Available online at www.sciencedirect.com

ScienceDirect

Appl. Comput. Harmon. Anal. 21 (2006) 145–167

Applied and Computational Harmonic Analysis

www.elsevier.com/locate/acha

Fast and accurate Polar Fourier transform

A. Averbuch [a], R.R. Coifman [b], D.L. Donoho [c], M. Elad [d,*], M. Israeli [d]

[a] *Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel*
[b] *Department of Mathematics, Yale University, New Haven, CT 06520-8283, USA*
[c] *Department of Statistics, Stanford University, Stanford, CA 94305-9025, USA*
[d] *Department of Computer Science, The Technion, Haifa 32000, Israel*

Received 16 August 2005; revised 25 October 2005; accepted 17 November 2005

Available online 27 December 2005

Communicated by Charles K. Chui

Abstract

In a wide range of applied problems of 2D and 3D imaging a continuous formulation of the problem places great emphasis on obtaining and manipulating the Fourier transform in Polar coordinates. However, the translation of continuum ideas into practical work with data sampled on a Cartesian grid is problematic. In this article we develop a fast high accuracy Polar FFT. For a given two-dimensional signal of size $N \times N$, the proposed algorithm's complexity is $O(N^2 \log N)$, just like in a Cartesian 2D-FFT. A special feature of our approach is that it involves only 1D equispaced FFT's and 1D interpolations. A central tool in our method is the pseudo-Polar FFT, an FFT where the evaluation frequencies lie in an oversampled set of nonangularly equispaced points. We describe the concept of pseudo-Polar domain, including fast forward and inverse transforms. For those interested primarily in Polar FFT's, the pseudo-Polar FFT plays the role of a halfway point—a nearly-Polar system from which conversion to Polar coordinates uses processes relying purely on 1D FFT's and interpolation operations. We describe the conversion process, and give an error analysis of it. We compare accuracy results obtained by a Cartesian-based unequally-sampled FFT method to ours, both algorithms using a small-support interpolation and no pre-compensating, and show marked advantage to the use of the pseudo-Polar initial grid.

Keywords: Polar coordinates; Cartesian coordinates; Pseudo-Polar coordinates; Fast Fourier transform; Unequally-sampled FFT; Interpolation; Linogram

1. Introduction

1.1. Polar Fourier transform

Fourier analysis is a fundamental tool in mathematics and mathematical physics, and also in signal and image processing. The discovery, popularization, and digital realization of fast algorithms for Fourier analysis—so called FFT—has had far reaching implications in science and technology in recent decades. The scientific computing com-

* Corresponding author.
E-mail address: elad@cs.technion.ac.il (M. Elad).

munity regards the FFT as one of the leading algorithmic achievements of the 20th century [1]. In fact, even ordinary consumer-level applications now involve FFT's—think of web browser decoding JPEG images—so that development of new tools for Fourier analysis of digital data may be of potentially major significance. In this paper we develop tools associated with Fourier analysis in which the set of frequencies is equispaced when viewed in Polar coordinates.

Let $f(x) = f(x_1, x_2)$ be a function on the plane $x = (x_1, x_2) \in \mathbf{R}^2$. Let

$$\hat{f}(\phi) = \int f(x)\exp(-i\phi' x)\,\mathrm{d}x$$

be the usual continuum Fourier transform of f. Writing the frequency $\phi = \{r\cos(\theta), r\sin(\theta)\}$ in Polar coordinates, we let

$$\tilde{f}(r,\theta) = \hat{f}\bigl(\phi(r,\theta)\bigr).$$

In this paper, the term *Polar Fourier transform* will always refer to the operation

$$\tilde{f}(r,\theta) = \mathcal{PF}\bigl\{f(x)\bigr\},$$

namely, getting $f(x)$ in Cartesian variables and computing $\tilde{f}(r,\theta)$ defined with Polar variables.

While changes of variables are, of course, banal per se, their significance lies in the change of viewpoint they provide. As it turns out, several fundamental procedures for manipulation of such continuum functions f can best be defined in terms of Polar variables. The Polar FT can be a powerful tool for organizing our understanding of operators and functions on the two-dimensional continuum. Much the same can be said of higher dimensions.

1.2. Digital problematics

The relatively simple and obvious continuum concepts, posing little or no intellectual challenge, correspond to concrete processes operating on sampled digital data, which are important and widely used. Such is the case in image rotation, image registration, tomography, and more. It seems natural to ask if the conceptual simplicity and clarity of the continuum Polar FT can be used in some way to assist, improve, or simplify practical procedures for digital processing, which appear as digitally sampled realizations of rotation, registration, radon transformation, and so on.

This leads naturally to the question of whether there could be a Polar FT for discrete data, particularly a fast algorithm, or *Polar FFT*. Such a hypothetical entity should have many of the properties of the continuum Polar FT, including relations to rotation, registration, Radon transform, and so on, and yet would be definable and rapidly computed for digital data in the now ubiquitous equispaced Cartesian format.

The prevailing belief seems to be that there is no such algorithm. For example, in Briggs' treatise *The FFT: An Owner's Manual for the Discrete Fourier Transform* [2], which is widely considered comprehensive and authoritative, the index contains the entry "Polar FFT," continuing with "no FFT for, 284"! Indeed, several difficulties stand in our way to construct such a fast Polar FT for discrete data: (i) the need to define the appropriate Polar grid in the frequency domain; (ii) finding a fast way to evaluate (or approximate) the FT on this grid points; and (iii) enabling a stable transform inversion.

1.3. This paper's contribution

In this paper we propose a notion of a Polar FT which is well suited for digital data—a procedure which is faithful to the continuum Polar FT concept, highly accurate,[1] fast, and generally applicable. As in Fig. 1, we define the Polar grid of frequencies $\xi_{p,q} = \{\xi_x[p,q], \xi_y[p,q]\}$ in the circle inscribed in the fundamental region $\xi \in [-\pi,\pi)^2$, and, given digital Cartesian data $f[i_1,i_2]$ we define the Polar FT to be the collection of samples $\{F(\xi_{p,q})\}$, where $F(\xi_{p,q})$ is the trigonometric polynomial

$$F(\xi_{p,q}) = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1,i_2]\exp\bigl(-i\bigl(i_1\xi_x[p,q] - i_2\xi_y[p,q]\bigr)\bigr). \tag{1}$$

[1] This implies that the proposed algorithm gives an approximated evaluation of the transform with high accuracy.

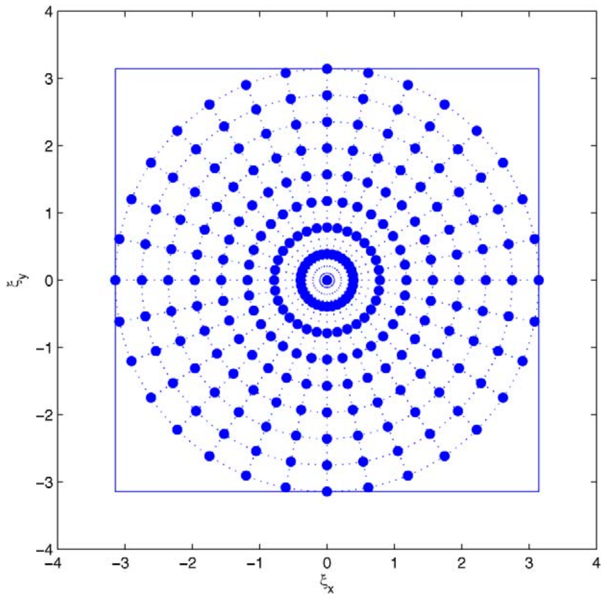

Fig. 1. Polar grid—intersection of 8 concentric circles and 16 angularly equispaced rays.

The sampled frequency points are given by

$$\left\{ \begin{array}{ll} \xi_x[p,q] = \frac{\pi p}{N}\cos(\pi q/2N) & \\ & \text{for } -N \leqslant p \leqslant N-1,\ 0 \leqslant q \leqslant 2N-1 \\ \xi_y[p,q] = \frac{\pi p}{N}\sin(\pi q/2N) & \end{array} \right\}. \tag{2}$$

Thinking of the Polar Discrete Fourier Transform (PDFT) mapping $\mathcal{PDFT}: f[i_1,i_2] \to F(\xi_{p,q})$ as a linear operator, we also consider a generalized inverse procedure of it, going back from discrete Polar Fourier data to Cartesian spatial data.

In this paper we carefully define the Polar FT concept for digital data, the associated fast algorithms, and discuss its features such as accuracy and computational complexity. We will make special effort to describe five advantages of the proposed PFFT, *speed, accuracy, stability, vectorizability*, and *nonexpansivity*. These terms will be given meaning as we deepen our description. A brief and partial version of this work appeared in [3], and here we give an expanded and more detailed description.

1.4. Relation to state of the art

Two existing bodies of literature contain ideas relevant to the above definition of Polar FT—one corresponds to methods to compute the inverse Radon transform, and the other treating the general problem of evaluating the Fourier transform on nonequally spaced frequency points.

The literature of tomography concerns reconstruction of an image from a collection of its one-dimensional integrals. An important approach to this problem is the direct Fourier method [4–16]. The direct Fourier method uses the projection-slice theorem, which suggests that one can convert a collection of projection data $(Rf)(\cdot,\theta)$ into a two-dimensional Fourier data $\hat{f}(\xi)$ on a Polar grid, and then reconstruct by Fourier inversion.

In the second body of literature [17–29] one has data at an equally spaced Cartesian grid, but wishes to evaluate its discrete Fourier transform as a nonequispaced non-Cartesian set of frequencies. This problem is known as either the USFFT (short for Unequally Spaced FFT), or the NUFFT (non-Uniform FFT) problem.

It is not hard to see the direct relevance of the above literature to the definition we have given. The Polar sampling set defined above is certainly a nonequispaced, non-Cartesian set. Thus, the problem of computing $F(\xi_{p,q})$ from digital data $f[i_1,i_2]$ is explicitly a problem of evaluating the Fourier transform of f at unequally spaced frequencies. In other words, the problem of computing the PDFT is a special case of the USFFT problem. Second, note that, if we take the projection data $(Rf)(\cdot,\theta)$ and perform a discrete Fourier transform in the (discretely sampled) t-variable, we

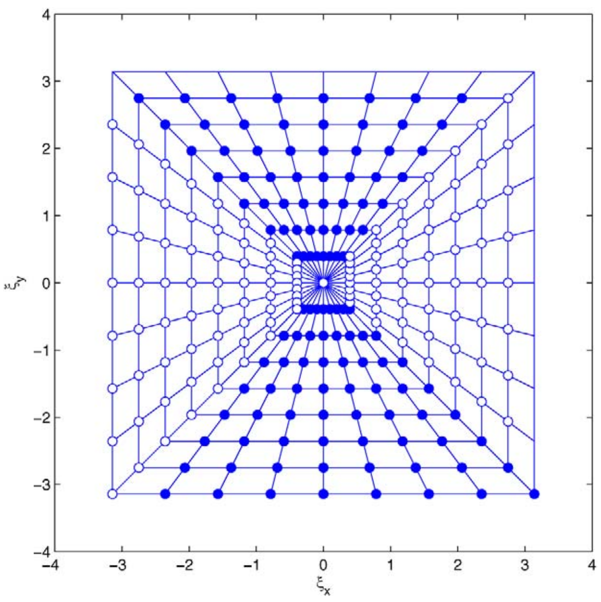

Fig. 2. The pseudo-Polar grid and its separation into BV (filled circles) and BH (empty circles) coordinates ($N = 8$)—intersection of 8 concentric squares and 16 slope-equispaced rays.

effectively have $\hat{f}(\xi_{p,q})$. Hence, the problem of Radon inversion is closely connected to the problem of going *back* from discrete Polar Fourier data $\hat{f}(\xi_{p,q})$ to Cartesian spatial data $f[i_1, i_2]$.

The existing state-of-the-art expressed by the above literature can be adapted to produce an approximated Polar FFT (PFFT) and its inverse. Adapting ideas from the USFFT literature, the forward Polar-FFT can be accomplished by the following stages: (i) pre-multiplication of $f[i_1, i_2]$ by pre-computed weights $s[i_1, i_2]$; (ii) computation of the 2D-FFT on an oversampled Cartesian grid; and (iii) interpolation of the desired values on the Polar grid based on those given on the Cartesian one. Reversing these operations can provide the adjoint operator. The inversion of the transform can be accomplished, for example, by least-squares fitting, as proposed in [28]. This approach is similar to ideas available in the direct Fourier reconstruction literature mentioned above.

The approach we propose here for PFFT is in fact similar to the above, using a special type of USFFT method, where a different launching grid is used, rather than the Cartesian one. Our method factors the problem into two steps: first, a *pseudo-Polar* FFT is applied, in which a pseudo-Polar sampling set is used, and second, a conversion from pseudo-Polar to Polar FT is performed.

At the heart of the method proposed here for the PFFT we use the pseudo-Polar FFT—an FFT where the evaluation frequencies lie in an oversampled set of nonangularly equispaced points (see Fig. 2). The pseudo-Polar FFT offers us a near-Polar frequency coordinate system for which exact rapid FT evaluation is possible [30]. Whereas the Polar grid points sit at the intersection between linearly growing concentric circles and angularly equispaced rays, the pseudo-Polar points sit at the intersection between linearly growing concentric squares and a specific choice of angularly nonequispaced rays.

The pseudo-Polar transform plays the role of a halfway point—a nearly-Polar system from which conversion to Polar coordinates uses processes relying purely on 1D FFT's and interpolation operations. We present this conversion process, along with an error analysis of it, showing far improved performance compared to the Cartesian-based USFFT-based alternative approach. The reason for the expected improvement in accuracy is the ability of the pseudo-Polar grid to provide a space-varying sampling of the frequency domain, which is close in density to the Polar destination one. More specifically, the pseudo-Polar grid gives a denser sampling near the origin, enabling better interpolation performance. In comparison, starting with a Cartesian grid, getting similar density (and thus accuracy) near the origin requires an increased oversampling all-over the frequency domain. Note also that for most signals, their energy resides near the origin, and thus, better treatment there implies higher overall accuracy.

As mentioned before, the above described approach enjoys the following important advantages:

- *Complexity*. For a given two-dimensional signal of size $N \times N$, the complexity of both the proposed Polar FFT and its inverse is of order $N^2 \log N$, just like in a Cartesian 2D-FFT operating on the same input array.
- *Accuracy*. Since polynomial interpolations are involved in the algorithm's path, exact results cannot be claimed. However, as will be shown later, the applied interpolations are expected to yield highly accurate results due to specific features of the frequency domain function slices. The accuracy obtained is substantially better than the one expected with Cartesian-based USFFT methods. We should note that this claim is demonstrated in our work for short-support interpolation schemes that provide an overall simple algorithm. Recent work by Fessler and Sutton [28] have clearly shown a marked improvement in the obtained accuracy with Cartesian USFFT when using large support Kaiser–Bessel interpolation and pre-compensation. Further work is required to use those options with initial pseudo-Polar grid and explore ways to further improve our method.
- *Stability*. This property refers mostly to our ability to invert the transform, and claim almost one-to-one mapping between spatial array and its inverse PFFT result. Two ingredients are responsible for such a behavior—adequate conditioning of the signal, prior to its transform, and a proper set of Polar frequency samples that would be considered representative.
- *Vectorizability*. The process we propose is organized as a series of purely one-dimensional signal manipulation processes, in which data form a single row or column of an array are subjected to a series of Fourier transforms, 1D interpolations, and ancillary operations. Owing to the use of vectorized operations, the overall algorithm is "cache-friendly."[2]
- *Nonexpansivity*. There is no drastic oversampling of the underlying array. As we show later on, a similar accuracy by a Cartesian-based USFFT method requires much higher oversampling.

1.5. Content

This paper is organized as follows: Section 2 describes the pseudo-Polar Fourier grid, and the forward/inverse transforms over this grid. In this section we closely follow the work in [30]. Section 3 then discusses the conversion from pseudo-Polar-to-Polar for the forward transform, and from Polar-to-pseudo-Polar in the inverse one. We show how these conversions amount to one-dimensional operations and discuss reasons for the high interpolation accuracy obtained. Section 4 analyzes the proposed algorithm from several aspects. Accuracy is studied by experimenting on images, and by finding the worst-case scenarios maximizing the approximation error. Section 5 gives a brief description of a freely available software performing the proposed PFFT transform. This software also includes the code to reproduce this paper's results. Section 6 concludes this paper, with discussion on future work and open questions.

2. Pseudo-Polar Fourier transform

The pseudo-Polar Fourier transform is based on a definition of a Polar-like 2D grid that enables fast Fourier computation. This grid has been explored by many since the 1970s. The pioneers in this field are Mersereau and Oppenheim [31] who proposed the concentric squares grid as an alternative to the Polar grid. Work by Pasciak [32], Edholm and Herman [33], and Lawton [34] showed that fast exact evaluation on such grids is possible. Later work by Munson et al. [8,10] have shown how these ideas can be extended and used for tomography. Recently, the pseudo-Polar grid was proposed as the base for a stable forward and inverse Radon transform called *Fast Slant-Stack* [30,35]. It has also been used for image registration [36].

As we shall see next, in this work we strongly build on this notion of near-Polar grid, and exploit the fast and stable forward and inverse transforms with the frequency domain sampled with such coordinate system. In this section we cover the basics of the pseudo-Polar grid and its use for forward and inverse transform.

[2] We should note that Cartesian-based USFFT methods may also be made cache-efficient in various ways [25]. However, this property is far more natural when the data is structured as in the pseudo-Polar coordinates.

Before we start we remind the reader of the Cartesian grid 2D DFT to set notations. Letting the horizontal and vertical frequencies

$$\left\{\xi_x = \frac{2\pi k_1}{N}, \xi_y = \frac{2\pi k_2}{N}\right\}_{k_1,k_2=0}^{N-1},$$

we get the familiar DFT in 2 dimensions by

$$\hat{f}(\xi_x, \xi_y) = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\bigl(-i(i_1\xi_x + i_2\xi_y)\bigr). \tag{3}$$

2.1. The pseudo-Polar coordinate system

We start by defining the pseudo-Polar grid points in the frequency domain. These points are separated into two groups—the basically vertical (BV) and the basically horizontal (BH) subsets, given by

$$BV = \left\{\xi_y = \frac{\pi\ell}{N} \quad \text{for } -N \leqslant \ell < N, \qquad \xi_x = \xi_y \cdot \frac{2m}{N} \quad \text{for } -\frac{N}{2} \leqslant m < \frac{N}{2}\right\} \tag{4}$$

and

$$BH = \left\{\xi_x = \frac{\pi\ell}{N} \quad \text{for } -N \leqslant \ell < N, \qquad \xi_y = \xi_x \cdot \frac{2m}{N} \quad \text{for } -\frac{N}{2} < m \leqslant \frac{N}{2}\right\}. \tag{5}$$

Figure 2 depicts this grid; BV points are marked with the filled disks and BH ones are marked as circles. Several properties about this grid should be mentioned:

- The grid points are at intersections of linearly growing concentric squares with angularly nonequispaced rays. The squares' sides are of size $\pi k/N$, $k = 0, 1, \ldots, N$. The BH rays have equispaced slope: $2k/N$, $k = -N/2 + 1, -N/2 + 2, \ldots, N/2$. The BV rays are similar but with clockwise rotation of 90°.
- The suggested grid is Polar-like. The main two differences are the concentric squares replacing concentric circles, and equispaced rays in slope replaced by equispaced rays in angle. This resemblance will be exploited for the development of a true Polar-FFT algorithm.
- Referring to the BV (and similarly to the BH) points, we can see that they are organized on lines with angles ranging from $-\pi$ to π. Along each such line, the points are spread uniformly, though in an angle-dependent way. These two properties are the driving force for the ability to compute the Fourier transform in a fast way for this grid.
- When looking at Fig. 2 one might get the impression that some points on the outer-most square are missing. However, since $\hat{f}(\xi_x, \xi_y)$ is periodic with a period of 2π in both axes, these points are actually redundant.
- In computing and storing the Fourier transform for these points, our data structure is given by two simple 2D arrays—one for the BV and the other for the BH sample sets. Referring to the BV array, the vertical axis corresponds to the index ℓ and the horizontal to m (see (4) and (5)). Thus, our array has $2N$ rows and N columns. Overall we have $4N^2$ frequency sample points, originating from an image of size $N \times N$. The factor 4 oversampling is helpful for numerical stability [30]. This data structure implies that when we draw a row or column from these arrays, we do not necessarily refer to horizontal or vertical set of frequency points, but rather refer to points along one ray or points along one of the concentric squares.
- In our treatment, the processing of the BV and the BH data are completely parallel. Our description will refer to the BV points only.

2.2. A fast forward transform

Given the pseudo-Polar BV grid points in the frequency domain, we are interested in computing the Fourier transform values. In what follows we shall show that simple 1D-FFT operations can be used to achieve this goal.

Theorem 1. *Given a 2D signal $f[i_1, i_2]$, $0 \leqslant i_1, i_2 < N$, evaluation of the FT on the pseudo-Polar grid as defined in* (4) *and* (5) *can be done by 1D operations only, and with complexity of $140N^2 \log N$ flops.*

Proof. This result can be considered known, applying ideas behind the work of Pasciak [32], Edholm and Herman [33], and Lawton [34] to the pseudo-Polar grid, which, however, is not precisely the grid they considered. Essentially, the result as stated has been obtained in [30], and is spelled-out here for completeness.

Using the definition of the Fourier transform for discrete functions in Eq. (3), and plugging the coordinates from Eqs. (4) and (5) we obtain

$$\begin{aligned}\hat{f}(\xi_x, \xi_y) = \hat{f}[m, \ell] &= \sum_{i_1=0}^{N-1} \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\bigl(-i(i_1\xi_x + i_2\xi_y)\bigr) \\ &= \sum_{i_1=0}^{N-1} \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\left(-i\left(\frac{2\pi i_1 \ell m}{N^2} + \frac{\pi i_2 \ell}{N}\right)\right) \\ &= \sum_{i_1=0}^{N-1} \exp\left(-\frac{i2\pi i_1 \ell m}{N^2}\right) \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\left(-\frac{i\pi i_2 \ell}{N}\right). \end{aligned} \tag{6}$$

Concentrating on the inner summation part, we define

$$\hat{f}_1[i_1, \ell] = \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\left(-\frac{i\pi i_2 \ell}{N}\right). \tag{7}$$

This expression stands for a 1D-FFT on the columns of the zero padded array $f[i_1, i_2]$. In order to show this, assume that $f[i_1, i_2]$ is zero padded to yield

$$f_Z[i_1, i_2] = \begin{cases} f[i_1, i_2], & 0 \leqslant i_2 < N, \\ 0, & N \leqslant i_2 < 2N. \end{cases}$$

Thus,

$$\hat{f}_1[i_1, \ell] = \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\left(-\frac{i\pi i_2 \ell}{N}\right) = \sum_{i_2=0}^{2N-1} f_Z[i_1, i_2] \cdot \exp\left(-\frac{i2\pi i_2 \ell}{2N}\right). \tag{8}$$

This expression stands for 1D-FFT of an array of $2N$ samples. However, another complicating factor is the range of ℓ ($-N \leqslant \ell \leqslant N-1$), as opposed to the regular FFT range that starts at 0. Thus, defining $n = \ell + N$ we get

$$\hat{f}_1[i_1, n] = \sum_{i_2=0}^{2N-1} f_Z[i_1, i_2] \cdot \exp\left(-\frac{i2\pi i_2(n-N)}{2N}\right) = \sum_{i_2=0}^{2N-1} f_Z[i_1, i_2] \cdot (-1)^{i_2} \cdot \exp\left(-\frac{i2\pi i_2 n}{2N}\right). \tag{9}$$

To summarize this part, computation of the inner summation (w.r.t. i_2) amounts to 1D-FFT of length $2N$ on the columns of the zero padded and pre-multiplied[3] array $f[i_1, i_2] \cdot (-1)^{i_2}$. The amount of operations needed for this stage are $5 \cdot 2N \log(2N)$ per each column,[4] and $10N^2 \log(2N)$ operations for all the N columns.

We return now to Eq. (6) and assume that the above process has been completed. Thus we hold the $N \times 2N$ array $\hat{f}_1[i_1, \ell]$ and we should proceed by computing the second summation

$$\hat{f}[m, \ell] = \sum_{i_1=0}^{N-1} \hat{f}_1[i_1, \ell] \exp\left(-\frac{i2\pi i_1 m \ell}{N^2}\right) = \sum_{i_1=0}^{N-1} \hat{f}_1[i_1, \ell] \exp\left(-\frac{i2\pi i_1 m}{N} \cdot \frac{\ell}{N}\right). \tag{10}$$

[3] We have to multiply our array by $(-1)^{i_2}$, but this could be replaced by later shift, owing to the periodic nature of the transform.

[4] We assume hereafter that a regular DFT on a vector of length N requires $5 \cdot N \log(N)$ operations by the FFT algorithm [2]. Although this applies originally to N being powers of 2, we use it for an arbitrary N, with the understanding that the coefficient may slightly change.

Removal of the factor $\alpha = \ell/N$ in the exponent turns this expression into a regular 1D-FFT, this time applied on the rows of the array $\hat{f}_1$. With this factor α in the above summation, the required operation is known as the Chirp-Z [37], or the Fractional Fourier Transform (FRFT) [38]. In Appendix A we show how this transform can be computed efficiently for any α, with $30N\log N$ operations, based on 1D-FFT use. As before, due to a shifted range of the index m, one has either to shift after the transform, or modulate the array, prior to the transform.

To summarize, we have $2N$ rows that are to be put through a Chirp-Z transform. Thus we need $60N^2\log N$ operations for this stage. Adding to the previous stage complexity, we get an overall of $70N^2\log N$ operations for the computation of the transform for the BV part of the grid. To cover both the BV and the BH grid points we need $140N^2\log N$. □

In this paper the pseudo-Polar grid is a stepping stone toward the Polar coordinates system. Since interpolations are to be used to convert from pseudo-Polar to Polar coordinates, we should consider computing the pseudo-Polar FFT on a more densely-spaced grid. We state here without proof the following result: given a 2D signal $f[i_1, i_2]$, $0 \leqslant i_1, i_2 < N$, the evaluation of the FT on the oversampled pseudo-Polar grid with NS concentric squares and $2NP$ slope-equispaced rays can be obtained by 1D vector operations only, and with complexity of $120N^2PS\log(NS)$ flops. In fact, this result could be shown to follow immediately from Theorem 1 with zero-padding of the original image.

2.3. *Fast inverse transform and quasi-Parseval relationship*

The pseudo-Polar FFT can be inverted by the method of least-squares (LS) (see [28,30] for the same concept). To see this, consider a matrix–vector formulation of our processes. For an image $f[i_1, i_2]$ of size $N \times N$, we define the column vector $\underline{f}$ of length N^2, containing the image pixel values in column-stack ordering. This vector is multiplied by a matrix $\mathbf{T}_{\mathrm{PP}} \in \mathcal{C}^{4N^2 \times N^2}$, representing the pseudo-Polar FFT. The outcome of this multiplication is a vector $\underline{\hat{f}}$ of length $4N^2$, containing the samples of the Fourier transform on the pseudo-Polar grid. Given $\underline{\hat{f}}$, inversion of the pseudo-Polar Fourier transform is achieved by solving

$$\underline{f} = \operatorname{Arg}\min_{\underline{x}} \|\mathbf{T}_{\mathrm{PP}}\underline{x} - \underline{\hat{f}}\|_2^2 = \left(\mathbf{T}_{\mathrm{PP}}^H\mathbf{T}_{\mathrm{PP}}\right)^{-1}\mathbf{T}_{\mathrm{PP}}^H\underline{\hat{f}} = \mathbf{T}_{\mathrm{PP}}^{+}\underline{\hat{f}}, \tag{11}$$

where $\mathbf{T}_{\mathrm{PP}}^{+}$ denotes generalized inverse.

Of course, the matrix-based solution is useless as a computational approach for reasonable sizes of input arrays; a pseudo-direct inversion of the matrix $\mathbf{T}_{\mathrm{PP}}$ is computationally prohibitive, and definitely beyond the desired $N^2\log(N)$ complexity. Instead, we approach the optimization problem iteratively by the iteration relation

$$\underline{f}_{k+1} = \underline{f}_k - \mathbf{D}\mathbf{T}_{\mathrm{PP}}^H(\mathbf{T}_{\mathrm{PP}}\underline{f}_k - \underline{\hat{f}}). \tag{12}$$

The expression $\mathbf{T}_{\mathrm{PP}}^H(\mathbf{T}_{\mathrm{PP}}\underline{x} - \underline{\hat{f}})$ is the function's gradient and the multiplication by a positive definite matrix $\mathbf{D}$ guarantees descent in the LS error. If chosen properly, $\mathbf{D}$ could speed-up convergence of this algorithm to the true solution, as posed in (11). In [30] a specific choice of diagonal matrix $\mathbf{D}$ is proposed that down-weights near-origin points in order to equalize the condition number. It is shown that with few (2–6) iterations this iterative process achieves high accuracy solutions. As to the initialization, it could be chosen as zeros ($\underline{f}_0 = \underline{0}$) for simplicity. This way we obtain a fast inverse transform of the same complexity as the forward one, as every iteration requires the application of both the forward transform (multiplying with $\mathbf{T}_{\mathrm{PP}}$) and its adjoint (multiplying with $\mathbf{T}_{\mathrm{PP}}^H$). It still remains to be seen that the adjoint is computable with the same complexity as the forward transform, which is the claim of the next theorem.

Theorem 2. *Given a 2D array $\hat{f}[m, \ell]$ of size $2N \times 2N$ representing a pseudo-Polar grid sampling in the frequency domain, the evaluation of the adjoint pseudo-Polar FFT to produce an $N \times N$ image can be done by 1D operations only, and with complexity of $O\{N^2\log(N)\}$ operations.*

Proof. Posed in a matrix notation, multiplication by $\mathbf{T}_{\mathrm{PP}}^H$ requires taking the conjugate of each element in the matrix $\mathbf{T}_{\mathrm{PP}}$, and summing with respect to columns, rather than rows. Thus, using the relation posed in (6) we have that the regular (forward) transform is applied by

$$\hat{f}[m,\ell] = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1,i_2]\cdot\exp\left(-i\left(\frac{2\pi i_1\ell m}{N^2}+\frac{\pi i_2\ell}{N}\right)\right).$$

Similarly, referring to the basically vertical (BV) values only, the adjoint is achieved by

$$\begin{aligned}\tilde{f}[i_1,i_2] &= \sum_{m=-N/2}^{N/2-1}\sum_{\ell=-N}^{N-1}\hat{f}[m,\ell]\cdot\exp\left(i\left(\frac{2\pi i_1\ell m}{N^2}+\frac{\pi i_2\ell}{N}\right)\right)\\ &= \sum_{\ell=-N}^{N-1}\exp\left(i\frac{\pi i_2\ell}{N}\right)\sum_{m=-N/2}^{N/2-1}\hat{f}[m,\ell]\cdot\exp\left(i\frac{2\pi i_1\ell m}{N^2}\right).\end{aligned} \tag{13}$$

The inner summation could be written as

$$\tilde{f}_1[i_1,\ell] = \sum_{m=-N/2}^{N/2-1}\hat{f}[m,\ell]\cdot\exp\left(i\frac{2\pi i_1 m}{N}\cdot\frac{\ell}{N}\right)$$

and this is a Fractional FFT, just as we obtained with the forward transform. We have seen that this operation could be done in $O\{N\log(N)\}$ per every ℓ, producing N values. Thus, for $-N \leqslant \ell \leqslant N-1$ we need to perform $O\{N^2\log(N)\}$ operations. Once performed, we then obtain the expression

$$\tilde{f}[i_1,i_2] = \sum_{\ell=-N}^{N-1}\exp\left(i\frac{\pi i_2\ell}{N}\right)\tilde{f}_1[i_1,\ell] \tag{14}$$

and this is a regular inverse-FFT, which requires $O\{N\log(N)\}$ per every $0 \leqslant i_1 \leqslant N-1$. Thus, again we get that $O\{N^2\log(N)\}$ operations are required to conclude this part of the computations.

As a last point in this proof, we should refer similarly to the BH rays. The complete adjoint operation is obtained by performing the same process as described above, and adding the two resulting arrays. □

3. From pseudo-Polar to Polar

Similar to the Cartesian-based USFFT approach, we suggest to compute the Polar-FT values based on a different grid for which a fast algorithm exists, and then go to the Polar coordinates via an interpolation stage. However, instead of using the Cartesian grid in the first stage, we use the pseudo-Polar grid of the previous section. Since this grid is closer to the Polar destination coordinates, there is a reason to believe that this approach will lead to better accuracy and thus lower oversampling requirements. However, as we shall see next, beyond the proximity of the pseudo-Polar coordinates to the Polar ones, another very important benefit is the ability to perform the necessary interpolations via pure 1D operations without losing accuracy. This property is vital in understanding the superiority of the proposed scheme over traditional Cartesian-based USFFT methods.

3.1. Pseudo-Polar–Polar: grid conversion

We define the Polar coordinate system based on the pseudo-Polar one, with manipulations that lay out the necessary interpolation stages discussed later on. Starting from the basically-vertical frequency sampling points in the pseudo-Polar grid as given in (4)

$$BV = \left\{\xi_y = \frac{\pi\ell}{N} \quad \text{for } -N \leqslant \ell < N, \qquad \xi_x = \frac{2\pi m\ell}{N^2} \quad \text{for } -\frac{N}{2} \leqslant m < \frac{N}{2}\right\},$$

the Polar ones are obtained by two operations:

(1) *Rotate the rays*. In order to obtain an angularly-uniform ray sampling as in the Polar coordinate system, the rays must be rotated. This is done by replacing the term $2m/N$ in ξ_x above with $\tan(\pi m/2N)$, leading to the grid points

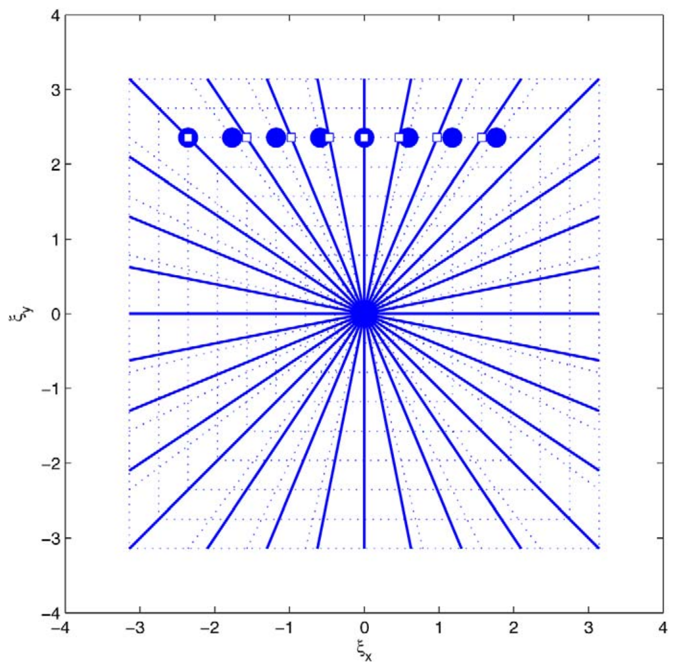

Fig. 3. The first interpolation stage for rotating the rays. Circles represent the known pseudo-Polar grid points and squares are the desired equiangularly spaced rays.

$$BV_{\text{new}} = \left\{ \xi_y = \frac{\pi \ell}{N} \quad \text{for } -N \leqslant \ell < N, \qquad \xi_x = \frac{\pi \ell}{N} \cdot \tan\left(\frac{\pi m}{2N}\right) \quad \text{for } -\frac{N}{2} \leqslant m < \frac{N}{2} \right\}.$$

The result is a set of points organized on concentric squares as before, but the rays are now equispaced in angle rather than slope. Figure 3 depicts this step as an interpolation stage. Rotating the rays amounts to a 1D operation along horizontal lines (for the BV points). A set of N equispaced points along this line are replaced by a new set of N points along the same line in different locations (marked as small squares) implementing equispaced angular sampling.

An interesting property of this interpolation stage is the fact that the underlying function to be interpolated assumes a simple form, which implies that interpolation accuracy is expected to be high, even for low oversampling factors. Furthermore, the given points are densely arranged around the desired locations, thus ensuring a better interpolation accuracy (compared to the same number of points spread from end-to-end, as in the Cartesian USFFT).

Returning to Eq. (6) and referring to a specific row with fixed ξ_y, we have

$$\begin{aligned} \hat{f}(m, \xi_y) &= \sum_{i_1=0}^{N-1} \sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\left(-i\left[\frac{2mi_1}{N} + i_2\right]\xi_y\right) \\ &= \sum_{i_1=0}^{N-1} \left[\sum_{i_2=0}^{N-1} \exp(-i\xi_y i_2) f[i_1, i_2]\right] \cdot \exp\left(-i\frac{2mi_1}{N}\xi_y\right) = \sum_{i_1=0}^{N-1} C[i_1] \cdot \left[\exp\left(-i\frac{2i_1}{N}\xi_y\right)\right]^m \end{aligned} \tag{15}$$

and this is a complex trigonometric polynomial of order N.

One implication of this observation is that given N samples of this function, all information about this function is given. Theoretically, this implies that a perfect interpolation is possible if for every destination point all the N given samples are used. From a more practical point of view, the above observation implies that the function defined in Eq. (15) is relatively smooth, and with a moderate oversampling, a near-perfect interpolation is expected for a small neighborhood operation.

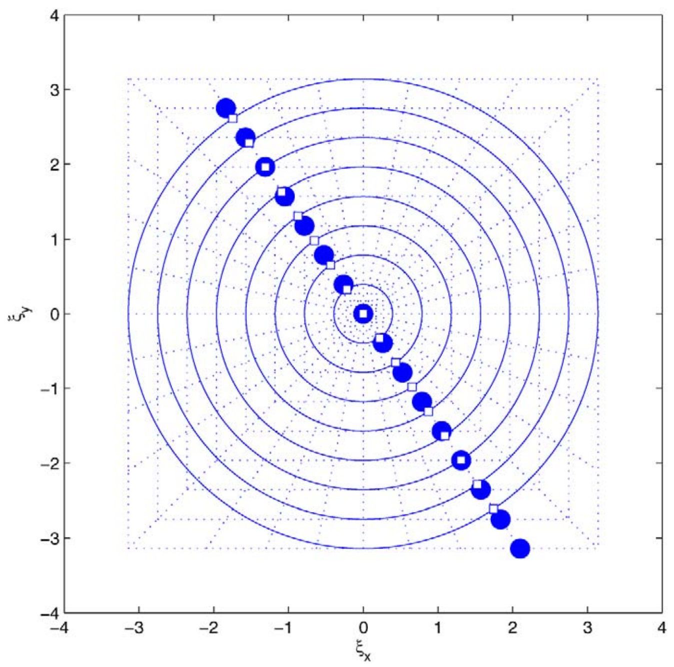

Fig. 4. Second interpolation stage for circling the squares. Small circular points represent the known grid points after the previous interpolation stage, and the square points are the desired final interpolated values.

(2) *Circle the squares*. In order to obtain concentric circles as required in the Polar coordinate system, we need to 'circle the squares.' This is done by dividing both ξ_x and ξ_y by a constant along each ray, based on its angle, and therefore a function of the parameter m, being

$$R[m] = \sqrt{1 + \tan^2\left(\frac{\pi m}{2N}\right)}. \tag{16}$$

The resulting grid is given by

$$\left\{ \begin{array}{c} \xi_y = \frac{\pi \ell}{N R[m]} \quad \text{for } -N \leqslant \ell < N \\ \xi_x = \frac{\pi \ell}{N R[m]} \cdot \tan\left(\frac{\pi m}{2N}\right) \quad \text{for } -\frac{N}{2} \leqslant m < \frac{N}{2} \end{array} \right\}.$$

Figure 4 depicts this step as an interpolation stage. Circling the squares amounts to a 1D operation along rays. A set of $2N$ equispaced points is replaced by a new set of $2N$ points along the same line in different locations (marked as small squares). This time the destination points are equispaced, but with a different spacing.
We have seen above that the first interpolation stage is applied on a trigonometric polynomial, which explains the expected accuracy obtained. In this later stage, the function over which we interpolate is not of the same simple form, but nevertheless smooth enough. Referring to one ray of slope α (in the range $[-1, 1]$, considering the *BH* rays) we have that the 1D slice to work on is given by

$$\hat{f}(\xi_x, \xi_y) = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\bigl(-i(i_1\xi_x + i_2\xi_y)\bigr) = \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\bigl(-i(i_1 + \alpha i_2)\xi_x\bigr) = \tilde{f}(\xi_x).$$

Consider the function $\tilde{f}(z)$ described above as a 1D function and take its Fourier transform

$$\mathcal{F}\bigl\{\tilde{f}(z)\bigr\} = \int_{-\infty}^{\infty} \tilde{f}(z)\exp(-i\omega z)\,\mathrm{d}z = \int_{-\infty}^{\infty} \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1, i_2] \cdot \exp\bigl(-i(i_1 + \alpha i_2)z\bigr)\exp(-i\omega z)\,\mathrm{d}z$$

$$= \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} f[i_1,i_2]\cdot\delta(i_1+\alpha i_2+\omega).$$

Here $\delta(i_1+\alpha i_2-\omega)$ is the Dirac function, being nonzero for $i_1+\alpha i_2-\omega=0$. This expression assumes its maximal frequency at $i_1=i_2=N-1$ and its minimal frequency at $i_1=i_2=0$. The frequency support of this function is the interval $[-(N-1)(1+\alpha),0]$. Thus, for $\alpha=1$ we obtain the widest bandwidth of $2N$, and then the Nyquist-sampling rate for this function is $T_{\max}=\pi/N$.

This means that the Fourier transform $\hat{f}$ restricted to the ray is a band-limited function with required maximal sampling period of π/N. If this function is sampled in this rate along the entire ray (from $-\infty$ to ∞) we have a complete representation of it that enables knowledge of its values at any location. In our case, for a limited interval $z\in[-\pi,\pi]$ we have $2N$ samples, which is the critical sampling rate exactly. Thus, with a moderate oversampling we may expect to represent this function very accurately. This implies that this function is relatively smooth as well and lends itself to high-accuracy interpolation.

3.2. The proposed scheme—overview

In performing the proposed Polar-FFT, we start with computation of the Fourier transform over a pseudo-Polar grid, followed by the two interpolation stages presented above. As we have seen, in order to obtain high accuracy, the pseudo-Polar grid should be oversampled, both radially and angularly. One approach to oversampling is the use of the over-sampled pseudo-Polar grid as presented in the previous section (e.g., by zero padding). Alternatively, as the pseudo-Polar fast transform can be broken into two parts, this separation could be used for better efficiency. For the BV coordinates, the vertical FFT can be done with oversampling by zero padding as before, and then, as we go to the second phase of Fractional FFT per each row, we can apply this stage one row at a time with the interpolation that leads directly to the rotated rays. The memory savings are substantial (directly proportional to the oversampling factor P). Note that in performing the 1D interpolations, complete rows or columns from the result array are brought to the cache, and processed to produce the desired values. Thus, memory management becomes extremely effective, as opposed to the regular USFFT methods that require general memory access that may lead to many cache misses.

As for the complexity of the overall algorithm, we have seen before that an amount of $120N^2SP\log_2(NS)$ operations are required for the computation of the oversampled pseudo-Polar FFT. Those operations are followed by the first interpolation requiring $O(N^2S)$ operations (every row uses NP points to compute N values requires $O(N)$ operations, and there are NS of those). The second interpolation requires $O(N^2)$ operations (every ray with $2NS$ values is used to produce $2N$ new values, and there are N rays). Thus, the overall operation count is dominated by the $120N^2SP\log_2(NS)$ operations of the pseudo-Polar FFT. This is $12SP$ times more complex than plain $N\times N$ Cartesian grid 2D-FFT.

In terms of memory requirements, the overcomplete pseudo-Polar FFT requires $4N^2SP$ float-values, and once those are allocated, all other operations can be done within this array. Alternatively, using row-wise interpolation, only $4N^2S$ values are required. Also, a factor 2 saving can be obtained if a proper separation between the basically vertical and basically horizontal parts is applied. Further savings could be obtained by breaking each of these groups (BV and BH) to sub-parts with small overlap.

All the above applies to a general image of size $N\times N$, resulting with an output Polar FT array of size $2N\times 2N$. If higher oversampling is desired in the destination grid, all the above is still applicable. One simple way to get such oversampling is to zero-pad the input array (this way modifying the initial N).

The inversion of the Polar-FFT requires an implementation of the inverse pseudo-Polar FFT as described in the previous section. Starting with a set of Polar coordinates in the frequency domain, we first apply two interpolation stages—*squaring the circles*, and then *rotating the rays*—just as described above but in reversed order. For such operation to succeed, we assume that the Polar grid is given in an over-complete manner (i.e., we start with an array of $2NP$[rays] $\times$ $2NS$[circles] and interpolate to a pseudo-Polar grid with $2N$[rays] $\times$ $2N$[squares]). Once those coordinates are filled with values, inverse pseudo-Polar FFT is applied as described previously, using the adjoint operator. The same arguments posed above explain the suitability of the 1D interpolation and their expected accuracy in the above reverse process.

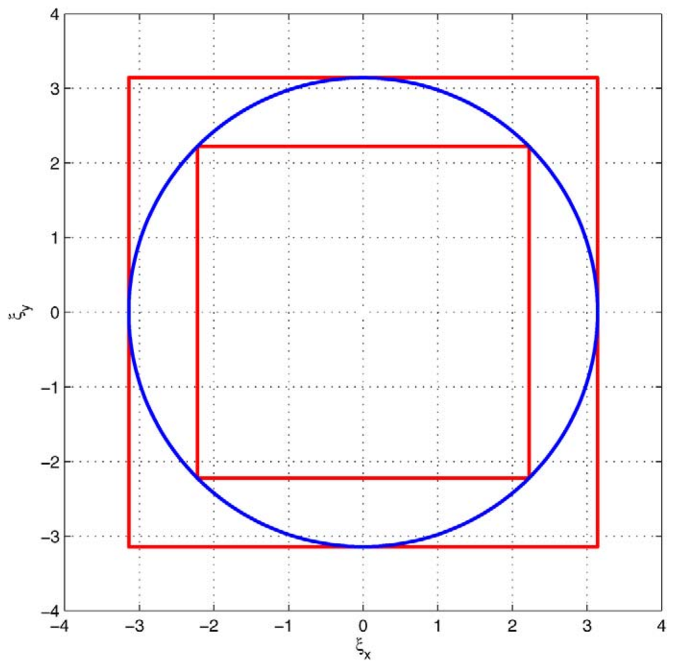

Fig. 5. The frequency support options: the outer square stands for the original frequency support of an arbitrary digital image. The inscribed disk stands for the required support for the Polar-FFT to behave well. The smaller square inside is the one archived by the simple re-sampling proposed, where the number of pixels in the image is increased by factor 2.

3.3. Disk band-limited support as regularization

A major difficulty in using the Polar coordinate system with digitally sampled data arises from the square shape of the fundamental frequency domain $[-\pi, +\pi]^2$. In our Polar FFT the corners are not represented and this may lead to some loss of information. From an algebraic point of view, one may say that the matrix representing the Polar FFT we have defined is not invertible, or badly conditioned.

A way around this problem is to assume that there is no content in those nonsampled corners. This means that the image should be supported in the frequency domain inside a disk of radius π, leaving the corners empty. If the given image is sampled at $\sqrt{2}$ times the Nyquist rate on both Cartesian axes then its frequency domain support is the square $[-\pi/\sqrt{2}, +\pi/\sqrt{2}]^2$, and this square is contained in the disk required. Thus, given any image, we can easily verify that this condition holds: zero-padding the image by factor $\sqrt{2}$ in both axes, and applying 2D-FFT followed by 2D-IFFT, the result image is twice as big (in pixel count) and its frequency domain support is as required. All these operations add $O(N^2 \log_2(N)$ flops to the Polar FFT that follows. Figure 5 illustrates these frequency supports.

In the inversion transform from the frequency Polar-coordinates to the spatial domain, the knowledge about the corners having zero content is valuable, and could be used to further stabilize the inverse transform. This is done in the interpolation to the pseudo-Polar grid, by assigning zero values to all grid-points outside the π-radius circle.[5]

Figure 6 presents the effect of the corner-nulling process proposed here on the condition number of the transform matrix. For an input array of size $N \times N$ (with N assuming the values $4, 6, 8, \dots, 30$) we compose the transform matrix for the $4N^2$ Polar coordinates and compute its condition number. As can be seen from the graph, this value grows exponentially. As an example, for a moderate size of 30×30 input array, the condition number is 2×10^9, implying that inversion by least-squares will require numerous iterations, and will be highly sensitive to numerical errors.

The second curve refers to the same transform matrix with an augmented part referring to the corners. The augmentation part is obtained by generating a regular $4N \times 4N$ Cartesian grid transform matrix, and choosing the rows

[5] Note that this discussion is not to be confused with the need to apply preconditioning in the inversion of the pseudo-Polar transform.

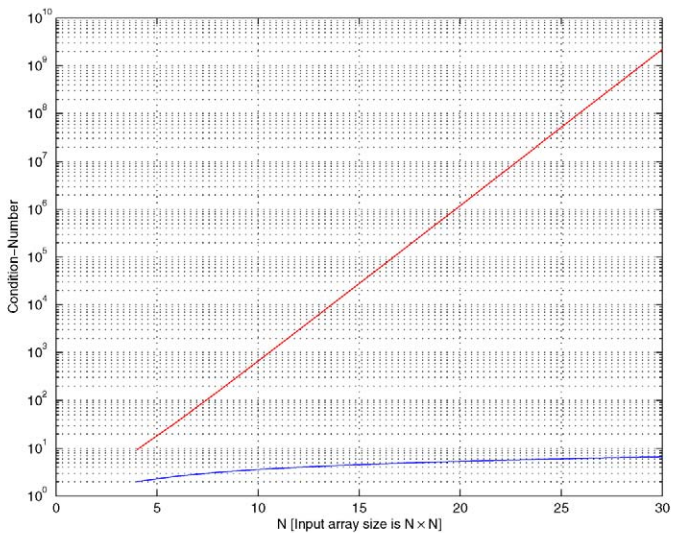

Fig. 6. The condition-number for the transform matrix for a direct Polar-coordinates transform and for an augmented one, that nulls the corners.

that refer to the corners (outside the π-radius disk). Roughly speaking, we have doubled the number of rows, and the condition-number this time is well controlled. For an input array of size 30×30 the condition number is 6.6, meaning that standard iterative techniques of linear algebra achieve a high accuracy inversion.

This numerical experiment comes to show that zero-forcing at the corners stabilizes the inversion process. However, in practice, there is no need for the augmentation as done in the prescribed experiment—all that is needed is nulling values outside the frequency support disk of radius π, and the improved condition-number presented here holds true nevertheless.

4. Analysis of the Polar FFT algorithm

4.1. Experimental evidence—USFFT versus Polar-FFT

We apply the USFFT and the Polar-FFT tools to the image *Lena* (scaled down to size 64×64 pixels, in order to speedup this and later experiments), and compare error behaviors. In particular, we are interested in studying the effect of the oversampling. Figure 7 presents the results of the first experiment.

For the Cartesian-based USFFT method, we use a 2D Hermite interpolation on a 2×2 support [39]. It exploits known Fourier values on the Cartesian grid, along with their horizontal and vertical derivatives (altogether, 12 values), in order to fit a polynomial. Those directional derivatives are obtained by pre-multiplying the image by linear terms and applying the 2D-FFT three times [21]. For the pseudo-Polar based method, the first interpolation stage is also based on Hermite interpolation with a 2-samples support (as for the USFFT, but applied in 1D, and exploiting 4 known values on this support to fit a cubic). The second stage interpolation uses a piecewise cubic spline using the Fourier values alone and forcing alignment of the first derivatives[6] [39]. Thus, the interpolation schemes deployed on our Polar-FFT are inferior to the one practiced for the Cartesian-USFFT. Those schemes remain fixed throughout the reported experiments. As already mentioned in the Introduction, better interpolation methods can be used for both algorithms. Here we chose to concentrate on short-support and relatively simple methods that are easy to understand and implement, and ones that eventually lead to an acceptable level of accuracy, nevertheless.

Figure 7 presents both ℓ^1 and ℓ^2 measures of error. The error presented is a relative error, normalized by the appropriate norm of the ideal result. The horizontal axis in the plots is the overall oversampling ratio. For the Cartesian-based USFFT, for an oversampling factor of S in each axis, the overall oversampling ratio is S^2. Similarly, for our

[6] Evaluating the directional derivatives for this stage requires a replication of the first stage interpolation for those as well, and thus was avoided, settling for a weaker scheme.

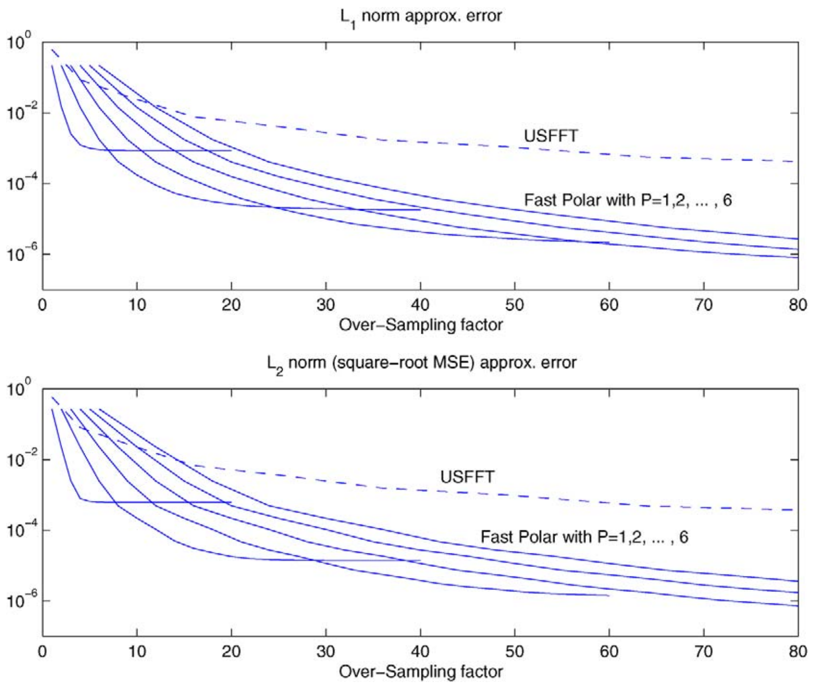

Fig. 7. USFFT and the Polar-FFT error as a function of the over-sampling. The USFFT oversampling is the same along the two axes, while for the PFFT we fix the angular oversampling ($P = 1, 2, \ldots, 6$, and thus 6 curves) and vary the radial one $S = 1, 2, \ldots, 20$.

method, the overall oversampling is PS, where S and P are the oversampling ratios for the number of squares and number of rays. Several choices of P and S are tested.

We see that for the proper choice of P and S ($S \approx 5P$) the Polar-FFT method is far more accurate than the USFFT, under either error measure. For a fixed P there is a saturation effect in the approximation error as $S \to \infty$ because errors caused by the first interpolation stage dominate and cannot be compensated. The higher needed oversampling along the rays can be explained in two ways: (i) the radial interpolation stage of the PFFT uses a lower-order procedure, based on splines instead of Hermite interpolation; and (ii) the radial interpolation task is more difficult as the signal to be interpolated is less smooth (see previous section). Extensive experiments with different signals, their sizes, and metrics of error evaluations, all confirm that the above conclusions comparing the two methods are typical.

4.2. Frequency domain spread of the error

Given the specific image and fixing the oversampling ratio, we study how the errors of the two methods behave in the frequency domain. Intuitively, we know that Cartesian-based USFFT method should cause higher errors near the origin where the interpolation starts from an inherently coarser grid. In contrast, we expect the Polar-FFT to perform equally well in most frequency locations.

Using the same test image from the previous experiment (scaled down *Lena*), Fig. 8 shows the relative errors obtained by the USFFT ($S = 9$ in both axes), and the Polar-FFT ($S = 20,\ P = 4$). The plots show log (with base 10, being crucial for a good visualization) of the absolute value of the transform errors, divided by the mean absolute value of the true transform, so as to get a relative error evaluation. The frequency domain is presented on Cartesian axes in order to give more intuitive frequency-content description. Since the frequency domain is sampled in Polar coordinates in the transforms, we use a Voronoi diagram to slice the plane into pieces per each Polar frequency sample. This is a matter of visual display only.

Our expectations about the distribution of errors in the frequency domain are validated—the Cartesian-USFFT concentrates the error at the origin, while the Polar-FFT errors are spread around. Moreover, the errors obtained by the Polar-FFT are substantially smaller.

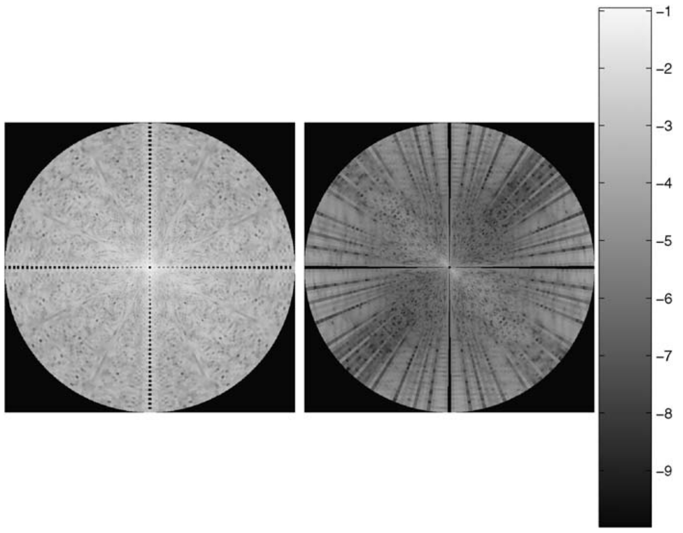

Fig. 8. The relative error as a function of frequency location using a specific image. This figure shows the error obtained by the USFFT method (left) and the PFFT algorithm errors (right). The error scale is relative and logarithmic.

4.3. Frequency domain spread of the error—worst-case analysis

A limitation of the above experiment is that it relies on a specific choice of a signal. We now address the same question from a worst-case point of view: for a specific frequency location in the destination Polar grid, what is the worst possible signal, maximizing the error at this point, and how big is this maximum error? Answering these questions per each location, we can draw a worst-case error map in the frequency domain and compare for both the USFFT and the Polar-FFT methods.

Let x of size $N \times N$ denote the signal to be transformed. The transform result has $2N$ rays containing $2N$ samples each. As all the involved operations are linear, we adopt a matrix–vector representation for the exact ($\mathbf{T}_e$), the USFFT ($\mathbf{T}_u$), and the Polar-FFT ($\mathbf{T}_p$). All these matrices are of size $4N^2 \times N^2$. As to the construction of these matrices, $\mathbf{T}_e$ can be built directly using the definition of the transform in Eq. (1) and the grid defined in (2). The USFFT and the proposed Polar-FFT matrices can be built by applying these transforms on N^2 signals drawn from the trivial basis (having ‘1’ in one location and zeros elsewhere). Every such transform produces a $4N^2$ long vector being a column in the desired matrices.

For the given signal $\underline{x}$ of length N^2 (due to its lexicographic ordering), the transform error for the USFFT is $(\mathbf{T}_e - \mathbf{T}_u)\underline{x}$, with a similar expression for the Polar-FFT method. To maximize this error is to solve

$$\max_{\underline{x}} \frac{\|\mathbf{W}(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\underline{x}\|_2^2}. \tag{17}$$

$\mathbf{W}$ is a diagonal matrix with all entries being zero apart from the one element chosen in the frequency domain. Thus, essentially, only one row of the matrix $\mathbf{T}_e - \mathbf{T}_u$ is used—denote this as $\underline{e}^T$. Clearly, by Cauchy’s inequality, the maximum is obtained for $\underline{x} = \underline{e}$, and the error obtained is $\|\underline{e}\|_2$.

Figure 9 presents these worst-case errors[7] as a function of frequency for both methods when $N = 16$. The Polar-FFT performs far better in all locations. Note that our relatively low choice of $N = 16$ in this experiment causes special sampling effects.

[7] In this and later experiments based on the matrix–vector representation of the transforms, we use low value for N because of the induced matrix sizes. One could use much higher N values if instead of explicitly forming the matrices, the transform and its adjoint operator are applied.

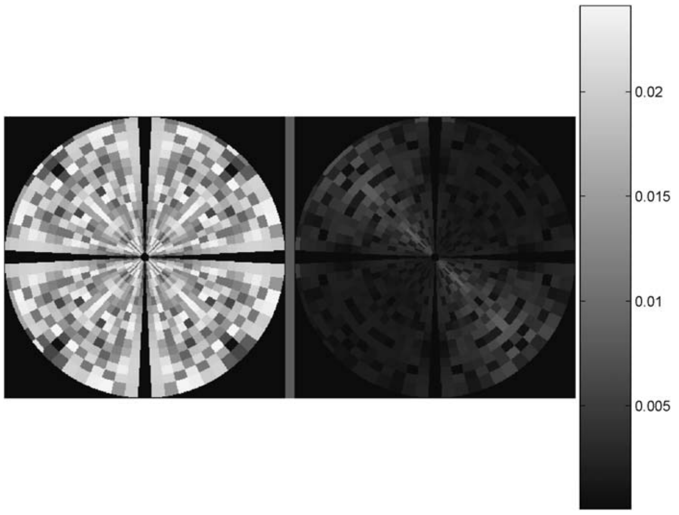

Fig. 9. The error as a function of frequency location: using the worst case signal per location (left: USFFT, right: the proposed Polar-FFT).

4.4. Worst case error via eigenspace analysis

4.4.1. Direct approach

We desire a plot of the worst case ℓ^2 error of each method under a fixed oversampling. Using matrix–vector notation, for a given signal $\underline{x}$, the transform error of the USFFT is $(\mathbf{T}_e - \mathbf{T}_u)\underline{x}$. Now we solve the optimization problem

$$\max_{\underline{x}} \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\underline{x}\|_2^2}, \tag{18}$$

seeking the worst-possible signal $\underline{x}$ maximizing error, subject to unit ℓ^2-norm. The answer is of-course the first right singular vector of $(\mathbf{T}_e - \mathbf{T}_u)$ [40] and the value of (18) is the square of the first singular value (a square-root is required to align with previous error measures).[8]

Figure 10 presents the real and imaginary parts of these worst-case signals for the USFFT ($S = 9$) and the Polar-FFT ($S = 20,\ P = 4$), and the absolute frequency description of this signal. The USFFT's maximal square-root MSE error is 3.6×10^{-3} while the Polar-FFT's is 1.9×10^{-4}. Again, the USFFT method is weaker, and its worst signal is concentrated near the frequency origin where the method is weakest. Note that the worst signal is modulated (notice the shift from the center in the spatial domain) to result in a very nonsmooth frequency behavior.

4.4.2. Relative approach

A major problem with the above analysis is the difficulty in understanding the meaning of the error found, being a ratio between energies in the frequency and the spatial domains. An interesting alternative is the definition of worst signals by

$$\max_{\underline{x}} \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\mathbf{T}_e\underline{x}\|_2^2}. \tag{19}$$

Put in words, we seek the worst error relative to transform size in Polar frequency coordinates. This problem amounts to a generalized eigenvalue problem [40]. Figure 11 presents the results for Cartesian USFFT method (maximal error is 3.6×10^{-4}) and Polar-FFT method (worst error is 4.5×10^{-5}). Both methods lead to similar worst-case signals,

[8] Alternatively, the objective function defined in (18) can be maximized by an eigendecomposition of the symmetric and positive semi-definite matrix $(\mathbf{T}_e - \mathbf{T}_u)^H(\mathbf{T}_e - \mathbf{T}_u)$, and taking the eigenvector that corresponds to the largest eigenvalue. This is theoretically equivalent to the SVD approach mentioned above.

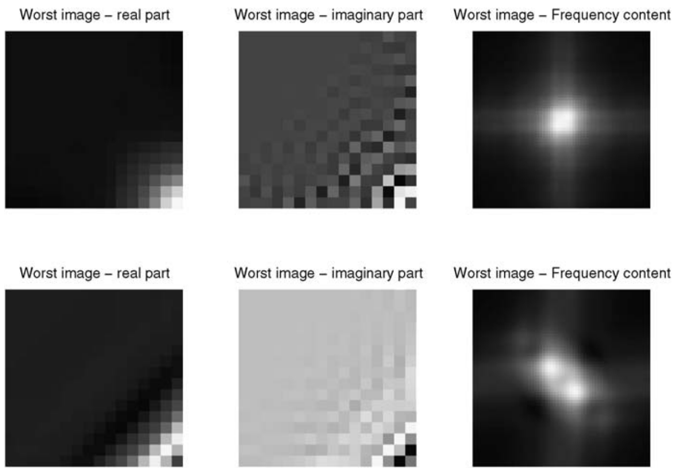

Fig. 10. Worst case signal—direct eigenvalue approach. The top row shows the worst signal for the USFFT—the real spatial part (left), the imaginary spatial part (middle), and its spectrum (right). The bottom row shows the same for the PFFT.

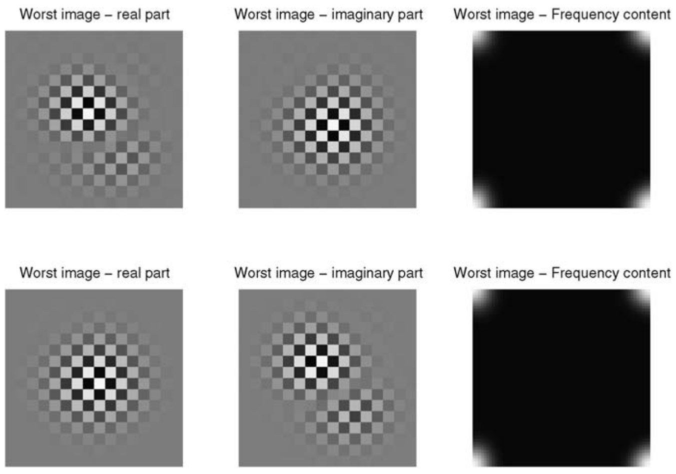

Fig. 11. Worst case signal—relative eigenvalue approach. The top row shows the worst signal for the USFFT—the real spatial part (left), the imaginary spatial part (middle), and its spectrum (right). The bottom row shows the same for the PFFT.

with energy mostly falling outside the circle of radius π, so that the denominator in the above definition is nearly zero. Nevertheless, our Polar-FFT has an advantages over Cartesian coordinates.

4.4.3. *Relative approach with support constraint*

In order to avoid signals of the sort obtained earlier, we seek the worst case signal for (19) with the side-constraint of no energy in the frequency domain outside the circle of radius π. Thus we solve

$$\max_{\underline{x}} \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\mathbf{T}_e\underline{x}\|_2^2} \quad \text{subject to} \quad \mathbf{F_1}\underline{x} = 0, \tag{20}$$

where $\mathbf{F_1}$ represents the regular Cartesian FFT in a predetermined density, restricted to frequency points outside the circle. Alternatively, we reformulate this using

$$\max_{\{\underline{x}|\mathbf{F_1}\underline{x}=0\}} \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\mathbf{T}_e\underline{x}\|_2^2} \approx \max_{\underline{x}} \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{x}\|_2^2}{\|\mathbf{T}_e\underline{x}\|_2^2 + \lambda\|\mathbf{F_1}\underline{x}\|_2^2}. \tag{21}$$

With this formulation we again have a generalized eigenvalue problem. For large λ, (21) yields an approximate solution of the original problem (20). We chose $\lambda = 1000$ and find that higher values does not change the results

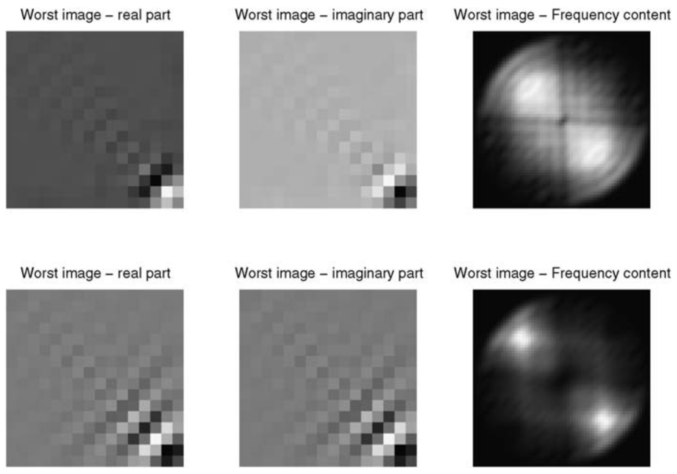

Fig. 12. Worst case signal—relative eigenvalue approach with constraint. The top row shows the worst signal for the USFFT—the real spatial part (left), the imaginary spatial part (middle), and its spectrum (right). The bottom row shows the same for the PFFT.

perceivably. Those results are shown in Fig. 12. The USFFT worst error is 6.5×10^{-5} and with the Polar-FFT method the error we obtain is 4.2×10^{-6}.

4.5. Signal-space ordering via eigenspace analysis

We return to the definition of error in Eq. (19). In solving the maximization problem for the Polar-FFT we find the worst signal and the accompanying worst error. This worst signal represents only a one-dimensional subspace of signals, and for signals orthogonal to it, all we can say is that the error on their transform computation is expected to be smaller, though we cannot say by how much.

Solving the same problem again while forcing the result to be orthogonal to the previous eigenvector (the already found worst-possible signal), we obtain a second worst rank-1 sub-space. Repeating this process, we essentially find ranked orthonormal basis that represents the signal-space $\mathbf{C}^{N\times N}$, such that the first vector spans the worst signals, the second spans the "next" worst signals, and so on. The process described above is simply an eigenvalue problem for the matrix

$$(\mathbf{T}_e - \mathbf{T}_u)^H(\mathbf{T}_e - \mathbf{T}_u).$$

The eigenvectors are the ranked ortho-basis and the eigenvalues are related squared errors. Denote the results of the above process on the Polar-FFT error by $\{\underline{u}_k\}_{k=1}^{k=N^2}$, such that $\underline{u}_{N^2}$ is the worst signal. Then the sequence

$$\{\lambda_k\}_{k=1}^{N^2} = \left\{ \frac{\|(\mathbf{T}_e - \mathbf{T}_p)\underline{u}_k\|_2^2}{\|\underline{u}_k\|_2^2} \right\}_{k=1}^{N^2} \tag{22}$$

measures the errors induced by each subspace, from ranked best to the worst. Using the same ortho-basis, we may compute

$$\{\delta_k\}_{k=1}^{N^2} = \left\{ \frac{\|(\mathbf{T}_e - \mathbf{T}_u)\underline{u}_k\|_2^2}{\|\underline{u}_k\|_2^2} \right\}_{k=1}^{N^2}, \tag{23}$$

the errors we will effectively obtain from the Cartesian USFFT method on the same subspaces. Plotting these two sequences in Fig. 13, we compare the distribution of errors by subspace and obtain a complete picture about the transform errors of the two methods for all the signal space.

Figure 14 is similar but uses (20) to obtain the ortho-basis and the errors for its subspaces. In this case the eigenvalue problem becomes a generalized eigenvalue one. Similarly, Fig. 15 presents results from the constrained generalized eigenvalue problem (21). All these figures show a consistent superiority (and even a growing one, as we turn from the worst-case direction to the remaining subspace) of our Polar-FFT over the USFFT method.

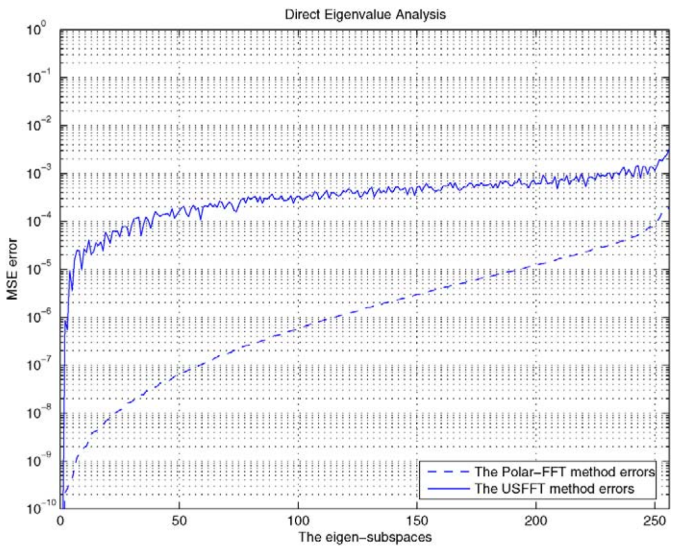

Fig. 13. The ranked signal space—the direct eigenvalue approach.

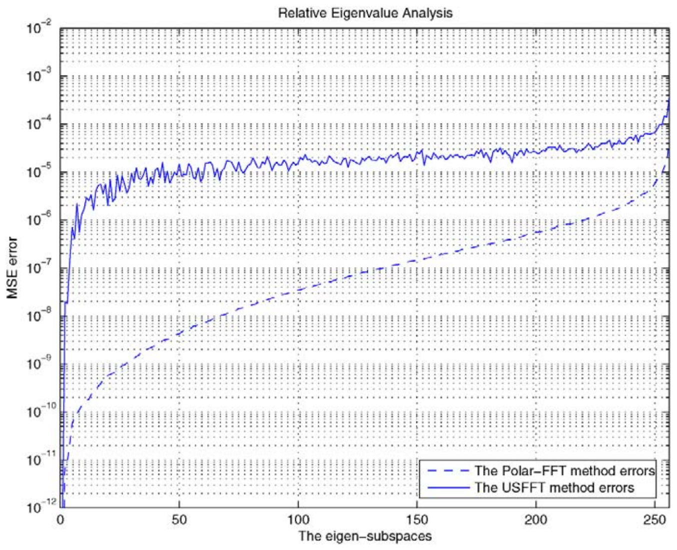

Fig. 14. The ranked signal space—the relative eigenvalue approach.

5. Available software and reproducible results

One of the main objectives of this project has been development of a software tool that performs the computations given here. We intend to make this toolbox available to the public, in a fashion parallel to `Wavelab` and `Beamlab` libraries [41,42]. Our implementation uses Matlab code to perform various tasks around the idea of Polar FFT. The PFFT toolbox is freely available in http://www.cs.technion.ac.il/~elad/PolarFFT.zip.

As part of the freely available software, we supply code reproducing every figure in this paper that is based on computation. We believe that this will help researchers entering this area, and engineers interested in better understanding the details behind the verbal descriptions supplied here. This also agrees with the recent emerging standard of guaranteeing reproducible results in scientific publications [41].

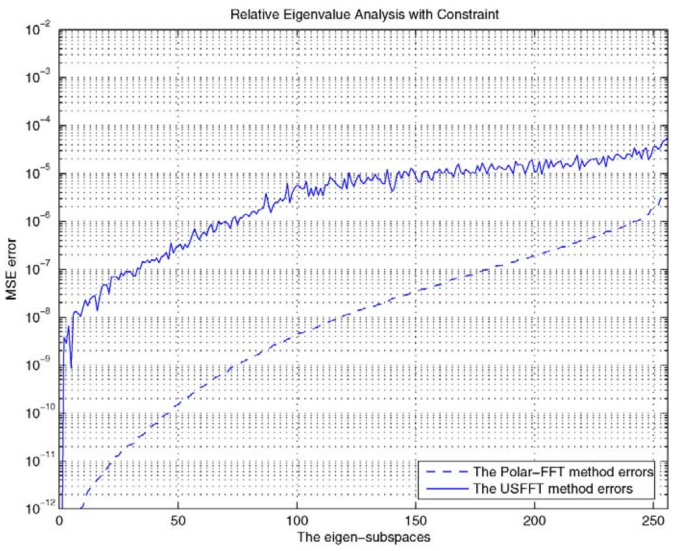

Fig. 15. The ranked signal space—the relative eigenvalue approach with constraints.

6. Conclusions

In this article we have developed a rapid and accurate Polar FFT. For a given two-dimensional signal of size $N \times N$, the proposed algorithm produces a Polar FFT with $C \times N^2$ coordinate samples, where C is a chosen parameter (e.g., $C = 4$). The complexity of this algorithm is $O(CN^2 \log N)$, just like in a Cartesian 2D-FFT. Two special features of this algorithm are (i) it involves only 1D equispaced FFT's and 1D interpolations, leading to a highly cache-aware algorithm; and (ii) it has very high accuracy for moderate oversampling factors.

The accuracy of the algorithm was studied both for specific images and from a worst-case standpoint. The new approach is shown to be far more accurate than Cartesian-based Unequally-Spaced Fast Fourier Transform (USFFT). Our analysis shows that the proposed scheme produces highly accurate transform values near the origin, where "typical" signals concentrate on. However, these comparative results are restricted to the use of simple and small-support interpolations and no pre-compensation. Further work is required in order to incorporate more advanced techniques from the USFFT literature.

In the present paper we emphasize the basic construction, leaving aside applications, improvements, and extensions. We believe that this paper serves the researchers/engineers interested in applying those tools in applications, offering a reasonable confidence in the accuracy and speed. The supplied software, along with this documentation, may make the adoption of these methods relatively painless.

Future work on this topic might consider various improvements to the interpolation stages implemented in our algorithm, using recent results in the USFFT literature (e.g., [28,29]). The proposed algorithm might have applications in stable and rapid inversion of Radon transform, fast image rotation and registration, and elsewhere. Extension to 3D will require definition of Polar-like[9] coordinates.

Appendix A. Chirp-Z transform (FRFT)

Definition 3. Given the discrete signal $x[n]$ over the support $0 \leqslant n < N$, and given an arbitrary scalar value α, the Chirp-Z Transform (also known as the Fractional Fourier Transform) is given by

$$\hat{x}[k] = \sum_{n=0}^{N-1} x[n] \exp\left(-\frac{i2\pi kn}{N} \cdot \alpha\right) \quad \text{for } 0 \leqslant k < N. \tag{A.1}$$

[9] There is no exact Polar sampling method for 3D, thus we must settle for Polar-like grids.

As we have seen through the definition of the pseudo-Polar Fourier transform, we need to perform this kind of transform as part of our process. The following theorem guarantees existence of a highly efficient algorithm for this transform. This theorem, and more importantly, the efficient process itself, are described in [37,38].

Theorem 4. *Evaluation of the Chirp-Z transform can be done with order of* $30N\log N$ *operations.*

Proof. Our proof will be given by construction of such an efficient process. Our starting point is the trivial assignment $2kn = k^2 + n^2 - (k-n)^2$. Using this relationship into the definition above we obtain

$$\begin{aligned}\hat{x}[k] &= \sum_{n=0}^{N-1} x[n]\exp\biggl(-\frac{i\pi\alpha}{N}\bigl(k^2+n^2-(k-n)^2\bigr)\biggr)\\ &= \exp\biggl(-\frac{i\pi\alpha}{N}k^2\biggr)\sum_{n=0}^{N-1} x[n]\cdot\exp\biggl(-\frac{i\pi\alpha}{N}n^2\biggr)\cdot\exp\biggl(\frac{i\pi\alpha}{N}(k-n)^2\biggr).\end{aligned} \tag{A.2}$$

If we define the sequence

$$s[n] = \exp\biggl(-\frac{i\pi\alpha}{N}n^2\biggr),$$

then the above equation can be rewritten as

$$\hat{x}[k] = s[k]\sum_{n=0}^{N-1} x[n]s[n]s[k-n]. \tag{A.3}$$

This equation suggests a way to evaluate the transform, starting with pre-multiplication of $x[n]$ by $s[n]$, followed by a convolution with $s[n]$, and finally, post-multiplying with $s[n]$ again.

The main saving in computation comes from replacing convolution by a 1D-FFT based method. A 1D-FFT transform should be applied on the two sequences, multiply the results, and apply inverse transform on the resulting sequence. FFT should be computed for the two sequences with zero padding to length $2N$ in order to avoid cyclic effects.

Three 1D-FFT's are needed, costing each $10N\log N$ operations. The creation of $s[n]$ and the pre-/post-multiplication by it require $O(N)$ additional operations. Thus, an overall of $30N\log N$ operations is needed for the entire process, as claimed by the theorem. □

References

[1] J. Dongarra, F. Sullivan, Introduction the top 10 algorithms, Comput. Sci. Eng. 2 (1) (2000) 22–79. Special issue.
[2] B. Briggs, E. Henson, The FFT: An Owner's Manual for the Discrete Fourier Transform, SIAM, Philadelphia, 1995.
[3] A. Averbuch, R.R. Coifman, D.L. Donoho, M. Elad, M. Israeli, Accurate and fast Polar Fourier transform, in: The 37th Asilomar on Signals, Systems and Computers, Pacific Grove, CA, 2003.
[4] F. Natterer, The Mathematics of Computerized Tomography, John Wiley & Sons, New York, 1986.
[5] H. Stark, J.W. Woods, I. Paul, R. Hingorani, Direct Fourier reconstruction in computer tomography, IEEE Trans. Acoust., Speech Signal Process. 29 (2) (1981) 237–245.
[6] H. Fan, J.L.C. Sanz, Comments on 'Direct Fourier reconstruction in computer tomography', IEEE Trans. Acoust., Speech Signal Process. 33 (2) (1985) 446–449.
[7] S. Matej, M. Vajtersic, Parallel implementation of the direct Fourier reconstruction method in tomography, Comput. Artificial Intelligence 9 (4) (1990) 379–393.
[8] K.J. Moraski, D.C. Munson Jr., Fast tomographic reconstruction using Chirp-Z interpolation, in: Proc. 25th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, IEEE Comput. Soc. Press, Los Alamitos, CA, 1991, pp. 1052–1056.
[9] A.H. Delaney, Y. Bresler, A fast and accurate Fourier algorithm for iterative parallel-beam tomography, IEEE Trans. Image Process. 5 (5) (1996) 740–753.
[10] H. Choi, D.C. Munson Jr., Direct-Fourier reconstruction in tomography and synthetic aperture radar, Int. J. Imaging Systems Technol. 9 (1) (1998) 1–13.
[11] J. Walden, Analysis of the direct Fourier method for computer tomography, IEEE Trans. Medical Imaging 19 (3) (2000) 211–222.
[12] D. Gottleib, B. Gustafsson, P. Forssen, On the direct Fourier method for computer tomography, IEEE Trans. Medical Imaging 19 (3) (2000) 223–232.

[13] F. Natterer, F. Wübbeling, Mathematical Methods in Image Reconstruction, SIAM, Philadelphia, 2000.

[14] S. Basu, Y. Bresler, An $O(n^2 \log n)$ filtered backprojection reconstruction algorithm for tomography, IEEE Trans. Image Process. 9 (10) (2000) 1769–1773.

[15] K. Fourmont, Non-equispaced fast Fourier transforms with applications to tomography, J. Fourier Anal. Appl. 9 (2003) 431–450.

[16] S. Matej, J.A. Fessler, I.G. Kazantsev, Iterative tomographic image reconstruction using Fourier-based forward and back-projectors, IEEE Trans. Medical Imaging 23 (4) (2004) 401–412.

[17] E. Suli, A. Ware, A spectral method of characteristics for hyperbolic problems, SIAM J. Numer. Anal. 28 (2) (1991) 423–445.

[18] J.P. Boyd, Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast Fourier transform, J. Comput. Phys. 103 (2) (1992) 184–186.

[19] J.P. Boyd, A fast algorithm for Chebyshev, Fourier, and sinc interpolation onto an irregular grid, J. Comput. Phys. 103 (2) (1992) 243–257.

[20] G. Beylkin, On the fast Fourier transform of functions with singularities, Appl. Comput. Harmon. 2 (1995) 263–381.

[21] C. Anderson, M.D. Dahleh, Rapid computation of the discrete Fourier transform, SIAM J. Sci. Comput. 17 (1998) 913–919.

[22] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data, SIAM J. Sci. Comput. 14 (1993) 1368–1393.

[23] A. Dutt, V. Rokhlin, Fast Fourier transforms for nonequispaced data II, Appl. Comput. Harmon. Anal. 2 (1995) 85–100.

[24] A.F. Ware, Fast approximate Fourier transform for irregularly spaced data, SIAM Rev. 40 (4) (1998) 838–856.

[25] D. Potts, G. Steidl, M. Tasche, Fast Fourier transforms for nonequispaced data: A tutorial, in: J.J. Benedetto, P. Ferreira (Eds.), Modern Sampling Theory: Mathematics and Application, Birkhäuser, 2000, pp. 253–274, ch. 12.

[26] A.J.W. Duijndam, M.A. Schonewille, Nonuniform fast Fourier transform, Geophysics 64 (2) (1999) 539–551.

[27] N. Nguyen, Q.H. Liu, The regular Fourier matrices and nonuniform fast Fourier transforms, SIAM J. Sci. Comput. 21 (1) (1999) 283–293.

[28] J.A. Fessler, B.P. Sutton, Nonuniform fast Fourier transforms using min–max interpolation, IEEE Trans. Signal Process. 51 (2) (2003) 560–574.

[29] L. Greengard, J.Y. Lee, Accelerating the nonuniform fast Fourier transform, SIAM Rev. 46 (2004) 443–454.

[30] A. Averbuch, R. Coifman, D.L. Donoho, M. Israeli, Fast Slant Stack: A notion of Radon transform for data in a Cartesian grid which is rapidly computible, algebraically exact, geometrically faithful and invertible, Technical Report, Statistics Department, Stanford University, 2003.

[31] R.M. Mersereau, A.V. Oppenheim, Digital reconstruction of multidimensional signals from their projections, Proc. IEEE 62 (10) (1974) 1319–1338.

[32] J.E. Pasciak, A note on the Fourier algorithm for image reconstruction, Preprint AMD 896, Applied Mathematics Department, Brookhaven National Laboratory, Upton, NY, 1981.

[33] P. Edholm, G.T. Herman, Linograms in image reconstruction from projections, IEEE Trans. Medical Imaging MI-6 (4) (1987) 301–307.

[34] W. Lawton, A new Polar Fourier-transform for computer-aided tomography and spotlight synthetic aperture radar, IEEE Trans. Acoust., Speech Signal Process. 36 (6) (1988) 931–933.

[35] A. Averbuch, Y. Shkolnisky, 3D Fourier based discrete Radon transform, Appl. Comput. Harmon. Anal. 15 (1) (2003) 33–69.

[36] Y. Keller, A. Averbuch, M. Israeli, A pseudoPolar FFT technique for translation, rotation and scale-invariant image registration, IEEE Trans. Image Process. 14 (1) (2005) 12–22.

[37] L.R. Rabiner, R.W. Schafer, C.M. Rader, The Chirp Z-transform algorithm and its application, Bell System Tech. J. 48 (1969) 1249–1292.

[38] D.H. Bailey, P.N. Swarztrauber, The fractional Fourier transform and applications, SIAM Rev. 33 (3) (1991) 389–404.

[39] G. Strang, Introduction to Applied Mathematics, Wellesley/Cambridge Press, 1986.

[40] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., John Hopkins Univ. Press, 1996.

[41] J. Buckheit, D.L. Donoho, WaveLab and Reproducible Research, Wavelets and Statistics, Springer-Verlag, Berlin, 1995.

[42] C.S. Choi, D.L. Donoho, A.G. Flesia, X. Huo, O. Levi, D. Shi, About BeamLab a Toolbox for New Multiscale Methodologies, Stanford University—Statistics Department Technical Report, http://www-stat.stanford.edu/~beamlab, 2002.