This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing

High-quality parallel-ray X-ray CT back projection using optimized interpolation

Michael T. McCann, *Member, IEEE* Michael Unser, *Fellow, IEEE*

***Abstract*—We propose a new, cost-efficient method for computing back projections in parallel-ray X-ray CT. Forward and back projections are the basis of almost all X-ray CT reconstruction methods, but computing these accurately is costly. In the special case of parallel-ray geometry, it turns out that reconstruction requires back projection only. One approach to accelerate the back projection is through interpolation: fit a continuous representation to samples of the desired signal, then sample it at the required locations. Instead, we propose applying a prefilter that has the effect of orthogonally projecting the underlying signal onto the space spanned by the interpolator, which can significantly improve the quality of the interpolation. We then build on this idea by using oblique projection, which simplifies the computation while giving effectively the same improvement in quality. Our experiments on analytical phantoms show that this refinement can improve the reconstruction quality for both filtered back projection and iterative reconstruction in the high-quality regime, i.e., with low noise and many measurements.**

***Index Terms*—X-ray tomography, computed tomography, interpolation, reconstruction algorithms.**

I. Introduction

In X-ray CT (and the closely-related SPECT and PET [1]), the underlying physical model only has meaning in the continuous domain, but reconstruction algorithms necessarily work on discrete signals: vectors and matrices. Therefore, we must chose a discretization scheme; i.e., fix how to represent a continuously-defined signal with a discrete one. In almost every case, this means that a continuous function, $f(\mathbf{x})$, is expressed (possibly only implicitly) as a weighted sum of kernels, $\hat{f}(\mathbf{x}) = \sum_k c_f[k]\psi_k(\mathbf{x})$. The choice of these kernels critically impacts the quality and speed of CT reconstruction. On one hand, the best quality will be obtained when the discretization can well approximate the signal to be reconstructed. On the other hand, to be practical, the discretization must allow fast forward and/or back projection, since these operations are the bottleneck of nearly all reconstruction methods. Unfortunately, these two criteria conflict: forward and back projection require repeatedly evaluating the X-ray transform of the discretization kernel, with the number of evaluations growing with the size of the (non-zero) support of the kernel. But, the kernels with the best approximation properties tend to be large or have complicated X-ray transforms.

One solution is to use a kernel with a good size/approximation trade-off and store its complicated X-ray transform in a lookup table. This was the approach of [2] with separable B-splines and [3] with the more-general box splines. Or, the same X-ray transforms can be computed approximately, for example approximating the X-ray transform of a voxel with a rectangle [4] or trapezoid [5], or of a separable B-spline with another separable B-spline of the same degree [6]. Another approach is to use isotropic kernels (also called *blobs*), which simplify projection because their X-ray transforms do not depend the projection direction. These can also be accelerated via lookup table [7], [8]. The downside is that isotropic kernels cannot satisfy the partition-of-unity property [9], which is necessary to ensure that the discretization error vanishes as the sampling step tends to zero. Using strips that match the geometry of the sensor [10], [11] is another way to enable fast forward and back projections, in exchange for increased approximation error. Finally, the discretization may be implicit; e.g., the filtered back projection (FBP) algorithm [12] presupposes a separable sinc discretization (because the projections are assumed to be bandlimited during filtering), but, in practice, the back projection step relies on some faster interpolation.

Though they are nonstandard in medical imaging, CT systems with parallel-ray geometry (where measurements of the sinogram are taken on a regular grid for each of a discrete set of projection directions) are used for phase-contrast X-ray imaging [13], [14], electron tomography [15], and single-particle cryo-electron microscopy [16]. In this case, the forward model, H, has the property that its normal operator, $H^T H$, is linear and shift-invariant. The result is that iterative reconstruction requires only a single back projection followed by one filtering operation per iteration, rather than the usual one forward and one back projection per iteration. This was noted first by [17] and extended recently by our group to 3D [8]. While this result only strictly holds in the bandlimited case, both groups opted instead to use discretizations with limited spatial support. This geometry also enables the back projection to be cheaply computed with convolution followed by interpolation. Using this approach, the cost of back projection is not tied to the support of the discretization kernel, but rather to the size of the interpolation kernel. The benefit of this is that it enables the use of large discretization kernels without a cost penalty; e.g., in the current work, we use the separable sinc as a discretization kernel, which matches the bandlimited assumption.

This work was funded (in part) by the Center for Biomedical Imaging of the Geneva-Lausanne Universities and EPFL, as well as by the Foundations Leenaards and Louis-Jeantet. This project has received funding from the European Reaserch Council (ERC) under the European Union's Horizon 2020 research and innovation program (GA No. 692726 GlobalBioIm).

Michael McCann is with the Center for Biomedical Imaging, Signal Processing Core and the Biomedical Imaging Group, EPFL, Lausanne, Switzerland (email: michael.mccann@epfl.ch).

Michael Unser is with the Biomedical Imaging Group, EPFL, Lausanne, Switzerland.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, which includes extended derivations of some of the mathematical results in the work.

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing

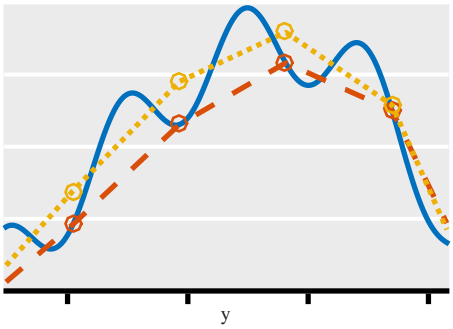


Fig. 1. Comparison of standard interpolation (dashed) and orthogonal projection (dotted) of a signal (solid). The standard interpolation matches the signal exactly at its knots, while the orthogonal projection is the best piecewise-linear fit.

In this paper, we propose a new interpolation scheme for parallel-ray X-ray CT back projection. Drawing on techniques from image resizing [18], [19], we apply a prefilter that has the effect of projecting the underlying signal onto the space spanned by the interpolator (Figure 1). This process creates an interpolation function that is the best possible fit (in the L_2 sense) to the signal. The outcome is a significant improvement in the back projection quality over the standard approach. In a further refinement, we show that nearly the same improvement can be obtained by instead using oblique projection, which simplifies the computation. Though this approach can be applied to any discretization scheme, we use the separable sinc basis. Not only does this enable analysis of spatial and angular sampling with the sampling theorem (along the lines of [20]), but it also makes $H^T H$ exactly shift invariant, enabling iterative reconstruction at the same speed as FBP (in the limit of high reconstruction size). This is because the cost of both FBP and the iterative method is dominated by the back projection (at a cost proportional to the number of pixels in the reconstruction times the number of views). The only additional computation required by the iterative method is a fixed number of filtering operations. Using a sinc discretization along with oblique interpolation allows efficient and high-quality parallel-ray X-ray CT reconstructions.

The outline of the paper is as follows. In Section II, we formulate the X-ray reconstruction problem and the back projection operation. In Section III, we describe the standard interpolation approach to back projection and propose two optimized interpolators based, respectively, on orthogonal and oblique projections. In Section IV, we describe a specific implementation of these methods using B-splines and separable sinc discretization. In Section V, we compare the proposed methods to standard interpolation in terms of speed and accuracy, both for back projection alone and for reconstruction of digital phantoms.

II. Back Projection

In this section, we fix our notation and formulate the back projection operation. We then show how this operation can be accelerated with interpolation.

A. Notation

We use lowercase letters for functions, infinite sequences, and scalars (e.g., f, g, and λ). To differentiate functions and sequences, we use parentheses for functions and square brackets for the sequences (e.g. $f(x)$ versus $g[m]$). Uppercase letters indicate operators (e.g., $H : l_2(\mathbb{Z}) \to l_2(\mathbb{Z})$). We use boldface lowercase letters for vectors (e.g., $\mathbf{x}$, $\mathbf{m}$, and $\boldsymbol{\theta}$) and boldface uppercase letters for matrices (e.g. $\mathbf{H}$, $\mathbf{P}$, and $\boldsymbol{\Lambda}$). Other symbols are defined as they are used.

B. Problem Setting

Let f be function of $\mathbb{R}^d$, $f : \mathbb{R}^d \to \mathbb{R}$ (usually with $d \in \{2, 3\}$). The X-ray transform of f, denoted $\mathcal{P}_{\boldsymbol{\theta}}\{f\}$, computes the line integrals of f along the direction specified by a unit vector $\boldsymbol{\theta} \in \mathbb{R}^d$. It is defined as

$$\mathcal{P}_{\boldsymbol{\theta}}\{f\}(\mathbf{y}) = \int_{\mathbb{R}} f(t\boldsymbol{\theta} + \mathbf{P}_{\boldsymbol{\theta}^\perp}^T \mathbf{y}) dt, \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^{d-1}$ is a coordinate on the detector plane, $\mathbf{P}_{\boldsymbol{\theta}^\perp}^T$ is a $d \times (d-1)$ matrix that has as its columns an orthogonal basis for the detector plane (which is the orthogonal complement of $\boldsymbol{\theta}$, a line in 2D or a plane in 3D), and $\mathbf{P}_{\boldsymbol{\theta}^\perp}^T \mathbf{y}$ is a vector specifying the offset of the line.

We fix a discretization scheme of the form

$$f(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^d} c_f[\mathbf{k}] \psi(\mathbf{x} - \boldsymbol{\Lambda}_{\mathbf{x}} \mathbf{k}), \tag{2}$$

where ψ is a discretization kernel, $\mathbf{k}$ is an integer multi-index, and $\boldsymbol{\Lambda}_{\mathbf{x}}$ is a $d \times d$ diagonal matrix containing the sampling step in each of d dimensions. This gives the discrete forward model, H, defined as

$$g[\mathbf{m}] = (Hc_f)[\mathbf{m}] = \sum_{\mathbf{k} \in \mathbb{Z}^d} c_f[\mathbf{k}] \mathcal{P}_{\boldsymbol{\theta}}\{\psi\}(\boldsymbol{\Lambda}_{\mathbf{y}} \mathbf{m} - \mathbf{P}_{\boldsymbol{\theta}^\perp} \boldsymbol{\Lambda}_{\mathbf{x}} \mathbf{k}),$$

where, paralleling Eq. (2), $\mathbf{m}$ is an integer multi-index with $\mathbf{m} \in \mathbb{Z}^{d-1}$ and $\boldsymbol{\Lambda}_{\mathbf{y}}$ is a $(d-1) \times (d-1)$ diagonal matrix containing the sampling step in the projection domain in each of $d-1$ dimensions.

In the X-ray reconstruction problem, we have g and aim to recover f (expressed in terms of its coefficients, c). Practically, we measure g for each of a discrete set of views (called a *sinogram*); however, because the X-ray transform is linear, we present everything here for a single view with the understanding that multiple views are handled by iterating over each and summing the results. When ψ is bandlimited and the sinogram is properly sampled in space, the system can be inverted by deconvolving the back projected measurements, $H^T g$, by the kernel corresponding to the normal operator, $H^T H$ [8]. Our focus here is the the back projection because it presents the computational bottleneck.

C. Back Projection

Back projection requires computing, for each $\mathbf{k} \in \mathbb{Z}^d$,

$$(H^T g)[\mathbf{k}] = \sum_{\mathbf{m} \in \mathbb{Z}^{d-1}} g[\mathbf{m}] \mathcal{P}_{\boldsymbol{\theta}}\{\psi\}(\boldsymbol{\Lambda}_{\mathbf{y}} \mathbf{m} - \mathbf{P}_{\boldsymbol{\theta}^\perp} \boldsymbol{\Lambda}_{\mathbf{x}} \mathbf{k}). \tag{3}$$

If the reconstruction and measurement grids are square with side length K and M, respectively, this costs $O(K^2 M)$

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing

operations in 2D and $O(K^3M^2)$ operations in 3D: for each $\mathbf{k}$ in the reconstruction, we sum over each $\mathbf{m}$ in the sinogram. This cost can be reduced to $O(K^2)$ (or to $O(K^3)$, in 3D), if ψ is chosen so that $\mathcal{P}_\theta\{\psi\}$ has a small spatial support. Back projection can be very expensive in practice, which has lead to the development of various methods to accelerate the computation via interpolation either in space or Fourier domain, at the cost of some loss of accuracy.

In preparation for our subsequent discussion of interpolation, we rewrite (3) as samples of a continuous function,

$$(H^T g)[\mathbf{k}] = r(\mathbf{P}_{\theta^\perp}\mathbf{\Lambda_x}\mathbf{k}) = (h * \mathcal{P}_\theta\{\psi\})(\mathbf{P}_{\theta^\perp}\mathbf{\Lambda_x}\mathbf{k}), \quad (4)$$

where $h(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} g[\mathbf{m}]\delta(\mathbf{y} - \mathbf{\Lambda_y}\mathbf{m})$. This convolutional form reveals that calculating the values of r on any grid with a sampling step $\mathbf{\Lambda_y}/n$ for $n \in \mathbb{Z}$ is possible with a discrete convolution between an upsampled version of the measurements, and a sampled version of the X-ray projection of the discretization kernel, $p[\mathbf{m}] = \mathcal{P}_\theta\{\psi\}(\mathbf{\Lambda_y}\mathbf{m}/n)$:

$$r\left(\frac{\mathbf{\Lambda_y}\mathbf{m}}{n}\right) = (U_n g * p)[\mathbf{m}],$$

where U_n is upsampling-by-n, i.e. $U_n g[\mathbf{k}n] = g[\mathbf{k}]$. As a result, we can compute these values efficiently with the FFT. Unfortunately, the points where we must sample r for back projection are not on this grid, and so we must interpolate them.

III. Interpolation

We now describe how interpolation can be used to compute back projections and present our proposed optimized interpolators.

A. Standard interpolation

The standard approach in interpolation (e.g., image rotation or rescaling) is to fit a continuous representation to the discrete samples at hand and to sample it at the desired points (Figure 1). For back projection, this amounts to evaluating r on a fine grid and using, e.g., bilinear interpolation to find the necessary values. This has been [21] and continues to be [8] standard in tomography. Explicitly, we pick a generating function, φ, and approximate r as

$$\tilde{r}(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} c[\mathbf{m}]\varphi\left(n\mathbf{\Lambda_y}^{-1}\mathbf{y} - \mathbf{m}\right), \quad (5)$$

where c is chosen such that $\tilde{r}((\Lambda_\mathbf{y}/n)\mathbf{m}) = r((\Lambda_\mathbf{y}/n)\mathbf{m})$ (note that we use c here and c_f in (2) to denote a sequence of coefficients, but not to imply that these sequences are the same). If the generating function is interpolatory (e.g., B-splines of degree 0 or 1, corresponding to nearest neighbor or bilinear interpolation), then $c[\mathbf{m}] = r((\Lambda_\mathbf{y}/n)\mathbf{m})$. In general, however, this simple resampling relation does not hold.

Doing this results in a significant speedup if the support of φ is much smaller than that of $\mathcal{P}_\theta\{\psi\}$. This is because the number of nonzero terms in either (3) or (5) that must actually be computed depends on the spatial support of φ and $\mathcal{P}_\theta\{\psi\}$, respectively. In 3D, the computational cost goes from $O(K^3M^2)$ to $O(K^3W^2)$, where W is the width of the support of φ: for each $\mathbf{k}$ in the reconstruction, we compute (5), which has the same form as the sum in (3), except that we can choose φ to have a small spatial support. Computing $\mathbf{r}$ is just a $(d-1)$-dimensional filtering operation and, in most practical cases, so is computing $\mathbf{c}$, meaning that these costs are negligible.

B. Interpolation With Orthogonal Projection

The strategy of standard interpolation is to find a continuous function that matches r exactly on a grid of points. Inspired by work on image resizing [18], we propose instead to compute the least squares projection of r onto the space spanned by the generating function, φ. The resulting interpolator, $\tilde{r}$, will be, by construction, closer (in the L_2 sense) than any other one using the same generating function; e.g., for linear interpolation, this process will return the best piecewise linear approximation to the function r (see Figure 1). Our motivation is that making $\tilde{r}$ closer (in the L_2 sense) to r should improve the accuracy of the interpolation at the points $\mathbf{P}_{\theta^\perp}\mathbf{\Lambda_x}\mathbf{k}$, resulting in a more faithful back projection.

For the moment, we take a sampling step of one; i.e., $n\mathbf{\Lambda_y}^{-1}\mathbf{y} = \mathbf{y}$. Let $V(\varphi)$ denote the space spanned by the integer shifts of φ and let $P_\varphi r$ denote the orthogonal projection of r onto this space. Then,

$$(P_\varphi r)(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} \langle r, \mathring{\varphi}(\cdot - \mathbf{m})\rangle\, \varphi(\mathbf{y} - \mathbf{m}), \quad (6)$$

where $\mathring{\varphi} \in V(\varphi)$ is the unique dual of φ so that $\langle\varphi, \mathring{\varphi}(\cdot - \mathbf{m})\rangle = \delta[\mathbf{m}]$. Substituting r from (4) gives

$$(P_\varphi r)(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} ((U_n g) * q)[\mathbf{m}]\varphi(\mathbf{y} - \mathbf{m}) \quad (7)$$

where $q[\mathbf{m}] = (\mathcal{P}_\theta\{\psi\} * \mathring{\varphi})(-\mathbf{m})$. The full development of this expression relies on the linearity of the inner product and changes of variables; we leave it to the supplementary material for brevity. For the general case where $n\mathbf{\Lambda_y}^{-1}\mathbf{y} \neq \mathbf{y}$, we rescale by $1/n$, giving

$$(P_\varphi r)(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} ((U_n g) * q)[\mathbf{m}]\varphi(n\mathbf{\Lambda_y}^{-1}\mathbf{y} - \mathbf{m}), \quad (8)$$

where

$$q[\mathbf{m}] = \frac{n}{\det(\mathbf{\Lambda_y})}(\mathcal{P}_\theta\{\psi\} * \mathring{\varphi}(n\mathbf{\Lambda_y}^{-1}\cdot))\left(-\frac{\mathbf{\Lambda_y}}{n}\mathbf{m}\right).$$

We leave the implementation details of (8) to Section IV, and, again, we leave further development of the expression to the supplementary material. We also note that it is a specific case of the method described in [19] and the references therein.

In contrast to the standard interpolation in (5), the orthogonal interpolation does not require the interpolator to exactly match the underlying signal at the knots. We compare these interpolation strategies in Figure 1.

C. Interpolation With Oblique Projection

The main drawback of the orthogonal projection in (8) is that it requires continuous inner products between $\mathcal{P}_\theta\{\psi\}$ and $\mathring{\varphi}$; these integrals are costly to compute accurately. We instead propose an oblique projection: we project r onto $V(\varphi_2)$ such

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing

that the error is orthogonal not to $V(\varphi_2)$, but to a different space, $V(\varphi_1)$. We denote this as $P_{\varphi_2,\varphi_1} r$. Intuitively, when $V(\varphi_1)$ and $V(\varphi_2)$ are similar (as will be the case when they are both useful for interpolation), $P_{\varphi_2,\varphi_1} r$ and $P_{\varphi_2} r$ will be close; [19] formalizes this with a theorem that states that the oblique projection has error no more than $1/\cos(\theta_{12})$ times that of the orthogonal projection, where θ_{12} is the largest angle between the subspaces. The advantage of the oblique projection is that φ_1 can be chosen to simplify the inner product computations, while φ_2 can be chosen to give superior interpolation.

The oblique projection is similar to the orthogonal one, (8), with the addition of discrete filtering operation:

$$(P_{\varphi_2,\varphi_1} r)(\mathbf{y}) = \sum_{\mathbf{m}\in\mathbb{Z}^{d-1}} (d * (U_n g) * q)[\mathbf{m}] \varphi_2(n\boldsymbol{\Lambda}_{\mathbf{y}}^{-1}\mathbf{y} - \mathbf{m}), \tag{9}$$

where

$$q[\mathbf{m}] = \frac{n}{\det(\boldsymbol{\Lambda}_{\mathbf{y}})} (\mathcal{P}_{\boldsymbol{\theta}} \{\psi\} * \varphi_1(-n\boldsymbol{\Lambda}_{\mathbf{y}}^{-1}\cdot)) \left(-\frac{\boldsymbol{\Lambda}_{\mathbf{y}}}{n}\mathbf{m}\right)$$

and the filter d is given by

$$\left(\langle \varphi_1(\mathbf{y} - \mathbf{m}'), \varphi_2(\mathbf{y})\rangle_{\mathbf{y}} \overset{m'}{*} d[\mathbf{m}']\right)[\mathbf{m}] = \delta[\mathbf{m}],$$

where $\overset{m'}{*}$ denotes discrete convolution over m'. For a verification that (9) indeed provides an oblique projection as described above, please refer to the supplementary material.

Implementing this projection only requires continuous inner products between $\mathcal{P}_{\boldsymbol{\theta}} \{\psi\}$ and φ_1 and between φ_1 and φ_2. In the next section we show how a particular selection of these functions makes the computation simple.

IV. Implementation

Our discussion so far has been general with regard to the selection of the discretization and interpolation functions. We now fix these and present the entire reconstruction algorithm.

A. Discretization Kernel

We use the separable sinc as our discretization kernel,

$$\psi(\mathbf{x}) = \prod_{i=1}^{d} \operatorname{sinc}(x_i/\lambda_i),$$

where $\operatorname{sinc}(x) = \sin(\pi x)/(\pi x)$. When $d = 2$, we have

$$\mathcal{P}_{\boldsymbol{\theta}} \{\psi\} (y) = \frac{\lambda_1\lambda_2}{\lambda_{\max}} \operatorname{sinc}(y/\lambda_{\max}),$$

where $\lambda_{\max} = \max(|\boldsymbol{\Lambda}_{\mathbf{x}} \mathbf{P}_{\boldsymbol{\theta}^\perp}^T|)$. This can be shown using the Fourier central slice theorem [22]. When $d = 3$, the X-ray transform is considerably more complicated. A useful intermediate case is when there is a fixed projection axis, i.e., $\mathbf{P}_{\boldsymbol{\theta}^\perp}^T[\cdot, 2] = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$. In this case,

$$\mathcal{P}_{\boldsymbol{\theta}} \{\psi\} (\mathbf{y}) = \frac{\lambda_1\lambda_2}{\lambda_{\max}} \operatorname{sinc}(y_1/\lambda_{\max}) \operatorname{sinc}(y_2/\lambda_3),$$

where $\lambda_{\max} = \max(|\boldsymbol{\Lambda}_{\mathbf{x}} \mathbf{P}_{\boldsymbol{\theta}^\perp}^T[\cdot, 1]|)$.

B. Interpolation Using B-Splines

B-splines are a good choice for the interpolation function due to their limited spatial support and good approximation properties [23]. In the standard interpolation approach in 2D, this means interpolating using (5) with $\varphi(y) = \beta^a(y)$ and $c = (b^a)^{-1} * r$, where $(b^a)^{-1}$ is the direct B-spline filter associated with the B-spline of degree a and can be implemented efficiently [24].

In the case of orthogonal projection in 2D, letting $\varphi(y) = \beta^a(y)$ gives

$$q[m] = (b^{2a+1})^{-1}[m] \overset{m}{*} (\mathcal{P}_{\boldsymbol{\theta}} \{\psi\} * \beta^a) \left(\frac{\lambda_y}{n} m\right),$$

where the matrix of sampling steps, $\boldsymbol{\Lambda}_{\mathbf{y}}$, has been replaced with a scalar, λ_y, because y is now a scalar rather than a vector. In words, this equation states that the sequence $q[m]$, which is needed for orthogonal or oblique interpolation (equations (8) and (9)), comes from a B-spline interpolation of the sequence formed by sampling the continuous convolution between the X-ray projection of the discretization kernel, $\mathcal{P}_{\boldsymbol{\theta}} \{\psi\}$, and a B-spline. The inner convolution will require computing definite integrals of polynomials times sinc. Specifically, for $a = 0$,

$$q[m] = \frac{n}{\lambda_y} \int_{\frac{\lambda_y}{n}(m-1/2)}^{\frac{\lambda_y}{n}(m+1/2)} \mathcal{P}_{\boldsymbol{\theta}} \{\psi\} (y) dy, \tag{10}$$

where the direct B-spline filter does not appear because $(b^1)^{-1} = \delta[m]$, which is the convolutional identity (or, equivalently, because degree-zero B-splines are interpolating). For $a = 1$, $q[m]$ is equal to

$$\frac{n}{\lambda_y} (b^3)^{-1} \overset{m'}{*} \int_{\frac{\lambda_y}{n}(m'-1)}^{\frac{\lambda_y}{n}(m'+1)} \left(1 - \left|\frac{n}{\lambda_y} y - m'\right|\right) \mathcal{P}_{\boldsymbol{\theta}} \{\psi\} (y) dy.$$

For the sinc kernel, the $a = 0$ case is known as the sine integral and can be computed efficiently [25] or stored in a lookup table. As a increases, however, these integrals become unwieldy and slow to compute. This is a problem because we would ideally interpolate with at least $a = 3$.

To avoid this complicated integration, we use oblique projection with $\varphi_1(y) = \beta^0(y)$ and $\varphi_2(y) = \beta^a(y)$. Then we can interpolate with (9) using $q[m]$ from (10) and $d[m] = (b^{a+1})^{-1}[m]$.

C. Reconstruction

We have so far described interpolation approaches to computing the back projection, $H^T g$. But we need to verify these approaches in the context of a full reconstruction system. To do this, we solve the optimization problem

$$\arg\min_{c_f} \|g - H c_f\|_2^2.$$

We do not include a regularization term because we assume we are in the high-quality regime, i.e., low noise and many measurements. We solve this using the steepest descent algorithm [26], which requires computation of $H^T g$ and iterative

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing

applications of $H^T H$. We have already discussed the former, and the latter is accomplished by convolutions with the discrete kernel

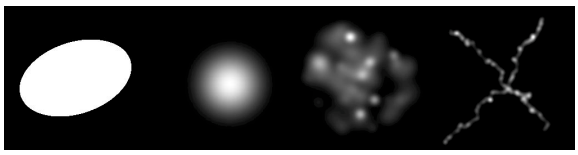
$$h[\mathbf{k}] = (\mathcal{P}_\theta \{\psi\} * \mathcal{P}_\theta \{\psi\} (-\cdot))(\mathbf{P}_{\theta^\perp} \mathbf{\Lambda}_\mathbf{x} \mathbf{k}).$$

For more details on the equivalence of this kernel and the normal operator, see [8]. In the case of the sinc kernel, the autocorrelation is simple to compute in closed form (the autocorrelation of a sinc is another sinc). In our experiments, we initialized the algorithm with an image of all zeros and stopped after 200 iterations, which was more than enough to show convergence empirically.

We also compare the methods using a FBP with Ram-Lak filtering [12]. This is implemented via MATLAB's `iradon` routine modified so as to use our back projectors.

V. Experiments and Discussion

We now present our experiments and discuss the results. Throughout, we compare the exact back projection (3), the standard interpolation (5), and the orthogonal (8) and oblique interpolation (9). We use four analytical phantoms: *spot*, *blob*, *blobs*, and *filament* (Figure 2). These are respectively, a single large ellipse, a single large Kaiser-Bessel window [27], a random collection of 100 small Kaiser-Bessel windows, and two filaments created by placing small Kaiser-Bessel windows along a biased random walk. We generate sinograms from these phantoms according to (1) and, as our focus here is purely on the accuracy of the back projection, we do not add noise. We note that the spot phantom has a large amount of high frequency content because of its sharp edges. The blob phantom is much smoother, though not strictly bandlimited. The blobs and filament phantoms are more complex, while remaining smooth and limited in space.



(a) spot (b) blob (c) blobs (d) filament

Fig. 2. The four phantoms used in our experiments.

A. Speed and Accuracy of Back Projection

In the first experiment (Figures 3, 4, 5, and 6), we compare the speed and accuracy of the methods on back projection only (no reconstruction) in 2D. For each phantom, we fix the sampling steps to be equal in all dimensions, $\mathbf{\Lambda}_\mathbf{x} = \lambda_\mathbf{x} I_d$ and $\mathbf{\Lambda}_\mathbf{y} = \lambda_\mathbf{y} I_{d-1}$, and set $\lambda_\mathbf{x}$ to be an integer multiple of $\lambda_\mathbf{y}$; this sets the density of the measurements with respect to the reconstruction step. We compute the analytical projection from $V = 101$ views (all methods have linear runtime in the number of views; we select this number simply to provide accurate timing results). We then compute the back projection using each of the described methods. We measure the accuracy of the back projection in terms of the signal to noise ratio in dB (SNR, defined as $10 \log_{10}(\sum_\mathbf{k} |g[\mathbf{k}]|^2 / \sum_\mathbf{k} |\tilde{g}[\mathbf{k}] - g[\mathbf{k}]|^2)$, where g is the exact adjoint and $\tilde{g}$ is the estimate).

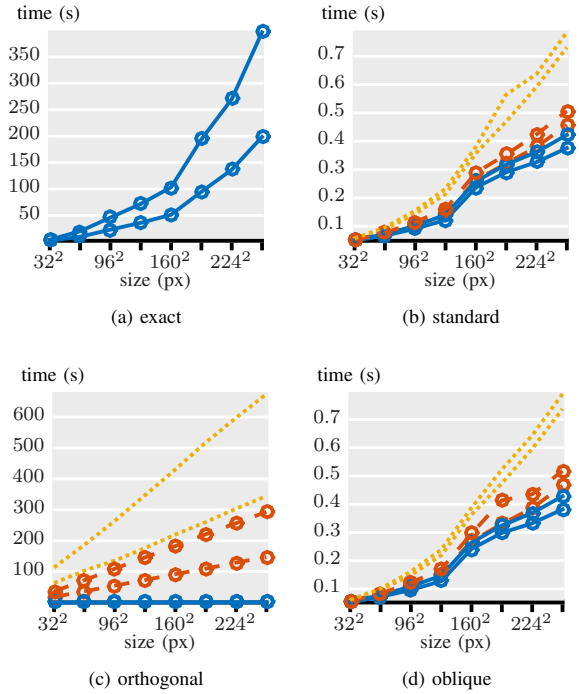


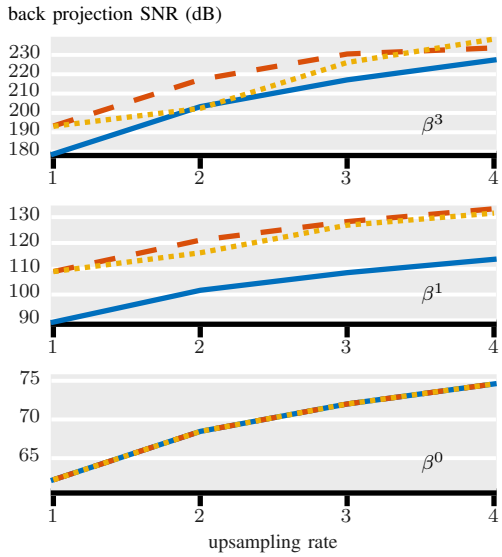
Fig. 3. Timing comparison. Line style indicates B-spline degree (solid, dashed, dotted corresponding to $n = 0,1,3$). For each style, there is a pair of lines: the top and bottom line have measurement densities of one and two, respectively.

To compare the speed of the methods, we compute back projections for upsampling rate 1, reconstruction size 32^2, 64^2, 96^2, ..., 256^2 pixels, measurement density 1 and 2, and spline degree 0, 1, and 3. All computations were performed on the same powerful desktop computer (Mac Pro with two 2.66 GHz 6-Core Intel Xeon processors and 48 GB of RAM) using Matlab. For the exact method, (3) is computed by looping over views and $\mathbf{m}$, with the calculation over $\mathbf{k}$ being vectorized. Both the standard interpolation and oblique interpolation use a loop over views, Matlab's `conv` command for convolutions (experimentally faster than explicitly using FFTs), and compiled C code for spline interpolation and evaluation. The orthogonal interpolation uses the same code as the oblique, except that it also includes numerical integration using Matlab's `integral` command.

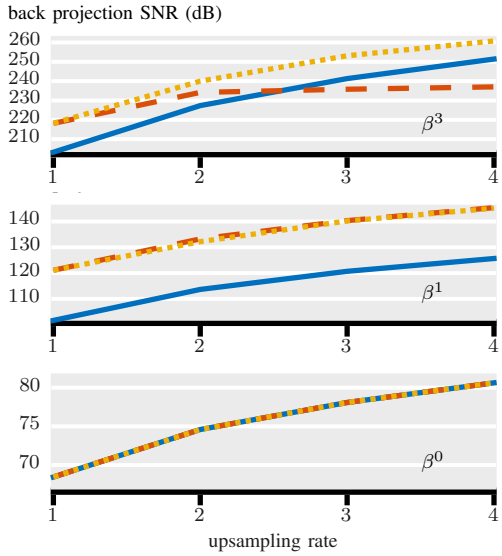
The results (Figure 3) show that the standard and oblique methods are similarly fast and both much faster than the exact method. By fitting a line to the log-log plot of this data, we estimate that the exact method scales as $O(K^n)$ with $n \geq 2.48$, whereas the standard has $n \geq 1.13$ and oblique has $n \geq 1.43$. This is in line with our analysis of (3), which predicts $O(K^3)$ for the exact method and $O(K^2)$ for the other two. The orthogonal method is much slower, except when the spline degree is 0, because it relies on slow numerical approximation of integrals. This also explains why it scales as $O(K)$ in this experiment: the number of integrals is directly proportional to

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing



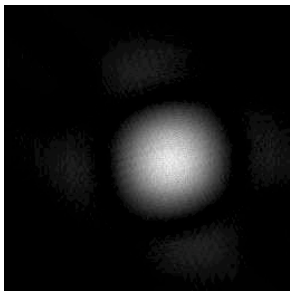
(a) blob, $\lambda_{\mathbf{x}} = \lambda_{\mathbf{y}}$



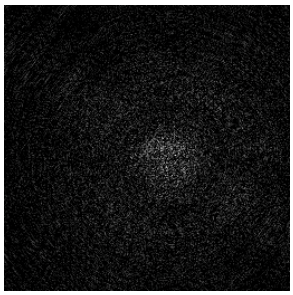
(b) blob, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$

Fig. 4. Back projection accuracy comparison for the blob dataset. Line style indicates method (solid, dashed, and dotted corresponding to standard, orthogonal, and oblique, respectively) and the stacked plots are for spline degrees zero, one, and three (bottom to top). The oblique projection is an excellent approximation of the orthogonal one.

number of measurements. At very large reconstruction sizes, we would expect the interpolation to dominate the runtime, returning it to $O(K^2)$. Increasing the density of measurements proportionally increases the runtime of the exact method, but has much less effect on the standard and oblique methods. Again, this is in line with our analysis because the dominant factor in the runtime of the latter does not include the number of measurement points. Finally, changing the spline degree has the expected effect: the supports are one, two, and four pixels (for B-spline degrees zero, one, and three, respectively) and



(a) standard, 114 dB



(b) oblique, 132 dB

Fig. 5. Back projection error of the standard and oblique interpolation methods for the blob phantom (measurement density two, spline degree one and upsampling rate two). The images are normalized individually to show the distribution of errors. The error of the oblique method is less structured than that of the standard method, which fits the intuition from Figure 1.

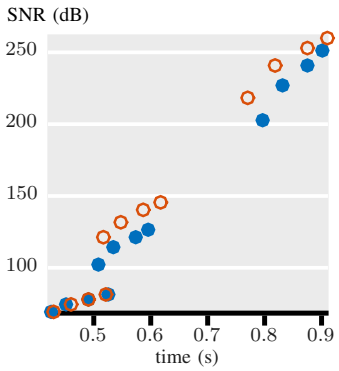


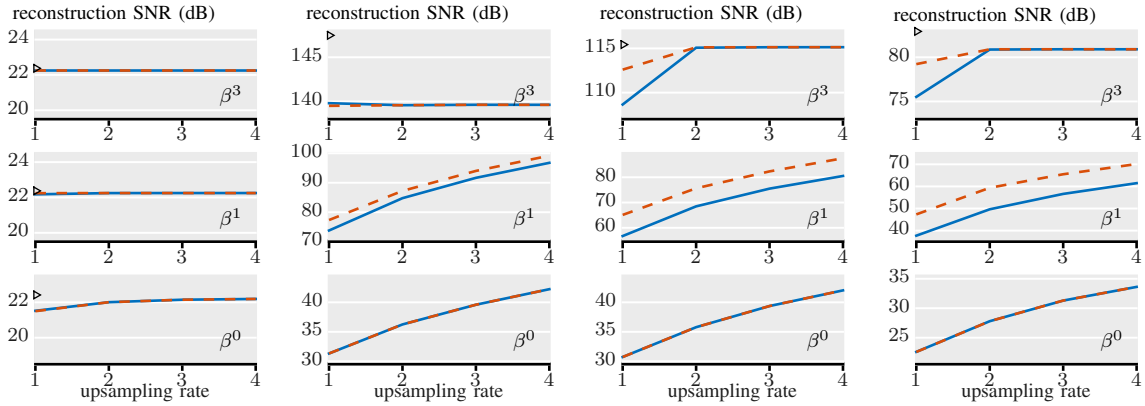
Fig. 6. Back projection accuracy versus time for the standard (filled dots) and oblique (hollow dots) methods for the blob phantom with $\mathbf{\Lambda_x} = 2\mathbf{\Lambda_y}$; up and left is better. The oblique method provides higher SNR with essentially the same runtime as the standard method.

these act as multipliers on the runtime.

To compare the accuracy of the methods, we set the reconstruction size to 65×65 pixels, vary the upsampling rate between 1 and 4, set the measurement density to 1 and 2, and the spline degree to 0, 1, and 3. The results (Figures 4 and 5) on all datasets showed that the orthogonal and oblique methods are nearly identical, except that the orthogonal method's SNR is limited to 240 dB by the accuracy of the numerical integration. For spline degree zero, they also match the standard method, but they generally outperform it when the spline degree is one or three. Varying the dataset affected the performance of all the methods, with the smoothest phantom, blob, giving the best SNR. And, in the same vein, increasing the measurement density improved the SNR, since doing this makes the signal being interpolated more low-pass. Increasing the spline order or increasing the upsampling rate both increase the SNR, but, as is common in interpolation, using a better interpolator (higher-order spline) is much more beneficial than upsampling. For almost all of this range of SNRs, the best time/accuracy trade-off comes from the oblique method (Figure 6).

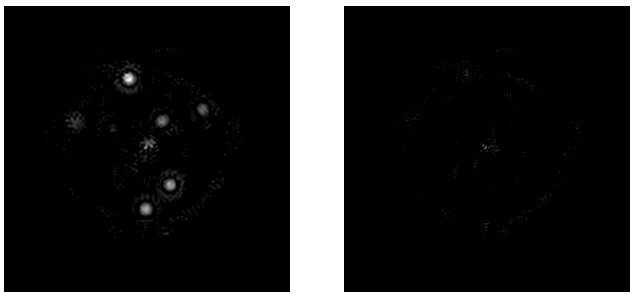
This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing



(a) spot, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (b) blob, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (c) blobs, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (d) filament, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$

Fig. 7. Accuracy comparison for iterative reconstruction. Line style indicates method (solid for standard and dashed for oblique) and the stacked plots are for spline degrees zero, one, and three (bottom to top). The black triangle marks the highest possible SNR in this discretization scheme. The accuracy for the spot phantom saturates immediately, but high quality interpolation is necessary to reach the maximum reconstruction SNR for the other two datasets.



(a) standard, 68 dB (b) oblique, 76 dB

Fig. 8. Absolute reconstruction error for the standard and oblique methods on the blobs phantom (measurement density two, spline degree one, upsampling rate two). The images use the same scale to show the relative size of errors. Error in the back projection propagates to the reconstruction in a complicated way, but, in general, the standard method gives a less faithful reconstruction in the areas where the phantom varies most rapidly.

B. Effect on 2D Reconstruction

We have shown so far that the oblique method consistently outperforms the standard interpolation for computing the back projection, but does this improvement in back projection accuracy translate to an improvement in reconstruction accuracy? This is not a given because the methods have different distributions of errors (Figures 1 and 5) and the reconstruction process may have a complicated effect on these errors.

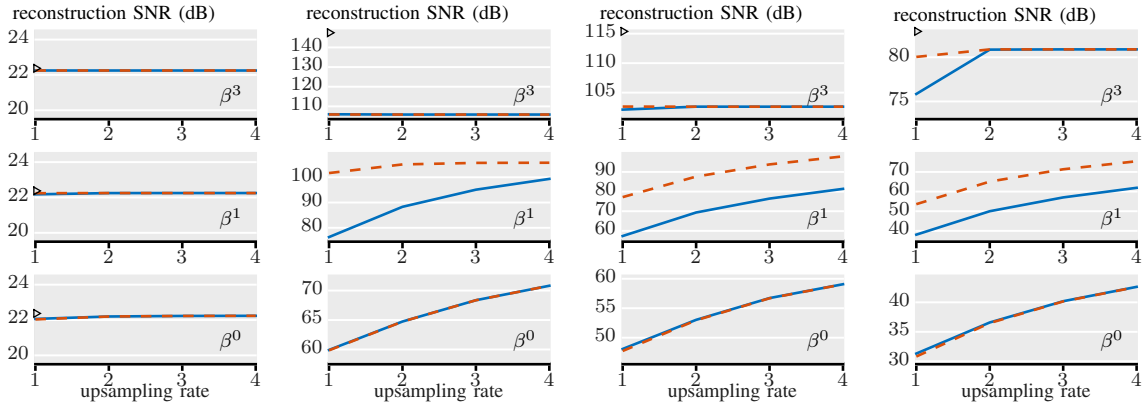
To test this, we compute unregularized reconstructions of the same phantoms with the standard and oblique interpolation methods. We estimate the L_2 (i.e. continuous) accuracy of the reconstruction by calculating the SNR between the reconstruction and the ground truth sampled on a grid six times as fine as the reconstruction. To aid in the interpretation of the results, we also compute the highest SNR achievable using the sinc kernel for each phantom using a separate optimization. We set the number of views to 400, the reconstruction size to 129×129 pixels, and the measurement density to 2. As before, we vary the upsampling rate between 1 and 4 and set the spline degree to 0, 1, and 3.

The results for iterative reconstruction (Figures 7 and 8) show a positive, saturating dependence on the quality of the back projection. Saturation occurs at 23, 148, 116, and 83 dB for the spot, blob, blobs, and filament phantoms, respectively. The low saturation point for the spot phantom is due to its high-pass nature: the sharp edges in these signals cannot be represented well by the sinc kernel (or by any other commonly-used discretization scheme) and therefore the error of the reconstruction is dominated by the error in the discretization. For the other phantoms, we see that using oblique projection increases the SNR of the reconstruction for the middle range of interpolation quality: when either the spline degree is one, or when the spline degree is three and there is no upsampling. This makes sense given that we did not observe oblique projection to increase the adjoint SNR much for splines of degree zero in the previous experiment, and because the reconstruction SNR saturates when using spline degree three plus upsampling.

The results for FBP (Figure 9) show the same trends, except that the SNR saturates at a lower point. That is, iterative reconstruction is more accurate than FBP when the back projection is highly accurate (including the case where the exact back projection is used, experiments not shown). We attribute this difference to the fact that the iterative method explicitly finds the solution most consistent with the measurements, while the FBP simply implements a discretized version of the continuous inverse. Before the SNR saturates, however, the advantages of optimized interpolation are larger for FBP than for iterative reconstruction. We also note that for the blob and blobs phantoms the FBP is actually better than the iterative reconstruction for spline degree 0 and 1 (c.f. Figure 7 and Figure 9). It seems that for these smooth phantoms, FBP is more tolerant to inaccuracy in the back projection calculation than is the iterative scheme.

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing



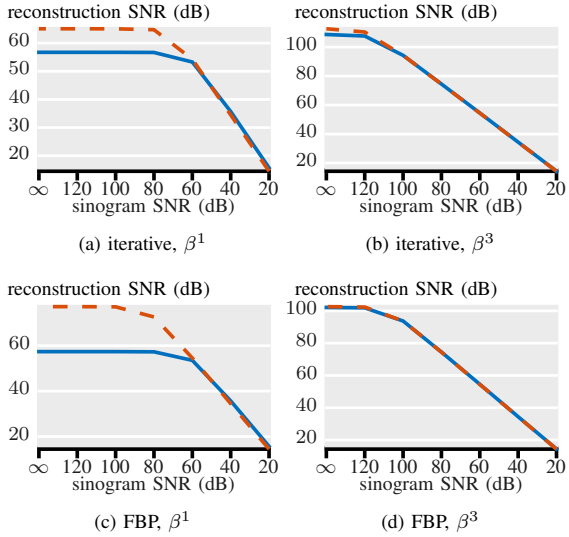
(a) spot, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (b) blob, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (c) blobs, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$ (d) filament, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$

Fig. 9. Accuracy comparison for FBP reconstruction. Line style indicates method (solid for standard and dashed for oblique) and the stacked plots are for spline degrees zero, one, and three (bottom to top).

C. Effect of Noise

We have so far shown experiments where the sinogram is measured without noise. This is useful to highlight the effect of the back projection accuracy, but unrealistic. To explore the interaction between our proposed back projection method and measurement noise, we repeat the 2D reconstruction experiments but with different levels of Gaussian noise added to the sinogram.

We report the results for the blobs dataset in Figure 10. We observe that the SNR of the reconstruction never exceeds the SNR of the sinogram. Conversely, noise in the sinogram has little effect on the reconstruction error when the SNR of the sinogram is above the SNR of the reconstruction. This result puts the 2D reconstruction results in perspective: to see an improvement in reconstruction from using optimized interpolation, the SNR of the sinogram must be above a threshold which varies between 40 and 70 dB, depending on the phantom.



(a) iterative, β^1 (b) iterative, β^3

(c) FBP, β^1 (d) FBP, β^3

Fig. 10. Effect of noise in the sinogram on reconstruction accuracy. Line style indications method (solid for standard and dashed for oblique). The SNR of the reconstruction never exceeds the SNR of the sinogram.

D. Effect on 3D Reconstruction

Finally, we perform a proof-of-concept experiment for 3D reconstruction with a fixed rotation axis. We set the number of views to 200, the reconstruction size to $65 \times 65 \times 65$ pixels, and the measurement density to 2. We calculate the SNR on a grid two times finer than the reconstruction. Based on the results of the previous experiment, we only investigate the blob phantom with spline degree 1 and 3.

The results (Figure 11) again show the same trends: increasing the quality of the interpolation (i.e. using more upsampling or a higher spline degree) improves the reconstruction accuracy up to some limit, which is achieved in our experiments at spline degree three with two times upsampling. Before this limit, using oblique projection improves the reconstruction accuracy.

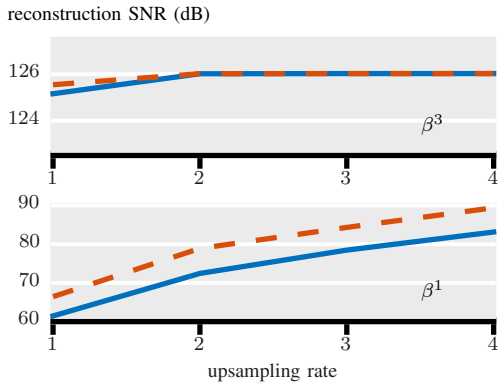
VI. Conclusions

Using interpolation during the back projection step of parallel-ray x-ray CT reconstruction decouples the choice of discretization kernel from the runtime, allowing the use of large discretization kernels with superior approximation properties. In this paper, we took this to the logical extreme by using the sinc kernel, which provides optimal discretization of bandlimited signals. To improve the quality of the interpolation, we proposed using projection-based interpolation, which has been used for image resizing but is new in this context. Our experiments show that using orthogonal projection improves back projection accuracy, but is too slow to be practical for any case except spline degree zero (nearest neighbor interpolation). However, oblique projection provides the same accuracy, but at a computational cost similar to that of standard interpolation.

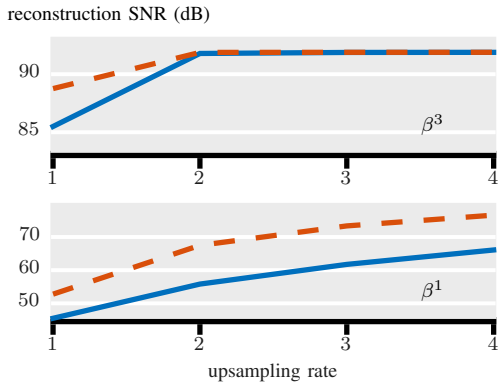
We also show that this improvement in back projection translates into an improvement in reconstruction quality, both for FBP and iterative reconstruction. In practice, this improve-

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2017.2706521, IEEE Transactions on Image Processing



(a) 3D blob, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$



(b) 3D blobs, $\lambda_{\mathbf{x}} = 2\lambda_{\mathbf{y}}$

Fig. 11. Accuracy comparison for iterative reconstruction in 3D. Line style indicates method (solid for standard and dashed for oblique) and the stacked plots are for spline degrees one and three (bottom and top). As in 2D, optimized interpolation improves the SNR of the reconstruction, up to a point of saturation.

ment will be most evident in the high-quality regime. Our experiments show that the SNR of the reconstruction will not exceed the SNR of the sinogram itself, with or without improved back projection. Likewise, if the number of views is too low (below half of the side length of the reconstruction) or the spatial sampling too coarse (over twice the sampling step in the reconstruction), aliasing artifacts will dominate. Barring this, the interpolation error during back projection does limit the SNR of the reconstruction and, for any given time budget or target SNR, it is better to use oblique projection-based interpolation than the standard kind.

References

[1] A. Webb, *Introduction to Biomedical Imaging*. Hoboken, NJ: Wiley-IEEE Press, Dec. 2002.

[2] S. Horbelt, M. Liebling, and M. Unser, "Discretization of the radon transform and of its inverse by spline convolutions," *IEEE Trans. Med. Imag.*, vol. 21, no. 4, pp. 363–376, Apr. 2002.

[3] A. Entezari, M. Nilchian, and M. Unser, "A box spline calculus for the discretization of computed tomography reconstruction problems," *IEEE Trans. Med. Imag.*, vol. 31, no. 8, pp. 1532–1541, Aug. 2012.

[4] B. D. Man and S. Basu, "Distance-driven projection and backprojection in three dimensions," *Phys. Med. Biol.*, vol. 49, no. 11, p. 2463, 2004.

[5] Y. Long, J. Fessler, and J. Balter, "3D forward and back-projection for X-ray CT using separable footprints," *IEEE Trans. Med. Imag.*, vol. 29, no. 11, pp. 1839–1850, Nov. 2010.

[6] F. Momey, L. Denis, C. Burnier, E. Thiebaut, J.-M. Becker, and L. Desbat, "Spline driven: high accuracy projectors for tomographic reconstruction from few projections," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4715–4725, Dec. 2015.

[7] A. Ziegler, T. Khler, T. Nielsen, and R. Proksa, "Efficient projection and backprojection scheme for spherically symmetric basis functions in divergent beam geometry," *Med. Phys.*, vol. 33, no. 12, pp. 4653–4663, Dec. 2006.

[8] M. T. McCann, M. Nilchian, M. Stampanoni, and M. Unser, "Fast 3D reconstruction method for differential phase contrast X-ray CT," *Opt. Express*, vol. 24, no. 13, p. 14564, Jun. 2016.

[9] M. Nilchian, J. Ward, C. Vonesch, and M. Unser, "Optimized Kaiser-Bessel window functions for computed tomography," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3826–3833, Nov. 2015.

[10] M. H. Buonocore, W. R. Brody, and A. Macovski, "A Natural Pixel Decomposition for Two-Dimensional Image Reconstruction," *IEEE Trans. Biomed. Eng.*, vol. BME-28, no. 2, pp. 69–78, Feb. 1981.

[11] F. Natterer and K. P. Hadeler, "Efficient implementation of Optimal algorithms in computerized tomography," *Math. Methods Appl. Sci.*, vol. 2, no. 4, pp. 545–555, Jan. 1980.

[12] A. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*. Society for Industrial and Applied Mathematics, 2001.

[13] P. Cloetens, M. Pateyron-Salom, J. Y. Buffire, G. Peix, J. Baruchel, F. Peyrin, and M. Schlenker, "Observation of microstructure and damage in materials by phase sensitive radiography and tomography," *J. Appl. Phys.*, vol. 81, no. 9, pp. 5878–5886, May 1997.

[14] M. Stampanoni, Z. Wang, T. Thring, C. David, E. Roessl, M. Trippel, R. A. Kubik-Huch, G. Singer, M. K. Hohl, and N. Hauser, "The first analysis and clinical evaluation of native breast tissue using differential phase-contrast mammography," *Invest. Radiol.*, vol. 46, no. 12, pp. 801–806, Dec. 2011.

[15] R. McIntosh, D. Nicastro, and D. Mastronarde, "New views of cells in 3D: an introduction to electron tomography," *Trends Cell Biol.*, vol. 15, no. 1, pp. 43–51, Jan. 2005.

[16] G. Skiniotis and D. R. Southworth, "Single-particle cryo-electron microscopy of macromolecular complexes," *Microscopy (Oxf.)*, vol. 65, no. 1, pp. 9–22, Feb. 2016.

[17] A. Delaney and Y. Bresler, "A fast and accurate Fourier algorithm for iterative parallel-beam tomography," *IEEE Trans. Image Process.*, vol. 5, no. 5, pp. 740–753, May 1996.

[18] M. Unser, A. Aldroubi, and M. Eden, "Enlargement or reduction of digital images with minimum loss of information," *IEEE Trans. Image Process.*, vol. 4, no. 3, pp. 247–258, Mar. 1995.

[19] C. Lee, M. Eden, and M. Unser, "High-quality image resizing using oblique projection operators," *IEEE Trans. Image Process.*, vol. 7, no. 5, pp. 679–692, May 1998.

[20] G. H. Weiss and R. A. Brooks, "Integration Errors in Image Reconstruction of Circularly Symmetric Objects in Computer Assisted Tomography," in *Statistical Mechanics and Statistical Methods in Theory and Application*. Springer US, 1977, pp. 669–684.

[21] T. M. Peters, "Algorithms for Fast Back- and Re-Projection in Computed Tomography," *IEEE Trans. Nucl. Sci.*, vol. 28, no. 4, pp. 3641–3647, Aug. 1981.

[22] M. Unser and P. D. Tafti, *An Introduction to Sparse Stochastic Processes*. Cambridge; New York: Cambridge University Press, Oct. 2014.

[23] P. Thevenaz, T. Blu, and M. Unser, "Interpolation revisited [medical images application]," *IEEE Trans. Med. Imag.*, vol. 19, no. 7, pp. 739–758, Jul. 2000.

[24] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Process. Mag.*, vol. 16, no. 6, pp. 22–38, Nov. 1999.

[25] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge, UK ; New York: Cambridge University Press, Sep. 2007.

[26] J. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Carnegie Mellon University, Tech. Rep., 1994.

[27] R. M. Lewitt, "Multidimensional digital image representations using generalized Kaiser-Bessel window functions," *J. Opt. Soc. Am. A.*, vol. 7, no. 10, pp. 1834–1836, Oct. 1990.

This work is licensed under a Creative Commons Attribution 3.0 License. For more information, see http://creativecommons.org/licenses/by/3.0/.