

# Accelerated Generation of Digitally Reconstructed Radiographs using Parallel Processing

Osama Dorgham, Mark Fisher and Stephen Laycock<sup>a</sup> \*

<sup>a</sup> School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, UK

## Abstract.

In this paper we present an approach for speeding-up the generation of Digitally Reconstructed Radiographs (DRRs). DRRs are needed to confirm patient setup before preplanned clinical procedures such as robotic surgery or radiation therapy in a process known as 2D/3D medical image registration. Rendering DRR images is a computationally intensive process and is considered a bottleneck in 2D/3D registration and there has been some recent interest in developing fast rendering techniques. This paper explores high speed rendering of DRR images from a CT data volume by parallel processing on multiple CPU cores. We investigate the relation between the execution time of our parallel DRR algorithm, the number of cores, and the number of rays (resolution) which are used to render the DRR image. We also compare the quality of DRR images rendered using an approximate method and compare this with approaches proposed by others. Our experimental results demonstrate a speed-up of better than three times using 4 CPU cores and better than 5 times using 8 cores. Our approximate approach gives a peak signal-to-noise ratio (PSNR) of 37 dB, which is comparable to that produced by other approximate techniques proposed and represents an overall speed-up of 26 times compared with a conventional ray casting approach.

## 1 Introduction

A Digitally Reconstructed Radiograph (DRR) is a two dimensional simulated image of the human phantom, rendered from medical tomography data sets, such as a Computed Tomography (CT) data set [1]. Rendering DRR images is important for many medical applications, such as, 2D/3D medical image registration [2] and brachytherapy [3]. In radiation therapy treatment systems, DRR floating images are a vital part of the patient positioning process and may be aligned manually or automatically using the 2D/3D registration process.

DRRs are generated from the medical tomography data by summing the attenuation of each voxel along known ray paths through the data volume. However, this conventional ray tracing approach to DRR rendering is an extremely computationally expensive process and forms a bottleneck for different medical applications, like in 2D/3D image registration [4]. Normally, conventional DRR generation requires  $p \times q$  rays to be cast to generate a DRR from a tomography data volume; where  $p$  and  $q$  are determined by the image resolution, usually chosen to match that of the solid-state flat panel X-ray detector.

However, various methods have been proposed to speed up the generation of DRRs. Rassakoff et.al. [5] implemented a special ray-based data structure called an Attenuation Field (AF) to be used in the generation of a DRR instead of the conventional ray casting method. According to the original proposal of light fields by Levoy and Hanrahan [6] and similar work in concept (Transgraphs) introduced by LaRose [7], they provide a way of parameterizing the set of all rays that emanate from a static scene to perform 3D rendering. Their approach uses a so called, *light slab*, which is a convex quadrilateral object consisting of two main planes  $(u, v)$  and  $(s, t)$ . This is used to parameterize each ray in the space as  $R \equiv Pi(u, v, s, t)$  where plane  $(u, v)$  is the focal plane and  $(s, t)$  is the image plane (camera plane). To create an image for an object inside the light slab, infinitely many rays must be calculated. An AF is generated from a set of DRR images rendered by ray casting from multiple view points. Typically 100 AF-DRR are generated for each anatomic and orientation ( $\Delta r = 10^\circ$ ,  $\Delta t = 10mm$ ), using random camera poses. So, for a typical AF 600 AF-DRR are required with resolution of  $256 \times 256$  pixels.

To generate DRR images using an AF Russakoff et.al. generated a sufficiently large number of AF-DRR and used an interpolation scheme to cover the missing ray samples. From the AF they generate AF-DRRs with a resolution of  $256 \times 256$  pixels in about 50 ms on a PC workstation, with a 2.2 GHz. Intel Xeon processor. Although the image quality is poorer than DRR images rendered by conventional ray tracing Russakoff et al. demonstrate that this does not seriously affect the target registration error.

This paper investigates parallel processing (i.e. the simultaneous use of more than one CPU to execute a program [8]) on the CPU for DRR generation. Parallel processing has been employed within many different types of applications.

---

\*email: {O.Dorgham,M.Fisher,S.Laycock}@uea.ac.uk

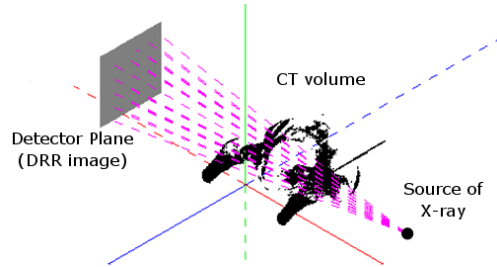
In our case, we generate DRR images by casting rays using more than one processor in order to obtain faster results, which means we are now be able to process more than one ray concurrently. Multi-threaded techniques are well suited to multi-core CPUs found in many modern PCs and we have obtained a significant performance increase in the generation of DRR images with this approach. We generate DRRs from multiple view point without any restriction or limitation on the camera (X-ray source) position for anatomic and orientation ( $\Delta r = 1^\circ, \Delta t = 1mm$ ) using conventional ray casting. We also investigate possibilities for further speedup using an interpolation approach to generate approximate DRR images similar in quality to those introduced by Russakoff et al. In our knowledge this is one of the first studies to generate DRR images exploiting CPU parallel processing which could be used within the 2D/3D registration framework.

## 2 Method

The complexity of DRR generation results from the massive number of calculations needed and the large number of ray casting operations. Compared with more general surface rendering techniques, rendering DRR images is considerably more computationally demanding as we need to compute the attenuation of a monoenergetic beam due to different anatomic material (e.g. bone, muscle tissue etc.) within each voxel, using Beer's Law [9].

$$I = I_0 * exp^{-\sum \mu_i x_i}$$

Where  $I_0$  is the initial X-ray intensity,  $\mu$  is the linear attenuation coefficient for the voxel (material) through which the ray is cast,  $x$  is the length of the X-ray path and subscript  $i$  denotes the voxel index along the path of the ray, as illustrated in Figure 1.



**Figure 1.** Illustration for the process of DRR generation.

Voxel values in CT volumes are represented by a CT number quantified in Hounsfield Units (HU). The attenuation coefficient of the material comprising each voxel can be recovered by [10]:

$$CTnumber = 1000 * [(\mu_i - \mu_w) / \mu_w]$$

where  $\mu_i$  is the attenuation value of a particular volume element of tissue (voxel) and  $\mu_w$  is the linear attenuation coefficient of water for the average energy in the CT beam.

### 2.1 Parallel Processing

Our objective is to increase the speed of DRR generation using parallel processing. To test the approach, we developed and implemented the following algorithm (Algorithm 1) in C++ using the OpenMP library. This algorithm is designed not only to describe how the DRR image will be generated, but also, how the work (generation process) can be distributed and decomposed across the multiple processors.

The special algorithm of box ray intersection points [11], has been implemented internally in Algorithm 1. The aim of this implementation is to quickly calculate the coordinates of the intersection points within the CT volume through the ray path. A point based algorithm [1] is implemented to speed up the rendering process by sampling the intersection points within the CT volume. Different types of rendering algorithm could be implemented, such as ray casting [12], splatting [13] or shear-warping [14], but generally they exhibit a higher time consumption of  $O(N^3)$  time complexity, compared to the point based algorithm of  $O(N^2)$  time complexity [1].

The execution time of the parallel DRR algorithm differs according to the number of cores. Normally a large number increase the number of parallel threads which reduces the rendering time of the DRR images, by casting multiple rays simultaneously. Also, the size of the CT volume affects the total time of the DRR generation and the PSNR ratio for the interpolated versus non-interpolated DRRs. More detailed information is presented in Table 1.

---

**Algorithm 1** Parallel processing of DRR generation using OpenMP

---

```
1: # openmp start parallel for
2: for all  $i$  suchthat  $0 \leq i \leq imgDimX$  do
3:    $count \leftarrow i \times imgDimY$ 
4:   for all  $j$  suchthat  $0 \leq j \leq imgDimY$  do
5:      $x\_ray \leftarrow x\_rays[count]$ 
6:      $absorptionSum \leftarrow 0$ 
7:     for all  $t_0$  suchthat  $stratTime \leq t_0 \leq endTime$  do
8:       if  $CTimg.intersection(x\_ray)$  then
9:          $inrsecPnt \leftarrow x\_ray.getPosition(t_0)$ 
10:         $absrp \leftarrow CTimg[CTimg.offset(inrsecPnt)]$ 
11:         $absorptionSum \leftarrow absorptionSum + absrp$ 
12:      end if
13:    end for
14:     $drr.setAbsorption(absorptionSum)$ 
15:     $count ++$ 
16:  end for
17: end for
```

---

DRR Image Size	Number of Processors								PSNR
	Single		Dual		Quad		Octal		
128×66 (pelvis)	56	14*	28	7*	15	4*	12	3*	29.4 dB
128×86 (lung)	75	19*	38	10*	19	5*	14	4*	30.4 dB
256×133 (pelvis)	689	204*	361	111*	191	<b>59*</b>	127	<b>38*</b>	38.1 dB
256×172 (lung)	879	268*	465	142*	243	76*	164	50*	36.6 dB
512×267 (pelvis)	5570	1629*	2895	884*	1543	478*	987	306*	43.8 dB
512×344(lung)	7158	2153*	3729	1138*	1986	619*	1276	407*	43.1 dB

**Table 1.** Time consumption of pelvis and lung DRR images generation in milliseconds, with and without interpolation\*, and the PSNR ratio for the interpolated versus non-interpolated DRRs. This results have been calculated using two different machines; Intel Core 2 Quad Q6600 2.4 GHz Quad Core Processor and Intel Core 1.8 GHz Octal Core Processor.

DRR rendering requires a large number of rays to generate high-quality images. DRR images rendered at full resolution require  $p \times q$  rays, where  $p$  and  $q$  are the dimensions of the required image. On the other hand, DRR images could be generated using a reduced number of rays, by interpolating the missing values. Generating approximated DRR images, will consume a low time in comparison to the full resolution DRR images, as illustrated in Table 1.

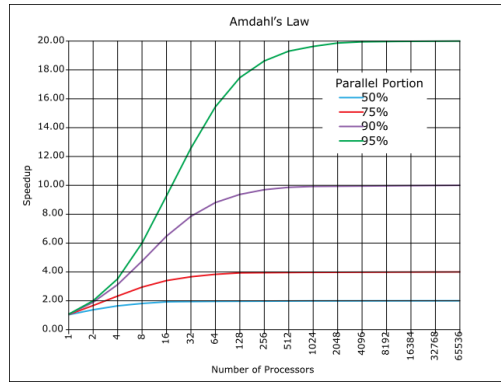
## 2.2 Performance

Writing our algorithm in OpenMP does not offer algorithmic capabilities that are not already available in C or C++. So the main reason to program in OpenMP is performance [15]. An obvious concept to achieving improved performance for parallel implementation is to parallelize a sufficiently large portion of code. But in some cases the performance of the application can be controlled by the serial portion of the program. So, according to Amdahl's law [15], if  $F$  is the parallelized portion of the code and  $S_e$  is the speedup achieved in the parallel portion, the overall speed-up  $S$  will be:

$$S = 1/[(1 - F) + F/(S_e)]$$

Therefore, We can find the maximum improvement of the DRR generation process when it is parallelized, as illustrated in Figure 2.

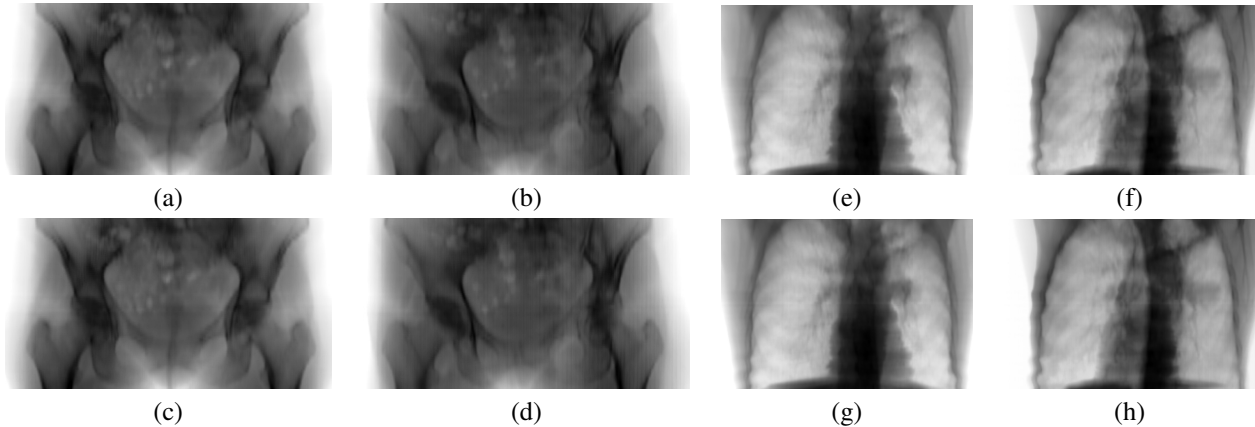
Our results in Table 1, match the curves of Amdahl's law at 90% of the parallel portion, with an overall speed-up of more than three times in comparison between a single and quad cores. This means that the parallelized algorithm represents about 90% of the DRR generation code.



**Figure 2.** The relation between the number of processors and the gained speed up [16].

### 3 Discussion

The acceleration we achieved varied according to the size of the CT volume data and the number of processors. The growth and availability of multi-core technology provides a low cost computing platform to speedup the generation of the DRR images. Validity and durability of our method will not stop while the improvement is going on to the processing capabilities. To enhance the speed of our approach we reduced the number of fired rays to generate an approximated DRR image (Approx-DRR). A nearest neighbour interpolation method [17] is used, providing a low cost computation to approximating the recovery of missing values of the DRR images. Samples of the resulting DRR images using parallel processing with and without interpolation are presented in Figure 3.



**Figure 3.** Sample of DRR images, were generated using pelvis/lung CT volume data, were (a,b,e,f) non-interpolated DRRs and (c,d,g,h) interpolated DRRs with PSNR ratios (43 dB) for the interpolated versus non-interpolated DRRs. All the samples are generated using parallel processing.

We performed a quantitative comparison between the full resolution DRR images (Full-DRR) and the Approx-DRR by computing the peak signal-to-noise ratio (PSNR).

$$PSNR = 20 \times \log_{10} \frac{R}{\sqrt{MSE}}$$

Where  $R$  is the maximum pixel value and  $MSE$  is the mean square error. A set of PSNR ratios were computed for a range of the most common used size (resolution) of the DRR images and from different directions. As in Figure 3, it is hard to notice the difference between the Full-DRR images and the Approx-DRR images, because the PSNR ratios are about 43 dB, and according to Huang et.al. [18], PSNR ratios above 36 dB represent an excellent image quality of compressed images (Approx-DRR).

However, we generate an Approx-DRR image with a resolution of  $256 \times 133$  pixels in approximately 60 ms using an Intel Core 2 Quad Q6600 2.4 GHz Quad Core Processor and in 38 ms using Intel Core 1.8 GHz Octal Core Processor. Our current results are comparable to the results in the literature [5] [4], and it could be improved for the high resolution DRR images  $\geq (512 \times 512)$ , by degrading the PSNR ratio to the acceptable level of image quality  $\geq$

36 dB. Experiments shows that using 25% of the total number of rays is the breaking point of generating Approx-DRR with resolution of  $256 \times 133$  pixels.

## 4 Conclusions

The results show that rendering DRR images can be speeded-up using CPU parallel processing, with speed improvement for more than five times comparing it to the serial approach. Our approach reduced the time required to generate  $256 \times 133$  pixels DRR image from  $256 \times 256 \times 133$  CT volume using the conventional way from 1 second to 59 ms for the approximated DRR image using quad cores, and to 38 ms using octal cores. So, DRR generation has been speeded-up more than 26 times. Furthermore, the quality of DRR images still in the excellent resolution category with PSNR value  $\geq 36$  dB for DRR image with  $256 \times 133$  pixels resolution. An important consideration is that our approach of CPU parallel processing does not require any pre-processing steps unlike other methods of DRR generation as [2] [19] [5] [7]. There is no requirement to apply anti-aliasing, blurring or heavy interpolation methods, in order to enhance the quality of the generated DRR images even when no interpolation is used. Additionally, we generate DRRs from multiple points of view over six degree of freedom x, y, z, yaw, pitch, roll), without any restriction or limitation on the camera position for anatomic and orientation ( $\Delta r = 1^\circ$ ,  $\Delta t = 1mm$ ).

## 5 Acknowledgment

This work was partly funded by EU FP6 project No. LSHC-CT-2004-503564, Methods and Advanced Equipment for Simulation and Treatment in Radio-Oncology (MAESTRO). The authors wish to acknowledge their collaboration with the Colney Oncology Centre, Norfolk and Norwich University Hospital and thank them for providing the CT data.

## References

1. L. L. Aodong Shen. "Point-based digitally reconstructed radiograph." In *ICPR08*, pp. 1–4. 2008.
2. O. Dorgham & M. Fisher. "Performance of 2D/3D medical image registration using compressed volumetric data." In *Proc. Medical Image Understanding and Analysis 2008*, pp. 261–265. July 2–3 2008.
3. N. Milickovic, D. Baltas, S. Giannouli et al. "Ct imaging based digitally reconstructed radiographs and their application in brachytherapy." *Physics in Medicine and Biology* **45**, pp. 2787–2800, 2000.
4. A. Khamene, P. Bloch & W. W. et al. "Automatic registration of portal images and volumetric ct for patient positioning in radiation therapy." *Medical Image Analysis* **10**, pp. 96–112, 2006.
5. D. B. Russakoff, T. Rohlfing & K. M. et al. "Fast generation of digitally reconstructed radiographs using attenuation fields with application to 2D-3D image registration." *IEEE Transaction on Medical Imaging* **24**, pp. 1441–1454, 2005.
6. P. H. M. Levoy. "Light field rendering." *23rd Annu. Conf. Comput. Graph. Interact. Tech. SIGGRAPH* **96**, pp. 3142, 1996.
7. D. A. LaRose. *Iterative X-ray/CT Registration Using Accelerated Volume Rendering*. Ph.D. thesis, Carnegie Mellon University, 2001.
8. K. Vipin, G. Ananth, G. Anshul et al. *Introduction to parallel computing: design and analysis of algorithms*. Benjamin-Cummings Publishing Co., Inc., 1994.
9. R. A. Ketcham & W. D. Carlson. "Acquisition, optimization and interpretation of X-ray computed tomographic imagery: applications to the geosciences." *Comput. Geosci.* **27**, pp. 381–400, 2001.
10. L. B. Schwartz, D. L. Olive & S. McCarthy. *Diagnostic Imaging for Reproductive Failure*. Taylor and Francis Ltd, UK, 1998.
11. W. Amy, B. Steve, R. K. Morley et al. "An efficient and robust ray-box intersection algorithm.", 2005.
12. C. Fox, H. E. Romeijn & J. F. Dempsey. "Fast voxel and polygon ray-tracing algorithms in intensity modulated radiation therapy treatment planning." *Medical Physics* **33(5)**, pp. 1364–1371, 2006.
13. W. Birkfellner, R. Seemann, M. Figl et al. "Wobbled splatting a fast perspective volume rendering method for simulation of x-ray images from ct." *Physics in Medicine and Biology* **50(9)**, pp. N73–N84, 2005.
14. G. S. W. Cai. "Drr volume rendering using splatting in shear-warp context." *Nuclear Science Symposium Conference Record, 2000 IEEE* **3**, 2000.
15. C. Robit, D. Leonardo, K. Dave et al. *Parallel programming in OpenMP*. Morgan Kaufmann Publishers Inc., 2001. 355074.
16. Daniels220. "Amdahl's law.", 2008. Available at: <http://en.wikipedia.org/wiki/File:AmdahlsLaw.svg> .
17. T. M. Lehmann, C. Gnner & K. Spitzer. "Survey: Interpolation methods in medical image processing." *IEEE Transactions on Medical Imaging* **18**, pp. 1049–1075, 1999.
18. H.-C. C. Sheng-Chieh Huang, Liang-Gee Chen. "A novel image compression algorithm by using log-exp transform." In *IEEE International Symposium on Circuits and Systems*, volume 4, pp. 17–20. 1999.
19. D. Sarrut & S. Clippe. "Fast DRR generation for intensity-based 2D/3D image registration in radiotherapy." Technical Report RR-LIRIS-2003-002, LIRIS UMR 5205 CNRS/INSA de Lyon/Universit Claude Bernard Lyon 1/Universit Lumire Lyon 2/Ecole Centrale de Lyon, December 2003.