The Monte Carlo Method: Versatility Unbounded In A Dynamic Computing World
Chattanooga, Tennessee, April 17–21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2003)

ISSUES RELATED TO THE USE OF MCNP CODE FOR AN EXTREMELY LARGE VOXEL MODEL VIP-MAN

Brian Wang and X. George Xu
Nuclear Engineering and Engineering Physics, Rensselaer Polytechnic Institute
Troy, NY 12180, USA
xug2@rpi.edu

J. Tim Goorley
MCNP Development Team, X-5, Los Alamos National Laboratory
Los Alamos, NM 87545, USA

Ahmet Bozkurt
Harran University, Department of Physics, 63300 Sanliurfa, Turkey

ABSTRACT

The increasing uses of voxelied human body geometry have created challenges in many aspects of Monte Carlo simulations. This paper presents an investigation on the maximum number of voxels the MCNP code can handle by using an extremely detailed VIP-Man model. The VIP-Man tomographic model is a radiation dosimetry model developed from the segmented Visible Human color cross sectional images. Several variables that MCNP uses to store the information related to the MCNP "universe" have been changed in this study to maximize the efficient usage of the system memory. Two modifications have been made in the MCNP source code for the variable, *laf*, which greatly reduce the amount of system memory required for these kinds of MCNP geometries. Another variable, mazu, which is not necessarily used when running the voxel models, is only allocated if it is needed. Without the modifications, the current release of MCNP5 crashed when lattice geometries used more than 25 million voxels. With the modifications, however, a simple voxel model was tested and results showed that the modified version of MCNP5 can handle lattice geometries of 400 million voxels in size. This modified version was used to calculate the transmission photon radiograph of a portion of the VIP-Man with 100 million voxels. From this study, we conclude that, although there have been many improvements in computer speed and tallies in the MCNP, both the current and modified versions of the MCNP remains unable to handle the whole VIP-Man model at the original voxel size of 0.33mm x 0.33 mm x 1.0 mm, with 3.7 billion voxels, which would exceed the operating system limits of 32 bit MS Windows PCs if every voxel's material was stored in memory. However, a VIP-Man with larger resolution, such as 1 mm cubic voxels, would potentially be able to run on the modified version of MCNP5.

Key Words: VIP-Man, MCNP, voxel, dosimetry

1 INTRODUCTION

The simulation of radiation transport inside the body of a worker or a patient involves defining human anatomy in a Monte Carlo code. In the past 10 years, anatomical definition has shifted from using simplified surface equations to segmented tomographic images that reflect the true anatomy of a specific human body. Medical images contain a large number of 2D pixels and 3D voxels that were initially homogenized to reduce the detail of the simulated geometry. Some of the earlier successes with Monte Carlo simulation in CT images came from work related to boron neutron capture therapy where CT slices of large pixels for a small portion of the human body were used [1]. In cases where a whole-body image set is necessary, however, the total number of voxels used to define the body can be overwhelming to the computer and the Monte Carlo codes.

In a previous study, a whole-body model called VIP-Man was developed from the segmented Visible Human color cross sectional images [2]. The image slice has a pixel resolution of 0.33 mm x 0.33 mm and a vertical slice thickness of 1 mm. The whole body consists of more than 3.7 billion voxels—the biggest that is ever attempted to improvise in a Monte Carlo code. The VIP-Man model was implemented into the Version 3C of the MCNP code several years ago, but only at a voxel size of 4mm x 4mm x 4mm due to limitations of the "*mdas*" and memory of the PC [2, 3]. The model was successfully implemented in to the EGS code at the original resolution [2].

This current study attempts to use a much finer resolution geometric model in the MCNP code. The VIP-Man model was represented in the MCNP input deck by a lengthy list of universes, which fill a single lattice cell. For example, using the 1-mm3 voxels, the VIP-Man would be represented in the MCNP by a lattice of 587 x 341 x 1877 voxels involving a total of 376 million filling universes. While this lengthy list makes the ASCII text input file to be quite large (on the order of 1.13 gigabytes), a text editor can handle a file of this size. On the other hand, the 32-bit Microsoft Windows operating systems do not allow the large memory allocations that are needed for the MCNP Version 5 to run this problem. Currently, the 32-bit MS Windows operating systems limit the amount of memory of a single executable program to roughly two gigabytes, independent of the amount of available physical RAM.

Most of the general-purpose Monte Carlo codes, including MCNP, are written with user-friendly interfaces that require minimal modification of the source code. Although this is an advantage for most applications and to users who do not want to do programming, the lack of flexibility in specifying geometry in the MCNP code can be a problem when the system needs to be optimized for special problems involving a large number of voxels. Some of the remedies had been tried in the past involving, for example, setting a higher value for the *mdas* parameter in the source code (the default value is 4 million) and turning off the "Table 128" option in the input file (*mdas* determines the amount of memory to be reserved for old versions of MCNP and Table 128 is the output table that lists the lattice coordinates of all the cells in an array) [3]. The result was a better management of system memory as well as smaller dumps in the output. However, even with these improvements, the VIP-Man images at the original 0.33 x 0.33 x l.0-mm3 voxel size could not be incorporated into the MCNP 3C at that time. The input file for such a fine-resolution model would be as large as 10 gigabytes in size, which was not possible to read without out further modification to the code.

In recent years, a computer's speed, hard-drive and system memory have been greatly improved. New versions of the MCNP code have also been refined by incorporating more capabilities in geometry definition, as well as a dynamic memory capability. The purpose of this study is to investigate how these advances can improve the simulation on the VIP-Man model and to address general issues related using the MCNP code for anatomical model of extremely large voxel numbers.

2 METHOD

Two test suites of increasingly finer resolution lattice geometries, with a corresponding higher number of voxels, were created to test the capabilities of the released and modified versions of MCNP. The following sections explain how the input decks were created, the tests that were performed, and the results of these tests.

2.1 Generation of MCNP input decks

The methods to generate MCNP input decks for the VIP-Man and other voxel models are available from previous publications [4-15]. Below we introduce the procedures for modifying the MCNP input features.

2.1.1 Reducing image resolution

The original, segmented whole-body VIP-Man dataset contains 3.4 billion (1760 by 1024 by 1877) voxels, each having a dimension of 0.33 mm x 0.33 mm x 1 mm^3. This extremely large number of voxels is much beyond the capability of current PCs and/or MCNP. We first increased the voxel size to 1 mm x 1 mm x 1 mm, resulting in 376 million (587 by 341 by 1877) voxels. However, this amount of voxels was still more than MCNP could handle. Several testing input decks were then created by including only a portion of the body. Additionally, a few input decks with larger resolution (4mm x 4mm x 4mm) were made.

2.1.2 Creating MCNP input decks

An image-based model is essentially an array of orthogonal hexahedra, 3D rectangular virtual blocks. With its built-in “repeated structures” feature, especially the type-1 lattice, MCNP is ready to accept such voxelized geometry. A unique ID number, used as a filling universe number, is assigned to every anatomical structure in the body to designate the material properties such as density and elemental composition, based on the CT Hounsfield number and visual inspection. The voxels with the same ID number essentially make up each organ, allowing convenient organ tallies. Each material, including air, is assigned to a volume of space, a cell, and a universe number. These filling universe numbers are used to specify what cells (in this case single materials) occupy which lattice locations. The entire human body is represented with a single lattice, which also contains many lattice elements of pure air surrounding the human anatomy.

The voxel models were created using a visual C++ program. The program first read in the organ ID number for each voxel from the reduced resolution images in RAW format, and then convert into plain text format that is required by MCNP input file. Each voxel occupies 1 byte in RAW format, while it takes three bytes in an MCNP input deck. As a result, the size of the MCNP input deck is roughly three times of the total number of voxels. MCNP input decks with different total number of voxels can be generated by selecting different number of slices. This program is very flexible and other functions can be easily added. For example, it calculated the total number of voxels for each organ during this study.

2.2 Modifications to MCNP code

Several variables that MCNP uses to store the information related to the “universe” have been changed in this study to maximize the usage of the system memory. Two modifications have been made for the universe number storage variable, *laf*. Another variable, mazu which is not used when running the VIP-Man input deck, was left unallocated. The last modification to the code is to remove a particular section in the chekcs.F90 file. These two modifications are further discussed below.

2.2.1 Modification to *laf*

In order to allow larger number of voxels in MCNP, several variables need to be changed. The most significant modifications were to the two dimensional array *laf*, which is allocated 1:3 by roughly the number of voxels in the lattice. In MCNP5, *laf* is used to store information for 4 different purposes, only two of which were used in the simple, yet memory-consuming, geometric model of VIP-Man. The first information stored here is the number of lattice voxels in each X, Y and Z dimension and the lowest lattice index for each. Then, the filling universes from the input deck and possible transformations of the filling universes are stored in the different portions of the array. Since there are usually no transformations in these types of voxel geometry problems, much of the memory allocated for this variable is unused.

To remove this inefficiency in the MCNP code, the variable *laf* was broken up into four separate variables, each variable for its distinct purpose. The arrays used for storage of translocated lattices are then allocated only if needed. For a 100 million voxel geometry, this prevented almost 762 Megabytes of memory from being allocated. Furthermore, the values of the filling universes in this problem are all less than 127. Since each filling universe usually represented a volume containing a single material, there are less than 127 different materials being used within the lattice. To further save memory space, the second portion of *laf*, now its own variable, was declared to be a 2 byte integer, whose values range from –126 to 127. While further modifications could be made to change the limit from 127 to 256, this was not necessary for this problem. Declaring the second *laf* variable to be a 2 bytes, rather than the default 4 bytes prevented about 190 Megabytes of memory from being used on the Windows PCs.

Another modification was made regarding the variable *laf*. During dynamic reallocation of memory during execution of MCNP in the routine nxtit1, this variable was briefly re-allocated to be much larger than it was needed for these types of problems. For large lattice geometries, this memory usage spike would cause MCNP to exceed the Windows operating system limit and thus cause the code to crash, even if the problem could have otherwise run. This memory usage spike was removed and replaced with a re-allocation much more appropriate in size.

2.2.2 Allocation of *mazu*

Finally, another variable, mazu, was also identified as using considerable amounts of memory and was a candidate for reduction. Mazu is an array which is used for the creation of print table 128 in the output, which lists the neutron activity in each repeated structure / lattice element. This array, whose size is roughly the number of lattice locations, was being allocated even when the print table was not being used. By not allocating the memory when the variable is not used, MCNP5 required 381 Megabytes less of RAM for this problem involving the VIP-Man model.

2.2.3 Remove of a section of code in the chekcs.F90

While investigating memory usage, it was discovered that a particular section of code in the chekcs.F90 was very time consuming. These few lines were running for hours or even days before being completed. This section of code was “commented out” for these modifications, drastically reducing the setup times for this problem to ~45 minutes. The purpose of this section was to perform some checks on the lattice, which the knowledgeable user can check on their own.

3 RESULTS AND DISCUSSIONS

In this section, we present a test on the relationship between the memory usage and the number of voxels in lattice geometry for two versions of the MCNP5: version 1.20 and the large lattice modification version. A geometry plot and mesh tally plot for VIP-Man are then introduced using the MCNP5 geometry plotter and a radiographic tally.

3.1 Testing the memory usage and voxel limit of MCNP

For the original VIP-Man model, the current version of MCNP5 exceeds the Windows OS limit during memory allocation for the variable *laf*, and thus terminates in the routine setdas. Using a small suite of test input decks (i.e., a portion of the VIP-Man model at a voxel size of 1mm3), we found that lattice geometries of the VIP-Man of up to 25 million voxels can be used with the current version of MCNP5. Since the operating system limit is for the total memory used by the executable program, adding tallies, variance reduction, or even certain print tables, will reduce the usable number of voxels in the lattice geometry.

The modifications were tested in two different ways. The first was to ensure that MCNP5 still passed the regression test suite in both sequential and parallel calculations on several different platforms, including Windows PCs. The second way was to create a small set of test problems and verify that the modified and original version of MCNP still tracked. These very simple input decks used the repeat text input (#r) capability of the MCNP input deck to create lattices with up to 400 million lattice locations but with very small file sizes. A sample input deck is shown in the appendix. This sample completely lists the title, cell card and surface card portion entries of an MCNP input deck with 100 million voxel entries, all of air. By changing the 99999999r to another appropriate value, the “fill=” range of values, and the surface cards 11, 21, and 31, an arbitrary large lattice models can be created.

Running this suite of calculations with the original and modified versions of MCNP5 resulted in data plotted Figure 1, which shows stable runtime memory usage as a function of the number of lattice voxels. Figure 1 shows that these modifications significantly extend the total number of voxels that MCNP can run. The MCNP5_RSICC_1.20 release, however, cannot run with lattice geometries of more than 25 million, since the “memory spike” discussed above caused the code to crash, even though it would have been capable of handling more voxels while running particles. The modified version was successfully tested with up to 400 million voxels.

Since the slope of this line is quite linear (MS Excel's R2=0.99990), it is possible to extrapolate that a run-able lattice geometry would have a maximum number of roughly 800 million voxels on the Windows operating system.

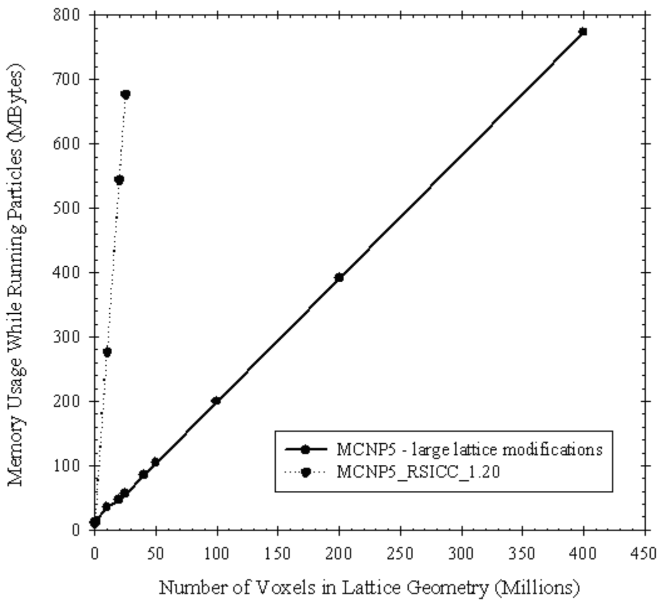


Figure 1. Memory usage as a function of the number of lattice voxels

3.2 Testing on computing time for different shapes of innermost lattice boundaries

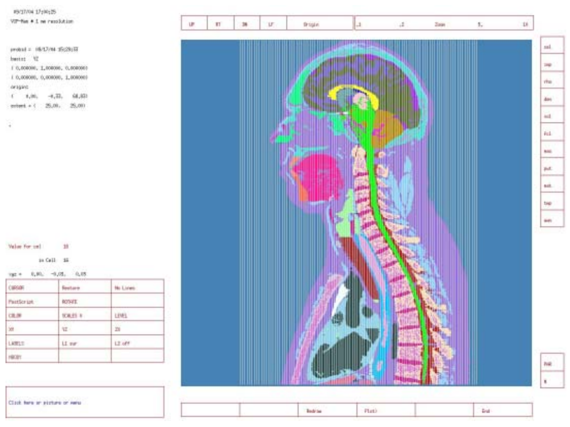
The shapes of innermost lattice boundaries in each universe are believed to have an impact on the computing efficiency. Two input decks were generated to test this assumption: one is a cube whose dimensions are equal to those of the innermost base lattice structure; the other is a simple sphere that is larger than the base lattice structure. Other parts are kept same for the two input decks. A total of 10 million particles were run for each case and the runtimes were compared.

The runtime differences are not as large as initially assumed. The total runtimes are 1085 and 1078 minutes for the cube and sphere cases, respectively. In detail, the startup time (cp0) is 48 minutes for the cubic case, which is slightly shorter than the 55 minutes of the sphere case. On the contrary, the transporting time for the cube is 1037 minutes, which is longer than the 1023 minutes for the sphere case.

As expected, the tally results for F6, F4 and F8 are exactly same for the two runs. This is because these innermost cells are being placed within the lattice, the actual shape of the cell is irrelevant. Particles will jump into the next lattice location when they reach the lattice boundary, which is well inside the cell boundary. Another issue is that each universe must be filled with cells according to MCNP manual [16]. However, this is not the case for base lattices. The cell that is the exterior of the base lattice cell (i.e., sphere) does not have to be filled if the cell is large enough to fully contain the lattice voxel.

3.3 VIP-Man plots from MCNP5

The 100 million voxel VIP man input deck was plotted in the MCNP5 geometry plotter and is shown in Figure 2. This geometry was used to create a transmission photon radiograph resulting from a parallel beam of photons created from 120 KVp source [17]. Particles were scored on a mesh tally, a user-defined geometry-independent grid. The tally volumes were also 1mm^3 voxels. In this case, the tally the photon flux was calculated several meters away from the geometry, so that scattered particles have been removed. The plot in figure 3 was generated with the MCNP mesh tally plotter, a capability which will be released in the next patch to MCNP5.

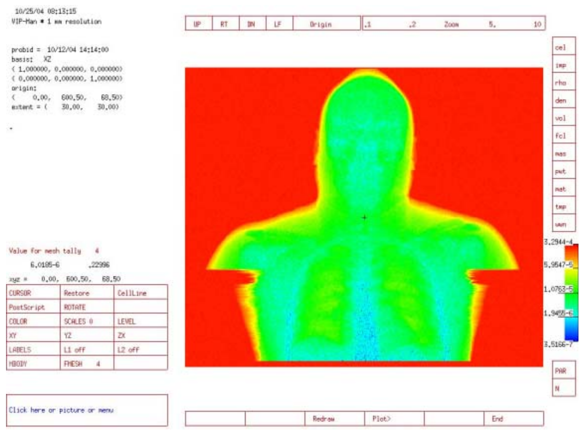


```
 09/17/04 17:04:25
VIP-Man * 1 mm resolution

probid =  09/17/04 15:29:53
basis:   YZ
( 0.000000, 1.000000, 0.000000)
( 0.000000, 0.000000, 1.000000)
origin:
(     0.00,    -4.33,    68.83)
extent = (    25.00,    25.00)

Value for cel        18
          in Cell   16
xyz =   0.00,  -9.05,   0.05
```

Figure 2. Sagittal view of 1 mm^{3}VIP-Man tors o in MCNP5 geometry plotter



```
 10/25/04 08:13:15
VIP-Man * 1 mm resolution

probid =  10/12/04 14:14:00
basis:   XZ
( 1.000000, 0.000000, 0.000000)
( 0.000000, 0.000000, 1.000000)
origin:
(     0.00,   600.50,    68.50)
extent = (    30.00,    30.00)

Value for mesh tally    4
      6.0186-6       .22996
xyz =    0.00,  600.50,   68.50
```

Figure 3. A transmission radiograph of 1mm^3 V IP-Man torso from 120 kVp x-ray source

3.4 Comparison of different tallies

For organ dose calculations, three different tally methods were compared in this study: F4, F6 and F8. This was done to ensure that different methods of calculating the same quantity (or nearly the same quantity) are the same (or nearly the same). The dose is explicitly calculated in the case involving the tally F6. In the case involving the tally F8, the deposited energy was recorded by the *F8 tally from the MCNP. Dose was then calculated by dividing the corresponding cell mass. A kerma approach was used to calculate the dose using the *F4 tally. Figure 4 presents the percentage differences between these three tallies for various organs in the VIP-Man model. In general, the F4 tallies agree within 10% with the F6 tallies, while the differences between the tallies F8 and F6 are relatively large. This is reasonable since the F6 tallies are derivatives of F4 tallies in MCNP, while F8 tallies are based on different physics from the F4 tallies. It is noted that the kerma of water was used for all the organs in the F4 method and this approximation may cause uncertainties for some organs. Readers are referred to a previous paper for the organ lists of each ID number [3].

We also tried to compare the mesh tally with lattice tally. The input decks for the lattice tallies are given below. While the input decks with mesh tally ran smoothly, the simulations with lattice tallies over the entire lattice failed to run. This failure is believed to be caused by the code limitations on the size of lattice tallies. When smaller subsets of the lattice were used, the lattice tally worked, albeit very slowly.

```
*f4:p  (200 < 200 [-73:73 -43:42 -235:234])

f6:p  (200 < 200 [-73:73 -43:42 -235:234])

*f8:p  (200 < 200 [-73:73 -43:42 -235:234])
```

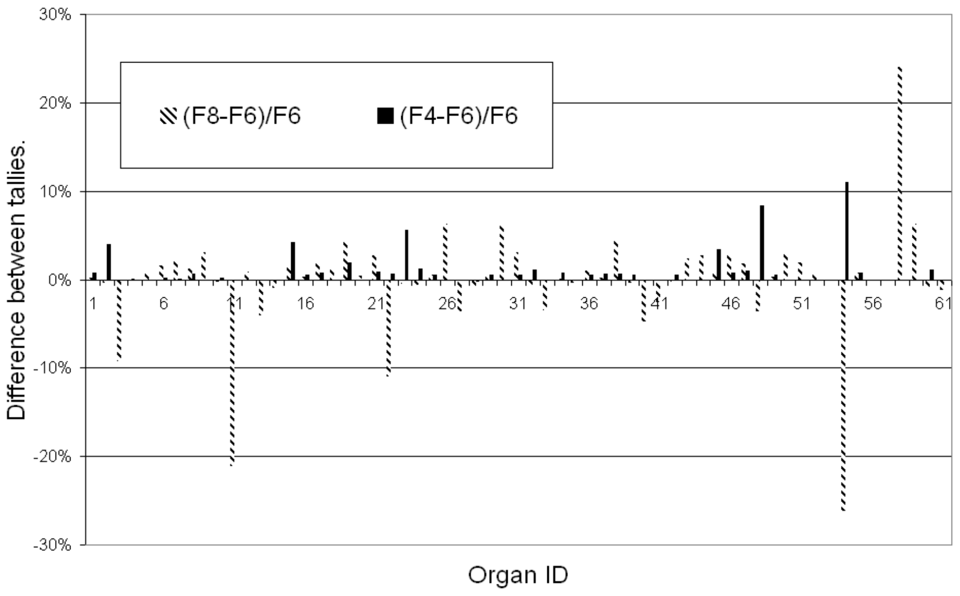


Figure 4. Comparison of different tallies in MCNP

4 CONCLUSIONS

Several issues were discussed regarding the extremely large number of the VIP-Man voxel model and its use as geometry in MCNP. The current version of the MCNP5 exceeds the Windows operating system limit during memory allocation for *laf*, and thus terminates in the routine setdas. Using a small suite of test input decks, we have shown that lattice geometries of up to 25 million voxels can be used with the current version of MCNP5. The original VIP-Man input deck and the MCNP code have been modified to maximize the efficiency of how MCNP uses the available memory. This allowed a 100 million voxel model of the VIP-Man to run a photon radiograph problem. Geometry and mesh tally plots were generated using the MCNP geometry and mesh tally plotters. We also did a comparison of different tallies in MCNP for organ dose calculations. From this study, we conclude that, although there have been many improvements in computer speed and tallies in the MCNP, the current version of the MCNP remains being unable to handle the whole VIP-Man model at the original voxel size of 0.33mm x 0.33 mm x 1.0 mm.

5 REFERENCES

1 Zamenhof, R.; Redmond, E. Ii; Solares, G.; Katz, D.; Riley, K.; Kiger, S.; Harling, O. Monte Carlo-based treatment planning for boron neutron capture therapy using custom designed models automatically generated from CT data. International Journal of Radiation Oncology Biology Physics, 35(2): 383-97, 1996.

2 Xu, X. G.; Chao, T.C.; Bozkurt A. VIP-Man: An image-based whole-body adult male model constructed from color photographs of the visible human project for multi-particle Monte Carlo calculations. Health Physics, 78(5):476-486, 2000.

3 Bozkurt A.; Chao, T.C.; Xu, X. G. Fluence-to-Dose Conversion Coefficients Based on The VIP-Man Anatomical Model and MCNPX Code for Monoenergetic Neutron Beams Above 20 MeV. Health Physics. 81(2):184-202, 2001.

4 Chao, T.C.; Xu, X. G. Specific absorbed fractions from the image-based VIP-Man body model and EGS4-VLSI Monte Carlo code: Internal electron emitters. Physics in Medicine and Biology. 46: 901-927, 2001.

5 Chao, T.C.; Bozkurt, A.; Xu, X. G. Conversion Coefficients Based on VIP-Man Anatomical Model and EGS4-VLSI code for Monoenergetic Photon Beams from 10 keV to 10 MeV. Health Physics. 81(2):163-183, 2001.

6 Chao, T.C.; Bozkurt, A.; Xu, X. G. Dose conversion coefficients for 0.1-10 MeV electrons calculated for the VIP-Man tomographic model. Health Physics. 81(2):203-214, 2001.

7 Xu, X.G. Dose conversion coefficients for 0.1-10 MeV electrons calculated for the VIP-Man tomographic model-Response to Zankl and Petoussi-Henss. Health Physics 82(2):255-256. 2002.

8 Xu, X.G.; Chao, T.C.; Bozkurt, A. A Male Radiation Worker Model Developed From Transverse Color Images Of The Visible Human Project. Proceedings of The Fourth Visible Human Conference, Keystone, Colorado October 17-19, 2002.

9 Xu, X.G. and T.C. Chao. Calculations of Specific Absorbed Fractions For GI-Tract using A Realistic Whole-Body Tomographic Model. Cancer Biotherapy and Radiopharmaceuticals Volume 18 (3):431-436. 2003.

10 Winslow, M.; Huda, W.; Xu, X. G.; Chao, T.C.; Shi, C.Y.; Ogden, K. M., Scalzetti, E. M. Use of the VIP-Man Model to Calculate Energy Impacted and Effective Dose for X-ray Examinations. Health Physics 86(2): 174-182. 2004.

11 Winslow, M.; Xu, X. G.; Huda, W. Monte Carlo Simulations of Patient X-ray Images. American Nuclear Society Transactions Vol. 90: 459-460, 2004.

12 Chao, T.C.; Bozkurt, A.; Xu, X. G.. S－Values Calculated from a Tomographic Head/brain Model For Brain Imaging. Accepted by Phys. Med. Biol.

13 Xu, X. G.; Chao, T.C.; Bozkurt A. Comparison of Effective Doses from Various Monoenergetic Particle Based On the Stylized and the VIP-Man Tomographic Models. Accepted by Rad Prot. Dosimetry.

14 Wang B.; Xu, X.G.; Goldstein, M.; Sahoo, N. Adjoint Monte Carlo method for prostate external photon beam treatment planning: An application to 3-D patient anatomy. Accepted by Phys. Med. Biol.

15 Bozkurt, A. and Xu, X. G.. Fluence-To-Dose Conversion Coefficients For Monoenergetic Proton Beams Based On The VIP-Man Anatomical Model. Accepted by Radiation Protection Dosimetry.

16 Brown et al. MCNP – A General Monte Carlo N-Particle Transport Code, Version 5 Volume II: User’s Guide. Los Alamos National Laboratory, Distributed by the RSICC of the Oak Ridge National Laboratory; 2003

17 J. T. Bushberg, J. A. Seibert, E. M. Leidholdt Jr., J. M. Boone, The essential physics of medical imaging, Lippincott Williams &Wilkins, Philadelphia, 2002

APPENDIX Sample input decks to test the modified and original version of MCNP

```
Memory Test of lattices in MCNP5 1K*1K*100 = 100,000,000 = 100M voxels.
1000 0 -11 10 -21 20 -31 30   $ Lat Cell bounding planes for 1 voxel
       lat=1 fill= 0: 999 0: 999  0: 99  $ fill=i1:i2 j1:j2 k1:k2
       56  99999999r           $ 56 Xr, change X equal to (#voxels-1)
       u=100                   $   lat cell is universe 100
 500  2  -1.29300E-03 -70 u=56 $ Cell which fills each lattice voxel
 1001 0 10 -12 20 -22 30 -32 fill=100 $ Window Cell, looking into lat
 1002 0 ( -10:  12:-20: 22:-30: 32) -1000 $ Outside window cell
 1003 0                              1000 $ Exterior of problem

   10  px    -10.5000
   11  px    -10.4790  $ generate 1K lat locations across x dimension
   12  px     10.5000
   20  py    -10.5000
   21  py    -10.4790  $ generate 1K lat locations across y dimension
   22  py     10.5000
   30  pz    -12.5000
   31  pz    -12.2500  $ generate 100 lat locations across z dimension
   32  pz     12.5000
c    Lattice entries = 1K * 1K * 100 = 100,000,000 = 100M voxels.
 1000   so  10.0E+01
   70   so  5.0E+01
```