Progress In Electromagnetics Research M, Vol. 23, 279–298, 2012

GPU-BASED ω-k TOMOGRAPHIC PROCESSING BY 1D NON-UNIFORM FFTS

A. Capozzoli*, C. Curcio, and A. Liseno

Dipartimento di Ingegneria Biomedica, Elettronica e delle Telecomunicazioni (DIBET), Università di Napoli Federico II, via Claudio 21, I 80125 Napoli, Italy

Abstract—We present an ω-k approach based on the use of a 1D Non-Uniform FFT (NUFFT) routine, of NER (Non-Equispaced Results) type, programmed on a GPU in CUDA language, amenable to real-time applications. A Matlab main program links, via mex files, a compiled parallel (CUDA) routine implementing the NUFFT. The approach is shown to be an extension of an already developed parallel algorithm based on standard backprojection processing to account also for near-field data. The implementation of the GPU-based, parallel NUFFT routine is detailed and the computational advantages of the developed approach are highlighted against other confronted sequential or parallel (on multi-core CPU) procedures. Furthermore, the benefits of the ω-k, NUFFT-based processing are pointed out by both comparing its accuracy and computational convenience against other interpolators, and by providing numerical results. By comparing the computational performance of the algorithm against a multi-core, Matlab implementation, the speedup has been about 20 for a medium size image. The performance of the approach has been pointed out in the applicative case of vegetation imaging against experimental data of a boxtree (*Buxus* tree), also under a source of temporal decorrelation (wind).

1. INTRODUCTION

ω-k processing, originated from seismics engineering and geophysics [1], is commonly accepted to be an ideal solution to Synthetic Aperture Radar (SAR) focuses on the case of spaceborne data [2], and, since recently, is becoming also popular to process airborne SAR data

Received 30 August 2011, Accepted 9 October 2011, Scheduled 5 March 2012

* Corresponding author: Amedeo Capozzoli (a.capozzoli@unina.it).

following a motion compensation step [3], airborne bistatic SAR data [4], ground-based SAR data [5, 6] and also Ground Penetrating Radar (GPR) [7, 8].

Different from range-doppler [9] and chirp-scaling [10] processing used in SAR, ω-k exploits an "exact" version of the kernel of the operator linking the on-ground reflectivity distribution to the data acquired by the sensor. In this way, the algorithm fully compensates for the "range migration" and is applicable also to the near-field data case [7, 8].

ω-k regards the data as the samples of the Fourier transform of the reflectivity function. However, such samples are arranged on a non-uniform (polar) grid, so that, prior to exploit 2D Fast Fourier Transform (FFT) algorithms, a Stolt interpolation problem must be solved [1, 10]. This step is time consuming since typically polynomial or sinc interpolators are used and the selection of the interpolator is then critical for obtaining a satisfactory trade-off between computational performance and image quality [11, 12].

Taking into account the needs of processing speed and focusing accuracy, the recent development of algorithms for rapidly and accurately performing Discrete Fourier Transforms (DFTs) from/to nonuniform grids (Non-Uniform DFTs-NUDFTs), also known as Non-Uniform Fast Fourier Transform (NUFFT) algorithms [13, 14], is of certain interest in radar processing [15–17]. Indeed, NUFFT schemes nest interpolation and Fourier transformation in a single step, they are accurate since they exploit the bandlimitedness information of the signal to be transformed and their computational cost is of the same order of standard FFTs, i.e., it grows asymptotically as $O(N^2 \log N)$, $N \times N$ being the image size.

At the same time, the issue of accurately focusing SAR images in real-time attracts more and more interest in a number of applications including disaster observation and management, surveillance of military sites, seismic studies and flood monitoring [18, 19]. This is of course possible provided that performing algorithms and hardware are available. Concerning the hardware, while commercial CPUs nowadays employ 4/8 cores, modern Graphics Processing Units (GPUs), like the NVIDIA C2050 employed in this paper, reach 448 cores. The introduction of graphical APIs in high-level language codes is furthermore being dramatically simplified by

- multicore functionalities in common high-level platforms for technical computing, as the Matlab Parallel Computing Toolbox;
- the "multicore-aware" functions in Matlab [20];
- Matlab code GPU accelerators, as Accelereyes Jacket [21];

- the development of programming languages, as CUDA (Computer Unified Device Architecture [22–24]);
- the development of C++ libraries, as Accelereyes LibJacket [25].

All that is raising the interest of the radar community into parallel programming [19, 26, 27].

Throughout the literature, processing algorithms based on parallel NUFFT implementations, on CPU [28], GPU [29–33] or Field Programmable Gate Arrays (FPGAs) [34], have been developed, mainly for applications of Magnetic Resonance Imaging (MRI), antenna fast analysis or Optical Coherence Tomography (OCT). In these instances, transformation of 2D signals from non-uniform to uniform grids or vice versa is necessary, thus requiring 2D Non-Equispaced Data (NED) or Non-Equispaced Results (NER) NUFFT routines, depending on the application of interest. More in detail, the NED and NER acronyms refer to NUFFTs performed from a non-uniform spatial grid to a uniform spectral one and from a uniform spatial lattice to a non-uniform spectral one, respectively.

In this paper, we present the development of an ω-k approach based on the use of a 1D NER NUFFT routine programmed on a GPU in CUDA language. At the best of our knowledge, this is the very first time that NUFFT-based parallel processing is applied to the ω-k algorithm and that a 1D NER NUFFT routine is coded in CUDA language. At variance with [29–33], the considered routine is a 1D Non-Equispaced Result (NER) NUFFT, fastly implementing a NUDFT, transforming a regularly sampled signal into a non-uniformly sampled one. More in detail, while the use of a 2D NUFFT requires the full data acquisition, the scheme presented in this paper is amenable of a very fast processing meanwhile the data are acquired.

The approach is conveniently coded in Matlab language, by linking, via mex files, a compiled parallel (CUDA) routine implementing the NUFFT [35, 36] and is shown to be an extension of an already developed parallel algorithm based on standard backprojection processing [26, 27] to account also for near-field data.

The implementation of the GPU-based, parallel NUFFT routine is detailed and the computational advantages of the developed approach are highlighted against other confronted sequential or parallel (on multi-core CPU) procedures. Furthermore, the benefits of the ω-k, NUFFT-based processing are pointed out by providing an experimental validation on data from a boxetree (*Buxus* tree) acquired in an indoor environment, also under a source of temporal decorrelation (wind).

The paper is organized as follows. In Section 2, the ω-k algorithm is briefly recalled and the data processing by a 1D NUFFT illustrated. In Section 3, the parallel GPU implementation of the 1D NER NUFFT

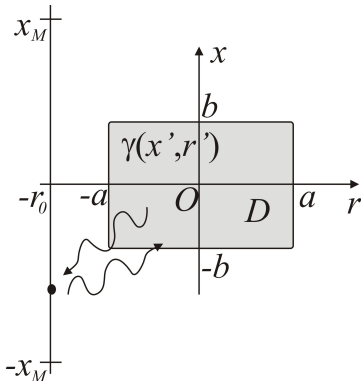

Figure 1. Geometry of the problem.

is described. Section 4 is devoted to present the computational software and hardware platforms as well as the experimental setup herewith employed to analyze the performance of the approach for vegetation imaging. In Section 5, the computational performance of the algorithm as well as its imaging capabilities are discussed. Finally, in Section 6, the conclusions are drawn.

2. THE ω-k ALGORITHM

Let us consider the 2D geometry in Figure 1, where the domain to be imaged is $D = [-a, a] \times [-b, b]$. The sensor illuminates the scene from the point $(x, -r_0)$, x belonging to $(-x_M, x_M)$, at different angular frequencies $\omega = 2\pi f$, f being the frequency, and at a generic polarization (e.g., vertical or horizontal) and measures a (possibly different) polar component of the scattered wave.

The acquired signal is modeled as [37]

$$\psi(x,\omega) = \iint_D \gamma\left(x', r'\right) e^{-j2\beta\sqrt{(x-x')^2+(r_0+r')^2}} dx' dr' \qquad (1)$$

where γ is the scene reflectivity distribution assumed to be frequency independent, $\beta = \omega/c$ is the wavenumber, and c is the speed of light. For simplicity and without any loss of generality, range spreading attenuation and antenna directivity effects have been neglected. In the SAR context, ψ represents the recorded raw data in frequency domain, after range compression if the scene is illuminated by the usual chirp pulse.

By resorting to the plane wave expansion of the kernel function in Equation (1) [38], then the raw data can be rewritten, apart from

unessential factors and neglecting the evanescent part of the plane wave spectrum, as

$$\psi(x,\omega) = \iint\limits_{D} \gamma\left(x',r'\right) \int_0^{2\beta} \frac{e^{-jk_r(r'+r_0)} e^{jk_x(x-x')}}{k_r} dk_x dr' dx', \qquad (2)$$

where $k_r = \sqrt{4\beta^2 - k_x^2}$ (Stolt mapping). Equation (2) can be also rewritten as

$$\psi(x,\omega) = \int_0^{2\beta} \frac{1}{k_r} e^{-jk_r r_0} e^{jk_x x} \Gamma(k_x,k_r) dk_x \qquad (3)$$

where $\Gamma(k_x,k_r)$ is the Fourier transform of $\gamma(x',r')$. Accordingly, on defining the function $\Psi(k_x,\omega)$ as

$$\Psi(k_x,\omega) = \int_{-\infty}^{\infty} \psi(x,\omega) e^{-jk_x x} dx, \qquad (4)$$

i.e., as the Fourier transform of $\psi(x,\omega)$ with respect to x, then Γ can be expressed as

$$\Gamma(k_x,k_r) = k_r \Psi(k_x,\omega) e^{jk_r r_0}. \qquad (5)$$

In other words, the spectral samples of the reflectivity function $\Gamma(k_x,k_r)$ are related to the Fourier transform of the raw data $\Psi(k_x,\omega)$. Since Ψ is available on a uniform grid of the (k_x,ω) plane, and due to the nonlinear mapping $\omega = (c/2)\sqrt{k_r^2 + k_x^2}$, then Γ is available on a nonuniform grid of the (k_x,k_r) plane. As a consequence (again apart from unessential factors),

$$\gamma(r',x') = \iint\limits_{(k_x,k_r)} \frac{k_r}{\beta} \Psi\left(k_x, \frac{c}{2}\sqrt{k_r^2+k_x^2}\right) e^{j[k_r(r'+r_0)+k_x x']} dk_r dk_x \qquad (6)$$

and an interpolation stage would be needed to bring the spectral data Ψ back to a cartesian grid in the (k_x,k_r) domain to then use a standard, 2D inverse FFT algorithm to determine $\gamma(x',r')$.

As an alternative, by letting $k_r = 2(\omega/c)\cos\phi$ and $k_x = 2(\omega/c)\sin\phi$, then Equation (6) can be rewritten as

$$\gamma\left(r',x'\right) = \int\limits_{\phi} \cos\phi \int\limits_{\omega} \omega \widetilde{\Psi}(\omega,\phi) e^{j2\frac{\omega}{c}[\cos\phi(r'+r_0)+\sin\phi x']} d\omega d\phi, \qquad (7)$$

where $\widetilde{\Psi}(\omega,\phi) = \Psi(2(\omega/c)\cos\phi,\omega)$. Now, since $\widetilde{\Psi}$ is uniformly sampled in ω, then, for each value of ϕ, the internal integral in ω can be evaluated by a 1D NER NUFFT [13].

Indeed, on defining an (r'_l, x'_l) grid for D, an estimate of γ can be expressed (removing unessential constants) as

$$\gamma(r'_l, x'_l) = \sum_p \cos\phi_p \underbrace{\sum_k k\widetilde{\Psi}\left(k\Delta\omega, p\Delta\phi\right) e^{jk\Delta\omega t_l^p}}. \tag{8}$$

where $t_l^p = [\cos\phi_p(r'_l+r_0)+\sin\phi_p x'_l]/c$, the uniform frequency sampling grid is defined by $k\Delta\omega$ and the uniform sampling in the ϕ variable is defined by $p\Delta\phi$. On taking into account that a 1D NER NUDFT is defined as

$$\hat{z}_l = \sum_{k=-N/2}^{N/2} z_k e^{-2\pi j y_l k/N} \tag{9}$$

then, for each value of ϕ_p, the bracketed term in Equation (8) can be evaluated by a 1D NER NUFFT of the vector $z_k = k\widetilde{\Psi}(k\Delta\omega, p\Delta\phi)$ over the output grid $y_l = -N\Delta\omega t_l^p/(2\pi)$.

We remark that the approach at hand can be regarded as a generalization of the standard backprojection algorithm [26, 27]. Indeed, when the acquisition domain is in the far-field zone of the investigated region, then the acquired field $\psi(x,\omega)$ becomes proportional to the plane wave spectrum of the scattered field $\cos\phi\Psi(2(\omega/c)\cos\phi)$, where ϕ is now $\arctan(x/r_0)$. Consequently

$$\psi(x,\omega) \propto \cos\phi\widetilde{\Psi}(\omega,\phi), \tag{10}$$

so that Equation (7) can be approximately rewritten, apart from constant proportionality factors, as

$$\gamma(r', x') = \iint\limits_{(\phi,\omega)} \omega\psi(r_0\tan\phi,\omega)e^{j2\frac{\omega}{c}[\cos\phi(r'+r_0)+\sin\phi x']}d\omega d\phi. \tag{11}$$

Equation (11) now coincides with the reconstruction formula of the standard backprojection [26, 27].

We further remark that, in [39, 40], the chirp-z transform has been applied to SAR data focusing. However, as also stressed in [39], such an algorithm is capable to computing a certain class of NUDFTs only, so that the 1D NER NUFFT herewith employed has a wider range of applicability.

3. THE PARALLEL NUFFT ALGORITHM

The NUFFT algorithm employed in this paper is that developed in [13], which is based on the use of Kaiser-Bessel interpolation windows and

offers better performance as compared to implementations exploiting other, e.g., Gaussian, interpolation kernels [14].

The main idea of the NUFFT algorithm is to approximate the "nonuniform" exponential $\exp(-j2\pi y_l k/N)$ by interpolating few, "oversampled", "uniform" exponentials according to [13]

$$e^{-j2\pi y_l \frac{k}{N}} = \frac{(2\pi)^{-1/2}}{\Phi(2\pi k/(cN))} \sum_{m\in\mathbb{Z}} \hat{\Phi}(cy_l - m) e^{-j2\pi jm \frac{k}{cN}} \tag{12}$$

where $c > 1$ is an "oversampling factor", Φ is the Kaiser-Bessel window, and $\hat{\Phi}$ is its Fourier transform.

By exploiting (12), Equation (9) can be rewritten as

$$\hat{z}_l = \frac{1}{\sqrt{2\pi}} \sum_{m\in\mathbb{Z}} \hat{\Phi}(cy_l - m) \underbrace{\sum_{m=-N/2}^{m=N/2} e^{-j2\pi m \frac{k}{cN}} \frac{z_l}{\Phi(2\pi k/(cN))}}_{U_m}. \tag{13}$$

Taking now into account that $\hat{\Phi}$ has finite support, then Equation (13) takes the form

$$\hat{z}_l \simeq \frac{1}{\sqrt{2\pi}} \sum_{p=-K}^{K} \hat{\Phi}(p) U_{p+\mu_l}, \tag{14}$$

where K is 3 or 6 for single or double precision arithmetics, respectively, μ_l is the nearest integer to cx_l and the subscript $p + \mu_l$ has to be considered as cN-periodic.

Accordingly, the NUDFT can be effectively evaluated in three steps

(i) **Scaling and zero padding**:

$$u_k = \begin{cases} 0 & k = -cN/2, \ldots, -N/2 - 1 \\ z_k/\phi_k & k = -N/2, \ldots, N/2 - 1 \\ 0 & k = N/2, \ldots, cN/2 - 1 \end{cases}; \tag{15}$$

(ii) **FFT of** $\{u_k\}_{k=-cN/2}^{cN/2-1}$ **on** cN **points to obtain** $\{U_m\}_{m=-cN/2}^{cN/2-1}$;

(iii) **Cyclic convolution to evaluate Equation (14)**.

The first step is inherently parallel.

The second step can be easily performed by a library parallel FFT routine. In this paper, the cuFFT routine, provided by NVIDIA, is adopted [41].

The convolution step is by far the most time consuming step [30, 32] and deserves more attention. An effective implementation has been obtained by committing the evaluation of each element of $\hat{z}_l$ to

a different thread. In other words, a grid of threads is created wherein each thread is in charge of calculating the $2K+1$-term summation in Equation (14) for a fixed l, according to the pseudocode 1. This favours the efficiency of the global memory accesses since collecting the $\hat{\Phi}(p)$'s can occur in a coalesced way [22], whereas collecting the $U_{p+\mu_l}$'s can either occur in a coalesced way, except for those threads for which the index $p+\mu_l$ is close to cN. The partial results of the summation are stored on the shared memory to improve the computational efficiency, as it emerges from pseudocode 1. The proposed solution is more efficient as compared to an alternative approach in which each datum U_m is "spread" on all the $\hat{z}_l$'s to which it contributes [30]. Indeed, by doing so, due to the mapping $m = p+\mu_l$, memory write access conflicts should be dealt with by resorting to atomic operations which would significantly slow down the execution [32].

4. SOFTWARE AND HARDWARE PERFORMANCE TESTING PLATFORMS

4.1. Code Implementations and Computing Hardware

Four different versions of the code have been implemented: A strictly

Algorithm 1 Parallel cyclic convolution

```
__global__ void Convolution(cufftComplex *Um,
      cufftComplex *PhiHat, float *Mu,
      cufftComplex *Result, int N, int M)
int i = threadIdx.x + blockDim.x * blockIdx.x;
int x = threadIdx.x; int m;

__shared__ cufftComplex temp[BLOCK];
__shared__ int PP[BLOCK];

temp[x] = make_cuFloatComplex(0,0);

if i < M then
   for 0 ≤ m < 2K + 1 do
      PP[x] = mod((((int)mu[i]) + m - K + cN),(cN));
      temp[x] = cuCaddf(temp[x],
      cuCmulf(PhiHat[i+M*m],Um[PP[x]]));
   end for
        Result[i] = temp[x];
end if
```

sequential one, one exploiting the multi-core parallelism of CPUs and two exploiting the parallelism of GPUs.

The sequential version has been written in ANSI C, the multi-core version in a Matlab 2010 script by exploiting multi-core aware functions, while for the GPU parallel versions, the CUDA language has been employed by exploiting both, single precision and double precision arithmetics.

The CUDA functions have been compiled by nvcc and transformed in C++ code by the command

system(sprintf('nvcc -I"%s/extern/include" –cuda "mexfun.cu" –output-file "mexfun.cpp"', matlabroot));

and then linked to Matlab by

mex -I/opt/cuda/include -L/opt/cuda/lib -lcudart mexfun.cpp

as suggested in [42].

The processing has been performed on a Genesis Tesla I-7950 workstation, with a 8-core Intel CPU i7-950, working at 3.06 GHz and with 6 Gb of RAM. The workstation is equipped with an NVIDIA Tesla C2050, benefitting of the state-of-the-art Fermi GPU architecture and consisting of 14 streaming multiprocessors (SMs), each containing 32 streaming processors (SPs), or processor cores, running at 1.15 GHz. The C2050 is further equipped with a 2.8 GB, off-chip, global memory and supports double precision arithmetics.

4.2. Experimental Setup

The data used in the following Section for assessing the imaging capabilities of the algorithm have been collected in the anechoic chamber at the Laboratory of μwaves and Millimeter Waves at the Università di Napoli Federico II, Dipartimento di Ingegneria Biomedica, Elettronica e delle Telecomunicazioni (DIBET) [26, 27]. The chamber is equipped with a planar scanner, made up of two orthogonal linear positioners, allowing a maximum scanning area of 180 cm $\times$ 180 cm, and driven by an external controller, MI-4190. Obviously, for the measurements and due to the considered geometry, only the horizontal (x-axis) positioner has been exploited.

Copolarized (VV) field data in the 8–12 GHz band have been acquired by a single Tx/Rx horn, with the Tx and Rx channels separated by a directional coupler and connected to a Vector Network Analyzer Ansirsu 37397 working in the 40 MHz–65 GHz frequency band

and operating in the stepped-frequency mode. The total number of collected frequency samples has been 801, and the parameters of the measurement configuration have been $a = b = 0.6\,\mathrm{m}$, $r_0 = 3.46\,\mathrm{m}$ and $x_M = 0.9\,\mathrm{m}$.

As Tx/Rx antenna, a Narda 640 standard gain pyramidal horn has been used. The angular behavior of its gain has been neglected, while its frequency behavior compensated by a calibration procedure (see also [27]).

5. DISCUSSING THE PERFORMANCE OF THE APPROACH

5.1. Computational Performance Analysis

Let us begin by highlighting the advantages arising from the use of programming the NUFFT routine on a massively parallel GPU as compared to a single- or multi-core implementation on a CPU. More in detail, Figure 2 shows the speedup of the CUDA single precision NUFFT implementation (which will be used for the experimental reconstructions in the next Section) over the single-core (ANSI C) and multi-core (Matlab) versions. As it can be seen, the speedup is about 20

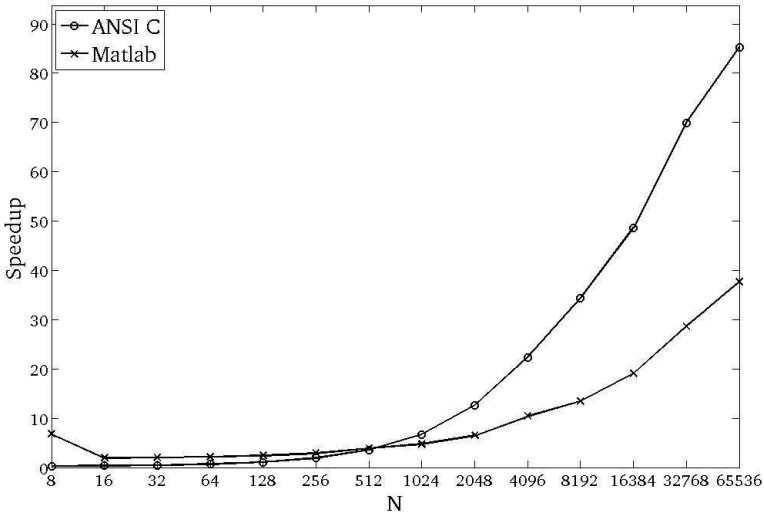

Figure 2. Speedup of the CUDA single precision implementation over the ANSI C and Matlab ones.

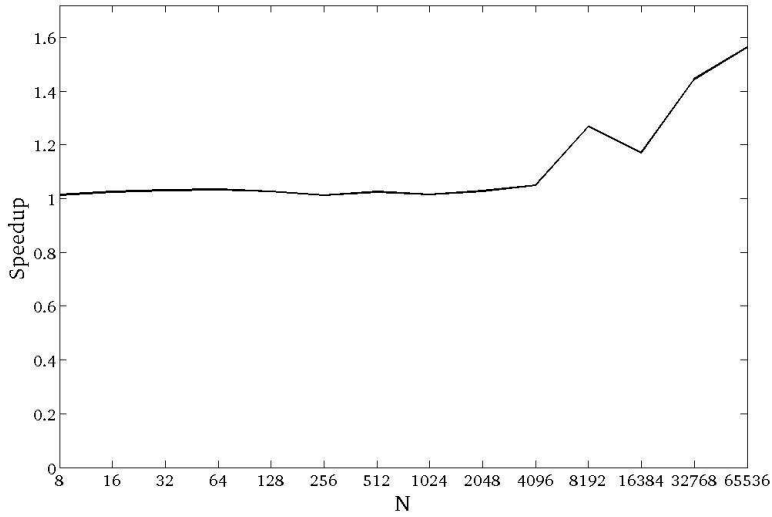

Figure 3. Speedup of the CUDA single precision implementation over the double precision one.

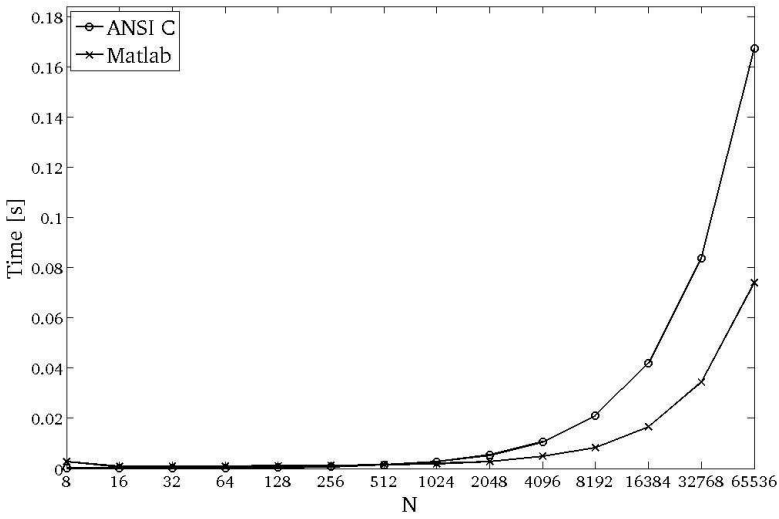

Figure 4. Processing times for the Matlab and ANSI C implementations.

or 40 for the multi-core and single-core implementations, respectively, for a medium size input vector to be transformed. Another point worth analyzing regards comparing the achievable GPU performance under single and double precision arithmetics. To this end, Figure 3 displays the speedup of the CUDA single precision NUFFT implementation over the double precision one. As it can be seen, although interpreted programming languages (as Matlab) are generally speaking more inefficient than compiled ones (as C) [43], the Matlab version is faster thanks to the capability of exploiting the multi-core structure of the CPU. It should be also noticed that, in applications of vegetation imaging, single precision arithmetics could be deemed as sufficient to achieve qualitative reconstructions, so to obtain a further speedup against the use of double precision [44]. At the same time, it should be also noticed that the use of double precision arithmetics anyway is not critical in terms of problem conditioning since the ω-k approach is an intrinsically regularized reconstruction scheme.

Let us now turn the attention to the possibility of performing real-time reconstructions by the developed algorithms. Figure 4 shows the processing time of each individual NUFFT call against the number of elements of the vector to be transformed for the Matlab and ANSI C versions. As it can be seen, in either cases, also when the number

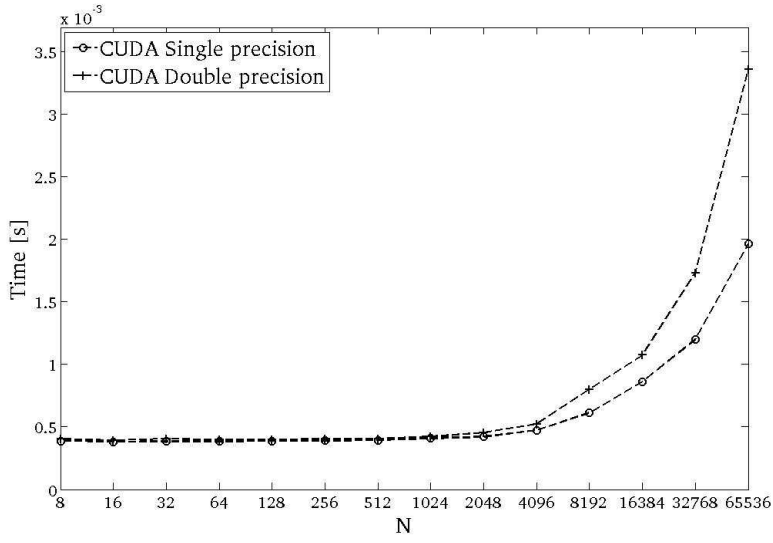

Figure 5. Processing times for the single and double precision CUDA implementations.

of elements to be transformed is large, the processing time of each NUFFT is of the order of fractions of seconds, thus being already useful for real-time imaging. Figure 5 depicts also the processing time concerning the CUDA single and double precision implementations. As it can be seen, the processing time of each individual NUFFT is of the order of milliseconds, thus being very well suited to real-time processing.

Finally, we mention that a favorable comparison of the parallel, NUFFT-based interpolation scheme herewith employed and others (linear, cubic, sinc, etc.), typically adopted in SAR applications, has been reported in [27].

5.2. Experimental Results

We now provide experimental results to point out the performance of the developed ω-k algorithm and we pursue this task with reference to vegetation imaging. This is of interest in a number of applications, including ground-based imaging of natural targets [45].

A first experiment has been carried out over a metallic cylinder with circular cross section, radius 0.2 m and height 1.8 m and with axis orthogonal to the (x, r) plane. Figure 6 illustrates the result achieved by the developed algorithm. For the sake of comparison, Figure 7 shows the same reconstruction, but obtained with numerical data. As

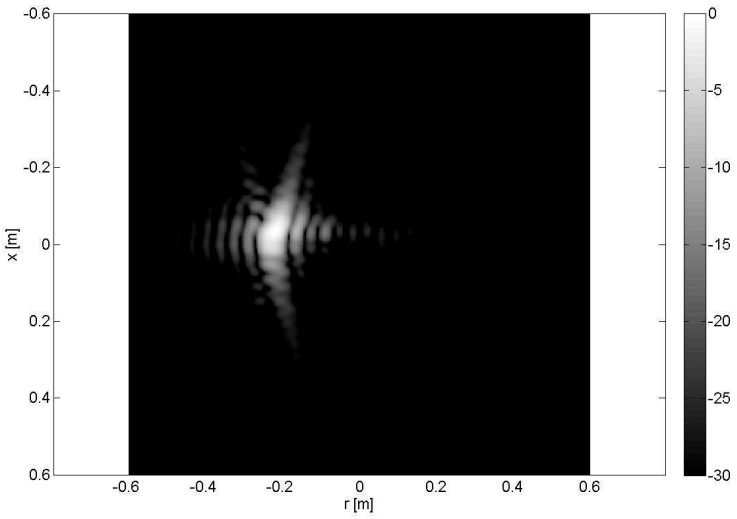

Figure 6. Experimental reconstruction of metallic circular cylinder.

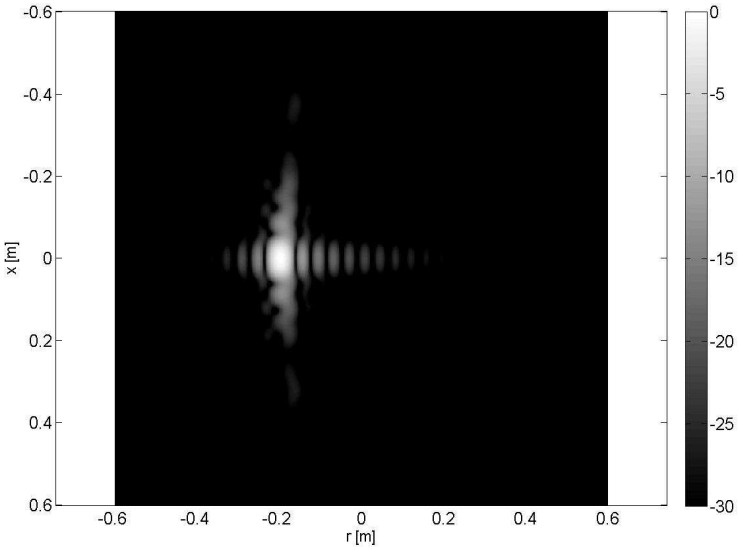

Figure 7. Numerical reconstruction of metallic circular cylinder.

Figure 8. The *Buxus* tree within the anechoic chamber of DIBET.

it can be seen, similar performance are obtained by the algorithm on both, experimental and numerical data.

A second experiment has been performed on a solitary tree, in order to show the performance of the approach for vegetation imaging. The tree is a boxtree (*Buxus*), see Figure 8, having leaves located on opposite sides of the branchelets, rounded to lanceolate, and leathery, about 1.52 cm long and 0.5 cm broad. The tree's cross sectional radius is about 0.4 m. Figure 9 shows the reconstruction of the tree's cross

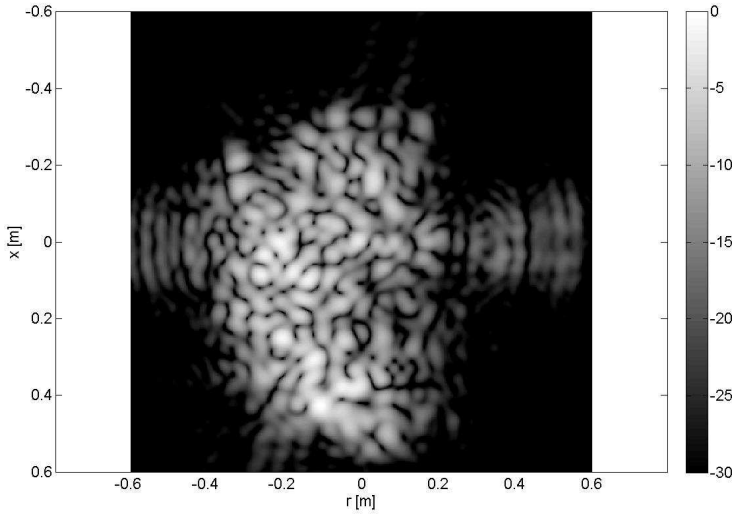

Figure 9. Reconstruction of the boxtree.

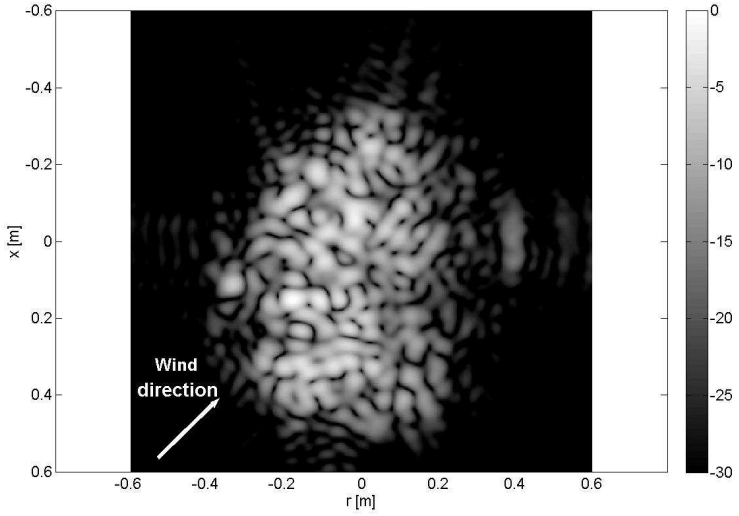

Figure 10. Reconstruction of the boxtree with a source of temporal decorrelation.

section. As it can be seen, the tree's cross section is satisfactorily reconstructed.

Finally, in order to test the algorithm stability against sources of temporal decorrelation, a different data set has been acquired with a fan artificially producing wind with a windspeed of about 2 m/s [46]. As it can be seen from Figure 10, the performance of the algorithm are robust, being the cross sectional shape of the tree still visible, although some sidelobes appear to be "attenuated" as the scatterers move during the data acquisition.

6. CONCLUSION

We have presented an ω-k approach based on the use of a 1D NER NUFFT routine programmed on a GPU in CUDA language, amenable to real-time applications.

Four different versions of the code have been implemented and compared: A strictly sequential one written in ANSI C, one exploiting the multi-core parallelism of CPUs written in Matlab 2010 script employing multi-core aware functions and two exploiting the parallelism of GPUs written in CUDA language (single and double precision).

The Matlab version has proven faster than the ANSI C version. In both cases (Matlab and ANSI C), the processing time for each NUFFT routine call has been of the order of fractions of seconds.

On the other side, the processing time of each individual CUDA NUFFT has been of the order of milliseconds. The overall speedup has been about 20 or 40 for the multi-core and single-core implementations, respectively, for a medium size input vector to be transformed.

The performance of the approach has been pointed out in the applicative case of vegetation imaging against experimental data of a boxtree (*Buxus* tree). The robustness of the algorithm performance under a source of temporal decorrelation (wind) has been also shown.

REFERENCES

1. Stolt, R., "Migration by Fourier transform techniques," *Geophys.*, Vol. 43, No. 1, 49–76, 1978.
2. Cafforio, C., C. Prati, and F. Rocca, "SAR data focusing using seismic migration techniques," *IEEE Trans. Aerosp. Electron. Syst.*, Vol. 27, No. 2, 194–207, Mar. 1991.
3. Reigber, A., A. Alivizatos, A. Potsis, and A. Moreira, "Extended wavenumber-domain synthetic aperture radar focusing with

integrated motion compensation," *IEE Proc. — Radar Sonar Navig.*, Vol. 153, No. 3, 301–310, Jun. 2006.

4. Shin, H.-S. and J.-T. Lim, "Omega-k algorithm for airborne forward-looking bistatic spotlight SAR imaging," *IEEE Trans. Geosci. on Remote Sens. Lett.*, Vol. 6, No. 2, 312–316, Apr. 2009.
5. Hamasaki, T., L. Ferro-Famil, E. Pottier, and M. Sato, "Applications of polarimetric interferometric ground-based SAR (GB-SAR) system to environment monitoring and disaster prevention," *Proc. of the Europ. Radar Conf.*, 29–32, Paris, France, Oct. 6–7, 2005.
6. Sun, B., J. Chen, C.-S. Li, and Y.-Q. Zhou, "FA-ScanSAR: Full aperture scanning pulse by pulse for the nearspace slow-moving platform borne SAR," *Progress In Electromagnetics Research B*, Vol. 25, 23–37, 2010.
7. Chen, H., R. Wu, J. Liu, and Z. Han, "GPR migration imaging algorithm based on NUFFT," *PIERS Online*, Vol. 6, No. 1, 16–20, 2010.
8. Song, J., Q. H. Liu, P. Torrione, and L. Collins, "Two-dimensional and three-dimensional NUFFT migration method for landmine detection using ground-penetrating radar," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 44, No. 6, 1462–1469, 2006.
9. Bamler, R., "A comparison of range-doppler and wavenumber domain SAR focusing algorithms," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 30, No. 4, 706–713, Jul. 1992.
10. Raney, R. K., H. Runge, R. Bamler, I. G. Cumming, and F. H. Wong, "Precision SAR processing using chirp scaling," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 32, No. 4, 786–799, Jul. 1994.
11. Hanssen, R. and R. Bamler, "Evaluation of interpolation kernels for SAR interferometry," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 37, No. 1, 318–321, Jan. 1999.
12. Li, A., "Algorithms for the implementation of Stolt interpolation in SAR processing," *Proc. of the IEEE Geosci. Remote Sens. Int. Symp.*, 360–362, Houston, TX, May 26–29, 1992.
13. Fourmont, K., "Non-equispaced fast Fourier transforms with applications to tomography," *J. Fourier Anal. Appl.*, Vol. 9, No. 5, 431–450, 2003.
14. Greengard, L. and J.-Y. Lee, "Accelerating the nonuniform fast Fourier transform," *SIAM Review*, Vol. 46, No. 3, 443–454, 2004.
15. Subiza, B., E. Gimeno-Nieves, J. M. Lopez-Sanchez, and J. Fortuny-Guasch, "An approach to SAR imaging by means of

non-uniform FFT's," *Proc. of the IEEE Geosci. Remote Sens. Int. Symp.*, 4089–4091, Toulouse, France, Jul. 21–25, 2003.

16. Li, S., H. Sun, B. Zhu, and R. Liu, "Two-dimensional NUFFT-based algorithm for fast near-field imaging," *IEEE Antennas Wireless Prop. Lett.*, Vol. 9, 814–817, 2010.
17. Huang, Y., Y. Liu, Q. H. Liu, and J. Zhang, "Improved 3-D GPR detection by NUFFT combined with MPD method," *Progress In Electromagnetics Research*, Vol. 103, 185–199, 2010.
18. Callison, R. J., "Spotlight Synthetic Aperture Radar (SAR) system and method for generating a SAR map in real-time using a modified polar format algorithm," US Patent, No. 7,511,656 B2, Mar. 31, 2009.
19. Di Bisceglie, M., M. Di Santo, C. Galdi, R. Lanari, and N. Ranaldo, "Synthetic aperture radar processing with GPGPU," *IEEE Signal Proc. Mag.*, Vol. 27, No. 2, 69–78, Mar. 2010.
20. Sharma, G. and J. Martin, "MATLAB®: A language for parallel computing," *Int. J. Parallel Prog.*, Vol. 37, No. 1, 3–36, 2009.
21. Rosario-Torres, S. and M. Vélez-Reyes, "Speeding up the MATLAB™ hyperspectral image analysis toolbox using GPUs and the Jacket toolbox," *Proc. of the Hyperspectral Image and Signal Proc. Workshop: Evolution in Remote Sens.*, 1–4, Grenoble, France, Aug. 26–28, 2009.
22. Kirk, D. B. and W. W. Hwu, *Programming Massively Parallel Processors*, Morgan Kaufmann, Burlington, MA, 2010.
23. Jiang, W.-Q., M. Zhang, and Y. Wang, "CUDA-based radiative transfer method with application to the EM scattering from a two-layer canopy model," *Journal of Electromagnetic Waves and Applications*, Vol. 24, Nos. 17–18, 2509–2521, 2010.
24. Jiang, W.-Q., M. Zhang, H. Chen, and Y.-G. Lu, "CUDA implementation in the EM scattering of a three-layer canopy," *Progress In Electromagnetics Research*, Vol. 116, 457–473, 2011.
25. Pryor, G., B. Lucey, P. Yalamanchili, C. McClanahan, and J. Malcolm, "High-level GPU computing with jacket: For MATLAB and C/C++," *Proc. of the SPIE Vol. 8060 Modeling and Simulation for Defense Systems and Applications VI*, Orlando, FL, USA, Apr. 26–27, 2011.
26. Zhou, Y., "Microwave imaging based on wideband range profiles," *Progress In Electromagnetics Research Letters*, Vol. 19, 57–65, 2010.
27. Capozzoli, A., C. Curcio, A. Di Vico, and A. Liseno, "NUFFT- & GPU-based fast imaging of vegetation," *IEICE Trans. on*

Commun., Vol. E94-B, No. 7, 2092–2103, Jul. 2011.

28. Zhang, Y., J. Liu, E. Kultursay, M. Kandemir, N. Pitsianis, and X. Sun, "Scalable parallelization strategies to accelerate NuFFT data translation on multicores," *Proc. of the Int. Euro-Par Conf., Part II*, 125–136, Ischia, Italy, Aug. 31–Sep. 3, 2010.
29. Gregerson, A., "Implementing fast MRI gridding on GPUs via CUDA," *NVIDIA Tech. Rep. on Med. Imag. Using CUDA*, 2008.
30. Sørensen, T. S., T. Schaeffter, K. Østergaard Noe, and M. Schacht Hansen, "Accelerating the nonequispaced fast Fourier transform on commodity graphics hardware," *IEEE Trans. Med. Imag.*, Vol. 27, No. 4, 538–547, Apr. 2008.
31. Jacob, M., "Optimized least-square nonuniform fast Fourier transform," *IEEE Trans. Signal Proc.*, Vol. 57, No. 6, 2165–2177, Jun. 2009.
32. Capozzoli, A., C. Curcio, G. D'Elia, A. Liseno, and P. Vinetti, "Fast CPU/GPU pattern evaluation of irregular arrays," *Applied Comput. Electromagn. Soc. J.*, Vol. 25, No. 4, 355–372, Apr. 2010.
33. Zhang, K. and J. U. Kang, "Graphics processing unit accelerated non-uniform fast Fourier transform for ultrahigh-speed, real-time Fourier-domain OCT," *Optics Express*, Vol. 18, No. 22, 23472–23487, Oct. 2010.
34. Kestur, S., S. Park, K. M. Irick, and V. Narayanan, "Accelerating the nonuniform fast fourier transform using FPGAs," *Proc. of the IEEE Annual Int. Symp. on Field-Programmable Custom Comput. Machines*, 19–26, Charlotte, NC, May 2–4, 2010.
35. Fatica, M. and W.-K. Jeong, "Accelerating Matlab with CUDA," *Proc. of the High Performance Embedded Comput. Workshop*, Lexington, MA, Sep. 18–20, 2007.
36. Capozzoli, A., C. Curcio, A. Liseno, M. Migliorelli, and G. Toso, "Accelarating phase-only reflectarray antenna synthesis by GPUs," *Proc. of the Int. Rev. of Progr. in Appl. Comput. Electromagn.*, Williamsburg, VI, Mar. 27–31, 2011, CD ROM.
37. Bamler, R., "A comparison of range-doppler and wavenumber domain SAR focusing algorithms," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 30, No. 4, 706–713, Jul. 1992.
38. Chommeloux, L., C. Pichot, and J.-C. Bolomey, "Electromagnetic modeling for microwave imaging of cylindrical buried inhomogeneities," *IEEE Trans. Microw. Theory Tech.*, Vol. 34, No. 10, 1064–1076, Oct. 1986.
39. Rabiner, L. R., R. W. Schafer, and C. M. Rader, "The chirp z-transform algorithm and its application," *Bell Syst. Tech. J.*,

Vol. 48, No. 5, 1249–1292, May–Jun. 1969.

40. Lanari, R., "A new method for the compensation of the SAR range cell migration based on the chirp z-transform," *IEEE Trans. on Geosci. Remote Sens.*, Vol. 33, No. 5, 1296–1299, Sep. 1995.
41. CUDA cuFFT Library, Aug. 2010.
42. http://www.mathworks.com/matlabcentral/newsreader/view -_thread/261866.
43. Kepner, J., M. Gokhale, R. Minnich, A. Marks, and J. DeGood, "Interfacing interpreted and compiled languages to support applications on a massively parallel network of workstations (MP-NOW)," *Cluster Computing*, Vol. 3, No. 1, 35–44, 2000.
44. Huang, B., J. Mielikainen, H. Oh, and H.-L. A. Huang, "Development of a GPU-based high-performance radiative transfer model for the Infrared Atmospheric Sounding Interferometer (IASI)," *J. Comput. Phys.*, Vol. 230, No. 6, 2207–2221, Mar. 2011.
45. Lim, K.-S. and V. C. Koo, "Design and construction of wideband VNA ground-based radar system with real and synthetic aperture measurement capabilities," *Progress In Electromagnetics Research*, Vol. 86, 259–275, 2008.
46. Narayanan, R. M., D. W. Doerr, and D. C. Rundquist, "Temporal decorrelation of x-band backscatter from wind-influenced vegetation," *IEEE Trans. Aerosp. Electron. Syst.*, Vol. 28, No. 2, 404–412, Apr. 1992.