Accelerating the Anti-aliased Algebraic Reconstruction Technique (ART) by Table-based Voxel Backward Projection

Klaus Mueller*, Roni Yagel+, and J. Fredrick Cornhill*

* Biomedical Engineering Center, The Ohio State University, 270 Bevis Hall, Columbus, OH 43210
+ Department of Computer and Information Science, The Ohio State University, 395 Dreese Lab, Columbus, OH 43210

Abstract — **ART is often used in CT for 2D or 3D reconstruction when the projections are sparse, noisy, or non-uniformly distributed. Recently, ART was also shown to be useful for PET. Reconstructions obtained with ART often exhibit considerable noise artifacts attributable to aliasing introduced in both grid projection and reprojection. While the table-assisted raycasting approach currently used for projection and reprojection allows some of the necessary anti-aliasing measures to be implemented efficiently, certain limitations remain: 1) Anti-aliasing by projection supersampling is time consuming; 2) An implementation on parallel architectures is awkward. In this paper, we show how the ART projection process can be reversed into a backward-viewing block-iterative procedure. Here, the reconstruction grid voxels are projected independently to the screen where their contributions accumulate. In this way, the overhead associated with stepping individual rays across the grid is eliminated. The "footprint" of a voxel's contribution is realized as a pre-computed lookup-table, and most projection operations are reduced to simple incremental operations and table lookups. Our approach cuts the cost for ART with respect to equivalent raycasting methods by about 30% while using much less memory. Finally, the independency of calculations among voxels facilitates a parallel implementation.**

I. INTRODUCTION

In ART, the object to be reconstructed is embedded in a regular discrete 2D (3D) voxel grid. Iteratively, for each projection image pixel, the grid is projected, the projection is compared with the pixel value in the aquired projection image, and a correction term is computed and reprojected onto the grid. Each reprojection updates the grid to correspond more closely to the aquired projection data. Iterations continue until some margin of error is reached. In our work, instead of the traditional row-action formulation of ART we use a block-iterative method: Grid correction is performed only after an entire projection image is computed. This approach, termed SART, was shown to reduce the striping artifacts in the reconstruction images obtained by row-action ART methods [1].

It was found that the disturbing noise artifacts often exhibited in images produced by ART methods is caused by aliasing introduced in the reconstruction procedure [2]. When aliasing is not sufficiently reduced, and instead the noise is supressed in a heuristic manner by weakening the effect of the correction terms, then more iterations may be needed for convergence, and important object detail may be supressed as well. Proper anti-aliasing measures have been identified [2]: 1) In both projection phases the grid must be sampled at a rate well above the Nyquist rate; 2) More sophisticated functions must be used for interpolation of the samples from the grid values which requires kernels of larger extents; 3) The projections must be supersampled well above the Nyquist rate to compensate for the inevitable imperfection of the interpolation functions.

Currently, all ART implementations use raycasting for both projection phases. In SART, a ray is traced from each projection pixel to the location of the source, sampling the grid at equidistant locations, and accumulating the sampled object attenuations. Since the computational cost of raycasting is linearly related to grid sampling rate, kernel extent, and the number of rays shot per pixel, the identified anti-aliasing measures are expensive to implement. The computational cost per ray associated with large interpolation kernels and high grid sampling rate can be greatly reduced if the ray indexes a pre-integrated interpolation kernel lookup-table (PKLUT) instead of sampling the function at equidistant locations [3]. However, the computational expense is still linearly related to the number of rays traced which hampers pixel supersampling. In the following we describe a method for SART that uses the concept of pre-integrated interpolation kernels but which cost is relatively insensitive to the number of rays traced.

II. METHODS

Our approach reverses the way we perceive the projection process: Instead of looking at rays and how they traverse the grid voxels v_j, we look at the v_j and how they "influence" the projection pixels p_i. Each v_j can be represented as a field of influence shaped by the interpolation function $h(u,v)$ and scaled by v_j's value. The extent of this field is determined by the region in which $h(u,v)$ is above some threshold. A ray sample value s_k is then calculated by the weighted sum of the values of all v_j in which field of influence s_k falls:

$$p_i = \sum_j v_j \sum_k h(X(s_k) - X(v_j), Y(s_k) - Y(v_j)) = \sum_j v_j \cdot w_{ij} \quad (1)$$

The inner sum represents the influence, or weight w_{ij}, of v_j with respect to the ray sum of p_i. A bundle of parallel rays originating at a row of neighboring pixels on the projection plane will integrate $h(u,v)$ along parallel lines. Therefore the total contribution of v_j to the projection image P_φ is similar to a "footprint" and is defined by the projection of $h(u,v)$ at angle φ onto the projection plane. Fig. 1 illustrates (for a nearest-neighbor interpolation kernel) that all grid voxels leave identical, though spatially offset, footprints on the projection plane.This

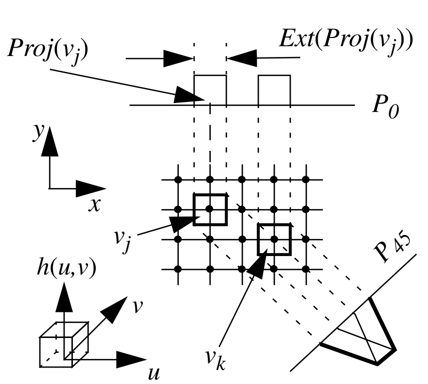

Fig. 1. Voxels v_j and v_k leave identical, though spatially offset, footprints on projection planes P_0 and P_{45}. In P_{45}, the composite footprint is depicted as the sum of the individual footprints of v_j and v_k.

enables us to transform ART into a backward-viewing algorithm in which voxels are projected individually onto the projection plane where their effects accumulate to form the projection image.

Similarly to the PKLUT's used in [3], voxel footprints can be implemented as high-resolution 1D *voxel projection tables* (2D for cubic voxels), one for each projection angle φ, where each table entry represents the result of a linear integration of $h(u,v)$ orthogonal to the projection plane of P_φ.

In the projection phase, for a particular projection angle φ, we compute for each grid voxel v_j the projection $Proj(v_j)$ of v_j's center, and subsequently the extent $Ext(Proj(v_j))$ of the voxel projection table on the projection plane. We then index the table for all pixels that fall within $Ext(Proj(v_j))$ to obtain v_j's contribution to these pixels. Note that most of the projection calculations can be performed as incremental operations.

SART's grid update equation [1] can then be implemented as follows: We associate two buffers with the projection plane: buffer $\mathbf{B_r}$ accumulating ray sum $\sum w_{ij} v_j$ and buffer $\mathbf{B_w}$ accumulating the weights $\sum w_{ij}$ as voxels are projected. Buffer $\mathbf{B_0}$ is the aquired projection image. After all voxels have been processed, the correction image buffer $\mathbf{B_c}$ is computed as:

$$\boldsymbol{B_c} = \lambda \frac{(\boldsymbol{B_0} - \boldsymbol{B_r})}{\boldsymbol{B_w}} \tag{2}$$

where λ, $0<\lambda\leq 1.0$ is a relaxation factor [1]. In the reprojection phase, the correction buffer $\mathbf{B_c}$ is distributed back onto the reconstruction grid. As in the projection phase, for each voxel v_j we compute $Proj(v_j)$ and $Ext(Proj(v_j))$. The correction term for v_j is then computed as the weighted sum of all correction terms affecting v_j:

$$c_j = \frac{\sum_{i \in I} w_{ij} \cdot \boldsymbol{B_c}[i]}{\sum_{i \in I} w_{ij}} \tag{3}$$

where I is the set of elements of buffer $\mathbf{B_c}$ for which $Ext(Proj(v_j))$ is the corresponding set of projection pixels. This value is then added to v_j.

Our discussion so far was limited to the parallel beam case, however, our approach is easily extended to fan- and cone beam data with one modification: One has to scale each voxel footprint according to the perspective distortion.

III. RESULTS

The computational cost C_{ray} for a raycasting approach using PKLUT's is given by:

$$C_{ray} = c_{start} \cdot f_s \cdot \sqrt{N} + c_{step} \cdot f_s \cdot N + c_{accu} \cdot Ext \cdot f_s \cdot N \tag{4}$$

where c_{start} is the cost per projection pixel for setting up a new ray, c_{step} is the cost per ray sample for advancing the ray and determining the voxels affected, and c_{accu} is the cost for updating $\mathbf{B_w}$ and $\mathbf{B_r}$. N is the voxel grid size, Ext is the kernel extent, and f_s is the degree of projection pixel supersampling.

The computational cost for a backward-viewing method using voxel projection tables C_{back} is:

$$C_{back} = c_{rot} \cdot N + c_{accu} \cdot Ext \cdot f_s \cdot N \tag{5}$$

where c_{rot} is the cost for rotating a voxel and calculating the limits of its projected extent. Analyis of the computational complexity reveals that $c_{rot} \approx c_{step}$ and $c_{accu} \approx 0.5 \cdot c_{rot}$.

It was shown [2] that if f_s=5 and a Gaussian$_{1/2}$ interpolation kernel (Ext = 3) is used, noise artifacts due to aliasing can be reduced to a neglible level (see Fig. 2). For f_s=5 the ratio C_{back}/C_{ray} is then approximately equal to 0.68.

In original SART, two arrays, both of size N, are needed to hold the temporary weight and correction term accumulations in the re-projection phase. In our approach, the size of these arrays reduces to the buffer size $\sqrt{N} \cdot f_s$. Hence, only $f_s / \sqrt{N}$ of the memory used in the original SART realization is required.

IV. CONCLUSION

We have described a table-based backward-viewing procedure which accelerates both projection phases of anti-aliased ART. The speedup is especially notable when pixel supersampling is performed. For block-iterative methods, memory consumption is significantly reduced. Our approach is easily extended to fan- and cone beam data, and a parallel implementation is facilitated by the inherent independence of voxels.

REFERENCES

[1] A.H. Andersen, A.C. Kak, "Simultaneous ART (SART), a superior implementation of ART," *Ultrason. Img.*, vol. 6, pp. 81-94, 1984.

[2] K. Mueller, R. Yagel, J.F. Cornhill, "Efficient and Accurate Implementation of ART," *Tech. Rep. OSU-CISRC-4/95-TR12*, Department of Computer.and Information Science, Ohio State University, 1995.

[3] K.M. Hanson, G.W. Wecksung, "Local basis-function approach to CT," *Applied Optics*, vol. 24, no. 23, pp. 4028-4039, 1985.

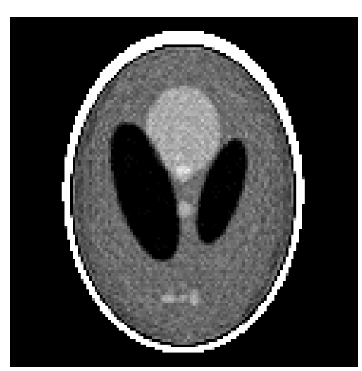

Fig. 2. Reconstruction of the Shepp-Logan phantom with anti-aliased ART (3 iterations, f_s=5, 80 projections of 128 pixels each, 128x128 voxels, Gaussian$_{1/2}$ interpolation kernel.) The computation time was 38.6 sec on a 200 MHz SGI Indigo2 workstation.