Published in final edited form as:
Conf Proc IEEE Eng Med Biol Soc. 2012 ; 2012: 3994–3997. doi:10.1109/EMBC.2012.6346842.

Scalable, High-performance 3D Imaging Software Platform: System Architecture and Application to Virtual Colonoscopy

Hiroyuki Yoshida,
Department of Radiology, Massachusetts General Hospital and Harvard Medical School, 25 New Chardon St., Suite 400C Boston, MA 02114 USA (phone: 617-643-2326; fax: 617-643-2743)

Yin Wu,
Department of Radiology, Massachusetts General Hospital and Harvard Medical School, 25 New Chardon St., Suite 400C Boston, MA 02114 USA (phone: 617-643-2326; fax: 617-643-2743)

Wenli Cai, and
Department of Radiology, Massachusetts General Hospital and Harvard Medical School, 25 New Chardon St., Suite 400C Boston, MA 02114 USA (phone: 617-643-2326; fax: 617-643-2743)

Bevin Brett [Member, IEEE]
Intel Corporation, Boston, 25 Manchester Street, Merrimack, NH 03054, USA

Hiroyuki Yoshida: yoshida.hiro@mgh.harvard.edu; Yin Wu: ywu14@partners.org; Wenli Cai: cai.wenli@mgh.harvard.edu

Abstract

One of the key challenges in three-dimensional (3D) medical imaging is to enable the fast turn-around time, which is often required for interactive or real-time response. This inevitably requires not only high computational power but also high memory bandwidth due to the massive amount of data that need to be processed. In this work, we have developed a software platform that is designed to support high-performance 3D medical image processing for a wide range of applications using increasingly available and affordable commodity computing systems: multi-core, clusters, and cloud computing systems. To achieve scalable, high-performance computing, our platform (1) employs size-adaptive, distributable block volumes as a core data structure for efficient parallelization of a wide range of 3D image processing algorithms; (2) supports task scheduling for efficient load distribution and balancing; and (3) consists of a layered parallel software libraries that allow a wide range of medical applications to share the same functionalities. We evaluated the performance of our platform by applying it to an electronic cleansing system in virtual colonoscopy, with initial experimental results showing a 10 times performance improvement on an 8-core workstation over the original sequential implementation of the system.

I. Introduction

In clinical practice, there is an increasing demand for fast turn-around time in obtaining high quality, multi-dimensional images [1]. In particular, real-time 3D imaging and postprocessing thereof is an area with promising potential for a wide range of medical applications.

One of the key challenges in high-speed 3D medical image processing (3D MIP) is that it requires high computational power and data throughput. In addition to the complexity, the main reason why 3D imaging algorithms are time-consuming is the large amount of data to be processed, which often reaches several gigabytes (GB) per examination for practical problems. Further, the data are often needed to be processed repeatedly because many algorithms are iterative in nature, and the data addressing and accessing patterns in these algorithms does not allow us to use simple data distribution approaches due to their high correlation and dependence to the neighboring areas.

Various efforts have been made to achieve the required high performance through careful, hardware-based approaches such as application-specific integrated circuits (ASICs), field programmable gate arrays (FPGAs) [2], and Graphical Processing Units (GPUs) [3]. While these approaches have yielded promising performance for the applications that they intend to process, they are not directly applicable to a wide range of 3D MIP applications required in clinical practice.

Increasingly available high performance multi-processor systems provide a promising avenue to address the above challenges and have the potential to be adaptive to the 3D MIP in clinical practice. Their availability in multiple configurations, such as stand-alone multi-core systems, clusters, and cloud computing systems, allow additional flexibility that could facilitate adoption in large and small clinical settings.

In this work, therefore, we designed a software platform called *high-performance 3D imaging software* (HPC-3D) platform for a rapid development of high-speed 3D imaging applications. The platform has the following characteristics.

First, to address the challenge of large data communication between processors, we employed a size-adaptive, distributed block volume structure [4] that minimizes the data storage, transfer and reduce computation, as a core data structure of the HPC-3D platform (Section II.A). Second, to load balance tasks among computing units and to avoid data locks that may restrict the performance of applications, we developed an efficient task scheduling method [5] that is suitable for a wide-range of 3D MIP algorithms (Section II.B). Third, to enable the maximum applicability to a wide range of 3D MIP algorithms, we designed the HPC-3D platform as layered parallel software libraries with reusable software components at each layer. These layers consist of (a) a block volume library that provides a mechanism to decompose a volume into overlapping sub-volumes and non-overlapping blocks, and distributed them to nodes for processing, and (b) a parallel image processing library that provides reusable parallel operations (Section II.C).

To evaluate the performance of the HPC-3D platform, we implemented an *HPC electronic cleansing* (*EC*) method by parallelizing the computationally intensive steps in our previously developed structure-analysis EC (SA-EC) method [6] on the platform, and compared its execution speed with that of the SA-EC (Section V).

II. High-Performance, Scalable Computing Platform: Architecture Design

A. Distributed Block Volume Structure

The HPC-3D platform employs *distributed block volume structures* (DBVS) as the fundamental data structure to enable efficient parallelization and optimization of a wide range of 3D MIP algorithms.

Fig. 1 illustrates the schematics of the DBVS. The volume data are divided into non-overlapping blocks, or grids, of voxels. The size of the block can be adapted to optimize for performance and parallelism. The cache effect can be tailored to fit the needs of the application or data. Blocks are accessible through an indexed list and they can be distributed to computing units on a multi-core, a cluster, or a cloud system. The background block is a spatially marked block that consists of the voxels that have been determined to be the background of a 3D image. The background block is logically shared among blocks for memory and computational efficiency, and it is replicated only when parallel processing of the blocks that share the same background is required.

The DBVS provides the following advantages: (1) it segregates background voxels into a shared background block for memory efficiency, and thus, it allows obviating unnecessary computation and data communication, which can be substantial in 3D MIP steps in which a large number of blocks are background. (2) A block naturally serves as a data unit to be distributed for parallel processing. (3) The DBVS supports an adaptive block size to optimize for parallelism and performance. DBVS provides adjustable computation data unit per process in order to avoid high overhead caused in repeated iterations or unbalanced workload among processes. This is particularly useful for 3D imaging processing algorithms that are data and data communication intensive, as the performance of such algorithms are extremely sensitive to the efficient use of memory and cache.

B. Parallel Task Scheduling Scheme

It is critically important to load-balance tasks among computing units and avoid data locks that yield suboptimal performance. The DBVS provides a reference to each block about its read/write/process status. To minimize the data wait and lock as well as to maximize local data reuse, we employ a task scheduling method that is suitable for 3D MIP algorithms, called a *wave-front parallelism* on volume block tiles [5]. Wave-front parallelism uses a scheduling algorithm that the local reuse of the data is exploited and the dependences are satisfied to avoid data lock [5].

Wavefront method can avoid wait while increasing data reuse for algorithms that involve multiple layers of iterations and the effect area from last iterations. Many of the 3D MIP algorithms contain nests of loops, in which the same data are reused in successive iterations of the outermost loop. Also, many of the 3D imaging algorithms process voxels that have dependences only on the voxels within the block and possibly the neighboring blocks. These can be achieved by processing various portions of the blocks, block slabs, or block tiles in parallel. In cases where locks are needed, it was often possible to break the iteration space into contiguous blocks, block slabs, or block tiles, and then process these units with intervals that avoided the need of lock.

Fig. 2 shows an example of block volume slabs with dependence on one neighboring block slab. The scheduling is applied to a slab of blocks by processing these interacting block slabs within an outer interval loop. A middle loop can be parallelized by distributing of blocks to different computing units, assuming that there is no shared unit across iterations of this loop, and hence does not need to be locked. The most inner layer is for the computing unit to process and reuse the local data independent of the other units.

C. Layered Parallel 3D Imaging Libraries

The software layers in HPC-3D are designed for easy parallelization of a wide range of 3D MIP algorithms. Fig. 3 shows the architecture of the software layers for the platform. The bottom layer abstracts the hardware platform, enabling targeting of any of the multi-core CPU-based computer system, cluster of computers, or cloud computing.

The rest of the layers are 3D MIP software layers that provide fundamental mechanisms for high-performance parallel processing. The second layer from the bottom is a block volume library that provides a mechanism to decompose a large volume into overlapping sub-volumes and non-overlapping blocks, and distribute them to computing units for parallel processing. The third layer from the bottom is a parallel image processing library that provides reusable operations such as parallel region growing, parallel filtering methods such as Gaussian and Hessian filtering, and parallel histogram analyses. The top layer consists of application-specific algorithms such as the SA-EC method. The right of the diagram shows example components for the SA-EC (see Section III for details).

Parallelization of the image processing library employed Microsoft Parallel Patterns Library (PPL). Many of the 3D MIP algorithms require a large amount of data and iterative computation. Our parallelization takes these factors into account and adds special speed up on the commonly used, computationally expensive algorithms. For efficiency, we created these functions on top of the Intel Integrated Performance Primitives (IPP).

III. Application to Virtual Colonoscopy

CT colonography, also known as *virtual colonoscopy*, is a viable alternative to optical colonoscopy for diagnosis of colorectal cancers [7]. *Non-cathartic* CT colonography (*nc*CTC) is an emerging CTC examination, in which no cathartic agent is used in the bowel preparation [7]. A broad adoption of *nc*CTC is regarded as the most promising next-generation CTC technique for screening of colorectal cancer [7]. However, the caveat is that *nc*CTC introduces a large quantity of solid stool that adheres to the colonic mucosa, which obscures small lesions, and distracts readers from focusing on small polyps.

To improve interpretation, tagged feces must be segmented and removed from the CTC images. Such an approach, called *electronic cleansing* (EC), is a promising approach for “virtually cleansing” of the colon [6] to reveal colonic lesions submerged in the feces. Cai *el al*. [6] have developed an EC scheme for *nc*CTC, called a *structure-analysis EC* (SA-EC) scheme, which effectively removes the solid stool in *nc*CTC. The four major steps in the SA-EC scheme are shown in Fig. 4. SA-EC requires a large amount of data (typically, approximately 1 GB per patient), and its original implementation took approximately 30 minutes per case to process [6].

We thus implemented the SA-EC scheme on the HPC-3D platform to develop a high-performance electronic cleansing (HPC-EC) scheme by parallelizing the computationally intensive steps in the SA-EC scheme as described in the next sections. The HPC-EC program was implemented on Windows Server 2008 with native C++ using Visual Studio 2010 and Intel Parallel Studio XE development environment.

A. Parallel Segmentation of the Colon

The colon segmentation process separates the colonic lumen from other parts of the body on CTC images to define the region of interest and to reduce the use of memory and computational power required in the later stages (Figure 4, Step 1).

The core method in the colon segmentation process is the region growing [8]. Thus, the use of the parallel region growing method in the second software layer in HPC-3D (see Section II.C) can substantially reduce the computation time required for this process, because it takes advantage of bitwise and computing-unit parallelization in the parallel region growing algorithm.

B. Parallel Structure Analysis of the Colonic Lumen

In the structure analysis step of SA-EC, characteristic soft-tissue structures in the colonic lumen, such as the haustral folds and polyps, are recognized and differentiated from stool based on their local morphology that are characterized by the eigenvalues of a 3D Hessian matrix [8].

In this step, Gaussian and Hessian filters are used multiple times to compute the local morphologic features. This process is both computationally expensive and memory intensive, because nine first and second derivatives are computed in this process. We thus employed the parallel Gaussian and Hessian filters in the second software layer in HPC-3D (see Section II.C) to speed-up this step.

C. Parallel Local Roughness Analysis

Local roughness, which is defined as the sum of the differences between the volumetric curvedness across scales, is used to differentiate an air-tissue-tagging layer from an air-tagging boundary [6]. The curvedness calculation at different scales involves repeated computation of Gaussian derivatives. We thus employed the parallel Gaussian derivatives filters in HPC-3D to speed-up this process.

D. Parallel Dynamic-Threshold Level Set Method

Dynamic-threshold level set method [9] is used to segment and remove the tagged fecal materials for recovering colonic wall, while preserving submerged soft-tissue structures such as polyps and folds. This step was performed by first initializing the level set front with the tagged regions that are segmented by thresholding of the image with a pre-defied CT numbers; then, the level set front is evolved by use of a speed function that was designed to stop the evolution of the front from evolving at the boundary between the tagged regions and the soft-tissue structures or the colonic wall, thus recovering the colonic wall.

We employed the wave-front parallel scheduling (Section II.B) to the dynamic-threshold level set method for efficient parallelization of this step. In this approach, seeds in the initial level set front with a pre-defined interval block distance (a block set that is not interacting during a predefined iteration periods) are computed in parallel. Fig. 5 illustrates a situation in that some potential tagging areas in the block sets with no interaction are processed in parallel in the first step, while the potential tagged areas in other noninteracting block sets are processed in parallel in the next step. The process will be repeated until the level set method converges.

IV. Experimental Results

We evaluated the execution speed of the HPC-EC scheme based on computational workstations equipped with 2-, 4-, and 8-core CPUs (Intel Xeon series, 3GHz) and 8 GB memory, and compared their execution time with that of the original, sequential SA-EC scheme.

Table I shows the preliminary results based on 5 sets of CTC image data (512×512×340–600 voxels in size). The average execution times were 28 min and 2.6 min on the original SA-EC and the HPC-EC, respectively, indicating that the HPC-3D platform enabled a 10-fold reduction in computing time (Table I).

Individual steps in SA-EC were also speeded up at a similar or greater proportion: Colon segmentation: 33-fold reduction (5 min to 9 sec); structure analysis: 12-fold reduction (4 min to 21 sec); roughness analysis: 12-fold reduction (7 min to 35 sec); dynamic level set method: 8-fold reduction (12 min to 90 sec). These results indicate that HPC is a useful platform for enabling near-real time computation in the EC processes.

The results also indicate that the total and individual execution times reduces as the number of CPU cores increases, indicating the high scalability of the HPC-EC and the HPC-3D platform (Fig. 6). However, the reduction of the execution time was not linear to the number of cores mainly because of data communication overhead and memory access bottleneck.

V. Conclusion

We have developed a software platform, called HPC-3D, designed to support high-performance 3D MIP using commodity parallel computing systems. Preliminary evaluation of the performance showed that the platform was effective in substantially reducing the execution time of computationally intensive processes in EC for CTC. It is thus expected to be useful for high-performance computing platform for 3D MIP.

Acknowledgments

The project described was partly supported by Grant Numbers R01CA095279, R01 CA131718, R03CA156664, and R01CA166816 from National Cancer Institute (NCI).

References

1. Blackmore CC, Mecklenburg RS, Kaplan GS. Effectiveness of clinical decision support in controlling inappropriate imaging. J Am Coll Radiol. 2011; 8(1):19–25. [PubMed: 21211760]
2. Li J, Papachristou C, Shekhar R. An FPGA-based computing platform for real-time 3D medical imaging and its application to cone-beam CT reconstruction. J Imaging Science and Technology. 2005
3. Sharpand GC, Kandasamy N, Singh H, Folkert M. GPU-based streaming architectures for fast cone-beam CT image reconstruction and deformable registration. Physics in Medicine and Biology. 2007; 52:5771–5783. [PubMed: 17881799]
4. Jackins C, Tanimoto SL. Oct-trees and their use in representing 3-d objects. Computer Graphics and Image Processing. 1980; 14:249–270.
5. Manjikian, N.; Abdelrahman, T. Scheduling of wavefront parallelism on scalable shared-memory multiprocessors. 25th Int'l. Conf. on Parallel Processing; Aug. 1995; 1995.
6. Cai W, Yoshida H, Zalis ME, et al. Informatics in Radiology: Electronic Cleansing for Non-cathartic CT Colonography: A Structure-Analysis Scheme. Radiographics. Mar 10.2010
7. Levin B, Lieberman DA, McFarland BR, et al. Screening and surveillance for the early detection of colorectal cancer and adenomatous polyps, 2008: a joint guideline from the American Cancer Society, the US Multi-Society Task Force on Colorectal Cancer and the American College of Radiology. CA Cancer J Clin. May-Jun;2008 58:130–60. [PubMed: 18322143]
8. Junichiro Toriwaki, HY. Fundamentals of Three-Dimensional Digital Image Processing. Springer; 2009.
9. Cai W, Holalkere N, Harris G, et al. Dynamic-threshold level set method for volumetry of porcine kidney in CT images: in-vivo and ex-vivo assessment of the accuracy of volume measurement. Acad Radiol. 2007; 14:890–896. [PubMed: 17574138]

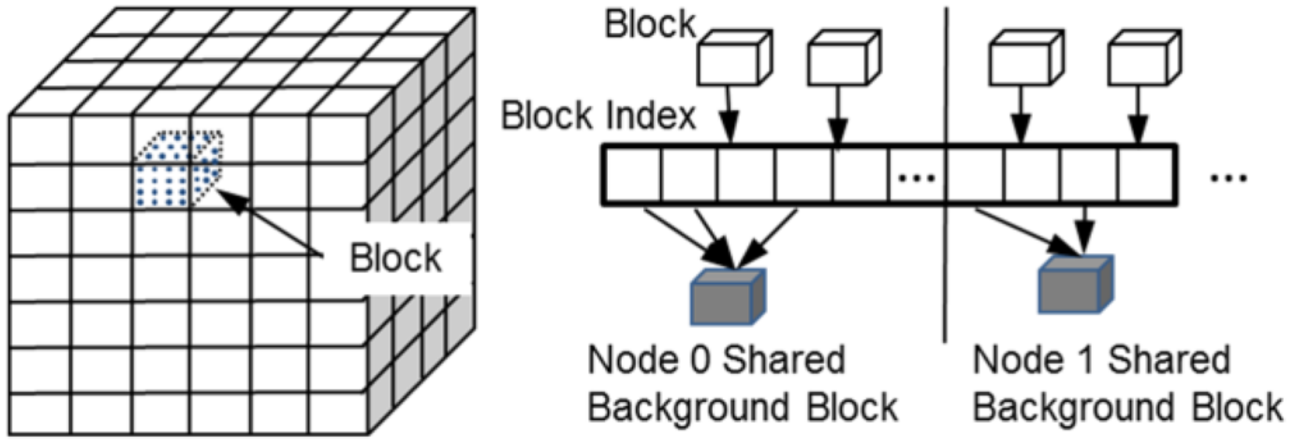
Block
Block
Block Index
...
...
Node 0 Shared Background Block
Node 1 Shared Background Block

Figure 1.
Distributed block volume structure.

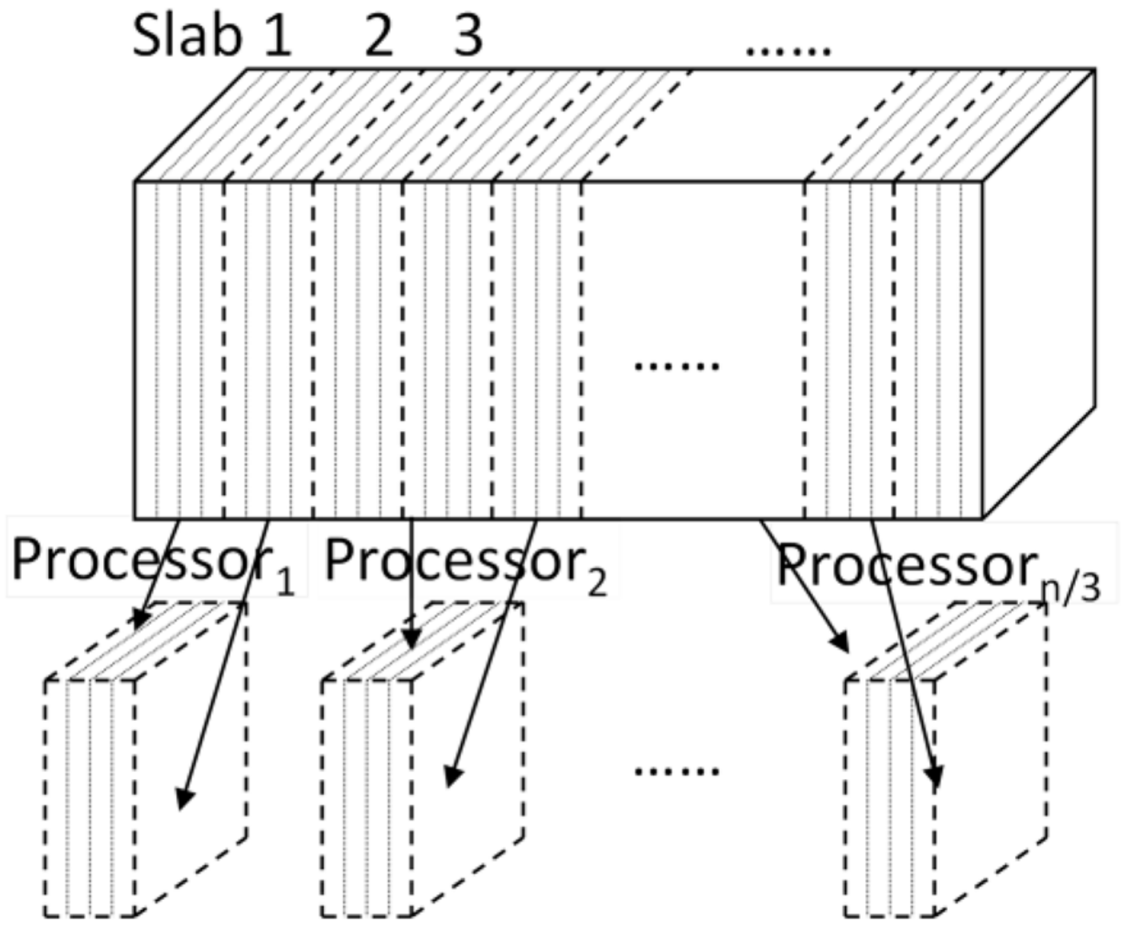

Figure 2.
Distribution of block volume slabs.

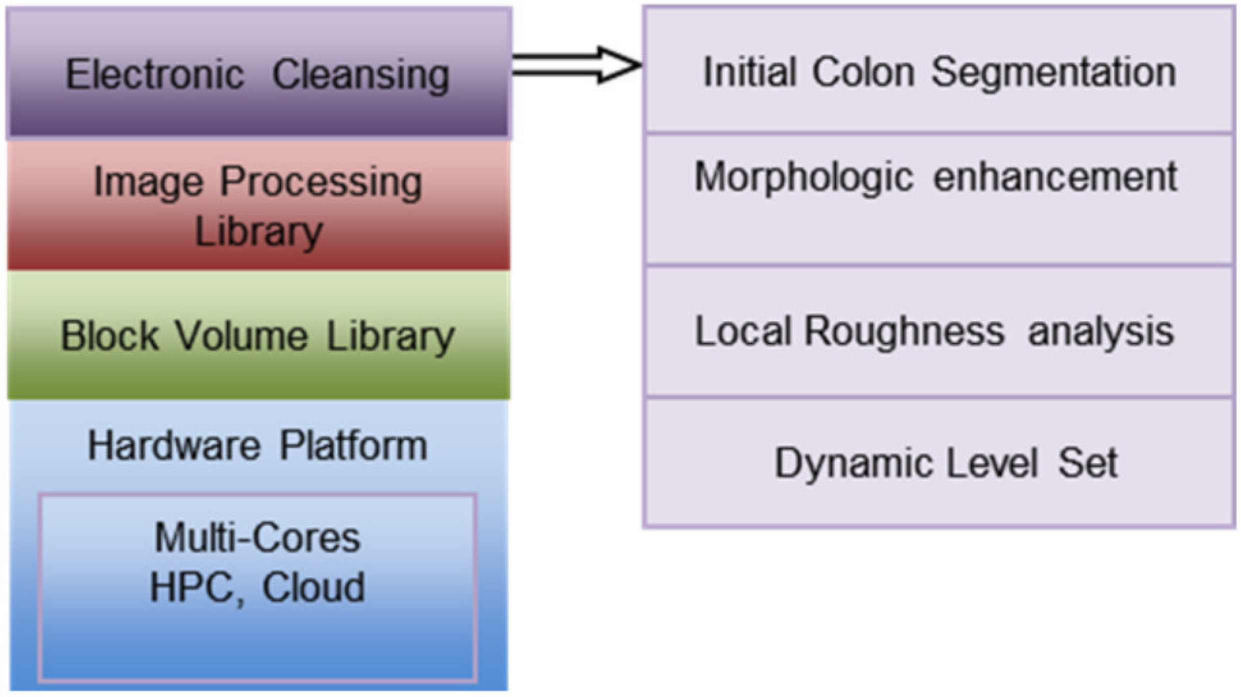

Figure 3.
Parallel software library layers in the HPC-3D platform. The right shows the detailed layer of the EC application layer.

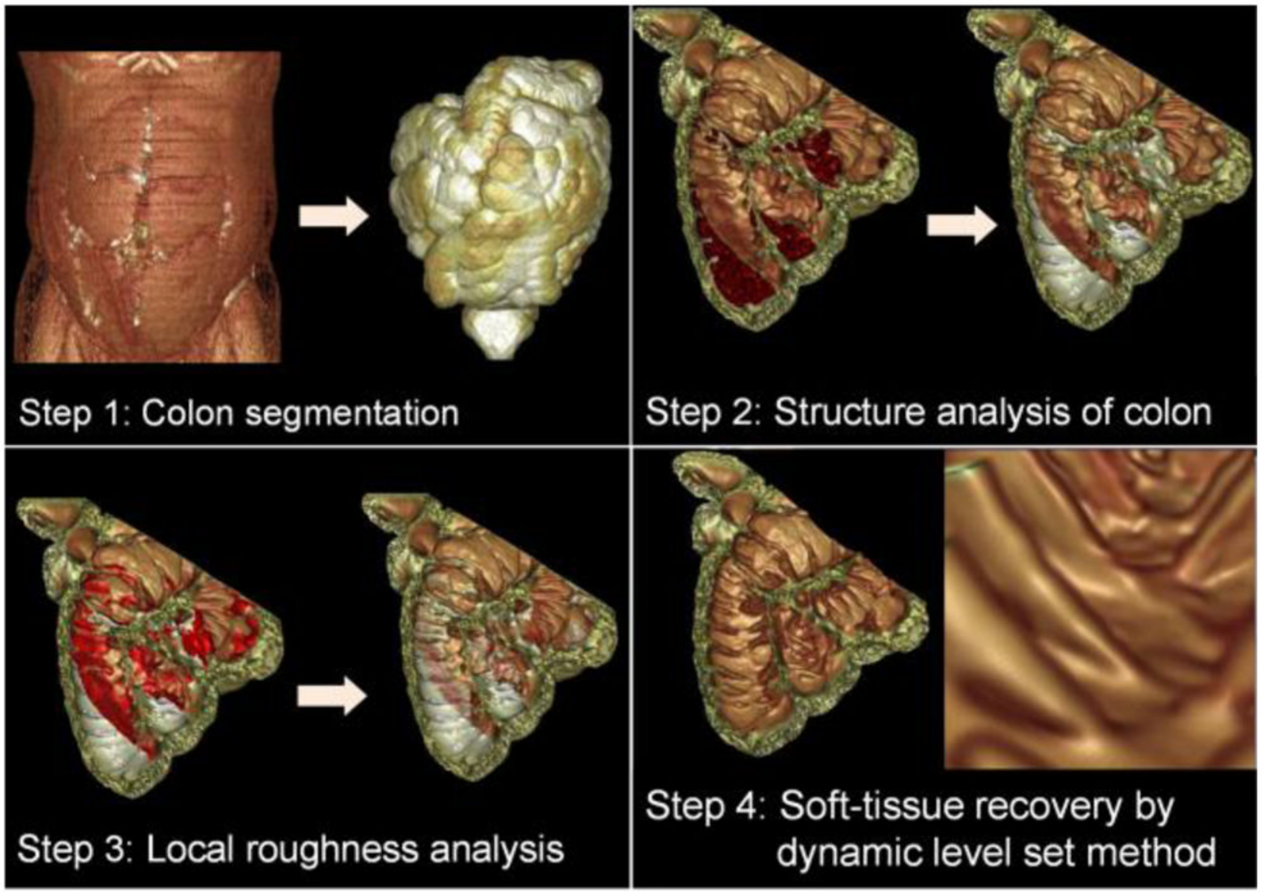
Step 1: Colon segmentation
Step 2: Structure analysis of colon
Step 3: Local roughness analysis
Step 4: Soft-tissue recovery by dynamic level set method

Figure 4.
The four major steps in the structure-analysis electronic cleansing process.

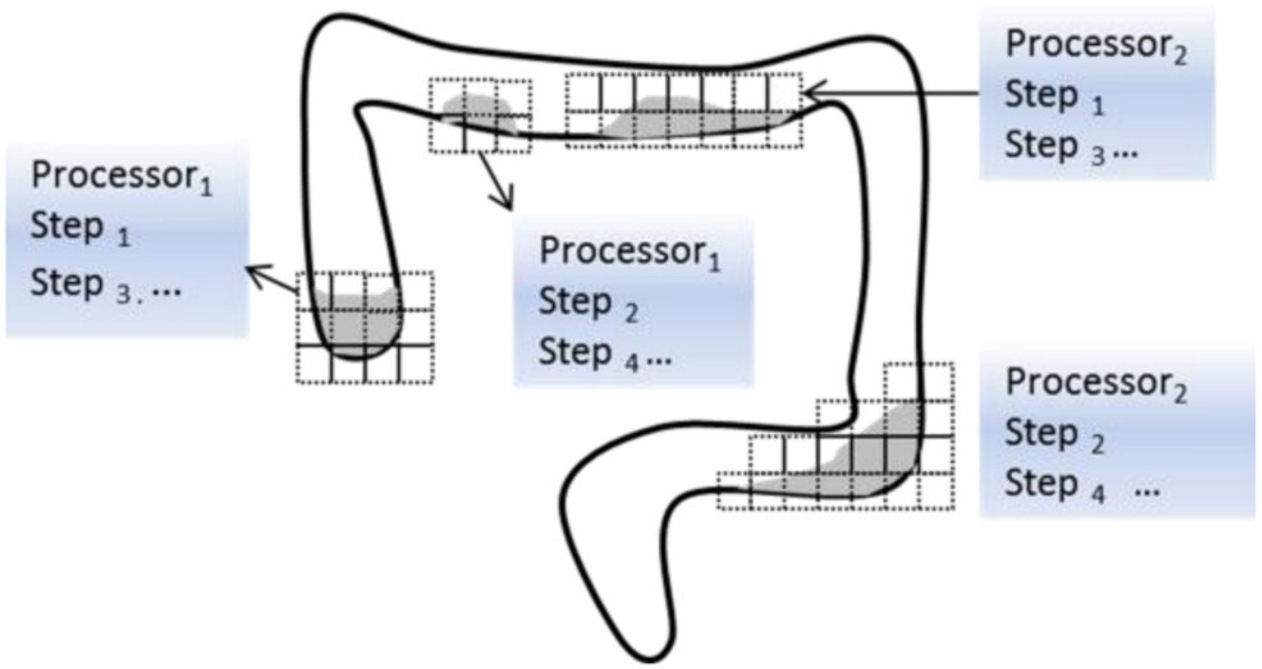

Figure 5.
Parallelization of dynamic level set method.

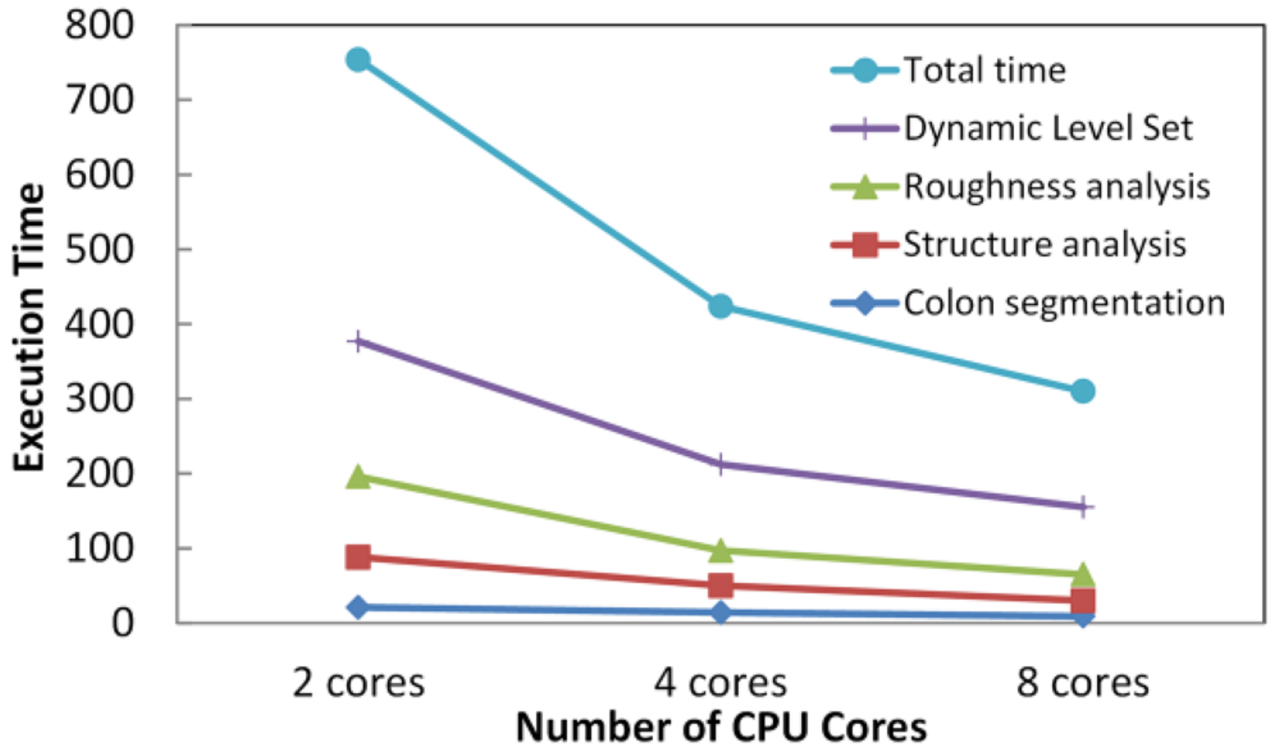

Figure 6.
Reduction of execution time as the number of CPU core changes in HPC-EC.

TABLE I

Comparison of Executin Time in HPC-EC and SA-EC

# of cores		**EC Steps**				
		Colon segmentation **(sec)**	***Structure analysis*** **(sec)**	***Roughness analysis*** **(sec)**	***Dynamic Level Set*** **(sec)**	***Total time*** **(sec)**
SA-EC		300	240	420	720	1680
HPC-EC	2 cores	21	67	108	181	377
	4 cores	14	36	47	115	212
	8 cores	9	21	35	90	155