Generalized Ray Tracing With Basis Functions for Tomographic Projections

Youssef Haouchat, Sepand Kashani, Philippe Thévenaz, *Fellow, IEEE*, and Michael Unser, *Life Fellow, IEEE*

***Abstract*—This work aims at the precise and efficient computation of the x-ray projection of an image represented by a linear combination of general shifted basis functions that typically overlap. We achieve this with a suitable adaptation of ray tracing, which is one of the most efficient methods to compute line integrals. In our work, the cases in which the image is expressed as a spline are of particular relevance. The proposed implementation is applicable to any projection geometry as it computes the forward and backward operators over a collection of arbitrary lines. We validate our work with experiments in the context of inverse problems for image reconstruction to maximize the image quality for a given resolution of the reconstruction grid.**

***Index Terms*—Image reconstruction, inverse problems, splines, X-ray.**

I. Introduction

In THIS paper, we propose an exact method to compute the x-ray transform of an image with arbitrary geometry. The accuracy results from the high order of approximation that comes with the representation of data through spline models, combined with the exact computation of the integrals found in x-ray transforms. Our approach turns out to be computationally efficient as well.

A. State of the Art

The computation of ray-based operators for the x-ray transform often takes advantage of a ray tracer that performs a line integration. Widely used open-source software packages such as the Astra toolbox [1], the TIGRE toolbox [2], or the Reconstruction Toolkit [3] employ the Siddon ray tracing algorithm [4] or its accelerated variant [5]. These methods assume the image is piecewise-constant over the cells delimited by a grid, which corresponds to a pixel-based representation. To avoid blocking artifacts or to make an implicitly smoother description of the image, they also implement interpolation-kernel methods [6] or exploit GPU-based texture managers. For predefined projection geometries, one can also resort to more advanced methods such as footprint-based [7], distance-driven [8], or convolutional [9] for more precise projection models. The authors of [10], [11] have studied the impact of a richer representation of the image in the context of x-ray imaging but they do not exploit the computational efficiency of ray tracing as we do in our approach.

Received 26 March 2025; revised 4 August 2025; accepted 8 September 2025. Date of publication 18 September 2025; date of current version 29 September 2025. This work was supported in part by the European Research Council under Grant ERC-2020-AdG FunLearn-101020573 and in part by Swiss National Science Foundation under Sinergia Grant CRSII5_198569. The associate editor coordinating the review of this article and approving it for publication was Prof. Yang Chen. *(Corresponding author: Youssef Haouchat.)*

Youssef Haouchat, Philippe Thévenaz, and Michael Unser are with the Biomedical Imaging Group, EPFL, 1015 Lausanne, Switzerland (e-mail: youssef.haouchat@eplf.ch).

Sepand Kashani is with the Center for Imaging, EPFL, 1015 Lausanne, Switzerland.

Digital Object Identifier 10.1109/TCI.2025.3611590

B. X-Ray-Transform With Basis Functions

Let a line on the plane $\mathbb{R}^2$ be described parametrically in terms of $y \in \mathbb{R}$ as the set

$$\{t\boldsymbol{\theta} + y\boldsymbol{\theta}^{\perp} \in \mathbb{R}^2 \mid t \in \mathbb{R}\}, \tag{1}$$

where $\boldsymbol{\theta} = (\cos\theta,\ \sin\theta) \in \mathbb{S}^1(\mathbb{R})$ is the unit vector directing the line that forms an angle $\theta \in \mathbb{R}$ with the horizontal axis. The vector $\boldsymbol{\theta}^{\perp} = (\sin\theta,\ -\cos\theta) \in \mathbb{S}^1(\mathbb{R})$ is a unit vector, orthogonal to $\boldsymbol{\theta}$, such that $y\boldsymbol{\theta}^{\perp}$ can be interpreted as the orthogonal shift by y of the ray relative to the origin. In the context of imaging, y is often taken to correspond to the position of a detector.

The x-ray transform [12], [13] of the integrable function $f : \mathbb{R}^2 \to \mathbb{R}$ corresponds to the collection of all its integrals along such lines. It is expressed in terms of $\theta \in \mathbb{R}$ and $y \in \mathbb{R}$ as

$$\mathcal{P}_{\theta}\{f\}(y) = \int_{\mathbb{R}} f\left(t\boldsymbol{\theta} + y\boldsymbol{\theta}^{\perp}\right)\, \mathrm{d}t. \tag{2}$$

This measurement operator is widely used in tomography to solve inverse problems in modalities such as x-ray scan, positron emission tomography, or cryogenic electron tomography [14], [15]. Every detector is indexed by y, while every ray is indexed by θ and y, as hinted in Fig. 1.

The function f in (2) is the model of an image. It is often assumed to consist in the translations of a basis generated from $\varphi : \mathbb{R}^2 \to \mathbb{R}$ placed on a uniform Cartesian grid. For simplicity, and without loss of generality, we choose the stepsize of the grid to be equal to 1. We also assume f to be compactly supported. Then, we have that

$$f = \sum_{\mathbf{k} \in \Omega} c_{\mathbf{k}} \varphi(\cdot - \mathbf{k}), \tag{3}$$

where $\Omega = \{1, \dots, N\} \times \{1, \dots, N\}$ with N^2 the number of basis functions needed to represent f, $\mathbf{k}$ is an index on the grid, and $c_{\mathbf{k}}$ is the coefficient of f associated to the shifted basis function $\varphi(\cdot - \mathbf{k})$. Thus, the x-ray projection of a function f

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

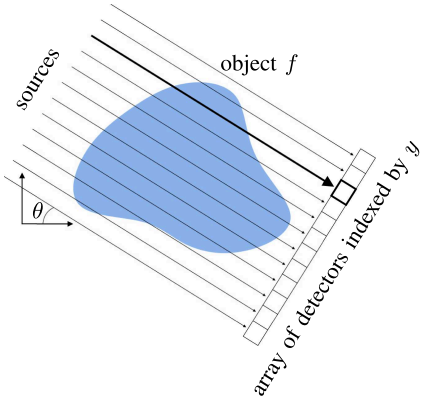

Fig. 1. Projection along a line at a specific angle indexed by θ onto a specific detector indexed by y. In this paper, we consider every projection line independently.

TABLE I
TABLE OF NOTATIONS

Symbol	Description
$\mathbb{S}^1(\mathbb{R})$	$\{(\cos\theta, \sin\theta) \in \mathbb{R}^2 \mid \theta \in [0, 2\pi)\}$
$\boldsymbol{\theta} \in \mathbb{S}^1(\mathbb{R})$	Direction of the projection
$\boldsymbol{\theta}^\perp \in \mathbb{S}^1(\mathbb{R})$	Direction orthogonal to the projection
$L^1(\mathbb{R}^2)$	$\{f : \mathbb{R}^2 \to \mathbb{R} \mid \int_{\mathbb{R}^2} \lvert f(\mathbf{x})\rvert \, \mathrm{d}\mathbf{x} < \infty\}$
Ω	Set of grid indices, denoted $\mathbf{k} \in \mathbb{R}^2$
(θ, y)	Line parameterization (angle, offset)
$\mathcal{P}_\theta\{f\}(y)$	Line integral of f along (θ, y)
$\varphi(\cdot - \mathbf{k})$	Shifted basis generator centered at $\mathbf{k}$
$\varphi_\theta(y)$	Line integral of φ along (θ, y)
x_+	$\max(x, 0)$
$\hat{f}$	Fourier transform of $f \in L^1(\mathbb{R}^2)$
$\mathbf{H}_\varphi$	Projection operator discretized with φ
δ	Dirac distribution

parameterized by its coefficients $(c_\mathbf{k})_{\mathbf{k}\in\Omega}$ is expressed as

$$\begin{aligned}\mathcal{P}_\theta\{f\}(y) &= \sum_{\mathbf{k}\in\Omega} c_\mathbf{k} \mathcal{P}_\theta\{\varphi(\cdot - \mathbf{k})\}(y) \\ &= \sum_{\mathbf{k}\in\Omega} c_\mathbf{k} \mathcal{P}_\theta\{\varphi\}(y - \langle \mathbf{k}, \boldsymbol{\theta}^\perp\rangle), \end{aligned} \tag{4}$$

with our concern being the efficient and accurate evaluation of (4) for arbitrary θ and y, given some underlying generator φ.

We summarize below the main notations used throughout the paper. These will be referenced frequently in the sequel.

C. Contribution

Ray-oriented methods such as ray tracing are particularly suitable for the handling of arbitrary geometries which may involve irregular contributions of y and θ in (4), thus prohibiting Fourier-based methods in such cases. Their principle

Algorithm 1: Contribution of One Basis Function to the X-Ray.

Require: $C, \mathbf{x}_k, \theta, (p, q)$
▷Image, intersection ray/cell, angle, cell index
$c = C[p, q]$ ▷Coefficient of the basis function
$\mathbf{o} = (p + \frac{1}{2},\ q + \frac{1}{2})$ ▷Position of the center
$y_k = \langle (\mathbf{o} - \mathbf{x}_k), \boldsymbol{\theta}^\perp \rangle$ ▷Offset relative to the center
return $P = c \times \varphi_\theta(y_k)$ ▷Integral contribution

(Bresenham's [16] or Siddon's [4], [17], [18] methods) is to cast a ray through a medium composed of simple structures that form a partition of the domain, to compute the intersections between the ray and these structures, and to perform partial integrations over each intersection. Traditional methods correspond to the case when the basis generator φ takes a constant value and has a shape that coincides with the cells of the grid. In that case, $\mathcal{P}_\theta\{\varphi\}(y)$ is precisely the length of such intersections, to which ray tracing provides direct access.

In this work, we generalize the ray-tracing algorithm to handle any basis function even if the support overflows a cell while remaining exact. The challenge is to combine the domain decomposition (a uniform Cartesian grid allowing ray-tracing techniques) with an expansion into basis functions whose support is general and does not coincide with that of the cell [19], [20]. Higher-order basis functions allow us to improve image representations, which ultimately leads to better quality. The contributions of this paper are as follows.

1) We propose an efficient variant of ray tracing for the exact computation of the x-ray transform and its adjoint, able to handle overlapping basis functions. The algorithm can handle any projection geometry and computes $\mathcal{P}_{\theta_m}\{f\}(y_m)$ for any set of rays parameterized by $(\theta_m, y_m)_{m\le M}$.
2) We provide explicit expressions of $\mathcal{P}_\theta\{\varphi\}(y)$ for specific basis functions, which we take advantage of in our implementation.
3) We present image reconstruction experiments that showcase the benefits of higher-order basis functions in tomographic reconstruction problems.

In the sequel, we rely on the shorthand notation φ_θ to denote the x-ray transform at the projection angle θ of the basis generator φ, as shown in Fig. 2, with $y \in \mathbb{R}$ such that

$$\varphi_\theta : y \mapsto \mathcal{P}_\theta\{\varphi\}(y). \tag{5}$$

II. Generalized X-Ray Projections

We propose an efficient algorithm to compute the x-ray projection (4) for an arbitrary ray of angle θ and offset y using ray tracing. To streamline the exposition, we defer the derivations of closed-form expressions for $\varphi_\theta(y)$ to Section III-C.

A. Ray-Tracing Routine

Our geometric setup is as follows: the coefficients of f in the chosen basis are stored in an image array C and placed

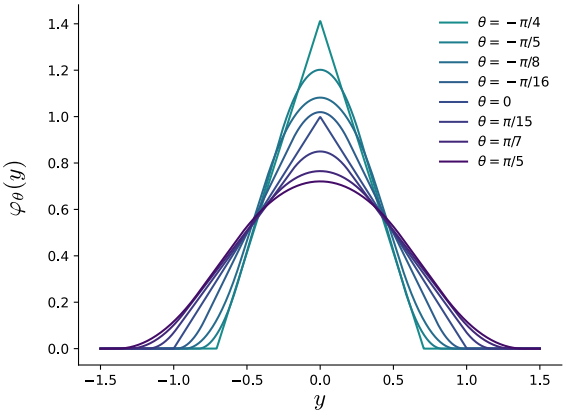

Fig. 2. Projected bases $y \mapsto \varphi_\theta(y)$. Here, φ is the 3-directional box-spline $\varphi_\theta^{\text{box}}$. The curves are indexed by the projection angle $\theta \in [-\pi/4, \pi/4)$.

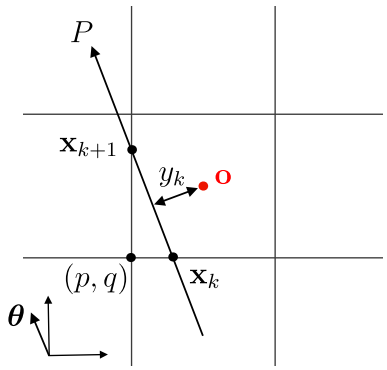

Fig. 3. Relevant quantities in Algorithm 1.

on a two-dimensional uniform Cartesian grid, as sketched in Fig. 4. This tiling allows us to efficiently compute the intersection points $\mathbf{x}_k$ between rays and cells. In our implementation, we sequentially compute such intersections with a ray-tracing routine by taking advantage of the computer-graphics library Dr.Jit [21]. Traditional pixel-based approaches output the sum of all the intersection lengths given by $c_k\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2$, where c_k is the value of the crossed pixel. This corresponds to the evaluation of (4) with φ being the rectangle function.

B. Proposed General Algorithm

In this work, we consider basis functions whose support extends over more than one cell. These generalized bases overlap and require us to revisit the traditional procedure. In order to compute (4) efficiently with overlapping basis functions, we embed the ray tracer in Algorithm 2 that also considers some non-intersected cells. The ray-tracing routine discussed in Section II-A efficiently computes the update "$\mathbf{x}_{k+1} = \text{Ray-trace}(\mathbf{x}_k, \theta)$", where the index k refers to the ray-tracing step. At each step k, as we cross an intersected cell, we employ Algorithm 1 (illustrated in Fig. 3) to compute the contribution of the basis function centered at this very cell. Then, the key point of our method is to complement this contribution with that of the basis functions centered at neighboring cells.

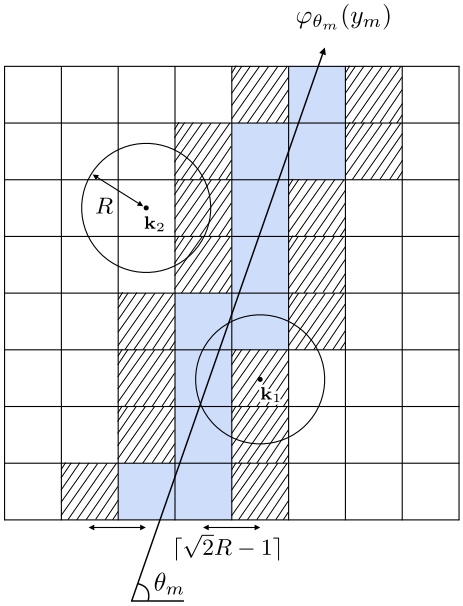

Fig. 4. Ray tracing when the basis generator has support included in a disk of radius R.

Algorithm 2: X-ray Projection With Overlapping Basis Functions via Ray Tracing.

Require: $C, \theta, \mathbf{x}_0$
▷Image, angle, first intersection point of ray tracing
while *ray tracing is ongoing* **do**
$\mathbf{x}_{k+1} = \text{Ray-trace}\ (\mathbf{x}_k, \theta)$ ▷ Ray-tracing update
$(p, q) = \lfloor \frac{\mathbf{x}_k + \mathbf{x}_{k+1}}{2} \rfloor$ ▷Indices of crossed cell
$P \leftarrow P + \text{Algorithm 1}\ (C, \mathbf{x}_k, \theta, (p, q))$
if $x_{k+1,1} \neq p$ **and** $x_{k,1} \neq p$ **then**
$(p_{\text{L}}, q_{\text{L}}) = (p - 1, q)$ ▷Indices of left cell
$P \leftarrow P + \text{Algorithm 1}\ (C, \mathbf{x}_k, \theta, (p_{\text{L}}, q_{\text{L}}))$
end if
if $x_{k+1,1} \neq p + 1$ **and** $x_{k,1} \neq p + 1$ **then**
$(p_{\text{R}}, q_{\text{R}}) = (p + 1, q)$ ▷Indices of right cell
$P \leftarrow P + \text{Algorithm 1}\ (C, \mathbf{x}_k, \theta, (p_{\text{R}}, q_{\text{R}}))$
end if
$k \leftarrow k + 1$
end while
return $\mathcal{P}_\theta\{f\}(y) = P$

We illustrate our approach in Fig. 4. While we evaluate the contribution of the basis functions centered at the cells that are crossed by the ray (shaded, identified via the ray-tracing routine), we also evaluate the contribution of the surrounding neighbors centered at the cells that are not crossed but still contribute to the line integral (hatched). The number of neighbors to evaluate on the left and right side at each ray tracing step is $\lceil\sqrt{2}R - 1\rceil$ according to Lemma 1, which is 1in this case. Two indices of basis functions are emphasized in Fig. 4: $\mathbf{k}_1$, which corresponds to a basis function that contributes to the line integral; and $\mathbf{k}_2$, which corresponds to a basis function that does not. In the

specific case where φ is a rectangle function that matches the shape of a cell, there is no neighbor to consider, and Algorithm 2 reduces to Siddon's method.

In Algorithm 2, we only take into account the case where $|\sin\theta| > 1/\sqrt{2}$ as in Fig. 4, namely, the "mainly vertical" rays where the left- and right-side neighbors are considered. For "mainly horizontal" rays ($|\cos\theta| \geq 1/\sqrt{2}$), upper and lower neighbors are considered instead. This case distinction on the value of θ is also part of traditional ray-tracing algorithms.

Algorithm 2 only includes the case where we consider immediate neighbors, but we have also implemented a version of the approach that is extended to larger basis functions by considering additional neighbors. Lemma 1 provides the number of neighbors to consider when one can bound the shape of the support by either a disk or an octagon. The computational cost of the algorithm depends on this number of neighbors, as discussed in Section IV-D.

Lemma 1: Let $\varphi \in L^1(\mathbb{R}^2)$ be a compactly supported basis generator. Let $f = \sum_{\mathbf{k}} c_{\mathbf{k}} \varphi(\cdot - \mathbf{k})$ and consider a line integral along a "mainly vertical" (horizontal, respectively) line. The basis functions that contribute to this line integral are among those whose centers lie within the grid cells that are at most K *horizontal* (*vertical*, respectively) neighbors away from the intersected cells.

1) If the support of φ is included in a disk of radius R, then

$$K = \left\lceil \sqrt{2}R - 1 \right\rceil. \tag{6}$$

2) If the support of φ is included in an octagonal shape of girth L as is Fig. 11, then

$$K = \lfloor L/2 \rfloor. \tag{7}$$

The proof of Lemma 1 is provided in Appendix.

In Algorithm 2 and its variants with additional neighbors, immediate neighbors must be evaluated under a specific condition. Indeed, let (p, q) be the index of the crossed cell and $\mathbf{x}_k = (x_{k,1}, x_{k,2})$ the entry line-cell intersection point at iteration k of ray tracing. The determination of the contribution is as follows:

- If (i) $x_{k,1}$ equals p or (ii) $x_{k+1,1}$ equals p, then the left neighbor is not evaluated. It is the case of the dotted ray of Fig. 5.
- If (iii) $x_{k,1}$ equals $p+1$ or (iv) $x_{k+1,1}$ equals $p+1$, then the right neighbor is not evaluated. It is the case of the dashed ray of Fig. 5.

One should *not* evaluate the contribution of a neighboring cell (I) if the ray just crossed this very cell at iteration $(k-1)$ or (II) if the ray will cross this very cell at iteration $(k+1)$. For "mainly horizontal" rays, the conditions concern $x_{k,2}$ and the vertical cell index q.

The outcome of Algorithm 2 is the x-ray projection of a linear combination of translations of a basis generator defined on a regular grid for a single ray parameterized by (θ, y). Instead of y, we equivalently take $\mathbf{x}_0$ as an input in the algorithm—it is the first intersection point between the ray and the grid. Our implementation loops in parallel over a collection of arbitrarily parametrized lines. The back-projection operator is the adjoint of the x-ray transform and has been implemented as well, and exactly matched with the forward projection.

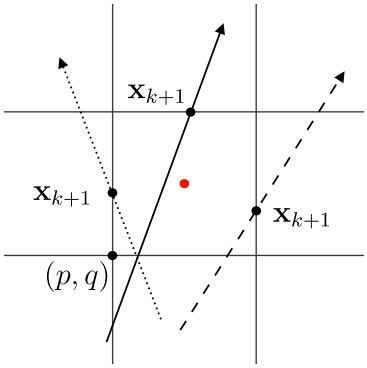

Fig. 5. Conditions for neighbor evaluation: one does not evaluate the contribution of the left- (right-, respectively) neighbor coefficient for the dotted (dashed, respectively) ray.

III. Explicit Projections

An efficient scheme to evaluate φ_θ is needed to make our method practical. This is addressed below with the derivation of closed-form expressions for useful basis generators.

A. Separable Functions

The x-ray projection of a 2D separable function is a starting point to apprehend our calculus. Let $\varphi \in L^1(\mathbb{R}^2)$ be a 2D separable function, in the sense that there exist functions ϕ_1 and ϕ_2 in $L^1(\mathbb{R})$ such that

$$\forall (x_1, x_2) \in \mathbb{R}^2, \quad \varphi(x_1, x_2) = \phi_1(x_1)\phi_2(x_2). \tag{8}$$

For the moment, consider only two directions and let $(\mathbf{e}_1, \mathbf{e}_2)$ be the Cartesian basis. Then, Proposition 1 with $D = 2$ states that, for $y \in \mathbb{R}$,

$$\varphi_\theta(y) = \left(\frac{1}{|\cos\theta|} \phi_1 \left(\frac{\cdot}{\cos\theta} \right) * \frac{1}{|\sin\theta|} \phi_2 \left(\frac{\cdot}{\sin\theta} \right) \right)(y). \tag{9}$$

Let $\hat{\phi}_d$ denote the Fourier transform of ϕ_d. If $\hat{\phi}_d(0) = 1$, then we can replace the term $\frac{1}{|a|}\phi_d\left(\frac{\cdot}{a}\right)$ by a Dirac distribution δ when $a = 0$. We manage here to write the x-ray transform of φ as a simple convolution, which will lead to explicit expressions of the latter in specific cases (Section III-C) and to efficient computations as a consequence.

B. Generalization With D Directions

Consider now the case when the Fourier transform of the basis generator $\varphi \in L^1(\mathbb{R}^2)$ can be expressed as a product of D terms instead of just two. Let $\hat{f}$ denote the Fourier transform of some function f, either in $L^1(\mathbb{R}^2)$ or in $L^1(\mathbb{R})$.

Then, for any projection angle, it is possible to express the x-ray projection of φ as a 1D convolution of D rescaled atoms.

Proposition 1: Let $\varphi \in L^1(\mathbb{R}^2)$ be such that its Fourier transform $\hat{\varphi} : \mathbb{R}^2 \mapsto \mathbb{C}$ can be written as

$$\forall \boldsymbol{\xi} \in \mathbb{R}^2, \;\; \hat{\varphi}(\boldsymbol{\xi}) = \prod_{d=1}^{D} \hat{\phi}_d \left(\langle \boldsymbol{\xi}, \mathbf{u}_d \rangle \right), \tag{10}$$

where $\hat{\phi}_d : \mathbb{R} \mapsto \mathbb{C}$ is the Fourier transform of the function $\phi_d : \mathbb{R} \mapsto \mathbb{R}$, and $\mathbf{u}_d \in \mathbb{R}^2$ is a corresponding direction.

Then, we have that the x-ray transform of φ along the projection direction $\boldsymbol{\theta} \in \mathbb{S}^1(\mathbb{R})$ is

$$\forall y \in \mathbb{R}, \quad \varphi_\theta(y) = \left(\tilde{\phi}_1 * \cdots * \tilde{\phi}_D \right)(y), \tag{11}$$

with

$$\tilde{\phi}_d = \begin{cases} \frac{1}{|\langle \boldsymbol{\theta}, \mathbf{u}_d \rangle|} \phi_d \left(\frac{\cdot}{\langle \boldsymbol{\theta}, \mathbf{u}_d \rangle} \right), & \langle \boldsymbol{\theta}, \mathbf{u}_d \rangle \neq 0 \\ \delta, & \langle \boldsymbol{\theta}, \mathbf{u}_d \rangle = 0. \end{cases} \tag{12}$$

The proof is provided in Appendix.

C. Derivation for Specific Basis Functions

The authors of [22] present a specific case of Proposition 1 when $D = 2$ and ϕ_d is a B-spline. Likewise, choosing $\phi_1, \ldots, \phi_D$ as rectangles recovers the case of D-directional box-splines as in [10]. In contrast, our formulation in Proposition 1 is more general: it applies to any function that is separable in the Fourier domain along arbitrary directions. In this section, we present cases of particular interest, where we now derive some novel explicit expressions of φ_θ for box-splines and B-splines. These expressions are not found in prior work and allow for the efficient computation of $\varphi_\theta(y)$ in Algorithm 1.

Although box-splines are not separable in space, they can be factorized along some directions in the Fourier domain. We now derive a closed-form expression of the 3-directional box-spline with directions $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$, where $(\mathbf{u}_1, \mathbf{u}_2) = (\mathbf{e}_1, \mathbf{e}_2)$ forms the Cartesian basis of $\mathbb{R}^2$, and $\mathbf{u}_3 = \mathbf{e}_1 + \mathbf{e}_2$. If ϕ_1, ϕ_2, ϕ_3 are centered unit rectangles, then (10) holds true with $D = 3$ and

$$\forall \boldsymbol{\xi} \in \mathbb{R}^2, \;\; \widehat{\varphi^{\text{box}}}(\boldsymbol{\xi}) = \prod_{d=1}^{3} \operatorname{sinc} \left(\frac{\langle \boldsymbol{\xi}, \mathbf{u}_d \rangle}{2\pi} \right). \tag{13}$$

Let $\boldsymbol{\alpha} \in \mathbb{R}^6$ denote the vetor

$$\begin{aligned} \boldsymbol{\alpha} = \; & (0, \; \sin\theta, 2\sin\theta + \cos\theta, \; \cos\theta, \\ & \sin\theta + 2\cos\theta, \; 2(\sin\theta + \cos\theta)). \end{aligned} \tag{14}$$

Then, by simplifying the convolution product (11) in the case of rectangle functions as suggested in [10], we are able to express the x-ray transform of this 3-directional box-spline as

$$\varphi_\theta^{\text{box}}(y) = \frac{-1}{\alpha_1 \alpha_3 \alpha_5} \sum_{n=1}^{6} (-1)^n \, (y - \alpha_n)_+^2, \tag{15}$$

where $y \in \mathbb{R}$, α_n is the nth component of $\boldsymbol{\alpha}$, and $(x)_+ := \max(x, 0)$. In the case where $\theta \in (0, \pi/4)$, we can perform the expansion of the sum in (15) and collect the polynomial terms given by

$$\varphi_\theta^{\text{box}}(y) = \begin{cases} 0, & y < \alpha_0 \\ \frac{y^2}{\alpha_1 \alpha_3 \alpha_5}, & \alpha_0 \leq y < \alpha_1 \\ \frac{2y - \alpha_1}{\alpha_3 \alpha_5}, & \alpha_1 \leq y < \alpha_5 \\ \frac{-y^2 + \alpha_5 y - 1}{\alpha_1 \alpha_3 \alpha_5}, & \alpha_5 \leq y < \alpha_2 \\ \frac{-2y + 4\alpha_3 + 3\alpha_1}{\alpha_3 \alpha_5}, & \alpha_2 \leq y < \alpha_4 \\ \frac{(y - 2\alpha_1 - 2\alpha_3)^2}{\alpha_1 \alpha_3 \alpha_5}, & \alpha_4 \leq y < \alpha_5 \\ 0, & y \geq \alpha_5. \end{cases} \tag{16}$$

This expression leads to the family of functions shown in Fig. 2. We derive similar results for the other angle ranges by leveraging inherent symmetries of the box-splines.

A particular case of box-splines is the case of separable tensor-product B-splines. Let β^n denote the univariate B-spline of degree n [23]. Then, for all $(x_1, x_2) \in \mathbb{R}^2$,

$$\varphi(x_1, x_2) = \beta^n(x_1) \beta^n(x_2). \tag{17}$$

This corresponds to a $2(n+1)$-directional box-spline with $n+1$ directions along $\mathbf{e}_1$ and $n+1$ directions along $\mathbf{e}_2$. We derived a closed-form expression for its x-ray transform in a similar way to (15) for $n = 2$. It is

$$\begin{aligned} \varphi_\theta(y) = \frac{1}{6\,\alpha_1^2 \alpha_2^2} \big(& y_+^3 - 2\,(y - \alpha_1)_+^3 - 2\,(y - \alpha_2)_+^3 \\ & + (y - 2\alpha_1)_+^3 + 4\,(y - \alpha_1 - \alpha_2)_+^3 + (y - 2\alpha_2)_+^3 \\ & - 2\,(y - \alpha_3)_+^3 - 2\,(y - \alpha_4)_+^3 + (y - \alpha_5)_+^3 \big). \end{aligned} \tag{18}$$

More generally, x-ray projections of any D-directional box-spline can be computed exactly since the convolution of dilated rectangle functions is known.

IV. Image Reconstruction

A. Formulation of the Inverse Problem

In tomography, the corrupted measurements $p \in L^2(\mathbb{R}^2)$ are x-ray projections of the signal of interest $f \in L^2(\mathbb{R}^2)$ and the corruption ε is commonly assumed to be an additive white Gaussian noise with variance σ^2. Our model is

$$p(\theta, y) = \mathcal{P}_\theta\{f\}(y) + \varepsilon(\theta, y). \tag{19}$$

Since real measurements are acquired via discrete sensors and detectors, the measurements p are sampled which yields $\mathbf{p} \in \mathbb{R}^M$ such that

$$\mathbf{p} = \left(p\left(\theta_m, y_m\right) \right)_{m=1}^{M}, \tag{20}$$

where the set $((\theta_m, y_m))_{m=1}^M$ encodes the acquisition geometry. Each component corresponds to a ray, as in (1), which yields

$$\mathbf{p} = \begin{bmatrix} p_1 \\ \vdots \\ p_M \end{bmatrix} = \begin{bmatrix} \mathcal{P}_{\theta_1}\{f\}(y_1) \\ \vdots \\ \mathcal{P}_{\theta_M}\{f\}(y_M) \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_M \end{bmatrix}. \tag{21}$$

From this finite number of measurements, we seek to recover a finite number of coefficients of f in some basis as in (4). By invoking the linearity of $\mathcal{P}$, one seeks to recover $\mathbf{c} \in \mathbb{R}^{N^2}$ such that

$$\mathbf{p} = \mathbf{H}_\varphi \mathbf{c} + \boldsymbol{\varepsilon}, \tag{22}$$

where $\mathbf{H}_\varphi \in \mathbb{R}^{M \times N^2}$ is the discretized version of the x-ray operator. More precisely, we have that

$$(\mathbf{H}_\varphi)_{m,\mathbf{k}} = \mathcal{P}_{\theta_m}\{\varphi(\cdot - \mathbf{k})\}(y_m) = \varphi_{\theta_m}(\tilde{y}_{m,\mathbf{k}}), \tag{23}$$

where the entries depend on the choice of the generator φ and $\tilde{y}_{m,\mathbf{k}} = (y_m - \langle \mathbf{k}, \boldsymbol{\theta}^\perp \rangle)$. We now face the task of finding the solution of a maximum-likelihood optimization problem to obtain

$$\mathbf{c}^* = \arg\min_{\mathbf{c} \in \mathbb{R}^{N^2}} \frac{1}{2\sigma^2} \|\mathbf{p} - \mathbf{H}_\varphi \mathbf{c}\|_2^2 + \lambda \mathbf{R}_\varphi(\mathbf{c}), \tag{24}$$

where $\mathbf{R}_\varphi$ is a regularization term that enforces desirable properties of the solution and depends on the choice of φ, as studied by the authors of [11], [24]. The parameter $\lambda \in \mathbb{R}_{\geq 0}$ controls the strength of the regularizer.

The recovery of the coefficients $\mathbf{c}^*$ of the target image from sampled measurements provides access to the continuous-domain signal defined by (4), which enables a reconstruction at any desired sampling rate. If the basis functions are uniform tiles matching the reconstruction grid (i.e., splines of degree 0, a.k.a. "pixels"), then the underlying continuous-domain image is piecewise-constant, thereby offering no enhancement in quality at a higher sampling rate. The projection operator $\mathbf{H}_\varphi$ and its adjoint are implemented in a matrix-free format in accordance with Algorithm 2. We demonstrate in Section IV-B the effectiveness of our implementation in the context of inverse problems.

B. Experiments with Synthetic Data

We now conduct experiments that compare the reconstruction performance of spline-based operators with that of traditional pixel-based ones. In order to single out the impact of the discretizations, we focus on scenarios where the problem is sufficiently well posed to avoid the need for regularization (i.e. $\lambda = 0$ in (24)). Specifically, we assume a regime in which M is large, with many angles and many offsets, and in which there is no missing cone. For a conventional geometry where the angles and offsets are uniformly spaced, it has been established that this regime occurs when one measures at least $\pi N/2$ angles with at least N offsets, where the reconstruction grid is of size $(N \times N)$. This is known as the Crowther criterion [25], [26]. Our own experiments in Fig. 6 confirm that about $\pi N/2$ angles are needed in the traditional pixel-based approach, which we round up to $2N$ in the sequel. For the basis functions of higher order but larger support proposed in this paper, however, we shall see that even fewer angles are required.

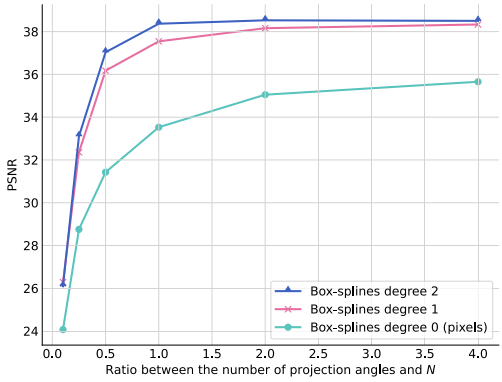

Fig. 6. PSNR of the reconstructions in terms of the number of projection angles with a grid of size $N = 500$.

We perform image reconstructions as follows: we choose the ground truth image $\mathbf{x}_{\text{GT}} \in \mathbb{R}^{N^2}$ from the public dataset [27] of lung CT medical images, and we upsample them with high resolution $N = 3000$ using cubic B-splines. We acquire from $\mathbf{x}_{\text{GT}}$ the projections $\mathbf{p} \in \mathbb{R}^{2N_{\text{down}} \times N_{\text{down}}}$. Experiments with parallel-beam and cone-beam projection geometries have been conducted. In order to avoid committing an inverse crime with our model, the data $\mathbf{p}$ are acquired with the Astra-toolbox package under the `'line'` (`'line_fanflat,'` respectively) projector with the `'parallel'` (`'fanflat,'` respectively) projection geometry. However, even though we add noise to the projections ($\sigma^2 = 10^{-3}$), we commit an inverse crime with Astra reconstructions. The solution of the basic optimization problem (24) is obtained via conjugate-gradient descent to recover the coefficients $\mathbf{c}^* \in \mathbb{R}^{N_{\text{down}}^2}$ in a reconstruction grid of size $(N_{\text{down}} \times N_{\text{down}})$. Using $\mathbf{c}^*$, we sample the underlying continuous signal (4) on a finer grid, yielding $\mathbf{x}^* \in \mathbb{R}^{N^2}$.

The reconstruction results are shown in Fig. 7 for the cone-beam setup. Except at the coarsest grid sizes, the Astra reconstructions are visually indistinguishable from those of box-splines of degree 0 (pixels), while box-splines of degree 1 or 2 provide a more accurate representation of the target image. In Fig. 9, both the peak signal-to-noise ratio and the structural similarity are reported as curves. Numerical values are provided in Table II. These clearly demonstrate that our model, based on higher-degree spline representations, consistently outperforms the traditional pixel-based approach (degree 0), with box-splines of degree 2 achieving the best overall reconstruction quality. The residual images in Fig. 8 show the absolute difference between the reconstruction and the ground truth, and further illustrate the improvements achieved by higher-degree box-spline models. (We only kept Astra's `'fanflat'` model and our model of degree 2 for clarity.)

C. Experiments With Real Tomographic Data

We performed image reconstructions with real cone-beam data. The acquisition device is a micro CT-scan with one source

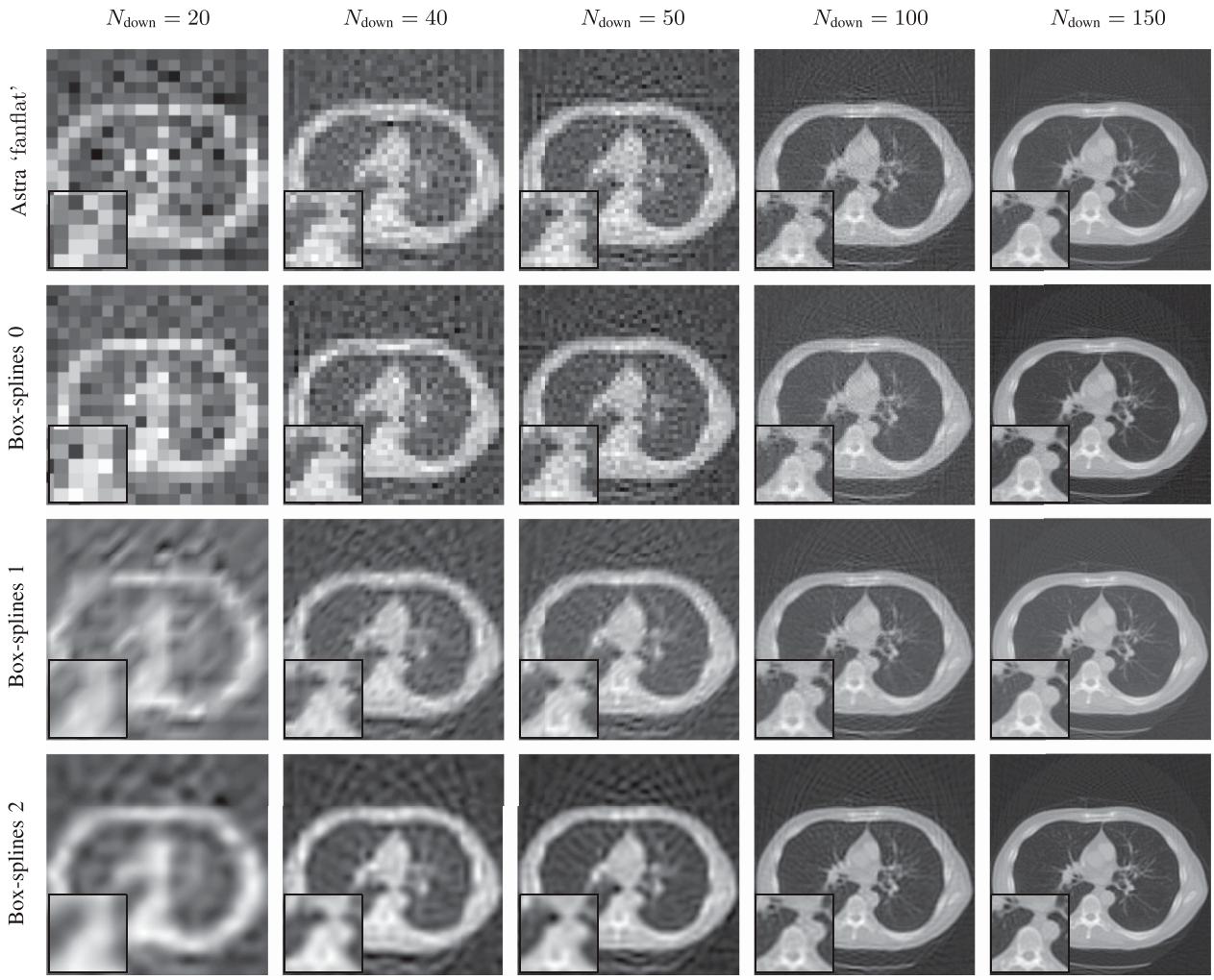

Fig. 7. Cone-beam reconstructions by Algorithm 2 in terms of grid size and model, with conjugate-gradient descent (30 iterations). Column: reconstructed signals with N_{down}^2 recovered coefficients. Row: Astra reconstruction and box-spline models of degree 0 (pixels), 1, and 2.

TABLE II
PSNR AND SSIM FOR THE RECONSTRUCTIONS OF FIG. 7 IN TERMS OF BASELINE METHOD (ASTRA AND BOX-SPLINES OF DEGREE 0) AND OUR BOX-SPLINES MODEL (DEGREES 1 AND 2), ACROSS GRID SIZE. BOLD: BEST PSNR AND SSIM IN EACH ROW

N_{down}	Astra 'fanflat'	Box-splines 0	Box-splines 1	Box-splines 2
50	(19.97, 0.58)	(18.29, 0.54)	(**20.68**, 0.70)	(20.46, **0.74**)
100	(24.81, 0.73)	(23.21, 0.71)	(25.74, 0.84)	**(26.05, 0.87)**
150	(27.54, 0.82)	(26.01, 0.81)	(27.97, 0.90)	**(28.62, 0.93)**
250	(32.08, 0.90)	(29.78, 0.89)	(32.50, 0.96)	**(32.85, 0.97)**
375	(34.90, 0.93)	(32.92, 0.93)	(35.71, 0.98)	**(35.96, 0.98)**
1000	(39.46, 0.97)	(38.74, 0.97)	(40.87, 0.99)	**(41.35, 0.99)**

and a flat detector screen, aligned with the rotation table of the sample between them. This experimental setup corresponds to a cone-beam geometry. The detector screen is composed of receptors of lateral size 0.127 mm; the source-to-detector distance is 765.7 mm and the source-to-object distance is 96.46 mm. We calibrate the center of rotation of the sample, which is experimentally imperfectly matched with the center of the sample. We achieve this correction through a grid-search algorithm. In this case, the calibration leads to a shift of the center of rotation by (-0.292) mm in the direction orthogonal to the source-detector axis. We have access to 800 projections and seek to reconstruct the center slice of the volume in a reconstruction

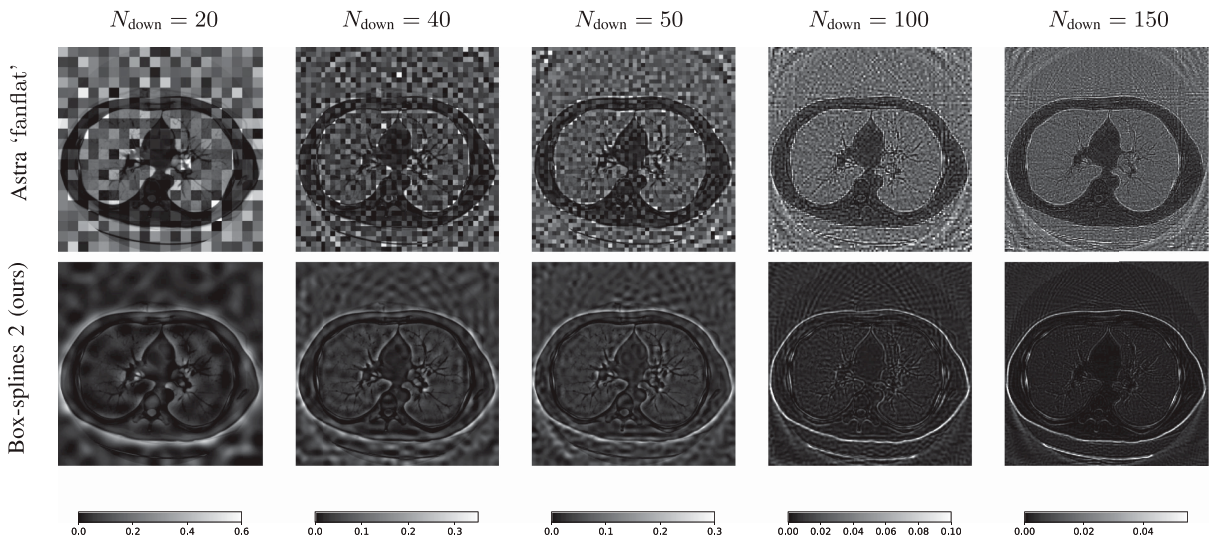

Fig. 8. Absolute difference images between the ground truth and the most relevant reconstructions from Fig. 7.

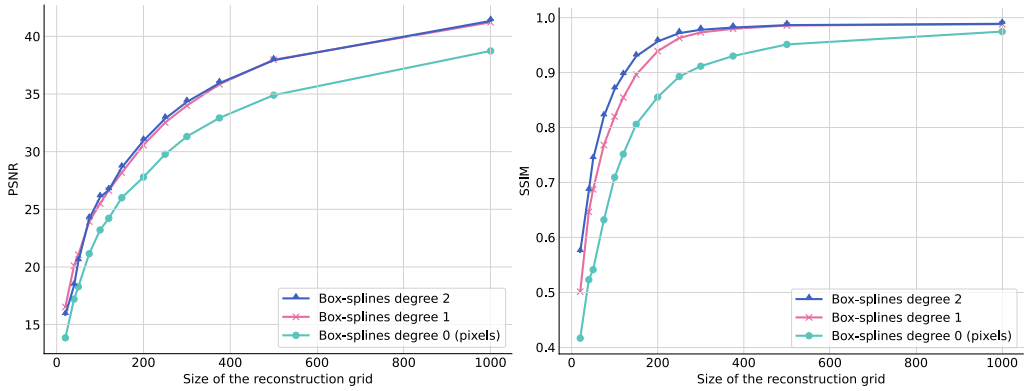

Fig. 9. PSNR and SSIM of the reconstructions in terms of grid size for the cone-beam setup.

grid of size (728×728). The imaged sample is a collection of rocks inside a cylindrical box. We demonstrate in Fig. 10 that our model provides superior image quality and artifact reduction compared to pixel-based ray-tracing baselines in two scenarios: with the sinogram of 800 views; and with a quarter of the measurements where only 200 views are retained. We compare our model of splines of degree 2 with two baselines: Astra's pixel-based `'line_fanflat'` model; and our model of splines of degree 0 (pixels), which yield very similar reconstructions. The reconstruction algorithm remains identical across all cases; only the underlying model differs.

D. Runtime Evaluation and Discussion

Sections IV-B and IV-C evaluated the performance of our algorithm in terms of reconstruction quality. We now evaluate the performance of the algorithm in terms of computation speed. As expected, the computation times of our algorithm, shown in Tables III and IV, does not depend on the projection geometry. By contrast, the performance of Astra, which is highly optimized for parallel- and cone-beam geometries, drops significantly for arbitrary geometries. This makes our algorithm particularly well suited for imaging modalities such as positron electron tomography [28], plasma diagnostic imaging, or additive volumetric manufacturing where the geometry is not structured. GPU benchmarks were carried out on an NVIDIA RTX A5000 with five warm-up runs. The reported times represent the average of ten executions, and provided by the `cupyx.benchmark` module. Our speedup on CPU is due both to the just-in-time compilation of the ray-tracer and to the fine-grain parallelism that is performed. Each ray is treated independently, which allows us to distribute the computation to multiple cores.

We report in Tables III and IV the computation times for both Astra and our spline-based model of degree 0 (pixel baseline), 1 and 2. On average, the runtime increases by a factor of 2.6 for degree 1 and 4.1 for degree 2 relative to the pixel baseline. These averages are computed across all setups of Table III and include both projection and back-projection operations on the GPU. The use of box-splines of degree 2 provides the best tradeoff between quality and runtime. They have the same approximation properties as the separable B-splines of degree 2 (presented in Table V of the Appendix) but have a more compact support

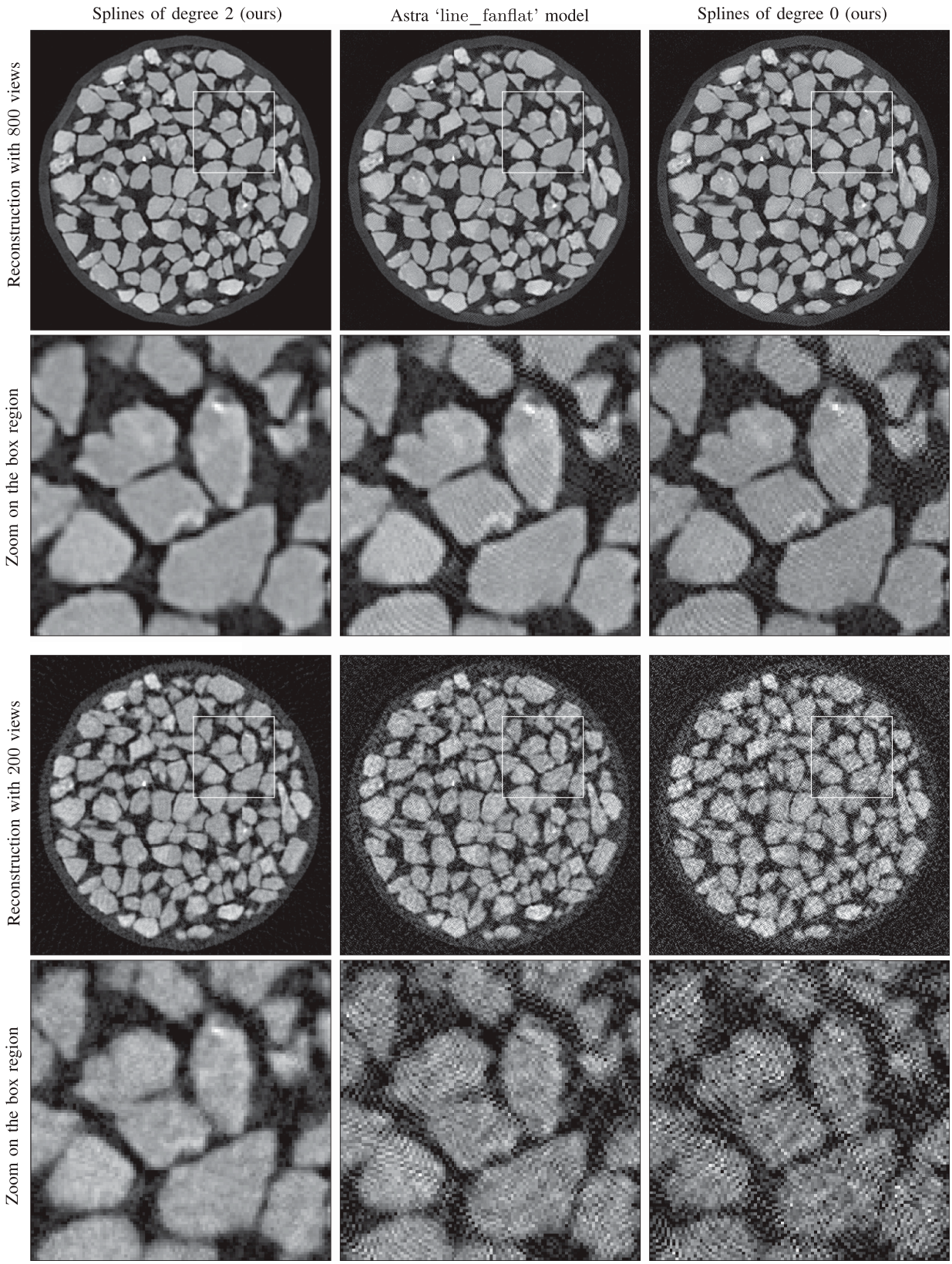

Fig. 10. Reconstructions with real tomographic data for different models, with conjugate gradient descent (30 iterations). Astra's `'line_fanflat'` model and our pixel-based model (splines of degree 0) are shown as a baseline for comparison with our model of splines of degree 2. First two rows: reconstructions with 800 projection angles. Last two rows: reconstructions with 200 projection angles.

TABLE III
GPU EXECUTION TIMES (IN MILLISECONDS) FOR PROJECTION AND BACK-PROJECTION OPERATIONS UNDER ARBITRARY AND CONE-BEAM GEOMETRIES. THE OPERATIONS ARE PERFORMED WITH $N \times N$ RAYS ON AN IMAGE OF SIZE $(N \times N)$

GPU times (ms)	Arbitrary geometry				Cone-beam geometry			
	$N = 250$	$N = 500$	$N = 750$	$N = 1000$	$N = 250$	$N = 500$	$N = 750$	$N = 1000$
ASTRA								
Projection	206	821	1850	3350	2.60	6.20	11.9	20.0
Back-projection	236	1050	2890	6490	1.80	4.80	9.20	17.5
Ours (degree 0)								
Projection	0.610	2.17	6.89	16.6	0.640	2.41	6.69	16.7
Back-projection	1.48	5.16	14.8	33.9	1.14	5.90	14.8	33.9
Ours (degree 1)								
Projection	2.23	5.08	16.8	38.7	1.50	5.26	16.9	38.8
Back-projection	3.85	16.5	43.3	93.5	3.78	16.5	43.2	93.2
Ours (degree 2)								
Projection	2.74	11.1	34.2	71.7	2.89	11.1	33.5	71.5
Back-projection	4.69	18.7	51.5	111	5.05	18.7	51.3	115

TABLE IV
CPU EXECUTION TIMES (IN MILLISECONDS) FOR PROJECTION AND BACK-PROJECTION OPERATIONS UNDER ARBITRARY AND CONE-BEAM GEOMETRIES. THE OPERATIONS ARE PERFORMED WITH $N \times N$ RAYS ON AN IMAGE OF SIZE $(N \times N)$

CPU times (ms)	Arbitrary geometry				Cone-beam geometry			
	$N = 250$	$N = 500$	$N = 750$	$N = 1000$	$N = 250$	$N = 500$	$N = 750$	$N = 1000$
ASTRA								
Projection	296	1620	4590	9820	110	840	2910	6860
Back-projection	297	1610	4590	9880	110	840	2890	6840
Ours (degree 0)								
Projection	1.53	8.14	32.0	77.9	1.55	8.13	30.8	72.3
Back-projection	12.6	58.2	288	263	13.7	59.5	254	385
Ours (degree 1)								
Projection	5.00	36.7	119	264	5.11	35.9	118	271
Back-projection	21.5	127	541	1280	23.3	124	484	1380
Ours (degree 2)								
Projection	10.7	127	195	454	8.54	128	192	448
Back-projection	24.7	148	579	1450	26.3	152	564	1490

that extends over three cells only, against five for the separable B-splines of the same degree. Their support corresponds to that of Fig. 11 with $K = 1$, an optimal configuration for our algorithm as it corresponds to the largest support for K neighbor evaluations. Splines of degree higher than 2 do not provide a sufficient gain in quality to justify their use, as saturation with respect to the degree is observed in our results (Table II and Fig. 9).

V. CONCLUSION

We presented a framework to compute the x-ray transform operator for 2D signals decomposed into basis functions, specifically, box-splines and separable B-splines. We derived closed-form expressions of their x-ray projections for arbitrary integration lines. Using these analytical derivations, our algorithm can tackle the case of overlapping basis functions with a neighbor-based approach in a scalable and efficient way. Our projector and back-projector form matched adjoint pairs and allow for any projection geometry without affecting performance, which makes our algorithm competitive with open-source packages. The proposed method uses ray tracing by taking advantage of a computer-graphics library. In the context of image reconstruction, coarse- and fine-grid regimes have both been studied. We obtain convincing experimental results for continuous-domain inverse problems, where basis functions of higher-degree lead to better reconstructions. From our experiments, we recommend the box-spline of degree 2 as the basis that provides the best tradeoff between runtime and quality. Our methodology can be naturally extended to 3D settings, which we leave for future work. The code of our implementation[1] will be made publicly available along with the scripts that reproduce the figures of the paper.

APPENDIX

Proof of Lemma 1: We provide a proof of (1) and note that (2) can be proven in the same manner.

Let $(y, \theta) \in \mathbb{R}^2$ represent the parameters of a "mainly vertical" ray. Consider a basis function $\varphi(\cdot - \mathbf{k})$ that contributes to the integral, for some $\mathbf{k} \in \Omega$. Without loss of generality, we can assume that $\mathbf{k} = \mathbf{0}$, up to a shift in the parameter y to $(y - \langle \mathbf{k}, \boldsymbol{\theta}^{\perp} \rangle)$. Since φ is compactly supported inside a disk of radius R, it holds that

$$\forall \mathbf{x} \in \mathbb{R}^2 \text{ s.t. } \|\mathbf{x}\|_2 \geq R, \quad \varphi(\mathbf{x}) = 0. \tag{25}$$

[1] https://github.com/HaouchatY/rt_tomo_basis_functions/

TABLE V
PSNR AND SSIM IN TERMS OF MODEL (SEPARABLE B-SPLINES OF DEGREE 0, 1, AND 2) AND GRID SIZE FOR CONE-BEAM RECONSTRUCTIONS. BOLD: BEST PSNR AND SSIM IN EACH ROW

N_{down}	B-splines 0	B-splines 1	B-splines 2
50	(18.29, 0.54)	(**21.12**, 0.70)	(20.50, **0.74**)
100	(23.21, 0.71)	(25.80, 0.84)	(**26.05, 0.87**)
150	(26.01, 0.81)	(28.27, 0.91)	(**28.63, 0.93**)
250	(29.78, 0.89)	(32.64, 0.96)	(**32.87, 0.97**)
375	(32.92, 0.93)	(35.86, 0.98)	(**35.94, 0.98**)
1000	(38.74, 0.98)	(41.10, 0.99)	(**41.34, 0.99**)

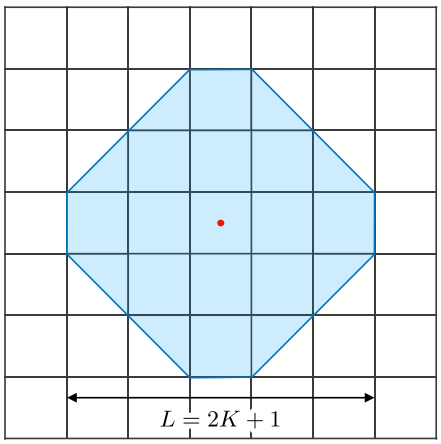

Fig. 11. Octagonal shape of size L used in Lemma 1. It corresponds to the largest support for K neighbor evaluations.

Since the ray is "mainly vertical," it intersects the horizontal axis directed by $\mathbf{e}_1$ at the horizontal component

$$\gamma(y, \theta) = y / \sin\theta, \tag{26}$$

where $|\sin\theta| \geq 1/\sqrt{2}$. From (1), we note that y represents the Euclidean distance from the ray to the origin. It follows that

$$|\gamma(y, \theta)| \leq |\sqrt{2}y| \leq \sqrt{2}R. \tag{27}$$

The last inequality holds because, if the ray crosses the support of φ, then y must be at most R, as stated in (25). Now, suppose that the domain is discretized into a uniform grid with basis vectors $(\mathbf{e}_1, \mathbf{e}_2)$ and stepsize 1. Then, the index along $\mathbf{e}_1$ of the cell containing the intersection of the ray with the horizontal axis is bounded by $\lceil \sqrt{2}R - 1 \rceil$. Consequently, the ray will cross one index in the set

$$\{\mathbf{k} - \lceil \sqrt{2}R - 1 \rceil \mathbf{e}_1, \ldots, \mathbf{k} + \lceil \sqrt{2}R - 1 \rceil \mathbf{e}_1\}. \tag{28}$$

Conversely, when we evaluate the contribution of the cells up to $\lceil \sqrt{2}R - 1 \rceil$ horizontal neighbors away from those directly crossed by the ray, the basis function centered at $\mathbf{k}$ will necessarily be included in the computation. ■

Proof of Proposition 1: Let φ be a function in $L^1(\mathbb{R}^2)$ that verifies (10). The Fourier-slice theorem [29] states that the Fourier transform of the x-ray projection in a direction $\boldsymbol{\theta}$ of φ equals the 1-dimensional slice through the origin of $\hat{\varphi}$ in the same direction. This is formalized as

$$\begin{aligned} \forall \xi \in \mathbb{R}, \quad \hat{\varphi}_\theta(\xi) &= \hat{\varphi}(\xi \boldsymbol{\theta}) \\ &= \prod_{d=1}^{D} \hat{\phi}_d(\xi \langle \boldsymbol{\theta}, \mathbf{u}_d \rangle), \end{aligned} \tag{29}$$

where φ_θ is as in (5). Here, each term of the Fourier factorization is a dilation of $\hat{\phi}_d$ by a factor $\langle \boldsymbol{\theta}, \mathbf{u}_d \rangle$. When $\langle \boldsymbol{\theta}, \mathbf{u}_d \rangle = 0$, the dth term is the constant $\hat{\phi}_d(0)$. Otherwise, the dilation property of the Fourier transform states that

$$\forall \xi \in \mathbb{R}, \quad \hat{f}(\lambda \xi) = \frac{1}{|\lambda|} \widehat{f_{\lambda^{-1}}}(\xi), \tag{30}$$

where $f_{\lambda^{-1}}(x) := f\left(\frac{x}{\lambda}\right)$ with $\lambda \neq 0$.

The proof is finally completed by injecting (30) into (29) with $f = \phi_d$. We conclude using the Fourier-convolution theorem. ■

ACKNOWLEDGMENT

We acknowledge the ENAC Interdisciplinary Platform for X-ray micro-tomography (PIXE) for providing the real CT data used in our experiments.

REFERENCES

[1] W. van Aarle et al., "Fast and flexible X-ray tomography using the Astra toolbox," *Opt. Exp.*, vol. 24, no. 22, pp. 25129–25147, Oct. 2016, doi: 10.1364/OE.24.025129.
[2] A. Biguri, M. Dosanjh, S. Hancock, and M. Soleimani, "TIGRE: A Matlab-GPU toolbox for CBCT image reconstruction," *Biomed. Phys. Eng. Exp.*, vol. 2, no. 5, Sep. 2016, Art. no. 055010, doi: 10.1088/2057-1976/2/5/055010.
[3] S. Rit et al., "The reconstruction toolkit (RTK), an open-source cone-beam CT reconstruction toolkit based on the insight toolkit (ITK)," *J. Phys., Conf. Ser.*, vol. 489, May 2013, Art. no. 012079, doi: 10.1088/1742-6596/489/1/012079.
[4] R. L. Siddon, "Fast calculation of the exact radiological path for a three-dimensional CT array," *Med. Phys.*, vol. 12, no. 2, pp. 252–255, 1985.
[5] G. Han, Z. Liang, and J. You, "A fast ray-tracing technique for TCT and ECT studies," in *Proc. IEEE Nucl. Sci. Symp. Conf. Record. Nucl. Sci. Symp. Med. Imag. Conf.*, Oct. 1999, vol. 3, pp. 1515–1518, doi: 10.1109/NSSMIC.1999.842846.
[6] P. M. Joseph, "An improved algorithm for reprojecting rays through pixel images," *IEEE Trans. Med. Imag.*, vol. TMI-1, no. 3, pp. 192–196, Nov. 1982, doi: 10.1109/TMI.1982.4307572.
[7] Y. Long, J. A. Fessler, and J. M. 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.
[8] B. De Man and S. Basu, "Distance-driven projection and backprojection," in *Proc. IEEE Nucl. Sci. Symp. Conf. Rec.*, Nov. 2002, vol. 3, pp. 1477–1480, doi: 10.1109/NSSMIC.2002.1239600.
[9] K. Zhang and A. Entezari, "A convolutional framework for forward and back-projection in fan-beam geometry," in *Proc. IEEE 16th Int. Symp. Biomed. Imag.*, Apr. 2019, pp. 1455–1458, doi: 10.1109/ISBI.2019.8759285.
[10] 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.
[11] M. Pourya, Y. Haouchat, and M. Unser, "A continuous-domain solution for computed tomography with Hessian total-variation regularization," in *Proc. 2024 IEEE Int. Symp. Biomed. Imag.*, May 2024, pp. 1–5.
[12] A. C. Kak, M. Slaney, and G. Wang, "Principles of computerized tomographic imaging," *Med. Phys.*, vol. 29, no. 1, pp. 107–107, Jan. 2002.
[13] F. Natterer, *The Mathematics of Computerized Tomography.*, Philadelphia, PA, USA: SIAM, 2001.
[14] J. A. Fessler, "Penalized weighted least-squares image reconstruction for positron emission tomography," *IEEE Trans. Med. Imag.*, vol. 13, no. 2, pp. 290–300, Jun. 1994.

[15] C. L. Epstein, *Introduction to the Math. of Med. Imag.*, 2nd ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007. [Online] Available: https://epubs.siam.org/doi/abs/10.1137/9780898717792

[16] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Syst. J.*, vol. 4, no. 1, pp. 25–30, Dec. 1965.

[17] F. Jacobs, E. Sundermann, B. De Sutter, K. Christiaens, and I. Lemahieu, "A fast algorithm to calculate the exact radiological path through a pixel or voxel space," *J. Comput. Inf. Technol.*, vol. 6, no. 1, pp. 89–94, 1998.

[18] H. Zhao and A. Reader, "Fast ray-tracing technique to calculate line integral paths in voxel arrays," in *Proc. IEEE Nucl. Sci. Symp.*, Dec. 2003, pp. 2808–2812.

[19] G. T. Herman, *Fundamentals of Computerized Tomography: Image Reconstruction From Projections.*, Berlin, Germany: Springer, Jul. 2009.

[20] M. Unser and T. Blu, "Wavelet theory demystified," *IEEE Trans. Signal Process.*, vol. 51, no. 2, pp. 470–483, Feb. 2003.

[21] W. Jakob, S. Speierer, N. Roussel, and D. Vicini, "Dr.Jit: A just-in-time compiler for differentiable rendering," *ACM Trans. Graph.*, vol. 41, no. 4, Jul. 2022, Art. no. 124, doi: 10.1145/3528223.3530099.

[22] 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, doi: 10.1109/TMI.2002.1000260.

[23] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing. I. Theory," *IEEE Trans. Signal Process.*, vol. 41, no. 2, pp. 821–833, Feb. 1993.

[24] M. Pourya, A. Boquet-Pujadas, and M. Unser, "A box-spline framework for inverse problems with continuous-domain sparsity constraints," *IEEE Trans. Comput. Imag.*, vol. 10, pp. 790–805, 2024.

[25] P. Rattey and A. Lindgren, "Sampling the 2-D Radon transform," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. TASSP-29, no. 5, pp. 994–1002, Oct. 1981.

[26] R. A. Brooks, G. H. Weiss, and A. J. Talbert, "A new approach to interpolation in computed tomography," *J. Comput. Assist. Tomogr.*, vol. 2, no. 5, pp. 577–585, Nov. 1978, doi: 10.1097/00004728-197811000-00010.

[27] B. Albertina et al., "The cancer genome atlas lung adenocarcinoma collection (TCGA-LUAD)," 2016. [Online] Available: https://www.cancerimagingarchive.net/collection/tcga-luad/

[28] A. Boquet-Pujadas et al., "A silicon-pixel paradigm for PET," *IEEE Trans. Radiat. Plasma Med. Sci.*, vol. 9, no. 2, pp. 228–246, Feb. 2025.

[29] R. Ng, "Fourier slice photography," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 735–744, Jul. 2005, doi: 10.1145/1073204.1073256.