Sub-second pencil beam dose calculation on GPU for adaptive proton therapy

Joakim da Silva[1,2,*], Richard Ansorge[1], Rajesh Jena[2]
[1]Cavendish Laboratory, University of Cambridge
[2]Department of Oncology, University of Cambridge

*Corresponding author:
jd491@cam.ac.uk
+44 1223 337010
BSS, Cavendish Laboratory
19 J J Thomson Avenue
Cambridge, CB3 0HE
United Kingdom

Abstract

Although proton therapy delivered using scanned pencil beams has the potential to produce better dose conformity than conventional radiotherapy, the created dose distributions are more sensitive to anatomical changes and patient motion. Therefore, the introduction of adaptive treatment techniques where the dose can be monitored as it is being delivered is highly desirable. We present a GPU-based dose calculation engine relying on the widely used pencil beam algorithm, developed for on-line dose calculation. The calculation engine was implemented from scratch, with each step of the algorithm parallelised and adapted to run efficiently on the GPU architecture. To ensure fast calculation, it employs several application-specific modifications and simplifications, and a fast scatter-based implementation of the computationally expensive kernel superposition step. The calculation time for a skull base treatment plan using two beam directions was 0.22 seconds on an Nvidia Tesla K40 GPU, whereas a test case of a cubic target in water from the literature took 0.14 seconds to calculate. The accuracy of the patient dose distributions was assessed by calculating the γ-index with respect to a gold standard Monte Carlo simulation. The passing rates were 99.2% and 96.7%, respectively, for the 3%/3 mm and 2%/2 mm criteria, matching those produced by a clinical treatment planning system.

Introduction

Adaptive proton therapy

Owing to the Bragg peak (BP), which is a direct result of the fundamental interactions between a beam of charged particles and the traversed matter, charged particle radiotherapy (RT) can offer better dose conformity than conventional RT using photons. Although not a new innovation (Wilson 1946), it has seen an increased interest in recent years, as demonstrated by the many proton and carbon ion RT centres recently opened or under construction around the world (PTCOG 2014). However, charged particle RT as a standard treatment is still developing, and centres often lack some of the auxiliary technologies that are standard in modern photon RT systems. In particular, to benefit fully from the better dose conformity offered, adaptive radiotherapy (ART) methods would need to be introduced in particle RT, and are thus the subject of much research. ART comprises a wide range of techniques, from selecting a "plan of the day" from a number of pre-calculated plans, through daily imaging and dose recalculation, to real-time motion detection and compensation during dose delivery. The aim of this work has been to develop a proton therapy dose calculation engine that is fast enough for on-line dose calculation, whilst maintaining similar accuracy to current clinical standard. Specifically, the ART applications considered here will be those where there is a need to repeatedly recalculate a dose distribution during the course of the dose delivery, without noticeable prolonging the delivery time. One such application would be four-dimensional (4D) dose reconstruction for pencil beam scanning (PBS) systems similar to what was proposed by Richter et al (2014). Rather than a retrospectively calculation, the idea would be to calculate the dose delivered by each energy layer, spill, or part thereof in the corresponding phase (e.g. given by monitoring the patient breathing) of a 4D computed tomography (CT) image, and map it back to the reference image before continuing the dose delivery. This would make it possible to monitor the progressive emergence of a motion-corrected dose distribution during treatment, which could be used for real-time detection, and potentially correction, of unacceptable motion artefacts. A different application of fast dose calculation would be in real-time interactive treatment planning, as described by Otto (2014). Here, the goal is to have a dose calculation engine that is fast enough that the dose distribution can be interactively manipulated by the clinician during treatment planning.

Proton therapy dose calculation

Monte Carlo (MC) simulation, where a beam is modelled as a collection of particles stochastically interacting with the surrounding matter, is the gold standard when calculating dose distributions for both conventional and charged particle RT. However, due to the statistical nature of MC simulations, a large number of particles need to be simulated to achieve results of acceptable accuracy. This is especially true when many different interactions are possible (as in the situation for charged particles), in which case the necessary calculation time has often been prohibitively long for practical applications (Jia et al 2012a). For this reason, a number of analytical algorithms have been developed for charged particle dose calculation. Many of these are variants on the pencil beam (PB) approach (Petti 1992, Hong et al 1996), which is widely used in clinical treatment planning. PB algorithms divide the fluence map of a beam or a field into a number of computational pencil beams (CPBs) and calculate the total dose as the sum of contributions from the CPBs. The dose calculation for each CPB can be broken down into two steps. First, the dose distribution along the central axis of the CPB is calculated by scaling the integral depth dose (IDD) curve containing the longitudinal dose profile of a beam of the considered energy stopping in water. Second, the central axis dose distribution is widened through a multiplication with a two-dimensional (2D), depth-dependent kernel, perpendicular to the beam direction, in order to account for beam divergence and multiple Coulomb scattering. This step will be referred to as the kernel superposition (KS) and is usually the most

computationally expensive step of a PB algorithm. A Gaussian function, or a combination of several Gaussians, is generally chosen as the kernel. The dose D to a point (x,y,z) given in the beam's eye view (BEV) Cartesian coordinate system with the beam direction parallel to the z-axis, is thus calculated as:

$$D(x,y,z) = \sum_i N_i \times I_{\mathrm{IDD}}(E_i, z_{\mathrm{WEPL},i}(z)) \times K(x-x_i, y-y_i, \sigma_i(E_i, z, z_{\sigma,i})) \qquad \text{Eq. 1}$$

The summation in Eq. 1 is over all CPBs i in a treatment plan, where N_i is the number of particles, or weight, of CPB i, I_{IDD} is the IDD, and K is a kernel describing the lateral extent of a CPB. E_i, x_i and y_i are, respectively, the initial energy and lateral coordinates of CPB i. (Note that in this notation the lateral position of two or more CPBs i and j may coincide as long as their initial energies are different, e.g. $x_i=x_j$ and $y_i=y_j$, as long as $E_i \neq E_j$.) $z_{\mathrm{WEPL},i}$ is the water-equivalent path length (WEPL) from the calculation stating depth (normally the patient surface), z_0, to the point z according to

$$z_{\mathrm{WEPL},i}(x_i, y_i, z) = \int_{z_0}^{z} S_{\mathrm{rel}}(x_i, y_i, z')\,\mathrm{d}z' \qquad \text{Eq. 2}$$

where S_{rel} denotes the linear stopping power (SP) ratio between the medium and water. σ_i in Eq. 1 is a parameter describing the width of CPB i, e.g. the standard deviation in the case of a Gaussian kernel. For a particular system, σ_i is usually dependent on the initial beam energy, the absolute depth z, and a line integral $z_{\sigma,i}$ along the CPB from z_0 to z. $z_{\sigma,i}$ accounts for the widening of the beam due to the different materials encountered along the CPB, and in a simple case $z_{\sigma,i} = z_{\mathrm{WEPL},i}$.

GPU dose calculation

With the stagnating increase in single-core processing power, recent years have seen a growing interest in many-core systems for speeding up computationally demanding tasks. Due to their low cost and high performance, the graphics processing unit (GPU) is likely the most popular such system, both in general and within the field of medical physics (Pratx and Xing 2011, Jia et al 2014). Initially developed for real-time rendering of three-dimensional (3D) scenes in computer games, modern GPUs with thousands of cores are now readily programmable through application programming interfaces (APIs) such as CUDA (Nvidia Corporation, Santa Clara, CA, USA) and OpenCL (Khronos Group, Beaverton, OR, USA). However, due to the very high level of parallelism and specialised hardware architecture of these systems, developing efficient GPU implementations of existing algorithms remains non-trivial. A substantial effort has gone into employing GPUs to speed up dose calculation for proton (as well as conventional) RT, with the majority of studies related to implementing MC methods (Jia et al 2012a). Despite considerable progress in decreasing calculation times, those reported in the literature remain relatively long: MC codes relying on realistic (but not complete) modelling of physical interactions needed 10 seconds to calculate a shallow energy layer of a patient case (Jia et al 2012b), while track repeating and simplified MC algorithms needed tens of seconds to achieve acceptable accuracy for full plans (Kohno et al 2011, Yepes et al 2010). Although this is sufficient for daily dose recalculation, dose monitoring or motion compensation during treatment would require faster calculation times by about two orders of magnitude. Despite the development of faster GPUs since these studies were published (e.g. the base performance of our GPU is estimated to be 4.2–4.9 times greater compared to those in the mentioned studies), there are two reasons to believe that such radical speedups of MC simulations will not be seen in the near future. First, since the accuracy of a MC based calculation is directly linked to the number of simulated particles, the computational burden will inevitably remain large. Second, the GPU architecture does not lend itself well to MC algorithms, due to the high levels of branching, synchronisation, and scattered memory accesses required, which makes it very challenging for such implementations to take full advantage of the available computational power (Jia et al 2012a). The PB

algorithm, on the other hand, is less computationally demanding and has a higher degree of inherent parallelism, which makes it a promising candidate for GPU implementation. Despite this, only a partial GPU implementation of a PB algorithm for charged particle RT has been found in the literature. In their paper, Fujimoto et al (2011) present an implementation where the computationally demanding KS step is carried out on a GPU, whereas all other steps are left to the central processing unit (CPU). Although showing some speedup compared to a single-threaded CPU implementation, they predict that better performance can be expected from an implementation running entirely on a GPU. To the best of our knowledge, we are the first to describe a PB implementation for proton RT to run all algorithm steps on a GPU. Through a novel scatter-based implementation of the KS operation, application and GPU specific optimisations and simplifications, and efficient use of GPU resources, we produce considerable speedups of the PB dose calculation. Importantly, the achieved calculation times show that on-line dose calculation using a standard PB algorithm is indeed feasible during the course of treatment delivery.

Methods

Pencil beam algorithm

The dose calculation engine presented here is based on the PB implementation described by Soukup et al (2005), but contains several simplifications and adaptations to suit both the intended ART applications and GPU implementation. Descriptions of the main parts of the algorithm, highlighting the modifications introduced, are given in the following sub-sections. The description assumes that dose is delivered by PBS, although the general methods apply also to passively scattered protons. To avoid confusion between the CPBs and physical PBs produced by the treatment delivery system, the physical PBs will hereafter be referred to as "spots". A collection of spots with the same energy and delivered from the same beam direction are further referred to as an energy layer.

Coordinate system

In the presented PB implementation, dose is calculated in a right-handed BEV coordinate system with its origin at the isocentre and the *z*-axis pointing towards the beam source, as shown in Figure 1. We use Δz to denote the constant step length along the *z* coordinate when ray tracing CPBs through the computed tomography (CT) image of the patient, and $z_n = z_0 - n\Delta z$ to denote the *z* coordinate at step *n* from the starting depth z_0. For non-divergent CPBs, Δx and Δy denote the CPB spacing in the *x*- and *y*-directions of an orthogonal system, resulting in a dose volume made up of voxels of size $\Delta x \times \Delta y \times \Delta z$ mm^3. For divergent CPBs, a coordinate system is chosen such that the *x*- and *y*-coordinates remain constant along any CPB and the *z*-coordinate coincides with that of the non-divergent system (Figure 1). Transformations between the divergent and orthogonal BEV coordinate systems are given by

$$\begin{cases} x_{\text{div}} = \dfrac{d_x}{d_x - z_{\text{ort}}} x_{\text{ort}} \\ y_{\text{div}} = \dfrac{d_y}{d_y - z_{\text{ort}}} y_{\text{ort}} \\ z_{\text{div}} = z_{\text{ort}} \end{cases} \qquad \begin{cases} x_{\text{ort}} = \dfrac{d_x - z_{\text{div}}}{d_x} x_{\text{div}} \\ y_{\text{ort}} = \dfrac{d_y - z_{\text{div}}}{d_y} y_{\text{div}} \\ z_{\text{ort}} = z_{\text{div}} \end{cases} \qquad \text{Eq. 3}$$

where d_x and d_y denote, respectively, the source distances along the *z*-axis in the *xz*- and *yz*-planes. We define Δx and Δy in a divergent system to be the CPB spacing at $z=0$, making the orthogonal voxel grid a special case of a divergent voxel grid with $d_x=d_y=\infty$. In the rest of this paper, we therefore assume coordinates and voxels to be given in a general divergent system. In a divergent voxel grid, the physical distances between voxel centres along the *x*- and *y*-axes are given by $\Delta x_n = \Delta x(1 - z_n/d_x)$ and $\Delta y_n = \Delta y\left(1 - z_n/d_y\right)$, respectively (Figure 1). The voxels take the shape of truncated wedges, with all voxels at step *n* having identical volume given by

$$V_n = \Delta x \Delta y \Delta z \left(1 - \frac{z_n}{d_x} - \frac{z_n}{d_x} + \frac{z_n^2}{d_x d_y} + \frac{(\Delta z)^2}{12 d_x d_y}\right) \qquad \text{Eq. 4}$$

(where the last term can generally be disregarded). The physical distance between voxel centres along any CPB will be $\Delta z\sqrt{1 + x^2/d_x^2 + y^2/d_y^2}$ rather than Δz, but although this was included in the WEPL calculation, the difference will likely be negligible for many realistic set-ups.

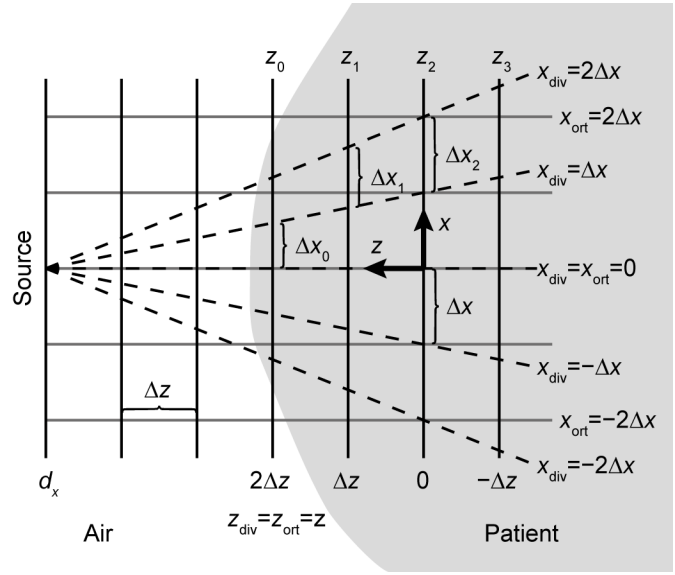

Figure 1. The orthogonal and divergent coordinate systems. To simplify the illustration, only the *x*- and *z*-coordinates are shown and the source distance has been greatly reduced. Lines of constant *z*, which coincide for the two systems, are shown as solid black lines with the corresponding coordinates and step numbers at the bottom and top, respectively. Lines of constant *x* are shown as solid grey and dashed black lines, respectively, for the orthogonal and divergent systems, with the corresponding coordinates on the right.

Single spot MC simulations

MC simulations of single spots stopping in water, used as the input to the PB implementation and to validate the single spot accuracy, were obtained using the Fluka MC code (Ferrari et al 2005, Böhlen et al 2014). The nozzle geometry and the parameters of the beams entering the nozzle (i.e. list of accelerator energies, momentum spread, and full width at half maximum) used in the simulation were provided by the CNAO treatment centre. Simulations using the same parameters have been shown to accurately reproduce the results obtained from dosimetric measurements and were used both in the commissioning of the treatment centre and as input to their analytical treatment planning system (TPS) used clinically (Rossi 2011, Mairani et al 2013). In total, spots of 147 beam energies corresponding to BP depths of 30–319 mm in water were simulated and used to determine the input parameters to the PB implementation.

Ray tracing and longitudinal CPB dose

For each step along a CPB, the mass density at the centre of the step and the WEPL at the distal edge of the step are calculated by ray tracing through the patient CT volume. The mass density is required for the radiation length calculation described in the following sub-section and if calculating approximate dose to medium rather than dose to water. In the discrete case, the WEPL at the distal edge of a voxel is given by

$$z_{\mathrm{WEPL},i}(x_i, y_i, z_{n+1/2}) = \sum_{k=0}^{n} S_{\mathrm{rel}}(x_i, y_i, z_k)\,\Delta z \qquad \text{Eq. 5}$$

The conversion from local Hounsfield units (HU), as obtained through tri-linear interpolation for each point along the CPBs, to mass density and relative SP was based on work by Schneider et al (1996, 2000). Specifically, the HU to relative SP conversion was the one optimised for head and neck cases employed at Centro Nazionale di Adroterapia Oncologica (CNAO) treatment centre in Pavia, Italy (adjusted to have air mapped to HU=−1000 to suit our implementation). Generally, S_{rel} in Eq. 5 has a weak dependency on the residual particle energy. However, the effect of this dependency on particle range was deemed small enough to be ignored, and thus the suggested energy-dependent formula by Fippel and Soukup (2004) was not applied. By assuming that the SP relative to water is not dependent on energy, the WEPL calculation becomes identical for all energy layers, which means that it needs to be evaluated only once for each beam direction.

The IDDs for scanning spots of different energies impinging on a water tank were derived from the single spot MC simulations. Since the IDDs have regions of rapidly varying derivative (e.g. around the BP), directly sampling the IDD as in Eq. 1 may lead to local under- or overestimation of the dose when discretising z. Therefore, cumulative IDDs (CIDDs), given by

$$I_{\mathrm{CIDD}}(E_i, z) = \int_0^z I_{\mathrm{IDD}}(E_i, z')\,\mathrm{d}z' \qquad \text{Eq. 6}$$

were used in the calculation. As long as the numerical integration of Eq. 6, which can be done offline, is carried out with a sufficiently small step, substituting

$$I_{\mathrm{IDD}}(E_i, z_{\mathrm{WEPL},i}(z_n)) \longrightarrow \frac{\left[I_{\mathrm{CIDD}}\left(E_i, z_{\mathrm{WEPL},i}(z)\right)\right]_{z=z_{n-1/2}}^{z=z_{n+1/2}}}{S_{\mathrm{rel}}(x_i, y_i, z_n)\,\Delta z} \qquad \text{Eq. 7}$$

in Eq. 1 solves this problem regardless of the size of Δz.

Lateral CPB model

There are two main approaches to modelling the lateral dose distribution in the PB algorithm. Either the kernel K in Eq. 1 is obtained directly from the dose distribution of a scanning spot or passively scattered field (measured or simulated) at the corresponding depth in water. Alternatively, its shape is given by an analytical kernel whose width at each depth is determined by the individual CPB history. In both cases, a scaling with the physical distance to the beam source can be incorporated to account for the beam divergence in air. Despite being able to reproduce exactly any beam shape in water, the first approach may produce less accurate results in the presence of heterogeneities (Szymanowski and Oelfke 2002). Therefore, the second approach was used in the presented implementation, employing a single Gaussian kernel, given by

$$K_{\mathrm{Gauss}}(r_x, r_y, \sigma) = \frac{1}{2\pi\sigma^2}\exp\left(\frac{-(r_x^2 + r_y^2)}{2\sigma^2}\right) \qquad \text{Eq. 8}$$

where σ is the standard deviation of the kernel and r_x and r_y are the respective distances between the CPB and the evaluation point along the x- and y-axes. Analogous to the substitution in Eq. 7, the accuracy when evaluating Eq. 8 over a discrete grid can be improved by replacing a direct function evaluation with an integral difference:

$$\frac{1}{2\pi\sigma^2}\exp\left(\frac{-(r_x^2 + r_y^2)}{2\sigma^2}\right) \longrightarrow k_{\mathrm{erf}}(r_x, \sigma)\,k_{\mathrm{erf}}(r_y, \sigma) \qquad \text{Eq. 9}$$

where

$$k_{\mathrm{erf}}(r_w,\sigma) = \frac{1}{\Delta w}\int_{r_w-\Delta w/2}^{r_w+\Delta w/2}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{-r^2}{2\sigma^2}\right)\mathrm{d}r = \left[\frac{1}{2\Delta w}\mathrm{erf}\left(\frac{r}{\sqrt{2}\sigma}\right)\right]_{r_w-\Delta w/2}^{r_w+\Delta w/2} \qquad \text{Eq. 10}$$

In the above equations, erf is the error function, $w \in \{x, y\}$, and $\Delta w \in \{\Delta x_n, \Delta y_n\}$ is the local voxel spacing at step *n*. Since the Gaussian function has infinite support, a cut-off has to be chosen beyond which the function value is regarded small enough to be neglected. For the convolution and KS operations in this paper, a cut-off of a minimum of 3σ was used in accordance with Fujimoto et al (2011).

Using a single Gaussian function to model the lateral dose implicitly neglects contributions from large angle scattering events including nuclear interactions. Many improvements to the single Gaussian beam model have been presented, most of which add the contribution from a second, wider Gaussian, e.g. by letting the kernel be a sum of two Gaussians or by adding a separate step for calculating the contribution from nuclear interactions. However, algorithms employing a single Gaussian beam model are still used for clinical dose calculation, e.g. Schaffner et al (1999), and the model was deemed adequate for a proof-of-principle implementation as presented here.

Assuming small scattering angles, the kernel variance, σ^2, can be modelled as a sum of squared contributions from independent sources. Specifically, we (explicitly) consider here only contributions from the inherent spot divergence in air, σ_{air}, and from multiple Coulomb scattering inside the patient, σ_{MS}. The variance in air as a function of *z*-position for spots of each initial energy, $\sigma_{\mathrm{air}}(E,z)^2$, was obtained from the single spot MC simulations and was seen to be adequately approximated by second order polynomials in *z* (expected from a beam in vacuum) in the region around the isocentre. The variance due to multiple scattering at step *n* along each CPB, $\sigma_{\mathrm{MS}}(E,z_n)^2$, was calculated by summing the contributions from the characteristic angle, θ_k, at each previous step *k* according to

$$\sigma_{\mathrm{MS}}(E,z_n)^2 = \sum_{k=0}^{n-1}\theta_k^2(z_n - z_k)^2 \qquad \text{Eq. 11}$$

(Soukup et al 2005). In the presented implementation, the characteristic angle was calculated according to the Rossi formula

$$\theta^2 = \left(\frac{E_S}{\beta p}\right)^2\frac{\Delta z}{X_0(x,y,z)} \qquad \text{Eq. 12}$$

where E_S is a constant energy parameter and X_0 is the radiation length (Rossi and Greisen 1941). The relativistic factor times the momentum found in the denominator is given by $\beta p = E_k + m_0 - m_0^2/(E_k + m_0)$, where m_0 is the proton rest mass expressed in MeV and the mean residual energy E_k of a CPB can be calculated according to the formula by Bortfeld (1997). The radiation length at a given point was calculated from the local mass density according to Fippel and Soukup (2004). When comparing results with single spot MC simulations (including contributions from all physical interactions) E_S=14.1 MeV was seen to reproduce most accurately the spot shape in water across the considered energies.

It should be noted that the sum in Eq. 11 does not have to be explicitly evaluated for each step along a CPB. To see this, let us first consider a simplified case where $\theta_k = \theta$ for all values of *k*. Then, Eq. 11 can be written $\theta^2\varphi_n^2(\Delta z)^2$, where $\varphi_n^m = 1^m + 2^m + \cdots + n^m$ is the sum of the *m*-th powers of the first *n* integers, by some referred to as the *m*-th degree snurkel of *n*. We note that $\varphi_n^2 = \varphi_{n-1}^2 + n^2$ and, in turn, that $n^2 = 2\varphi_n^1 - \varphi_n^0$, where $\varphi_n^1 = \varphi_{n-1}^1 + \varphi_n^0$ and $\varphi_n^0 = \varphi_{n-1}^0 + 1$. In the real case where θ_k depends on *k*, Eq. 11 can similarly be written as $\psi_n^2(\Delta z)^2$, where $\psi_n^m = 1^m\theta_{n-1}^2 + 2^m\theta_{n-2}^2 + \cdots +$

$n^m\theta_0^2$ for $n>0$ and 0 otherwise. Expanding this expression in the same way as the simplified case gives

$$\begin{cases} \psi_n^2 = \psi_{n-1}^2 + (2\psi_n^1 - \psi_n^0) \\ 2\psi_n^1 - \psi_n^0 = 2\psi_{n-1}^1 + \psi_n^0 = \left(2\psi_{n-1}^1 - \psi_{n-1}^0\right) + \psi_{n-1}^0 + \psi_n^0 \\ \psi_n^0 = \psi_{n-1}^0 + \theta_{n-1}^2 \end{cases} \qquad \text{Eq. 13}$$

Thus, to calculate ψ_n^2 at step n we only need the three values ψ_{n-1}^2, $(2\psi_{n-1}^1 - \psi_{n-1}^0)$, and ψ_{n-1}^0 from the previous step. In our implementation, these are kept as temporary variables between steps so that, using $\sigma_{\mathrm{MS}}(E, z_n)^2 = \psi_n^2(\Delta z)^2$ and Eq. 13, all σ_{MS} up to $n=N$ are calculated using in total $4N$ addition and N multiplication operations. For large values of N, this is a considerable improvement compared to the $(N^2+N)/2$ addition and N^2+N multiplication operations required for explicit evaluation.

CPB subdivision

To make the implementation as fast as possible, rather than calculating the dose contribution from each spot in an energy layer individually, the total contribution from all spots in an energy layer is calculated simultaneously (Schaffner et al 1999). Before the dose calculation starts, the depth at which any CPB first enters the patient, z_0, is found (Figure 1). The fluence map of energy layer j in air at depth z_0 is obtained by convolving the spot weight map with a Gaussian function of variance $\sigma_{\mathrm{air}}(E_j,z_0)^2$, and the CPB weights are calculated by resampling the fluence map according to the CPB grid at this depth. From this point, the width of each CPB, given by its associated path-dependent standard deviation σ_{tot}, develops independently according to

$$\sigma_{\mathrm{tot}}(E, z) = \sqrt{\sigma_{\mathrm{MS}}(E, z)^2 + \sigma_{\mathrm{air}}(E, z)^2 - \sigma_{\mathrm{air}}(E, z_0)^2} + \delta \qquad \text{Eq. 14}$$

δ in Eq. 14 is an empirical term added to the standard deviation to account for a small difference seen between the analytically calculated and MC simulated beam widths. The difference was seen for spots of all energies in water at all depths except for in the region close to the entry point. The mean value across the different energy spots, δ=0.21 mm, was used for all energies.

GPU implementation

A schematic overview of the dose calculation engine is shown in Figure 2. It comprises several components, each consisting of a separate GPU program, referred to as a kernel function (KF), implemented in the CUDA C/C++ programming language. Descriptions of the main KFs are given in the following paragraphs. In addition, auxiliary code is responsible for allocating and initialising memory on the GPU; copying data between the CPU and the GPU; launching KFs; and calculating additional input to the KFs, e.g. coordinate system transforms, offsets, and number of calculation steps for different energies. The auxiliary functions run either on the CPU or on the GPU, where CPU code was used where required by the API or for calculations small enough not to benefit from running on a GPU. It should be noted that none of the BEV volumetric intermediates, (i.e. WEPL, mass density, CPB depth dose, CPB variance, and final dose) are copied between the CPU and GPU during the whole computation. This considerably limits the amount of data transferred over the low-bandwidth system bus, which is a common bottleneck in GPU programming. Due to their graphics rendering legacy, GPUs are designed to achieve best performance when working with single-precision floating-point arithmetic, and thus all (non-integer) calculations in the presented implementation were carried out in single precision. No part of the calculation where small errors would accumulate was identified, and thus no difference in the biologically relevant dose range is expected compared to a double-precision calculation.

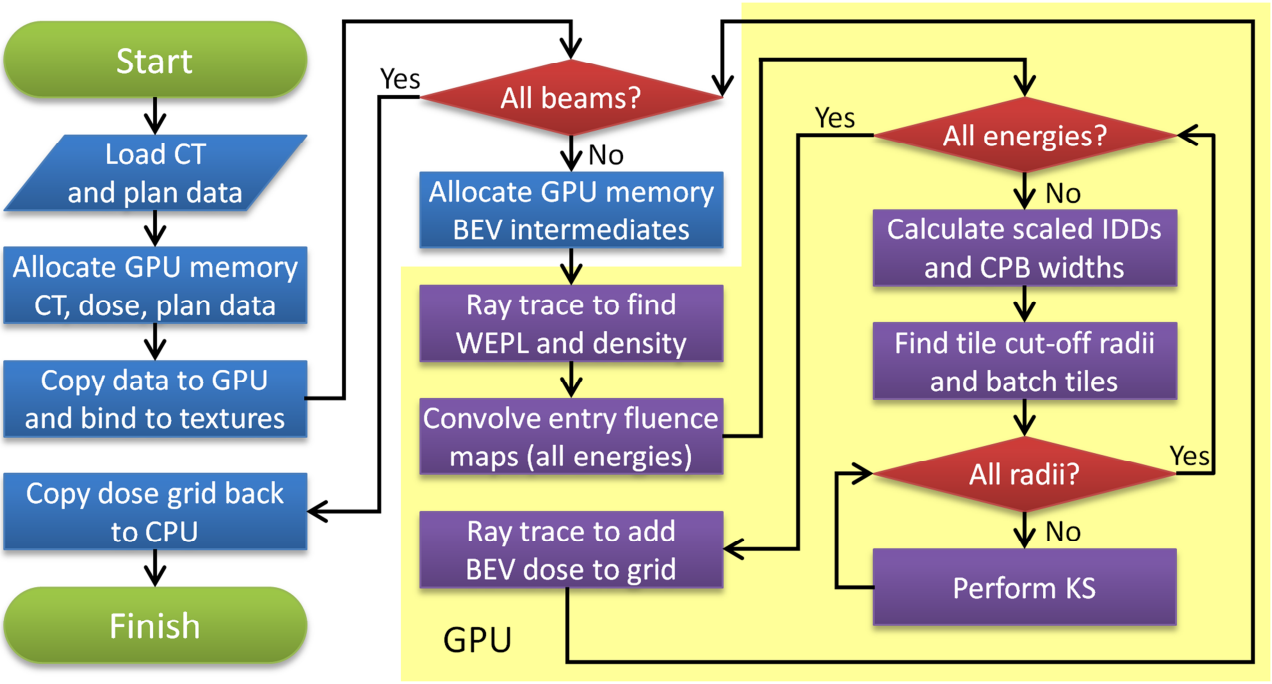

Figure 2. Flow chart of the presented PB dose calculation engine. Blue boxes represent retrieval of input data, memory allocations, and memory transfers, and purple boxes inside the shaded area marked GPU represent the main KFs executed on the GPU.

Ray tracing and fluence map calculation

The first KF handles the ray tracing and is called once for each beam direction. Threads are assigned one per CPB according to their spatial layout in the BEV coordinate system, and all CPBs are traced from where the first one enters the CT volume to where the last one exits. For each step, the WEPL and mass density is obtained from the local HU through look-up tables in texture memory, and stored in two separate arrays in global memory according to the spatial layout given by the BEV voxel grid. This ensures coalesced memory accesses, meaning that adjacent threads access adjacent memory locations, in this and all following steps, which is necessary to achieve good performance. It further constitutes the implicit transformation from the patient to the BEV coordinate system for a given beam direction. During the ray tracing, the step number at which each CPB enters and exits the patient is determined by comparing the local HU with a threshold value. Before the ray tracing KF finishes execution, this information stored in global memory to be used in the subsequent calculation steps to limit the amount of redundant dose calculation in the air outside the patient. The CT volume serving as input for the KF is stored in 3D texture memory, which allows us to take advantage of the spatial layout and the free linear interpolation of the texture cache. After the ray tracing, a small KF is called which calculates the fluence maps at the entry depth and assigns corresponding weights to the CPBs. Although this could be done individually for each energy layer, the CPB weights for all energy layers of a given beam direction are calculated simultaneously to ensure that the GPU is saturated.

IDD scaling and CPB width calculation

Once the CPB weights have been calculated the rest of the calculation is done per energy layer. A KF calculates the WEPL-scaled IDD and σ_{tot} (Eq. 14) at each step along the CPBs. As input the KF takes the WEPL and mass density arrays from the previous steps, as well as the CPB weights and the step numbers of the first and last step inside the patient, as calculated by auxiliary functions. The scaled IDD values and values of $\left(\sqrt{2}\sigma_{\text{tot}}\right)^{-1}$, the latter compensated for the voxel width at the given depth,

are stored in global memory arrays. If an approximation to dose to medium rather than dose to water is desired, Srel in the denominator of the right-hand side of equation (8) is replaced by the local mass density (Paganetti 2009). To avoid redundant calculations in the computationally expensive KS step, the status of the individual CPBs is checked at each step. A CPB is considered live if it is inside the patient, has not reached the end of its range, and has a particle number of at least one. The dose and variance at steps where a CPB is not live are both set to zero.

Kernel superposition

Before calling the KS KF, the BEV volume is divided into a large number of *xy*-tiles and a KF that calculates the cut-off radius corresponding to the largest value of σ_{tot} in each tile is launched. To ensure that the GPU is saturated, the KS is then carried out simultaneously for all tiles in the BEV volume that have the same cut-off radius. The KS algorithm used employs a scatter-based approach, described in detail elsewhere (da Silva et al 2015), instead of the conventional gather-based approach. Briefly, the KF assigns one thread to each input voxel and scatters the calculated results to the neighbouring voxels in the output (rather than assigning threads to the output voxels and gathering the results from neighbouring input voxels). The advantage of the scatter approach is that for separable kernels, such as the 2D Gaussian function, the number of kernel evaluations can be significantly reduced, which in turn can speed up the calculation considerably. However, to avoid having multiple threads trying simultaneously to write to the same memory location, a non-intrusive way of synchronising the threads is required. This was achieved by keeping the output from each tile in shared memory and relying on the lock-step execution of threads within a warp to reduce the number of explicit synchronisations necessary. To allow for loop unrolling and compile time optimisations, the KS KF was implemented as a template function, with the tile cut-off radius, expressed as an integer number of voxels, as the template parameter. Each call to the KS KF takes as input a list of tiles with the considered cut-off radius and their corresponding IDD values and variance from the previous steps, and outputs the corresponding final dose in the BEV system.

Transformation to global dose grid

At the end of the dose calculation for each beam direction, a KF analogous to the ray tracing KF transforms the dose calculated in the BEV system back to the global dose grid. This is done by first binding the BEV dose to 3D texture memory and then ray tracing along the dose grid *z*-coordinate whilst interpolating into the BEV dose volume and adding the contribution at each step to the global dose grid.

Validation and benchmarking

Single spot validation

Single spot validation of the PB implementation was carried out by comparing the radial dose distribution in water of three spots with BP depths of 70, 131, and 220 mm, with the corresponding MC simulations described previously. Additional MC simulations of the 131 mm range spot were carried out after introducing 30 mm thick slabs of air and cortical bone perpendicular to the beam direction from 50 to 80 mm depth in the water tank. The results were compared to the equivalent dose distributions produced by the PB dose calculation engine by comparing the spot central axis dose and width along the path. The resolution of the dose grid was set to $1\times1\times1$ mm^3 in all the calculations, using $\Delta x=\Delta y=\Delta z=1$ mm in the PB implementation.

Patient case validation

To evaluate the accuracy of the implemented dose calculation engine in a realistic setting, the planning CT image and plan files for a representative, 55.4 cm^3 planning tumour volume (PTV) skull base case were obtained from the CNAO treatment centre. The plan employs two oblique beam

directions of 38 and 45 energy layers, respectively, including a total of 6776 spots of 34–131 mm BP depth in water. MC dose distribution as simulated in Fluka in accordance with Mairani et al (2013), to be used as the ground truth, and a dose distribution produced by the commercial Syngo PT Planning VB10 (Siemens AG, Munich, Germany) TPS, to serve as a comparison, were also provided. Syngo PT Planning is based on the PB algorithm by Szymanowski and Oelfke (2002) and uses a double Gaussian beam model. For both the provided dose distributions, the resolution was 2×2×2 mm^3, which was also adopted for the dose calculation with the presented PB implementation, maintaining $\Delta x=\Delta y=\Delta z=1$ mm in the BEV calculation. The dose distribution from the presented PB calculation engine was evaluated directly against the MC ground truth by calculating the γ-indices according to Ju et al (2008) for the 3% (of prescription dose)/3 mm (distance to agreement) and 2%/2 mm criteria. It was also indirectly compared to the dose distribution produced by Syngo PT Planning by comparing γ-index passing rates. When calculating the passing rates, only non-air voxels (HU>−850) receiving at least 10% of the prescription dose were considered.

Benchmarking

Benchmarking was carried out for the same patient case as the validation. In addition, a plan for an artificial target consisting of a cube of side length 100 mm extending 100–200 mm below the surface of a water tank was created following Fujimoto et al (2011). 20 energy layers were used to cover the target and the calculation was done over a 256×256×256 voxel dose grid of resolution 1×1×1 mm^3, with $\Delta x=\Delta y=\Delta z=1$ mm (resulting in 128×128 CPBs). The calculation times reported for the complete plans are for all the calculation steps shown in Figure 2 (including memory allocation, transfers and deallocation, and the auxiliary functions not shown). However, it does not include initialisation of the GPU and the time to read data into CPU memory from the storage medium, which can both be done independent of the calculation. The per-energy layer times reported do not include memory transfers between the CPU and GPU, since in a setting where partial dose contributions are of interest it is assumed that all necessary data, e.g. a 4D CT image, are available before starting the treatment. They do, however, include the ray tracing and dose transformation steps, which would otherwise have to be carried out only once per beam direction, and which in the case of a single energy layer take up a non-negligible amount of the calculation time. The calculations were carried out on an Nvidia Tesla K40 GPU with 2880 cores at a clock frequency of 875 MHz.

Results

Single spot validation

Figure 3 shows the difference between the radial dose distributions for individual spots as calculated by the PB implementation and by Fluka. As expected, the single Gaussian beam model cannot account for the extended low-dose halo resulting from large-angle scattering events and nuclear interactions. Since the laterally integrated dose at each depth remains unchanged, this small understatement of the dose to a large volume away from the spot central axis results in a larger exaggeration of the dose to a small volume close to the central axis. Due to the larger number of nuclear interactions, this behaviour is more pronounced for higher initial energies, as can be seen from the bottom panel of Figure 3. Still, the agreement is within −1.1 to 5.3 percent of the reference max dose for BP depths up to about 220 mm, and, as is shown in the patient case validation, the overlap between spots in a real treatment leads to a much smaller error in the total dose distribution.

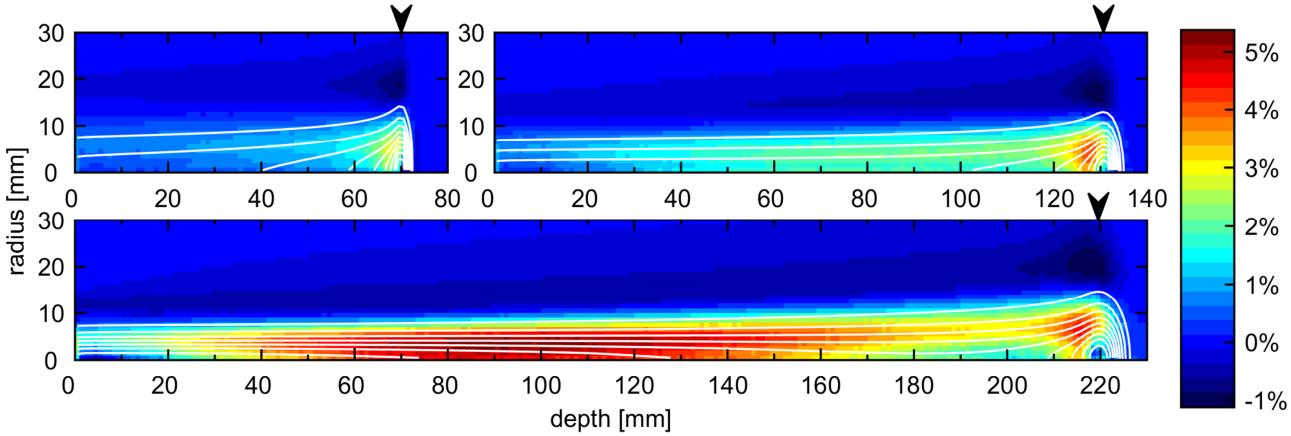

Figure 3. Difference between the dose for individual spots as calculated by the presented PB implementation and by Fluka, as a percentage of the maximum dose for each spot. The contours show the MC dose, with each line corresponding to 10% of the max dose. The BP depths of the three spots are 70.0 mm (top left), 130.7 mm (top right), and 219.6 mm (bottom), as indicated by black arrow heads.

Figure 4 compares the lateral dose distributions and central axis doses for the spot of intermediate energy in Figure 3, as calculated by Fluka and the presented PB implementation. In the case when no slab is present, very good agreement is seen both in the lateral distribution and central axis dose, with the PB implementation slightly overestimating both in the plateau region. The PB implementation further accurately captures the spot characteristics in both the considered slab geometries, with the central axis doses showing the same level of agreement as for the water-only case. The difference in lateral profile is slightly larger in the presence of the air slab, with the PB calculation consistently overestimating the spot standard deviation by about 0.1 mm downstream from the slab. A narrowing of the spot lateral profiles near the end of the particle range, as seen in the left panel of Figure 4, was seen for all energies. This was assumed to be due to particles being scattered at large angles stopping earlier, and thus leaving particles close to the central axis to travel further. In general, the spots were seen to be widest 1–2 mm before the BP. In the PB implementation, this behaviour was mimicked by ignoring the width calculation in Eq. 14 and instead subtracting an amount $\Delta\sigma^2_{\text{end}}$ from σ^2_{tot} for each step beyond or containing the BP. $\Delta\sigma^2_{\text{end}} = 3S_{\text{rel}}\Delta\sigma^2_{\text{BP}}/2$, where S is the local relative stopping power and $\Delta\sigma^2_{\text{BP}}$ is the increment of σ^2_{tot} just before the BP, was empirically determined to give a fair agreement with experimental data.

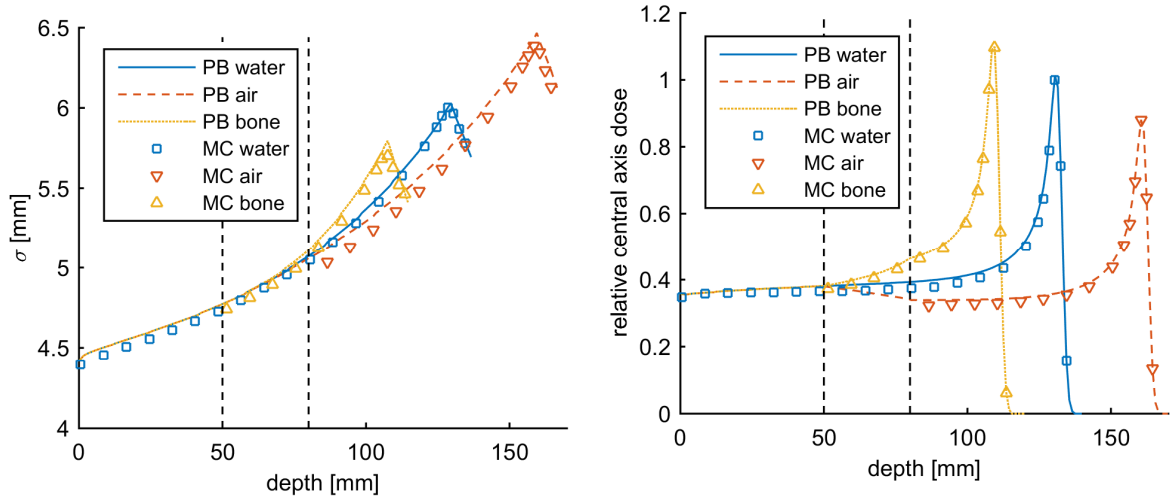

Figure 4. Lateral beam profiles (left) and central axis doses (right) for a spot with 130.7 mm BP depth in water passing though slabs of different materials in a water tank. The slab materials are water (i.e. no slab), air, and cortical bone, and the extent of the slab is indicated by the vertical dashed lines. Upstream of the slab all values are expected to be equal, and for clarity only data for the case of no slab is shown. Due to the high level of noise inside the air slab, MC data for this region is not shown, which can be justified since dose to air is generally not of interest in clinical dose calculation.

Patient case validation

2D slices through the centre of the patient dose distributions as calculated by Fluka and the PB algorithm are shown, respectively, in the top and middle rows of Figure 5. Qualitatively, the agreement is good in the in the high-dose region for all views. The PB dose shows fewer sharp details (e.g. in the axial slices) and, as expected, the low-dose region is smaller for the PB dose calculation (e.g. in the coronal slices). The bottom row of Figure 5 shows the γ-index map of the test case. Due to its higher detail content, only the map corresponding to the stricter 2%/2 mm criterion is displayed. γ-indices below one are seen for most voxels, with the exceptions mainly found close to air cavities or bony anatomy. The 3%/3 mm passing rate for the PB algorithm was 99.2%, decreasing to 96.7% for the 2%/2 mm criterion. The same passing rates for the clinical dose calculation produced by Syngo were 99.0% and 96.8%, respectively, indicating a similar accuracy in the high- and medium-dose regions.

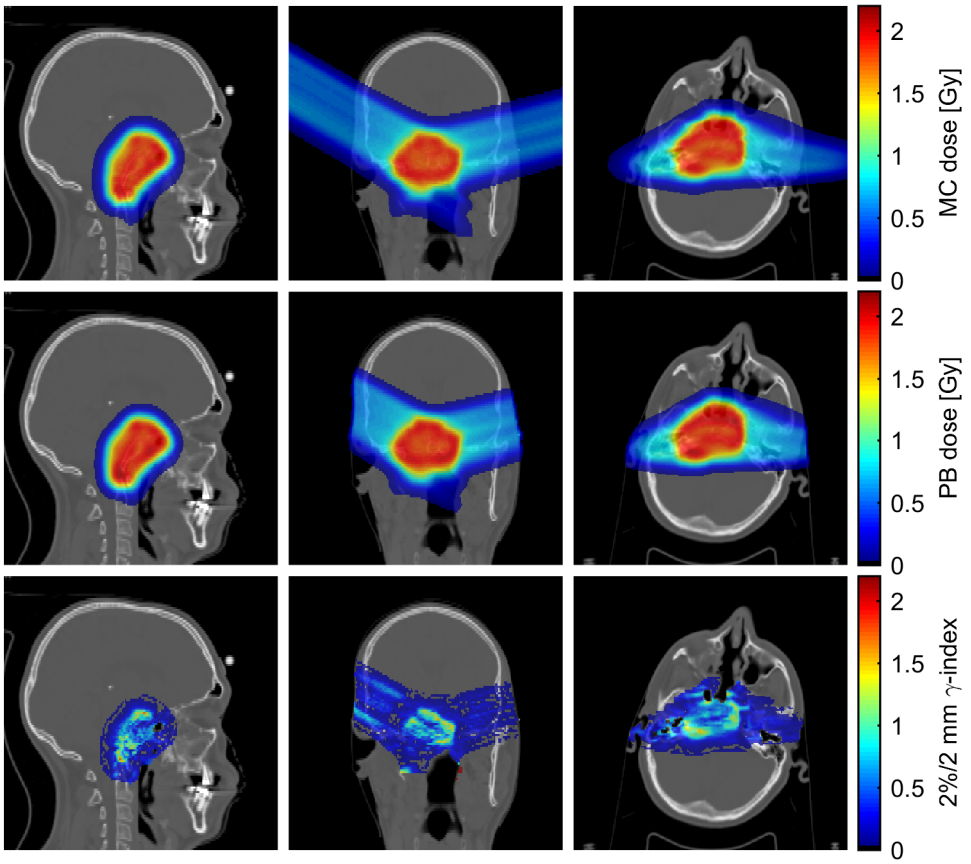

Figure 5. Sagittal (left column), coronal (central column), and axial views (right column) of the reference clinical case. The colour washes show the MC dose (top row), PB dose (middle row) and the 2%/2mm γ-indices (bottom row) for each view.

Benchmarking

The calculation time for the patient case was 0.22 seconds, with individual energy layers (excluding memory transfers between the CPU and GPU), taking between 2.2 and 6.4 ms to calculate. The same numbers for the cubic target in water were 0.14 seconds for the full plan and 5.9–13.0 ms for individual energy layers.

Discussion

The measured agreement between the presented PB implementation and the reference MC code is very good and comparable to that of a TPS in clinical use. The accuracy indicated by the γ-index passing rates should be sufficient for applications requiring on-line monitoring of dose conformity and detection of hotspots or interplay effects. However, the γ-index is a poor measure of agreement in the low-dose region, where a single Gaussian does not accurately account for the long-range, low-dose halo. Further, as indicated by Figure 3, the agreement with MC simulations is worse for higher beam energies, meaning that for plans with deep-seated tumours the γ-index passing rates might be slightly lower than presented here. Although the single Gaussian beam model is still used clinically (and much of our clinical experience is based on it), due to the above reasons, the trend is for TPSs to move towards more complex (e.g. multiple Gaussian) models. A natural extension to the work presented here, with the aim to improve the accuracy in the low-dose region, would thus be to implement the nuclear interaction correction introduced by Soukup et al (2005). Provided that the parameters for such a correction can be obtained, it can be implemented by carrying out a second calculation stage for the ‘nuclear’ CPBs, reusing the same workflow as described in Figure 2 (without the need to

repeat the memory transfers and ray tracing step). The kernel radii required for the nuclear correction are estimated to be about three times wider than those used for multiple scattering. However, if we follow Soukup et al (2005) and use only one nuclear CPB per treatment spot (and thus, in our case, let $\Delta x=\Delta y=3$ mm for this step), the CPB grid spacing grows by a similar amount, making the computational burden per CPB of the KS equal to the multiple scattering case. Using $\Delta x=\Delta y=3$ mm further reduces the number of CPBs needed for the calculation, and hence the overall computational burden, by a factor of 9. We therefore estimate that, as long as there is enough work to keep the GPU saturated, including a second Gaussian to account for nuclear interactions would add roughly 10% to the presented calculation times.

As expected from an analytical algorithm, the presented dose calculation engine is substantially faster than MC simulations: ignoring the difference in hardware, the calculation time for the clinical case presented here is two orders of magnitude, or more, shorter than what is reported for GPU MC codes. The simplified MC code by Kohno et al (2011), which is the fastest reported, required 19 and 130 seconds, respectively, to calculate the dose (from two beam directions) for head and neck cases with PTVs of 11.8 and 214.5 cm^3. Assuming that these calculation times scale with GPU performance and correcting for the estimated difference of 4.9 times (448 cores at 1150 MHz compared to 2880 cores at 875 MHz), this corresponds to their code requiring 18 times longer for a PTV of 0.2 time the size or 120 times longer for a PTV 3.9 times the size. The presented calculation times compare favourably also to the partial GPU implementation of a PB algorithm by Fujimoto et al (2011), where the computationally intensive KS step alone runs on a GPU. They reported a calculation time of 0.41 seconds (which was in turn faster than the corresponding CPU implementation) for the KS step of the deepest energy layer of the cubic target test case, when using a dose grid resolution of $1\times1\times1$ mm^3. The base performance of the GPU used in their study is estimated to be 3.7 times lower than the one used here (480 cores at 1401 MHz compared to 2880 cores at 875 MHz). Still, the 0.14 seconds reported here to complete all steps of all 20 energy layers of the plan (or 13.0 ms for the deepest energy layer excluding memory transfers) makes the presented implementation substantially faster, also when taking into account the difference in GPU performance. We attribute the difference to the novel, scatter-based implementation of the KS (da Silva et al 2015); to the fact that the presented calculation engine was built from scratch to suit the GPU architecture; and to all parts of the engine running on the GPU. The last point means that rather than having to copy the BEV intermediates between the CPU and GPU for each energy layer, these are kept in GPU memory throughout the calculation. Together, these points also ensure that calculation times will continue to decrease with the development of new hardware with an even higher degree of parallelism.

The price of using an analytical algorithm compared to a MC calculation is lower accuracy in heterogeneous volumes. A full MC code is expected to reproduce exactly the gold standard and, although the GPU MC codes discussed above all employ some level of simplification, they are assumed to produce higher γ-index passing rates than the presented PB implementation. This is especially true for the code implementing realistic interaction models (c.f. Jia et al (2012b)) which is also the slowest. Further, the use of a single Gaussian beam model in the presented implementation results in less information in the low-dose region. Nonetheless, the accuracy of the presented PB dose calculation engine is deemed sufficient for the intended ART applications. With a typical time between energy layers or spills of between 0.1 and 3 seconds, the calculation times per energy layer should also be short enough for on-line dose calculation (with time to spare for dose mapping and other necessary steps in an ART application). In case the delivery time of an energy layer is long, and the delivery therefore spans several motion phases, several calculations would have to be carried out per energy layer. In this case, the reported per-energy layer times should be regarded as upper limits

for the partial calculations, and it is clear that these are well below the period of any motion that could reasonably be compensated for. If even shorter calculation times would be required, however, and a reduction in accuracy can be tolerated, the PB algorithm could be replaced by a ray casting algorithm (Schaffner et al 1999). Ray casting algorithms rely mostly on the same steps as PB algorithms, and could thus be implemented reusing many of the components presented here, but circumvent the need for the expensive KS step, which should considerably shorten the calculation time.

A drawback of calculating the dose contribution per energy layer rather than spot by spot is that the calculation engine cannot be used in conventional plan optimisation, where knowledge of the dose contribution from each individual spot is required. Although this does not directly affect the intended application in ART systems, a fast calculation engine suitable for use in plan optimisation might be of interest. The per-spot contribution to the total dose could be calculated using the same approach as presented, simply by assigning each spot its own "energy layer" (c.f. Figure 2). However, to obtain better performance, the existing code would likely have to be reworked to suit this purpose. The performance increase when calculating the dose per energy layer compared to per spot stems from the lateral overlap between spots. This means that the weight of each CPB is the sum of contributions from several spots which would otherwise have required one CPB each (Schaffner et al 1999). By estimating this overlap we assume that calculating the per-spot dose contributions would increase the calculation time by a few tens of times compared to what has been presented here. However, if the dose calculation for optimisation purposes does not require as high resolution, the increase in calculation time could potentially be mitigated by lowering the resolution. The computational load of the dominating KS step scales with $((\Delta x)^2(\Delta y)^2\Delta z)^{-1}$, and thus, provided that there is enough work available to saturate the GPU, going from Δx=Δy=Δz=1 mm to Δx=Δy=Δz=2 mm could decrease the KS calculation time by a factor of 32. More exotic optimisation tasks, such as for intensity-modulated, passively scattered proton therapy (Sánchez-Parcerisa et al 2014), where entire fields have to be recalculated in each optimisation cycle, might directly benefit from the short calculation times of the presented calculation engine.

We have identified two additional implications resulting from the per-energy layer calculation in conjunction with our choice of coordinate system. First, the particles contributed by a spot to CPBs away from the spot's central axis when entering the patient, will be transported along the central axes of other spots, and thus diverge slightly from the parent spot direction. The resulting lateral shift is proportional to the longitudinal distance from z_0 (i.e. the depth inside the patient) and the lateral distance from the spot centre, and inversely proportional to the source distance. Therefore, it mainly affects the end of the beam range and the small weight contributions found far away from the spot central axis. Even for these, however, the effect was small; the maximum shift for any contribution in water ranged from 0.15 mm to 0.55 mm for spots with 30 mm and 319 mm BP depth, respectively. Second, since away from the *z*-axis the CPBs are not perfectly perpendicular to the *xy*-plane (Figure 1), an effective tilt of the kernel will be introduced when performing the KS (Sharpe and Battista 1993). The resulting longitudinal shift is proportional to the distance between the contributing CPB and the receiving voxel and to the distance between the contributing CPB and the *z*-axis, and inversely proportional to the source distance (which, in general, is considerably larger for proton than for photon RT). Therefore, analogous to before, the effect is largest at deep depths (because of larger associated σ_{tot}) and for the small dose contributions at the tails of the Gaussian kernel. The largest shift among CPBs of a 200×200 mm^2 field was still limited to between 0.10 and 0.58 mm for ranges of 30 to 319 mm respectively. Although the effect of both these shifts were considered small enough to ignore, it is noted that employing the coordinate system introduced by Lu (2010) limits the

effective kernel tilt at the expense of more complex expressions for the voxel volume and the coordinate transformations.

Although the focus of this paper has been on dose calculation for PBS systems, it is worth noting that the presented dose calculation engine could, with minor modifications to the beam model, be used to calculate dose for passively scattered protons. Similarly, a fast dose calculation engine for heavier ions, such as carbon, could be built using the same components as presented here, with the addition of one or more components for biological equivalent dose calculation.

Conclusion

We have developed a fast, GPU-based proton dose calculation engine, employing the widely used PB algorithm, for use in on-line dose calculation. For a representative skull base case, the calculation time was 0.22 seconds on a single GPU, with individual energy layers taking 2.2–6.4 ms to calculate. The γ-index passing rates of the resulting dose distributions matched those of a commercial treatment planning system. We conclude that dose calculation using the PB algorithm can be made fast enough for on-line ART applications whilst maintaining the accuracy of current clinical systems.

Acknowledgements

We would like to thank Mario Ciocca, Giuseppe Magro, Andrea Mairani, and Silvia Molinelli at CNAO (Pavia, Italy) for sharing data and models of the CNAO beam line, and for providing the patient case and dose distributions calculated with Fluka and Syngo. We would further like to thank Till Böhlen at MedAustron (Wiener Neustadt, Austria) for help with MC simulations and Andrea Attili at INFN (Turin, Italy) for insightful discussions on 4D treatment applications. This research was funded by the European Commission Seventh Framework People Programme through the ENTERVISION project, grant agreement 264552. Dr Jena is funded in part by Cancer Research UK, grant 13716. The Tesla K40 GPU used for benchmarking was donated by the Nvidia Corporation through their Hardware Grant Program.

References

Böhlen T T, Cerutti F, Chin M P W, Fassò A, Ferrari A, Ortega P G, Mairani A, Sala P R, Smirnov G and Vlachoudis V 2014 The FLUKA Code: Developments and Challenges for High Energy and Medical Applications *Nuclear Data Sheets* **120** 211–4

Bortfeld T 1997 An analytical approximation of the Bragg curve for therapeutic proton beams. *Medical physics* **24** 2024–33

Ferrari A, Sala P R, Fasso A and Ranft J 2005 FLUKA: a multi-particle transport code *CERN-2005-10, INFN/TC_05/11, SLAC-R-773*

Fippel M and Soukup M 2004 A Monte Carlo dose calculation algorithm for proton therapy. *Medical physics* **31** 2263–73

Fujimoto R, Kurihara T and Nagamine Y 2011 GPU-based fast pencil beam algorithm for proton therapy. *Physics in medicine and biology* **56** 1319–28

Hong L, Goitein M, Bucciolini M, Comiskey R, Gottschalk B, Rosenthal S, Serago C and Urie M 1996 A pencil beam algorithm for proton dose calculations. *Physics in medicine and biology* **41** 1305–30

Jia X, Pawlicki T, Murphy K T and Mundt A J 2012a Proton therapy dose calculations on GPU: advances and challenges *Translational Cancer Research* **1** 207–16

Jia X, Schümann J, Paganetti H and Jiang S B 2012b GPU-based fast Monte Carlo dose calculation for proton therapy. *Physics in medicine and biology* **57** 7783–97

Jia X, Ziegenhein P and Jiang S B 2014 GPU-based high-performance computing for radiation therapy. *Physics in medicine and biology* **59** R151–R182

Ju T, Simpson T, Deasy J O and Low D A 2008 Geometric interpretation of the γ dose distribution comparison technique: Interpolation-free calculation *Medical physics* **35** 879–87

Kohno R, Hotta K, Nishioka S, Matsubara K, Tansho R and Suzuki T 2011 Clinical implementation of a GPU-based simplified Monte Carlo method for a treatment planning system of proton beam therapy. *Physics in medicine and biology* **56** 287–94

Lu W 2010 A non-voxel-based broad-beam (NVBB) framework for IMRT treatment planning. *Physics in medicine and biology* **55** 7175–210

Mairani A, Böhlen T T, Schiavi A, Tessonnier T, Molinelli S, Brons S, Battistoni G, Parodi K and Patera V 2013 A Monte Carlo-based treatment planning tool for proton therapy *Physics in medicine and biology* **58** 2471–90

Otto K 2014 Real-time interactive treatment planning *Physics in medicine and biology* **59** 4845–59

Paganetti H 2009 Dose to water versus dose to medium in proton beam therapy. *Physics in medicine and biology* **54** 4399–421

Petti P L 1992 Differential-pencil-beam dose calculations for charged particles *Medical physics* **19** 137–49

Pratx G and Xing L 2011 GPU computing in medical physics: a review. *Medical physics* **38** 2685–97

PTCOG 2014 PTCOG: Particle Therapy Co-Operative Group Online: http://ptcog.web.psi.ch/

Richter D, Saito N, Chaudhri N, Härtig M, Ellerbrock M, Jäkel O, Combs S E, Habermehl D, Herfarth K, Durante M and others 2014 Four-Dimensional Patient Dose Reconstruction for Scanned Ion Beam Therapy of Moving Liver Tumors *International Journal of Radiation Oncology* Biology* Physics* **89** 175–81

Rossi B and Greisen K 1941 Cosmic-Ray Theory *Review of Modern Physics* **13** 240–309

Rossi S 2011 The status of CNAO *The European Physical Journal Plus* **126** 1–39

Sánchez-Parcerisa D, Kondrla M, Shaindlin A and Carabe A 2014 FoCa: a modular treatment planning system for proton radiotherapy with research and educational purposes. *Physics in medicine and biology* **59** 7341–60

Schaffner B, Pedroni E and Lomax A 1999 Dose calculation models for proton treatment planning using a dynamic beam delivery system: an attempt to include density heterogeneity effects in the analytical dose calculation. *Physics in medicine and biology* **44** 27–41

Schneider U, Pedroni E and Lomax A 1996 The calibration of CT Hounsfield units for radiotherapy treatment planning. *Physics in medicine and biology* **41** 111–24

Schneider W, Bortfeld T and Schlegel W 2000 Correlation between CT numbers and tissue parameters needed for Monte Carlo simulations of clinical dose distributions. *Physics in medicine and biology* **45** 459–78

Sharpe M and Battista J 1993 Dose calculations using convolution and superposition principles: The orientation of dose spread kernels in divergent x-ray beams *Medical physics* **20** 1685–94

Da Silva J, Ansorge R and Jena R 2015 Efficient scatter-based kernel superposition on GPU *(in review)*

Soukup M, Fippel M and Alber M 2005 A pencil beam algorithm for intensity modulated proton therapy derived from Monte Carlo simulations. *Physics in medicine and biology* **50** 5089–104

Szymanowski H and Oelfke U 2002 Two-dimensional pencil beam scaling: an improved proton dose algorithm for heterogeneous media. *Physics in medicine and biology* **47** 3313–30

Wilson R R 1946 Radiological use of fast protons *Radiology* **47** 487–91

Yepes P P, Mirkovic D and Taddei P J 2010 A GPU implementation of a track-repeating algorithm for proton radiotherapy dose calculations. *Physics in medicine and biology* **55** 7107–20