oup/backfile/content_public/journal/imammb/18/4/10.1093/imammb/18.4.343/2/180343.pdf. URL: +%20http://dx.doi.org/10.1093/imammb/18.4.343.

[4] ancer.Net Editorial Board. *https://www.cancer.net/navigating-cancer-care/how-cancer-treated/surgery/what-cancer-surgery.* July 2016. (Visited on 14/03/2018).

[5] Smith M.J. et al. Coffey J.C. Wang J.H. "Excisional surgery for cancer cure: therapy at a cost."

In: *Lancet Oncol.* 4.12 (Dec. 2003), pp. 760–768. URL: https://www.ncbi.nlm.nih.gov/pubmed/14662433 (visited on 14/03/2018).

[6] A. Coates et al. "On the receiving end–patient perception of the side-effects of cancer chemotherapy." In: *Eur. J. Cancer* 19.2 (Feb. 1983), pp. 203–208.

[7] J. B. Posner L.M. DeAngelis. *Neurologic Complications of Cancer.* Oxford University Press, Oct. 2011. Chap. 12, pp. 447–510.

[8] G. Giaccone H.M. Pinedo. "Chemotherapy". In: *The Lancet* S7-S9 (May 1997). DOI: http://dx.doi.org/10.1016/S0140-6736(97)90012-X.

[9] M. Goitein. In: *Radiation Oncology: A Physicist's-Eye View.* New York, NY: Springer New York, 2008. Chap. Proton Therapy in Water, pp. 211–245. ISBN: 978-0-387-72645-8. DOI: 10.1007/978-0-387-72645-8_10.

[10] D.E. Thrall. "Orthovoltage radiotherapy of acanthomatous epulides in 39 dogs". In: *Journal of the American Veterinary Medical Association* 184.7 (Apr. 1984), pp. 826–829. ISSN: 0003-1488. URL: http://europepmc.org/abstract/MED/6427160.

[11] James M. Slater. "From X-Rays to Ion Beams: A Short History of Radiation Therapy". In: *Ion Beam Therapy: Fundamentals, Technology, Clinical Applications.* Ed. by Ute Linz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 3–16. ISBN: 978-3-642-21414-1. DOI: 10.1007/978-3-642-21414-1_1. URL: https://doi.org/10.1007/978-3-642-21414-1_1.

[12] Julien Vignard, Gladys Mirey and Bernard Salles. "Ionizing-radiation induced DNA double-strand breaks: A direct and indirect lighting up".

In: *Radiotherapy and Oncology* 108.3 (2013), pp. 362–369. ISSN: 0167-8140. DOI: https://doi.org/10.1016/j.radonc.2013.06.013. URL: http://www.sciencedirect.com/science/article/pii/S0167814013002910.

[13] Rainer K Sachs et al. "DNA damage caused by ionizing radiation". In: *Mathematical biosciences* 112.2 (1992), pp. 271–303.

[14] Miral Dizdaroglu et al. "Free radical-induced damage to DNA: mechanisms and measurement1, 2". In: *Free Radical Biology and Medicine* 32.11 (2002), pp. 1102–1115.

[15] B. Kaina W.P. Roos. "DNA damage-induced cell death by apoptosis". In: *Trends in Molecular Medicine* (July 2006). DOI: 10.1016/j.molmed.2006.07.007. URL: http://dx.doi.org/10.1016/j.molmed.2006.07.007 (visited on 17/03/2018).

[16] N. J. Tarbell T. I. Yock. "Technology insight: proton beam radiotherapy for treatment in pediatric brain tumors". In: *Nature clinical practice. Oncology.* 1.2 (Dec. 2004), pp. 97–103.

[17] W.S. Browner et al. "The genetics of human longevity". In: *The American Journal of Medicine* (11 Dec. 2004), pp. 851–860. DOI: 10.1016/j.amjmed.2004.06.033. URL: http://dx.doi.org/10.1016/j.amjmed.2004.06.033 (visited on 17/03/2018).

[18] S.M. Bentzen et al. "The UK Standardisation of Breast Radiotherapy (START) Trial B of radiotherapy hypofractionation for treatment of early breast cancer: a randomised trial". In: *Lancet (London, England)* 371.9618 (Mar. 2008), pp. 1098–1107.

ISSN: 0140-6736. DOI: 10.1016/s0140-6736(08)60348-7.
URL: http://europepmc.org/articles/PMC2277488.

[19] J.C. Horiot et al. "Hyperfractionation versus conventional fractionation in oropharyngeal carcinoma: final analysis of a randomized trial of the EORTC cooperative group of radiotherapy".
In: *Radiotherapy and Oncology* 25.4 (1992), pp. 231–241. ISSN: 0167-8140. DOI: https://doi.org/10.1016/0167-8140(92)90242-M. URL: http://www.sciencedirect.com/science/article/pii/016781409290242M.

[20] *Principles of Proton Therapy [Online]*. 2016.
URL: http://www.proton-cancer-treatment.com/proton-therapy/principles-of-proton-therapy/ (visited on 20/10/2017).

[21] D. Letourneau et al.
"Multileaf collimator performance monitoring and improvement using semiautomated quality control testing and statistical process control".
In: *Medical Physics* 41 (2014).

[22] T. C. Ling et al. "Evaluation of normal tissue exposure in patients receiving radiotherapy for pancreatic cancer based on RTOG 0848".
In: *Journal of Gastrointestinal Oncology* 6.2 (2014). ISSN: 2219-679X.
URL: http://jgo.amegroups.com/article/view/3448.

[23] Robert R Wilson. "Radiological use of fast protons".
In: *Radiology* 47.5 (1946), pp. 487–491.

[24] C. Patrignani et al. "Review of Particle Physics".
In: *Chin. Phys.* C40.10 (2016). DOI: 10.1088/1674-1137/40/10/100001.

[25] R. M. Sternheimer.
"The Density Effect for the Ionization Loss in Various Materials".
In: *Phys. Rev.* 88 (4 Nov. 1952), pp. 851–859.

DOI: 10.1103/PhysRev.88.851.

URL: https://link.aps.org/doi/10.1103/PhysRev.88.851.

[26] T.W. Armstrong and R.G. Alsmiller. "An approximate density-effect correction for the ionization loss of charged particles".
In: *Nuclear Instruments and Methods* 82 (1970), pp. 289–290.
ISSN: 0029-554X. DOI: https://doi.org/10.1016/0029-554X(70)90365-4.
URL: http://www.sciencedirect.com/science/article/pii/0029554X70903654.

[27] Daniele Davino. "Theory, Design and Tests on a Prototype Module of a Compact Linear Accelerator for Hadrontherapy". In: (2002).

[28] J. Allison et al. "Recent developments in Geant4". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 835.Supplement C (2016), pp. 186–225.
ISSN: 0168-9002. DOI: https://doi.org/10.1016/j.nima.2016.06.125.
URL: http://www.sciencedirect.com/science/article/pii/S0168900216306957.

[29] A. Dell'Acqua A. Rimoldi. *ATLAS Detector Simulation with Geant4*.
URL: http://atlas-computing.web.cern.ch/atlas-computing/packages/simulation/geant4/geant4.html (visited on 19/03/2018).

[30] Vladimir Ivantchenko Daniel Elvira. *CMS Offline Software (CMSSW) Guide*.
URL:
https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideSimulation
(visited on 19/03/2018).

[31] Andreas Morsch. *Particle Transport and Detector Simulation*.
URL: http://alice-

offline.web.cern.ch/Activities/Simulation/ParticleTransport.html (visited on 19/03/2018).

[32] Nigel Wason. *The GAUSS Project*. URL: http://lhcbdoc.web.cern.ch/lhcbdoc/gauss/ (visited on 19/03/2018).

[33] Geant4 Collaboration. *Tracking and Physics: Physics Processes*. Dec. 2017. URL: http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/TrackingAndPhysics/physicsProcess.html (visited on 19/03/2018).

[34] Geant4 Collaboration. *Tracking and Physics: Tracking*. Dec. 2017. URL: http://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/ForApplicationDeveloper/html/TrackingAndPhysics/tracking.html (visited on 19/03/2018).

[35] Satoshi Tanaka.
Fukui Renderer DAWN (Drawer for Academic WritiNgs) [ONLINE].
Sept. 2010.
URL: http://geant4.kek.jp/GEANT4/vis/DAWN/About_DAWN.html (visited on 24/03/2018).

[36] *Particle therapy facilities in operation*. Mar. 2018.
URL: https://www.ptcog.ch/index.php/facilities-in-operation (visited on 21/03/2018).

[37] JM Sisterson.
"Clinical use of protons and ion beams from a world-wide perspective".
In: *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms* 40 (1989), pp. 1350–1353.

[38] DE Bonnett, A Kacperek and M Sheen.
"The clatterbridge proton therapy facility". In: *HEAVY PARTICLE THERAPY WORKSHOP (PTCOG/EORTC/ECNEU)*. 1990, p. 143.

[39] Edward R Siciliano et al. "Comparison of PVT and NaI (Tl) scintillators for vehicle portal monitor applications". In: *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 550.3 (2005), pp. 647–674.

[40] LE Roscoe et al. "Stereolithography interface specification".
In: *America-3D Systems Inc* 27 (1988).

[41] R. Chytracek et al. "Geometry Description Markup Language for Physics Simulation and Analysis Applications".
In: *IEEE Transactions on Nuclear Science* 53.5 (Oct. 2006), pp. 2892–2896. ISSN: 0018-9499. DOI: 10.1109/TNS.2006.881062.

[42] *GDML USER'S GUIDE Version 2.7 [Online]*.
URL: http://gdml.web.cern.ch/GDML/doc/GDMLmanual.pdf (visited on 20/03/2018).

[43] Andrii Tykhonov. *A light-weight tool for converting a CAD drawings into the GDML format. [ONLINE]*. Jan. 2018.
URL: https://github.com/tihonav/cad-to-geant4-converter#usage (visited on 25/03/2018).

[44] G. T. Armstrong et. al. "Aging and Risk of Severe, Disabling, Life-Threatening, and Fatal Events in the Childhood Cancer Survivor Study".
In: *Journal of Clinical Oncology* 32.12 (Apr. 2014).