IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 22, 2018, accepted April 19, 2018, date of publication April 25, 2018, date of current version May 16, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2829861

Fast Projection Algorithm for LIM-Based Simultaneous Algebraic Reconstruction Technique and Its Parallel Implementation on GPU

SHUNLI ZHANG[1], GUOHUA GENG[1], GUOHUA CAO[2], YUHE ZHANG[1], BAODONG LIU[3,4], AND XU DONG[2]

[1]School of Information Science and Technology, Northwest University, Xi'an 710127, China
[2]Department of Biomedical Engineering and Mechanics, Virginia Tech, Blacksburg, VA 24061, USA
[3]Beijing Engineering Research Center of Radiographic Techniques and Equipment, Institute of High Energy Physics, Chinese Academy of Sciences, Beijing 100049, China
[4]School of Nuclear Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China

Corresponding author: Yuhe Zhang (zhangyuhe0601@nwu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772421, Grant 61572400, Grant 61731015, and Grant 61602380, and in part by the National Key Research and Development Program of China under Grant 2017YFF0107201. The work of G. Cao and X. Dong was supported from the U.S. National Science Foundation under Grant CBET 1351936.

ABSTRACT Simultaneous algebraic reconstruction technique (SART) is a well-known iterative method in X-ray computed tomography, which provides better image quality than analytical methods when dealing with incomplete or noisy data. The disadvantage of SART is the slow speed compared with the analytical methods. Since forward projection and backprojection are two major time-consuming operations in iterative reconstruction, we propose an algorithm for fast forward projection and improved backprojection for the line integral model-based SART. Using the proposed algorithm, the SART method was implemented on a GPU platform with NVIDIA's parallel computing architecture. Both computer simulations and physical phantom experiments were carried out, and their results show that our approach is highly efficient and accurate. The computation time for the system matrix using our proposed projector is 10 times faster than that using the Siddon's projector, and our improved backprojection algorithm is 1.5 times faster than Li's method in determining the minimum bounding interval. The GPU-based SART using our proposed projection algorithm can obtain about 7.4 times reconstruction speed-up compared with that using the traditional projection approach, while preserving the accuracy of the results.

INDEX TERMS Computed tomography, image reconstruction, simultaneous algebraic reconstruction technique (SART), forward projection, backprojection, GPU.

I. INTRODUCTION

X-ray computed tomography (CT) is an important technique for non-invasive imaging, which has been widely used in the field of diagnostic medicine and nondestructive testing. In practical CT system, image can be reconstructed either by using analytical or iterative methods. Currently, the algebraic reconstruction technique (ART) [1] and its major refinement, simultaneous algebraic reconstruction technique (SART), are well-known iterative reconstruction methods [2]. Compared to analytical methods, the SART algorithm usually performs better especially when the projections are sparse, incomplete, or non-uniformly distributed in angles [3]. However, the disadvantages of SART are its high computational complexity and costs.

Over the decades, many efforts have been made to improve the reconstruction speed of SART. Jiang and Wang [4] have established the convergence of SART and proved that the sequence generated by the SART converges to a weighted least square solution from any initial guess. Hudson and Larkin [5] used ordered subset (OS) technique to accelerate the expectation maximization (EM) algorithm; they showed the OS-EM algorithm can provide an order of magnitude acceleration over traditional EM algorithm. Wang and Jiang [6] studied the convergence of the OS-SART, they

found that the smaller the size of subsets, the faster the convergence. Recently, Ji *et al*. [7] proposed a new simultaneous algebraic reconstruction technique based on guided image filtering (SART-G), which is designed mainly for such situation where the prior image is known for CT image reconstruction. Their study showed that the convergence of the SART-G is faster than the convergence of SART. These methods improve the reconstruction speed of SART by accelerating the convergence. However, they cannot improve the speed of individual iteration.

The order in which the projections are accessed, as well as the selection of the relaxation parameter, have a significant impact on both the reconstruction quality and the speed of convergence [8]–[12]. Wan *et al*. [13] proposed an adaptive simultaneous algebraic reconstruction technique (ASART) for incomplete data and noisy conditions. They developed a modified multilevel access scheme to minimize the correlation between consecutive views in the limited angle range. In addition, they applied an adaptive adjustment of relaxation parameters to correct the discrepancy between actual and computed projections for each iteration during the reconstruction process. They showed that the reconstruction by ASART outperforms classical SART in speed.

In order to accelerate the reconstruction speed of SART, many parallel techniques have been studied. Mueller and Yagel [14] utilized OpenGL texture mapping graphics hardware to accelerate 3D SART. However, the reconstruction quality is rather sensitive to the resolution of the frame buffer. Besides, the programming is nontrivial to implement. Liu and Zeng [15] extended SART algorithm to linear scan cone-beam CT, and implemented SART on an IBM Blade Center HS20 cluster using message passing interface (MPI). The parallel techniques have successfully reduced the computation time of SART. However, the final image synchronization and the overhead caused by load imbalance will consume more time as the number of processors increases. Over the past few years, commodity graphics processing unit (GPU), which can be programmed for massive parallel computing, has become available and widely used for overcoming the computation challenges in image reconstructions [16]–[18]. Pang *et al*. [19] utilized CUDA-enabled GPU to accelerate SART, and obtained faster reconstruction without compromising image quality.

During the implementation of SART, most of the computation time is spent on executing the forward projection and backprojection operations. The forward projection is a discretized evaluation of the radon transform, which models the imaging geometry and physics [20]. The simplest and most widely used forward projection model is the line integral model (LIM) [21]. In this model, a single ray is described as a zero-width line that connects the X-ray source and the center of the detector cell, and the projection data is a linear integral along the path from the detector cell center to the source focal center. Existing forward projectors for SART mostly use the interpolation scheme, where sample points are equidistant. To improve the reconstruction quality, interpolation scheme has to increase the sample points by reducing the sampling distance. However, it will increase the computational complexity and significantly affect the efficiency of the forward projection process. Moreover, the interpolation scheme will lead to the unmatched projector/backprojector pair [22]. The Siddon algorithm is a widely used forward projector, where ray-pixel intersection length is used as the weighting factor for the line integral calculation [23]. Xu [24] studied the fast implementation of ray-driven projector on GPUs, and pointed out that Siddon's projector performs better than many other popular forward projectors, particularly on low-frequency data. However, calculating indices and merging arrays take about 41% and 26% of the implementation time, respectively, which affects the efficiency of Siddon's projector. Although Zhao and Reader [25] extended Siddon's projector and proposed an improved ray-tracing method, the forward projection process is still time-consuming. The backprojection is a transpose operation of the forward projection. Li *et al*. [26] proposed an easy and efficient method, which used the minimum bounding rectangle (MBR) to determine all the intersection points between a set of rays and voxels. However, Li's method involves frequent uses of sorting operations in determining the MBR, which needs to be further optimized.

Previously, we have proposed a fast forward projection algorithm for area integral model based iterative reconstruction algorithm [27]. Inspired by the work, in this paper we propose a fast forward projection algorithm for fan-beam LIM-based iterative reconstruction algorithm. We also improved Li's backprojection approach by eliminating its sorting operations. Then, we implemented the fast forward projection and improved backprojection method on a GPU platform to improve the reconstruction speed of SART.

The rest of this paper is organized as follows. In Section II, the SART algorithm is briefly introduced. Then, an algorithm for fast forward projection and improved backprojection is proposed, followed by its parallel implementation on GPU. In Section III, the reconstruction speed and quality are presented and discussed. In the last section, we discuss relevant issues and conclude the paper.

II. METHODS

A. THE SART ALGORITHM

Let $\boldsymbol{f} = (f_1, \ldots, f_N) \in R^N$ be a discrete representation of an unknown image (N is the total number of image pixels) to be reconstructed, $\boldsymbol{p} = (p_1, \ldots, p_M) \in R^M$ be the measured projection data (M is the total number of rays), and $\boldsymbol{A} = (w_{ij})$ be a known *system matrix*, whose element w_{ij} is the weighting factor that represents the contribution of the jth pixel to the ith ray integral. For LIM, w_{ij} denotes the intersection length of the ith ray with the jth pixel. The image reconstruction problem can be formulated as a system of linear equations:

$$\boldsymbol{Af} = \boldsymbol{p} \tag{1}$$

In practice, M and N are usually very large. Hence, it is difficult to use traditional matrix method to solve equation (1). The SART algorithm, as invented by Andersen and Kak [2],

solves (1) by an iterative procedure, whose iterative formula can be stated as follows:

$$f_j^{(k+1)} = f_j^{(k)} + \lambda \frac{\sum_{p_i \in p_\varphi} \left(\frac{p_i - \sum_{n=1}^{N} w_{in} f_n^{(k)}}{\sum_{n=1}^{N} w_{in}} \right) w_{ij}}{\sum_{p_i \in p_\varphi} w_{ij}} \tag{2}$$

where $k = 0, 1, \ldots, \varphi$ is a projection angle, λ is the *relaxation factor*, usually chosen within (0, 2) [28].

The reconstruction procedure of SART is as follows. An initial guess $\boldsymbol{f} = \boldsymbol{f}^{(0)}$ is made, usually with all the pixel values assigned to zero. For a given projection angle φ, compute line integrals $\sum_{n=1}^{N} w_{in} f_n^{(k)}$ for each ray of φ. Then, compute image correction, i.e. for measured p_i, subtract $\sum_{n=1}^{N} w_{in} f_n^{(k)}$ from p_i, and normalize it by $\sum_{n=1}^{N} w_{in}$. After computing all the corrections of φ, update each pixel $f_j^{(k)}$ by all the corrections (weighted by w_{ij}) of the ray passing through the jth pixel, normalized by the sum of weights $\sum_{p_i \in p_\varphi} w_{ij}$ and scaled by λ. Then select another projection angle and update all the pixels, and so on. Such iteration is continued till the convergence condition is satisfied.

Obviously, the weighting factors w_{ij} bear an important role in the iteration of SART. However, due to the huge amount of measurements and pixels, it is impossible to precompute and store all the weighting factors. In practice, they are computed on the fly. Hence, a fast projection algorithm is necessary for improving the reconstruction speed of SART.

B. FORWARD PROJECTION ALGORITHM

For the 2D case, we discretize the image space by superimposing an $I \times I$ square grid on the image $\boldsymbol{f}$, and assume that f_n is a constant, where $0 \leq n < N$, $N = I \times I$ is the total number of pixels. To determine whether a pixel n is on the boundary of the image space, we use its row index u and column index v simultaneously. Obviously,

$$f_n = f_{u,v}, \quad n = u \times I + v \tag{3}$$

A ray PQ is defined as a line segment, joining the X-ray source $P(P_x, P_y)$ and the center $Q(Q_x, Q_y)$ of the detector cell. Without loss of generality, we assume that $|P_x - Q_x| \geq |P_y - Q_y|$, whose equation in slope-intercept form is $y = mx + b$, considering $0 \leq m \leq 1$. Ray PQ intersects pixel n at point G on its left side, as shown in Fig. 1. Let d_1 be the longitudinal distance from ray PQ to the vertex B of the pixel n, obviously $0 \leq d_1 < \delta$, where δ is the length of the pixel side. Then the longitudinal distance d_2 from ray PQ to the vertex C of the pixel n is given by

$$d_2 = d_1 + \Delta \tag{4}$$

where $\Delta = m \times \delta$. Since $d_1 < \delta$, from (4) we have

$$d_2 < \delta + \Delta = (1 + m) \times \delta \leq 2\delta \tag{5}$$

From Inequality (5) we conclude that $d_2 < 2\delta$, which means that ray PQ will traverse at most two pixels in a column of the image. There are three cases, as ray GH, $GD$and GE shows in Fig. 1.

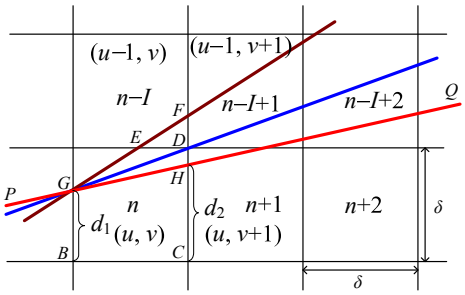


FIGURE 1. Three cases of a ray passes through the pixel n as $0 < m \leq 1$: the ray passes through the right side of the pixel n, as shown by the ray GH; the ray passes through the upper right vertex D of the pixel n, as shown by the ray GD; the ray passes through the right side of the pixel n-I on top of the pixel n, as shown by the ray GF.

We point out that both the pixel index and the intersection length can be determined according to d_2. If $d_2 < \delta$, as the ray GH shows, the ray will pass through the pixel n and intersect with its right side, whose intersection length $|GH|$ is given by

$$|GH| = \delta\sqrt{1 + m^2} \tag{6}$$

Then let $d_1 = d_2$, $v = v + 1$, the next pixel to be traversed will be the pixel $n+1$. If $d_2 = \delta$, as the ray GD shows, the ray will traverse the pixel n and pass through its upper right vertex D. For this case, the intersection length $|GD|$ can also be calculated by (6). Then let $d_1 = 0$, $u = u - 1$, $v = v + 1$, and the next pixel to be traversed will be the pixel $n - I + 1$. If $d_2 > \delta$, as the ray GF shows, the ray will traverse the pixel n and $n-I$ in turn, and their intersection lengths $|EF|$ and $|GE|$ can be respectively calculated by

$$|EF| = (d_2 - \delta)\sqrt{1 + m^2}/m \tag{7}$$

$$|GE| = |GF| - |EF| = \delta\sqrt{1 + m^2} - |EF| \tag{8}$$

Then let $d_1 = d_2 - \delta$, $u = u - 1$, $v = v + 1$, and the next pixel to be traversed will be the pixel $n - I + 1$. Note that, for a given ray, both $\delta\sqrt{1 + m^2}$ and $\sqrt{1 + m^2}/m$ are constants. Thus, the intersection lengths, such as $|GH|$ and $|GD|$, can be obtained by an assignment operation, and $|EF|$ can be obtained by a multiplication operation.

After obtaining the new pixel index and its longitudinal distance d_1, we continue the above procedure and calculate the following intersection lengths and pixel indices. Before the implementation of our algorithm, we need to determine the initial pixel index n_1, u_1, v_1 and calculate its longitudinal distance d_1. If $0 \leq R - Rm + b < 2R$, where R is the half length of the image space, then the ray PQ passes through the image space and intersects with its left boundary, as shown by the lineL_1 in Fig. 2.

In this case, n_1, u_1, v_1 and d_1 can be calculated by

$$\begin{aligned} u_1 &= I - 1 - floor((R - Rm + b)/\delta) \\ v_1 &= 0, \quad n_1 = u_1 I + v_1 \\ d_1 &= R - Rm + b - (I - 1 - u_1)\delta \end{aligned} \tag{9}$$

Therefore, beginning with the pixel n_1, we can calculate intersection lengths and corresponding pixel indices according to

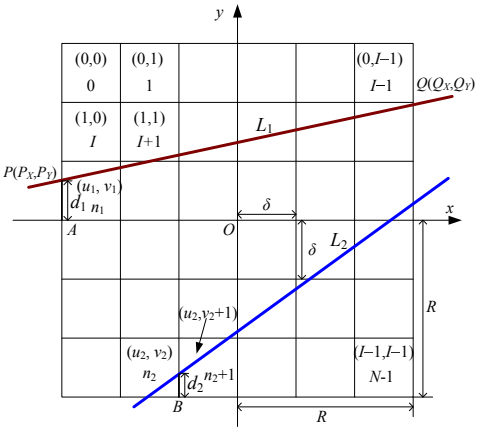


FIGURE 2. The two cases of rays passing through the left and bottom boundaries of an image in 2D forward projection, as $|P_x - Q_x| \geq |P_y - Q_y|$ and $0 \leq m \leq 1$.

the above method, until the boundary pixel is reached. In this process, we create two arrays, *index*[..] and *length*[..], to store the pixel indices and the intersection lengths, respectively. The 2D fast forward projection algorithm is summarized as follows.

We use the above algorithm to calculate the projection at each angle, and store them to the array *proj*[..]. At the same time, we can obtain the corresponding ray intersection length $\sum_{n=1}^{N} w_{in}$ and calculate the image correction. The calculated image correction can be stored in the array *proj*[..] as well.

If $-2R < R - Rm + b < 0$, the ray PQ passes through the bottom boundary of the image space, as shown by the line L_2 in Fig. 2. In this case, the initial pixel index n_2, u_2, v_2, the intersection length l between the ray and the pixel, and the longitudinal distance d_2 can be calculated by

$$\begin{aligned} v_2 &= floor((Rm - R - b)/\Delta) \\ u_2 &= I - 1, \quad n_2 = u_2 I + v_2 \\ l &= ((v_2 + 1)\delta - (Rm - R - b)/m)\sqrt{1 + m^2} \\ d_2 &= (v_2 + 1)\Delta - (Rm - R - b) \end{aligned} \tag{10}$$

We first store n_2 and l into their corresponding arrays, and then proceed from the pixel $n_2 + 1$ in the same way as described by the algorithm above.

The projection of $-1 \leq m < 0$ can be referred to the case of $0 \leq m \leq 1$. As for $|P_x - Q_x| < |P_y - Q_y|$, we can inverse x and y and represent the ray by the equation $x = m'y + b'$, with $0 \leq m' < 1$ or $-1 < m' \leq 0$. Similarly, these cases can be referred to the cases of $0 \leq m \leq 1$ or $-1 \leq m < 0$.

C. BACKPROJECTION ALGORITHM

The backprojection operation in SART is to update each pixel by an accumulated correction term in a *pixel-driven* (*voxel-driven* in 3D case) manner. Hence, the key operation of backprojection is to find those rays passing through a specified pixel. Intuitively, we can compute the intersection of each ray with a given pixel. However, due to fact that only very few rays would intersect with the pixel in a certain projection angle, this method is not desirable. Li *et al*. proposed a method to compute all the intersection points between a set of rays and a voxel. Adapting the Li's method, we can use the minimum bounding interval (MBI) to determine the rays passing through a pixel in the 2D case. However, this process needs to project all vertices of each pixel onto the detector line, and sort each vertex's projection to identify the MBI. It involves redundant computations. In contrast, in our improved method the sorting operation can be avoided according to the relative position between the X-ray source and the pixel. The Li's method and our improved method are illustrated in Fig. 3.

As shown in Fig. 3, let $E(E_x, E_y)$ and $F(F_x, F_y)$ be the center of the two boundary cells of a line detector. The parameter t is used to indicate the distance between a point on the detector and the start point E. For this purpose, the linear equations of the line detector can be represented as

$$\begin{aligned} x(t) &= E_x + t(F_x - E_x) \\ y(t) &= E_y + t(F_y - E_y) 0 \leq t \leq \end{aligned} \tag{11}$$

Let $P(P_x, P_y)$ be the position of the X-ray source. For a given pixel j, we connect P and its upper left vertex $A(x, y)$. If line PA intersects with the detector, the parameter t_a of the

Algorithm 1 2D Fast Forward Projection Algorithm

Input: image array f[..], initial pixel index n_1, u_1, v_1, and its longitudinal distance d_1.
Output: projection *proj*, two arrays of *index*[..] and *length*[..], as well as the total number of elements s.

1. $\Delta \leftarrow m \times \delta$, $s \leftarrow 0$, $n \leftarrow n_1$, $i \leftarrow u_1$, $j \leftarrow v_1$, $d \leftarrow d_1$, *proj*$\leftarrow$0,
   const1$\leftarrow \sqrt{1 + m^2} \times \delta$, const2$\leftarrow \sqrt{1 + m^2}/m$
2. **while** $i \geq 0$ and $j < I$ **do**
3. $d \leftarrow d + \Delta$
4. **if** $d > \delta$ **then**
5. $d \leftarrow d$–δ, *index*[s]$\leftarrow n$, *length*[s] $\leftarrow (\Delta$–$d)\times$const2
6. $s \leftarrow s$+1, $i \leftarrow i$–1, $n \leftarrow n$–I
7. **if** $i \geq$0 **then**
8. *index*[s]$\leftarrow n$, *length*[s]$\leftarrow$const1–$(\Delta$–$d)\times$const2
9. $s \leftarrow s$+1, $n \leftarrow n$+1, $j \leftarrow j$+1
10. **end if**
11. **else if** $d < \delta$ **then**
12. *index*[s]$\leftarrow n$, *length*[s]$\leftarrow$const1
13. $s \leftarrow s$+1, $j \leftarrow j$+1, $n \leftarrow n$+1
14. **else**
15. $d \leftarrow$0, *index*[s]$\leftarrow n$, *length*[s]$\leftarrow$const1
16. $j \leftarrow j$+1, $i \leftarrow i$–1, $s \leftarrow s$+1, $n \leftarrow n$–I+1
17. **end if**
18. **end while**
19. **for** $i \leftarrow$0 **to** s–1
20. *proj*$\leftarrow$*proj*+f[*index*[i]]$\times$*length*[i]
21. **end for**

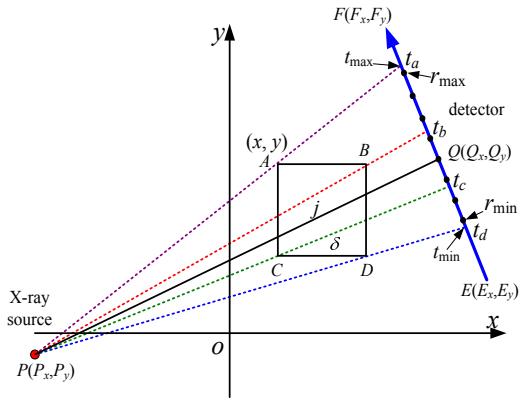


FIGURE 3. I Illustration of backprojection.

intersection point at the detector can be computed by

$$t_a = \frac{(x - P_x)(P_y - E_y) + (y - P_y)(E_x - P_x)}{(x - P_x)(F_y - E_y) + (y - P_y)(F_x - E_x)} \tag{12}$$

Similarly, for other vertices of the pixel j, we use (12) to calculate their parameter t_b, t_c, and t_d, respectively. Then, we can obtain the MBI, i.e. $[t_{\min}, t_{\max}]$, where $t_{\min} = \min\{t_a, t_b, t_c, t_d\}$, $t_{\max} = max\{t_a, t_b, t_c, t_d\}$. Note that, if both $t_{\min}$ and $t_{\max}$ are greater than 1, or less than 0, it means that no ray will intersect with pixel j. For simplicity, we assume that $t_{\min}$ and $t_{\max}$ are in the range [0, 1]. Then, the index range of those rays that can intersect with the pixel j can be determined by

$$\begin{aligned} r_{\min} &= ceil(t_{\min}(nRay - 1)) \\ r_{\max} &= floor(t_{\max}(nRay - 1)) \end{aligned} \tag{13}$$

where $nRay$ is the total number of the detector cells; $r_{\min}$ and $r_{\max}$ are the minimum and the maximum ray index, respectively. Then, for any ray r lying between $r_{\min}$ and $r_{\max}$, we can obtain its intersection point $Q(Q_x, Q_y)$ on the detector by (11), in which $t = r/(nRay - 1)$. On this basis, the intersection length of the ray PQ with the pixel j can be calculated using the ray-pixel intersection algorithm.

The above method is very effective. However, considering the position of the X-ray source to the pixel, we can further improve Li's backprojection algorithm. As can be seen from Fig. 3, there are eight cases of the position between X-ray source P and the pixel j: $P_x \le x$ and $P_y \ge y$, $P_x \le x$ and $P_y \le y - \delta$, $P_x \le x$ and $y - \delta < P_y < y$, $P_x \ge x + \delta$ and $P_y \ge y$, $P_x \ge x + \delta$ and $P_y \le y - \delta$, $P_x \ge x + \delta$ and $y - \delta < P_y < y$, $x < P_x < x + \delta$ and $P_y > y$, $x < P_x < x + \delta$ and $P_y < y - \delta$.

If $P_x \le x$ and $P_y \ge y$, the MBI is only determined by the line PC and the line PB, where $t_{\min}$ is calculated by PC, and $t_{\max}$ is calculated by PB. If $P_x \le x$ and $P_y \le y - \delta$, the MBI is only determined by the line PD and the line PA, where $t_{\min}$ is calculated by PD, and $t_{\max}$ is calculated by PA. If $P_x \le x$ and $y - \delta < P_y < y$, the MBI is only determined by the line PC and the line PA, where $t_{\min}$ is calculated by PC, and $t_{\max}$ is calculated by PA. If $P_y > y$ and $x < P_x < x + \delta$, the MBI is only determined by the line PA and the line PB, where $t_{\min}$ is calculated by PA, and $t_{\max}$ is calculated by PB. The other four cases can be handled in the same way according to the symmetry. The improved *pixel-driven* based backprojection algorithm for a given projection angle is as follows.

Algorithm 2 Improved Li's Backprojection Algorithm

Input: image array $f[..]$, projection angle φ, image correction $proj[..]$.
Output: updated image array $f[..]$.
1. compute the coordination of X-ray source $P(P_x, P_y)$, center of
   two boundary cell $E(E_x, E_y)$ and $F(F_x, F_y)$.
2.**for** $j \leftarrow 0$ **to** $N - 1$
3. compute the MBI $[t_{\min}, t_{\max}]$ for pixel j, as well as $r_{\min}$ and $r_{\max}$.
4. $accum\_weight \leftarrow 0$, $accumu\_error \leftarrow 0$
5. **for** $i \leftarrow r_{\min}$ **to** $r_{\max}$
6. compute the intersection length w_{ij} of pixel j with ith ray.
7. $accum\_error \leftarrow accum\_error + w_{ij} \times proj[i]$
8. $accum\_weight \leftarrow accum\_weight + w_{ij}$
9. $f[j] \leftarrow f[j] + \lambda \times accum\_error / accum\_weight$
10. **end for**
11. **end for**

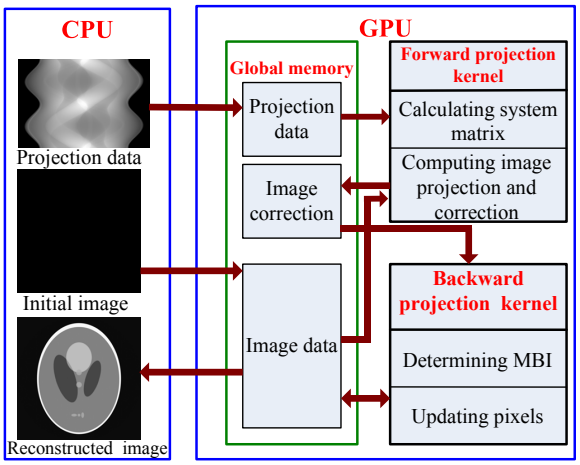
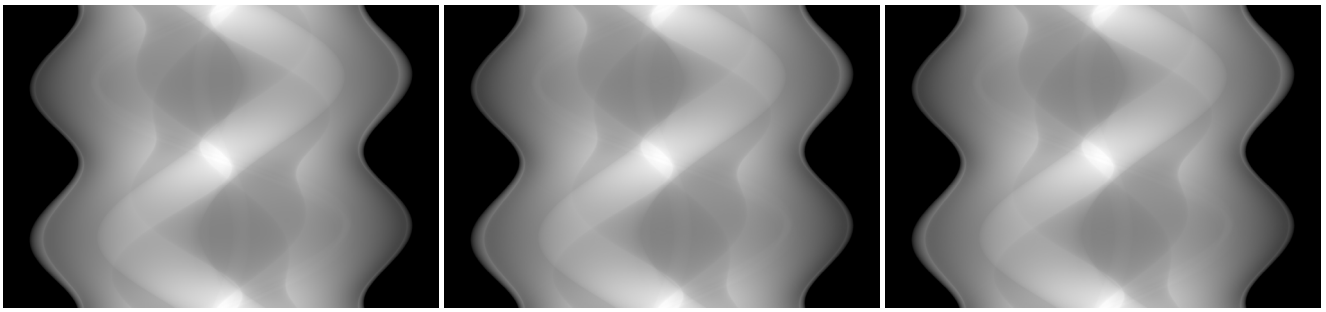


FIGURE 4. Flowchart of the GPU-accelerated SART reconstruction.

D. GPU-ACCELERATED SART RECONSTRUCTION

The iteration procedure of SART can be divided into two main steps: forward projection (including computing the corrections) and backprojection. The forward projection is a *ray-driven* approach, in which computing the projection and correction of each ray are independent. Similarly, the backprojection is a *pixel-driven* approach, in which updating the value of each pixel is independent. Both steps are very suitable for parallel implementation on current graphics hardware architecture. Therefore, we devised two CUDA kernels to implement the forward projection and backprojection steps, respectively. Each kernel corresponds to a ray or a pixel, and all kernels run in parallel threads within GPU. The schematics of the GPU-accelerated SART reconstruction with our proposed method is shown in Fig. 4.



(a) (b) (c)

FIGURE 5. Projection images generated by different methods. (a) analytical method, (b) our proposed forward projector, and (c) Siddon's projector.

As shown in Fig. 4, we first allocate three arrays in the global memory of GPU to store the projection data, the image correction, and the image data, respectively. Then, we load the projection data and the initialized image from CPU to the GPU arrays. Next, we use the forward projection kernel to compute the image projection and the correction at a view angle. Later, we use the backprojection kernel to determine the MBI and update the pixels. The procedure is repeated until all the view angles are processed. After a certain number of iterations, we transfer the reconstructed image from the GPU to the CPU.

III. EXPERIMENTS AND RESULTS

We tested the proposed method by performing experiments on both numerical phantom and physical phantom. All the implementation runs on a PC platform with Intel Core^{TM}4.2 GHz, 8 GB RAM, NVIDIA GTX 1080 Ti graphic card (11 GB global memory), and the development environment is Microsoft Visual Studio 2013 with CUDA 8.0.

A. NUMERICAL SIMULATION STUDIES

We simulated a fan-beam X-ray CT system with a detector size of 1024 cells spaced by 0.384 mm. The source to detector distance is 1150 mm and the source to rotation center distance is 650 mm. The 2D Shepp-Logan head phantom was used. To evaluate the efficiency and accuracy of our proposed forward projector, we discretized the Shepp-Logan phantom to 512×512 and 1024×1024 matrix size, respectively. We used the two images to generate the simulated projection data by our proposed projector and the Siddon's projector, respectively. The simulated projection data were generated without noise. All data contain 720 views uniformly distributed over 360 degrees. Both the proposed projector and the Siddon's projector were implemented in an ANSI C routine without any optimization. Table 1 lists the computation time.

For LIM-based model, the major process of forward projection is to compute the system matrix, i.e. determining the pixel indices and intersection lengths. As shown in table 1, the computation time of the system matrix using our proposed projector is about 10 times faster than that using Siddon's projector, and the total forward projection time of our projector

TABLE 1. Comparisons of computation time for the proposed forward projector and the Siddon's projector.

	512×512		1024×1024	
	Proposed	Siddon	Proposed	Siddon
system matrix	1.799 s	18.355 s	3.541 s	36.618 s
projection	0.999 s	1.004 s	2.122 s	2.195 s
Total time	2.798 s	19.359 s	5.663 s	38.813 s

is 6.8 times faster than that of Siddon's projector. The slow performance of Siddon's projector results from three main factors. Firstly, the pixels of the CT array are considered as the intersection areas of two orthogonal sets of the equally spaced parallel lines, with one set for the intersections with the horizontal lines, and another set for intersections with the vertical lines. The intersections of ray with pixels are calculated by a subset of the intersections of the ray with these parallel lines. Although determining the intersections of the ray with the equally spaced parallel lines can be done in a recursive way, it needs to calculate the intersections of the ray with both horizontal and vertical lines. Secondly, merging the two intersection sets is a time-consuming process, involving a large amount of comparison operations. Lastly, calculating the intersection length of ray with pixels involve a multiplication operation, and moreover, determining the corresponding pixel indices involves multiple operations of multiplication, division, and integer rounding. Compared to the Siddon's projector, our proposed projector determines the first pixel index as well as the longitudinal or lateral distance, and the other pixel indices and intersection lengths can be determined by recursion along either x-axis or y-axis. Most importantly, the pixel indices are calculated by addition or subtraction operations.

To evaluate the accuracy of our projector, we used the analytical method to generate projection data, i.e. calculating line integrals of the projection ray with the Shepp-Logan phantom, and compared the analytically generated projection data with those generated by our forward projector and Siddon's projector, respectively. Fig. 5 shows the projection images generated with different methods, where the digital phantom size is 1024×1024.

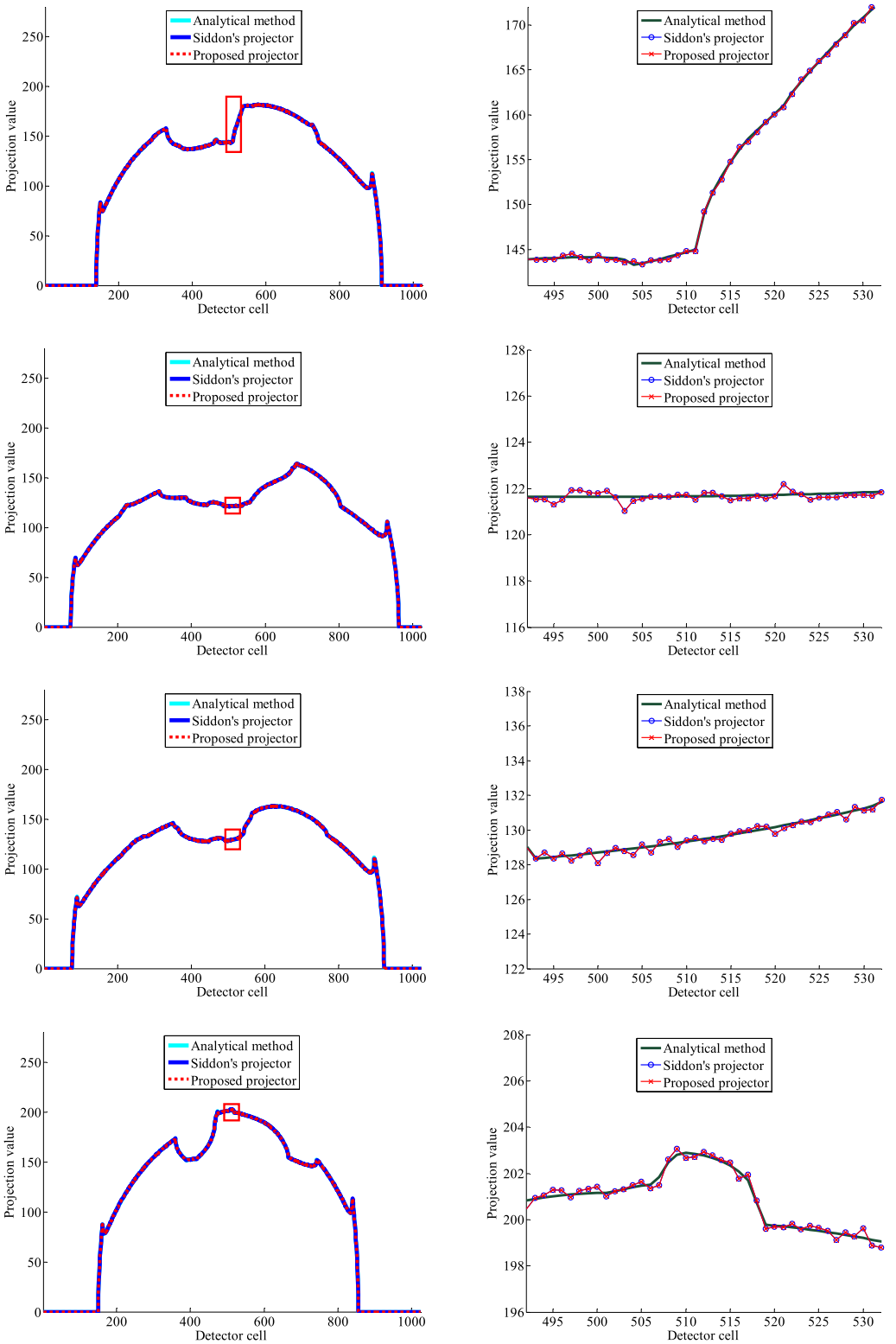


FIGURE 6. Profiles of the projection data at a specific angle. From top to bottom, the projection angle is at 40°, 80°, 120°, and 160°, respectively. The left column show the projections generated by the analytical method, Siddon's projector, and our proposed projector at each view angle. The right column show the zoom-in profiles corresponding to their counterparts in the left column (inside the red rectangles).

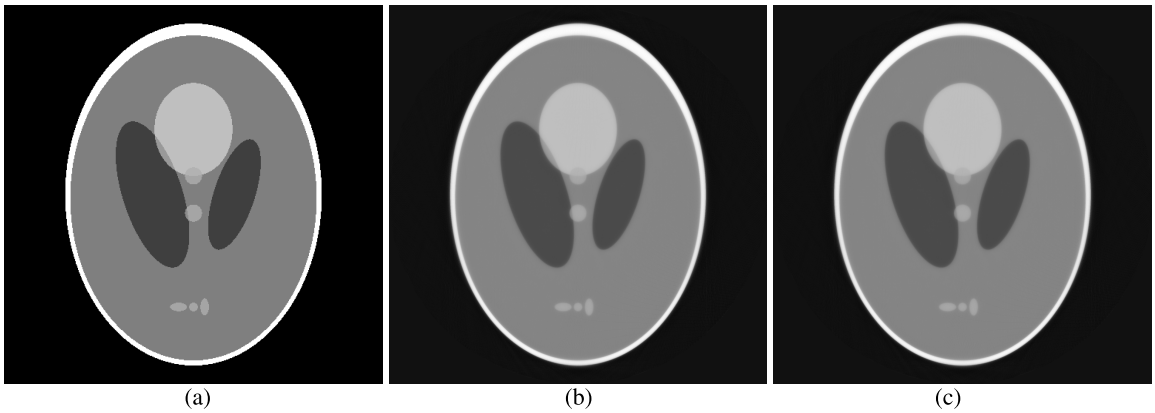


FIGURE 7. The original phantom image and the reconstructed images. (a) the original phantom image, (b) the reconstructed image with SART using our proposed method after one iteration, (c) the reconstructed image with SART using the traditional method after one iteration.

In Fig. 5, it is difficult to see the difference among the three methods visually. As introduced in Section II *B*, the implementation of our forward projector can be divided into four cases according to the slope of a projection ray: $0 \leq m \leq 1$, $m > 1$, $m < -1$, and $-1 \leq m < 0$. Therefore, to evaluate the performance of our forward projector at different angles, we compared the generated projection data at angle 40°, 80°, 120°, and 160°, respectively. Fig. 6 shows the profiles of the projection data at different angles.

As can be seen from the left column of Fig. 6, our forward projector is effective for different angles. From the magnified profiles in Fig. 6, we can see the discrepancy between the non-analytical methods and the analytical method. However, the line profiles of the projection generated by our proposed projector agree well with those generated by the Siddon's projector, which shows that our proposed forward projector has the same accuracy as the Siddon's projector.

To evaluate the performance of our improved backprojection method, we used the SART algorithm to reconstruct a 512×512 image, with a pixel size of 0.418 mm $\times$ 0.418 mm. Two projection/backprojection pairs were used in the reconstruction. One pair is our proposed forward projector with the improved Li's backprojection pair, and the other is the Siddon's projector with the original Li's backprojection pair, referred as the traditional method. During the reconstruction, the relaxation factor λ was set to 0.2, the initial value of image was set to zero, and the random projection access scheme was utilized to improve the reconstruction quality. As described in subsection II *C*, the backprojection process of SART includes two steps: determining the MBI and updating all the pixels with corrections. Correspondingly, we tested the computation time of each step. Table 2 shows the computation time of projection and backprojection of SART with one iteration.

Note that, the computation time of forward projection in table 2 includes the computation of image correction. As can be seen from table 2, our improved method is about 1.5 times faster than the original Li's method in determining MBI. The reason is that, for a given pixel, our improved method just projects two vertices of a pixel onto the detector line, and it does not need sorting operation in the calculation of $t_{\min}$ and $t_{\max}$. With our forward and improved backprojection method, the reconstruction speed of SART is improved by about 40% compared to the traditional method. Fig. 7 shows the reconstructed images of two methods after one iteration. Fig. 8 shows the profiles along the central horizontal lines in the reconstructed images of Fig. 7.

In Fig. 7 and Fig. 8, there is no apparent difference between the reconstructed results of the two methods. Fig. 8 indicates that, after only one iteration, the reconstruction quality of SART is relatively good in the smooth part of the phantom, while the deviation is apparent in the high-frequency part. However, this deviation can be reduced by increasing the

TABLE 2. Computation time of the projection and back projection of SART on CPU.

Method	Forward projection	Backprojection		Total
		Determining MBI	Updating pixels	
Proposed	3.754 s	3.827 s	19.688 s	27.269 s
Traditional	20.315 s	5.861 s	19.672 s	45.848 s

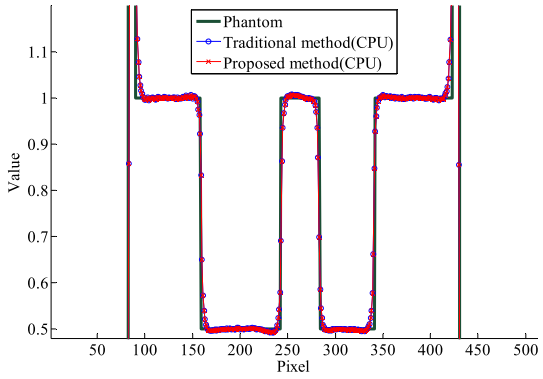


FIGURE 8. Profile comparison between the original image and the reconstructed images in Fig. 7.

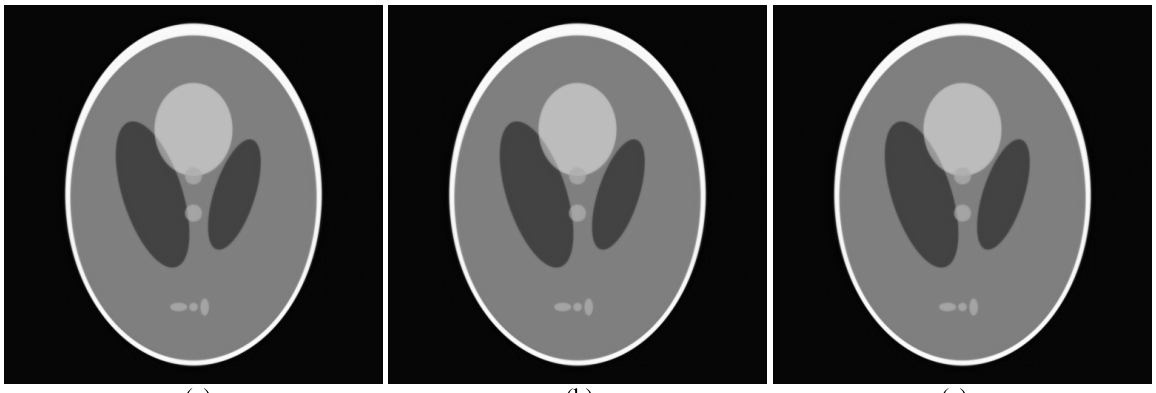


FIGURE 9. Reconstructed images of the Shepp-Logan head phantom with different methods: (a) traditional method on CPU, (b) our proposed method on GPU, and (c) traditional method on GPU.

TABLE 3. Comparison of the reconstruction quality of SART with different image sizes.

Method	512×512		1024×1024	
	NRMS	NMA	NRMS	NMA
Proposed	0.132947	0.039314	0.129585	0.043125
Traditional	0.132947	0.039314	0.129585	0.043125

number of iterations. To quantitatively evaluate the accuracy of our method, two different error measures were used in this paper. The first is the normalized root mean square (NRMS) error, which is defined as:

$$\mathrm{NRMS} = \sqrt{\frac{\sum_{u=1}^{I}\sum_{v=1}^{I}(t_{u,v}-r_{u,v})^2}{\sum_{u=1}^{I}\sum_{v=1}^{I}(t_{u,v}-\bar{t})^2}} \tag{14}$$

where $t_{u,v}$ and $r_{u,v}$ are the pixel values of the original and the reconstructed images, respectively, and $\bar{t}$ is the mean pixel value of the original image. The second is the normalized mean absolute (NMA) error, which is defined as:

$$\mathrm{NMA} = \frac{\sum_{u=1}^{I}\sum_{v=1}^{I}\left|t_{u,v}-r_{u,v}\right|}{\sum_{u=1}^{I}\sum_{v=1}^{I}\left|t_{u,v}\right|} \tag{15}$$

Both the NRMS and NMA of the reconstructed images with different size are listed in Table 3.

As can be seen from Table 3, the reconstruction quality of our method is identical with that of the traditional method for the numerical phantom experiment.

To evaluate the efficiency and accuracy of our method with GPU, we performed the SART reconstruction using our method and the traditional method on CPU and GPU, respectively. The CPU code was written in ANSI C routine, and the GPU components were programmed with CUDA 8.0 runtime API. The reconstruction image was 512 × 512, with a pixel size of 0.418 mm × 0.418 mm. In the experiment, we found that the configuration of grid and block has a great effect on the computing performance of the GPU. For the forward projection process, we used 1024 blocks. Each block has only one thread, corresponding to a projection ray. For the backprojection process, we used a 2D grid (64×64) and block (8 × 8) respectively, each thread corresponding to a pixel. Table 4 lists the computation time of SART reconstruction with the two methods on CPU and GPU, respectively.

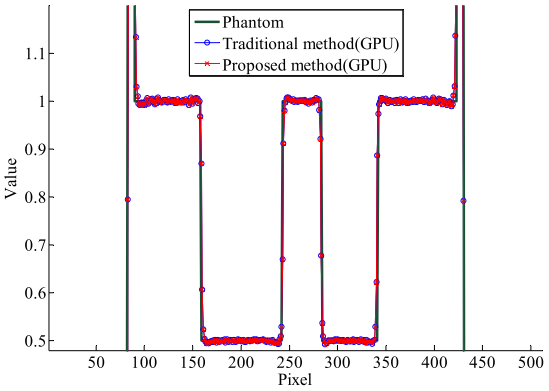


FIGURE 10. Profiles along the central horizontal lines of the reconstructed images in Fig. 9(b) and (c).

TABLE 4. Computation time of SART reconstruction with the proposed method and the traditional method.

Method	First iteration		Second iteration	
	CPU	GPU	CPU	GPU
Proposed	27.269 s	0.624 s	54.506 s	1.232 s
Traditional	45.848 s	4.618 s	91.556 s	9.188 s

As shown in table 4, the CPU-based SART using our method is about 1.6 times faster than that using the traditional method. However, for the GPU implementation of one iteration, the computation time of our proposed method (0.624 s) is 7.4 times faster than that of the traditional method (4.618 s). As Table 2 shows, updating pixels spends at least 77% of the computing time on backprojection, regardless of our proposed method or the traditional method. Hence, the gain in the efficiency of GPU-based SART using our proposed method is mainly attributed to our new forward projector. The results

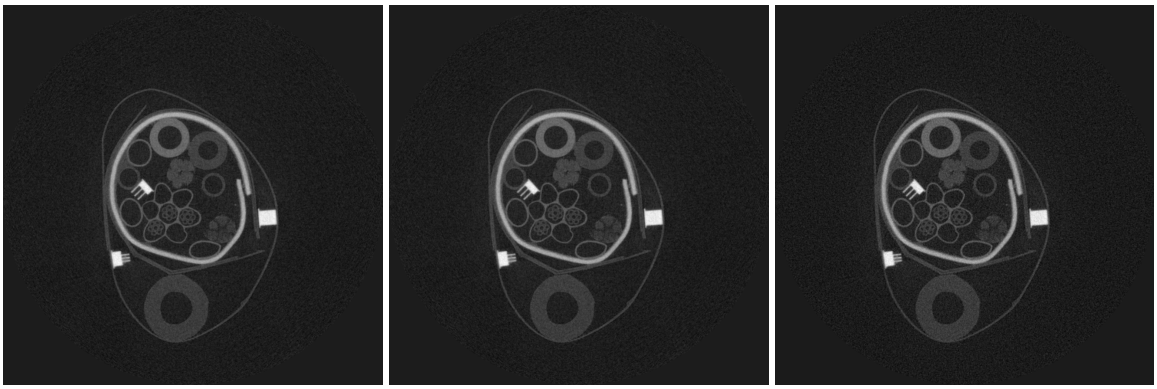


FIGURE 11. Reconstructed images of the physical phantom using (a) GPU implementation of traditional method after three iterations, (b) GPU implementation of our proposed method after three iterations, and (c) FBP algorithm.

TABLE 5. Comparison of the reconstruction quality of SART with different methods.

Method	First iteration		Second iteration	
	NRMS	NMA	NRMS	NMA
Traditional (CPU)	0.132947	0.039314	0.101481	0.024673
Proposed (GPU)	0.132947	0.039314	0.101481	0.024673
Traditional (GPU)	0.132947	0.039314	0.101481	0.024673

further indicate that our forward projector is more suitable for implementation on GPU. Compared to CPU-based SART using the traditional method, we achieved 73 times speed-up with our method on GPU. The reconstructed images after two iterations are shown in Fig. 9 and Fig. 10.

As shown in Fig. 9, the result of GPU implementation of our method is perfect, and we cannot observe any difference among the reconstructed images. In Fig. 10, the line profiles of the image reconstructed with GPU are very close to that of the original phantom. Fig. 10 shows that we can obtain excellent reconstruction result after only two iterations for the numerical phantom. To quantitatively evaluate the performance of the GPU implementation of our proposed method, the NRMS and NMA of the reconstructed images are listed in Table 5.

Note that, both the CPU and GPU implementations in this experiment are based on 32-bit floating point data. It can be seen from Table 5 that, the GPU implementation of the proposed method keeps the same precision as the CPU implementation of the traditional method.

B. PHYSICAL PHANTOM STUDIES

To evaluate the performance of our proposed reconstruction method on realistic CT data, a physical phantom experiment was carried out on a micro-CT system developed in Dr. Cao's lab at Virginia Tech with a circular scanning trajectory. The system consists of an Ultrabright ^{TM} microfocus X-ray source (Oxford Instruments, Oxfordshire, UK) and a C7921 flat panel detector (Hamamatsu Corp., Bridgewater, NJ). We built a customized physical phantom, which consists of metal wires, plastic tubes, and metal pins of pre-known

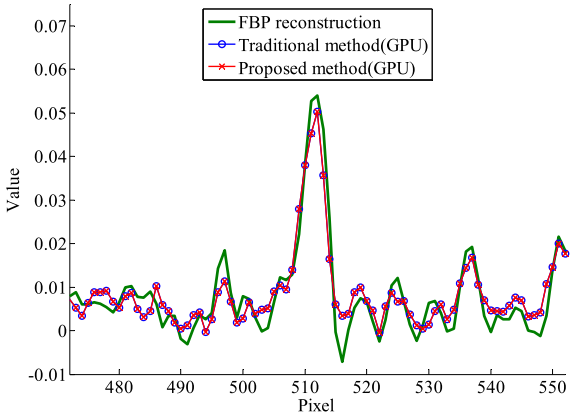


FIGURE 12. Profiles along the central horizontal lines of the reconstructed images in Fig. 11.

materials [29]. The phantom was scanned under 80 kV tube voltage and 250 μA tube current. The source to detector distance was 459.5 mm and the source to rotation center distance was 106.3 mm. The projection data contain 720 projections uniformly sampled over 360ř, and the projection image size is 1024×1024 with pixel at size 50 μm $\times$ 50 μm. We assembled the sinogram for the central CT slice image in the equidistance fan-beam geometry from its cone-beam projection data. The reconstruction image size is 1204×1024 with pixel size 11.5 μm$\times$11.5 μm. In this study, the total global memory of GPU used in our reconstruction is about 6.82 MB.

We performed the SART algorithm using the proposed method and the traditional method on GPU, respectively. For comparison, we also performed reconstruction with the filtered backprojection (FBP) algorithm. The relaxation parameter was 0.2, and the initial value of reconstruction image was set to zero. Due to the presence of noise in the realistic projection data from the physical experiments, more iterations are needed to obtain satisfactory reconstruction result. Fig. 11 shows the reconstructed images using different methods, where the iteration number is three. Fig. 12 shows the profiles comparison.

As shown in Fig. 11, little difference was found. The FBP algorithm is a gold standard for complete projection data. It can be observed in Fig. 12 that the image reconstructed using GPU implementation of our proposed method keeps the same precision with the traditional method, and it is also in good agreement with the result from FBP. These results demonstrate that our proposed method is effective for real data collected from physical CT scanning experiments.

IV. CONCLUSION

In this work, we proposed a new algorithm for forward projection and backprojection in the line integral model (LIM) based SART. Our forward projector is a completely new approach to calculate the intersections of a ray with pixels, which can be performed in a recursive and efficient manner. Most operations in our projector are comparison, addition, and subtraction, which are inherently more computationally efficient compared to the extensive use of multiplication, division, and integer rounding operations in the classical Siddon's projector. Furthermore, our projector does not need to allocate extra arrays, which is necessary for the Siddon's projector. Allocating large arrays in a CUDA kernel is usually difficult, due to the restrictions by hardware limits and software constrains. Therefore, our proposed forward projector is more suitable for GPU implementation. In addition, the backprojection operation is also improved in this work. Compared to the Li's method, our backprojection method calculates only the projection of two (rather than four as required in the Li's method) vertices for a pixel, and our method does not perform the sorting operation in determining the MBI. This improvement can reduce unnecessary computations, especially for the reconstruction of high-resolution image with large image matrix size.

In conclusion, we developed an efficient algorithm for fast forward projection and improved backprojection for LIM-based SART, and further implemented our method on a GPU platform to achieve accelerated reconstruction speed. Results from the numerical phantom and physical phantom experiments show that our method is computationally efficient and accurate. Compared to the CPU-based Siddon's projector and Li's backprojection, our forward projector and improved backprojection implemented on GPU can speed up the reconstruction of LIM-based SART algorithm by 73 times, without loss in reconstruction image quality. Considering that 3D CT reconstruction has been implemented for all mainstream clinical or preclinical CT systems, it is very necessary to extend our approach to the 3D CT reconstruction. In the near future, we plan to extend the algorithm into 3D cone beam CT, and we expect this new algorithm for fast forward projection and improved backprojection could find potential use in speeding up CT image reconstruction speed in many application scenarios.

REFERENCES

[1] R. Gordon, R. Bender, and G. T. Herman, "Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and X-ray photography," *J. Theor. Biol.*, vol. 29, no. 3, p. 471, 1970, doi: 10.1016/0022-5193(70)90109-8.

[2] A. H. Andersen and A. C. Kak, "Simultaneous algebraic reconstruction technique (SART): A superior implementation of the art algorithm," *Ultrason. Imag.*, vol. 6, no. 1, pp. 81–94, 1984, doi: 10.1016/0161-7346(84)90008-7.

[3] A. C. Kak, M. Slaney, and G. Wang, "Principles of computerized tomographic imaging," *Med. Phys.*, vol. 29, no. 1, p. 107, 2002, doi: 10.1118/1.1455742.

[4] M. Jiang and G. Wang, "Convergence of the simultaneous algebraic reconstruction technique (SART)," *IEEE Trans. Image Process. Publ. IEEE Signal Process. Soc.*, vol. 12, no. 8, pp. 957–961, Aug. 2003, doi: 10.1109/TIP.2003.815295.

[5] H. M. Hudson and R. S. Larkin, "Accelerated image reconstruction using ordered subsets of projection data," *IEEE Trans. Med. Imag.*, vol. 13, no. 4, pp. 601–609, Dec. 1994, doi: 10.1109/42.363108.

[6] G. Wang and M. Jiang, "Ordered-subset simultaneous algebraic reconstruction techniques (OS-SART)," *J. X-Ray Sci. Technol.*, vol. 12, no. 3, pp. 169–177, Jan. 2004.

[7] D. Ji, G. Qu, and B. Liu, "Simultaneous algebraic reconstruction technique based on guided image filtering," *Opt. Exp.*, vol. 24, no. 14, pp. 15897–15911, 2016, doi: 10.1364/OE.24.015897.

[8] X. Intes, V. Ntziachristos, J. P. Culver, A. Yodh, and B. Chance, "Projection access order in algebraic reconstruction technique for diffuse optical tomography," *Phys. Med. Biol.*, vol. 47, no. 1, pp. 1–10, 2002.

[9] H. Guan and R. Gordon, "A projection access order for speedy convergence of ART (algebraic reconstruction technique): A multilevel scheme for computed tomography," *Phys. Med. Biol.*, vol. 39, no. 11, pp. 2005–2022, 1994.

[10] H. Guan, M. W. Gaber, F. A. DiBianca, and Y. Zhu, "CT reconstruction by using the MLS-ART technique and the KCD imaging system. I. Low-energy X-ray studies," *IEEE Trans. Med. Imag.*, vol. 18, no. 4, pp. 355–358, Apr. 1999, doi: 10.1109/42.768844.

[11] K. Mueller, R. Yagel, and J. F. Cornhill, "The weighted-distance scheme: A globally optimizing projection ordering method for ART," *IEEE Trans. Med. Imag.*, vol. 16, no. 2, pp. 223–230, Apr. 1997, doi: 10.1109/42.563668.

[12] G. T. Herman and L. B. Meyer, "Algebraic reconstruction techniques can be made computationally efficient," *IEEE Trans. Med. Imag.*, vol. 12, no. 3, pp. 600–609, Sep. 1993, doi: 10.1109/42.241889.

[13] X. H. Wan *et al.*, "Three-dimensional reconstruction using an adaptive simultaneous algebraic reconstruction technique in electron tomography," *J. Struct. Biol.*, vol. 175, no. 3, pp. 277–287, Sep. 2011, doi: 10.1016./j.jsb.2011.06.002.

[14] K. Mueller and R. Yagel, "Rapid 3-D cone-beam reconstruction with the simultaneous algebraic reconstruction technique (SART) using 2-D texture mapping hardware," *IEEE Trans. Med. Imag.*, vol. 19, no. 12, pp. 1227–1237, Dec. 2000, doi: 10.1109/42.897815.

[15] B. Liu and L. Zeng, "Parallel SART algorithm of linear scan cone-beam CT for fixed pipeline," *J. Xray Sci. Technol.*, vol. 17, no. 3, pp. 221–232, 2009, doi: 10.3233/XST-2009-0224.

[16] D. Xu, Y. Huang, and J. U. Kang, "GPU-accelerated non-uniform fast Fourier transform-based compressive sensing spectral domain optical coherence tomography," *Opt. Exp.*, vol. 22, no. 12, pp. 14871–14884, 2014, doi: 10.1364/OE.22.014871.

[17] X. Jia, B. Dong, Y. Lou, and S. B. Jiang, "GPU-based iterative cone-beam CT reconstruction using tight frame regularization," *Phys. Med. Biol.*, vol. 56, no. 13, pp. 3787–3807, 2011, doi: 10.1088/0031-9155/56/13/004.

[18] P. B. Noël, A. M. Walczak, J. Xu, J. J. Corso, K. R. Hoffmann, and S. Schafer, "GPU-based cone beam computed tomography," *Comput. Methods Programs Biomed.*, vol. 98, no. 3, pp. 271–277, 2010, doi: 10.1016/j.cmpb.2009.08.006.

[19] W.-M. Pang, J. Qin, Y. Lu, Y. Xie, C.-K. Chui, and P.-A. Heng, "Accelerating simultaneous algebraic reconstruction technique with motion compensation using CUDA-enabled GPU," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 6, no. 2, pp. 187–199, 2011, doi: 10.1007/s11548-010-0499-3.

[20] Y. Long, J. A. Fessler, and J. M. Balter, "3D forward and back-projection for X-ray CT using separable footprints," *IEEE Trans. Med. Imag.*, vol. 29, no. 11, pp. 1839–1850, Nov. 2010, doi: 10.1109/TMI.2010.2050898.

[21] G. W. H. Yu, "Finite detector based projection model for high spatial resolution," *J. X-Ray Sci. Technol.*, vol. 20, no. 2, pp. 229–238, 2012, doi: 10.3233/XST-2012-0331.

[22] G. L. Zeng and G. T. Gullberg, "Unmatched projector/backprojector pairs in an iterative reconstruction algorithm," *IEEE Trans. Med. Imag.*, vol. 19, no. 5, pp. 548–555, May 2000, doi: 10.1109/42.870265.

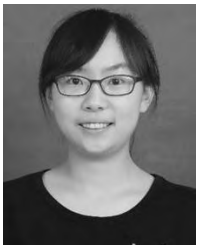
[23] R. I. Siddon, "Fast calculation of the exact radiological path for a three-dimensional CT array," *Med. Phys.*, vol. 12, pp. 252–255, Mar. 1985, doi: 10.1118/1.595715.
[24] F. Xu, "Fast implementation of iterative reconstruction with exact ray-driven projector on GPUs," *Tsinghua Sci. Technol.*, vol. 15, no. 1, pp. 30–35, Feb. 2010.
[25] H. Zhao and A. J. Reader, "Fast projection algorithm for voxel arrays with object dependent boundaries," in *Proc. Conf. Rec. IEEE Nucl. Sci. Symp.*, vol. 3. Nov. 2002, pp. 1490–1494, doi: 0.1109/NSSMIC.2002.1239603.
[26] N. Li, H.-X. Zhao, S.-H. Cho, J.-G. Choi, and M.-H. Kim, "A fast algorithm for voxel-based deterministic simulation of X-ray imaging," *Comput. Phys. Commun.*, vol. 178, no. 7, pp. 518–523, 2008, doi: 10.1016/j.cpc.2007.11.008.
[27] S. Zhang, D. Zhang, H. Gong, O. Ghasemalizadeh, G. Wang, and G. Cao, "Fast and accurate computation of system matrix for area integral model-based algebraic reconstruction technique," *Opt. Eng.*, vol. 53, no. 11, pp. 113101–113109, 2014, doi: 10.1117/1.OE.53.11.113101.
[28] W. Wu, H. Yu, C. Gong, and F. Liu, "Swinging multi-source industrial CT systems for aperiodic dynamic imaging," *Opt. Exp.*, vol. 25, no. 20, pp. 24215–24235, 2017, doi: 10.1364/OE.25.024215.
[29] K. S. Sharma, H. Gong, O. Ghasemalizadeh, H. Yu, G. Wang, and G. Cao, "Interior micro-CT with an offset detector," *Med. Phys.*, vol. 41, no. 6, pp. 061915-1–061915-10, 2014, doi: 10.1118/1.4876724.

SHUNLI ZHANG received the B.S. degree in applied mathematics from Xidian University, Xi'an, China, in 1997, and the M.S. and Ph.D. degrees in aeronautical and astronautical manufacturing engineering from Northwestern Polytechnical University, Xi'an, in 2004 and 2010, respectively.

From 2011 to 2014, he was a Post-Doctoral Researcher with Northwestern Polytechnical University. Since 2014, he has been a Professor with the School of Information Science and Technology, Northwest University, Xi'an. His current research interests include computed tomography, image processing, parallel computing, and the digital restoration of cultural heritage.

GUOHUA GENG received the Ph.D. degree in computer software and theory from Northwest University, Xi'an, China, in 2004. Since 2002, she has been a Professor with the School of Information Science and Technology, Northwest University. Her research interests include image processing, intelligent information processing, and the digital restoration of cultural heritage.

GUOHUA CAO received the Ph.D. degree in physical chemistry from Brown University, Providence, RI, USA, in 2005. After post-doctoral training at Brown University and UNC-Chapel Hill from 2005 to 2007, he became a Research Assistant Professor with the Department of Physics and Astronomy, UNC-Chapel Hill from 2008 to 2011. He is currently an Assistant Professor of biomedical engineering with the Department of Biomedical Engineering and Mechanics, Virginia Tech, Blacksburg, VA, USA. He has been recognized by a number of awards including a NSF CAREER Award in 2014. His research is directed at biomedical imaging, with a focus on developing novel imaging hardware and software for image acquisition and formation.



YUHE ZHANG received the Ph.D. degree in computer applied technology from Northwest University, Xi'an, China, in 2017. Since 2017, she has been a Lecturer with the School of Information Science and Technology, Northwest University. Her research interests include image processing, intelligent information processing, and the digital restoration of cultural heritage.

BAODONG LIU received the B.S. degree in information and computing sciences and the Ph.D. degree in instrument science and technology from Chongqing University, Chongqing, China, in 2005 and 2010, respectively.

From 2011 to 2013, he was a Post-Doctoral Research Fellow with Wake Forest University, Winston-Salem, NC, USA. Since 2013, he has been an Associate Professor with the Institute of High Energy Physics, Chinese Academy of Sciences. He has authored/co-authored over 30 peer-reviewed journal papers. His research interests include computed tomography and image processing.

XU DONG received the bachelor's degree in physics from the University of Science and Technology of China. He is a currently pursuing the Ph.D. degree with the Biomedical Engineering and Mechanics Department, Virginia Tech. Mentored by Prof. G. Cao, X. Dong has been focusing on developing novel CT imaging technology. His research areas cover the imaging system validation and simulation, model development, and medical image processing and analysis.