A Fast GPU-Based Approach to Branchless Distance-Driven Projection and Back-Projection in Cone Beam CT

Daniel Schlifske[ab] and Henry Medeiros[a]
[a]Marquette University, 1250 W Wisconsin Ave, Milwaukee, WI, USA;
[b]GE Healthcare Imaging, 3000 N Grandview Blvd, Waukesha, WI, USA;

ABSTRACT

Modern image reconstruction algorithms rely on projection and back-projection operations to refine an image estimate in iterative image reconstruction. A widely-used state-of-the-art technique is distance-driven projection and back-projection. While the distance-driven technique yields superior image quality in iterative algorithms, it is a computationally demanding process. This has a detrimental effect on the relevance of the algorithms in clinical settings. A few methods have been proposed for enhancing the distance-driven technique in order to take advantage of modern computer hardware. This paper explores a two-dimensional extension of the branchless method proposed by Samit Basu and Bruno De Man. The extension of the branchless method is named "pre-integration" because it gets a performance boost by integrating the data before the projection and back-projection operations. It was written with NVIDIA's CUDA platform and carefully designed for massively parallel GPUs. The performance and the image quality of the pre-integration method were analyzed. Both projection and back-projection are significantly faster with pre-integration. The image quality was analyzed using cone beam image reconstruction algorithms within Jeffrey Fessler's Image Reconstruction Toolbox. Images produced from regularized, iterative image reconstruction algorithms using the pre-integration method show no significant impacts to image quality.

Keywords: computed tomography, projection, back-projection, CUDA

1. INTRODUCTION

De Man and Basu introduced the concept of distance-driven projection and back-projection in 2002 [1]. In 2004, they extended it to three dimensions [2]. In De Man and Basu's 3D distance-driven model, the area of the detector-voxel overlap is used to determine projection and back-projection weights. The distance-driven method provides the accuracy needed to meet the high image quality goals of novel image reconstruction algorithms. There have been many approaches to optimizing distance-driven projection and back-projection on Graphics Processing Units (GPUs). However, all of these fail to address one of the most significant shortcomings of the distance-driven method: the overlap kernel. In the overlap kernel, projection or back-projection sums are accumulated based on the region of overlap of voxel and detector boundaries. Because these boundaries are traversed sequentially, the overlap kernel can be fairly slow on devices with highly parallel Single Instruction Multiple Data (SIMD) architectures such as GPUs.

With the inefficiency of the overlap kernel in mind, Basu and De Man proposed a "branchless" approach in 2006 that works for geometries with evenly spaced detectors such as cone beam [3]. The branchless method factors the overlap kernel into three operations: integration of the input signal, linear interpolation of the integrated signal to obtain values at detector locations, and digital differentiation of the integrated signal at the interpolated detector locations. The method proposed in this paper, called "pre-integration," is an extension of the branchless method. However, pre-integration has the potential for more significant performance improvements because it integrates the input signal in two dimensions, not one. Additionally, several aspects of the pre-integration method are designed specifically for modern GPUs.

2. METHOD

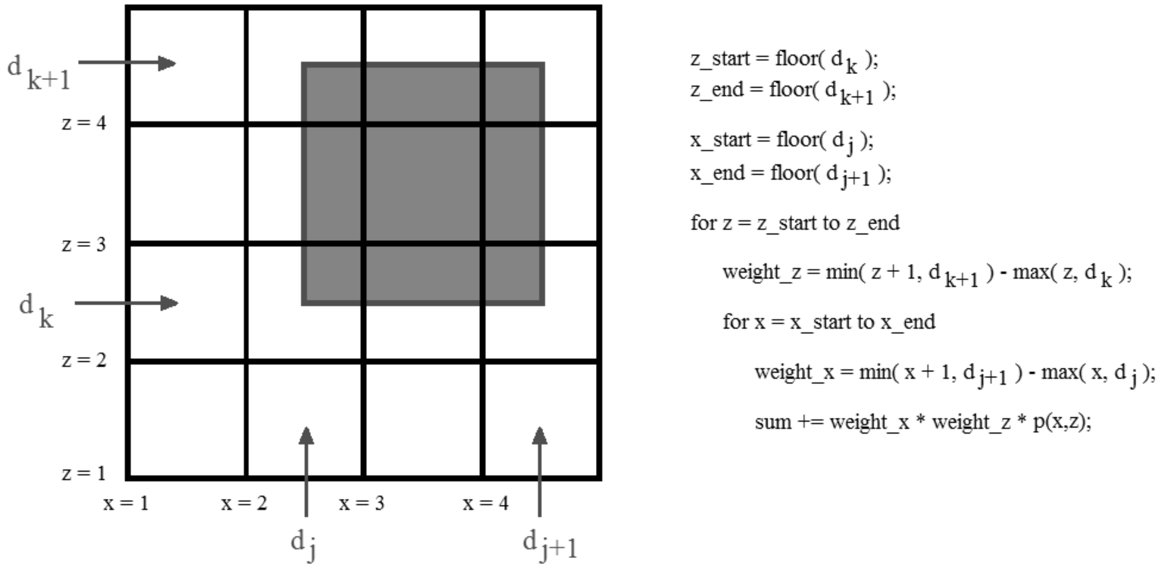
During projection, the algorithm requires the sum of the intensities of the voxels that fall within rectangular detector boundaries: d_j, d_{j+1}, d_k, and d_{k+1}. The function *p(x, z)* represents the image intensities. For each 2D slice in the 3D image volume, the projection operation is essentially a weighted 2D integral, where the bounds of integration are the detector boundaries. This is shown in equation (1). Note that back-projection is similar, but the image and sinogram are reversed.

$$d_{j,j+1,k,k+1} = \frac{1}{d_{j+1} - d_j} \frac{1}{d_{k+1} - d_k} \int_{d_j}^{d_{j+1}} \int_{d_k}^{d_{k+1}} p(x,z)\, dz\, dx \tag{1}$$

The function *p(x,z)* that represents the image intensities is piecewise constant. Therefore, the double integral can be calculated by using an overlap kernel and a summation as shown in equation (2).

$$d_{j,j+1,k,k+1} = \frac{1}{d_{k+1} - d_k} \frac{1}{d_{j+1} - d_j} \sum_x \sum_z p(x,z) \cap d_{j,j+1,k,k+1} \tag{2}$$

The overlap kernel from equation (2) is illustrated in Figure 1. The pseudocode in Figure 1 demonstrates a traditional way to calculate the overlap kernel.



```
z_start = floor( $d_k$ );
z_end = floor( $d_{k+1}$ );

x_start = floor( $d_j$ );
x_end = floor( $d_{j+1}$ );

for z = z_start to z_end
    weight_z = min( z + 1, $d_{k+1}$ ) - max( z, $d_k$ );
    for x = x_start to x_end
        weight_x = min( x + 1, $d_{j+1}$ ) - max( x, $d_j$ );
        sum += weight_x * weight_z * p(x,z);
```

Figure 1. Detector boundaries d_j, d_{j+1}, d_k, and d_{k+1} mapped onto a 2D image slice *p(x,z)* (left) and pseudocode for the overlap kernel (right). The overlap kernel calculates the sum of the image intensities within the rectangular overlap region.

However, note that the sum of the image intensities within any rectangular region can also be calculated using the sum of the intensities of four different rectangular regions: one defined by the origin and the top right point, one defined by the origin and the top left point, one defined by the origin and the bottom right point, and one defined by the origin and the bottom left point [4]. This is shown in Figure 2.

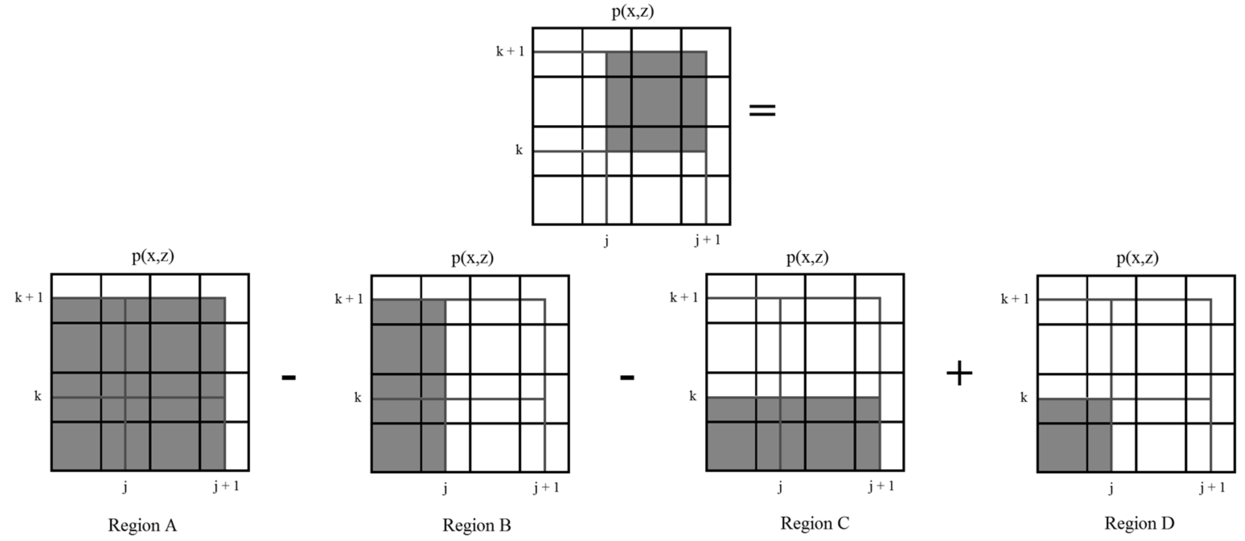


Figure 2. The sum of the intensities in the shaded region in *p(x,z)* (top) can be found by taking the intensities in Region A, subtracting the intensities in Regions B and C, and adding the intensities in Region D.

Pre-integration differs from the standard distance-driven method because it integrates all of the data in each image slice or sinogram view *before* the projection or back-projection operation. The function *P(X, Z)* represents the 2D integral of the image intensities. See equation (3).

$$P(X,Z) = \int_0^X \int_0^Z p(x,z)dz\,dx \tag{3}$$

Integrating the data before the projection can substantially speed up the projection and back-projection operations. The sum of the intensities of any rectangular region of *p(x,z)* defined by the origin and a point *(j,k)* is equivalent to *P(j,k)*. Note that linear interpolation is almost always required to find *P(j,k)* because *j* and *k* are likely not integers. Figure 3 shows how this property is used to find the sum of the intensities of any rectangular region of *p(x,z)* [4].

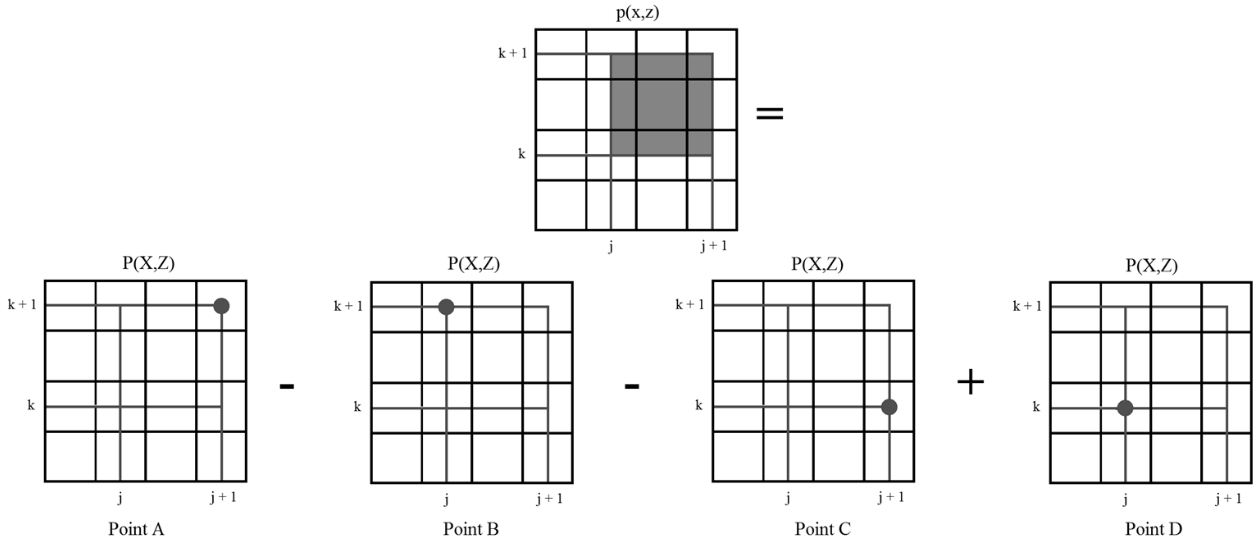


Figure 3. The sum of the intensities in the shaded area of *p(x,z)* can also be calculated by using four linearly-interpolated points from the integrated data, *P(X,Z)*, whose values represent the sum of *p(x,z)* from the origin to each point.

Therefore, the sum of the intensities in the region of overlap can be calculated using the integrated data as shown in equation (4).

$$d_{j,j+1,k,k+1} = \frac{1}{d_{j+1} - d_j} \frac{1}{d_{k+1} - d_k} [P(j+1,k+1) - P(j+1,k) - P(j,k+1) + P(j,k)] \tag{4}$$

Unlike the overlap kernel of the traditional distance-driven technique, equation (4) can be calculated with code that does not contain branches. All threads perform the same memory operations although on different image coordinates. All threads also carry out the same arithmetic operations to determine the sum of intensities in a voxel-detector overlap. For these reasons, the use of pre-integrated data helps the distance-driven technique achieve optimal performance on GPUs.

3. RESULTS

In order to investigate the pre-integration method, a 32-slice GE LightSpeed CT system with a cone beam geometry was simulated. The 3D image volume has dimensions N_X, N_Y, and N_Z, which vary according to the experiment. A 64-bit Linux workstation with two 4-core Intel Xeon CPUs and an Nvidia Tesla K20 workstation graphics card was used. GPU performance was profiled using Nvidia Nsight within the Eclipse IDE. The pre-integration method was evaluated both for its GPU performance and for its impact on image quality.

3.1 GPU Performance Results

The GPU performance was measured for three versions of the distance-driven projector and back-projector: a single-threaded CPU version, a version optimized for GPUs (called the "GPU-optimized" method), and the pre-integration version. The single-threaded CPU version of the distance driven projector and back-projector served as the baseline.

The GPU-optimized versions of the projector and back-projector were written in CUDA and were highly optimized for the Nvidia Tesla K20. Private memory (in the form of registers) was used for the partial projection and back-projection sums in order to limit global memory transactions. However, using a large number of registers per thread meant that the threadblock size needed to be smaller in order to maximize the kernel occupancy. The image data was stored as a texture in order to take advantage of the texture caching on Nvidia GPUs. Finally, in order to get a high L1 cache hit rate on texture reads, the kernel was configured to prefer texture memory over local memory. This means that the GPU decreased the amount of local memory available in order to increase the amount of L1 cache available. The threads in the threadblock were also organized in order to maximize the spatial locality of the texture reads, further improving the L1 cache hit rate.

After optimizing it for the K20, the GPU-optimized projector and back-projector were enhanced with the pre-integration method. First, CUDA functions needed to be written to do the 2D integration of each image slice (for projection) or sinogram view (for back-projection). Then, the overlap kernel in the projector and back-projector were modified to use four linearly-interpolated points from the pre-integrated image or sinogram.

For an image size of $N_X = 512$, $N_Y = 512$, and $N_Z = 48$, the projector and back-projector performance for the single-threaded CPU versions, the GPU-optimized versions, and the pre-integration versions are all shown in Table 1. The table also combines the projection and back-projection performance for each to create an overall performance mark. The pre-integration approach provides a 2x speedup over the GPU-optimized version for this particular system geometry and image size.

Note that the pre-integration method requires all of the 2D slices in the image volume or all 2D views in the sinogram to be integrated in two dimensions before projection. It also requires all 2D views of the sinogram to be integrated before back-projection. The integration times for the entire image and sinogram are twelve milliseconds and twenty milliseconds, respectively. These are shown in the rows labeled "Integration" in Table 1. Fortunately, these execution times are orders or magnitude smaller than the execution times for projection and back-projection.

Table 1. Execution times for various methods of projection and back-projection. In this scenario, the 3D image volume has dimensions of 512x512x48.

Operation	**Execution Time (seconds)**			**Pre-Integration Speedup over GPU-Optimized**
	Single-Threaded CPU	**GPU-Optimized**	**Pre-Integration**	
Integration	-	-	0.012	-
Projection	162.7	1.98	0.61	3.2x
Integration	-	-	0.020	-
Back-Projection	162.5	2.30	1.49	1.5x
Total	**325.2**	**4.28**	**2.13**	**2.0x**

While this combination of image size and CT system geometry y it is important to note that the performance gain of the pre-integration method is dependent on the ratio of the detector cell size to the image voxel size. For example, Figure 4 demonstrates that the pre-integration speedup for projection increases as the number of voxels in the image increases and the voxel size decreases.

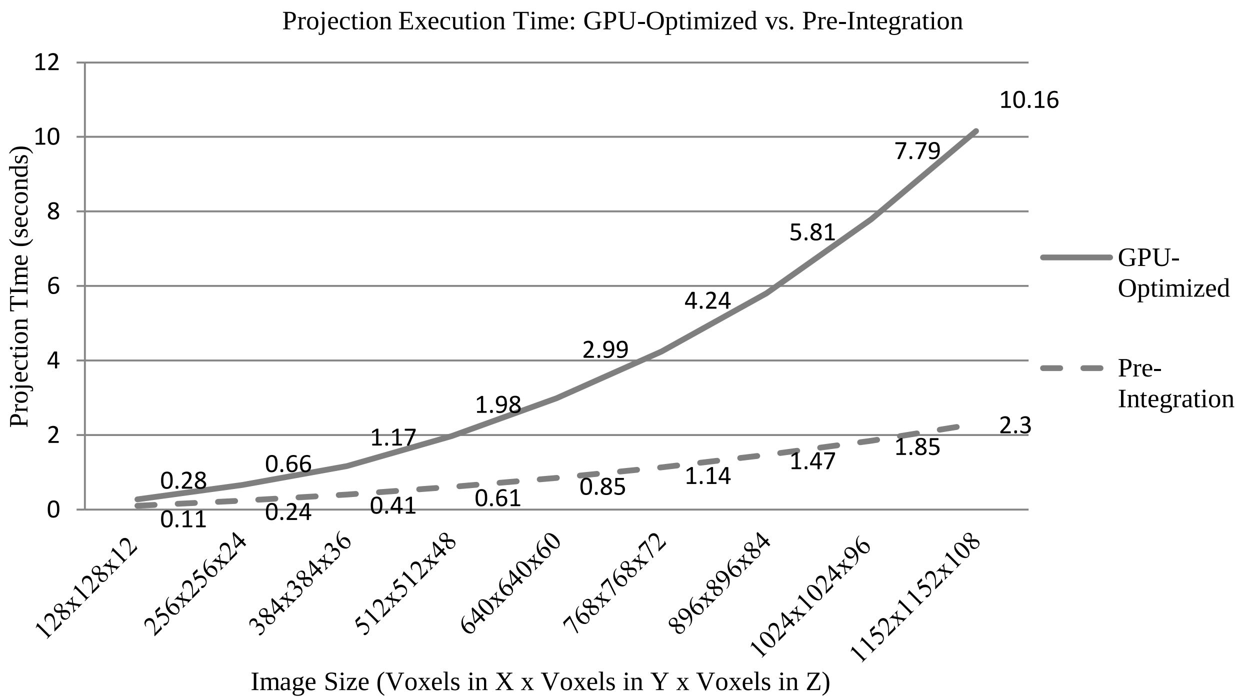


Figure 4. Projection execution times with and without pre-integration. For image volumes with fewer voxels, the pre-integration method is 2.5x faster than the GPU-optimized method. For image volumes with more voxels, the pre-integration method is 4.4x faster.

The performance gain of the pre-integration method for back-projection is also dependent on the detector cell size and image voxel size. However, the result is not as advantageous. Figure 5 demonstrates that the pre-integration speedup for back-projection decreases as the number of voxels in the image increases and the voxel size decreases. Still, pre-integration is at least 20% faster in all of the cases that we examined.

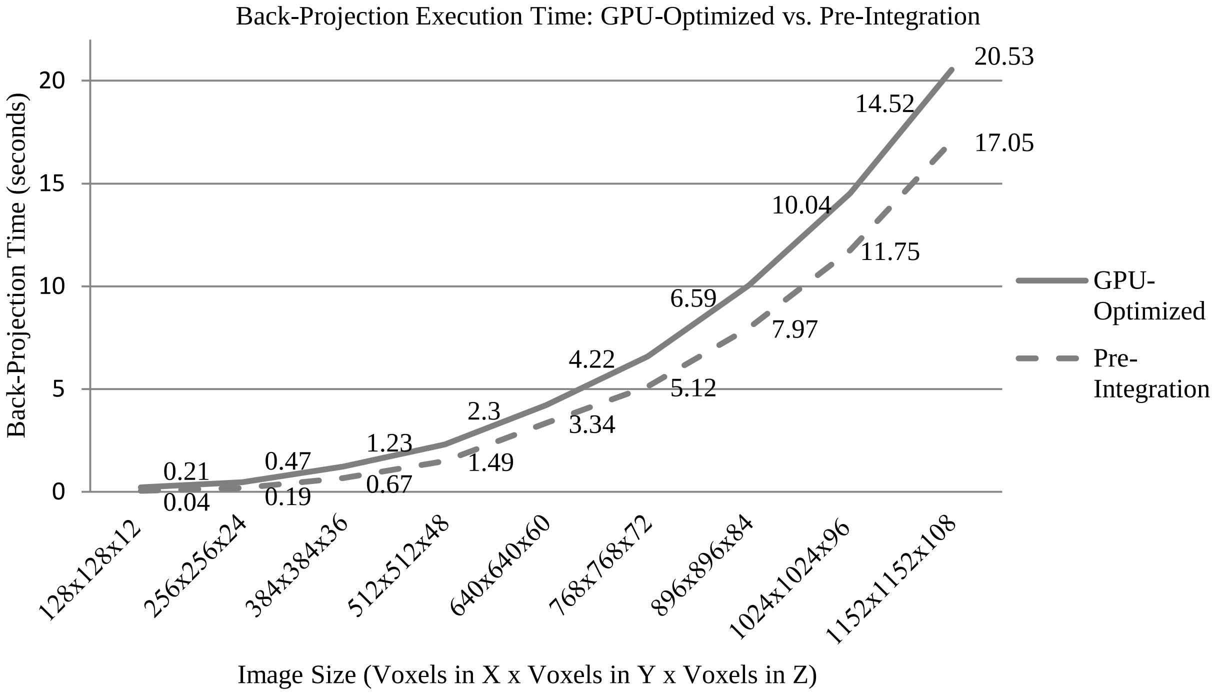


Figure 5. Back-projection speedup from pre-integration. For smaller image sizes (and thus larger voxels), the pre-integration method is 5.3x faster than the GPU-optimized method. For larger image sizes (and thus smaller voxels), the pre-integration method is 1.2x faster.

3.2 Image Quality Results

It is important to note that the integration phase of pre-integration affects the numerical precision of the projection and back-projection operations. The absolute precision of floating-point numbers is lost as the magnitude of the values increase. Specifically, floating-point numbers of greater magnitude have less absolute precision than floating-point numbers of smaller magnitude. Because the values of an image or a sinogram are accumulated in the integration phase of pre-integration, the values being manipulated during projection or back-projection have greater magnitude than they would have without pre-integration. This leads to a loss of precision. However, this loss of precision does not necessarily have an impact on overall image quality.

In order to determine whether the loss of precision affects image quality, the pre-integration method was tested within a regularized, iterative image reconstruction algorithm for cone beam CT from Jeffrey Fessler's Matlab-based Image Reconstruction Toolbox (IRT) [5]. In order to run the CUDA projector/back-projector pairs within the IRT, they needed to be compiled as MATLAB Executable (MEX) programs.

An image size of $N_X = 512$, $N_Y = 512$, and $N_Z = 48$ was used in this experiment. A Penalized Weighted Least-Squares (PWLS) algorithm was chosen to evaluate the pre-integration and GPU-optimized projector/back-projector pairs [6]. The algorithm uses a Preconditioned Conjugate Gradient (PCG) method with a circulant pre-conditioner [7]. Three iterations of the algorithm were run. The true image was known because the input data came from ideal projections of the 3D Shepp-Logan phantom [8].

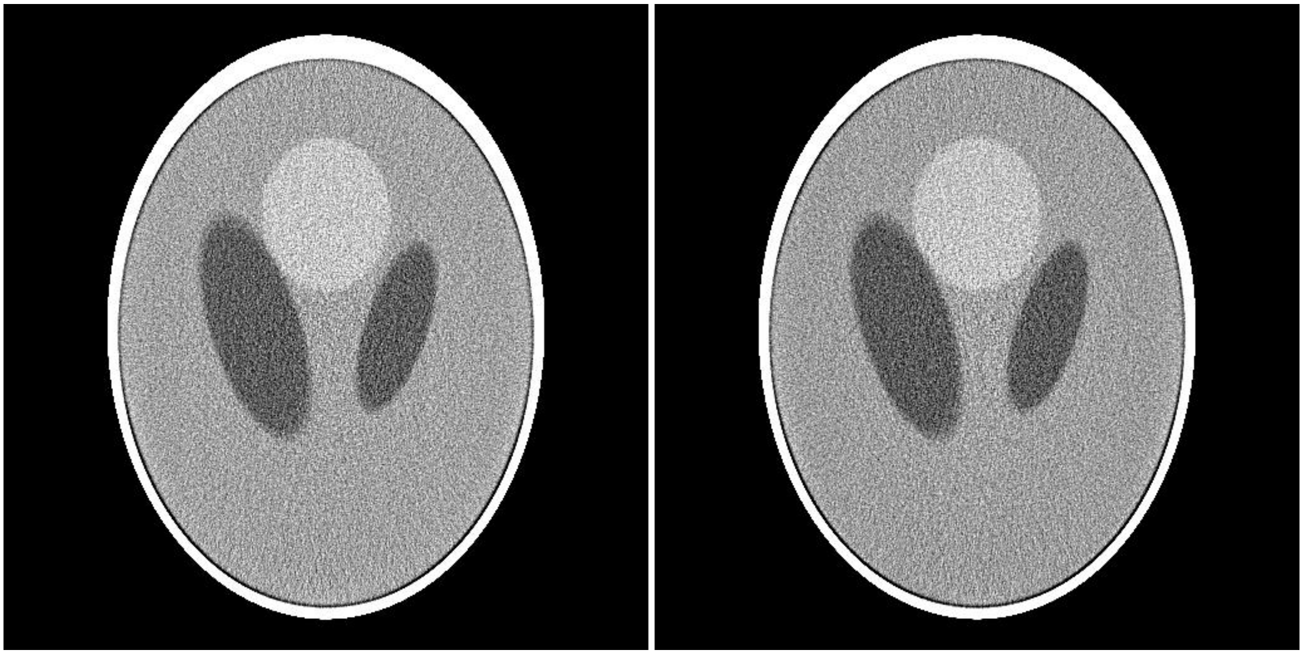


Figure 6. Slices from image volumes created by the PWLS-PCG algorithm. The pre-integration projector and back-projector were used to create the slice on the left and the GPU-optimized projector and back-projector were used to create the slice on the right. The histogram has been stretched to enhance the visualization of the interior of the phantom.

The results from both the GPU-optimized projector/back-projector and the pre-integration projector/back-projector were compared to the true image. A slice from the center of each of the difference images is shown in Figure 7. Visual inspection of the difference images shows that they are almost identical.

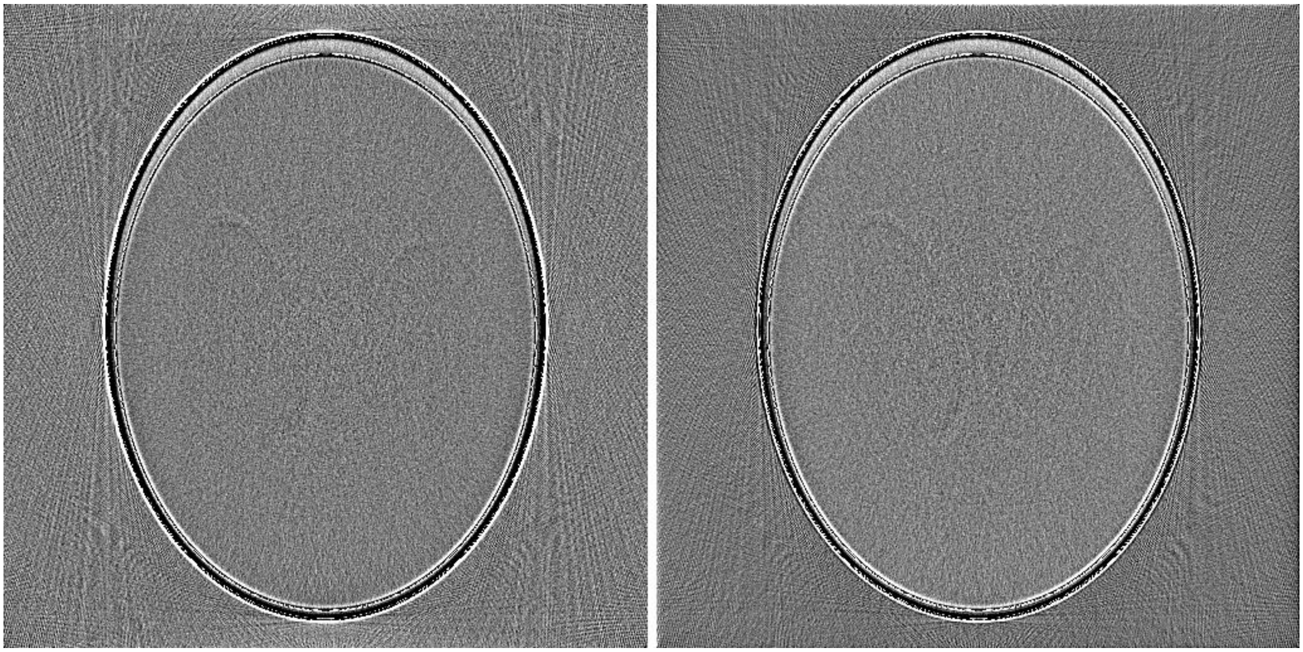


Figure 7. True image minus pre-integration output (left) and true image minus GPU-optimized (right). The histogram has been stretched to enhance the visualization of the error. Both difference images show the same ring artifacts around the edges of the Shepp-Logan phantom.

4. CONCLUSIONS

In this paper, a novel technique called pre-integration is proposed with the goal of speeding up distance-driven projection and back-projection in cone beam CT. In the distance-driven model, both projection and back-projection involve the overlap kernel in which the sum of the elements within a region defined by detector or image boundaries is computed. Although the overlap kernel accounts for a significant portion of the overall projection and back-projection execution time, it has seen very little optimization. The pre-integration technique seeks to minimize the amount of time the overlap kernel takes when executed on GPUs while maintaining the high image quality of the distance-driven model. It uses a pre-integrated image or sinogram to quickly get the sum of the intensities in a region defined by detector or image boundaries in a constant amount of time.

The performance boost that pre-integration can deliver depends on the size of the CT detector and the image being reconstructed. GPUs perform at the highest level when memory accesses are limited. Using pre-integrated data for projections means that a constant, small number of global memory accesses are required. This can substantially reduce the number of reads needed. For example, images that have small voxels would typically require many reads to ascertain all values. Using pre-integration requires only four reads, even if the detector shadow falls upon tens of voxels. However, images that have larger voxels would not require as many reads in the traditional implementation. Thus, using pre-integration would save fewer reads. The speedup that pre-integration offers for projection increases from 2.5x to 4.4x as the number of voxels in the image volume increases. Conversely, the speedup that pre-integration offers for back-projection decreases from 5.3x to 1.2x as the number of voxels in the image volume increases. Still, both offer a speedup of over 4x for certain image sizes.

The pre-integration method has an effect on the precision of the projection and back-projection results, and therefore could have an effect on the overall image quality. To investigate further, a regularized, iterative image reconstruction algorithm was used to compare the pre-integration projector/back-projector pair with a GPU-optimized projector/back-projector pair. Images produced by the PWLS algorithm from Jeffrey Fessler’s IRT showed no differences between the pre-integration projector/back-projector pair and the GPU-optimized projector/back-projector pair.

REFERENCES

[1] De Man, B. and Basu, S., "Distance-driven projection and backprojection," Nuclear Science Symposium Conference Record(3), 2002.

[2] De Man, B. and Basu, S., "Distance-driven projection and backprojection in three dimensions." Physics in Medicine and Biology 49(11), 2463, 2004.

[3] Basu, S. and De Man, B., “Branchless distance driven projection and backprojection,” Proc. SPIE 6065, 60650Y, 2006.

[4] Crow, F.C., “Summed-area tables for texture mapping,” ACM SIGGRAPH Computer Graphics, 18(3), 207-212, 1984.

[5] Fessler, J., “Image Reconstruction Toolbox,” http://web.eecs.umich.edu/~fessler/code/, 2015.

[6] Fessler, J., “Penalized weighted least-squares image reconstruction for positron emission tomography,” IEEE Transactions on Medical Imaging, 13(2), 290-300, 1994.

[7] Fessler, J. and Booth, S. D., “Conjugate-gradient preconditioning methods for shift-variant PET image reconstruction,” IEEE Transactions on Image Processing, 8(5), 688-699, 1999.

[8] Shepp, L. A., and Logan, B. F., “The Fourier reconstruction of a head section,” IEEE Transactions on Nuclear Science, 21(3), 21-43, 1974.