Journal of X-Ray Science and Technology 19 (2011) 111–126
DOI 10.3233/XST-2010-0281
IOS Press

Unified reconstruction framework for multi-modal medical imaging

Di Dong, Jie Tian*, Yakang Dai, Guorui Yan, Fei Yang and Ping Wu
Medical Image Processing Group, Institute of Automation Chinese Academy of Sciences, Beijing, China

Received 1 September 2010
Revised 23 November 2010
Accepted 12 December 2010

Abstract. Various types of advanced imaging technologies have significantly improved the quality of medical care available to patients. Corresponding medical image reconstruction algorithms, especially 3D reconstruction, play an important role in disease diagnosis and treatment assessment. However, these increasing reconstruction methods are not implemented in a unified software framework, which brings along lots of disadvantages such as breaking connection of different modalities, lack of module reuse and inconvenience to method comparison. This paper discusses reconstruction process from the viewpoint of data flow and implements a free, accelerated, extensible Unified Reconstruction Software Framework (URSF). The software framework is an abstract solution that supports multi-modal image reconstruction. The goal of this framework is to capture the common processing work flow for different modalities and different methods, make the development of reconstruction for new devices much easier, and implement a set of popular reconstruction algorithms, so that it is convenient for researchers to compare against. The overall design and certain key technologies are introduced in detail. Presented experiment examples and practical applications commendably demonstrate the validity of this framework.

Keywords: Unified Reconstruction Software Framework, work flow, CT, freehand 3D ultrasound imaging, acceleration schemes

1. Introduction

Modern medical imaging technologies, such as computed tomography (CT), ultrasound imaging (US), magnetic resonance imaging (MRI), positron emission tomography (PET), single photon emission computed tomography (SPECT) and optical tomography, provide tremendous benefits for easy disease diagnoses. Corresponding medical image reconstruction algorithms have been strongly developed and practically implemented in almost every modern imaging modality, but there are many problems which still remain unresolved or can be improved. One serious problem is that there is no unified software platform covering a wide spectrum of reconstruction methods. Lack of unified framework brings along lots of disadvantages, such as breaking relationship between different modalities, duplicate efforts during code development and inconvenience to method comparison. Similar work flow can not be shared if two methods are implemented in different platforms. During implementation of new methods, substantial efforts should be spent on trivial but necessary code parts such as data module, data input module, data output module, and algorithm interfaces, etc. Moreover, classic methods are coded again and again by different researchers just for comparison against their own methods. What is worse, imaging

*Corresponding author: Jie Tian, Medical Image Processing Group, Institute of Automation Chinese Academy of Sciences, P.O. Box 2728, Beijing, China. Tel.: +8610 82628760; Fax: +8610 62527995; E-mail: tian@ieee.org, jie.tian@ia.ac.cn.

system developers have to put much more energy to design software if there is no free and off-the-shelf reconstruction framework. Aforementioned requirements make consolidating existing reconstruction algorithms to a unified software framework necessary.

Some researchers have designed modular reconstruction softwares in their imaging modalities. Thielemans et al. have designed an open-source object-oriented library in C++ for 3D PET reconstruction [38]. The library is called Software for Tomographic Image Reconstruction (STIR). In the library CPU Multiple Instruction Multiple Data technique (MIMD) is supported. Currently, the emphasis of STIR is mainly on iterative image reconstruction in PET. Cook and Bai have implemented a free, open-source, object-oriented software package for analysis and reconstruction of diffusion MRI data [6,7]. The software package is called Camino which supports Windows, Mac OS X and Linux. Camino supports a range of standard and advanced diffusion MRI reconstruction algorithms. Jeff Fessler et al. have developed a collection of open source algorithms for image reconstruction written in Matlab language. The software is called Image Reconstruction Toolbox (IRT) [12]. The toolbox includes iterative and non-iterative algorithms for tomographic imaging (PET, SPECT, X-ray CT), methods for MRI reconstruction, iterative image restoration tools and so on. ASPIRE (A sparse precomputed iterative reconstruction library) is a collection of ANSI C programs for tomographic image reconstruction also developed by Jeff Fessler et al. Central components of ASPIRE code are about spatial resolution properties of image reconstruction methods [11]. However, the newest version of ASPIRE only runs on Linux and Mac OS X.

Although aforementioned toolkits are very powerful, they are restricted and defective. STIR is mainly for iterative PET reconstruction; Camino is only for diffusion MRI reconstruction; IRT is written in Matlab language and not efficient enough; ASPIRE is only for sparse pre-computed iterative reconstruction and not for Windows. There is still no uniform software framework for multi-modal medical image reconstruction which facilitates both usage and extension. Therefore, a free, extensible, efficient and unified framework solution needs to be investigated urgently.

This paper analyzes reconstruction process from the standpoint of work flow and implements a unified reconstruction software framework, which provides an abstract software solution for multi-modal reconstruction. In URSF, raw data from different imaging devices, data input/output (I/O) interfaces, and reconstruction algorithms are encapsulated to modules for reuse. Meanwhile, lots of acceleration techniques are also modularized, including CPU multi-thread, single instruction multiple data (SIMD) technique, graphic process unit (GPU) parallel technique [15] and out-of-core technique [44]. Modular programming makes the platform easy to use and extend. Furthermore, lots of classical reconstruction algorithms have been implemented, which make URSF an effective platform for testing and comparing different reconstruction methods. Currently, URSF has been combined with Medical Imaging ToolKit (MITK) [39,51], which is a free toolkit for medical imaging and analyzing.

In Section 2, we will first introduces inverse problem and work flow idea in reconstruction. Section 3 will give the practical implementation details of URSF. In Section 4, we present experiments, while Section 5 offers discussion of this framework. The last section is about final conclusions as well as an outlook onto the work ahead.

2. Inverse problem and reconstruction methods

Inverse problem is a task that often occurs in medical imaging where images of target must be obtained from observed data [32]. Methods that solve inverse problems are usually called reconstruction or restoration. This section mainly introduces analytical CT reconstruction and pixel-based freehand 3D ultrasound reconstruction, since these two modalities have been integrated in URSF.

Nowadays, CT is widely used in clinical diagnosis. The success of CT benefits from rapid development of reconstruction methods and great improvement of CT devices [9,20]. Current CT reconstruction methods are usually divided to analytic algorithms and iterative ones. By comparison with iterative methods, analytic algorithms are more popular in practical application. The first important analytic method was Filtering Back Projection (FBP) method, which was applied to 2D fan-beam reconstruction [5,35]. Then FBP method was extended to 3D cone-beam reconstruction with circular source locus (FDK method) [10]. However, FDK is an approximate method. When cone angle increases, artifacts in reconstruction images deteriorate. Lots of improved algorithms were proposed to solve cone angle artifacts, such as TFDK algorithm [16], shift-variant filtering algorithm [48], weighted CB-FBP algorithm [37] and so on [26]. To solve long object problem, FBP algorithm was extended to helical trajectory reconstruction [41,42]. During the development of exact reconstruction, Katsevich proposed the first exact algorithm with FBP-type for cone-beam helical scanning locus [21,23]. After that, Zou et al. proposed exact image reconstruction algorithm based on PI-lines in helical cone-beam CT [55,56]. Then exact algorithms were greatly improved in general scanning locus [22,29–31,47,52,27].

3D ultrasound (3DUS) is also widely used for diagnostics and image guidance in clinic. Although 3D ultrasound probes exist, the frequently used 2D US probes can also perform 3DUS, which is usually called freehand 3D ultrasound imaging [36]. However, a positioning sensor is demanded in freehand 3DUS for recording position data of each 2D image. Freehand ultrasound reconstruction algorithms can be classified into three categories: pixel-based algorithms (PBAs), voxel-based algorithms (VBAs) and function-based algorithms (FBAs). PBAs traverse through all pixels of 2D images and insert them into related target volume voxels. VBAs traverse through all voxels in a target volume and fill them with corresponding pixels. FBAs generate 3D volume by estimating functions of input data [33,36]. A PBA usually consists of two stages: a bin-filling stage (BFS) and a hole-filling stage (HFS). BFS traverses all pixels in 2D ultrasound images and applies each pixel to one or several voxels. After BFS, there are usually many empty voxels. HFS traverses all voxels and fills empty ones. Pixel Nearest Neighbor (PNN) method is widely used in pixel-based algorithms. In BFS, each pixel value is inserted to its nearest voxel. If there are multiple contributions to a voxel, final value of the voxel can be average [14], maximum [27] or most recent value [28]. Hole-filling stage fills empty voxels with average of nearby voxels [27], maximum of nearby voxels [19] or median of nearby nonzero voxels [19]. There are also a number of PBAs without BFS-plus-HFS step. These methods traverse each pixel in 2D image and assign pixel value to more than one voxels around the pixel position [3,25,28].

Pan et al. have described chain of data flow in CT reconstruction [32], which runs from projection views to images. Through analyzing procedure of medical image reconstruction, we find that reconstruction methods in all modalities always follow data flow model. Reconstruction runs from observed medical data and device information to images. Thus, similar work flows can be shared by different methods or even different modalities. Taking FBP-type methods in CT reconstruction as an example, these methods have two similar steps: filtering and back projection. Different methods can share these similar steps in a unified framework. For instance, FDK method with circular locus and Katsevich method with spiral locus have nearly the same back projection processes. They both need to back project filtered projection views to 3D volume. By using modular programming, FDK method and Katsevich method can share one back projection module together. Meanwhile, FBP-type methods are also widely used in other tomography imaging such as MRI, PET and SPECT. FBP-type methods in these modalities can also share similar work flows. Our software framework is greatly inspired by the work flow idea. One of the aims of URSF is to modularize similar work flows and make them easy to use. Moreover, back projection steps have been implemented in URSF. Therefore, all methods that have similar back projection processes can share these existed modules.

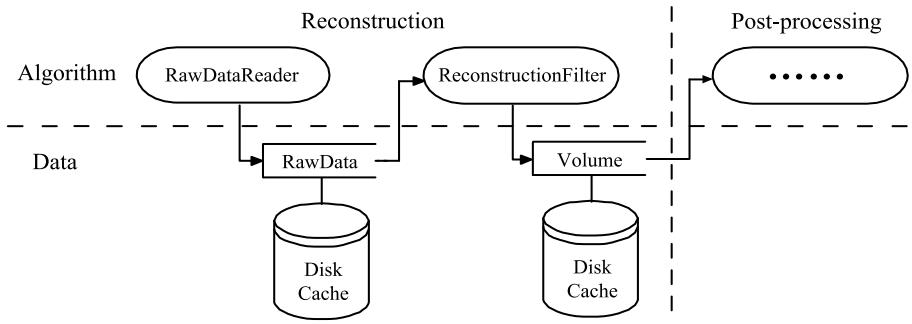


Fig. 1. Computational framework of URSF. The left part represents reconstruction process in URSF. The right part means post-processing on reconstruction result.

3. Software framework implementation

This section is divided into four parts to introduce basic structure of URSF in detail. Subsection 3.1 introduces overall design of this framework. Subsection 3.2 and 3.3 are about CT sub-framework and ultrasound sub-framework, respectively. The last subsection is mainly about acceleration techniques and out-of-core data management.

3.1. Software framework overview

'A software framework is a set of classes embodying an abstract design for solutions to a family of related problems' [4]. URSF is actually an abstract encapsulation of necessary reconstruction modules, which include raw data module, I/O interface module, algorithm module, etc.

Figure 1 shows computational framework of URSF. Medical raw data from imaging devices and necessary hardware information are abstracted to a RawData class; medical regular data is abstracted to a Volume class; reconstruction algorithms are abstracted to a ReconstructionFilter class that receives RawData and generates Volume. The pipeline from RawData to Volume forms a consistent reconstruction data flow. After reconstruction, lots of post-processing algorithms, such as segmentation, registration and visualization, can be applied to reconstruction results. Moreover, both RawData and Volume are connected to disk cache and manage data exchange between internal and external storage, which provide function of out-of-core data storage to this platform.

Figure 2 illustrates main modules in this framework. The star '*' in the figure means that preceding variable is a pointer. Taking '#m_InData:CTProjectionData*' in Fig. 2(b) as an example, it means that CTReconstructionFilter has a member variable m_InData. m_InData is a pointer to CTProjectionData. As shown in Fig. 2(a), there are two concrete subclasses of Volume, ICVolume and OoCVolume. OoCVolume is designed for containing out-of-core data. It implements necessary management functions of out-of-core data, including storage buffer management, data exchange between main memory and hard disk, etc. This class works very well when data are larger than memory capability. While ICVolume works faster when all data can be loaded to main memory at one time. CTProjectionData and TrackedBscanData, demonstrate CT projection data and 2D ultrasound data, respectively. As shown in Fig. 2(b) ReconstructionFilter is top-level abstraction of reconstruction algorithms. CT and freehand 3D ultrasound reconstruction algorithms are abstracted as CTReconstructionFilter and 3DUSReconstructionFilter. Figure 2(c-d) show input and output modules in URSF. Reader is an abstract input interface responsible for reading data from disk, while Writer is responsible for writing data to disk files. The I/O modules support various file formats, i.e. BMP, JPEG, TIFF, DICOM, raw format, etc.

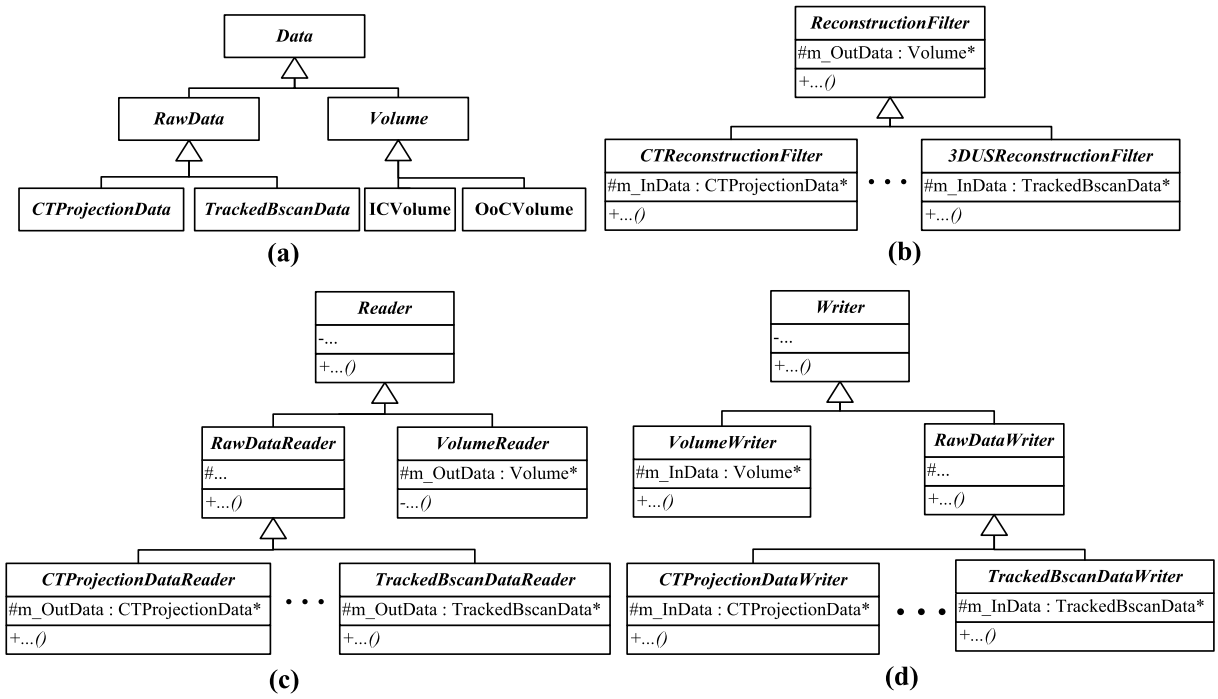


Fig. 2. Main modules of URSF. (a) Data modules. (b) Reconstruction algorithm modules. (c) Input modules. (d) Output modules.

3.2. CT reconstruction sub-framework

X-ray computed tomography is the first imaging modality that allows non-destructive interior imaging of an object. CT technology also greatly promoted development of other tomographic modalities [32, 43].

A powerful CT reconstruction sub-framework is implemented, which is based on existed modules in URSF. Taking CTProjectionData as an example, it is an abstract subclass of RawData demonstrating CT projection data. CT projection data can be classified to different groups using different criteria. According to data acquisition mode, it can be sorted into three categories: parallel-beam projection data, fan-beam projection data and cone-beam projection data. As shown in Fig. 3, CT data are divided into two groups: projection data from 2D object (2DBeamPD) and cone-beam projection data (ConeBeamPD). Suffix "PD" means projection data. 2DBeamPD abstracts parallel-beam projection data and fan-beam projection data. 2DBeamRealTimePD and ConeBeamRealTimePD are two special classes which allow sequential access of projection data. They are mainly applied to real time reconstruction, which means reconstruction is completed as soon as scan finishes.

Figure 4 shows inheritance hierarchy of CT reconstruction classes. According to projection data acquisition mode, CT reconstruction algorithms are sorted into three classes: parallel-beam reconstruction algorithms with circular scanning locus (ParallelBeamFromCircle), fan-beam reconstruction algorithms with circular scanning locus (FanBeamFromCircle) and cone-beam reconstruction algorithms with helical scanning locus (ConeBeamFromHelix). The circular locus method is classified to helical locus class because circle can be seen as a special helix with a pitch of zero. Back projection step is the most time-consuming step of FBP-type algorithms. As shown in Fig. 4, back projection step is encapsulated

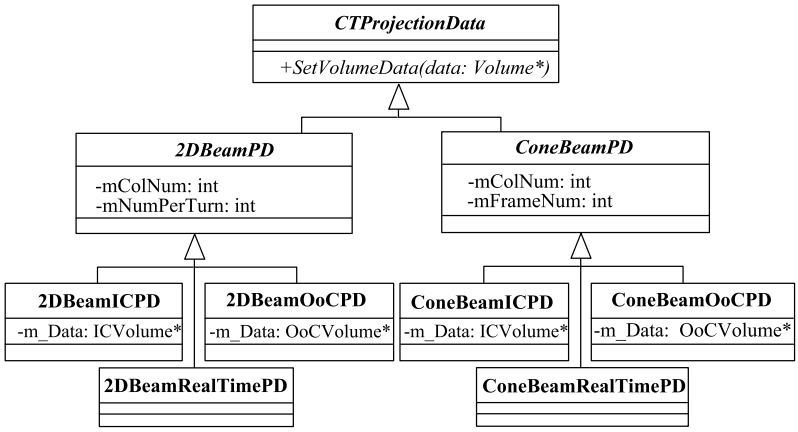


Fig. 3. Inheritance hierarchy of CT data classes. CTProjectionData is divided into two classes: 2DBeamPD and ConeBeamPD.

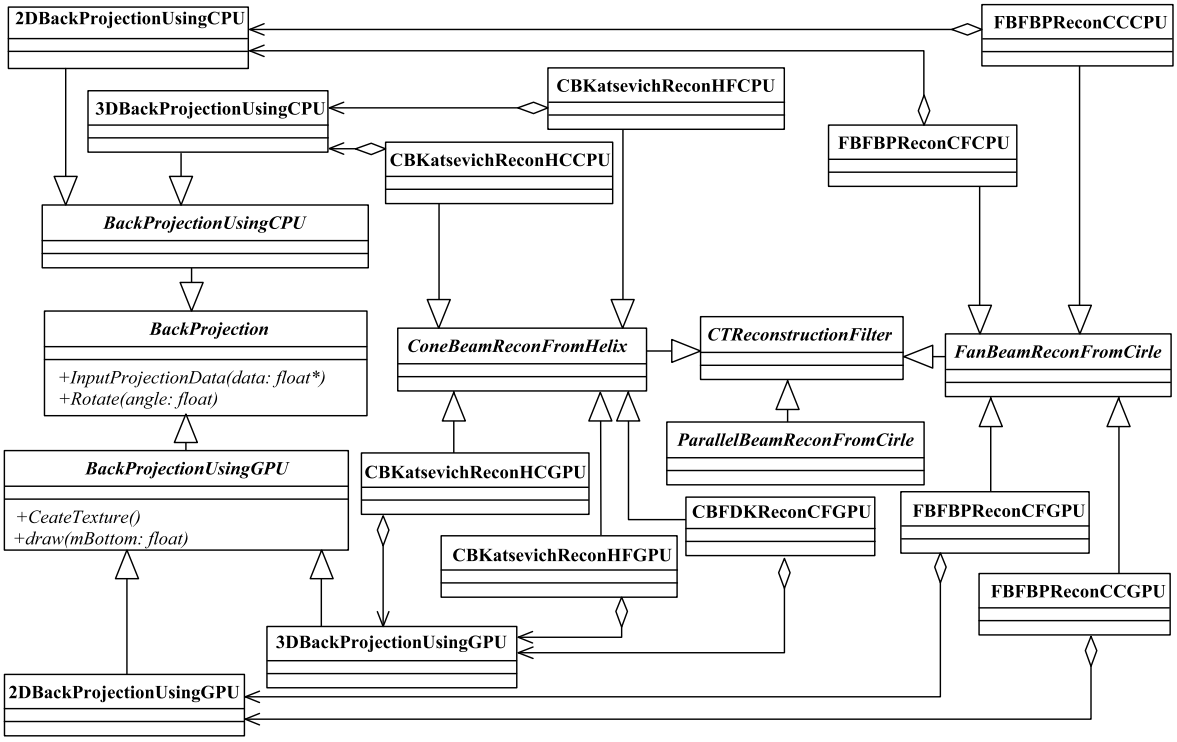


Fig. 4. Inheritance hierarchy of CT reconstruction algorithm classes.

to BackProjection class. BackProjection accepts filtered projection views and related geometry information, then backprojects them to each slice in 3D volume. The back projection step has already been accelerated by GPU parallel scheme (BackProjectionUsingGPU).

Furthermore, about a dozen algorithms have already been implemented in CT sub-framework. Names of these algorithms are regulated as Table 1. Taking CBKatsevichReconHCGPU for example, the prefix “CB” means cone-beam projection, “Katsevich” stands for Katsevich reconstruction method [23],

Table 1
Regulation to name concrete CT reconstruction algorithms

Projection type	Algorithm	Scanning locus	Detector type	Acceleration Hardware
FanBeam (FB)	FBP	Cricle (C)	Curve (C)	–
ConeBeam (CB)	FDK	Helix (H)	Flat (F)	GPU
ParallelBeam (PB)	TFDK	...	...	CPU
	Katsevich			CELL
	...			...

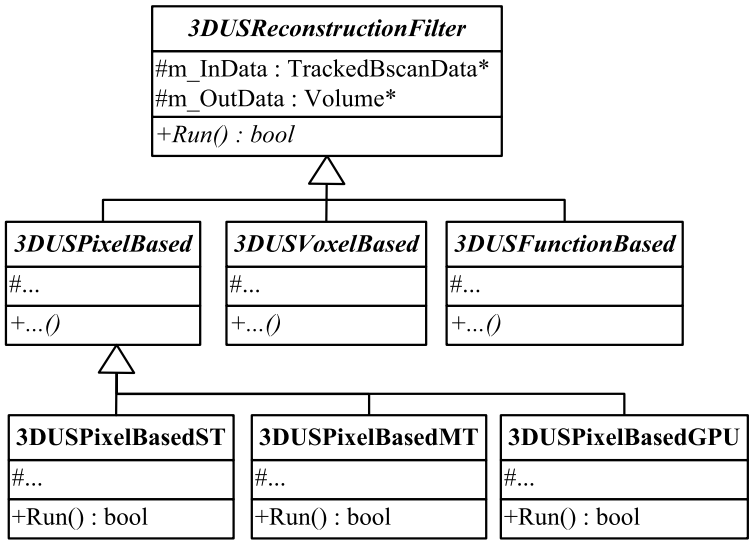


Fig. 5. Inheritance hierarchy of freehand 3D ultrasound reconstruction classes.

"HC" represents helical scanning trajectory and curved detector, and the suffix "GPU" means this algorithm has been accelerated by GPU technique. So CBKatsevichReconHCGPU represents an exact Katsevich reconstruction algorithm based on helical cone-beam data from a curved detector [46]. In Table 1, "FDK" means Feldkamp-Davis-Kress algorithm which is the most widely used cone-beam reconstruction algorithm with circular locus [10], "TFDK" represents tent Feldkamp approach which is a modified FDK method proposed by Grass et al. [16].

3.3. Freehand 3D ultrasound reconstruction sub-framework

A freehand 3D ultrasound reconstruction sub-framework has also been built up. Similar with CT sub-framework, ultrasound sub-framework consolidates all essential modules such as ultrasound data module, ultrasound data I/O interfaces and reconstruction algorithm module. Ultrasound raw data is abstracted as TrackedBscanData which represents a serial of 2D ultrasound images and corresponding positioning data. Ultrasound data classes manage in-core and out-of-core storage as well as CT data classes.

Figure 5 shows main inheritance hierarchy of ultrasound reconstruction classes. 3DUSPixelBased represents pixel-based algorithms. 3DUSVoxelBased abstracts voxel-based algorithms, while 3DUSFunctionBased means function-based algorithms. Three concrete pixel-based methods (3DUSPixelBasedST, 3DUSPixelBasedMT and 3DUSPixelBasedGPU) have been implemented. 3DUSPixelBasedST represents classical PNN ultrasound method with single CPU thread. In bin-filling stage of 3DUSPixel-

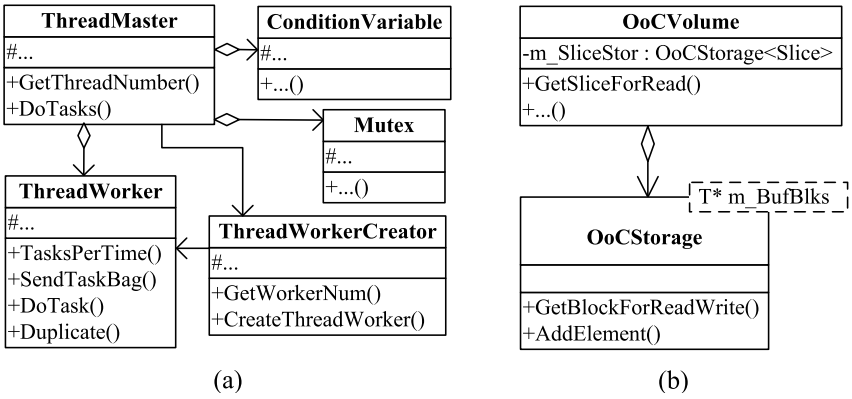


Fig. 6. Advanced schemes in URSF. (a) Parallel acceleration framework. (b) Out-of-core Storage.

BasedST, each pixel value is inserted into its nearest voxel. If there are multiple contributions to a voxel, final value of the voxel is average. The hole-filling stage fills empty voxels with average of nearby voxels. 3DUSPixelBasedMT accelerates PNN method with CPU multi-thread technique, while 3DUS-PixelBasedGPU applies GPU parallel scheme on PNN method. Next section will show that multi-thread techniques especially GPU acceleration scheme can dramatically speed up reconstruction.

3.4. Some advanced schemes

Amount of data from imaging device continues to grow due to increasing precision. Taking micro-CT system as an example, size of projection data is usually more than 2 Gbyte and reconstruction volume even larger [2,54]. It is a great challenge to perform fast reconstruction on such huge amount of data. Therefore, lots of acceleration schemes and data management techniques are implemented in this framework. Figure 6 shows two main schemes, multi-thread technique and out-of-core data management. GPU scheme has been applied to speed up time-consuming methods as well.

3.4.1. Parallel acceleration framework

We have implemented a parallel acceleration architecture in URSF. Master/Worker design pattern is quite suited to solve such parallel problems: dynamic load balance; frequent switch between parallel part and serial part; using different kinds of processors to do the work [1,13,24]. Hence, Master/Worker design pattern has been applied to organize this parallel framework. As shown in Fig. 6(a), a ThreadMaster sets up a pool of ThreadWorkers and a bag of reconstruction tasks. ThreadWorkers execute concurrently, with each thread repeatedly taking a task away from the task bag and processing it, until the bag is empty. Actually, ThreadMaster first creates ThreadWorkers by using a ThreadWorkerCreator provided by user, then creates corresponding number of threads to drive these workers. Task removing is done by threads, while ThreadWorker simply do the tasks it receives. ConditionVariable and Mutex are used when there is any communication between ThreadWorkers. This parallel framework is suitable for speeding up time-consuming reconstruction algorithms. Taking ultrasound PNN mehtod as an example, both BFS and HFS are ideal for parallel. In 3DUSPixelBasedMT, ThreadMaster divides BFS by images and sends them to its ThreadWorkers. Then each ThreadWorker processes in a loop one image after another. When all images are inserted to 3D volume, BFS is finished. HFS is accelerated similarly.

3.4.2. Out-of-core data storage

With increasing size of medical data, mass data problem becomes a great challenge to computing hardware, especially to main memory. When data is larger than memory size, whole data can not be loaded to main memory at one time, which is usually called mass data problem. The most common solution of this trouble is to apply auxiliary memory (out-of-core data management). However, without an encapsulation of out-of-core data management, researchers have to additionally design special data storage functions. URSF has implemented out-of-core data modules which can be shared by all methods. Figure 6(b) shows out-of-core storage management. A template class OoCStorage is implemented to achieve out-of-core storage management, which covers storage buffer management and data exchange between main memory and hard disk. OoCVolume contains a variable of OoCStorage to manage out-of-core data, which makes a flexible structure. With such structure OoCVolume becomes basic out-of-core data class. Other classes can gain out-of-core function by containing a variable of OoCVolume.

3.4.3. GPU scheme and SIMD technique

Current parallel framework only supports CPU thread, doesn't support GPU thread yet. Although not integrated in acceleration framework, GPU schemes are widely applied in URSF. For instance, back projection step of CT reconstruction has been accelerated by GPU parallelization in BackProjectionUsingGPU class. The mapping between a projection view and a slice of 3D volume is a projection transform [40], which is similar to projective texture mapping in computer graphics [34]. Therefore, BackProjectionUsingGPU is implemented by using texture mapping on GPU [45]. With this back projection class, developers even new to GPU programming can implement efficient FBP-type methods with great ease. Meanwhile, CPU SIMD technique has also been applied to back projection. SIMD technique is the ability to perform same operation on multiple data simultaneously. Better performance can be achieved using this data level parallelism, since less cycles are used with the same amount of data [49]. In FBFBPReconCFCPU and CBFDKReconCFCPU, instructions in back projection such as interpolation, weighting, and accumulation are all implemented with SIMD. What's more, GPU scheme has also been applied to ultrasound methods. In 3DUSPixelBasedGPU both BFS and HFS are accelerated by GPU threads. Due to powerful parallel computing capability of GPU, GPU based methods are much faster than normal CPU methods.

4. Experiment examples

We have testified this framework by performing some experiments of CT and 3DUS. The following experiments show URSF's great power of comparing different methods. These experiments are performed on a Windows-PC, with an Intel Core2 2.66 GHz CPU (bus speed is 266 MHz), a 2GByte DDR2 memory, and a NVIDIA GTX 275 GPU card [18].

4.1. Comparison of different circular fan-beam CT algorithms

Figure 7 shows a comparison of different circular fan-beam 2D CT reconstruction. There are three methods for equal-spaced detector in URSF: FBFBPReconCFGPU (FBP method accelerated by GPU), FBFBPReconCFCPU (FBP method accelerated by SIMD technique), FBFBPReconCF (FBP method based on single CPU thread). The performances of these methods are tested on a 2D Shepp-Logan phantom. Sinogram is calculated from 2D Shepp-Logan phantom analytically with parameters: maximum value of phantom pixel (2.000), minimum value of phantom pixel (0.000), detector element number

Table 2
Performance comparison of different circular fan-beam 2D CT reconstruction methods

	FBFBPReconCFGPU	FBFBPReconCFCPU	FBFBPReconCF
time	0.624s	1.326s	4.960s
MSE	0.007	0.008	0.007

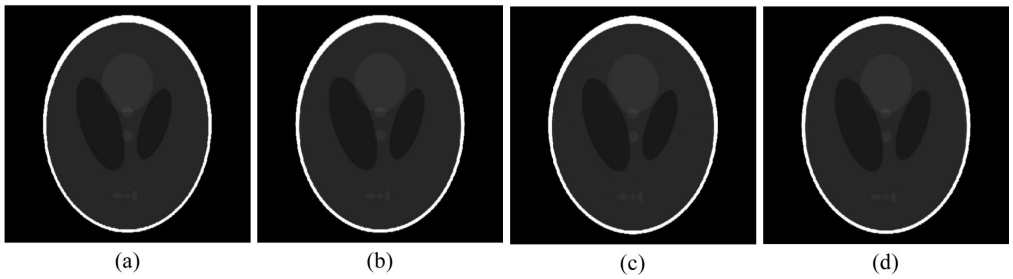


Fig. 7. CT reconstruction results on a 2D Shepp-Logan phantom. (a) Original 2D phantom. (b) Image reconstructed by FBFBPReconCFGPU. (c) Image reconstructed by FBFBPReconCFCPU. (d) Image reconstructed by FBFBPReconCF.

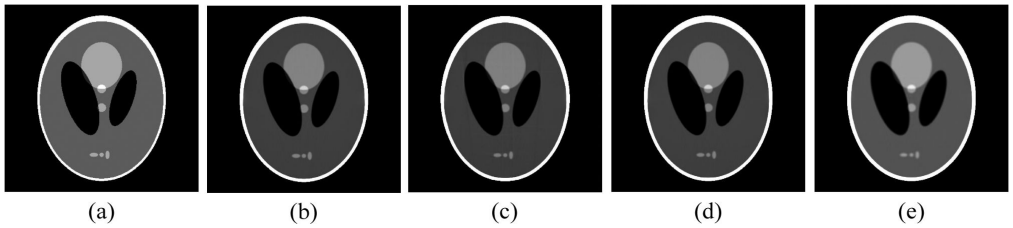


Fig. 8. CT reconstruction results on a 3D Shepp-Logan phantom. (a) A slice of original phantom. (b) The same slice reconstructed by CBFDKReconCFGPU. (c) The same slice reconstructed by CBFDKReconCFCPU. (d) The same slice reconstructed by CBFDKReconCF. (e) The same slice reconstructed by CBTFDKReconCF.

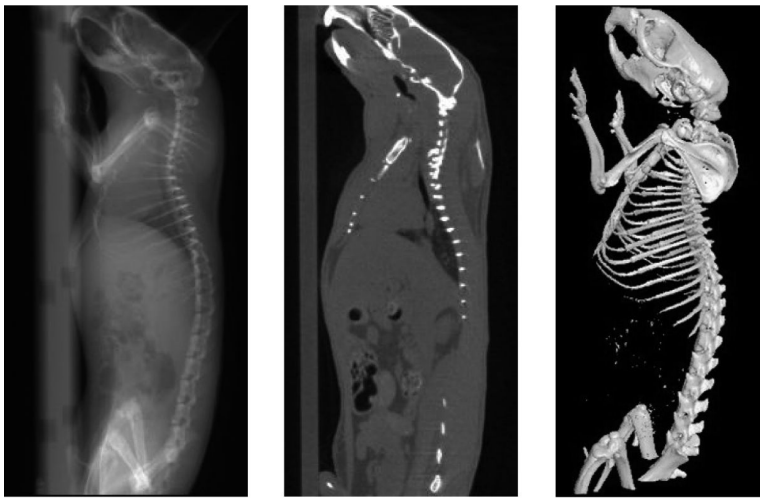
(672), number of projection angle (720) and data type (32-bit floating point). All these methods are applied to reconstruct 512×512 image from sinogram. Fig. 7(a) shows original 2D phantom. Fig. 7(b-d) show reconstruction results by the three methods. Table 2 shows reconstruction time (time of reading sinogram from disk and storing result to disk are not included) and Mean Squared Errors (MSEs) between original phantom and three results. We can see that GPU method is faster than CPU methods. Moreover, errors between reconstruction images and original phantom are very small.

4.2. Comparison of different circular cone-beam CT algorithms

Figure 8 shows a comparison of different circular cone-beam 3D CT reconstruction. There are four approximate reconstruction methods for flat-panel detector in URSF: CBFDKReconCFGPU (FDK method accelerated by GPU), CBFDKReconCFCPU (FDK method accelerated by CPU SIMD technique), CBFDKReconCF (FDK method based on single CPU thread), CBTFDKReconCF (TFDK method based on single CPU thread). The performances of these methods are tested on a 3D Shepp-Logan phantom. Amount of projection data is about 400 Mbyte with 360 views. All these methods are applied to reconstruct 512 cubed volume. Figure 8(a) shows 194th slice of original 3D phantom. Figure 8(b-e) show the same slices reconstructed by above-mentioned four methods, which seem nearly the same with original slice. Table 3 shows reconstruction time and MSEs between results and original 3D phantom. By

Table 3
Performance comparison of different circular cone-beam 3D CT reconstruction methods

	CBFDKReconCFGPU	CBFDKReconCFCPU	CBFDKReconCF	CBTFDKReconCF
time	6.226s	824.625s	1137.203s	1319.205s
MSE	0.006	0.009	0.008	0.012



(a) (b) (c)

Fig. 9. A CT reconstruction experiment on a mouse with GPU accelerated FDK method. (a) One project view of the mouse. (b) One slice of the reconstructed volume. (c) Bone structure by surface rendering.

taking advantage of a great many GPU parallel cores, CBFDKReconCFGPU behaves more powerfully than other three methods. Although there are some errors between reconstruction volumes and original 3D phantom, results are acceptable.

4.3. CT reconstruction on mass data

Figure 9 shows reconstruction result of a anesthetized mouse. The mouse is scanned by our own Micro-CT system [54] with a circular scanning locus. The number of projection views is 500 with each projection size of 2240×2344. Total size of projection data is about 5Gbyte and 3D volume size is $512 \times 512 \times 512$. GPU accelerated FDK method is applied to construct the 3D volume. The total time is only 182.680 seconds which includes reading data from disk and reconstructing 3D volume. In this experiment out-of-core technique is used because data size is larger than size of main memory. In Fig. 9, (a) shows one projection view; (b) shows one slice of reconstructed 3D volume; and (c) shows bone structure of the mouse. The bone is extracted and shown by 3-Dimensional Medical Image Processing and Analyzing system (3DMed) [39,50].

4.4. Freehand 3D ultrasound reconstruction on a foetus phantom

To evaluate ultrasound methods, we acquired 3000 B-scan images from a plastic phantom (hollow) with our ultrasound system [8]. Each 2D ultrasound image size is $552 \times 274 \times 8$bits. Then 3D volumes ($374 \times 281 \times 274$) are obtained from three ultrasound reconstruction methods. Figure 10 shows the plastic phantom and results by 3DUSPixelBasedGPU, 3DUSPixelBasedMT and 3DUSPixelBasedST.

Table 4
Performance comparison of different ultrasound methods

	3DPixelBasedGPU	3DPixelBasedMT	3DPixelBasedST
time	6.226s	824.625s	1137.203s
MSE	0.069	0.136	0.000

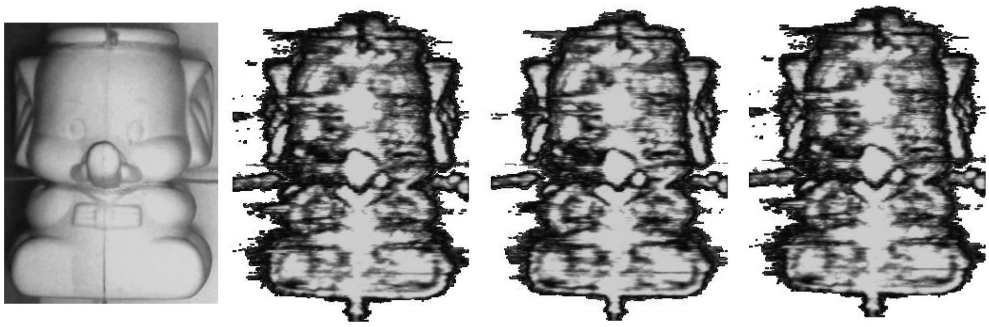


Fig. 10. Freehand ultrasound reconstruction experiment on a plastic phantom. (a) The hollow plastic phantom. (b) Volume reconstructed by 3DUSPixelBasedGPU. (c) Volume reconstructed by 3DUSPixelBasedMT. (d) Volume reconstructed by 3DUSPixelBasedST.

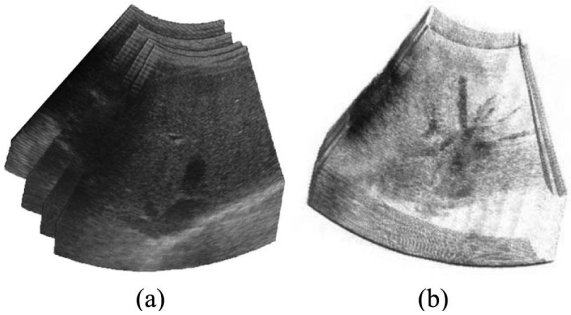


Fig. 11. Freehand ultrasound reconstruction experiment on *in vivo* hepatic data set. (a) Some slices of the hepatic data set. (b) Volume rendering result of reconstructed 3D volume.

Table 4 shows reconstruction time of each method and MSEs between 3DPixelBasedST and other two methods. Result of 3DPixelBasedST is taken as a standard to testify other two accelerated methods. 3DPixelBasedGPU performs 26.4 times faster than 3DUSPixelBasedST. Meanwhile, the errors between 3DUSPixelBasedST and accelerated methods are very small, and there is no obvious visual difference in results.

4.5. Freehand 3D ultrasound reconstruction on in vivo data set

Figure 11 shows an ultrasound reconstruction experiment on a hepatic data set. The dataset is 135 B-scan images from an *in vivo* human liver (original data is from the Medical Imaging Group, University of Cambridge [17]). Figure 11(a) shows some slices of the dataset. 3D volume is reconstructed by 3DUSPixelBasedGPU. Figure 11(b) shows volume rendering result of reconstruction volume.

Table 5
Comparison of URSF with other software framework

Framework / Criteria	URSF	STIR	Camino	IRT	ASPIRE
Concerned field	Nearly all medical imaging modalities	Interactive PET reconstruction	Diffusion-MRI Reconstruction	Tomographic reconstruction, MR imaging	Tomographic sparse iterative reconstruction
Operating system	Windows	Windows, Linux, AIX, Solaris	Windows, Linux, Mac OS X,	Windows	Linux, Mac OS X
Programming language	C/C++	C++	Java	Matlab	C
Free or not	Free	Free	Free	Free	Free
Source code available or not	Open interfaces	Open source	Open source	Open source	Open interfaces
Acceleration scheme	Multi-thread, SIMD, GPU	MIMD	No	No	No
Mass data scheme	Out-of-Core technique	No	No	No	No
Real-time capability	support real-time processing	No	No	No	No
Post-processing support	segmentation, registration, visualization, etc.	No	No	No	No
File format	BMP, JPEG, TIFF, DICOM, raw format, etc.	GE Advance sonogram format, ECAT 6/7 format	Images in Analyze format or in raw format	JPEG, raw format and other formats	fld format of AVS (Application Visualization System)
Developer support	Email, webpage, internet forum	Email, webpage	Email, webpage	Email, webpage	Email, webpage
Update frequency	Three times a year	Nearly once a year	More than three times a year	About six times a year	Sporadically
Documentation and manual	User manual, API and software architecture documents, internet forum pages, examples	User manual, Source code and software documents	User manual, Source code and software documents	Source code and remarks	End-user manual

5. Discussion

From the standpoint of data flow, reconstruction processes are similar in different modalities. Therefore, new things in one modality can promote innovation in other modalities. With this work flow idea and modular structure, URSF provides a powerful tool for both scientists and engineers. The flexible architecture reduces efforts and time for further development. Meanwhile, new reconstruction methods and modalities can be integrated to this framework with great ease. The above-mentioned CT and ultrasound sub-frameworks are very good examples for integration of new modalities. On the other hand, the platform is user-friendly. It provides standard input and output interfaces, which can exchange data with conventional image formats, i.e. BMP, JPEG, TIFF, DICOM, raw format, etc. Moreover, this framework offers a complete solution for some key issues involved with reconstruction such as mass data problem and time-consuming problem. Out-of-core data management has solved mass data problem effectively, and moreover, acceleration framework and GPU techniques have solved time-consuming problem. What's more, URSF is an effective platform for method comparison. Researchers can compare their own methods against existed algorithms. Nowadays, URSF has been applied to lots of engineering projects, such as fossil scanning software system using industrial CT, our own micro-CT system [54],

lossless defect detection using industrial CT, surgery programming and navigation for ultrasound guided tumor ablation [53], etc.

Bohn et al. have proposed a process and criteria for evaluation of software frameworks in the domain of computer assisted surgery [4]. 63 evaluation criteria have been defined to evaluate software frameworks in that paper. We have chosen some criteria to provide an objective evaluation of our reconstruction framework and other frameworks. Table 5 shows comparison of URSF with other four frameworks under 13 criteria. We can see that URSF has lots of advantages not only on many concerned fields, but also on well designed architecture and quality technical supports.

6. Conclusions and future perspective

This paper discusses inverse problems in medical imaging and analyzes reconstruction from the viewpoint of work flow. The goal of this project is to develop a software framework which covers reconstruction methods for multi-modal medical imaging. The experimental results above demonstrate that this framework is well suited to implement and accelerate computationally intensive reconstruction. Moreover, two sub-frameworks and lots of advanced schemes are presented in detail, which will be meaningful for other engineers.

Currently, URSF has combined with MITK 2.3.0 version, which can be downloaded free of charge at www.mitk.net. With URSF, MITK becomes a general algorithm platform for medical imaging and analyzing. Nowadays, MITK has both Windows and Linux versions, but URSF only runs under Windows. Further extension to other operating systems will be a meaningful work. In addition, integration of more modalities and more practical algorithms to URSF will be a long-term task.

Acknowledgements

This paper is supported by the National Basic Research Program of China (973 Program) under Grant No. 2011CB707700, the Knowledge Innovation Project of the Chinese Academy of Sciences under Grant No.KGCX2-YW-907, the National Natural Science Foundation of China under Grant No.81027002, 81071205, 30970778, the Science and Technology Key Project of Beijing Municipal Education Commission under Grant No. KZ200910005005, the Fellowship for Young International Scientists of the Chinese Academy of Sciences under Grant No. 2010Y2GA03.

References

[1] A. Alexandrescu, Modern C++ design: generic programming and design patterns applied. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.
[2] C.T. Badea, M. Drangova, D.W. Holdsworth and G.A. Johnson, In vivo small-animal imaging using micro-CT and digital subtraction angiography, *Physics in Medicine and Biology* **53**(19) (2008), R319.
[3] C.D. Barry, C.P. Allott, N.W. John, P.M. Mellor, P.A. Arundel, D.S. Thomson and J.C. Waterton, Three dimensional freehand ultrasound: image reconstruction and volume analysis, *Ultrasound in Medicine and Biology* **23**(8) (1997), 1209–1224.
[4] S. Bohn, W. Korb and O. Burgert, A process and criteria for the evaluation of software frameworks in the domain of computer assisted surgery, *Medical and Biological Engineering and Computing* **46**(12) (2008), 1209–1217.
[5] R.N. Bracewell and A.C. Riddle, Inversion of Fan-Beam Scans in Radio Astronomy, *Astrophysical Journal* **150** (Nov. 1967), 427.
[6] Camino. Available: http://www.cs.ucl.ac.uk/research/medic/camino/.

[7] P.A. Cook, Y. Bai, S. Nedjati-Gilani, K.K. Seunarine, M.G. Hall, G.J. Parker and D.C. Alexander, Camino: Open-source diffusion-mri reconstruction and processing. In 14th Scientific Meeting of the International Society for Magnetic Resonance in Medicine, Seattle, WA, USA, page 2759, May 2006.
[8] Y. Dai, J. Tian, D. Dong, G. Yan and H. Zheng, Real-Time Visualized Freehand 3D Ultrasound Reconstruction Based on GPU, *IEEE Transactions on Information Technology in Biomedicine* **14**(6) (2010), 1338–1345.
[9] M. Defrise and G.T. Gullberg, Image reconstruction, *Physics in Medicine and Biology* **51**(13) (2006), R139.
[10] L. Feldkamp, L. Davis and J. Kress, Practical cone-beam algorithm, *Journal of the Optical Society of America A* **1** (1984), 612–619.
[11] J.A. Fessler, ASPIRE. Available: http://www.eecs.umich.edu/
[12] J.A. Fessler, Image Reconstruction Toolbox (IRT). Available: http://www.eecs.umich.edu/
[13] E. Gamma, R. Helm, R. Johnson and J. Vlissides, Design Patterns, Elements of Reusable Object-Oriented Software. Pearson Education, 1994.
[14] D.G. Gobbi and T.M. Peters, Interactive intra-operative 3D ultrasound reconstruction and visualization. In Proceedings of Medical Image Computing and Computer Assisted Intervention (MICCAI), Tokyo, Japan: Springer, volume 2489, 2002, pp. 156–163.
[15] N. Goodnight, R. Wang and G. Humphreys, Computation on programmable graphics hardware, *IEEE Computer Graphics and Applications* **25**(5) (2005), 12–15.
[16] M. Grass, T. Kohler and R. Proksa, 3D cone-beam CT reconstruction for circular trajectories, *Physics in Medicine and Biology* **45**(2) (2000), 329.
[17] Hepatic. Medical imaging group, university of cambridge. Available: http://mi.eng.cam.ac.uk/
[18] GeForce GTX275. NVIDIA. Available: http://www.nvidia.com/.
[19] R.S. José-Estépar, M. Martín-Fernández, P.P. Caballero-Martínez, C. Alberola-López and J. Ruiz-Alzola, A theoretical framework to three-dimensional ultrasound reconstruction from irregularly sampled data, *Ultrasound in Medicine and Biology* **29**(2) (2003), 255–269.
[20] W.A. Kalender, X-ray computed tomography, *Physics in Medicine and Biology* **51**(13) (2006), R29.
[21] A. Katsevich, Theoretically exact filtered backprojection-type inversion algorithm for spiral CT, *Siam Journal on Applied Mathematics* **62**(6) (2002), 2012–2026.
[22] A. Katsevich, A general scheme for constructing inversion algorithms for cone beam CT, *International Journal of Mathematics and Mathematical Sciences* **2003**(21) (2003), 1305–1321.
[23] A. Katsevich, An improved exact filtered backprojection algorithm for spiral computed tomography, *Advances in Applied Mathematics* **32**(4) (2004), 681–697.
[24] T. Mattson, B. Sanders and B. Massingill, Patterns for parallel programming. Addison-Wesley Professional, 2004.
[25] S. Meairs, J. Beyer and M. Hennerici, Reconstruction and visualization of irregularly sampled three- and fourdimensional ultrasound data for cerebrovascular applications, *Ultrasound in medicine and biology* **26**(2) (2000), 263–272.
[26] S. Mori, M. Endo, S. Komatsu, S. Kandatsu, T. Yashiro and M. Baba, A combination-weighted feldkamp-based reconstruction algorithm for cone-beam CT, *Physics in Medicine and Biology* **51**(16) (2006), 3953.
[27] T.R. Nelson and D.H. Pretorius, Interactive acquisition, analysis and visualization of sonographic volume data, *International Journal of Imaging Systems and Technology* **8** (1997), 26–37.
[28] R. Ohbuchi, D. Chen and H. Fuchs, Incremental volume reconstruction and rendering for 3D ultrasound imaging. In In SPIE Proceedings on Visualization in Biomedical Computing, volume 1808, 1992, pp. 312–323.
[29] J.D. Pack and F. Noo, Cone-beam reconstruction using 1D filtering along the projection of M-lines, *Inverse Problems* **21**(3) (2005), 1105–1120.
[30] J.D. Pack, F. Noo and R. Clackdoyle, Cone-beam reconstruction using the backprojection of locally filtered projections, *IEEE Transactions on Medical Imaging* **24**(1) (2005), 70–85.
[31] V.P. Palamodov, Reconstruction from ray integrals with sources on a curve, *Inverse Problems* **20**(1) (2004), 239–242.
[32] X. Pan, E.Y. Sidky and M. Vannier, Why do commercial CT scanners still employ traditional, filtered backprojection for image reconstruction? *Inverse Problems* **25**(12) (2009), 123009.
[33] R. Rohling, A. Gee and L. Berman, A comparison of freehand three-dimensional ultrasound reconstruction techniques, *Medical Image Analysis* **3**(4) (1999), 339–359.
[34] M. Segal, C. Korobkin, R. van Widenfelt, J. Foran and P. Haeberli, Fast shadows and lighting effects using texture mapping. In Proceedings of the 19th annual conference on Computer graphics and interactive techniques, SIGGRAPH'92, New York, NY, USA, 1992. ACM, pp. 249–252.
[35] L.A. Shepp and B.F. Logan, The fourier reconstruction of a head section, *IEEE Transactions on Nuclear Science* **21** (1974), 21–43.
[36] O.V. Solberg, F. Lindseth, H. Torp, R.E. Blake and T.A. Nagelhus Hernes, Freehand 3D ultrasound reconstruction algorithms-A Review, *Ultrasound in Medicine and Biology* **33**(7) (2007), 991–1009.

[37] X. Tang, J. Hsieh, A. Hagiwara, R.A. Nilsen, J. Thibault and E. Drapkin, A three-dimensional weighted cone beam filtered backprojection (CB-FBP) algorithm for image reconstruction in volumetric CT under a circular source trajectory, *Physics in Medicine and Biology* **50** (Aug. 2005), 3889–3905.
[38] K. Thielemans, S. Mustafovic and C. Tsoumpas, Stir: Software for tomographic image reconstruction release 2. In IEEE Nuclear Science Symposium Conference Record, volume 4, 2007, pp. 2174–2176.
[39] J. Tian, J. Xue, Y. Dai, J. Chen and J. Zheng, A novel software platform for medical image processing and analyzing, *IEEE Transactions on Information Technology in Biomedicine* **12**(6) (2008), 800–812.
[40] H. Turbell, Cone-beam reconstruction using filtered backprojection. Technical report, 2001.
[41] G. Wang, T.H. Lin and P.C. Cheng, Scanning cone-beam reconstruction algorithms for x-ray microtomography, In Proc. SPIE, volume 1556, July 1991, pp. 99–122.
[42] G. Wang, T.H. Lin, P.C. Cheng and D.M. Shinozaki, A general cone-beam reconstruction algorithm, *IEEE Transactions on Medical Imaging* **12**(3) (1993), 486–496.
[43] G. Wang, H. Yu and B.D. Man, An outlook on x-ray CT research and development, *Medical Physics* **35**(3) (2008), 1051–1064.
[44] J. Xue, J. Tian, J. Chen and Y. Dai, An efficient out-of-core volume ray casting method for the visualization of large medical data sets. In Proccedings of SPIE Symposium on Medical Imaging 2007, San Diego, USA, 2007, p. 650920.
[45] G. Yan, J. Tian, S. Zhu, Y. Dai and C. Qin, Fast cone-beam CT image reconstruction using GPU hardware, *Journal of X-Ray Science and Technology* **16**(4) (2008), 225–234.
[46] G. Yan, J. Tian, S. Zhu, C. Qin, Y. Dai, F. Yang, D. Dong and P. Wu, Fast katsevich algorithm based on GPU for helical cone-beam computed tomography, *IEEE Transactions on Information Technology in Biomedicine* **14**(4) (2010), 1053–1061.
[47] Y.B. Ye, S.Y. Zhao, H.Y. Yu and G. Wang, A general exact reconstruction for cone-beam CT via backprojection-filtration, *IEEE Transactions on Medical Imaging* **24**(9) (2005), 1190–1198.
[48] L. Yu and X. Pan, A new approach to image reconstruction in cone-beam CT with a circular orbit. In Proceedings of the VIIth International Conference on Fully 3D Reconstruction In Radiology and Nuclear Medicine, Saint Malo, France, 2003.
[49] K. Zeng, E. Bai and G. Wang, A fast CT reconstruction scheme for a general multi-core PC, *International Journal of Biomedical Imaging*, 2007.
[50] M. Zhao, J. Tian, J. Xue and X. Zhu, 3DMed: An integrated 3D medical image processing and analyzing system, In Proceedings of RSNA'04, 2004.
[51] M. Zhao, J. Tian, X. Zhu, J. Xue, Z. Cheng and H. Zhao, The design and implementation of a C++ toolkit for integrated medical image processing and analyzing. In Proceedings of SPIE Medical Imaging 2004, volume 5367, 2004, pp. 39–47.
[52] S.Y. Zhao, H.Y. Yu and G. Wang, A unified framework for exact cone-beam reconstruction formulas, *Medical Physics* **32**(6) (2005), 1712–1721.
[53] J. Zheng, J. Tian, Y. Dai, X. Zhang, D. Dong and M. Xu, Ultrasound-directed robotic system for thermal ablation of liver tumors: a preliminary report. volume 7629, SPIE, 2010, pp. 76290Z.
[54] S. Zhu, J. Tian, G. Yan, C. Qin and J. Feng, Cone beam micro-CT system for small animal maging and performance evaluation, *International Journal of Biomedical Imaging* (vol. 2009), Article ID 960573, 2009.
[55] Y. Zou and X.C. Pan, Exact image reconstruction on pi-lines from minimum data in helical cone-beam CT, *Physics in Medicine and Biology* **49**(6) (2004), 941–959.
[56] Y. Zou and X.C. Pan, Image reconstruction on pi-lines by use of filtered backprojection in helical cone-beam CT, *Physics in Medicine and Biology* **49**(12) (2004), 2717–2731.
[57] Y. Zou, X.C. Pan and E.Y. Sidky, Theory and algorithms for image reconstruction on chords and within regions of interest, *Journal of the Optical Society of America A-Optics Image Science and Vision* **22**(11) (2005), 2372–2384.