Journal of
Synchrotron
Radiation

ISSN 0909-0495

Received 25 May 2009
Accepted 29 March 2010

Image processing pipeline for synchrotron-radiation-based tomographic microscopy

C. Hintermüller,[a,b] F. Marone,[a] A. Isenegger[a] and M. Stampanoni[a,b]*

[a]Paul Scherrer Institut (PSI), CH-5232 Villigen-PSI, Switzerland, and [b]Institute for Biomedical Engineering, ETH and University Zürich, 8092 Zürich, Switzerland.
E-mail: marco.stampanoni@psi.ch

With synchrotron-radiation-based tomographic microscopy, three-dimensional structures down to the micrometer level can be visualized. Tomographic data sets typically consist of 1000 to 1500 projections of 1024 × 1024 to 2048 × 2048 pixels and are acquired in 5–15 min. A processing pipeline has been developed to handle this large amount of data efficiently and to reconstruct the tomographic volume within a few minutes after the end of a scan. Just a few seconds after the raw data have been acquired, a selection of reconstructed slices is accessible through a web interface for preview and to fine tune the reconstruction parameters. The same interface allows initiation and control of the reconstruction process on the computer cluster. By integrating all programs and tools, required for tomographic reconstruction into the pipeline, the necessary user interaction is reduced to a minimum. The modularity of the pipeline allows functionality for new scan protocols to be added, such as an extended field of view, or new physical signals such as phase-contrast or dark-field imaging *etc.*

Keywords: synchrotron radiation tomographic microscopy; imaging pipeline; data postprocessing.

1. Introduction

Structures down to the micrometer level of millimeter-sized objects can be visualized using synchrotron radiation microscopy. In combination with tomographic techniques (Kak & Slaney, 2001), it is possible to investigate and quantify the three-dimensional structure of materials (Le *et al.*, 2008; Zabler *et al.*, 2008; Trtik *et al.*, 2007; Gallucci *et al.*, 2007; Seright *et al.*, 2003), biological systems (Risser *et al.*, 2009; Stiller *et al.*, 2009; Heethoff *et al.*, 2008; Heinzer *et al.*, 2008; Schneider *et al.*, 2007; Schittny *et al.*, 2007), fossils (Gostling *et al.*, 2008; Hagadorn *et al.*, 2006) and a wide variety of other samples in a non-destructive way. Additionally, time-, temperature- and stress-dependent (Bouchard *et al.*, 2008; Voide *et al.*, 2008; Prodanovic *et al.*, 2006; Drummond *et al.*, 2006) changes can be tracked. Such tomographic experiments can be conducted at the Swiss Light Source (SLS) at the beamline for tomographic microscopy and coherent radiology experiments (TOMCAT) (Stampanoni *et al.*, 2006).

The TOMCAT beamline has been designed and optimized for tomographic imaging in the hard X-ray regime (8–45 keV). A fast and efficient detector system allows for continuous acquisition of projections while the sample is rotating at a constant angular velocity. The detector consists of a scintillator which converts the X-rays to visible light and a camera capable of writing a projection image to disk while simultaneously acquiring the next. A standard absorption-contrast tomographic scan in this mode takes about 2–20 min, depending on exposure time, and produces a data set with 1000 to 1500 projections and between 1024 × 1024 and 2048 × 2048 pixels per projection. Scan protocols for differential phase-contrast (DPC) (Weitkamp *et al.*, 2005) and dark-field imaging (Pfeiffer *et al.*, 2008) require the acquisition of multiple projections per angular step.

One way to deal with the growing amount of data recorded by state-of-the-art synchrotron-based tomographic microscopy systems is to use hardware purpose-built for image processing and computer graphics (Wang *et al.*, 1999; Dowd *et al.*, 1999) and large computing clusters (De Carlo & Tieman, 2004; De Carlo *et al.*, 2002) to reconstruct the data. The cluster farms and computing grids may even be centralized off-site (Wang *et al.*, 1999) in dedicated computing departments. These are mainly used for running the reconstruction process, either using filtered back-projection or by regridding techniques. The image correction and normalization prior to reconstruction is carried out on a dedicated computer which acts as a server for actively distributing the data to the compute nodes and to control them. Fully integrated graphical user interfaces have been developed (De Carlo *et al.*, 2002; Rivers *et al.*, 2004) to control the experiment, the preprocessing and the reconstruction process.

We implemented a light-weight modular system which offers a well defined interface for seamless and fast integration of emerging image acquisition methods, like differential (Weitkamp *et al.*, 2005) or propagation-based phase-contrast (Groso *et al.*, 2006) and dark-field imaging (Pfeiffer *et al.*, 2008) and allows to tune the reconstruction parameters and control the pre- and post-processing of the recorded tomographic data sets while the next samples are scanned. For studies with a large number of similar samples a fast online preview is provided for managing all the samples and to verify the quality of the scan before the sample is dismounted. To achieve this, we analyzed and restructured the work- and dataflow of the reconstruction process and developed strategies to use the computing resources provided by the available soft and hardware platform as efficient as possible. The following requirements were considered important for a software system dedicated to postprocess raw tomographic data sets of up to 20 GB in size: (i) run all processing in RAM, (ii) reduce disk-based input and output to a minimum, (iii) provide a user-friendly interface. These requirements are mainly imposed by the technical environment like file-servers, network connection, physical memory, swap-space on disk *etc*.

2. Methods

The main components of the postprocessing pipeline (Fig. 1) are described in the following. The data are directly fed from the camera into the pipeline *via* the camera server (§2.1). The initial processing of the data is performed by the projection-to-sinogram conversion software *Prj2Sin* described in §2.2. To allow users to preview their data based on a representative set of slices, the corresponding sinograms are precomputed while a sample is scanned. This is accomplished by updating each sinogram on the fly as soon as the next angular projection of the sample is available. If necessary, the strategy which and how many slices are selected for this set, currently every 50th, can easily be changed. Based on this set of sinograms and slices, various parameters for the reconstruction, the size of the images and the gray value range can be easily tuned *via* a web-based reconstruction manager (§2.3) on the fly. We decided to use a web-page-based interface, since it does not require the installation of any additional software and a web browser provides a commonly used and familiar user interface, with well defined standardized graphical components.

The reconstruction of the full volume is initiated from within the web interface. Since each slice can be calculated independently from other slices, the reconstruction job is split among the nodes of a cluster and managed with our custom *DICAT* application (§2.4).

2.1. Data acquisition: camera server

The first element in the pipeline (Fig. 1) is the camera server. It controls the data acquisition and transfers the projection data to the file server. Currently, the camera server handles a PCO.2000 CCD camera which features up to 14 frames s^{-1} at a nominal dynamic range of 14-bit (see Table 1 for additional details).

The beamline hardware is accessed and controlled through EPICS, the Experimental Physics and Industrial Control System (Advanced Photon Source, Argonne National

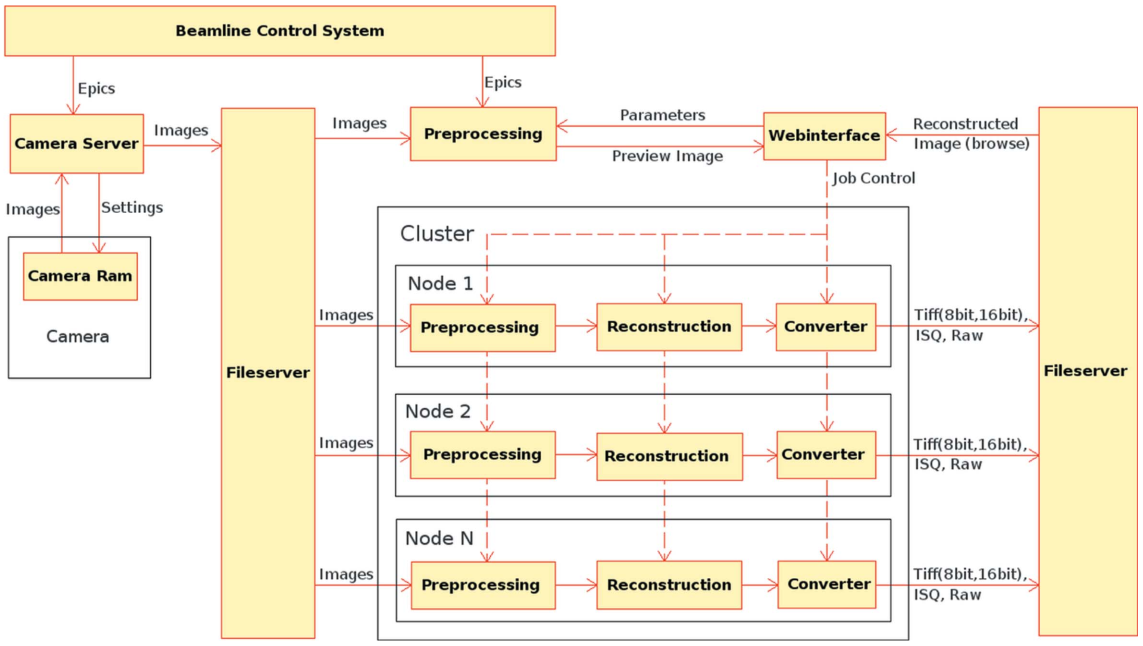


Figure 1
Sketch of the postprocessing pipeline. The data are fed into the pipeline by the camera server. Immediately after a scan the results for a selection of slices can be previewed and the reconstruction parameters tuned *via* a web-based application. The workload of the reconstruction process is distributed among several computing nodes. The reconstructed slices are available on the file server at the end of the reconstruction process in different image file formats.

Table 1
Specification of the PCO.2000 (PCO AG, Donaupark 11, 93309 Kelheim, Germany) CCD camera used as standard detector at the TOMCAT beamline.

Type	Interline CCD
Dynamic range	14-bit nominal
Readout noise	9 e^- r.m.s.
Dark current	0.5 e^- per pixel
Size	2048 × 2048 pixels
Binning	2× maximum
Frame rate	14.7 frames s^{-1} maximum
On device RAM	4 GB
FIFO mode	RAM operated as cyclical image cache
Connectors	IEEE 1394a FireWire

Laboratory, Argonne, IL 60439, USA), from a Linux environment.

'Physical' EPICS channels are defined for each device like motors, pumps, vacuum gauges or filters. 'Virtual' channels can also be defined, for example for scan parameters such as number of projections or exposure time. A tomographic experiment is controlled by two applications. The first application, written in SPEC (Certified Scientific Software, Cambridge, MA 02139, USA) and running on a Linux environment, controls the tomographic scan. The second application, written in C++, runs on the camera Windows PC and regulates the data acquisition. Communication between these two applications occurs *via* 'virtual' EPICS channels. The event sequence used by these applications to communicate to each other is summarized in Table 2. Fast preview features are provided by a camera plug-in for ImagePro (Media Cybernetics, Bethesda, MD, USA). At the beginning of the beam time the fast preview is used to align the rotation axis with the axis of the camera and center it in the field of view. Currently, such alignments are still done manually, but we are developing a fully automated alignment tool (Mader *et al.*, 2010).

Table 2
Event sequence used by the camera server (C++, windows) and control script (SPEC, Linux) during a tomographic scan (continuous rotation mode).

	SPEC application	Camera server
1	Monitor EPICS channel for scan start	Monitor EPICS channel for scan start
2	Read relevant scan parameters from EPICS channels	Read relevant scan parameters from EPICS channels
3	Log scan parameters	Divide camera RAM into four segments according to number of projections and image size
4	Calculate rotation speed from frame time and number of projections	Prepare camera for acquisition
5	Move sample to starting position	Wait for software trigger
6	Close shutter for dark images	
7	Send software trigger to the camera	Detect software trigger
8	Wait for acquisition completion	Acquire dark images in FIFO mode
9	Open the shutter; move sample out of beam	Wait for software trigger
10	Send software trigger to the camera	Detect software trigger
11	Wait for acquisition completion	Acquire flat images in FIFO mode
12	Move sample in beam	Wait for software trigger
13	Start sample rotation	
14	Send software trigger to the camera	Detect software trigger
15	Wait for acquisition completion	Acquire projection images in FIFO mode
16	Move sample out of beam	Wait for software trigger
17	Send software trigger to the camera	Detect software trigger
18	Wait for acquisition completion	Acquire flat images in FIFO mode
19	Move sample to initial position	
20	Monitor of EPICS channel for scan start	Monitor of EPICS channel for scan start

2.2. Sinogram generation

The principles of X-ray computer tomography are described by Kak & Slaney (2001). For a parallel beam geometry, p projection images $I_{\rm P}$ of a sample are recorded at equidistant angles between 0 and π. Each scan starts with a set of d dark $I_{\rm D}$ and f_p flat $I_{\rm F}$ images, and at the end again f_p flat images are recorded. These images are used to baseline correct (dark) and normalize (flat) the raw projections. In conventional absorption tomography, equation (1) can be used to compute the normalized projections P based on the average dark $\overline{I_{\rm D}}$ and flat image $\overline{I_{\rm F}}$,

$$P = -\ln \frac{I_{\rm P} - \overline{I_{\rm D}}}{\overline{I_{\rm F}} - \overline{I_{\rm D}}}. \qquad (1)$$

The corrected projections P are then used to generate the sinograms. Each sinogram represents the Radon transform (Kak & Slaney, 2001) of one tomographic slice. In our pipeline (Fig. 1) the generation of corrected projections and sinograms is part of the data preprocessing step. In the following, we describe the strategies used to optimize the usage of resources processing and manipulating the images (§2.2.1 and §2.2.2) in RAM. Further, methods to find the center of rotation (§2.2.3) or to compensate for time-dependent fluctuations of the beam (§2.2.4) are presented.

2.2.1. Image processing. During preprocessing, the projections are read from disk and written back again line by line. By using a so-called scan-line approach, which allows direct access to single lines within an image, the overhead for computing one single corrected projection can be reduced.

While transferring and converting the data from the input image file into the internal image representation, it is possible to apply several standard numerical operations such as binning, horizontal and vertical shifting, and horizontal flipping. Further operations can be easily added at this stage: the desired functionality has to be provided to the ImageIO module (Fig. 2) *via* so-called callback functions which operate on single pixels. This approach allows for example to dark correct angular projections or update the average flat image $\overline{I_{\rm F}}$ on a per-pixel basis while the next projection or the flat is loaded. On reformulating equation (1) to

$$\begin{aligned} P &= -\ln\left(I_{\rm P} - \overline{I_{\rm D}}\right) + \ln\left(\overline{I_{\rm F}} - \overline{I_{\rm D}}\right) \\ &= P' + F', \end{aligned} \qquad (2)$$

it becomes obvious that the logarithmic projection $P' = -\ln(I_{\rm P} - \overline{I_{\rm D}})$ and flat-field $F' = \ln(\overline{I_{\rm F}} - \overline{I_{\rm D}})$ images can be calculated independently of each other. The projection P = $P' + F'$ can than be computed when both P' and F' are finally available. This enables the efficient imple-

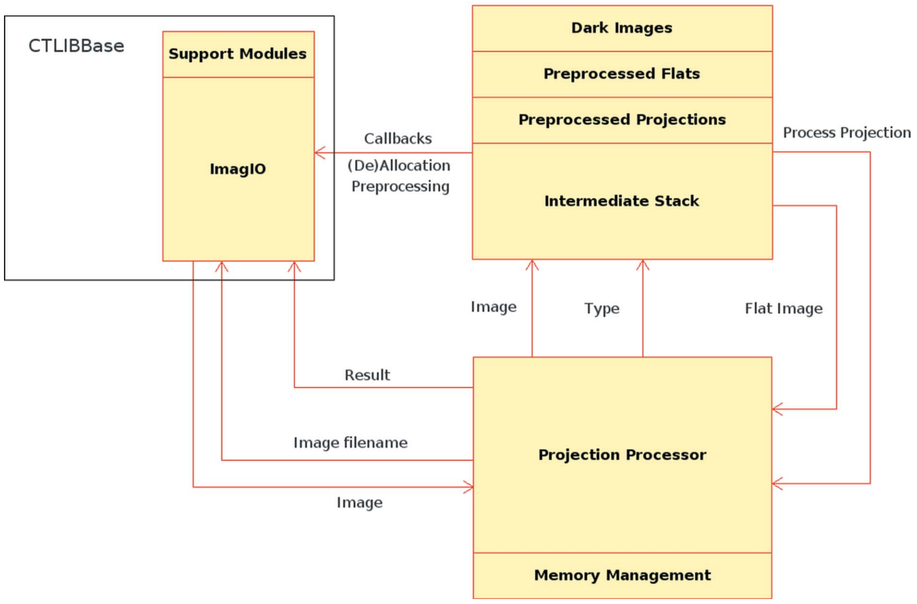


Figure 2
Schematic drawing of the sinogram generator. The main element is the intermediate image stack which keeps the darks and flats separate from the projections. It provides callbacks to the image loading module and to the image processor for image space management and flat finalization. The separation of the different types of images and the postprocessing is performed by the projection processor which offers several modules for the different postprocessing modes depending on the physical signal (absorption, phase, dark-field) that needs to be extracted.

mentation of a flat-field tracking routine which exploits the gray value gradient ∇_y along the vertical direction y of P (see §2.2.4).

2.2.2. Image separation and virtual subdivision of the image stack. All images in a scan are labeled in order, using consecutive indexes. This results in two distinct index ranges, one for the images recorded by the camera, including darks and flats, and one for the resulting set of corrected projections P computed during the sinogram generation. To separate the dark I_D and flat I_F images from the projections I_P, based on the index encoded in the file name of the images, the values for d, f_p and the total number p of P, an intermediate stack (Fig. 2), is used. This stack holds a subset of the projections currently processed. The number n_v of projections in this subset may be less than p. This enables the pipeline to handle data sets which are larger than the available amount of memory, by subdividing the input data into several subsets of images. In order to do this subdivision on the fly when the images are processed, it is necessary to know exactly how many images fit into the computer memory. Whenever space in RAM is needed, the operating system transfers data which are not accessed for a certain time as well as programs waiting for an external event to the swap file on disk. If needed again, they are moved back into RAM. We avoid the time overhead for moving data between RAM and swap using the following algorithm, based on the binary search method explained by Knuth (1998), to estimate the number of images n_v which fits into the RAM or for which space in RAM could be made available by the operating system.

(i) Assume that all p projections fit into RAM and set $n_v = p$.
(ii) Query the size of the physical memory $m_{\rm RAM}$.
(iii) Query the size of the free space in RAM $m_{\rm free}$.
(iv) Query the size of the free space on swap $s_{\rm free}$.
(v) Set the amount of memory available $m_{\rm avail} = \min(m_{\rm RAM}, m_{\rm free} + s_{\rm free})$.
(vi) Define a value $\Delta n_v = n_v/2$ for decrementing and/or incrementing n_v when fitting it to $m_{\rm avail}$.
(vii) Estimate the amount of memory $m_{\rm o}$ consumed by the n_l already loaded images and the structures allocated.
(viii) Estimate the amount of memory $m_{\rm r}$ that the remaining $n_{\rm r} = n_v - n_l$ images require.
(ix) If $m_{\rm r} > m_{\rm avail}$ or $m_{\rm r} + m_{\rm o} > m_{\rm RAM}$, decrement n_v by Δn_v; else increment n_v by Δn_v.
(x) Set $\Delta n_v = \Delta n_v/2$.
(xi) While $\Delta n_v > 0$ re-execute steps (vii) to (xi).

By updating n_v for the very first subset after every newly loaded image, n_v can be well adjusted to the effective space available from memory, which changes when other programs terminate or are started.

After the first subset of n_v images has been processed, the memory space is immediately re-used for the next set of images. This avoids allocation and de-allocation of storage through (slower) operating system calls. A further reduction in memory consumption and decrease in execution time is achieved by *load on demand* mechanisms, where only relevant parts of the images are loaded and postprocessed.

2.2.3. Center of rotation. After the sinograms have been generated, an estimate $C_{\rm e}$ for the center of rotation $C_{\rm r}$ is computed from the projections at 0° (P_0) and 180° (P_{180}) which are X pixels wide and Y pixels in height. We use a two-step approach to achieve this. First, the proper horizontal displacement $\xi_{0,180}$ between P_0 and P_{180} is determined. In case $\xi_{0,180} = 0$, P_{180} is the horizontally flipped version of P_0. For each discrete displacement of ξ pixels, with $-X < \xi < X$, the function $\delta^2(\xi)$, based on the mean squared difference of the two images (Fig. 3*a*), is computed as follows,

$$\delta^2(\xi) = \frac{1}{(X-|\xi|)\,Y}\sum_{y=1}^{Y}\sum_{x=1}^{X-|\xi|} \times \begin{cases} \left[P_0(\xi+x,y) - P_{180}(X-x+1,y)\right]^2 & \text{if } \xi \ge 0, \\ \left[P_0(x,y) - P_{180}(X-|\xi|-x+1,y)\right]^2 & \text{if } \xi < 0. \end{cases} \quad (3)$$

In contrast to cross correlation, which indicates the ξ at which the highest similarity between P_0 and P_{180} occurs, the value of $\delta^2(\xi)$ provides a direct measure for the difference between the two images within their overlapping area at the given offset ξ.

The discrete function $\delta^2(\xi)$ and its gradient $\nabla\delta^2(\xi)$, shown in Fig. 3(*a*), were computed for a pair of images (1024 × 1024

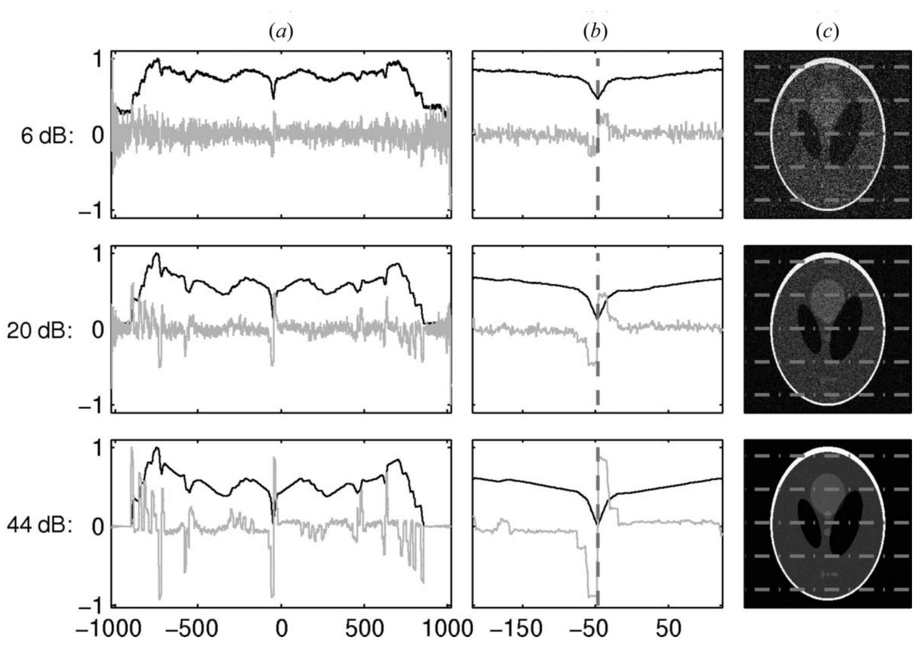


Figure 3
Method to find the effective displacement $\xi_{0,180}$ of two projections at 0° and 180°. (*a*) For each displacement $-X < \xi < X$ the mean squared difference $\delta^2(\xi)$ (black curve) and its gradient $\nabla\delta^2(\xi)$ (gray curve) is computed. (*b*) Zoom in to the offset position $\xi_{0,180}$ at which the function $\delta^2(\xi)$ exposes a minimum which is characterized by a notch-like shape, indicated by the biggest jump in the gradient $\nabla\delta^2(\xi)$ (dashed line). (*c*) Detector systems with increasing signal-to-noise ratios (6, 20, 44 dB) have been simulated by adding Gaussian noise. The stability of the algorithm is demonstrated by using only five rows (dash-dotted lines) of each image to compute $\delta^2(\xi)$.

pixels) displaying a Shepp–Logan phantom (Shepp & Logan, 1974). Gaussian noise (Fig. 3*c*) was added to simulate detector systems with increasing signal-to-noise ratios and, to demonstrate the stability of the algorithm, only every *k*th line (dashed dotted lines) was used. The function $\delta^2(\xi)$ is searched for local minima satisfying the following criteria,

$$\begin{aligned}&\nabla\delta^2(\xi-1) < 0, \qquad \nabla\delta^2(\xi+1) > 0,\\ &\nabla\delta^2(\xi_l) > 0, \qquad \nabla\delta^2(\xi_u) < 0,\\ &h(\xi) = \min\left[\delta^2(\xi_l) - \delta^2(\xi), \delta^2(\xi_u) - \delta^2(\xi)\right],\end{aligned} \tag{4}$$

with $-X < \xi_l < \xi$ and $\xi < \xi_u < X$. At each of these minima, $\delta^2(\xi)$ shows a notch. The value of the parameter $h(\xi)$, approximating the depth of these notches, is largest at the effective displacement $\xi = \xi_{0,180}$ (Fig. 3*b*). From $\xi_{0,180}$, the estimate C_e for the rotation center can be computed finally to $C_e = (X + \xi_{0,180})/2$.

The robustness of the algorithm is enhanced by excluding lines with a rather homogeneous or very broad gray-value distribution. This exclusion is based on the mean μ_g and standard deviation σ_g of the gray values of each line. Every line for which one of the following conditions holds is omitted:

$$\begin{aligned}\sigma_g < 0.01\Delta_g, \quad \sigma_g < 0.07\mu_g\\ \mu_g < 0.10\Delta_g, \quad \mu_g > 0.90\Delta_g.\end{aligned} \tag{5}$$

The above limits, based on the gray-value range Δ_g of the corrected projections P_0 and P_{180}, were found empirically on a large amount of images and proved to be independent of the extracted physical signal (absorption or phase).

The estimated C_e is used as an initial value for the further refinement of the rotation center calculation. This additional step is necessary, as the vertical orientation of the rotation axis can mechanically only be adjusted within a precision of one pixel, which can result in a variation of the center of rotation by the same amount between the top and bottom of a reconstructed data volume. To overcome this, we implemented a procedure based on image metrics (Donath *et al.*, 2006). This approach scores the tomographic reconstruction with three image metrics and iteratively determines the optimal position of the center of rotation. For our algorithm, we selected the 'integral of absolute value' metric, because, in our experience, it showed the highest robustness and reliability when applied to a variety of different samples. The optimal center of rotation is determined by minimization of this metric, efficiently performed through an iterative minimum search in subsequent steps with adjusted parameters starting from the initial guess C_e. If C_e has an accuracy of a few pixels, the implemented algorithm quickly converges to the optimal solution.

For the case where the rotation axis is not perfectly parallel to the vertical axis of the camera, an optimal center of rotation common to all slices will not be found. Determining the optimal rotation center for each single slice and using it for the reconstruction would overcome the misalignment problem. However, this is a very time-consuming operation, and thus rather impractical to run on every slice. We followed a different approach consisting of virtually aligning the rotation axis to the *z*-direction (*z*-axis) of the three-dimensional data set by appropriately shifting the lines in each sinogram along the horizontal direction.

The shifting parameter d_θ depends on the angle θ between the two axes. If the difference between the rotation center $C_{\rm top}$ of the top slice and $C_{\rm bottom}$ of the bottom slice is small, *i.e.* $|C_{\rm top} - C_{\rm bottom}| < 2$, than it can be assumed that $\sin\theta \simeq \theta << 1$ and d_θ can be set to $d_\theta = (C_{\rm top} - C_{\rm bottom})/N$.

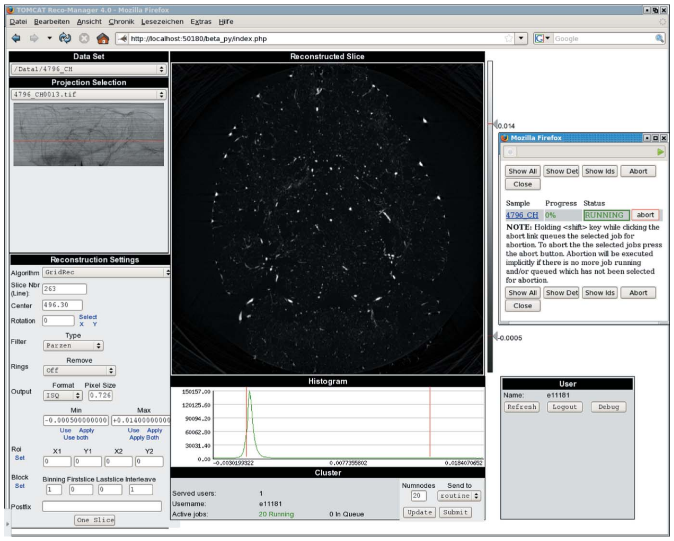
2.2.4. Flat-field tracking: correction for beam instabilities. In the ideal case, given a beam profile constant in time, the change in gray values in P is caused only by the sample, and ∇_y is minimal. By minimizing ∇_y^2, or the sum $S(\nabla_y^2)$ over all pixels [see equation (6)], it is possible to correct for time-dependent variations of the beam profile. In equations (1) and (2) each logarithmic projection P' is normalized with respect to the average $\overline{I_F}$ of all flats I_F. For flat-field tracking, the logarithmic flat image F' is computed for each I_F individually. With the ensemble of F_n' using (2) we generate all N possible corrected projections P_n for the image $I_{\rm P}$. For each P_n, the sum $S_n(\nabla_y^2)$ of ∇_y^2 can be computed as follows,

$$S_n(\nabla_y^2) = \sum_y \sum_x \left[\frac{P_n(x, y+1) - P_n(x, y-1)}{2} \right]^2, \qquad (6)$$

with n being the index of the nth logarithmic flat image out of a total of N available flats F' [equation (2)]. The projection P_n which corresponds to the lowest value of $S_n(\nabla_y^2)$ is selected and further processed.

2.3. Previewing and parameter selection

To allow rapid evaluation of the data quality at the end of a tomographic scan, a selection of sinograms is computed on the fly. In this way a subset of slices can be quickly reconstructed and previewed using a web-based application (Fig. 4). This application allows the reconstruction parameters to be selected and adjusted and the necessary commands to be sent to the cluster for reconstruction of the full three-dimensional data set. It consists of a set of ‘PHP Hypertext Processor’ scripts called by the browser *via* extended markup language (XML) based remote procedure calls (RPC). On the client side, scripts and functions written in JavaScript are used to verify the values adjusted by the user, trigger the RPC, prepare the data for reconstruction and adapt the set of displayed parameters to the selected reconstruction algorithm. Some of the parameters such as region of interest, range of slices to reconstruct, angle for rotating the slices, and gray-value range can be selected on the preview images by means of pointing and dragging. Furthermore, the interface allows to monitor and control (*e.g.* removing running or queued jobs) the progress of the reconstruction jobs on the cluster, based on the information provided by *DICAT* (see §2.4). New jobs can be submitted to the cluster and, if necessary, pending jobs can be removed from the job queues. In addition to the web interface, we also developed a console-based Python script for sending reconstruction processes to the cluster. This approach is useful for submitting several jobs to the cluster, when the reconstruction parameters are known, without the need to load the web interface. Both systems, the web interface and the command-line script, can be used to concurrently initiate the computation of several tomographic data sets.



Data Set: /Data1/4796_CH
Projection Selection: 4796_CH0013.tif

Algorithm: GridRec
Slice Nbr (Line): 263
Center: 496.30
Rotation: 0
Filter Type: Parzen
Rings Remove: Off
Output Format: ISQ Pixel Size: 0.726
Min: -0.000500000000 Max: +0.01400000000
Roi X1: 0 Y1: 0 X2: 0 Y2: 0
Block Binning: 1 Firstslice: 0 Lastslice: 0 Interleave: 1
Postfix:

Sample: 4796_CH Progress: 0% Status: RUNNING abort

NOTE: Holding <shift> key while clicking the abort link queues the selected job for abortion. To abort the the selected jobs press the abort button. Abortion will be executed implicitly if there is no more job running and/or queued which has not been selected for abortion.

Served users: 1
Username: e11181
Active jobs: 20 Running 0 In Queue
Numnodes: 20 Send to: routine

Name: e11181

Figure 4
Web interface for previewing a selected set of reconstructed slices from the acquired data set. The interface allows various reconstruction parameters to be tuned and the output format to be selected. The reconstruction can then be initiated on the cluster and its progress monitored.

2.4. Cluster control and management

To control and monitor the cluster we use an in-house-developed application, called *Distributed Computing Application for Tomography* (*DICAT*). It is a Java-based client/server application used to distribute the workload of the data processing to a set of compute nodes, thereby balancing the load among the different nodes.

The client is designed to run on any PC that is allowed to access the compute nodes *via* the network. It is used to submit and control the data processing jobs on the nodes such as image filtering and tomographic reconstruction. The servers execute scripts or applications that are used for reconstruction and pass the parameters submitted by the clients to them.

The goal is to have a flexible, lightweight, scalable and robust batch-job processing framework. This was achieved by a loose coupling between the clients and servers. The instances of the *DICAT* servers work independently and are unaware of each other. The *DICAT* server is designed such that clients can connect and disconnect at will, and thus machine reboots, network failures and other events will not crash the system. Once connected, a client can submit and control single or multiple jobs and query the status information of the jobs. The design supports the ‘submit and forget’ policy: if a job has been submitted by a client, it can terminate as the job queues are maintained by the servers.

A simple text file is used to configure the clients once they are started. This file contains a list of compute nodes to which the client shall try to connect. Using different server lists for each client allows groups of compute nodes to be formed. These groups may share any number of compute nodes or even be identical. Before a job is submitted to the servers, the client queries the currently connected servers for their load. Based on the corresponding numbers, the client tries to find a job distribution which ensures that all compute nodes finish their work simultaneously.

Access to the data has been limited on a per-user basis by utilizing the Unix framework for sharing accounts over the network between multiple computer systems. In this case the servers can be configured in such a way that a specific user can only access his/her data, but not the data of other users. When submitting jobs, the client application passes the information on the user account under which it has been started along to the *DICAT* servers.

The communication between client and server is achieved *via* transmission control protocol (TCP) sockets. Command and status information is exchanged by means of an application-specific 'command/acknowledge' protocol. This combination provides a robust communication inside a local area network (LAN). Automatic recovery from network interruptions, server terminations (for example by an administrator) or server hangs (for example when a *DICAT* process is denied CPU time) are handled by the *DICAT* client application.

3. Results

During a typical 48 h beam time at TOMCAT, users record up to 200 measurements, each of size between 4 and 13 GB. One of our goals was to provide the users with the reconstructions of their data before the end of their beam time. To achieve this, our postprocessing pipeline minimizes the number of manually initiated steps and optimizes the usage of computing resources, thereby reducing the required user interaction to a minimum. The recorded data sets are preprocessed and reconstructed while another sample is being scanned. Basically, the users need only to mount the sample and start the scan. A few seconds after the end of a scan they can preview the acquired data, tune the parameters for the reconstruction, and initiate the reconstruction process from within the web interface. A few seconds after termination of the reconstruction, data are available for simple three-dimensional visualization (ortho-slicing) which allows a first quality assessment of the measurement. During reconstruction, data are simultaneously backed-up on external hard drives so that users can leave the beamline with all their data right after the end of their allotted beam time.

Our imaging pipeline can handle several contrast mechanisms and select the corresponding algorithms to process the data: tomographic data are available in logarithmic scale for absorption-based tomograms. Phase information is extracted using either a grating interferometer method (Weitkamp *et al.*, 2005) or *via* a modified Bronnikov algorithm (MBA) (Groso *et al.*, 2006). For grating interferometry, a phase and an analyzer grating are used to extract the interference patterns related to the phase shifts introduced by the sample. For each angular position the phase grating is translated relative to the analyzer over one or two grating periods, with n equidistant steps. The phase gradient is extracted by computing the Fourier transform of the resulting intensity curve along each pixel (Weitkamp *et al.*, 2005). This complex sequence of mathematical operations is fully integrated in the sinogram generator allowing a smooth and fast postprocessing of large data sets (a grating interferometer tomographic scan contains usually up to 10000 single projections). The sinogram generator fully integrates the standard 360° off-center sample rotation method to double the horizontal field of view.

Table 3 lists the parameters and the overall time achieved to process the image data acquired in six different tomographic experiments conducted at TOMCAT. Three of them were performed in standard absorption and three using the DPC set-up. The size of the data sets varies depending on the number of dark and flat images, the number of projections, the range of sample rotation, the number of phase steps and whether hardware-binning was activated on the camera or not. The field of view available in the DPC experiments was vertically limited by the maximum height of the beam at 25 keV and horizontally by the dimensions of the gratings. All data sets were reconstructed on a cluster consisting of five two dual-core 3 GHz Intel Xeon (64-bit) nodes with 8 GB of RAM. Each node had a direct fiber-channel connection to a 15 TB General Parallel File System (GPFS) (version 3.2) based disk storage and was operated by the 64-bit version of Scientific Linux (SL) 5.

In Table 3 we compare the time needed to preprocess and reconstruct the data sets on the cluster using three different job configurations (single job, one job per node, four jobs per node) and on a single 2.6 GHz Intel Pentium 4 (32-bit) computer with 2 GB of RAM. This computer was operated by the 32-bit version of SL 5 and had access to the data through our beamline file server *via* a Network Filesystem version 3 (NFS3) mount. The *gridrec* algorithm (Dowd *et al.*, 1999) was used to reconstruct the absorption data while the phase (DPC) data were reconstructed using filtered back projection. Fig. 5 displays one reconstructed axial slice from four different samples measured in experiment 1, 2, 3 and 4 (see Table 3). In all three DPC experiments the same sample was used, but only the result of experiment 4 is shown.

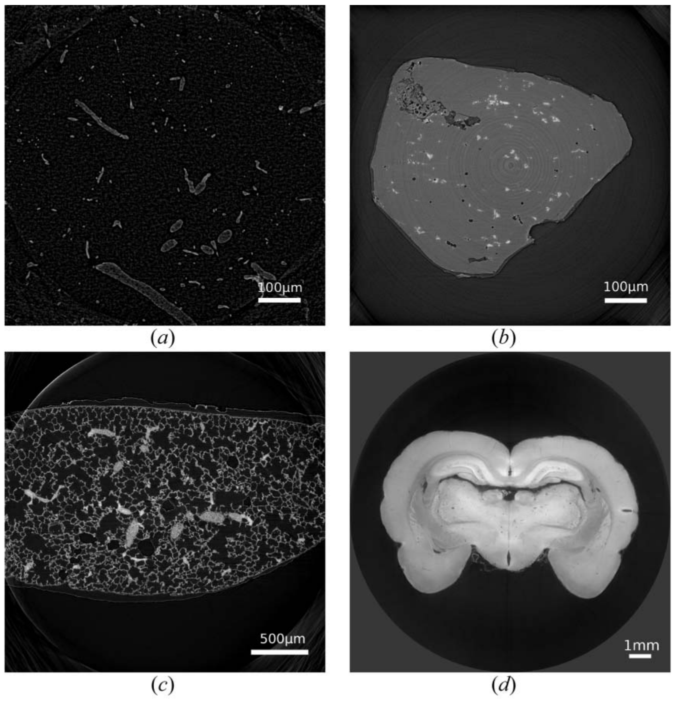


Figure 5
Tomographic slices of the samples scanned in the six experiments discussed. (*a*) Vascular cast of a mouse brain for assessing changes in the morphometry after irradiation (Hintermüller *et al.*, 2009). (*b*) Aluminium alloy Al2024-T38 (Taylor *et al.*, 2008). (*c*) Lung sample from the study of Schittny *et al.* (2007). (*d*) Dehydrated and paraffin-embedded rat brain (McDonald *et al.*, 2009).

Table 3
Acquisition parameters of six different tomographic experiments, three for absorption and three for phase contrast, and the total time to process their data.

All images mentioned in the table are 16-bit Tiff. The time listed for the single computer and the three different cluster job configurations includes all the necessary I/O, the generation of the corrected projections, the sinograms and the reconstruction of each single slice.

	Absorption			Phase contrast		
Experiment	1	2	3	4	5	6
Sample rotation	180°	180°	360°	180°	180°	360°
Magnification	20×	20×	10×	1.25×	4×	2×
Pixel size (µm)	0.74	0.37	0.74	11.84	1.85	3.7
Energy (keV)	13.5	19	12.5	25	25	25
Exposure time (ms)	80	80	125	200	140	500
Scan duration	5 min 1 s	8 min 57 s	28 min 23 s	20 min 56 s	1 h 35 min 26 s	2 h 21 min 24 s
X-ray projections						
No. of image files	1026	1507	3666	3715	31736	15966
No. of darks	5	2	5	5	5	5
No. of flats	20	4	60	10	10	10
No. of phase steps				10	21	11
No. of periods				1	2	1
No. of projections	1001	1501	3601	361	1501	1441
Pixels/image	1024 × 1024	2048 × 2048	2048 × 2048	944 × 318†	1686 × 501†	818 × 420†
Size (GB)	2	12	28, 6	2, 1	51	11
Results						
Algorithm	Gridrec			Filtered back-projection		
No. of slices	1024	2048	2048	318	501	420
Pixels/slice	1024 × 1024	2048 × 2048	3836 × 3836	944 × 944	1686 × 1686	1345 × 1345
Size (GB)	2.1	17	56	0.54	2.7	1.5
Single computer	2.6 GHz Intel Pentium 4, 2 GB RAM (32-bit)					
1 job/experiment	1 h 34 min 4 s	9 h 7 min 7 s	21 h 19 min 47 s	54 min 42 s	15 h 11 min 28 s	3 h 12 min 20 s
Cluster (five nodes)	3 GHz Dual-Core Intel Xeon, 8 GB RAM (64-bit)					
1 job/experiment	27 min 54 s	2 h 7 min 31 s	7 h 59 min 52 s	44 min 38 s	9 h 14 min 33 s	2 h 18 min 9 s
1 job/node	3 min 54 s	24 min 3 s	1 h 22 min	8 min 41 s	2 h 25 min 38 s	38 min 42 s
4 jobs/node	1 min 59 s	7 min 53 s	22 min 50 s	3 min 26 s	54 min 9 s	19 min 56 s

† Beam (V) and grating (H) size limited.

Experiment 1 is representative of a large number of absorption-based experiments (2 × 2 binning, 80 ms exposure) at the beamline. The 2 GB of raw data are reconstructed by a single cluster job within 28 min and by the single computer in 1.56 h. If the load is split among 20 jobs, the cluster accomplishes the reconstruction within 2 min.

By recording 1501 unbinned 2k × 2k images the resolution can be increased by a factor of two without changing the field of view. The cluster was able to process and reconstruct this data set in 8 min. This method was used in experiment 2 for failure analysis purposes on aluminium alloys (Taylor *et al.*, 2008) (Fig. 5*b*).

In experiment 3, Schittny *et al.* (2007) used a 360° off-center rotation to double the horizontal field of view. Fig. 5(*c*) shows the tomographic reconstruction of a rat lung obtained from 3601 projections. By splitting the reconstruction process into 20 jobs (four jobs per node), the time needed to preprocess and reconstruct the whole data set could be reduced to 23 min compared with the 8 h required when submitted as a single job. On the single computer (32-bit) it took 21.3 h to process this data set.

Experiments 4 to 6 are grating interferometry acquisitions. As mentioned before, this technique implies the acquisition of multiple images (phase steps) per angular projection in order to extract phase-contrast information. Experiment 4 is a standard low-resolution scan with ten phase steps per angular view. Our imaging pipeline performs a Fourier analysis of the intensity curve described by the ten phase steps on a pixel-by-pixel basis. In this particular case (361 projections) this implies the processing of 3715 images, including dark and baseline correction. Such work is performed in 55 min on a single machine and in 3 min on the cluster.

Experiment 5 describes a more sophisticated experiment where 21, instead of 10, phase steps have been acquired over two grating periods. In addition, the resolution has been improved and as a consequence the number of angular views increased. The imaging pipeline postprocessed the resulting 31736 images in 55 min (cluster) or more than 15 h on a single machine.

Experiment 6 is the phase-contrast analogy of experiment 3, namely a so-called wide-field DPC scan (McDonald *et al.*, 2009). The cluster processed the 15966 X-ray images (11 phase steps, 1441 projections) in 20 min.

Reconstructed data can be converted into standard 8-bit or 16-bit Tiff (Adobe Systems, San Jose, CA, USA) single images, signed or unsigned single file ISQ (SCANCO Medical AG,

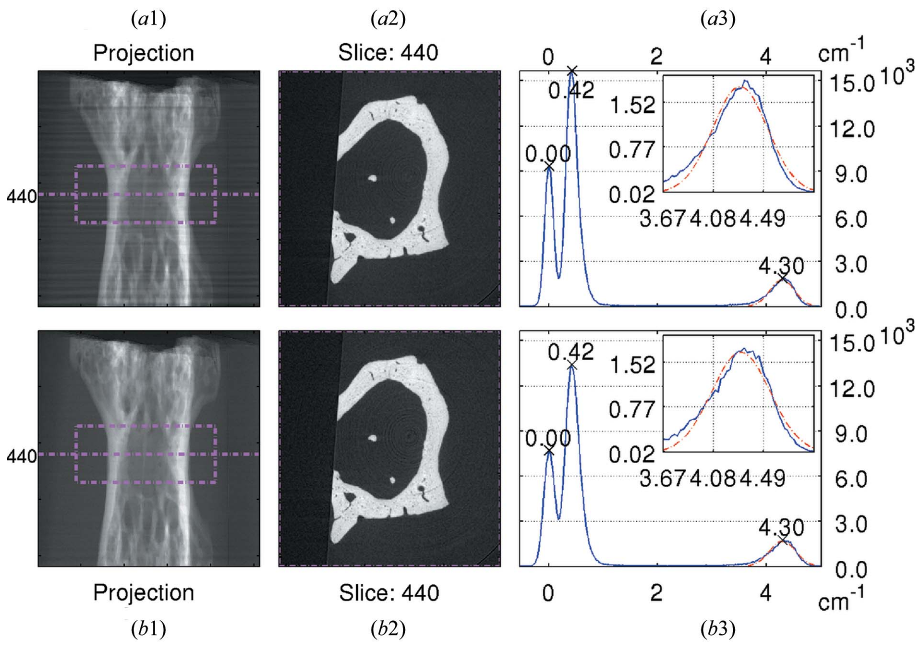


Figure 6
(*a*1), (*b*1) Corrected projection and (*a*2), (*b*2) tomographic slice (dash-dotted line) for a bone sample embedded in paraffin wax computed without (*a*1–*a*3) and with (*b*1–*b*3) flat-field tracking. (*a*3), (*b*3) Histograms computed for 201 adjacent slices from within the dash-dotted box. Plots were calibrated with the linear absorption coefficient of paraffin wax ($\mu_{\rm paraffin}$ = 0.42 cm^{-1}). The reconstructed slices were cropped to the area defined by this box. For evaluating the differences between the results computed without and with flat-field tracking we fitted a Gaussian function. The detailed plots show the fitted function for the peak at μ = 4.30 ± 0.054 cm^{-1} (dash-dotted line).

Fabrikweg 2, 8306 Brüttisellen, Switzerland) format or raw 32-bit floating point images.

4. Discussion

The pipeline design groups many complex details of the reconstruction process within a limited set of tasks. This results in a reduction of the required user interaction to a few well defined steps (mounting and aligning the sample, selecting the scan parameters, verifying the rotation center estimate, tuning the reconstruction parameters and initiating the reconstruction on the cluster).

Even with the short acquisition time achieved at TOMCAT it is feasible to reconstruct the data sets during the beam time by processing the data in parallel to the running scan. The users can then access the tomographic data to run preliminary evaluation and visualization during their beam time and after that they can take the reconstructed data with them on an external disk.

Support of new tomographic techniques and algorithms can be implemented rather quickly thanks to the modular structure of the pipeline software. We are currently integrating both dark-field (Pfeiffer *et al.*, 2008) and single-step DPC imaging (Zhu *et al.*, 2010).

Fig. 6 clearly shows that our flat-field tracking routine is able to correct single radiographic projections quite well. The increased background homogeneity in Fig. 6(*b*1) with respect to Fig. 6(*a*1) is evident. Figs. 6(*a*2) and 6(*b*2) show the same slice reconstructed without and with active flat-field tracking, respectively. Unfortunately, no visual improvement can be identified in this contrast-rich image. Even a more quantitative evaluation [Fig. 6(*a*3) and Fig. 6(*b*3)] shows no significant difference. One of the reasons for this is that not all angular projections of the tomographic data set could be corrected as well as in Fig. 6(*b*1). This is because the number of acquired flat-field projections was insufficient for 'sampling' all the beam fluctuation during a 5–10 min scan (and exposure time of 80–200 ms). We are systematically investigating this issue and are developing a more robust flat-field correction algorithm. This code will also take into account flat-field tracking for DPC imaging and be part of a future work.

Further developments aiming at the acquisition of full tomographic data sets in less than 1 s (Mokso *et al.*, 2009) will require a performance increase during acquisition and postprocessing of an order of magnitude. Options to upgrade the pipeline consider, for instance, replacing active file creation and modification detection by file-server-based notification and writing several files to disk in parallel. This parallelization could be implemented by triple buffering, based on an intermediate cache in the RAM of the Windows-based camera PC, combined with multiple worker threads, by tuning the file server mounting on Windows PC or even implementing the camera server and control software on Linux.

We thank Rene Kapeller and Colin Higgs from the SLS IT Infrastructure group and Heiner Billich from the Information Technology Division of PSI for supporting us in setting up the reconstruction pipeline and for generally maintaining the IT infrastructure at the TOMCAT beamline. We thank Samuel A. McDonald (SLS, now at University of Manchester) for providing the DPC data and Kerry L. Taylor (Materials Performance Centre, University of Manchester) and Professor Johannes C. Schittny (Institute of Anatomy, University of Bern) for permission to include data from their studies in this work.

References

Bouchard, P. O., Bourgeon, L., Lachapèle, H., Maire, E., Verdu, C., Forestier, R. & Logé, R. E. (2008). *Mater. Sci. Eng. A*, **496**, 223–233.
De Carlo, F., Albee, P., Chu, Y. S., Mancini, D. C., Tieman, B. & Wang, S. Y. (2002). *Proc. SPIE*, **4503**, 1–13.
De Carlo, F. & Tieman, B. (2004). *Proc. SPIE*, **5535**, 644–651.
Donath, T., Beckmann, F. & Schreyer, A. (2006). *J. Opt. Soc. Am. A*, **23**, 1048–1057.
Dowd, B. A., Campbell, G. H., Marr, R. B., Nagarkar, V., Tipnis, S., Axe, L. & Siddons, D. P. (1999). *Proc. SPIE*, **3772**, 224–236.

Drummond, J. L., De Carlo, F., Sun, K. B., Bedran-Russo, A., Koin, P., Kotche, M. & Super, B. J. (2006). *Proc. SPIE*, **6318**, 63182B.

Gallucci, E., Scrivener, K., Groso, A., Stampanoni, M. & Margaritondo, G. (2007). *Cement Concrete Res.* **37**, 360–368.

Gostling, N. J., Thomas, C. W., Greenwood, J. M., Dong, X., Bengtson, S., Raff, E. C., Raff, R. A., Degnan, B. M., Stampanoni, M. & Donoghue, P. C. J. (2008). *Evol. Develop.* **10**, 339–349.

Groso, A., Abela, R. & Stampanoni, M. (2006). *Opt. Express*, **14**, 8103–8110.

Hagadorn, W., Xiao, S. H., Donoghue, P. C. J., Bengtson, S., Gostling, N. J., Pawlowska, M., Raff, E. C., Raff, R. A., Turner, F. R., Chongyu, Y., Zhou, C., Yuan, X., McFeely, M. B., Stampanoni, M. & Nealson, K. H. (2006). *Science*, **314**, 291–294.

Heethoff, M., Helfen, L. & Cloetens, P. (2008). *J. Visual. Exp.* doi:10.3791/737.

Heinzer, S., Kuhn, G., Krucker, T., Meyerm, E., Ulmann-Schuler, A., Stampanoni, M., Gassmann, M., Marti, H. H. & Müller, R. (2008). *Neuroimage*, **39**, 1549–1558.

Hintermüller, C., Coats, J. S., Obenaus, A., Nelson, G., Krucker, T. & Stampanoni, M. (2009). *Proceedings of the 11th International Congress of the IUPESM*, edited by O. Dössel and W. C. Schlegel, pp. 423–426.

Kak, A. C. & Slaney, M. (2001). *Principles of Computerized Tomographic Imaging.* Philadelphia: Society for Industrial and Applied Mathematics.

Knuth, D. E. (1998). *Art of Computer Programming*, Vol. 3, *Sorting and Searching*, 2nd ed. Reading: Addison-Wesley.

Le, T. H., Dumont, P. J. J., Orgéas, L., Favier, D., Salvo, L. & Boller, E. (2008). *Composites*, A**39**, 91–103.

McDonald, S. A., Marone, F., Hintermüller, C., Mikuljan, G., David, C., Pfeiffer, F. & Stampanoni, M. (2009). *J. Synchrotron Rad.* **16**, 562–572.

Mader, K., Marone, F., Hintermüller, C., Mikuljan, G., Isenegger, A., Müller, R., Thiran, J.-P. & Stampanoni, M. (2010). In preparation.

Mokso, R., Marone, F. & Stampanoni, M. (2009). *10th International Conference on Synchrotron Radiation Instrumentation*, Melbourne, Australia, 27 September–2 October 2009.

Pfeiffer, F., Bech, M., Bunk, O., Kraft, P., Eikenberry, E. F., Brönnimann, C., Grünzweig, C. & David, C. (2008). *Nat. Mater.* **7**, 134–137.

Prodanovic, M., Lindquist, W. & Seright, R. (2006). *J. Colloid Interface Sci.* **298**, 282–297.

Risser, L., Plouraboué, F., Cloetens, P. & Fonta, C. (2009). *Int. J. Develop. Neurosci.* **27**, 185–196.

Rivers, M. L., Wang, Y. & Uchida, T. (2004). *Proc. SPIE*, **5535**, 783–791.

Schittny, J. C., Mund, S. & Stampanoni, M. (2007). *Am. J. Physiol. Lung Cell Mol. Physiol.* **294**, L246–L254.

Schneider, P., Stauber, M., Voide, R., Stampanoni, M., Donahue, L. R. & Müller, R. (2007). *J. Bone Miner. Res.* **22**, 1557–1570.

Seright, R., Liang, J., Lindquist, B. & Dunsmuir, J. (2003). *J. Petrol. Sci. Eng.* **39**, 217–230.

Shepp, L. A. & Logan, B. F. (1974). *IEEE Trans. Nucl. Sci.* **21**, 228–236.

Stampanoni, M., Groso, A., Isenegger, A., Mikuljan, G., Chen, Q., Bertrand, A., Henein, S., Betemps, R., Frommherz, U., Böhler, P., Meister, D., Lange, M. & Abela, R. (2006). *Proc. SPIE*, **6318**, 63180M.

Stiller, M., Rack, A., Zabler, S., Goebbels, J., Dalügge, O., Jonscher, S. & Knabe, C. (2009). *Bone*, **44**, 619–628.

Taylor, K. L., Sherry, A. H. & Goldthorpe, M. R. (2008). *Proceedings of ASME Pressure Vessels and Piping Conference*, pp. 921–926.

Trtik, P., Dual, J., Keunecke, D., Niemz, P., Mannes, D., Stähli, P., Kaestner, A., Groso, A. & Stampanoni, M. (2007). *J. Struct. Biol.* **159**, 46–55.

Voide, R., van Lenthe, G. H., Stauber, M., Schneider, P., Thurner, P. J., Wyss, P., Stampanoni, M. & Müller, R. (2008). *J. Jpn. Soc. Bone Morphom.* **18**, 9–21.

Wang, Y., De Carlo, F., Foster, I., Insley, F., Kesselman, C., Lane, P., von Laszewski, G., Mancini, D., McNulty, I., Su, M.-H. & Tieman, B. (1999). *Proc. SPIE*, **3772**, 318–327.

Weitkamp, T., Diaz, A., David, C., Pfeiffer, F., Stampanoni, M., Cloetens, P. & Ziegler, E. (2005). *Opt. Express*, **13**, 6296–6304.

Zabler, S., Rack, A., Manke, I., Thermann, K., Tiedemann, J., Harthill, N. & Riesemeier, H. (2008). *J. Struct. Geol.* **30**, 876–887.

Zhu, P., Zang, K., Wang, Z., Liu, Y., Liu, X., Wu, Z., McDonald, S. A., Marone, F. & Stampanoni, M. (2010). *Proc. Natl. Acad. Sci.* Submitted.