High-Speed Reconstruction for C-Arm Computed Tomography

Benno Heigl, Markus Kowarschik

***Abstract*—For the applicability of C-arm computed tomography in the interventional environment, fast computational times have to be reasonably achieved. To achieve timings below one minute for high-quality soft-tissue protocols, latest FPGA technology and software optimization for multi-core CPUs have been applied. For a comprehensive reduction of computational time, successively all potential bottlenecks have been identified and eliminated.**

***Index Terms*—Image reconstruction, Field programmable gate arrays, Software performance**

I. INTRODUCTION

IN the last several years, the use of 3D imaging with C-arm systems has become state of the art, particularly for high-contrast objects. In the most prevalent applications, contrast agent is used for angiography applications. For these applications, usually no more than 150 projection images have been acquired in a rotational x-ray run around the patient. For a subsequent analysis by the physician of the vessel topography and shape, it was sufficient to view the reconstructed data set in the viewing application, in about one minute.

With the release of DynaCT in 2005 by Siemens Medical Solutions, this 3D imaging capability has been improved such that besides high-contrast structures, also low contrasts like soft tissue and bleedings could be visualized. Besides acquiring an increased number of projection images (more than 500), these enhancements could be achieved only by both improving the X-ray sensor using flat panel detectors as well as the reconstruction algorithms to cope with physical influences like scattered radiation, beam hardening, and truncated projections, that usually are neglected in standard reconstruction algorithms such as the widely used FDK algorithm [1]. A detailed description of the correction algorithms used for DynaCT can be found in [2].

Although the achievable imaging performance showed CT-like image quality, the practical application during the interventional procedure was partially restricted due to the long computational times. The duration of a 3D volume computation is dependent on chosen acquisition protocol and took up to several minutes.

the applicability was partially restricted because the computation of the 3D volume took up to several minutes. To reduce computational time, the whole reconstruction pipeline was analyzed concerning computational efforts. Based on these results, an optimized solution was designed and realized, leading to clinically satisfying computational times.

After a brief description of the reconstruction pipeline under consideration, we show a performance analysis of the original implementation, the optimization by hardware and software, and finally the achieved results.

B. Heigl is with Siemens AG Medical Solutions, division AX in Forchheim, Germany (e-mail: benno.heigl@siemens.com).

M. Kowarschik is with Siemens AG Medical Solutions, division CO in Erlangen, Germany (e-mail: markus.kowarschik@siemens.com).

II. ALGORITHMIC PIPELINE

In this section, we briefly recapitulate the pipeline of used algorithms with a particular focus on computational efforts. For algorithmic details, we refer to [2] and [3]. The whole reconstruction process can be sub-divided into three parts:

A. *2D Pre-Processing*

Before creating a 3D volume, each projection image passes a sequence of 2D image processing algorithms.

Intensity and beam hardening correction assures that observed gray values correspond to a measured normalized intensity and restores a linear relationship between projection value and path length through an assumed water-equivalent object. The computational complexity is minimal, because each pixel can be processed separately with few floating point operations. Nevertheless, several table look-ups consume a non-negligible amount of time, because some of them are multi-dimensional and need interpolation between neighboring samples.

The **scatter estimation** and **correction** reduces the influence of scattered radiation. For the estimation step, the projection image is downsized and a non-linear function is applied to each pixel value. It is then followed by a convolution with a separable smoothing kernel which results in an image showing the estimated scatter distribution. With this image, the projection image is corrected by single-pixel operations. The process of downsizing and convolution both require the consideration for a wide image neighborship. Hence much more computational efforts are spent during these steps than for the previous ones.

Truncation correction extends each particular image line by estimating the location and extension of a water cylinder model and Gaussian extrapolation. As a result of this step, the projection image is extended horizontally up to double length.

The FDK algorithm requires the convolution of each image line with a high-pass reconstruction kernel having identical size to the extended image line. This step usually is named **filtering** and is realized by applying FFT, point-wise multiplication in frequency domain, and inverse FFT.

B. Back-Projection

The computationally most expensive step is the creation of a 3D volume out of the pre-processed projection images, called back-projection. For each projection image, and for each discrete volume element (voxel), the hit-point of the corresponding X-ray beam onto the projection image is computed, a bilinear interpolation between the four neighboring pixels is performed, and the result is accumulated to the current voxel. The projection geometry is described by matrices transforming 3D volume coordinates to 2D image coordinates, see [3]. The back-projection requires the most extensive computational power because of both the deeply nested loops resulting in a high number of arithmetic operations, as well as a huge amount of data which cannot be processed sequentially, but rather in an irregular order given by the projection geometry.

C. 3D Post-Processing

Similar to CT scanners, detector gain inhomogeneities cannot completely be eliminated by calibration and finally cause ring artifacts in the reconstructed volume slices. To correct this effect, a ring image is computed for each slice by converting it to a polar grid, applying a median filter in the radial direction, followed by a subsequent smoothing in circular direction and a conversion back to a Cartesian grid. Finally, the ring image is subtracted from the reconstructed slice.

III. Performance Analysis

The following table shows the time fractions of the particular algorithmic steps described in the previous section:

TABLE I
DISTRIBUTION OF CALCULATION TIMES

Algorithmic step	Fraction of computational time
Pre-processing (w/o filtering)	5%
Filtering	19%
Back-projection	71%
Post-processing	5%

In this example, the relative timings were measured running the whole reconstruction pipeline on a single-core CPU for 543 projection images with 1240x960 pixels each and a volume consisting of 512x512x440 voxels. The cone angle is 19 degrees in rotational direction and 14 degrees in axial direction.

The table clearly identifies the filtering and back-projection as the computationally most expensive parts and thus contributing to a huge bottleneck in the computational chain. It is obvious that the major focus of optimization had to be given to those two algorithms. Therefore, we decided to introduce additional hardware for this purpose. This decision was further supported by the fact that the algorithms for filtering and back-projection are fixed and compared to the other correction algorithms, do not seem to contain a similar potential for further algorithmic improvements. Thus, a realization of filtering and back-projection in hardware, combined with pre- and post-processing realized in software, results in a reasonable compromise between reduced computational time and flexibility for algorithmic improvements.

IV. FPGA Based Hardware Acceleration

As was outlined previously, the reconstruction step covers about 90% of the image processing pipeline and thus represents its most time-consuming part. In order to drastically reduce image reconstruction time and, as a consequence, to expedite interactivity in clinical workflow, the hardware accelerator platform ImageProX (Image Processing Accelerator) by Siemens Medical Solutions is employed.

ImageProX denotes a family of scalable FPGA (field programmable gate array) based boards for both PC and server computers. These hardware components are used to speed up computationally intensive algorithmic tasks; e.g., in the application area of medical imaging. Due to the dynamic reconfigurability of FPGAs, ImageProX represents a flexible general-purpose computing platform. See [4] for details on reconfigurable hardware architectures, particularly FPGAs.

Fig. 1 shows the ImageProX accelerator platform with the cooling elements and the fans being removed for the sake of illustration. ImageProX covers nine Xilinx Virtex-4 FPGAs [5]. There is one Virtex-4 SX55 chip located in the center of the board, which is typically used for control and communication tasks. The remaining eight Virtex-4 SX35 chips are arranged in two independent rings located to the left and to the right of the control FPGA, respectively.

Each FPGA is assigned a local DDR2 SDRAM memory module. Currently, ImageProX can be equipped with up to 9 GByte of external memory (1 GByte assigned to each FPGA).

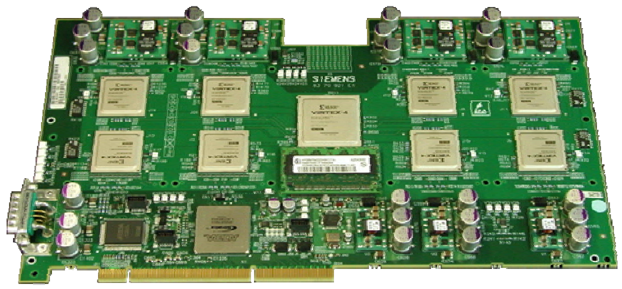

Fig. 1: ImageProX accelerator platform.

In order to speed up the image reconstruction step of the algorithmic pipeline described previously, the well-known FDK method [1] has been implemented on ImageProX. The linewise filtering of the projection data in the Fourier domain is performed during projection upload by the center SX55 FPGA, with any two adjacent image rows being filtered simultaneously. The remaining eight SX35 FPGAs perform the expensive task of back-projecting the (filtered) projection data into the volume, where the projections are equally distributed across them such that each SX35

FPGA's external memory stores approximately the same number of projection images.

Fig. 2 illustrates the internal back-projection design of an SX35 FPGA. The term “ICI” stands for “inter-chip interconnect” and refers to the high-speed communication paths between neighboring FPGAs.

Due to the limited memory capacity of FPGAs (approx. 430 kByte per Virtex-4 SX35 chip) the volume can only be reconstructed in small blocks (subvolumes) of 16^3 voxels each, for example. The required projection data is retrieved from external memory, bilinearly interpolated, and then back-projected into the currently computed subvolume. In order to hide memory access time, double buffering is employed. This means that projection data is loaded into one of two projection data buffers while the data stored in the respective other buffer is being back-projected. A similar approach targeting the Cell BE microprocessor architecture is presented in [6].

Since there are 16 parallel back-projection units (PBUs) per SX35 FPGA (cf. Fig. 2), there are eight SX35 FPGAs running in parallel, and the design currently runs fully pipelined at a clock rate of 200 MHz, the ImageProX accelerator hardware is capable of performing up to $25.6 \cdot 10^9$ back-projection steps per second.

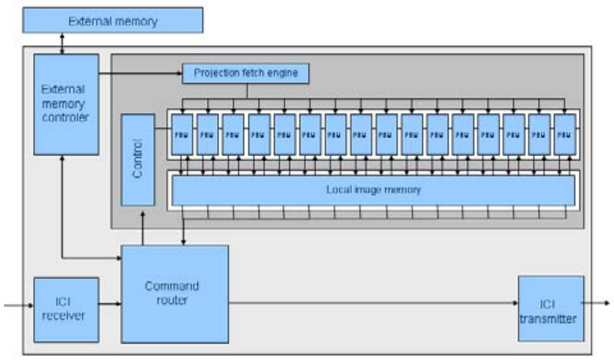

Fig. 2: Back-projection FPGA design.

In order to evaluate the performance of the FDK implementation on ImageProX by itself, we consider the previously mentioned case of 543 projection images of 1240x960 pixels each. Truncation correction and subsequent zero-padding of the image rows (to avoid aliasing artifacts) require that vectors of length 4k are filtered. The linewise filtering of all projections takes about 5.5s (center FPGA only), whereas the following parallel back-projection into a volume of 512x512x440 voxels takes about 3.5s (eight FPGAs). For comparison, on two dual core Intel CPUs, the linewise filtering takes 21s, the back-projection 180s. Note that the back-projection performance scales linearly with the number of projection images and the number of voxels to be reconstructed.

V. Software Optimizations

Eliminating the major bottle necks of the reconstruction pipeline by the use of the ImageProX hardware, the previously inexpensive steps become the time-limiting factors. To reduce their influence to the over-all timing, further optimizations have been performed.

A. *Parallelization*

Modern high-performance workstations comprise two CPUs. For example, the latest *syngo* X-Workplace from Siemens Medical Solutions contains two Intel Core Duo Processors, each of them providing two separate arithmetic units. As a consequence, four computing threads can run in parallel.

There are several approaches towards parallelizing image processing algorithms on a network of processing nodes [7]. The following aspects have a positive influence on speed-up reducing overhead: minimum number of synchronization points, minimum data exchange between threads, and maximum load balance.

For this, we decided to partition data on the highest possible level, in our case image by image. With a fixed number of worker threads, each image received from the acquisition system is assigned to the next available thread. Assuming a sufficiently high number of projections (more than 100), with this approach all three previously mentioned aspects are addressed.

For post-processing, the volume is partitioned into stacks of identical numbers of slices. Each stack is processed by a separate thread, correcting artifacts for each particular slice independently of the others.

B. *Code Optimization*

We analyzed the source code of the correction algorithms to identify loops and particular statements consuming a remarkable computational time. For this analysis, the VTune Performance Analyzer from Intel provides means to identify computationally expensive code fragments (e.g., a function or an individual C++-code line), to analyze the reason for a time delay (e.g., cache misses), all that while executing the release version of binaries without the need for code instrumentation.

There were some surprising hints for performance optimization which we had not expected. For example, in the scatter estimation routine, VTune identified time delays caused by a large number of indirect addressing commands. Looking into the affected code parts, a look-up table for computing logarithms was implemented in order to speed up the code for a previous CPU version. Substituting the table look-up by the direct computation using the compiler built-in floating point command, the Intel Compiler was able to vectorize the inner loop, which for this routine delivered a speed-up of a factor of 3. VTune also helped to locate and eliminate other expensive commands such as type conversions, for example, and indicated which loops to split for allowing automatic vectorization by the compiler.

VI. Results

In this section we show some examples of computational times for protocols, which are relevant for clinical applications. For these measurements, we used a workstation with two Intel Core Duo Processors with 4 MByte of second-level cache per processor, running under Microsoft Windows XP Professional.

The protocols shown in Table II are typically used in today's Siemens AXIOM Artis dBA systems in combination with the *syngo* X-Workplace.

TABLE II
SUMMARY OF RECONSTRUCTION RESULTS

Clinical Application	**# images**	**Image size [pixels]**	**Reconstruction time [sec]**
High-contrast angiography	126	1240x960	14
Subtracted angiography	2 x 126	1240x960	19
Soft tissue for body imaging	419	616x480	27
Soft tissue for neuro imaging	543	1240x960	43

These runtimes are measured starting with the availability of the first projection image and ending with the completion of the post-processing. In all cases, the volume cube had a size of 512x512x440 voxels and covered the maximum available field of view. For the high-contrast angiography applications referred to in the first two lines of the table, scatter correction, truncation correction, and ring artifact correction have been turned off. For the soft tissue imaging protocols mentioned in the third and fourth line of the table, however, all correction algorithms have been applied.

VII. CONCLUSIONS

With the described efforts, the computational times for reconstructing CT-like images from C-arm X-ray projections could be reduced, remarkably. Even for worst-case protocols, after less than a minute after acquisition, the reconstruction results are available in the visualization application.

The established speed-up increases the clinical applicability of this new technology for established interventional procedures, but even opens up completely new fields of applications. For example, the progress of a complex interventional procedure potentially could now be controlled by repeated low-dose acquisitions.

Beyond solely reducing the time between acquisition and visualization, the increased computational power facilitates further algorithmic extensions for improving 3D image quality within a clinically realistic reconstruction time. We also investigate possibilities for utilizing the ImageProX platform for other computationally expensive algorithms.

REFERENCES

[1] L.A. Feldkamp, L.C. Davis, and J.W. Kress, "Practical cone-beam algorithm", *J. Opt. Soc. Am. A* 1(6), 1984

[2] M. Zellerhoff, B. Scholz, E.-P. Ruehrnschopf, and T. Brunner, "Low contrast 3D reconstruction from C-arm data", *Proceedings of SPIE, Medical Imaging 2005: Physics of Medical Imaging*, Michael J. Flynn, Editor, April 2005, pp. 646-655

[3] K. Wiesent, K. Barth, P. Durlak, T. Brunner, O. Schuetz, and W. Seissler, "Enhanced 3-D-reconstruction algorithm for C-arm systems suitable for interventional procedures", *IEEE Trans. Med. Imag.* 19(5), pp. 391-403, 2000.

[4] U. Meyer-Baese, Digital Signal Processing Using Field Programmable Gate Arrays, Springer, 2nd edition, 2004

[5] Xilinx, Inc., http://www.xilinx.com

[6] H. Scherl, M. Koerner, H. Hofmann, W. Eckert, M. Kowarschik, J. Hornegger, „Implementation of the FDK Algorithm for Cone-Beam CT on the Cell Broadband Engine Architecture", to appear in *Proc. of SPIE, Medical Imaging 2007*

[7] A.Y.H. Zomaya (Editor), "Parallel & Distributed Computing Handbook", McGraw-Hill Professional, 1995

Benno Heigl received a master's degree in computer science in 1997 and a PhD degree in computer science in 2003, both from University of Erlangen-Nuremberg. During this time his research interest mainly was focused on combining computer vision and computer graphics methods for image based rendering. He has been with with Siemens Medical Solutions, Angiography, Fluoroscopic- and Radiographic Systems (AX) division in Forchheim, Germany since 2001. Currently he is heading a team responsible for 3D application development, covering 3D-reconstruction (DynaCT) and image fusion applications.

Markus Kowarschik received a master's degree in computer science in 1998 and a PhD degree in computer science in 2004, both from University of Erlangen-Nuremberg. During this time his research focused on hardware-oriented numerical methods for the solution of partial differential equations, particularly multi-grid algorithms. He has been with Siemens Medical Solutions, Components division (CO) in Erlangen, Germany since 2004. His work at CO focuses on the development of hardware acceleration techniques for applications in medical image processing.