Discretization of the Radon Transform and of its Inverse by Spline Convolutions

Stefan Horbelt, *Member, IEEE*, Michael Liebling, *Student Member, IEEE*, and Michael Unser*, *Fellow, IEEE*

***Abstract*—We present an explicit formula for B-spline convolution kernels; these are defined as the convolution of several B-splines of variable widths h_i and degrees n_i. We apply our results to derive spline-convolution-based algorithms for two closely related problems: the computation of the Radon transform and of its inverse. First, we present an efficient discrete implementation of the Radon transform that is optimal in the least-squares sense. We then consider the reverse problem and introduce a new spline-convolution version of the filtered back-projection algorithm for tomographic reconstruction. In both cases, our explicit kernel formula allows for the use of high-degree splines; these offer better approximation performance than the conventional lower-degree formulations (e.g., piecewise constant or piecewise linear models). We present multiple experiments to validate our approach and to find the parameters that give the best tradeoff between image quality and computational complexity. In particular, we find that it can be computationally more efficient to increase the approximation degree than to increase the sampling rate.**

***Index Terms*—B-spline convolution kernel, computer tomography, filtered back-projection, Radon transform.**

I. Introduction

THE Radon transform of a two-dimensional (2-D) function is given by the collection of its line-integrals (or projections); each ray is indexed by its distance t to the origin and its angle θ [1], [2]. The Radon transform plays a crucial role in medical imaging because it constitutes a good model of the tomographic acquisition process [3]–[5]. For instance, the forward step—or re-projection—that is required in some iterative reconstruction algorithms is similar to the computation of a Radon transform. Re-projection is also used explicitly for beam-hardening correction [6], streak suppression [7], and the removal of artifacts caused by the presence of radio-opaque material such as metallic implants [8]. Other applications of the Radon transform in image processing are the detection of lines (Hough transform) [9] and the recently proposed ridgelet transform [10], which is essentially a wavelet transform applied in the Radon domain.

An attractive feature of the continuously defined Radon transform is that it has an exact inversion formula [2]. The digital implementation of this analytical formula leads to the standard filtered back-projection (FBP) algorithm, which goes back to the early 1970s [11]. Despite the considerable research efforts devoted to alternative reconstruction techniques—in particular, algebraic (ART) [4] and statistical ones, including maximum likelihood [12] and Bayesian [13]–[16]—FBP is still the method of choice used in commercial CT scanners. It owes its success to the fact that it is direct, fast, and reasonably simple to implement. Even though the standard implementation uses a rather rudimentary discretization—at least by modern standards, it has not been much improved over the years, except for the aspect of filter design [17]. One noteworthy exception is the work of Guédon *et al.* who derived an optimal reconstruction filter based on a piecewise-constant model of the image [18]. Some wavelet approaches can also be viewed as multi-scale variations on FBP [19]–[21]. In this paper, we are not dealing with the problem of noise which is better treated statistically. However, even in the presence of noise it makes good sense to apply a good algorithm for noiseless data first and to reduce noise afterwards by suitable nonlinear wavelet denoising; Kalifa *et al.* [22] have shown that this two-step approach is essentially as good as a regularized reconstruction during FBP and much easier to implement.

The practical computation of the Radon transform or of its inverse necessarily involves some form of interpolation because the underlying mathematical objects are defined in the continuous domain. The same holds true for Fourier-based reconstruction techniques [23]–[25]. For efficiency, practitioners tend to use relatively simple techniques such as piecewise constant or linear interpolation. One of the arguments that is often made in favor of FBP is that it is less sensitive to interpolation errors than direct Fourier-based reconstruction [26], [27]. Thus, there appears to be a general perception that the role of interpolation is not predominant. Our purpose in this paper is to investigate this issue in greater details and determine the extent to which the use of high quality interpolation models together with an optimal discretization can improve performance. Indeed, we will see that a careful design—i.e., the use of a good model and a sampling method that is optimal in the least-squares sense—can make quite a difference. We have chosen to base our approach on splines because this type of representation offers the best cost-performance tradeoff for interpolation; this is a finding that was confirmed recently by three research groups in medical imaging [28]–[31].

The novelty of our approach, which uses splines both in the image and the Radon domains, is fourfold: first, our Radon kernel is applicable in both directions for the computation of the Radon transform and of its inverse—in fact, the direct and the inverse algorithms are duals of each other. Second, we explicitly

Manuscript received November 7, 2000; revised January 22, 2002. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was C. Crawford. *Asterisk indicates corresponding author.*

S. Horbelt and M. Liebling are with the Biomedical Imaging Group, IOA, STI, Swiss Federal Institute of Technology Lausanne, CH-1015 Lausanne EPFL, Switzerland (e-mail: horbelt@ieee.org; stefan@horbelt.com).

*M. Unser is with the Biomedical Imaging Group, IOA, STI, Swiss Federal Institute of Technology Lausanne, CH-1015 Lausanne EPFL, Switzerland (e-mail: michael.unser@epfl.ch).

Publisher Item Identifier S 0278-0062(02)04691-8.

fit a continuously defined spline model to the sinogram (respectively, to the image), instead of determining the values by resampling, as is done usually. Third, we select the spline basis function in a way that allows for an explicit solution with B-spline convolution kernels and, therefore, an optimal discretization of the Radon transform. Finally, the method is general and works for splines of any degree n. Choosing a larger degree (typically, $n = 3$) results in better quality, but it also implies more computation. Thus, there is a compromise to be found.

The paper is organized as follows: In Section II, we review classical B-splines and derive explicit formulas for the multiple B-spline convolution kernels that will be used in our algorithms. In Section III, we briefly recall the basics of B-spline signal processing and generalized sampling. In Section IV, we use these results to derive our spline-based implementation of a Radon transform which is optimal in the least-squares sense. In Section V, we introduce a Radon-based version of a FBP algorithm that also uses spline kernels. In Section VI, we discuss implementation issues for both algorithms (direct and inverse). Finally, in Section VII, we present experimental results for each of the algorithms. In particular, we investigate the influence of the various parameters (degrees of the splines, spatial and angular sampling steps) on the overall performance. We determine the best tradeoff between computation time and image quality.

While we believe that our spline-formulation of FBP and the Radon transform is original, we are aware of three other instances where splines have been used for tomographic reconstruction. The first is a finite-element formulation of the problem using splines as basis functions [32]; it is essentially an algebraic method that requires solving a large system of linear equations. The second applies box splines—a multidimensional extension of B-splines—for improved approximation in an algorithm based on Fourier reconstruction [33]. The third is an approach where the ramp-filtering is evaluated analytically based on the continuously defined derivative of the projection data [34]. Finally, it is interesting to note that the benefit of bi-cubic interpolation—albeit not splines—has been demonstrated indirectly in rotation-based methods for iterative reconstruction [35].

II. B-Spline Convolution Kernels

A. The B-Spline Function

We define the centered B-spline of degree n and of width h as the $(n+1)$fold convolution of the normalized box function of size h

$$\beta_h^0(x) = \begin{cases} \frac{1}{h}, & \text{for} -\frac{h}{2} \le x < \frac{h}{2} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$\beta_h^n(x) = \beta_h^0 * \beta_h^{n-1}(x) = \underbrace{\beta_h^0 * \cdots * \beta_h^0}_{n+1 \text{ terms}}(x). \tag{2}$$

Note that these functions are normalized to have a unit integral; this implies that $\beta_h^n(x) = (1/h)\beta_1^n(x/h)$.

Let us also define the one-sided power function

$$x_+^n = \begin{cases} x^n, & x \ge 0 \text{ and } n > 0 \\ 1, & x \ge 0 \text{ and } n = 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

and the centered $(n+1)$th finite-difference operator scaled at width h

$$\Delta_h^{n+1} = \underbrace{\Delta_h^1 * \cdots * \Delta_h^1}_{n+1 \text{ terms}} = \sum_{k=0}^{n+1} (-1)^k \binom{n+1}{k} \frac{\delta(x + h \cdot (\frac{n+1}{2} - k))}{h^{n+1}}. \tag{4}$$

Using this notation, β_h^0 can also be written as the central finite difference of the Heaviside step function x_+^0

$$\beta_h^0 = \Delta_h^1 * x_+^0 = \frac{1}{h}\left(\left(x + \frac{h}{2}\right)_+^0 - \left(x - \frac{h}{2}\right)_+^0\right).$$

Thus, we may rewrite β_h^n in terms of the following convolution operators:

$$\beta_h^n = \beta_h^0 * \cdots * \beta_h^0 = \left(\Delta_h^1 * x_+^0\right) * \cdots * \left(\Delta_h^1 * x_+^0\right).$$

Using the commutativity of convolution operators, we get

$$\beta_h^n = \underbrace{\Delta_h^1 * \cdots * \Delta_h^1}_{n+1 \text{ terms}} * \underbrace{x_+^0 * \cdots * x_+^0}_{n+1 \text{ terms}}$$

which, as shown in the first part of Appendix I, can be expressed in the more succinct form

$$\beta_h^n(x) = \Delta_h^{n+1} * \frac{x_+^n}{n!} \tag{5}$$

where Δ_h^{n+1} and x_+^n are given by (4) and (3), respectively.

Yet another way to define the centered B-spline is by its Fourier transform

$$\widehat{\beta}_h^n(\omega) = \text{sinc}^{n+1}\left(\frac{\omega h}{2\pi}\right)$$

where $\text{sinc}(x) = \sin(\pi x)/\pi x$.

B. B-Spline Convolution Kernels

To implement the Radon transform, we will need to convolve several B-splines of variable widths (see Figs. 1 and 2). The spline *bikernel* $\beta_{h_1,h_2}^{n_1,n_2}(x)$ is defined as the convolution of two B-splines of degrees n_1, n_2 and widths h_1, h_2

$$\beta_{h_1,h_2}^{n_1,n_2}(x) = \beta_{h_1}^{n_1} * \beta_{h_2}^{n_2}(x). \tag{6}$$

In the same way, we define the spline m-kernel, which is the convolution of m different B-splines of degrees $n_1, \ldots, n_m$ and widths $h_1, \ldots, h_m$, by

$$\beta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m}(x) = \beta_{h_1}^{n_1} * \cdots * \beta_{h_m}^{n_m}(x).$$

Proposition 1: The spline m-kernel can be computed as

$$\beta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m}(x) = \Delta_{h_1,\ldots,h_m}^{n_1+1,\ldots,n_m+1} * \frac{x_+^{N_m}}{N_m!} \tag{7}$$

where $\Delta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m} = \Delta_{h_1}^{n_1} * \cdots * \Delta_{h_m}^{n_m}$ is the convolution of several finite-difference operators as defined in (4) and where $N_m = m - 1 + \sum_{i=1}^m n_i$.

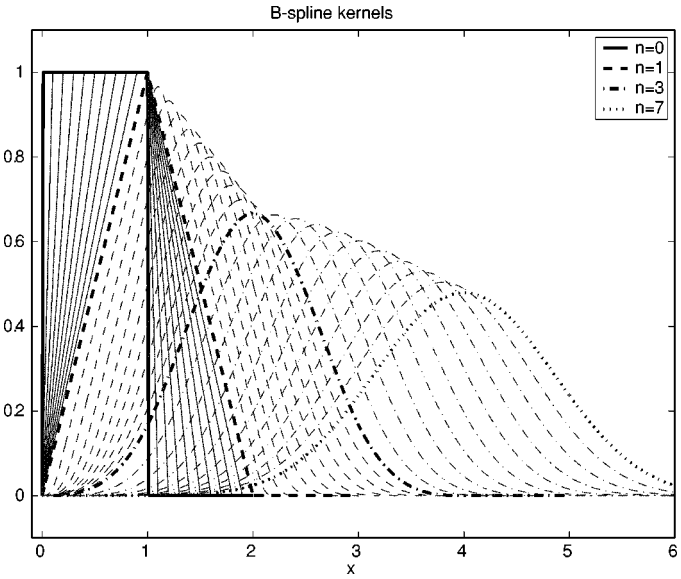


Fig. 1. Spline bikernels by convolution of two causal B-splines $\beta_1^n * \beta_h^n(x)$ with $n = \{0, 1, 3\}$ and $h \in [0.1]$. The kernels generate a smooth transition between B-splines of degree n and degree $2n + 1$. For $h = 0$ and $h = 1$, the kernels are B-splines, as $\lim_{h\to 0} \beta_1^n * \beta_h^n(x) = \beta_1^n(x)$ and $\beta_1^n * \beta_1^n(x) = \beta_1^{2n+1}(x)$.

The spline m-kernel is a nonuniform spline of degree N_m and its support is the sum of the supports of the convolved B-splines

$$\text{supp}\left(\beta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m}(x)\right) = \left[-\sum_{i=1}^{m}\frac{h_i(n_i+1)}{2}, \sum_{i=1}^{m}\frac{h_i(n_i+1)}{2}\right]. \tag{8}$$

The proof of Proposition 1 is given in Appendix I. □

In the case where the h_i are equal, the compound finite difference operator simplifies to

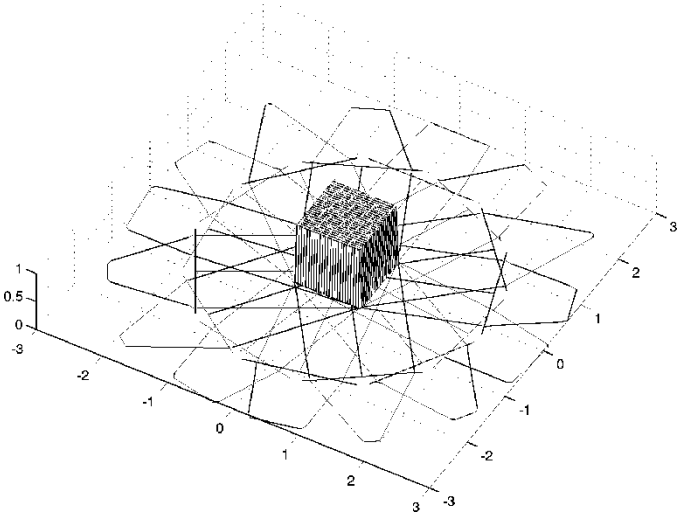
$$\Delta_{h,\ldots,h}^{n_1+1,\ldots,n_m+1} = \Delta_h^{N_m+1}. \tag{9}$$

We can use (7) and (9) to derive the well-known convolution property of two B-splines

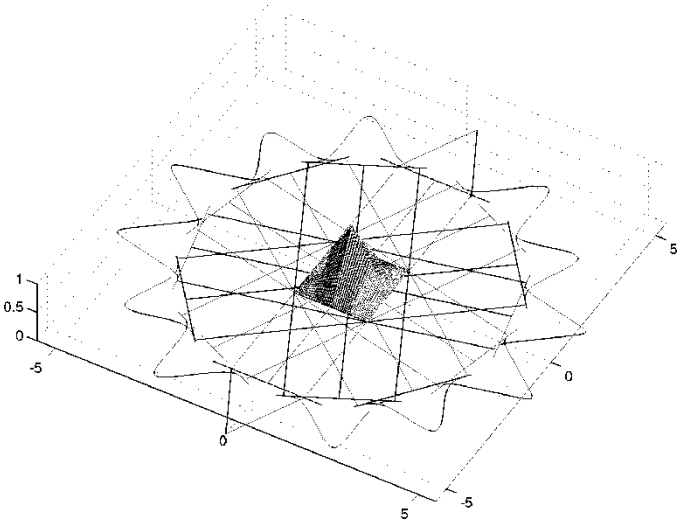
$$\beta_h^{n_1} * \beta_h^{n_2}(x) = \Delta_h^{n_1+n_2+2} * \frac{x_+^{n_1+n_2+1}}{(n_1+n_2+1)!} = \beta_h^{n_1+n_2+1}(x).$$

The spline m-kernel formula (7) expanded by (4) leads to a closed-form expression. For instance, with $m = 2$, we get the following explicit formula for the spline *bikernel*

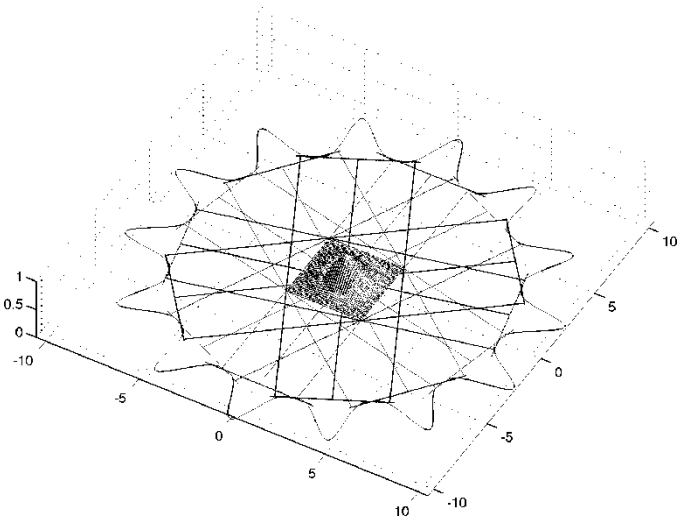
$$\begin{aligned}\beta_{h_1,h_2}^{n_1,n_2}(x) =& \Delta_{h_1,h_2}^{n_1+1,n_2+1} * \frac{x_+^{n_1+n_2+1}}{(n_1+n_2+1)!} \\ =& \sum_{k_1=0}^{n_1+1}\sum_{k_2=0}^{n_2+1}(-1)^{k_1+k_2}\binom{n_1+1}{k_1}\binom{n_2+1}{k_2} \\ & \cdot \frac{\left(x+\left(\frac{n_1+1}{2}-k_1\right)\cdot h_1+\left(\frac{n_2+1}{2}-k_2\right)\cdot h_2\right)_+^{n_1+n_2+1}}{(n_1+n_2+1)!\cdot h_1^{1+n_1}\cdot h_2^{1+n_2}}.\end{aligned} \tag{10}$$



(a)



(b)



(c)

Fig. 2. Spline flowers. The Radon transform of the 2-D B-spline $\beta^n(x)\cdot\beta^n(y)$ is a B-spline convolution kernel. We show several examples. They are plotted on the floor, for various projection angles θ. At angles that are multiples of $\pi/4$, the B-spline convolution kernel simplifies to an ordinary B-spline. (a) $n = 0$. (b) $n = 1$. (c) $n = 3$.

III. Least-Squares Sampling

Continuous signals have to be sampled into discrete signals before they can be processed digitally. The spline formalism described in [36] provides an attractive alternative to the classical formulation dictated by Shannon's sampling theorem [37]. The main idea is to use compactly supported basis functions—B-splines—which are much more convenient to handle than the ideal sinc function which decays rather slowly. A good reason for using polynomial splines over other forms of interpolation is that they offer the best cost-performance tradeoff; this is a property that is now well documented in the literature. In particular, we refer to the extensive work of Meijering [30] who compared as many as 126 interpolators and concluded that splines were significantly better in all cases. One theoretical explanation for their superior performance is that B-splines are the shortest and smoothest functions that allow the reproduction of polynomials of degree n. This polynomial reproduction property is essential in wavelet and approximation theory; it determines the approximation order ($L = n+1$); i.e., the rate of decay of the approximation error as a function of the sampling step h [38], [39].

A. Polynomial Splines

Here, we will consider polynomial splines of degree n defined on a uniform grid with step size h. These functions are made up of polynomial segments of degree n and of size h which are joined together in a way that guarantees the continuity of the function and its derivatives up to order $n-1$. A fundamental result is that any such spline with spacing h can be represented in terms of its B-spline expansion [40]

$$f_h(x) = \sum_{k\in\mathbb{Z}} c_k \beta_h^n(x - hk) \tag{11}$$

where the basis functions $\{\beta_h^n(x-hk)\}$ are the shifted versions of the B-splines at scale h defined in Section II. The c_k are the so-called B-spline coefficients; note that there is exactly one such coefficient per grid point. The same type of representation also applies in higher dimensions with separable (tensor product) B-splines.

Interestingly, there are also spline basis functions other than the B-splines that yield a representation similar to (11). Any linear combination of the B-splines $\varphi_h(x) = \sum_{k\in\mathbb{Z}} p(k)\beta_h^n(x-k)$ is acceptable as long as $\sum_{k\in\mathbb{Z}} p(k) = 1$ and $0 < m \le \left|P(e^{j\omega})\right|^2 \le M < +\infty$ [42]. This ensures that the spline function $\varphi = \varphi_{h=1}$ fulfills the partition of unity $\sum_{k\in\mathbb{Z}} \varphi(x-k) = 1$ and that it is a Riesz basis [41].

The basic question in B-spline signal processing is how to determine the expansion coefficients in (11). There are essentially two approaches: interpolation and least-squares approximation.

B. Spline Interpolation

This is the approach of choice when the signal is represented by its samples $f(hk)$. It is then possible to determine the c_k such that the model (11) interpolates the samples $f(hk)$ exactly. This involves the solution of a linear system of equations. This problem is solved most effectively using the recursive filtering algorithm described in [36] and [43].

By applying this spline interpolation to a Kronecker delta (unit impulse at origin), one obtains the so-called cardinal spline function which is denoted by $\varphi(x) = \eta^n(x)$. It has the interesting property that the cardinal spline coefficients are simply the sampled values of the continuous function: $c_k = f(hk)$. It is an interpolator of infinite support which resembles the sinc function [44]. This is another example of a valid spline basis functions. With this particular basis function in mind, one can interpret the B-spline interpolation algorithm (recursive filtering) as a change from the cardinal spline basis (where the samples are the coefficients) to the B-spline basis.

C. Least-Squares Spline Approximation

This is a refinement on straightforward sampling; it is applicable whenever the input function $f(x)$ is continuously defined. Here, the goal is to get the best spline representation $f_h(x)$ of $f(x)$ so that the integral of the square difference between $f(x)$ and $f_h(x)$ (L_2-norm) is minimized. Mathematically, this corresponds to the orthogonal projection of $f(x)$ onto the spline space with step size h.

The optimum expansion coefficients are determined from the inner product with the dual basis functions (cf. [41])

$$c_k = h \cdot \langle f(x), \tilde{\varphi}_h(x - hk)\rangle\,. \tag{12}$$

The basis function $\varphi(x)$ and its dual $\tilde{\varphi}(x)$ (which is unique) form a bi-orthogonal basis system that fulfills the bi-orthogonality condition $\langle \varphi(x), \tilde{\varphi}(x-k)\rangle = \delta_k$. Note that the computation of the inner products (12) is akin to applying a prefilter to $f(x)$ and sampling thereafter. The main advantages of least-squares approximation over interpolation-and-sampling are a closer fit and a suppression of aliasing artifacts.

Interestingly, the role of basis functions and duals can be interchanged. This means that by performing the inner product between $f(x)$ and the B-splines $\{\beta_h^n(x-hk)\}$, one obtains the least-squares coefficients of $f_h(x)$ in terms of the dual B-splines $\{\tilde{\beta}_h^n(x-hk)\}$. This is a property that will be used in our algorithm because we know how to compute inner products with B-splines analytically. As the B-splines and the dual B-splines span the same space, it is then possible to go back to the B-spline basis by performing a change of basis, which is equivalent to a digital filtering operation [45].

Next, we will apply these principles to the discretization of the Radon transform and show how a suitable choice of target and source basis functions allows to perform these calculations exactly using the B-spline convolution kernels defined in Section II-B.

IV. Spline-Based Radon Transform

A. Radon Transform

The *Radon transform* $R_\theta f$ [1], [2] of an image $f(\vec{x})$, $\vec{x} \in \mathbb{R}^2$, is the set of line integrals along the direction $\vec{\theta}$ at the distance t from the origin

$$(R_\theta f)(t) = R_\theta\{f(\vec{x})\} = \int_{\vec{x}\in\mathbb{R}^2} f(\vec{x})\delta\left(t - \vec{x}^{\top}\cdot\vec{\theta}\right) d\vec{x} \tag{13}$$

where $\delta(t)$ is the Dirac impulse and $\vec{\theta} = (\cos\theta, \sin\theta)$ is a unit vector specifying the direction of the integration. Note that R_θ is called the *projection operator*, because it is the Radon transform for one particular projection angle θ; R on its own stands for the complete Radon transform with all angles θ.

The Radon transform is clearly a linear operator. Thus, if we represent the initial image as a sum of 2-D basis functions (B-splines), we can compute its Radon transform exactly provided we know the Radon transform of the initial basis functions. Moreover, if we know the Radon transform of the one prototype $\varphi(\vec{x})$, we can easily derive the Radon transform of all basis functions because of the shift-invariance property $R_\theta\{\varphi(\vec{x} - \vec{x}_0)\} = (R_\theta\varphi)(t - t_0)$ where $t_0 = \left\langle \vec{\theta}, \vec{x}_0 \right\rangle$.

B. Radon Transform of a B-Spline

We now consider the case where the basis function φ is a separable B-Spline.

Proposition 2: Radon Transform of the 2-D B-spline: The Radon transform of the 2-D separable B-spline $\beta_h^n(x,y) = \beta_h^n(x) \cdot \beta_h^n(y)$ of degree n is

$$R_\theta \beta_h^n(t) = \beta_{h|\cos\theta|}^n * \beta_{h|\sin\theta|}^n(t) = \beta_{h|\cos\theta|,h|\sin\theta|}^{n,n}(t) \quad (14)$$

where $t = \vec{x}^\top\vec{\theta}$ and $\vec{\theta} = (\cos\theta, \sin\theta)$ with the projection angle θ; it is precisely a spline bikernel whose explicit form is given by (7) or (10).

The proof of Proposition 2 is easily derived using the Fourier Slice Theorem (see appendix II). □

Fig. 2 shows the Radon transform of the 2-D B-splines of degrees $n \in \{0,1,3\}$ and for various projection angles θ. In the cases where $\cos\theta = 0$ or $\sin\theta = 0$, the Radon transform of the 2-D B-spline reduces to a one–dimensional (1-D) B-spline of the same degree n, as $\lim_{h\to 0} \beta_h^n(x) = \delta(x)$. For angles where $|\cos\theta| = |\sin\theta| = 1/\sqrt{2}$, the Radon transform simplifies to a 1-D B-spline $\beta_{h/\sqrt{2}}^{2n+1}(x)$ of degree $2n+1$. For the angles in between, the spline bikernels range between these two extremes (compare Figs. 1 and 2).

C. Radon Transform of the Image

Let us now assume that the input image $f(\vec{x})$ is represented by a polynomial spline as a sum of shifted B-spline basis functions of degree n_1

$$f_h(\vec{x}) = \sum_{k,l\in\mathbb{Z}} c_{k,l}\beta_h^{n_1}\left(\vec{x} - h\vec{k}\right). \quad (15)$$

In practice, the B-spline coefficients $c_{k,l}$ are specified such that the model (15) interpolates the given pixel values exactly, as described in Section III-B. By using (14), (15), and the linearity of the Radon transform, we obtain the analytical expression of its projection at angle θ—the *sinogram* $g_\theta(t)$

$$\begin{aligned} g_\theta(t) =& R_\theta f_h(t) = \sum_{k,l\in\mathbb{Z}} c_{k,l} R_\theta \beta_h^{n_1}(t - h\vec{k}^\top\vec{\theta}) \\ =& \sum_{k,l\in\mathbb{Z}} c_{k,l}\beta_{h|\cos\theta|,h|\sin\theta|}^{n_1,n_1}(t - h\vec{k}^\top\vec{\theta}). \end{aligned} \quad (16)$$

D. Least-Squares Discretization

The final step is to discretize the continuously defined expression of the Radon transform (16). The simplest approach would be to sample $g_\theta(t)$ at an appropriate sampling step s. Here, we want to be more sophisticated and approximate $g_\theta(t)$ by a polynomial spline of degree n_2 to minimize the approximation error in the L_2-sense. The solution is to compute the orthogonal projection as described in Section III-C. We denote this operation by P_s.

In order to make use of our spline convolution kernels, it is now appropriate to use a representation in terms of dual B-spline functions such that the analysis function is a B-spline. In other words, we now evaluate the best approximation $g_{\theta,s}(t)$ of the sinogram $g_\theta(t)$ as

$$g_{\theta,s}(t) \equiv P_s g_\theta(t) = \sum_{i\in\mathbb{Z}} \underbrace{s\left\langle g_\theta(t), \beta_s^{n_2}(t - is)\right\rangle}_{\tilde{c}_{i,\theta}} \cdot \tilde{\beta}_s^{n_2}(t - is) \quad (17)$$

where $g_\theta(t)$ is given by (16).

Here, too, the operator is linear. Thus, to evaluate the inner product we look at what happens to one of the individual terms in (16)

$$P_s\{\beta_{h|\cos\theta|,h|\sin\theta|}^{n_1,n_1}\}\left(t - h\vec{k}^\top\vec{\theta}\right) = \sum_{i\in\mathbb{Z}} s \cdot d_{i,\theta,k,l} \cdot \tilde{\beta}_s^{n_2}(t - is) \quad (18)$$

where the coefficients $d_{i,\theta,k,l}$ are sampled values of the *Radon kernel*

$$d_{i,\theta,k,l} = \langle \beta_{h|\cos\theta|,h|\sin\theta|}^{n_1,n_1}(t - h\vec{k}^\top\vec{\theta}), \beta_s^{n_2}(t - is)\rangle_t.$$

This scalar product over the variable t is equivalent to the convolution of the two scalar product factors evaluated at the cumulated shift $h\vec{k}^\top\vec{\theta} - is$. The Radon kernel is a spline trikernel

$$d_{i,\theta,k,l} = \beta_{h|\cos\theta|,h|\sin\theta|,s}^{n_1,n_1,n_2}(h\vec{k}^\top\vec{\theta} - is). \quad (19)$$

The identification of the *Radon kernel* is the key to our approach. It allows for an efficient evaluation of the Radon transform and yields an exact implementation of the least-squares algorithm. The logical notation for the trikernel is (n_0, n_1, n_2)-trikernel, where the first two integers stand for the spline degree in the x and y directions of the image space and the last integer for the spline degree of the sinogram space. In our case, the two degrees of the image space are equal; therefore, we simplify the notation to (n_1, n_2)-trikernel, where the degrees refer to (19).

Putting everything together, we find that the least-squares coefficients $\tilde{c}_{i,\theta}$ in (17) are given by

$$\tilde{c}_{i,\theta} = \sum_{k,l\in\mathbb{Z}} d_{i,\theta,k,l} \cdot c_{k,l} \cdot s$$

where s is the sampling step in the sinogram, h is the sampling step in the image, $c_{k,l}$ are the image coefficients (15), and $d_{i,\theta,k,l}$ are the sampled values of the Radon kernels (19). Note that the spatial summation in (17) only extends over the domain for which the trikernel is nonzero.

E. Analysis of Least-Squares Approximation Error of the Radon Transform

The above least-squares solution is usually not exact and will introduce some approximation error depending on the sampling step s. The Radon transform of a 2-D B-spline $\beta^{n_1}(x)\cdot\beta^{n_1}(y)$ of degree n_1 is a spline bikernel $\beta^{n_1,n_1}_{|\sin\theta|,|\cos\theta|}$ which is a nonuniform polynomial of degree $n_2 = 2n_1 + 1$. We compute the approximation error of the orthogonal projection f_s of the spline bikernel $f(t) = \beta^{n_1,n_1}_{|\sin\theta|,|\cos\theta|}(t)$ in the spline space with sampling step s using the general formula given in [38]

$$\varepsilon^2(s) = \|f - f_s\|^2_{\mathcal{L}_2} = \frac{1}{2\pi}\int_{-\infty}^{\infty} E(s\omega)\left|\hat{f}(\omega)\right|^2 d\omega \qquad (20)$$

where $\hat{f}(\omega)$ is the Fourier transform of the spline bikernel

$$\hat{f}(\omega) = |\sin\theta|\cdot \mathrm{sinc}^{n_1+1}\left(\frac{\omega\cdot|\sin\theta|}{2\pi}\right) \cdot |\cos\theta|\cdot \mathrm{sinc}^{n_1+1}\left(\frac{\omega\cdot|\cos\theta|}{2\pi}\right)$$

and $E(\omega)$, the error kernel, is given by

$$E(\omega) = 1 - \frac{\left|\hat{\beta}^{n_2}(\omega)\right|^2}{\sum_k \left|\hat{\beta}^{n_2}(\omega + 2k\pi)\right|^2}$$

with $\widehat{\beta}^{n_2}(\omega) = \mathrm{sinc}^{n_2+1}\left(\frac{\omega}{2\pi}\right)$. The approximation error depends on four parameters: the degree of the image space n_1, the degree in the Radon space n_2, the sinogram sampling step s, and the projection angle θ. Obviously, the approximation error can be decreased by increasing the spline degrees and by reducing the sampling step s (see Fig. 3), but at the expense of more computation complexity (see Section VI-D). Note that in Fig. 3, the kernels with a higher degree on the sinogram have lower errors. Especially the (1,3)-kernel and the (2,5)-kernel have local error minima at sampling steps $s = 1/2$ and $s = 1/4$.

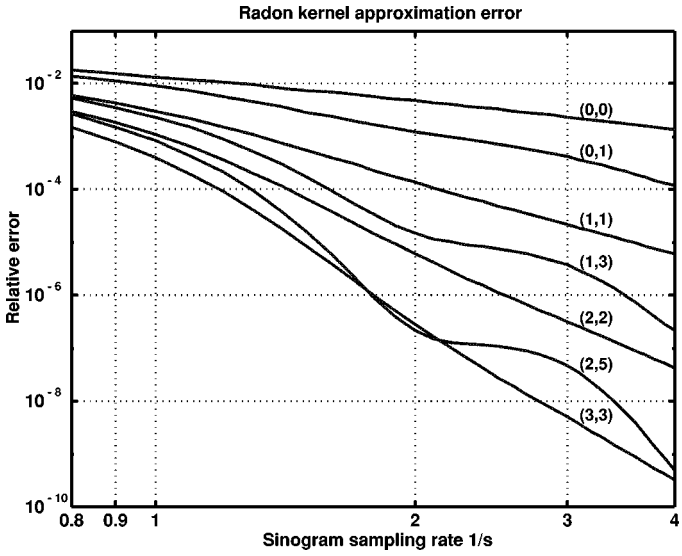


Fig. 3. Approximation error (20) for various Radon kernel degrees and sinogram sampling rates.

V. Spline-Based Filtered Back-Projection

We will now see that we can use our B-spline convolution kernels also to compute the inverse of the Radon transform. This yields a refined version of the FBP algorithm which is used for the tomographic reconstruction.

A. Filtered Back-Projection

The basis for the inverse Radon transform is the well-known identity (cf. [2])

$$f(\vec{x}) = \left(R^*(q * Rf)\right)(\vec{x}), \qquad (21)$$

where each projection $R_\theta f$ is convolved by a one-dimensional (1-D) ramp filter q defined in the Fourier domain by $\hat{q}(\omega) = |\omega|$. The back-projection operator R^* is the adjoint of R

$$\left(R^* g_\theta\right)(\vec{x}) = R^*\left\{g_\theta(t)\right\} = \int_0^\pi g_\theta\left(\vec{x}^\top\cdot\vec{\theta}\right)d\theta$$

where $g_\theta(t) = R_\theta f(t)$ is the sinogram line at angle θ. The widely used filtered back-projection (FBP) algorithm corresponds to the direct discretization of the right-hand side of the inversion formula (R^*q) [11].

B. Spline-Convolution Filtered Back-Projection

The spline-convolution filtered back-projection implements the inverse of the spline-convolution Radon transform. Except for the additional ramp prefiltering, the inverse method is a flow-graph transpose of the forward method. Because both the image and the sinogram are modeled by splines, we can again base the algorithm on the evaluation of the Radon kernel. This time, the filtered sinogram is approximated in the B-spline space, while the resulting image is in the dual-spline space. In the following, we give the necessary formulas for the three steps of our method.

Step1') Ramp filter: In the first step, each sinogram line $\hat{g}_\theta(\omega) = \mathcal{F}_{1\mathrm{D}}\{g_\theta(t)\}$ is filtered in the Fourier domain by $\hat{q}(\omega)$

$$\hat{p}_\theta(\omega) = \hat{g}_\theta(\omega)\cdot\hat{q}(\omega).$$

Step 2') B-spline approximation of the sinogram: In the second step, the filtered sinogram $p_\theta(t) = \mathcal{F}^{-1}_{1\mathrm{D}}\{\hat{p}_\theta(\omega)\}$ is filtered with a spline of step size s

$$p_{\theta,s}(t) = \sum_{i\in\mathbb{Z}} c_{\theta,i}\cdot\beta^{n_2}_s(t - is). \qquad (22)$$

This can be done either using interpolation or projection (cf. Section III).

Step 3') Back-projection into the dual-spline space: In the third step, the back-projection $R^*_\theta\{p_{s,\theta}(t)\}$ is calculated and approximated in the image space, using dual-splines as basis functions

$$\tilde{f}_h(\vec{x}) = \sum_{k,l\in\mathbb{Z}} \tilde{c}_{k,l}\cdot\tilde{\beta}^{n_1}_h(\vec{x} - h\vec{k}) \qquad (23)$$

where

$$\tilde{c}_{k,l} = h^2\cdot\sum_{k,l\in\mathbb{Z}}\langle R^*_\theta p_{\theta,s}(\vec{x}), \beta^{n_1}_h(\vec{x} - h\vec{k})\rangle.$$

TABLE I
SUMMARY OF PROCEEDURE

Sampled Radon kernel: $d_{i,\theta,k,l} = \langle \varphi_s(t-is), R_\theta \beta_h^n(t-k\cdot h\cos\theta - l\cdot h\sin\theta)\rangle_t$				Representation
1.	$f_h(x,y) = \sum_{k,l} c_{k,l}\cdot \beta_h^n(x-hk, y-hl)$,	where	$c_{k,l} = h^2 \left\langle \widetilde{\beta}_h^n(x-hk, y-hl), f(x,y)\right\rangle$.	B-spline
	$\Downarrow$		$\Downarrow$	
2.	$g_{\theta,s}(t) = \sum_i \tilde{c}_{i,\theta}\cdot \overline{\beta}_s(t-is)$,	where	$\tilde{c}_{i,\theta} = s\cdot \sum_{k,l} c_{k,l}\cdot d_{i,\theta,k,l}$.	Dual-spline
1'.	$p_\theta(t) = (g_\theta * q)(t)$,	where	$\widehat{q}(\omega) = \lvert\omega\rvert$.	Continuous
	$\Downarrow$			
2'.	$p_{\theta,s}(t) = \sum_i e_{i,\theta}\cdot \beta_s(t-is)$,	where	$e_{i,\theta} = s\cdot \left\langle \widetilde{\beta}_s(t-is), p_\theta(t)\right\rangle$.	B-spline
	$\Downarrow$		$\Downarrow$	
3'.	$\bar{f}_h(x,y) = \sum_\theta \sum_{k,l} \tilde{c}_{k,l}\cdot \overline{\beta}_h^n(x-hk, y-hl)$,	where	$\tilde{c}_{k,l} = h^2\cdot \sum_{i,\theta} e_{i,\theta}\cdot d_{i,\theta,k,l}$.	Dual-spline

Explanation: The spline-convolution Radon transform is specified in steps 1 and 2. In step 1, the original image is approximated in the B-spline space. In Step 2, the Radon transform is calculated. The resulting sinogram is in the dual-spline space. The spline-convolution filtered back-projection is specified in steps 1'–3'. A practical application starts in Step 1' with a set of discrete measurement from the CT scanner $g_\theta(t)$. In Step 1', the sinogram is ramp-filtered in the Fourier domain, then, in Step 2', the filtered sinogram is approximated in the B-spline space. In the final Step 3', the back-projection is performed and summed over all angles θ. The reconstructed image is in the dual-spline space.

Proposition 3: For any given angle θ, the following adjoint relationship holds:

$$\langle f, R_\theta^* g\rangle = \langle R_\theta f, g\rangle$$

where f stands for the image and g for the sinogram. The proof of Proposition 3 is given in Appendix III. □

Using Proposition 3, the coefficients $\tilde{c}_{k,l}$ are written as

$$\tilde{c}_{k,l} = \sum_{i\in\mathbb{Z},\theta} c_{\theta,i}\cdot h^2 \cdot \underbrace{\langle \beta_s^{n_2}(t-is), R_\theta \beta_h^{n_1}(t-h\vec{k}^\top\vec{\theta})\rangle_t}_{d_{i,\theta,k,l}} \quad (24)$$

where $d_{i,\theta,k,l} = \beta^{n_1,n_1,n_2}_{h\lvert\cos\theta\rvert,h\lvert\sin\theta\rvert,s}(h\vec{k}^\top\vec{\theta} - is)$ is the same Radon kernel as in the spline-convolution Radon transform (19). Again, the spline spaces were chosen such that spline trikernels could be used. In practice, the sinogram is specified by its sample values. Thus, we can combine steps 1 and 2 into a single filtering operation in the Fourier domain using a slightly modified discrete ramp filter h. In Appendix IV, we derive the frequency response $H(e^{j\omega})$ of the modified ramp filter that is optimized for our spline framework.

VI. IMPLEMENTATION

The important formulas are summarized in Table I. The key is the sampled *Radon kernel*, which is a convolution of three B-splines of various widths. In the first column of the table, we give the formulas to get the continuous functions from the coefficients. The coefficients are listed in the second column. In the third column, we specify the type of basis functions used for the representation. Whenever the Radon kernel is used, we start with B-splines and compute an approximation in terms of dual-splines. At the very end, the results are provided in the cardinal basis (pixel values) which involves an additional postfiltering step (resampling of the spline model).

A. Spline-Convolution Radon Transform

Algorithm 1 describes the spline-convolution Radon transform: the image representation is changed to the B-spline space in the first line (interpolation). The algorithm loops over a discrete set of K projections angles and over all image coefficients $c_{k,l}$. A point $\vec{k} = (k,l)$ in the image projects at the position $t = h\vec{k}^\top\vec{\theta}$ on the sinogram. The relative shift $t - i$ determines the points at which the trikernel is evaluated. The lines of the sinogram are changed from the dual-spline to the cardinal spline representation to get the sample values at the very end.

Algorithm 1 Spline-convolution Radon transform:

```
BsplCoeff=ComputeBsplineCoeff(Image, n1)
for j:=0 to K − 1
  h1 = sin(θ[j]) · h
  h2 = cos(θ[j]) · h
  Support=TriKernelSupport(n1, n1, n2, h1, h2, s)
  for k:=0 to Nx − 1
  for l:=0 to Ny − 1
    t = x[k] · h1 + y[l] · h2
    [imin, imax] =GetConcernedIndices(t, Support)
    for i := imin to imax
      RadonDual[i, j]+=TriKernel(n1, n1, n2,
                        |h1|, |h2|, s, t − i)· BsplCoeff[k, l]
    end
  end
  end
  Radon[:, j]=ComputeCardCoeff(RadonDual[:, j], n2)
end
```

B. Spline-Convolution Filtered Back-Projection

Algorithm 2 describes the spline-convolution filtered back-projection: It sums the back-projections of a set of K projection angles. First, each sinogram line is filtered by the modified ramp filter h in the Fourier domain to yield the B-spline coefficients of the filtered projection. Note that, we apply four-times zero-padding to suppress dishing artifacts (see [26]). Second, the loop over all image coefficients is performed. The Radon kernels are the same as in algorithm 1, with one exception: The data is transformed in the opposite direction, from the Radon space to the

image space. Finally, the reconstructed image is divided by the number K of summed back-projections and changed to the cardinal spline representation.

Algorithm 2 Spline-convolution back-projection:

```
for j:=0 to K − 1
  Proj = Radon[:,j]
  FiltProjBsplCoeff=
            FilterInFourier(Proj,'SplineRamp')
  h1 = sin(θ[j]) · h
  h2 = cos(θ[j]) · h
  Support=TriKernelSupport(n1, n1, n2, h1, h2, s)
  for k:=0 to Nx − 1
  for l:=0 to Ny − 1
    t = x[k] · h1 + y[l] · h2
    [imin, imax]=GetConcernedIndices(t, Support)
    for i := imin to imax
      ImageDual[k,l]+=TriKernel(n1, n1, n2,
            |h1|, |h2|, s, t − i)·FiltProjBsplCoeff[i]
    end
  end
  end
end
Image=ComputeCardCoeff(ImageDual, n1)
Image=π/K·Image
```

C. Efficient Computation of B-Spline Convolution Kernels

The evaluation of spline m-kernels in (7) requires the computation of polynomials. It gets more expensive when the degree increases or when the support (8) of the spline m-kernel increases. If the same spline m-kernel needs to be evaluated many times, it is efficient to precalculate the spline m-kernel.

In the case of the Radon transform, we precompute the Radon kernel—a spline trikernel—for various projection angles θ and store them in a lookup table (LUT). The Radon kernels are precomputed for N_θ angles between 0 and $\pi/4$ and for N_x values of x between 0 and half the support of the Radon kernel. The values for other angles or negative x can be deduced from the following symmetry properties of the Radon kernels:

$$\beta^{n_1,n_1,n_2}_{|\sin\theta|,|\cos\theta|,s}(x) = \begin{cases} \beta^{n_1,n_1,n_2}_{\left|\cos\left(\frac{\pi}{2}-\theta\right)\right|,\left|\sin\left(\frac{\pi}{2}-\theta\right)\right|,s}(x), & \text{for } \frac{\pi}{4} < \theta < \frac{\pi}{2} \\ \beta^{n_1,n_1,n_2}_{\left|\sin\left(\theta-\frac{\pi}{2}\right)\right|,\left|\cos\left(\theta-\frac{\pi}{2}\right)\right|,s}(x), & \text{for } \frac{\pi}{2} < \theta < \pi \end{cases}$$

$$\beta^{n_1,n_2,n_3}_{h_1,h_2,h_3}(x) = \beta^{n_1,n_2,n_3}_{h_1,h_2,h_3}(-x), \text{ for } x < 0.$$

The lookup table method can introduce errors if the table size $N_x \times N_\theta$ is too small. The spline-convolution algorithm was tested with Radon-kernel lookup tables with two different table resolutions: At a LUT resolution of 100×100, the PSNR [defined in (28)] of the reconstructed images decreases by approximately 1%. At a LUT resolution of 1000×1000, the error is insignificant ($< 0.001\%$). This has important practical implications as we found that the precalculated LUTs allows for the acceleration of the trikernel-based Radon transforms by one to two orders of magnitudes.

D. Computational Complexity

The computational complexity of the spline-convolution algorithm is proportional to the number of evaluated image coefficients and to the support of the Radon kernel.

Support of the Radon-kernel: The support of the Radon kernel varies with the projection angle θ as

$$\text{supp}(\beta^{n_1,n_1,n_2}_{h|\cos\theta|,h|\sin\theta|,s}) = h \cdot (|\cos\theta| + |\sin\theta|) \cdot (n_1 + 1) + s \cdot (n_2 + 1). \quad (25)$$

The rounded-up support is equivalent to the number of sinogram points contributing to the projection. The average of the rounded-up supports over all projection angles is proportional to the computational complexity as

$$\overline{\text{supp}}(\beta^{n_1,n_1,n_2}_{h|\cos\theta|,h|\sin\theta|,s}) = \frac{1}{K}\sum_{i=1}^{K}\left\lceil \text{supp}(\beta^{n_1,n_1,n_2}_{h|\cos\theta_i|,h|\sin\theta_i|,s})\right\rceil. \quad (26)$$

The averaged support increases with the sinogram degree n_2 and—by a $4/\pi$ greater factor—with the image degree n_1.

Complexity of the spline-convolution Radon transform and of its inverse: The computational complexity of the spline-convolution Radon transform (and of its inverse) depends on the image size (N_x, N_y), the number of projection angles K, the sinogram sampling step s, the image sampling step h and the support of the spline trikernel. The computation time is proportional to

$$N_x \cdot N_y \cdot K \cdot \frac{1}{s} \cdot \left(\overline{\text{supp}}(\beta^{n_1,n_1,n_2}_{h|\cos\theta|,h|\sin\theta|,s}) + c\right) \quad (27)$$

where the constant c stands for some overhead. The averaged support (26) of the Radon kernel lies between three (for the (0, 0)-kernel) up to ten (for the (2, 5)-kernel).

VII. Experiments

In this section, we evaluate the performance of the proposed spline-convolution Radon transform R and of its inverse $q * R^*$. We took the analytical Shepp–Logan phantom definition as a starting point for the evaluation of our algorithms. It consists of ten ellipses whose Radon transform (projections) can be computed analytically [26]. All our algorithms take discrete values as input, but assume an implicit continuous function defined by the interpolation of the samples by a B-Spline model. To avoid aliasing, those samples were taken such that the error between the underlying spline and the analytical phantom (respectively, the analytical projection) was minimized. This was done by over-sampling the analytical phantom (respectively, projection) by a factor of four and then reducing it using the L_2-pyramid approach [45] which ensured optimal approximation in the L_2-sense.

For the comparison, the standards were the analytical continuous Shepp and Logan phantom projections for the Radon transform and the phantom itself for the FBP. While all our algorithms return discrete pixel or projection values, they also correspond to an underlying continuous spline representation of the solution. We took advantage of this property to evaluate our solution against the analytical one (universal gold standard) in the continuous domain using the L_2-norm. Specifically, we used the

TABLE II
SPLINE-CONVOLUTION-BASED (a) RADON TRANSFORM AND (b) FBP: PROJECTION AND RECONSTRUCTION ERROR

$n_1 \backslash n_2$	0	1	2	3	4
0	**33.79**	**38.00**	38.50	38.65	38.71
1	34.26	**39.59**	**40.27**	**40.52**	40.64
2	34.31	39.79	40.46	**40.72**	**40.85**
3	34.32	39.84	40.51	40.77	**40.90**
4	34.33	39.87	40.54	40.79	**40.92**

(a)

$n_1 \backslash n_2$	0	1	2	3	4
0	**22.54**	22.77	22.80	22.81	22.81
1	**24.44**	**24.94**	24.99	25.01	25.03
2	24.69	**25.21**	25.26	25.29	25.30
3	24.78	**25.32**	**25.37**	25.39	25.40
4	24.82	25.36	**25.41**	**25.43**	**25.45**

(b)

Experimental conditions: Shepp–Logan phantom of size $N = 128$ with $K = 256$ projections and sampling step $s = 1$. The index $n_1 \in \{0, 1, 2, 3, 4\}$ on the left indicates the degree of the 2-D spline space in the image domain and the index $n_2 \in \{0, 1, 2, 3, 4\}$ on top indicates the degree of the 1-D spline-space in the Radon transform domain and the trikernel is given by $\beta^{n1,n1,n2}_{|\sin\theta|,|\cos\theta|,1}(t)$. The bold numbers indicate solutions that lie on the quality-speed optimal curve in Figs. 5 and 6.

TABLE III
STANDARD (INTERPOLATION AND SAMPLING) (a) RADON TRANSFORM AND (b) FBP: PROJECTION AND RECONSTRUCTION ERROR

$n_1 \backslash n_2$	0	1	2	3	4
0	32.93	35.02	35.86	35.93	35.96
1	34.10	37.83	39.48	39.64	39.76
2	34.12	38.35	40.06	40.24	40.35
3	34.13	38.46	40.22	40.42	40.53
4	34.14	38.51	40.30	40.51	40.63

(a)

$n_1 \backslash n_2$	0	1	2	3	4
0	22.24	22.58	22.63	22.65	22.66
1	23.34	24.29	24.42	24.46	24.47
2	23.98	25.00	25.16	25.21	25.24
3	24.05	25.09	25.25	25.31	25.34
4	24.08	25.14	25.31	25.37	25.40

(b)

Experimental conditions: Shepp-Logan phantom of size $N = 128$, recovered from $K = 256$ projections, and sampling step $s = 1$. n_1 denotes the degree in the images space and n_2 in the Radon transform space.

following definition of the PSNR for the comparison of the reconstructed image:

$$\text{PSNR} = 10\log_{10}\left(\frac{(\max\{f\} - \min\{f\})^2}{\frac{1}{\text{Area}}\left\|f - \widetilde{f}\right\|^2_{L_2}}\right) \qquad (28)$$

where f and $\widetilde{f}$ are the reference and the reconstructed solution, respectively. For the Radon transform, the denominator is the average L_2-error over the set of computed projections and the max and min are the extrema of the analytical Radon transform. Practically, the L_2-norm in the denominator was estimated from the average l_2-error of the oversampled versions of the reference using an oversampling factor of four.

In the following sections, we investigate the relation between the spline degrees on the image and on the sinogram, the kernel support, the sinogram spatial and angular sampling, as well as the influences of these parameters on the image quality, on the computational complexity and on the runtime of the algorithms. The goal is to find the best set of parameters for the proposed algorithms.

A. Degree on Sinogram Versus Degree on Image

Is it better to raise the degree of the spline model in the image or in the sinogram? This experiment evaluates 25 different spline trikernels with all possible combinations of spline degrees n_1, $n_2 \in \{0, 1, 2, 3, 4\}$ on the image and on the sinogram. The experimental results are listed in Table II(a) for the Radon transform and Table II(b) for FBP; some of the FBP error images are depicted in Fig. 4. The PSNR is plotted versus the computation time in Figs. 5 and 6, respectively. The envelope—a convex hull—represents the best compromise between computation time and image quality. We conclude that for the Radon transform it is better to use a degree on the sinogram that is equal or superior to the degree on the image, whereas for FBP the opposite is true. In summary, the degree of the target space should be equal or higher than the degree of the input space.

B. Comparison With Standard Technique

The main difference between our implementation and the standard technique is twofold: First, we use better interpolation models (higher order splines); second, we use a least-squares discretization technique instead of straightforward resampling.

To answer the question of whether this is really worth the effort, we present results for the standard implementation, but with the same models as in our previous experiment to facilitate the comparison (cf. Table III). To implement the Radon transform according to the standard method, we resampled (15) to get the values of the sinogram. Likewise, we resampled the spline-interpolated sinogram (22) to compute the back-projection.To answer the question of whether this is really worth the effort, we present results for the standard implementation, but with the same models as in our previous experiment to facilitate the comparison (cf. Table III). To implement the Radon transform according to the standard method, we resampled (15) to get the values of the sinogram. Likewise, we resampled the spline-interpolated sinogram (22) to compute the back-projection.

Here, too, the performance improves with the order of the model with a tendency to saturation for $n \geq 3$. For example, one gets almost 1 dB improvement if one uses cubic splines instead of the bilinear interpolation used in most implementations [cf. Table III(b)].

Now, if we compare Tables II and III, we observe that the least-squares sampling provides an additional boost in performance, especially for lower order models (up to 2.75 dB for the Radon transform, or up to 1.14 dB for FBP). Thus, it makes sense to use the more sophisticated methods if the goal is to get the best possible images.

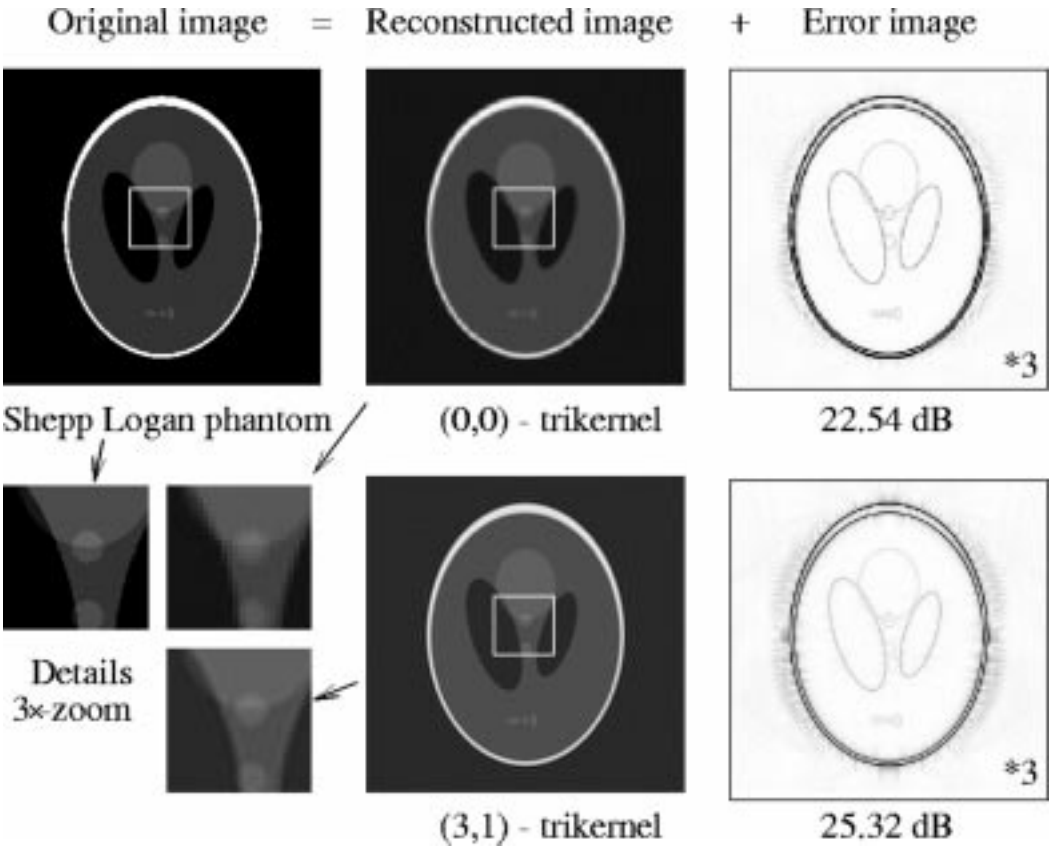


Fig. 4. Spline-convolution filtered back-projection. The displayed error images are amplified by a factor of 3.

C. Predictions Versus Measurements: Runtime and Image Quality

The predicted computation time (27) approximates well the measured computation time from Fig. 6 with the parameters $h = 1$, $s = 1$, $N_x = N_y = 128$, $K = 256$, $s = 1$, and $c \approx 1.23$; the correlation coefficient is 0.9994. From the analytical expression (20) of the approximation error for a B-spline, we can estimate the expected image quality by $\widehat{\mathrm{PSNR}} = 10 \log\left(255/\varepsilon^2(s)\right)$. Fig. 7 depicts the estimated PSNR versus the estimated computation time. A similar envelope as in Fig. 6 is reproduced.

D. FBP: Smaller Sampling Step on the Sinogram Lines

Is it better to raise the spline trikernel degrees or to refine the sinogram sampling step? The answer is provided by Fig. 8. It shows the reconstruction error (PSNR) and the computation time. The analytical projection of the Shepp-Logan head phantom is back-projected by the spline-convolution FBP. The sinogram sampling steps are $s \in \{1, 1/2, 1/4\}$ and the spline trikernel degrees are taken from the envelope of Fig. 6 (best cost/performance compromise). When the sampling step s on the sinogram lines gets finer, the image quality increases, but the computation time rises as well. One can see that a (2,1)-trikernel at sampling step $s = 1$ is nearly as good and fast as the (1,0)-trikernel at the finer sampling step $s = 1/2$. But a finer sampling step requires that more sinogram data is acquired. For a fixed amount of sinogram data, e.g., measurements from a CT scanner, it is clear that a higher kernel degree yields a better quality of the reconstructed image. For finer sampling steps the kernels of higher degree quickly reach an upper-quality limit. Already for a quadric kernel degree, there is hardly a difference between two-times and four-times sinogram oversampling. Therefore, we can recommend to use either high-degree kernels at the sampling step $s = 1$, or at least the (1,0)-trikernel with a sampling step of $s = 1/2$. Four-times sinogram oversampling ($s = 1/4$) results in insignificant improvements only.

E. FBP: Angular Resolution Versus Sinogram Sampling Step

Is it better to use a finer angular or a finer sinogram sampling step? In the FBP literature [18], the customary rule is to use four-times over-sampling on the sinogram and twice as many angles as the size of the image along one dimension. Our experiments refines this rule and suggests similar rules for higher approximation orders. In a large FBP experiment (Table IV), we have evaluated the image quality and computation time for numerous projection angles, $K \in \{128, 192, 256, 384, 512\}$ and sinogram sampling steps, $s \in \{1, 1/2, 1/4\}$, for trikernels of degrees taken from the envelope of Fig. 6.

We draw three conclusions from Table IV: first, higher kernel degrees give the best improvement in image quality for comparable small costs. Second, it is faster and slightly better to double the sampling rate ($s = 1/2$ instead of $s = 1$) while halving the angular resolution ($K = N$ instead of $K = 2N$). Third, increasing the angular sampling above $K \geq 2N$ or decreasing the sampling step $s \leq 1/2$ does not improve the quality significantly.

In additional experiments, we have verified that the presented results reproduce as well for larger image sizes and the same

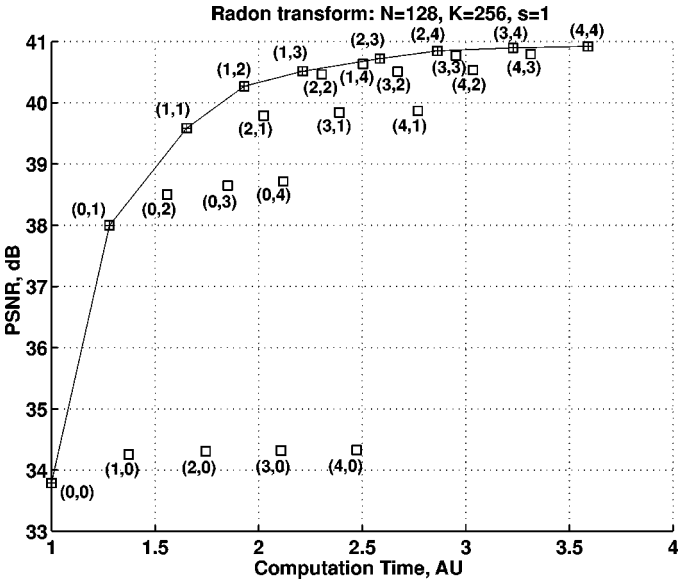


Fig. 5. Speed-quality evaluation of the spline-based Radon transform: Comparison of PSNR versus computation time for each (n_1, n_2)-trikernel $0 \leq n_1 < 4, 0 \leq n_2 < 4$. By convention, the time is set to 1 for the (0, 0)-kernel. The first number represents the degree n_1 on the image and the last the degree n_2 on the sinogram. For the (0, 0)-kernel, the absolute runtime is 5.5 s (on Mac G3, $N = 128$, $K = 256$, $s = 1$). The envelope is the best compromise between speed and quality.

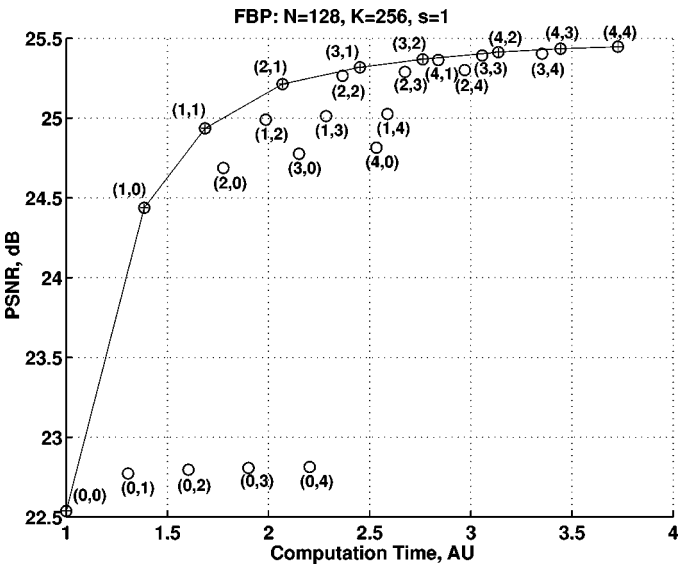


Fig. 6. Speed-quality evaluation of the spline-based filtered-back projection: Comparison of PSNR versus computation time for each (n_1, n_2)-trikernel. For the (0,0)-kernel, the absolute runtime is 6 s (on Mac G3, $N = 128$, $K = 256$, and $s = 1$). The envelope is the best compromise between speed and quality.

ratio between the angle resolution and the image resolution. The improvements in image quality are marginal for kernels of higher degrees $n_1, n_2 \geq 2$. We conclude that for the spline-convolution FBP, the best compromise between speed and quality is obtained with a trikernel of degrees $(n_1, n_2) = (3, 1)$ at sinogram sampling step $s = 1$ and as many angles $K = N$ as the image size N. The highest quality is achieved already at $s = 1/2$ and $K = 1.5N$.

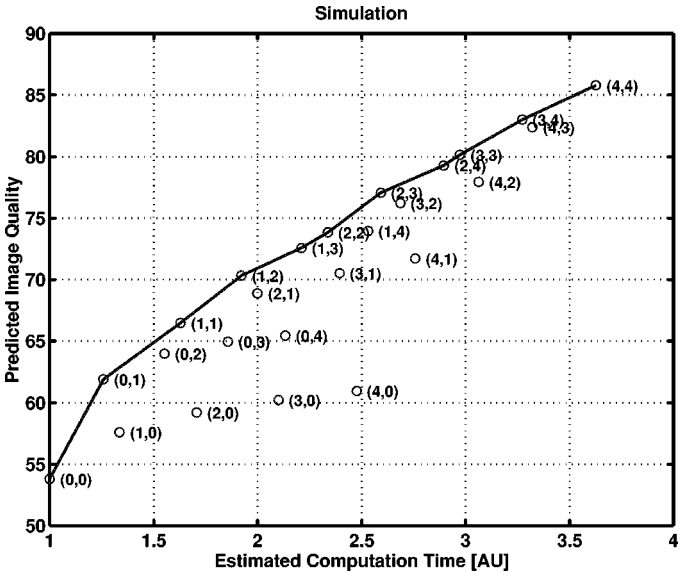


Fig. 7. Prediction of speed-quality optimal solutions of the spline-based Radon transform: The image quality is plotted versus the computation time. The predicted envelope is similar to the one found for the measured values (compare with Fig. 5). In reality, the quality will be upper-bounded by additional approximation errors, e.g., due to the angular sampling or the ramp filtering. These errors were not taken into account here.

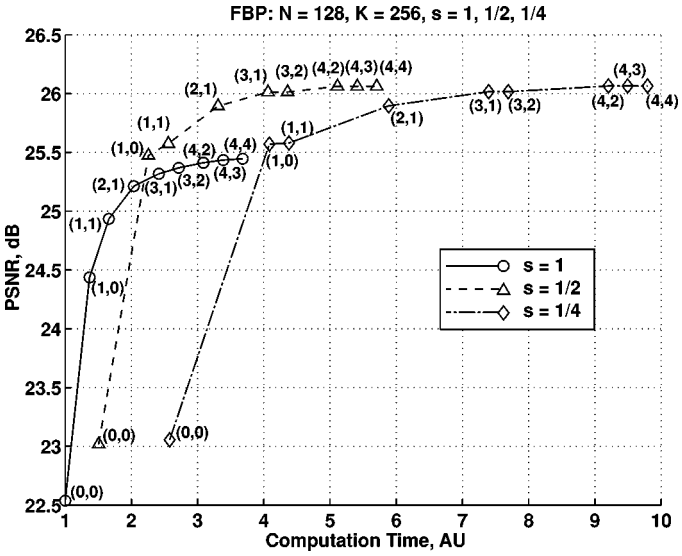


Fig. 8. Reconstruction quality (in PSNR) as a function of the computation time and of the sampling step s on the sinogram lines. The time scale is normalized to 1 for the routine with the lowest trikernel degree (0,0). The first number represents the degrees on the image and the last the degree on the sinogram.

In practice, the resolution of the measured CT data is limited by the maximum sinogram sampling rate and the maximum angular resolution of the CT scanner. There is a quality trade-off between the number of given angular projections and the achievable resolution of the reconstructed image. First, a higher kernel degree preserves image quality even when the acquisition sampling rate is halved (less angles and/or less samples per projection), which might imply a speed-up of the acquisition system (compare Table IV). Second, a higher kernel degree also allows a finer reconstruction resolution without quality degradations. But if the reconstructed image is too large ($N > K$), then artifacts—due to angular under-sampling—occur in the borders.

TABLE IV
COMPARISON FOR THE SPLINE-CONVOLUTION FBP OF ANGULAR RESOLUTION K VERSUS SINOGRAM SAMPLING STEP s

PSNR in dB (Computation time (Arbitrary units)), $N = 128$, K=number of angles						
trikernel	sampl. step	$K = 128$	$K = 192$	$K = 256$	$K = 386$	$K = 512$
(0,0)	$s = 1/1$	**22.50 (0.5)**	22.53 (0.8)	22.54 (1.0)	22.54 (1.5)	22.54 (2.0)
(0,0)	$s = 1/2$	22.97 (0.8)	23.01 (1.2)	23.02 (1.5)	23.02 (2.3)	23.02 (3.0)
(0,0)	$s = 1/4$	22.98 (1.3)	23.05 (2.0)	23.06 (2.6)	23.06 (3.9)	23.06 (5.1)
(1,0)	$s = 1/1$	**24.40 (0.7)**	24.43 (1.0)	24.44 (1.4)	24.44 (2.1)	24.44 (2.7)
(1,0)	$s = 1/2$	25.40 (1.2)	25.47 (1.7)	25.47 (2.3)	25.48 (3.4)	25.48 (4.5)
(1,0)	$s = 1/4$	25.47 (2.1)	25.57 (3.1)	25.57 (4.1)	25.57 (6.2)	25.57 (8.2)
(1,1)	$s = 1/1$	**24.89 (0.9)**	24.93 (1.3)	24.94 (1.7)	24.94 (2.5)	24.94 (3.3)
(1,1)	$s = 1/2$	25.45 (1.3)	25.57 (1.9)	25.57 (2.6)	25.58 (3.9)	25.58 (5.1)
(1,1)	$s = 1/4$	25.46 (2.3)	25.58 (3.3)	25.58 (4.4)	25.58 (6.6)	25.58 (8.8)
(3,1)	$s = 1/1$	**25.28 (1.2)**	25.32 (1.8)	25.32 (2.4)	25.32 (3.7)	25.32 (4.9)
(3,1)	$s = 1/2$	25.92 (2.1)	26.01 (3.1)	26.01 (4.1)	26.01 (6.2)	26.01 (8.2)
(3,1)	$s = 1/4$	25.93 (3.8)	26.01 (5.6)	26.02 (7.5)	26.02 (11.3)	26.02 (14.9)

The Shepp-Logan phantom's image size is $N = 128$. The time scale is normalized to 1 for the (0,0)-trikernel and $K = 256$. The best tradeoff between quality and speed is indicated by the bold values.

VIII. CONCLUSION

We derived an explicit formula for the convolution of multiple B-splines, which we called B-spline convolution kernels. We imposed a continuous B-spline model on the image and its Radon transform. We showed that the Radon transform of the basis function (a tensor-product B-spline) is a spline bikernel. The approximation of the spline bikernel in the dual-spline space corresponds to the sampled convolution of three B-splines, the so-called Radon kernel. With it, the Radon transform and its inverse are consistently and efficiently discretized using B-spline signal processing.

The approximation power of our method increases with the degree of the splines. The best compromise between computational complexity and approximation accuracy is achieved with Radon kernels with a higher degree on the target space. The upper image-quality bound is already reached at two-times sinogram oversampling. This makes the commonly recommended four-times oversampling superfluous and allows for better resolution of the reconstructed image for a fixed sinogram resolution.

We suggest to use the spline-convolution Radon transform and FBP with Radon kernels of degrees (n_1, n_2) of at least (1,1), together with up to two-times sinogram oversampling $s \geq 1/2$ and an angular resolution $K \leq 2N$ less than twice the image size N along one axis.

APPENDIX I
PROOF OF PROPOSITION 1

First, it is easy to establish that

$$x_+^{n-1} * x_+^0 = \int_0^x x_+^{n-1} \mathrm{d}x = \frac{x_+^n}{n}.$$

By induction, we find that

$$\underbrace{x_+^0 * \cdots * x_+^0}_{n+1 \text{ terms}} = \frac{x_+^n}{n!}. \tag{29}$$

Next, we consider the convolution of iterated finite difference operators

$$\Delta_h^{n+1} = \underbrace{\Delta_h^1 * \cdots * \Delta_h^1}_{n+1 \text{ terms}}.$$

The transfer function (Fourier transform) of this convolution operator is

$$\widehat{\Delta}_h^{n+1}(\omega) = \left(\frac{e^{j\frac{\omega h}{2}} - e^{-j\frac{\omega h}{2}}}{h} \right)^{n+1}$$

which can be expanded to yield

$$\widehat{\Delta}_h^{n+1}(\omega) = \sum_{k=0}^{n+1} (-1)^k \binom{n+1}{k} \frac{e^{j\omega h \cdot \left(\frac{n+1}{2} - k\right)}}{h^{n+1}}. \tag{30}$$

The explicit time-domain formula (4) is then obtained by interpreting the complex exponentials of (30) as time shifts. Combining (29) and (4), then yields (5).

Using the B-spline formula (5) and the commutativity of convolution operators, we can now express the spline m-kernel as

$$\begin{aligned} \beta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m}(x) &= \left(\Delta_{h_1}^{n_1+1} * \frac{x_+^{n_1}}{n_1!} \right) * \cdots * \left(\Delta_{h_m}^{n_m+1} * \frac{x_+^{n_m}}{n_m!} \right) \\ &= \left(\Delta_{h_1}^{n_1+1} * \cdots * \Delta_{h_m}^{n_m+1} \right) * \left(\frac{x_+^{n_1}}{n_1!} * \cdots * \frac{x_+^{n_m}}{n_m!} \right). \end{aligned}$$

Thanks to (29), this can be rewritten as

$$\beta_{h_1,\ldots,h_m}^{n_1,\ldots,n_m}(x) = \left(\Delta_{h_1}^{n_1+1} * \cdots * \Delta_{h_m}^{n_m+1} \right) * \frac{x_+^{N_m}}{N_m!} \tag{31}$$

which is equivalent to (7). □

APPENDIX II
PROOF OF PROPOSITION 2

Let $g_\theta(t) = R_\theta \beta^n(t)$ be the Radon transform of the tensor product of two B-splines. The Fourier Slice Theorem states the equivalence between the 1-D Fourier transform of a parallel projection and a 1-D cut in the 2-D Fourier transform of an image

$$\mathcal{F}_{1\mathrm{D}} \left\{ R_\theta f(\cdot) \right\} (\omega) = \mathcal{F}_{2\mathrm{D}} \left\{ f(\cdot,\cdot) \right\} (\omega \cos(\theta), \omega \sin(\theta)). \tag{32}$$

By performing a 1-D inverse Fourier transform on both sides of (32) and from the scaling properties and convolution properties of the Fourier Transform, we deduce

$$\begin{aligned} R_\theta \beta^n(t) =& \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{\beta}^n(\omega\cos\theta)\cdot\hat{\beta}^n(\omega\sin\theta)e^{i\omega t}d\omega \\ =& \frac{1}{|\cos\theta|\cdot|\sin\theta|}\beta^n\left(\frac{\cdot}{|\cos\theta|}\right) * \beta^n\left(\frac{\cdot}{|\sin\theta|}\right)(t) \\ =& \beta^n_{|\cos\theta|} * \beta^n_{|\sin\theta|}(t). \end{aligned}$$

Therefore, the Radon transform of a 2-D B-spline can be written as a 1-D convolution of two scaled B-splines. A further scaling of this results by h yields the result in Proposition 2.

Appendix III
Proof of Proposition 3

Without loss of generality, we assume in this proof that the projection direction $\vec{\theta} = (0,1)$ of the Radon projection R_θ and of its inverse R^*_θ is aligned with the coordinate system (x, y). Otherwise, the coordinate system has to be rotated by an angle $-\theta$. Let f be the image and g the sinogram.

We take a Cartesian coordinate system (x, y) that is aligned with the projection direction $\vec{\theta} = (0,1) = (x, y)$ of the Radon projection R_θ and of its inverse R^*_θ. Let f be the image and g the sinogram.

Then, $(R_\theta f)(x) = \int f(x,y)dy$ and $(R^*_\theta g)(x,y) = g(x)\,\forall y$. The 2-D scalar product of $R^*_\theta g$ and f is

$$\begin{aligned} \langle R^*_\theta g, f\rangle =& \langle (R^*_\theta g)(x,y), f(x,y)\rangle \\ =& \iint f(x,y)\cdot(R^*_\theta g)(x,y)\,\mathrm{d}x\mathrm{d}y \\ =& \iint f(x,y)\cdot g(x)\,\mathrm{d}x\mathrm{d}y \\ =& \iint f(x,y)\,\mathrm{d}y\cdot g(x)\,\mathrm{d}x \\ =& \int (R_\theta f)(x)\cdot g(x)\,\mathrm{d}x \\ =& \langle (R_\theta f)(x), g(x)\rangle \\ =& \langle R_\theta f, g\rangle\,. \end{aligned}$$

□

We have shown that the 2-D scalar product in the image domain between a back-projected sinogram line and an image is equivalent to the 1-D scalar product of the projected image and the sinogram line.

Appendix IV
The Spline Ramp Filter

We propose an implementation of a ramp filter that is consistent with our B-spline framework. We consider a function $f(x)$ that is given by its expansion in a shift-invariant basis $\{\varphi(x-k)\}_{k\in\mathbb{Z}}$

$$f(x) = \sum_{k\in\mathbb{Z}} c_k \varphi(x-k) \tag{33}$$

whose Fourier transform is

$$\hat{f}(\omega) = C(e^{j\omega})\cdot\hat{\varphi}(\omega), \quad \text{with} \quad C(e^{j\omega}) = \sum_{k\in\mathbb{Z}} c_k e^{jk\omega}. \tag{34}$$

We will apply the ramp filter $\widehat{q}(\omega) = |\omega|$ to this function and then approximate the result as a linear combination of $\{\varphi_2(x-k)\}_{k\in\mathbb{Z}}$

$$g(x) = \sum_{k\in\mathbb{Z}} a_k \varphi_2(x-k). \tag{35}$$

The coefficients a_k in (35) that provide the least-squares approximation of $q * f$ are given by

$$a_k = \langle q * f, \widetilde{\varphi}_2(\cdot - k)\rangle$$

where $\widetilde{\varphi}_2$ is the dual of φ_2. Clearly, the a_k are the integer samples of $q*f*\widetilde{\varphi}_2(-x)$ whose Fourier transform is $|\omega|\,\widehat{f}(\omega)\widehat{\widetilde{\varphi}}^*_2(\omega)$. The Fourier transform of a_k then follows by simple periodization

$$A(e^{j\omega}) = \sum_{k\in\mathbb{Z}} a_k e^{j\omega} = \sum_{k\in\mathbb{Z}} |\omega+2k\pi|\,\widehat{f}(\omega+2\pi)\widehat{\widetilde{\varphi}}^*_2(\omega+2k\pi).$$

Replacing $\hat{f}(\omega)$ by its expression in (34) we get

$$A(e^{j\omega}) = H(e^{j\omega})\cdot C(e^{j\omega})$$

where $H(e^{j\omega})$ is the frequency response of the digital filter we are looking for, which is given by

$$H(e^{j\omega}) = \sum_{k\in\mathbb{Z}} |\omega+2k\pi|\,\widehat{\widetilde{\varphi}}^*_2(\omega+2\pi)\widehat{\varphi}(\omega+2k\pi).$$

The choice of the basis functions that are relevant for our implementation are the cardinal spline $\varphi(x) = \eta^{n_1}(x)$ and the B-spline $\varphi_2(x) = \beta^{n_2}(x)$. The Fourier transforms of the cardinal and the dual B-splines are (cf. [44] and [45])

$$\begin{aligned} \widehat{\varphi}(x) =& \widehat{\eta}^{n_1}(x) = \frac{\mathrm{sinc}^{n_1+1}\left(\frac{\omega}{2\pi}\right)}{\sum\limits_{k=-n_1/2}^{+n_1/2}\beta^{n_1}(k)e^{-j\omega k}}, \\ \widehat{\widetilde{\varphi}}_2(x) =& \widehat{\beta}^{n_2}(x) = \frac{\mathrm{sinc}^{n_2+1}\left(\frac{\omega}{2\pi}\right)}{\sum\limits_{k=-n_2}^{+n_2}\beta^{2n_2+1}(k)e^{-j\omega k}}. \end{aligned}$$

Thus. the optimized digital ramp filter for our implementation is

$$H(e^{j\omega}) = \frac{\sum\limits_{k\in\mathbb{Z}}|\omega+2k\pi|\,\mathrm{sinc}^{n_1+n_2+1}\left(\frac{\omega}{2\pi}\right)}{\left(\sum\limits_{k=-n_1/2}^{+n_1/2}\beta^{n_1}(k)e^{-j\omega k}\right)\cdot\left(\sum\limits_{k=-n_2}^{+n_2}\beta^{2n_2+1}(k)e^{-j\omega k}\right)}.$$

REFERENCES

[1] J. Radon, "Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten," *Ber. Verh. König. Sächs. Ges. Wiss. Leipzig, Math.-Phys. Kl*, vol. 69, pp. 262–277, 1917.

[2] D. Ludwig, "The Radon transform on Euclidean space," *Comm. Pure Appl. Math.*, vol. 19, pp. 49–81, 1966.

[3] S. R. Deans, *The Radon Transform and Some of its Applications*. New York: Wiley, 1983.

[4] G. Hermann, *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*. New York: Academic, 1980.

[5] F. Natterer, *The Mathematic of Computerized Tomography*. Chicester, U.K.: Wiley, 1986.

[6] P. M. Joseph and R. D. Spital, "A method for correcting bone included artifacts in computed tomography scanners," *JCAT*, vol. 2, no. 1, pp. 100–108, 1978.

[7] G. Henrich, "A simple computational method for reducing streak artifacts in CT images," *Comput. Tomogr.*, vol. 4, pp. 67–71, 1980.

[8] G. Glover and N. Pelec, "An algorithm for the reduction of metal clip artifacts in CT reconstructions," *Med. Phys.*, vol. 8, no. 6, pp. 799–807, 1981.

[9] W. A. Götz and H. J. Druckmüller, "A fast digital Radon transform—An efficient means for evaluating the Hough transform," *Pattern Recogn.*, vol. 29, pp. 711–718, 1996.

[10] E. J. Candès and D. L. Donoho, "Ridgelets: A key to higher-dimensional intermittency," *Phil. Trans. R. Soc. Lond.*, vol. 357, no. 1760, pp. 2495–2509, Sept. 1999.

[11] G. N. Ramachandran and A. V. Lakshminarayanan, "Three dimensional reconstructions from radiographs and electron micrographs: Application of convolution instead of Fourier transform," in *Proc. Nat. Acad. Sci*, vol. 68, 1971, pp. 2236–2240.

[12] L. Shepp and Y. Yardy, "Maximum likelihood reconstruction for emission tomography," *IEEE Trans. Med. Imag.*, vol. 1, pp. 113–122, Mar. 1982.

[13] C. A. Bouman and K. Sauer, "A unified approach to statistical tomography using coordinate descent optimization," *IEEE Trans. Image Processing*, vol. 5, pp. 480–492, Mar. 1996.

[14] P. J. Green, "Bayesian reconstruction from emission tomography data using a modified EM algorithm," *IEEE Trans. Med. Imag.*, vol. 9, pp. 84–93, Mar. 1990.

[15] T. Hebert and R. Leahy, "A generalized EM algorithm for 3-D bayesian reconstruction from poisson data with Gibbs priors," *IEEE Trans. Med. Imag.*, vol. 8, pp. 194–202, June 1989.

[16] G. T. Herman and D. Odhner, "Performance evaluation of an iterative image reconstruction algorithm for positron emission tomography," *IEEE Trans. Med. Imag.*, vol. 3, pp. 316–324, Dec. 1991.

[17] L. A. Shepp and B. F. Logan, "The Fourier reconstruction of a head section," *IEEE Trans. Nucl. Sci.*, vol. NS-21, pp. 21–43, 1974.

[18] J.-P. Guédon and Y. Bizais, "Bandlimited and Haar filtered back-projection reconstruction," *IEEE Trans. Med. Imag.*, vol. 12, pp. 430–440, June 1994.

[19] A. H. Delaney and Y. Bresler, "Multiresolution tomographic reconstruction using wavelets," *IEEE Trans. Image Processing*, vol. 4, pp. 799ff–799ff, June 1995.

[20] T. Olson and J. DeStefano, "Wavelet localization of the Radon transform," *IEEE Trans. Signal Processing*, vol. 42, pp. 2055–67, Aug. 1994.

[21] F. Rashid-Farrokhi, K. J. R. Liu, C. A. Berenstein, and D. Walnut, "Wavelet-based multiresolution local tomography," *IEEE Trans. Image Processing*, vol. 6, pp. 1412–1430, Oct. 1997.

[22] J. Kalifa, A. Laine, and P. D. Esser, "Tomographic reconstruction with nonlinear diagonal estimators," *Wavelets Application in Signal and Image Processing VIII*, vol. 4119, no. 2, pp. 576–586, 2000.

[23] D. Gottleib, B. Gustafsson, and P. Forssen, "On the direct Fourier method for computer tomography," *IEEE Trans. Med. Imag.*, vol. 19, pp. 223–232, Mar. 2000.

[24] H. Schomberg and J. Timm, "The gridding method for image reconstruction by Fourier transformation," *IEEE Trans. Med. Imag.*, vol. 14, pp. 596–607, Sept. 1995.

[25] J. Walden, "Analysis of the direct Fourier method for computer tomography," *IEEE Trans. Med. Imag.*, vol. 19, pp. 211–222, Mar. 2000.

[26] A. C. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*, 1988.

[27] H. Stark, J. W. Woods, I. Paul, and R. Hingorani, "Direct Fourier reconstruction in computer tomography," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-29, pp. 237–244, 1981.

[28] T. M. Lehmann, C. Gönner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Trans. Med. Imag.*, vol. 18, pp. 1049–75, Nov. 1999.

[29] ——, "Addendum: B-spline interpolation in medical image processing," *IEEE Trans. Med. Imag.*, vol. 20, pp. 660–665, July 2001.

[30] E. H. W. Meijering, W. J. Niessen, and M. A. Viergever, "Quantitative evaluation of convolution-based methods for medical image interpolation," *Med. Image Anal.*, vol. 5, no. 2, pp. 111–126, June 2001.

[31] P. Thévenaz, T. Blu, and M. Unser, "Interpolation revisited," *IEEE Trans. Med. Imag.*, vol. 19, no. 7, pp. 739–758, July 2000.

[32] M. Lautsch, "A spline inversion formula for the Radon transform," *SIAM J. Num. Ana.*, vol. 26, no. 2, pp. 456–467, 1989.

[33] M. Richter, "Use of box splines in computer tomography," *Computing*, vol. 61, pp. 133–150, 1998.

[34] P. J. La Rivière and X. Pan, "Spline-based inverse Radon transform in two or three dimensions," *IEEE Trans. Nucl. Sci.*, vol. 45, pp. 2224–2231, Aug. 1998.

[35] E. V. R. Di Bella, A. B. Barclay, R. L. Eisner, and R. W. Schafer, "A comparison of rotation-based methods for iterative reconstruction algorithms," *IEEE Trans. Nucl. Sci.*, vol. 43, pp. 3370–3376, Dec. 1996.

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

[37] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, pp. 10–21, 1949.

[38] T. Blu and M. Unser, "Quantitative Fourier analysis of approximation techniques: Part I—Interpolators and projectors," *IEEE Trans. Signal Processing*, vol. 47, pp. 2783–2795, Oct. 1999.

[39] M. Unser and I. Daubechies, "On the approximation power of convolution-based least-squares versus interpolation," *IEEE Trans. Signal Processing*, vol. 45, pp. 1697–1711, July 1997.

[40] I. J. Schoenberg, "Contributions to the problem of approximation of equidistant data by analytical functions, Part A:—On the problem of smoothing or graduation," *Quart. Appl. Math.*, vol. 4, no. 1, pp. 44–99, 1946.

[41] M. Unser, "Sampling-50 years after Shannon," *Proc. IEEE*, vol. 88, pp. 569–587, Apr. 2000.

[42] A. Aldroubi and M. Unser, "Sampling procedure in function spaces and asymptotic equivalence with Shannon's sampling theory," *Numer. Funct. Ana. Optimization*, vol. 15, no. 1 & 2, pp. 1–21, 1994.

[43] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part I—Theory and Part II—Efficient design," *IEEE Trans. Signal Processing*, vol. 41, pp. 821–848, Feb. 1993.

[44] A. Aldroubi, M. Unser, and M. Eden, "Cardinal spline filters: Stability and convergence to the ideal sinc interpolator," *Signal Processing*, vol. 28, no. 2, pp. 127–138, 1992.

[45] M. Unser, A. Aldroubi, and M. Eden, "The L2 polynomial spline pyramid," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 364–379, Apr. 1993.