A fast tube of response ray-tracer

Colas Schretter[a)]
PET/Cyclotron Unit, Université Libre de Bruxelles, Bruxelles, 1070 Belgium

(Received 20 March 2006; revised 25 September 2006; accepted for publication 28 September 2006; published 30 November 2006)

A ray-tracing algorithm is proposed to quickly approximate volumes of intersection between an arbitrary tube of response and a voxel array. The method is based on the idea of the Wu antialiased line tracer that is well known in the computer graphics community. However, our method works in three dimensions and supports arbitrary symmetrical response profile functions. The inner loop implementation does not use any conditional branching and is aware of low-level optimization strategies. The running speed of a fast incremental Siddon routine appears to be about 60% slower than our algorithm.
[DOI: 10.1118/1.2369467]

I. INTRODUCTION

The running speed of many statistical reconstruction methods used in emission tomography is mostly dominated by the backprojection operations that map input data to image space.[1] The most popular backprojector is the tracing of a ray through an array of squares (pixels) or cubes (voxels) by the Siddon method.[2] In the case of list-mode reconstruction,[3] this procedure is called for each coincidence event detected by the PET scanner. Hence, a clever and efficient implementation of the backprojection is the cornerstone of clinically applicable list-mode reconstruction methods.

In our context, the ideal ray-tracer collects every image elements that are crossed by lines of sight between a given pair of detectors and evaluates the areas (or volumes) of intersection between the fan of lines and the collected pixels (or voxels). In the Siddon ray-tracer, an incremental procedure marches through a regular grid of pixels or voxels and explicitly computes the intersection points between the line segment joining the centers of the detectors and the edges of these image elements. Step-by-step, the ray-tracer determines the next point of intersection with axis-aligned lines (or planes) that define the boundaries of image elements and the length of the line segment joining this point to the previous one is computed.

The work of Siddon is not new; however, the logic behind his algorithm could be improved. As shown in Fig. 1(a), at the exception of extremities, the intersection length between the line segment and columns is constant and only up to two pixels are crossed. Therefore, the length associated to the second pixel of the column is equal to the difference between the constant length mentioned before and the length of intersection with the first pixel. One of these two values could possibly be equal to zero as in the case of all pixels in the example, except for those of the seventh column.

In the case of an incremental ray-tracing procedure, the intersection lengths of a pair of pixels can be computed, column-by-column when the absolute value of the slope is below or equal to 45 degrees, or row-by-row otherwise. In the three-dimensional case, this observation remains valid and the values of four voxels that belong to a slice can be computed at once. Moreover, in the incremental implementations of Siddon,[4,5] the next plane of intersection is determined in the inner loop of the ray-tracer. Hence, by computing, step-by-step, the intersections with every voxel of a slice, it is possible to remove the conditional jumps.

The Siddon algorithm models lines of responses, but thin lines do not match so well the width of detectors. For instance, a pair of detectors could be more accurately seen as a tube of response linking the detectors. A volumetric tracing procedure, such as the algorithm introduced by Joseph,[6] will produce refined results. Note that the real tubes of responses are not necessary cylinders of constant section, however, in this work, we neglect to handle the corrections for these second order effects.

Our contributions are twofold. First, we propose to improve the efficiency of the numerical implementation of the ray-tracer by taking advantage of the mathematical simplifications pointed above. Second, we model the rays by volumetric tubes of responses instead of lines. Our procedure is based on the Wu's antialiased line drawer[7] that is well know in the computer graphics community. In the next section, we present our volumetric ray-tracer. In Sec. IV, we demonstrate both quality and performance improvement versus an incremental implementation of Siddon.

II. METHOD

For ease of illustration, we first describe the problem and its solutions in two dimensions. Extension to the three-

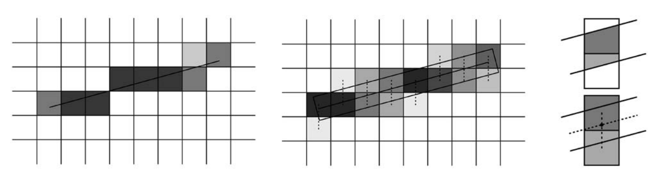

FIG. 1. Siddon (a) and Wu (b) ray-tracing results. The tones of black are proportional to the lengths of the intersection between the line and pixels for Siddon and are proportional to an approximation of the intersection areas between the strip and pixels for Wu. The dotted segments link the centers of the two pixels, closest to the central line of the strip in (b). In (c), the ideal area (up) is approximated by the length of the vertical projection on the central line of the strip (down).

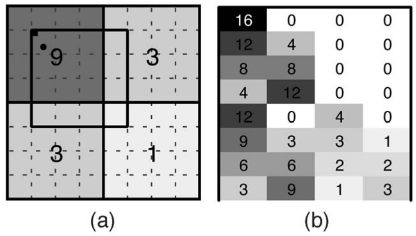

FIG. 2. Precalculated subvolumes of intersection for a quantized pair of subvoxel coordinates (a) corresponding to the sixth row of the look-up table (b). The disk represents the precise pair of subvoxel coordinates. The square is the quantized pair obtained after projection on a 4×4 lattice. The black tones are proportional to the approximated volumes of intersection. These values ranges between 0 and 16 and must be divided by 16 to be normalized.

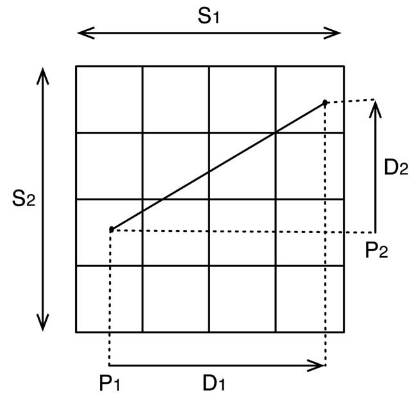

FIG. 3. Geometrical relations of parameters for the fast tube of response ray-tracer algorithm in the two-dimensional case.

dimensional case is natural. Given a two-dimensional array of pixels and a pair of points defining the extremities of a ray, the output of the ray-tracer algorithm is a list of pixels intersecting the ray and a value associated to each of these pixels. We assume that the width of the detector is one pixel wide.

A very accurate solution to trace a ray in image space is setting pixels' values that are proportional to the area covered by a thick strip of one pixel large. The Wu antialiased line method, shown in Fig. 1(b), sets the two pixels, closest from the central line of the strip, with a value that is proportional to one minus the distance between the center of each pixel and its axis-aligned projection on the central line. In comparison, the Siddon method, shown in Fig. 1(a), affects each pixel with a value that is proportional to the length of the intersection of the central line of the tube of response. Remark in the example that less pixels are affected by this method.

As shown in Fig. 1(c), the rationale of the Wu method is to approximate, for each pixel, the area covered by the strip. Area computations are replaced by evaluations of the distance between the center of the pixel and its vertical projection on the central line. This approximate area is equal to the difference between one and the fractional part of the vertical coordinate of the points on the central line. For a given point inside a pixel, the horizontal and vertical offsets relative to the upper left corner of the pixel are exactly equal to the respective fractional parts of the point coordinates. These offsets are called subpixel coordinates.

Clearly, with a tube's thickness of one pixel, up to three pixels can be crossed by the strip in any given column, as in the fifth vertical slice of Fig. 1(b). The Wu method avoids handling this case to keep the inner loop of the algorithm simple. This approximation is a trade-off between speed and quality that has been proven in practice. The values of the two pixels shall be selected so that they sum to exactly one and the center of intensity lies exactly on the central line. The algorithm could be extended to trace rays that are thicker or thinner than one pixel. However, this possibility is not discussed in this work.

In a similar way, in three dimensions, up to four voxels are crossed by a thick tube. We approximate the computation of the volumes of intersections by four products involving two of the subvoxel coordinates. This method is the straightforward extension of the Wu ray-tracer for the three-dimensional case. Furthermore, our experiments demonstrate the possibility to significantly increase the speed by precalculating volumes of intersections for a number of possible pairs of subvoxel coordinates.

For each iteration, four voxels are inserted in the output list. While the approximation of volumes of intersection is simple and appears to be efficient, the computation of the associated values involves a total of four subtractions and

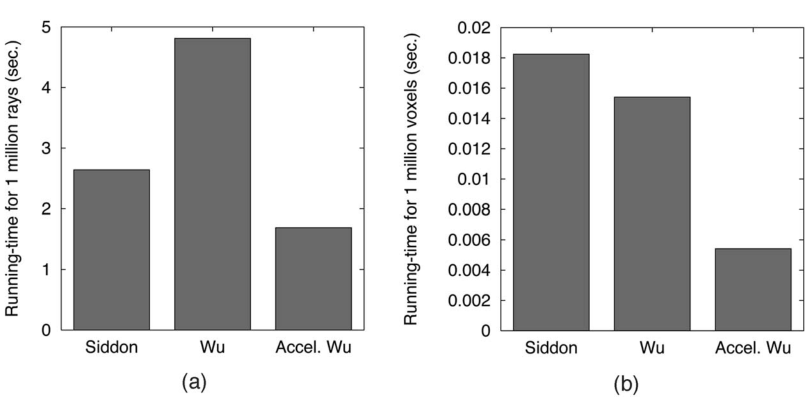

FIG. 4. Performance breakdown for per-ray (a) and per-voxel (b) running time. The accelerated ray-tracing is clearly faster than the incremental Siddon implementation, combining better physical accuracy and improved performances.

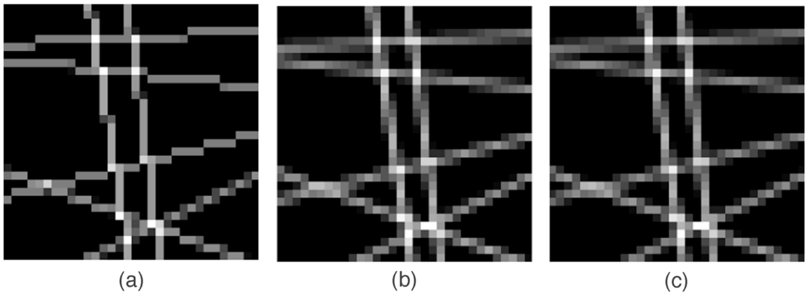

FIG. 5. Ray-tracing results for some random events with Siddon (a), the accelerated Wu (b), and the original Wu (c) methods. We show only a 32^2 portion of the accumulated depth projections of a 128^3 voxels image. The nearly diagonal lines look similar, however, nearly vertical or horizontal lines appear jaggy with the Siddon's algorithm.

four multiplications. By precalculating these four values for all possible pairs of subvoxel coordinates, we only need to fetch the results in a contiguous chunk of memory.

The fractional part of coordinates is quantized to 16 discrete values in each of the two dimensions, to limit the number of possible pairs of subvoxel coordinates to 256. A look-up table (LUT) stores for each of these pairs, the volumes of the four voxels straddling the central line. Four floats are stored for each of the 256 possible configurations. Hence, the total size of our table is 4 kB and will fit in the first level cache of most processors.

Retrieving the approximated subvolumes can be done quickly, by computing an offset in the LUT in function of the fractional parts of two coordinates. Figure 2 shows an example of configuration in a four by four quantization lattice. This technique can support arbitrary symmetrical response profiles since any volume computation strategy can be used in this precalculation stage.

III. ALGORITHMS

The tube of response tracer is described by algorithm 2 and the precalculation of the LUT is specified by algorithm 1. For convenience and speed, we used fixed-point numbers in our C++ implementation. Fixed-point math allows to extract integer and fractional parts of real coordinates, simply by masking and shifting. The image of voxels is stored in a contiguous room of memory, row-by-row and then slice-by-slice.

Algorithm 1 Look-up table pre-calculation.

$offset \leftarrow 0$
for $j \in [0,16)$
  for $i \in [0,16)$
    $a = i/16$
    $b = j/16$
    $V_{(offset+0)} \leftarrow (1-a)\times(1-b)$
    $V_{(offset+1)} \leftarrow a\times(1-b)$
    $V_{(offset+2)} \leftarrow (1-a)\times b$
    $V_{(offset+3)} \leftarrow a\times b$
    $offset \leftarrow offset+4$
  end for
end for

Algorithm 2 Fast tube of response ray-tracer.

if $|D_1|>|D_2|$ **and** $|D_1|>|D_3|$ **then**
  $l=\lfloor |D_1| \rfloor$, $a=2$, $b=3$
  $O=[0, S_1, S_1S_2, S_1(1+S_2)]$
else if $|D_2|>|D_3|$ **then**
  $l=\lfloor |D_2| \rfloor$, $a=1$, $b=3$
  $O=[0, 1, S_1S_2, 1+S_1S_2]$
else
  $l=\lfloor |D_3| \rfloor$, $a=1$, $b=2$
  $O=[0, 1, S_1, 1+S_1]$
end if
$D_i \leftarrow D_i/l \quad 1\le i\le 3$
repeat l
  $c=\sum_{i=1}^{3}\lfloor P_i\rfloor \prod_{j=1}^{i} S_{j-1}$
  $d=4(\lfloor 16(P_a-\lfloor P_a\rfloor)\rfloor+16\lfloor 16(P_b-\lfloor P_b\rfloor)\rfloor)$
  $I_{(c+O_i)} \leftarrow V_{(d+i-1)}/l \quad 1\le i\le 4$
  $P_i \leftarrow P_i+D_i \quad 1\le i\le 3$
end repeat

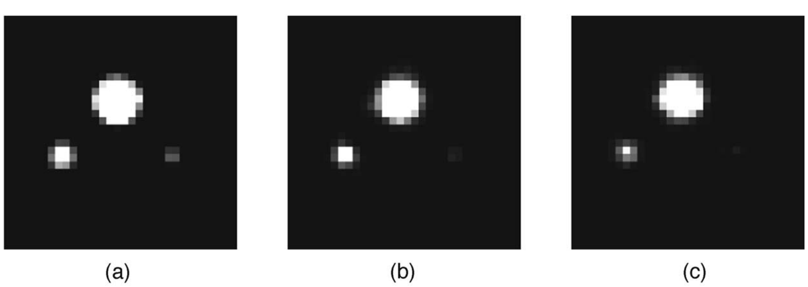

FIG. 6. Two 32^3 voxels images have been reconstructed using both ray-tracing operators. We considered a set of 10 000 events. The images show a slice through the volume estimated after the fourth iteration of the LMEM algorithm. The emission image (a) is clearly more accurately recovered using our algorithm (b). The Siddon operator (c) does not fully recover the border of emission regions, thus small balls of the reconstructed phantom vanish.

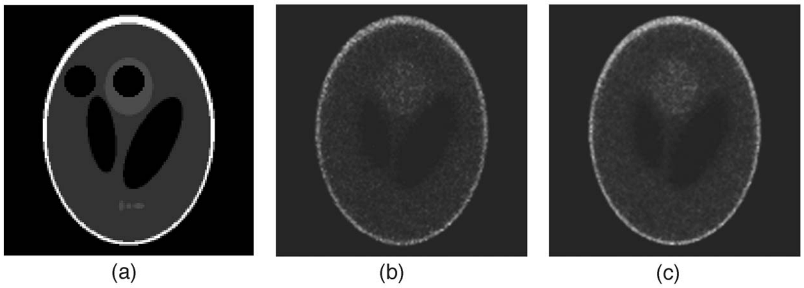

FIG. 7. Transversal slices of 128^3 images of the Shepp–Logan 3D phantom. A spherical volume of interest (VOI) and a second reference VOI are respectively marked by the black upper-central and upper-left spots in the emission image (a). The images reconstructed after ten iterations of the LMEM algorithm minimize the mean square error with respect to the emission image and are shown for the Siddon (b) and accelerated Wu (c) ray-tracers.

For each dimension $1 \leqslant i \leqslant 3$ of the image I, S_i is the image definition in terms of number of voxels along the dimension i, P_i is the real coordinate of the first extremity, and D_i is the difference between the two extremities of the tube of response. We assume that both extremities of the tube of response lie inside the image I. Hence, clipping of the rays against the image boundaries is not necessary.

For convenience, we define $S_0=1$. V is the LUT of precalculated subvolumes of intersection described above and O will be initialized with four offsets pointing to nearest neighbor's voxels in the current image slice. There are three different cases to fill the array O. Indeed the neighbor voxels lie in the axis-aligned slice straddling the line. Hence, there are three possible assignations of the offsets, depending on the major (longer) axis of the central line of the tube.

Figure 3 shows the relations between the former parameters of algorithm 2. Two memory offsets are computed in the inner loop: c is the index of the current voxel in image space and d is a line number in the LUT, multiplied by 4, since four subvolumes of intersections will be fetched at each iteration.

IV. RESULTS

We implemented from scratch both Siddon and Wu ray-tracers in the C++ programming language, sharing every possible common parts of code. We evaluated both quality and performance of the Wu method, with and without using the LUT of precalculated subvolumes. For performance comparisons, wall-clock speed comparison seems appropriate since the ray-tracers are very sensitive to low-level issues such as memory cache misses penalties and faster access to the precalculated results instead of explicitly computing these values. However, we only performed the benchmarks on one workstation, based on the IBM G4 1.33 MHz processor.

We traced a set of one million random events, in a sequential 128^3 voxels image, such that both extremities are lying on a sphere enclosed in the image volume. The tracing procedures fill an array with an offset-value pair for each crossed voxel. The Siddon method crossed 144 713 010 voxels in 2.64 sec. The Wu method affected 312 068 264 voxels in 4.81 sec and the Wu version accelerated by LUT took 1.69 sec only. The Wu method computes explicitly the values associated to each voxel by multiplying two subvoxel coordinates for each of the four voxels of the current slice, while the accelerated Wu version only fetch these precalculated values in memory.

Figure 4 charts both per-ray and per-voxel timing. A direct consequence of the tube of response model is that more voxels are crossed by a thick tube, than by its central line. Hence, the size of the output list is always greater or equal than for the Siddon case. Surprisingly, the accelerated ray-tracer still outperforms Siddon because of its advantageous per-voxel performances in terms of speed.

Figure 5 demonstrates improvement in visual quality by tracing several random events in image space. The accelerated version of the Wu ray-tracer produces nearly identical images than the slower original method. Hence, the approximation of volumes of intersection with fetches in a LUT seems to be of acceptable accuracy.

Figure 6 demonstrates a loss of resolution recovery when using the Siddon ray-tracer. We launched two iterative reconstructions using the LMEM algorithm[8] by using two different ray-tracing techniques. Three balls of various diameters and equal densities define the phantom. We processed a set of

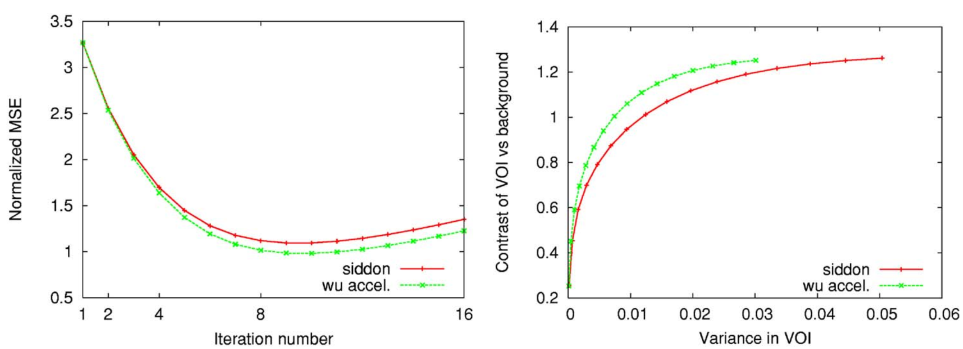

FIG. 8. Quantitative analyses of reconstruction results with the Shepp–Logan 3D head phantom for the 16 first iterations of the LMEM algorithm. The normalized sum of squares of residuals between the emission image and the images reconstructed with Siddon and accelerated Wu ray-tracers is plotted. The relation between contrast recovery coefficient (CRC) and variance inside the first VOI demonstrates that the Wu model yield lower variance for very similar CRC values.

10 000 events only and used an image definition of 32^3 voxels to emphasize the resolution recovery of small features with a low-count statistics data set.

We generated the dataset by randomly simulating the emission process. For each event, we selected one ball and randomly picked an annihilation position inside the sphere and an emission direction. From the radius and density of each ball, we defined a discrete probability density function. Hence, the probability of selecting a ball depends on its density and its volume.

Finally, Fig. 7 shows images of the Shepp-Logan three-dimensional (3D) head phantom and reconstruction results after ten iterations of the LMEM algorithm. The dataset contains the first ten million events generated by a random simulation of the emission process. Quantitative results are plotted in Fig. 8.

V. CONCLUSION

This work proposes an incremental volumetric ray-tracing algorithm that sets, for each iteration, four voxels straddling the central line of a tube of response. The voxel values are proportional to an approximation of volumes of intersection with the tube of response. The implementation uses a look-up table of 256×4 real values, containing precalculated subvolumes of intersection. Hence, handling arbitrary symmetrical response profiles is straightforward. The running speed of a fast incremental Siddon routine appears to be about 60% slower than our algorithm. Thus our method seems to combine better physical modeling and improved performances.

ACKNOWLEDGMENTS

The author gratefully acknowledge Ralph Brinks for sharing his implementation of the incremental Siddon ray-tracer and therefore giving motivations for this work. The author would like to express appreciation to Michel Defrise for insightful discussions, Raymond Devillers and Eythan Levy for their helpful reviews, and Gaetan Van Simaeys and Serge Goldman for their medical guidance.

[a]Electronic mail: cschrett@ulb.ac.be

[1]B. De Man and S. Basu, "Distance-driven projection and backprojection in three dimensions," Phys. Med. Biol. **49**, 2463–75 (2004).

[2]R. Siddon, "Fast calculation of the exact radiological path length for a three-dimensional CT array," Med. Phys. **12**, 252–257 (1985).

[3]A. J. Reader, S. Ally, F. Bakatselos, R. Manavaki, R. J. Walledge, A. P. Jeavons, P. J. Julyan, S. Zhao, D. L. Hasting, and J. Zweit, "One-pass list-mode em algorithm for high resolution 3d pet image reconstruction into large arrays," IEEE Trans. Nucl. Sci. **49**, 693–699 (2002).

[4]H. Zhao and A. J. Reader, "A fast ray-tracing technique to calculate line integral paths in voxel arrays," IEEE Nuclear Science Symposium, 2003, pp. M11–218.

[5]B. D. Sutter, M. Christiaens, K. D. Bosschere, J. V. Campenhout, and I. Lemahieu, "A fast cache-aware algorithm for the calculation of radiological paths exploiting subword parallelismJournal of Systems Architecture, Special Issue on Parallel Image Processing **10**, 781–790 (1999).

[6]P. Joseph, "An improved algorithm for reprojecting rays through pixel images," IEEE Trans. Med. Imaging **1**, 192–197 (1983).

[7]X. Wu, "An efficient antialiasing technique," ACM Computer Graphics **4**, 143–152 (1991).

[8]H. Barrett, T. White, and L. Parra, "List-mode likelihood," J. Opt. Soc. Am. A **14**, 2914–2923 (1997).