

# An Algorithm for the Reconstruction of Entrance Beam Fluence from Virtual Patient Exit Electronic Portal Images

Nicholas N. Sperling, E. Ishmael Parsai

Department of Radiation Oncology, University of Toledo Medical Center, Toledo, USA  
Email: [e.parsai@utoledo.edu](mailto:e.parsai@utoledo.edu)

Received 12 January 2015; accepted 20 May 2015; published 25 May 2015

Copyright © 2015 by authors and Scientific Research Publishing Inc.  
This work is licensed under the Creative Commons Attribution International License (CC BY).  
<http://creativecommons.org/licenses/by/4.0/>



Open Access

---

## Abstract

The problem of determining the *in vivo* dosimetry for patients undergoing radiation treatment has been an area of interest since the development of the field. More recent methods of measurement employ Electronic Portal Image Devices (EPID), or dosimeter arrays, for entrance or exit fluence determination. The more recent methods of *in vivo* dosimetry make use of detector arrays and reconstruction techniques to determine dose throughout the patient volume. One method uses an array of ion chambers located upstream of the patient. This requires a special hardware device and places an additional attenuator in the beam path, which may not be desirable. An alternative to this approach is to use the existing EPID, which is part of most modern linear accelerators, to image the patient using the treatment beam. Methods exist to deconvolve the detector function of the EPID using a series of weighted exponentials [1]. Additionally, this method has been extended to the deconvolution of the patient scatter in order to determine *in vivo* dosimetry. The method developed here intends to use EPID images and an iterative deconvolution algorithm to reconstruct the impinging primary fluence on the patient. This primary fluence may then be employed, using treatment time volumetric imaging, to determine dose through the entire patient volume. Presented in this paper is the initial discussion of the algorithm, and a theoretical evaluation of its efficacy using monte-carlo derived virtual fluence measurements. The results presented here indicate an agreement of 1% dose difference within 95% the field area receiving 10% of the entrance fluence for a set of sample highly modulated fields. These results warrant continued investigation in applying this algorithm to clinical patient treatments.

## Keywords

Beam Fluence, EPID Images, *In-Vivo* Dosimetry, Monte Carlo Algorithm, Dose Reconstruction

---

## 1. Introduction

The problem of determining the dose delivered to a patient post treatment for complex treatment delivery methods such as intensity modulated radiotherapy (IMRT) has been a focus of research for two decades now. Dosimetric verification of IMRT techniques is typically performed by applying special phantoms in which detectors such as diodes, ionization chambers, or film is positioned at a known depth to measure planar fluence, allowing comparison to computed planar fluence at the same depth generated by treatment planning system. Instead of employing an external detector, Electronic portal imaging devices (EPIDs) have been investigated for the purpose of treatment verification in terms of both accuracy and stability [2]-[4]. The latest technique, converts the image acquired by EPID from exit patient dose to an incident fluence distribution and uses this fluence distribution as input to a dose algorithm, which computes the dose to the patient [5]. Through an algorithm, this fluence is traced back through the patient to derive the incident fluence, which is then used to compute the dose to the patient. This paper describes an innovative method to reconstruct the entrance fluence from the exit fluence obtained from the EPID device.

## 2. Materials and Methods

The method proposed for dose reconstruction using exit portal imaging devices is as follows. A model of the accelerator system will be created in the BEAMnrcmontecarlo code to accurately model the accelerator to the Polymethylmethacrylate (PMMA) exit window of the accelerator. At this point, the model will produce a phase space file containing the position, energy, weight, particle type, and direction cosines of each particle arriving at this plane. The format of the phase space file is described in the BEAMnrc documentation [6]. It notes that in addition to the previously mentioned properties, each particle also has a 32 bit unsigned integer allocated for LATCH-bits. The LATCH bits are used in the BEAMnrc software to indicate regions in which the particle has interacted in the model.

The documentation intends for bits 1 - 22 to be used to store information about the particle interactions in the particle history. We may coopt this position information for the purpose of creating a 10x10-bit array with 2 bits extra for indicating if the particle is outside of the grid in either X or Y. By storing the position information in the LATCH bits, and using the correct simulation parameters, we can track the dose deposition from a particle origination in the entrance fluence. This allows us to create an image for each pixel in the input fluence. Each image is a 1024 × 1024-pixel grid representing particle density with a 64 bit floating point number.

Using these parameter space files we may produce an output fluence for any given input fluence in the simulated region. This will be used in our algorithm for the reconstruction of entrance fluence from exit fluence.

### 2.1. Fluence Calculation

The algorithm for calculating the output fluence from a given input fluence estimate involves the use of the parameter space created as described above. Equation (1), below, describes how the computed fluence ( $f_l$ ) is computed from a summation over the entire parameter space with weights ( $K(x', y')$ ). As the parameter space weight array is allowed to be of a higher resolution than the parameter space, an interpolation method is needed to compute the resultant fluence,

$$f_l(K) = \sum_{x', y'} K(x', y') \cdot [XYArr(x', y')] \quad (1)$$

The algorithm designed to interpolate from a higher resolution entrance fluence to a lower resolution parameter space involves treating each parameter space component as a separate mutable kernel which will be shifted to align with the entrance fluence position. The appropriate kernel is selected for each pixel in the entrance fluence by selecting the nearest parameter space array to the x and y position of the pixel to be calculated. The result is then shifted in the computed fluence plane by the ratio of pixel sizes times the offset from the center of the parameter space position of the kernel. This results in each input pixel of an arbitrary resolution entrance fluence producing an appropriate output at the exit fluence plane.

### 2.2. Fluence Solver

The method described here to solve for the entrance fluence given an exit fluence depends on the parameter

space described in section above. This parameter space is used to calculate the exit fluence from a given entrance fluence as described in 2.1 above. This exit fluence is then used as the calculated fluence at the level of the EPID. The difficulty then becomes determining the appropriate weights of the input matrix to produce a calculated fluence which matches with the measured (or simulated) fluence at the level of the EPID.

To solve for the entrance fluence, we will create a function to quantify the quality of the match, and use an iterative minimization solver to find the set of coefficients that minimize the quality function. The quality function we have decided upon is the square root of the sum of the square of the residuals of the two fluence matrices in Equation (2), below; where  $fl(x, y)$  is the input fluence at position  $(x, y)$ ; and,  $fl_c(x, y)$  is the calculated fluence at position  $(x, y)$ .

$$R^2 = \sum_{x,y} \left[ (fl(x, y) - fl_c(x, y))^2 \right] \quad (2)$$

The solver used is from a package called OpenOpt, a software package for Python [7] which has collected several solver packages together, including some packages written by the OpenOpt maintainers [8]. The package selected for use in this project was the Generalized epsilon-subgradient (gsubg) algorithm, part of the OpenOpt package implementing sub-solvers to seek local minima of the function. The “gsubg” solver was selected because it is one of the few large scale NLP solvers available in the OpenOpt package which allows one to supply a user defined derivative function. The ability to supply a user defined function for defining the gradient of the problem function allows us to significantly improve on calculation speed over a blind approach provided in the OpenOpt package.

### 2.2.1. Derivative Calculation Function

The derivative function is a finite difference calculation function designed to accept any user defined stencil for numerical analysis of the derivative. At current, three stencils have been implemented: a Newtonian quotient [8], a central two point stencil [8], and a five point stencil [9]; requiring 1, 2, and 4 additional calculations of the match quality respectively. For the highest accuracy, a default choice of the five point stencil was used, though evaluation of the need for this may be worthwhile. In order to speed calculation, the input fluence minus the previous full calculated fluence along with the current coefficients are provided to each worker process. The calculation is then performed according to Equation (3) below, where:  $flMpcfl$  is the input fluence minus the previous full calculated fluence; and  $fl_{c,S_v}$  is the differential fluence for the current  $x, y$  in the coefficient space with the addition of the stencil weight;  $S_v$  is the stencil difference addition; and  $S_m$  is the stencil weight, or multiplier,

$$\frac{dR^2(x, y)}{dx, dy} = \sum_{stencil} S_m \cdot \sum_{x, y} \left[ (flMpcfl - fl_{c,S_v}(x, y))^2 \right] \quad (3)$$

The advantage of this method is that we may calculate each  $x, y, S_m, S_v$  position separately, allowing us to massively parallelize the calculation and sum the components at the end.

### 2.2.2. Parameter-Space Weight Initial Solution

To enhance convergence, an initial entrance fluence is calculated by recasting the provided exit fluence onto a grid the size of the parameter space and used to initialize the solver. This first guess is used to create a computed exit fluence, then computing a normalization factor for the computed exit fluence to the provided exit fluence. As the normalization factor and fluence are all positive real numbers, we may safely apply the normalization factor to the initial guess to provide a starting point for the optimizer (c.f., Equation (1), Equation (4)).

$$fl_c(c \cdot K) = \sum_{x', y'} c \cdot K(x', y') \cdot [XYArr(x', y')] = c \cdot \sum_{x', y'} K(x', y') \cdot [XYArr(x', y')] = c \cdot fl_c(K) \quad (4)$$

### 2.2.3. Fluence Solver Program Design

The program is constructed in Python and is designed to take advantage of the PBS batch system to run in parallel, or to run independently on one system. The program is designed with the intent of being launched once on each node, it will then run one processing thread for each processor it has been granted on the node. If there is only one processor available, it does not spawn subthreads but instead performs all calculations from within the master thread.

It is the function of the master node to set up the problem to be solved, decide on what requires calculation, distribute the jobs, collect the results, and decide on the next step. The Open Opt package is used for the optimization in this program. The function to be minimized is the sum of the squares of the differences between the computed fluence grid at the level of the EPID and the deconvolved fluence grid obtained via measurement; in this case, the measurement is from the simulation described in 2.

The distributed nodes are tasked with calculating the components of the derivative functions at each point in the planned grid. The optimizer then determines which variables will be modified and proceeds through a step in the optimization.

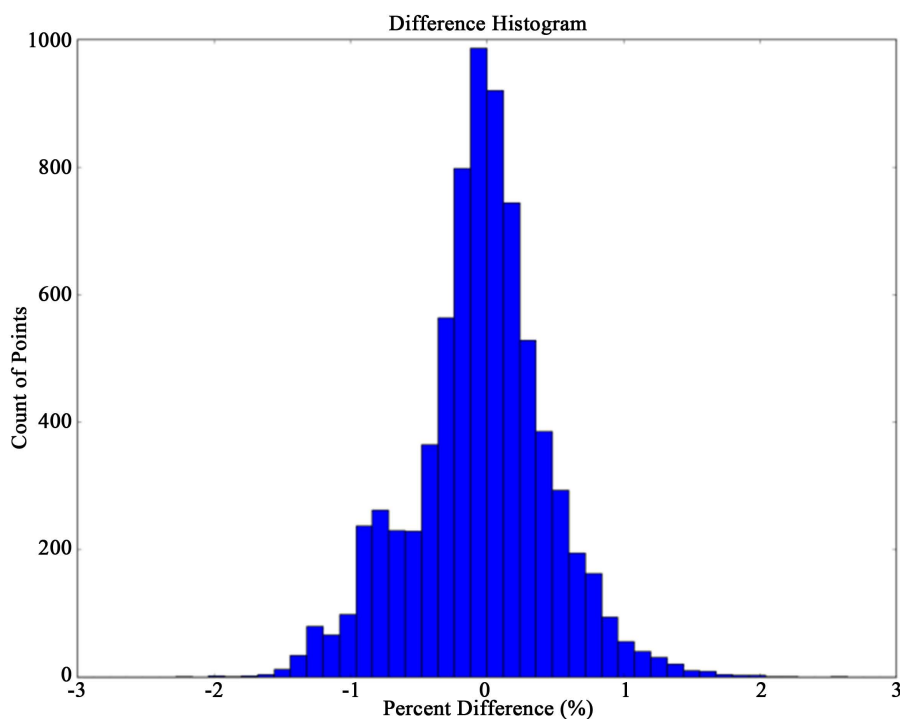
### 3. Results and Discussion

To verify the functionality of the fluence solver program described above, two sample IMRT fields were delivered onto the vEPID phantom. The two fields were designed based on contours in the Pinnacle 3 Treatment Planning System (Philips), using a machine model of a Varian iX (Varian Oncology) with a Millennium 120 MLC.

Each solver was allowed to run until the change in match quality was less than 0.1%. The average run time was 14.4 minutes, using a total of 862.3 CPU minutes to complete. Some overhead in the method of calculation results in an average of 46% of the CPU run time to be dominated by a single process on the master node, resulting in the deviation from the expected total CPU time of 1843.2 minutes from 14.4 minutes of calculation time and 128 active nodes.

The results of the fluence solver were then compared to the binned fluence obtained from the simulation to determine the capability of the system to reconstruct a known entrance fluence from a given exit fluence. The deviation histograms in percent deviation from planned fluence are shown in [Figure 1](#) and [Figure 2](#) for the two IMRT fields. The criterion for evaluation of the match is based on the recommendations of the AAPM Task Group 119, which suggests that 90% of points greater than the threshold value of 10% be within 3% dose or 3 mm distance (DTA of 90%, 3%, 3 mm) [10]. The first IMRT field was found to pass with a simple percentage difference of 94.5% of points within 1%. The second IMRT field was found to have 97.9% of points within 1%.

A review of the quality of the match may be gained visual representation of the percentage difference between the images in [Figure 3](#) and [Figure 4](#).



**Figure 1.** Histogram of percent difference for the first IMRT field.

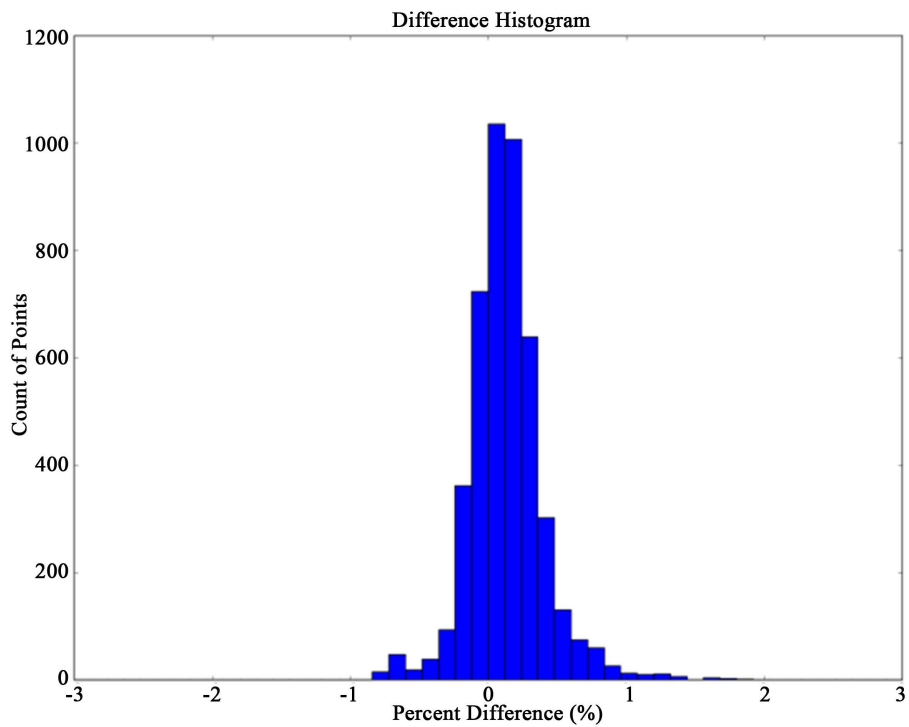


Figure 2. Histogram of percent difference for the second IMRT field.

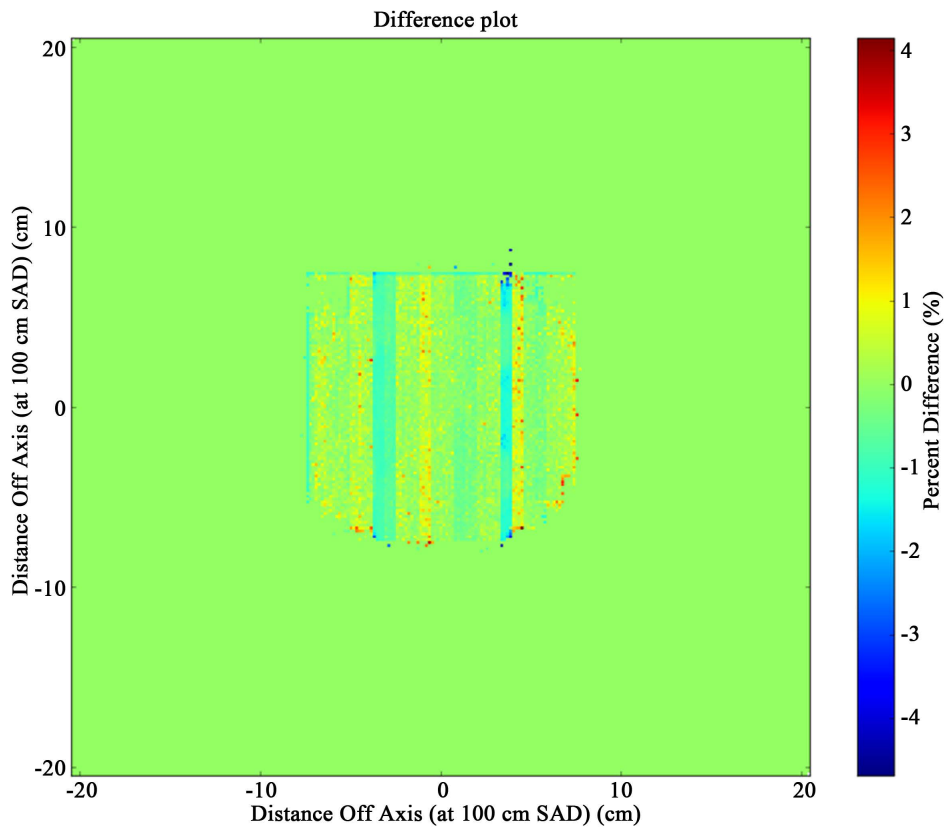
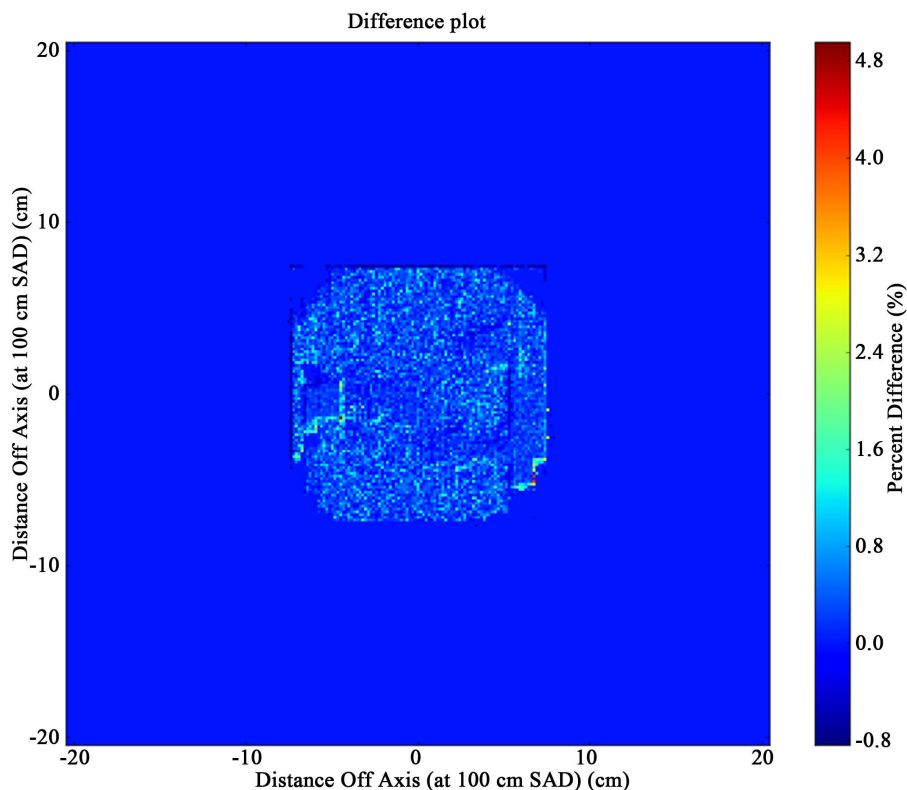


Figure 3. Plot of percentage difference between simulated and computed entrance fluence for the first IMRT field.



**Figure 4.** Plot of percentage difference between simulated and computed entrance fluence for the second IMRT field.

From these results it is clear that the software package as designed may be used to simulate a quality assurance phantom and produce a parameter space which is capable of determining a fluence exiting the machine from measured convolved EPID images. With this package the process of verifying the delivered fluence from the machine may be conducted using the EPID device to measure the exit fluence from a QA phantom. Additionally, with the use of a treatment time acquired CT of the patient, one could adapt this program to calculate entrance fluence on the patient from images collected at the time of treatment. The introduction of the patient CT into the beam path could be used to produce a set of patient and beam specific parameter spaces to be used in the fluence solver module. This would allow for post treatment verification of dose delivered to the patient for each fraction of treatment.

Further work is needed to verify the applicability of this algorithm to other linear accelerator designs, as well as automating the process from treatment time image collection. The requirements for use of this algorithm with an existing linear accelerator would include an existing commissioned Monte Carlo model for the accelerator in BeamNRC and a phantom verification to demonstrate this applicability of this algorithm on the existing model.

## 4. Conclusion

We have shown in this paper that the proposed algorithm is able to reproduce in a timely manner for one virtual linear accelerator. With this algorithm, we feel that direct post-delivery verification of delivered plans may be possible, and the resultant calculated fluence may be used to compute post-treatment dosimetry on patients with minimal intervention.

## References

- [1] Renner, W.D., Norton, K.J. and Holmes, T.W. (2005) A Method for Deconvolution of Integrated Electronic Portal Images to Obtain Fluence for Dose Reconstruction. *Journal of Applied Clinical Medical Physics*, **6**. <http://dx.doi.org/10.1120/jacmp.v6i4.2104>

- 
- [2] Louwe, R.J.W., McDermott, L.N., Sonke, J.J., Tielenburg, R., Wendling, M., van Herk, M.B. and Mijnheer, M.J. (2004) The Long-Term Stability of Amorphous Silicon Flat Panel Imaging Devices for Dosimetry Purposes. *Medical Physics*, **31**, 2989-2995.
- [3] Louwe, R.J.W., Tielenburg, R., van Ingen, K.M., Mijnheer, M.J. and van Herk, M.B. (2004) The Stability of Liquid-Filled Matrix Ionization Chamber Electronic Portal Imaging Devices for Dosimetry Purposes. *Medical Physics*, **31**, 819-827. <http://dx.doi.org/10.1118/1.1668411>
- [4] McDermott, L.N., Louwe, R.J.W., Sonke, J.J., van Herk, M.B. and Mijnheer, B.J. (2004) Dose-Response and Ghosting Effects of an Amorphous Silicon Electronic Portal Imaging Device. *Medical Physics*, **31**, 285-295. <http://dx.doi.org/10.1118/1.1637969>
- [5] Renner, W.D., Sarfaraz, M., Earl, M.A. and Yu, C.X. (2003) A Dose Delivery Verification Method for Conventional and Intensity Modulated Radiation Therapy Using Measured Field Fluence Distributions. *Medical Physics*, **30**, 2996-3005. <http://dx.doi.org/10.1118/1.1610771>
- [6] Rogers, D.W.O., *et al.* (2007) BEAMnrcUsers Manual National Research Council of Canada Report PIRS-0509A-RevL.: NRCC.
- [7] Van Rossum, G. and Drake, F.L. (2006) Python Reference Manual. <http://docs.python.org/ref/ref.html>
- [8] Kroshko, D.L., *et al.* (2010) OpenOpt. Software Package. <http://openopt.org>
- [9] Trefethen, L.N. (1996) Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations. Cornell University, Ithaca.
- [10] Ezzell, G.A., *et al.* (2009) IMRT Commissioning: Multiple Institution Planning and Dosimetry Comparisons, a Report from AAPM Task Group 119. *Medical Physics*, **36**, 5359-5373. [http://www.aapm.org/pubs/reports/RPT\\_119.pdf](http://www.aapm.org/pubs/reports/RPT_119.pdf)