RSFT: A Realistic High Dimensional Sparse Fourier Transform and Its Application in Radar Signal Processing

Shaogang Wang, Vishal M. Patel and Athina Petropulu
Department of Electrical and Computer Engineering
Rutgers University
94 Brett Road, Piscataway, NJ 08854
Email: {shaogang.wang, vishal.m.patel, athinap}@rutgers.edu

***Abstract*—We propose a realistic high dimensional sparse Fourier transform (RSFT) algorithm, which detects frequencies in multidimensional data, provided that the data is sparse in the frequency domain. Although sparsity has been exploited before to reduce the complexity of the Discrete Fourier Transform, unlike previous approaches, the RSFT allows for off-grid frequencies. We provide a concrete application example on short range ubiquitous radar signal processing, and verify the feasibility of the RSFT in that scenario via simulations.**

***Index Terms*—Array signal processing, sparse Fourier transform, radar signal processing.**

I. Introduction

Many practical applications in radar, communications and imaging require one to take the Discrete Fourier Transform (DFT) of high-dimensional signals in order to identify frequencies in the data. The DFT is usually implemented via the Fast Fourier Transform (FFT), whose computational complexity is $\mathcal{O}(N \log N)$ for N data points. Recently, by leveraging the sparsity of signals in the frequency domain, the Sparse Fourier Transform (SFT) [1], [2] can further reduce the complexity required to identify the underlying frequencies. Different versions of the SFT related techniques have been successfully applied in several practical applications, such as a fast Global Positioning System (GPS) receiver, wide-band spectrum sensing, light field reconstruction, etc. [3]–[5].

High order extension of the SFT has also been considered. Andre *et al.* [6] extended the exactly-K-sparse SFT algorithm from [1] into two dimensions. Ghazi *et al.* [7] proposed a sample optimal 2-D SFT both for exactly sparse and approximately sparse signals. Ong *et al.* [8] proposed a 2-D SFT algorithm based on sparse-graph decoding. An extension to an arbitrary constant dimension is reported in [9]. However, all the aforementioned algorithms rely on a grid, and assume that the signal frequencies are on the grid. In practice, however, the signal frequencies lie in the continuous space of $[0, 2\pi)$, and are usually off-grid. The consequence of off-grid frequencies is leakage to other frequency bins, which essentially destroys the sparsity of the signal. To refine the estimation of off-grid frequencies, in [5], Shi *et al.* proposed a gradient descent-based method to find off-grid frequencies from the initial SFT estimates. However, the computation of gradient descent is not efficient, due to the unknown gradient of the signal in the frequency domain. An SFT algorithm for off-grid frequencies was proposed by Boufounos *et al.* in [10]. The underlying assumption in [10] is that signal and noise are well separated by predefined gaps in the frequency domain. However, this assumption does not hold for many practical signal processing applications.

In this paper, we propose a new algorithm called Realistic Sparse Fourier Transform (RSFT), which does not requires the frequencies to be on-grid and does not rely on the restrictive assumption that signal and noise are well separated by predefined gaps in the Fourier domain. Furthermore, we extend the proposed algorithm to arbitrary fixed high dimensions so that it can be used to replace the N-dimensional FFT (N-D FFT) in a sparse setting. To the best of our knowledge, the RSFT algorithm is the first SFT algorithm, which addresses the issue of off-grid frequencies for arbitrary high dimensional signals. Finally, we present an application of the RSFT algorithm in multi-dimensional radar signal processing, in which a 3-D RSFT is applied on short range ubiquitous radar [11] (SRUR) to detect targets and estimate their range, velocity and direction of arrival (DOA). Due to the computational efficiency of RSFT, a faster reaction or lower cost of hardware for this kind of radar is expected.

Notation: We use lower-case and upper-case bold letters to denote vectors and matrices, respectively. $[S]$ refers to the set of indices $\{0, \cdots, S-1\}$. The DFT of signal $\mathbf{s}$ is denoted as $\hat{\mathbf{s}}$.

This paper is organized as follows. A brief background on the SFT algorithm is given in Section II. Details of the proposed RSFT algorithm are given in Section III. An application of the RSFT algorithm in radar signal processing is presented in Section IV. Section V concludes the paper with a brief summary and discussion.

II. Background

As opposed to the FFT that computes the coefficients of all N frequency components of a N-samples long signal, the

SFT [2] computes only the K frequency components of a K-sparse signals. At a high level, the SFT consists of two kinds of loops, i.e., the *Location* loop and the *Estimation* loop. The former finds the indices of the K most significant frequencies in the input signal, while the latter estimates the corresponding Fourier coefficients. Here we emphasize on *Location* more than *Estimation*, since the former is more relevant to the radar application that we consider. The *Location* step provides frequency locations, which in the radar case is directly related to target parameters.

In the *Location* loop, a *permutation* procedure reorders the input data in the time domain, causing the frequencies to also reorder. The permutation causes closely spaced frequencies to appear in well separated locations with high probability. Mathematically, the permutation is defined as

$$(P_{\sigma,\tau}x)_i = x_{\sigma i+\tau}, \tag{1}$$

where x_i is the i_{th} entry of input signal $\mathbf{x} \in \mathbb{C}^N$, $\sigma, \tau \in [N]$, and σ is invertible mod N, i.e., there exists a σ^{-1} satisfying $\sigma\sigma^{-1} \equiv 1 \ (mod\ N)$. Consequently, the frequency is dilated modularly by σ times, and an additional phased rotation by τ is introduced, i.e., $\widehat{(P_{\sigma,\tau}x)}_{\sigma i} = \hat{x}_i e^{-j\tau\frac{2\pi}{N}}$. It is also assumed that the data length for each dimension is a power of 2. Then, a flat-window [2] is applied on the permuted signal for the purpose of extending a single frequency into a (nearly) boxcar, for a reason that will become apparent in the following. The windowed data are aliased, by creating a periodic extension of the data with period B with $B << N$, and B a power of 2. The frequency domain equivalent of this aliasing is undersampling by N/B. The window used at the previous step ensures that no peaks are lost due to the effective undersampling in the frequency domain. After this stage, a FFT of length B of one period is employed.

The first detection stage finds the significant frequencies' peaks and their indices are reverse mapped into the original frequency space. However, the reverse mapping yields not only the true location of the signal frequency, but also N/B ambiguous locations for each significant frequency. To remove the ambiguity, multiple iterations of *Location* with randomized permutation are performed. Finally, the second stage detection locates the K most significant frequencies from the accumulated data for each iteration.

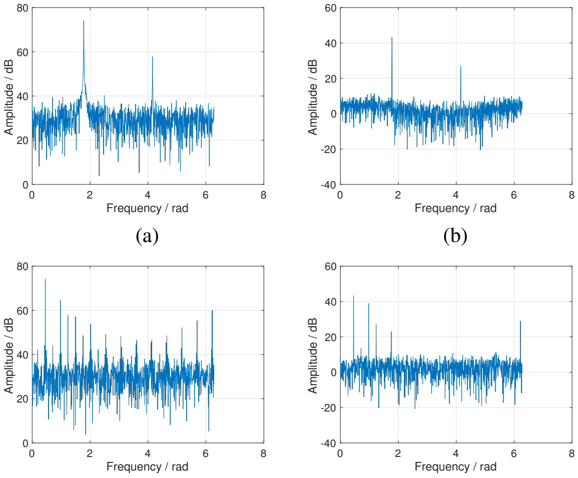
III. The RSFT Algorithm

In this section, we introduce the proposed RSFT algorithm, which is basically an SFT that is robust to off-grid frequencies and present its extensions to high dimensional problems.

A. Leakage Suppression for Off-grid Frequencies

The SFT algorithm only holds for the discrete on-grid frequencies. In real world applications, the frequencies are continuous and can take any value in $[0, 2\pi)$. When fitting a grid on these frequencies, leakage occurs from off-grid frequencies, which can jeopardize the natural sparsity of the signal. As a result, it is difficult to determine the frequency domain peaks after permutation, since the leakage of a strong frequency component would easily mask the main lobe of a weak frequency component (See Fig. 1 (c)). To address this problem, we multiply the received time domain signal with a window before permutation, and call this procedure *Pre-permutation Windowing*. The idea is to confine the leakage within a finite number of frequency bins, as illustrated in Fig. 1.

The choice of the pre-permutation window is determined by dynamic range, frequency resolution and computational complexity requirements. More specifically, the dynamic range specification determines the attenuation of the side-lobes, and the side-lobe level should be lower than the noise level after windowing. However, the larger attenuation of the side-lobes, the more wide would be the main-lobe, leading to a worsen resolution in frequency domain. Meanwhile, a more broaden main-lobe will cause greater computational overhead, which will be discussed in Section III-D.



(a) (b) (c) (d)

Fig. 1. **Effect of pre-permutation Windowing.** The signal contains two significant frequency components, one of which is $15dB$ stronger than the other. A Dolph-Chebyshev window is applied to the time-domain signal. Windowed signal after permutation is more sparse in the frequency domain as compared to the permuted unwindowed signal. (a) Spectrum of unwindowed signal. (b) Spectrum of windowed signal. (c) Spectrum of unwindowed signal after permutation. (d) Spectrum of windowed signal after permutation.

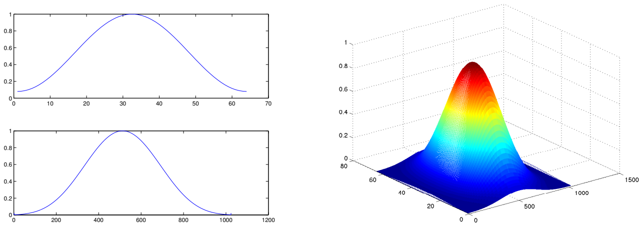
B. High Dimensional Extensions

Due to the separability of the DFT, one could easily extend the FFT to high dimensions by simply applying 1-D FFT on each dimension of the data sequentially. For the SFT algorithm, however, the extension is not obvious. In what follows, we elaborate on the high dimensional extension for its main stages.

1) Windowing: In the pre-permutation windowing and the flat-windowing stages, the window for each dimension is designed separately. After that, the high dimension widow is generated by combining each 1-D window. For instance, in the 2-D case, assuming that $\mathbf{w}_x$ and $\mathbf{w}_y$ are the two windows in the x and y dimension, respectively, a 2D window can be computed as

$$\mathbf{W}_{xy} = \mathbf{w}_x \mathbf{w}_y^H, \tag{2}$$

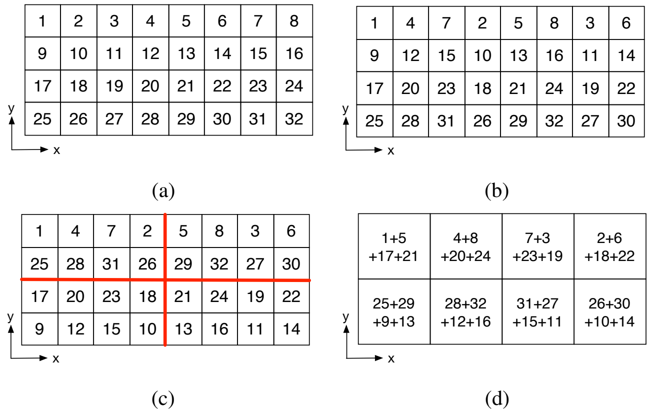
where $(\cdot)^H$ denotes for the conjugate transpose. Fig. 2 shows a compound 2-D window which is a combination of a hamming window and a Dolph-Chebyshev window. We apply those windows on the data by point-wise multiplications.



(a) (b)

Fig. 2. **Compound Window in 2-D.** (a) Top: a 64-points Hamming window; bottom: a 1024-points Dolph-Chebyshev window. (b) The 2D window.

2) *Permutation:* The permutation parameters are generated for each dimension in a random way according to (1). Then, we carry the permutation on each dimension *sequentially*. An example for the 2-D case is illustrated in Fig. 3.



(a)

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32

(b)

1	4	7	2	5	8	3	6
9	12	15	10	13	16	11	14
17	20	23	18	21	24	19	22
25	28	31	26	29	32	27	30

(c)

1	4	7	2	5	8	3	6
25	28	31	26	29	32	27	30
17	20	23	18	21	24	19	22
9	12	15	10	13	16	11	14

(d)

1+5 +17+21	4+8 +20+24	7+3 +23+19	2+6 +18+22
25+29 +9+13	28+32 +12+16	31+27 +15+11	26+30 +10+14

Fig. 3. **Permutation and Aliasing in 2D.** (a) Original 2D data forms a 4×8 matrix. (b) Permutation in $x-$dimension, $\sigma_x = 3, \tau_x = 0$. (c) Permutation in $y-$dimension, $\sigma_y = 3, \tau_y = 0$. After permutation, data is divided into four 2×4 sub-matrices. (d) Aliasing by adding sub-matrices from (c).

3) *Aliasing:* The aliasing stage compresses the high dimensional data into much smaller size. In 2-D, as shown in Fig. 3 (d), a periodic extension of the $N_x \times N_y$ data matrix is created with period B_x in the x dimension and B_y in the y dimension, with $B_x << N_x$ and $B_y << N_y$, and the basic period is extracted.

4) *First stage detection and reverse mapping:* We carry first stage detection after taking the magnitude of N-D FFT on the aliased data. Since the size of aliased data is much smaller than original size, the saving of the computation is remarkable. After that, we find the $dK, d > 1$ highest peaks and then reverse map their indices back to the original space. The combination of the reverse mapped indices from each dimension provides the tentative locations of the original frequency components. Assuming the side-lobes are below the noise level after pre-permutation windowing, empirically, we can choose d as

$$d = \text{round}(\prod_{i \in [U]} d_i), \tag{3}$$

where U is the number of dimensions, $\text{round}(\cdot)$ denotes for rounding to the nearest integer, and d_i is the 6.0-dB bandwidth of the pre-permutation window for the i_{th} dimension. For instance, according to [12], a Hamming window has its 6.0-dB bandwidth approximately as 1.81. As a result, a 2-D Hamming window gives $d = 3$.

5) *Accumulation and second stage detection:* The accumulation stage collects the tentative frequency locations found in the reverse mapping for each iteration, and the number of occurrences for each location is calculated after running over T iterations. The second stage detection finds K peaks in the data with the highest number of occurrences.

C. The RSFT Algorithm

Based on the discussion above, we summarize the RSFT method in Algorithm 1. We set $\tau = 0$ in each permutation, since the random phase rotation does not affect the performance of a detector after taking magnitude of the signal in the intermediate stage.

Algorithm 1 RSFT algorithm

```
Input: complex signal $\mathbf{r}$ in any fixed high dimension
Output: $\mathbf{o}$, sparse frequency locations of input signal
procedure RSFT($\mathbf{r}$)
    Pre-Permutation Windowing: $\mathbf{y} \leftarrow \mathbf{W}\mathbf{r}$
    Generate a set of $\sigma$ randomly for each dimension
    $\mathbf{x} \leftarrow 0$
    for $i \leftarrow 0$ to $T$ do
        Permutation: $\mathbf{p} \leftarrow P_\sigma \mathbf{y}$
        Flat-windowing: $\mathbf{z} \leftarrow \mathbf{W}_f \mathbf{p}$
        Aliasing: $\mathbf{a} \leftarrow \text{Aliasing}(\mathbf{z})$
        N-D FFT: $\hat{\mathbf{a}} \leftarrow \text{FFT}(\mathbf{a})$
        First-stage-detection: $\mathbf{c} \leftarrow \text{Det1}(|\hat{\mathbf{a}}|^2)$
        Reverse-mapping: $\mathbf{x}_i \leftarrow \text{Reverse}(\mathbf{c})$
        Accumulation: $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{x}_i$
    end for
    Second-stage-detection: $\mathbf{o} \leftarrow \text{Det2}(\mathbf{x})$
    return $\mathbf{o}$
end procedure
```

D. Computational Complexity

We compute the computational complexity of the RSFT algorithm by counting the number of operations in Algorithm 1, as shown in Table I. The RSFT yields a complexity of

$$\mathcal{O}\left(T(N + B + B\log B + \frac{KdN}{B}) + N\right), \tag{4}$$

while the N-D FFT gives the complexity of $\mathcal{O}(N \log N)$. Here N, B denote for the total number of data points in the original and shrunken high dimensional dataset, respectively. From Fig. 4, one can see that the complexity ratio of FFT over RSFT rises almost linearly versus N, which grows exponentially. For $N = 2^{50}, B = 2^{12}, T = 5, K = 100, d = 2$, the RSFT algorithm is approximately 8 times more efficient than the FFT. Note that the core operation in RSFT is still FFT but on a reduced dimensional space. By leveraging the existing high performance FFT libraries such as FFTW [13], the implementation of the RSFT algorithm could be further improved.

As discussed in Section III-A, the choice of pre-permutation window is a compromise between the resolution and dynamic range specifications. Now, from Eq. (3) and (4), we can see the pre-permutation window also affects the complexity of the RSFT, i.e., a window with larger d will demand more computation, as shown in Fig. 4.

The complexity of RSFT is also influenced by B. With other parameters fixed, we can solve the optimal B, which minimizes f in Eq. (4). In the high dimensional setting, B is the multiplication of the shrunken data length in each dimension, i.e., $B = \prod_{i \in [U]} B_i$, with B_i a power of 2. And in order to hash each significant frequency into a distinct location of the shrunken space with high probability, we make $B >> dK$.

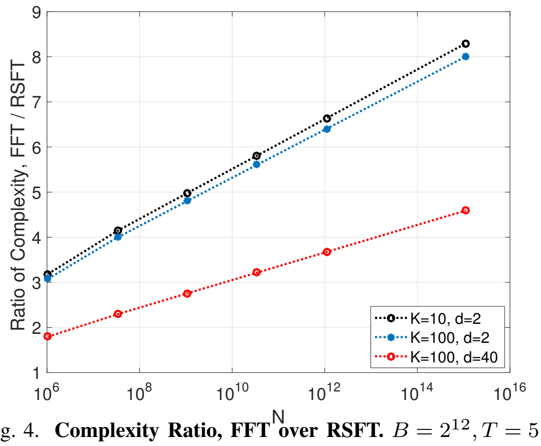


Fig. 4. **Complexity Ratio, FFT over RSFT.** $B = 2^{12}, T = 5$.

IV. RSFT for Ubiquitous Radar Signal Processing

The complexity analysis above reveals that the RSFT algorithm can greatly reduce the complexity of certain high dimensional problems. This can be signifiant in many applications, since lower complexity means faster reaction time and more economical hardware. However, in order to apply RSFT, the signal to be processed should meet the following requirements:

- It should be sparse in some domain.
- It should be sampled uniformly whether in temporal or spacial domain.
- The SNR should be moderately high so that the algorithm can detect the peaks of significant frequencies reliably.

TABLE I
Computational Complexity of RSFT

Procedure	**Number of Operations**
Pre-Permutation Win	N
Permutation	TN
Flat Win	TN
Aliasing	$TB(N/B-1)$
FFT	$T\frac{B}{2}\log B$
Square	TB
First Stage Detection	TB
Reverse Mapping	$TKdN/B$
Second stage Detection	N
Total Operations	$T(3N + B + \frac{B}{2}\log B + \frac{KdN}{B}) + 2N$
Complexity	$\mathcal{O}\left(T(N + B + B\log B + \frac{KdN}{B}) + N\right)$

While many applications satisfy these requirements, in what follows, we discuss an example in SRUR signal processing.

A. *Short Range Ubiquitous Radar (SRUR)*

An ubiquitous radar [11] or SIMO radar can see targets everywhere at anytime without steering its beams as a traditional phased array radar does. In SRUR, a broad transmitting beam patten is achieved by an omnidirectional transmitter and multiple narrow beams are formed simultaneously after receiving of the reflected signal. The beam pattens of an ubiquitous radar is shown in Fig. 5 with an Uniform Linear Array (ULA) configuration.

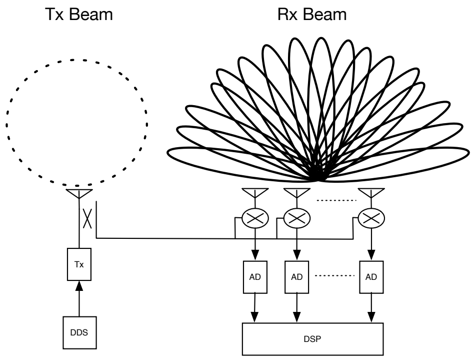


Fig. 5. **Ubiquitous Radar System Structure and Transmitting / Receiving Beam Patten.** A broad beam patten is transmitted with an omnidirectional antenna, while multiple narrow beams are formed simultaneously by the receiving array. Each receiving channel is mixed with a coupled signal from the transmitter to de-chirp the LFMCW signal, before the A/D converting.

An SRUR with range coverage of several kilometers could be important both in military and civilian vehicular applications. For instance, in an active protection system [14], sensors on the protected vehicle have to detect and locate the warheads from a closely fired rocket-propelled grenade (RPG) within milliseconds. Among other sensors, SRUR's simultaneous wide angle coverage, high precision of measurement and all-weather operation make it the ideal sensor for such situation.

In order to achieve high range resolution and cover near range, SRUR utilizes a LFMCW waveform, as shown in Fig.

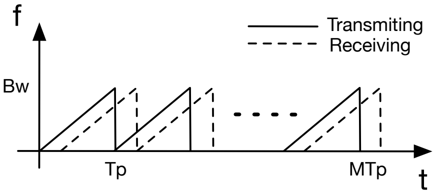


Fig. 6. **LFMCW Waveform.** The signal frequency change linearly in time with a repetition internal T_p. A burst contains M repetitions. The range of frequency changing is the bandwidth of the system. The received signal is a delayed version of the transmitted signal.

6. Mathematically, the transmitted waveform can be expressed as

$$s(t, v) = A\cos(2\pi(f_c(t - vT_p) + \pi\alpha(t - vT_p)^2), \quad (5)$$

where T_p is the repetition interval (RI), $v \in [M]$ denotes the v_{th} RI, A is amplitude of the signal, f_c is the carrier frequency and α is the chirp rate. Furthermore, without loss of generality, we assume that the initial phase of the signal is zero.

Upon reception, a de-chirp process is implemented by mixing the received signal with the transmitted signal, followed by a lowpass filter. The received signal is a delayed version of the transmitted one, hence by mixing the two signals, the range information of the targets is linearly encoded in the difference of the frequencies. Hence for the $i_{th}, i \in [N]$ receiving channel, the de-chirped signal is expressed as

$$\begin{aligned} r_i = &\sum_{k\in[K]} a^{[k]} \cos\left(2\pi((f_r^{[k]} + f_d^{[k]})(t - vT_p) + i\pi\sin\theta^{[k]}\right) \\ &+ n(t), \end{aligned} \quad (6)$$

which is a superposition of K sinusoids and additive noise $n(t)$. For the k_{th} sinusoid, $a^{[k]}$ represents its amplitude and $f_r^{[k]}, f_d^{[k]}$ are the frequency components respect to target's range and velocity respectively, i.e.,

$$f_r^{[k]} = \frac{2\alpha r_t^{[k]}}{c}, \quad f_d^{[k]} = \frac{2v_t^{[k]}}{\lambda}, \quad (7)$$

where $r_t^{[k]}, v_t^{[k]}, c$ are the k_{th} target's range, velocity and speed of wave propagation respectively.

The DOA of the k_{th} target, i.e., $\theta^{[k]}$ is defined as the angle between the line of sight (from the array center to the target) and the array normal. Assuming that the element wise spacing is $\lambda/2$, under the narrowband signal assumption, $\theta^{[k]}$ will cause an increase of phase at the neighboring array element equal to $\pi\sin\theta^{[k]}$. We omit the constant phase term in each sinusoids of Eq. (6), since they are irrelevant to the performance of the algorithm.

After AD conversion of each receiving channel, we can use the processing scheme shown in Fig. 7 to detect the targets as well as estimate their range, velocity and DOA. More specifically, grid-based versions of $f_r^{[k]}, f_d^{[k]}, \pi\sin\theta^{[k]}$ can be calculated by applying a 3-D FFT on the windowed data cube, followed by a detection procedure.

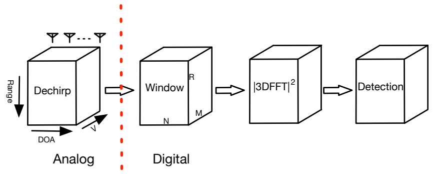


Fig. 7. **Conventional Processing Scheme for SRUR.**

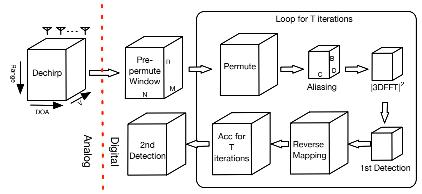


Fig. 8. **RSFT Based Processing Scheme for SRUR.**

B. RSFT-based SRUR Signal Processing

Although the number of samples of SRUR is reduced significantly with the analog de-chirp processing, the realtime processing with 3-D FFT is still challenging. The RSFT algorithm is suitable for reducing the computational complexity of SRUR since, 1) the number of targets is usually much smaller than the number of spatial resolutions cells, which implies that the signal is sparse after proper translation; 2) with an ULA and digitization of each received element, the signal is uniformly sampled both in spatial and temporal domain; and 3) the short range coverage implies that moderate high SNR is easy to achieve.

The RSFT-based SRUR processing architecture is shown in Fig. 8. Compared to the conventional processing, the 3-D FFT is replaced with the looping block, in which the aliasing procedure converts the data cube dimension from $R \times N \times M$ to $B \times C \times D$. The 3-D FFT operated on the smaller data cube could save the computation time significantly.

C. Simulations

In this section, we verify the feasibility of RSFT-based SRUR processing and compare to the SFT-based processing via simulations. The main parameters of the system are listed in Table II. The design of the system can guarantee non-ambiguous measurements of the target's range and velocity, assuming the maximum range and velocity are less than $1.5km$ and $300m/s$, respectively.

We generate a signal from 4 targets according to (6). The parameters of targets can be arbitrarily chosen within the unambiguous space, which implies the corresponding frequency components do not necessarily lie on the grid points. The targets' parameters used in the simulation are listed in Table III.

The SFT from [2] is 1-dimensional. In order to reconstruct targets in the 3-D space, we extend the SFT to high dimension with the techniques described in Section III-B. In the experiment, we choose B, C, D equal to $64, 32, 16$, respectively, and gradually increase the number of counting peaks in the first

TABLE II
SRUR PARAMETERS

Parameter	Symbol	Value
Number of range bins	R	2048
Number of receiving elements	N	64
Number of RI	M	32
Wave length	λ	$0.03m$
Wave propagation speed	c	$3 \times 10^8 m/s$
Bandwidth	B_w	$150MHz$
Repetition interval	T_p	$5 \times 10^{-5} s$
Maxima range	R_{max}	$1.5 \times 10^3 m$
Chirp rate	α	$3 \times 10^{12} Hz/s$
Sampling frequency (IQ)	f_s	$41MHz$

TABLE III
TARGET PARAMETERS

Target	Range (m)	Velocity (m/s)	DOA ($\circ$)	SNR (dB)
1	1000	100	30	0
2	500	50	0	-10
3	350	240	-16	-20
4	350	240	-20	-20

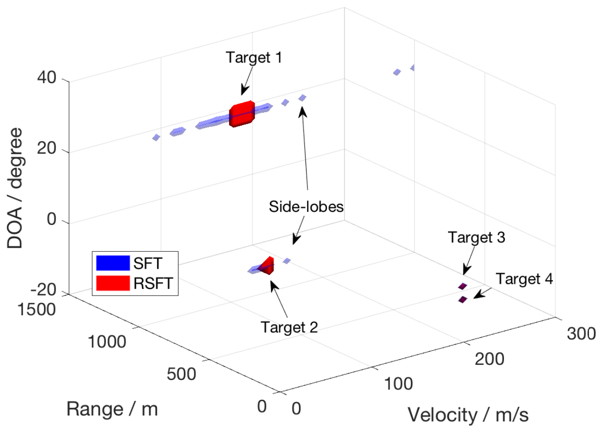


Fig. 9. **Target Reconstruction via 3-D SFT and RSFT.** The SFT-based processing recovers the side-lobes of the stronger targets, while the RSFT-based method only recovers the main-lobes of targets.

stage detection until all the targets are recovered. Fig. 9 shows the targets reconstruction results from both methods. The SFT-based method shows the side-lobes of the stronger targets, while the RSFT-based method only recovers the (extended) main-lobes of all the targets.

V. DISCUSSION AND CONCLUSION

In this paper, we have addressed some practical limitations of the SFT algorithm by developing the RSFT algorithm. It has been shown that the RSFT algorithm is computationally more efficient than N-D FFT in sparse settings and is robust to off-grid frequencies. Furthermore, we have presented an application of our RSFT algorithm in ubiquitous radar signal processing.

Further work is needed to determine the ability of RSFT to detect weak targets, and its behavior when the exact sparsity is not known.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Predrag Spasojevic and Dr. Anand Sarwate from Rutgers university for initial support of this work. The work of SW was jointly supported by China Scholarship Council and Shanghai Institute of Spaceflight Electronics Technology. The work of VMP was partially supported by an ARO grant W911NF-16-1-0126.

REFERENCES

[1] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Nearly optimal sparse fourier transform," in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pp. 563–578, ACM, 2012.

[2] H. Hassanieh, P. Indyk, D. Katabi, and E. Price, "Simple and practical algorithm for sparse fourier transform," in *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pp. 1183–1194, SIAM, 2012.

[3] H. Hassanieh, F. Adib, D. Katabi, and P. Indyk, "Faster gps via the sparse fourier transform," in *Proceedings of the 18th annual international conference on Mobile computing and networking*, pp. 353–364, ACM, 2012.

[4] H. Hassanieh, L. Shi, O. Abari, E. Hamed, and D. Katabi, "Ghz-wide sensing and decoding using the sparse fourier transform," in *INFOCOM, 2014 Proceedings IEEE*, pp. 2256–2264, IEEE, 2014.

[5] L. Shi, H. Hassanieh, A. Davis, D. Katabi, and F. Durand, "Light field reconstruction using sparsity in the continuous fourier domain," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 1, p. 12, 2014.

[6] A. Rauh and G. R. Arce, "Sparse 2d fast fourier transform," *Proceedings of the 10th International Conference on Sampling Theory and Applications*, 2013.

[7] B. Ghazi, H. Hassanieh, P. Indyk, D. Katabi, E. Price, and L. Shi, "Sample-optimal average-case sparse fourier transform in two dimensions," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*, pp. 1258–1265, IEEE, 2013.

[8] F. Ong, S. Pawar, and K. Ramchandran, "Fast and efficient sparse 2d discrete fourier transform using sparse-graph codes," *arXiv preprint arXiv:1509.05849*, 2015.

[9] P. Indyk and M. Kapralov, "Sample-optimal fourier sampling in any constant dimension," in *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pp. 514–523, IEEE, 2014.

[10] P. Boufounos, V. Cevher, A. C. Gilbert, Y. Li, and M. J. Strauss, "What's the frequency, kenneth?: Sublinear fourier sampling off the grid," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 61–72, Springer, 2012.

[11] M. Skolnik, "Systems aspects of digital beam forming ubiquitous radar," tech. rep., DTIC Document, 2002.

[12] F. J. Harris, "On the use of windows for harmonic analysis with the discrete fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[13] M. Frigo and S. G. Johnson, "Fftw: An adaptive software architecture for the fft," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 3, pp. 1381–1384, IEEE, 1998.

[14] D. A. Schade, T. C. Winant, J. Alforque, J. Faul, K. B. Groves, V. Horvatich, M. A. Middione, C. Tarantino, and J. R. Turner, "Fast acting active protection system," Apr. 10 2007. US Patent 7,202,809.