Efficient Architecture and Hardware Implementation of Coherent Integration Processor for DVB-based Passive Bistatic Radar

Tao Shan[†‡], Shengheng Liu[†], Yimin D. Zhang[‡], Moeness G. Amin[‡], Ran Tao[†], and Yuan Feng[†]

[†]School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China

[‡]Center for Advanced Communications, Villanova University, Villanova, PA 19085, USA

Email: shantao@bit.edu.cn

Abstract

In this paper, the problem of efficient implementation of coherent integration processor in passive bistatic radars (PBRs) in the presence of range migration is addressed. We present a coherent integration architecture for PBR, which consists of a frequency-domain pulse compression module to reduce the overall runtime for the computation of the cross ambiguity function, and an efficient decimated Keystone transform module based on the chirp z-transform to compensate the range migration. The proposed architecture is then implemented in a hybrid central processing unit plus graphic processing unit scheme. Real measurement data are used to verify the superior integration performance and reduced computational complexity achieved by the proposed scheme.

Index Terms

Passive bistatic radar (PBR), range migration compensation, frequency-domain pulse compression (FDPC), chirp z-transform (CZT), graphic processing unit (GPU).

I. Introduction

Passive bistatic radars (PBRs) exploit non-cooperative illuminators, such as commercial broadcast stations [1]–[3], cellular base stations [4], [5], and Wi-Fi hotspots [6], [7], to perform detection and localization of moving targets. The inherent covertness of PBR makes it immune from various electronic countermeasures, such as active jamming and anti-radiation weapon. In addition, the exploitation of the very high frequency (VHF) or ultrahigh frequency (UHF) bands and the spatial diversity of PBR achieved through multi-static operation offer the potential capability for anti-stealth. Furthermore, most illuminators of opportunity have a wide coverage and low beam directivity and, therefore, can benefit the detection of low altitude targets. PBR systems reap these advantages [8] at a low cost compared to traditional active radar counterparts and can be easily embedded into different mobile platforms. These facts enabled PBR to witness a significant growth of interest and extensive research in recent years [9]–[11].

Among the available non-cooperative illuminators, commercial broadcast stations, particularly those for television broadcasting, are considered attractive for surveillance purposes due to their high power and wide coverage [2], [12]. Nowadays digital video broadcast (DVB) schemes are gradually replacing the traditional analog systems to become the mainstream form of terrestrial television broadcast. DVB signals have more desirable ambiguity-domain characteristics with low sidelobes because of the wide signal bandwidth and the pseudo-random digital modulation [1], [13]. These signal properties make the DVB signal one of the most preferred illumination sources of opportunity in the foreseeable future.

In order to perform coherent integration, PBR systems compute the cross ambiguity function (CAF) [14], [15], which is the two-dimensional cross-correlation with respect to the time delay and the Doppler shift, between the reference signal directly received from an illuminator and the echo signal received from target scattering. The peaks of the CAF surface reveal the bistatic range and velocity of moving targets. The computation of the CAF is the most computationally demanding step in PBR signal processing and, therefore, has been the subject of intensive investigations [1], [15]–[18].

On the other hand, without dedicated transmitters, the echo signals received from distant targets with a small radar cross section become extremely weak and, thereby, may easily be obscured by noise and residue clutter. To overcome this problem, an extended integration time is usually exploited to obtain a sufficiently high signal-to-noise ratio (SNR). In this case, a high speed target will yield an extended cross-correlation peak that spreads over several time delay bins. This gives rise to range migration which significantly degrades the coherent integration result. Two methods, i.e., the envelope shift (ES) and the Keystone transform (KT), are commonly used to compensate for range migration in PBR. In [19], we showed that both KT and ES methods exhibit similar integration gains when applied in a PBR system.

One major limitation of the ES [20] is that the exact or at least an approximate bound of the target's radial velocity must be known *a priori*. In addition, to reduce the estimation error caused by fractional time delays, the ES method requires the interpolation of the echo signals which is computationally demanding. In [21], a generalized CAF to extend integration time was introduced, where the envelope interpolation (or resampling) was suggested to be simplified by calculating the chirp z-transform (CZT) [22]. However, as the change of the time scale corresponds to the change of the echo time delay, this stretch processing was actually a duplicate of the ES method. The KT [23] also involves computationally intensive sinc interpolation. In [24], the sinc interpolation was replaced by the CZT, which lowers the computational complexity and induces less phase and amplitude errors in data processing. For PBR, a similar method to effectively perform range migration compensation by using CZT-based KT is also feasible. This fact motivated our proposed scheme.

Regarding the hardware implementation of coherent integration for PBR, a system based on IBM Cell BE was presented in [25]. Subsequently, a real-time system using graphic processing unit (GPU) was developed in [26], [27] to speed up the signal processing chain. Hybrid schemes jointly exploiting central processing unit (CPU) and programmable GPU were reported in [28], [29], where the computationally intensive parallel processes are implemented in the GPU, whereas the sequential processes, such as flow control, are implemented in the CPU, making the architecture more flexible. The aforementioned contributions on efficient hardware implementation have laid the foundation of our proposed alternative method.

In this paper, the fast CAF computation is realized by exploiting frequency-domain pulse compression (FDPC) [30], and a highly efficient CZT-based decimated KT module is developed, then the proposed architecture is implemented on a CPU+GPU heterogeneous platform. The remainder of this paper is organized as follows. Section II introduces the mathematical model of moving target echo signals. In Section III, the proposed architecture is presented. An efficient GPU implementation of the proposed architecture is described in Section IV. The results of a real data experiment and their analysis are presented in Section V. Finally, the conclusion is drawn in Section VI.

II. Problem formulation

Fig. 1 sketches the geometry of a typical PBR system, where Tx and Rx denote the respective locations of the transmitter and the receiver. The reference and surveillance antennas are co-located, where the reference antenna is steered toward the transmitter to receive the reference signal, while the surveillance antenna points in the direction to be surveyed to receive the target echo signal.

The target is located at position O at the initial time t =0 , and then flies along a straight line, which slants at an angle of δ to the bistatic angular bisector, with a constant velocity v. Its position at time t is denoted as P. The bistatic angle is γ, and L_b denotes the baseline distance. In addition, r_{T0} and $r_T(t)$ represent the distances between the target and the transmitter measured at the initial time and at time instant t , respectively, whereas r_{R0} and $r_R(t)$ denote the distances between the target and the receiver measured at the initial time and at time instant t .

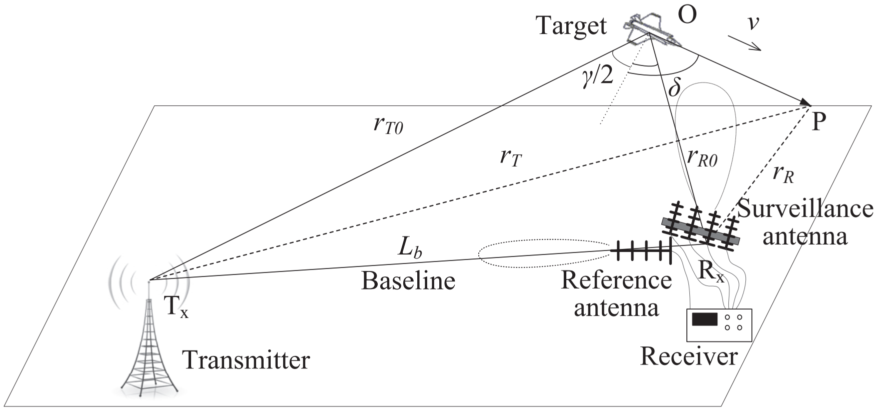


Fig. 1. PBR configuration.

From Fig. 1, we can obtain the bistatic range as

$$r(t){=}r_T(t){+}r_R(t){=}\sqrt{r_{T0}^2{+}(vt)^2{-}2r_{T0}vt\cos(\delta{+}\frac{\gamma}{2})}{+}\sqrt{r_{R0}^2{+}(vt)^2{-}2r_{R0}vt\cos(\delta{-}\frac{\gamma}{2})}. \tag{1}$$

The time delay and the Doppler frequency are respectively represented as [31]

$$\tau(t){=}\frac{r(t)-L_b}{\mathrm{c}}, \tag{2}$$

$$f_d(t)=-\frac{1}{\lambda}\frac{\mathrm{d}r(t)}{\mathrm{d}t}, \tag{3}$$

where c and λ represent the velocity of light and the wavelength of the transmitted signal, respectively.

Substituting (1) into (2) and (3), performing Taylor series expansion at $t=0$, and neglecting the quadratic and higher terms, we obtain [31]

$$\tau(t)\approx\tau_0{+}a_\tau t=\frac{r_{T0}+r_{R0}-L_b}{\mathrm{c}}-\frac{2v\cos\delta\cos\frac{\gamma}{2}}{\mathrm{c}}t, \tag{4}$$

$$f_d\approx-\frac{2v}{\lambda}\cos\delta\cos\frac{\gamma}{2}. \tag{5}$$

Neglecting the reference channel noise, we express the baseband reference signal as $s_r(t)=s(t)$, where $s(t)$ is the transmitted DVB baseband signal. On the other hand, the baseband echo signal of the target is expressed as [15]

$$s_e(t)=As(t-\tau_0-a_\tau t)\mathrm{e}^{-\mathrm{j}2\pi f_c(\tau_0+a_\tau t)}=As(t-\tau_0-a_\tau t)\mathrm{e}^{\mathrm{j}2\pi(f_d t-f_c\tau_0)}, \tag{6}$$

where A is an attenuation coefficient. Note that the $a_\tau t$ term causes range migration of the target echo signal. Generally, when the overall integration time is relatively short, the coherent integration gain increases with the integration time. When the integration time reaches a certain extent, however, the range migration will incur an energy loss and, therefore, must be properly compensated for to achieve effective signal integration for target detection.

III. Design of signal processing flow

The proposed scheme extends the integration time to enhance the SNR. In so doing, it causes the following three main problems:

First, the sidelobe peaks on the CAF surface of a DVB signal are folded in the frequency range between 0 and $f_{d\max}$ when the proposed batch processing method is used. We have proposed different approaches to solve this problem. For Chinese standard DVB signals, the sidelobe peaks can be eliminated by conditioning the frame pseudorandom noise header of the reference signal [32]; while for European standard DVB signals, we excluded the sidelobe peaks by treating the pilot carriers [33].

Second, the sidelobe peaks of clutters on the CAF surface, if unsuppressed, can mask weak targets, degrade the detection performance, and increase the false alarm rate. The effect of clutter sidelobe peaks can be mitigated using the clutter cancellation algorithms [4], [34].

Third, due to the long integration time used in the proposed scheme, the Doppler walk can cause a strong effect on the integration performance. In this case, a chirp-Fourier or fractional Fourier transform based method can be adopted to mitigate the Doppler walk [19], [21], [35].

In summary, since solutions to these problems have been well developed, we will not discuss them in this paper.

A. CAF based on FDPC

In DVB-based PBR applications, the discrete CAF is defined as [2]

$$\Psi\left(d,q\right)=\sum_{n=0}^{L-1}s_e\left(n+d\right)s_r^*\left(n\right)\mathrm{e}^{-\mathrm{j}2\pi nq/L}, \tag{7}$$

where $s_e(n)$ and $s_r(n)$ denote the sampled echo and reference signals, respectively, d and q represent the number of time delay samples and Doppler frequency bins, and L refers to the number of integration data samples. In addition, $(\cdot)^*$ denotes complex conjugate. It is clear from (7) that the CAF can be interpreted as the Fourier transform of the product of $s_e(n)$ and $s_r^*\left(n\right)$ with different delays.

For a long integration time and a large number of time delay samples, the computation of CAF in (7) becomes highly complex. To reduce the computation complexity, we propose the FDPC [30] method which proceeds as follows.

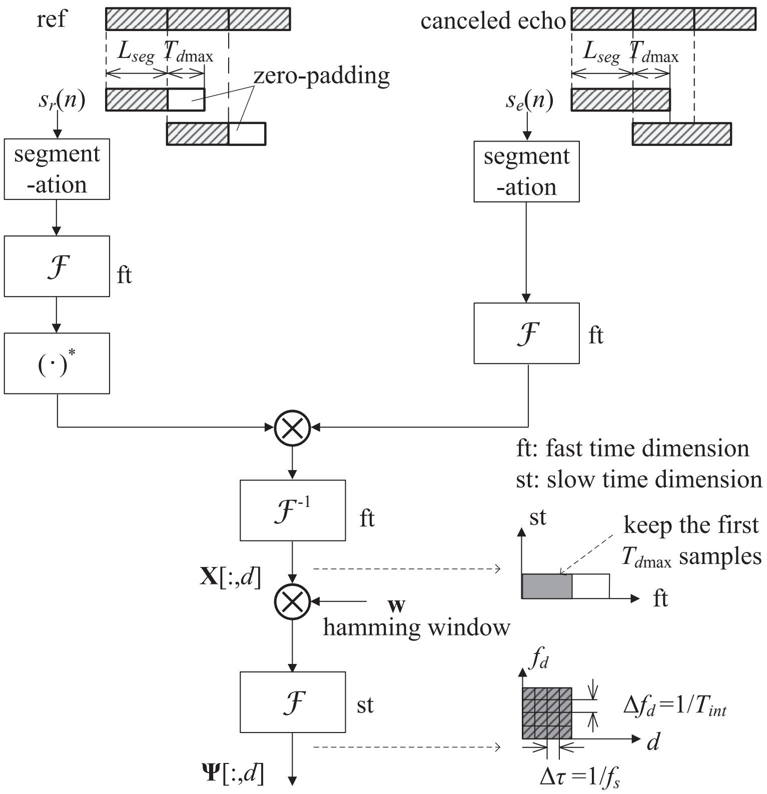


Fig. 2. Schematic diagram of the proposed CAF algorithm flow.

In the preprocessing stage, the input continuous-wave signal is divided into multiple segments, which are treated as pulses afterwards. The details of the proposed algorithm, including the segmentation procedure and the subsequent processing, are illustrated in Fig. 2, where $\mathcal{F}$ and $\mathcal{F}^{-1}$ respectively denote the fast Fourier transform (FFT) and the inverse FFT (IFFT) operations. In the segmentation procedure, both the reference and the echo signals are divided into N_{seg} segments. The length of the reference signal in each segment is L_{seg} samples, whereas it is L_{seg} + T_{dmax} samples for the echo signal, and T_{dmax} denotes the number of maximum time delay samples in the CAF map. As such, there is an overlap of T_{dmax} samples between two adjacent echo signal segments. The FDPC is performed to every N_{seg} segments. After segmentation, $T_r = L_{seg}/f_s$ can be considered as the pulse repetition time, where f_s represents the baseband sampling rate.

Denote m as the number of segments corresponding to the integration time. Then, the segmented reference signal can be written in a two-dimensional form as

$$\begin{aligned} \tilde{s}_r(m,\tilde{n}) = [s_r\left(\tilde{n}+(m-1)L_{seg}\right), \mathbf{0}_{1\times T_{d\max}}], \\ m = 1,2,\ldots,N_{seg}, \tilde{n} = 1,2,\ldots,L_{seg}, \end{aligned} \tag{8}$$

where m and $\tilde{n} = n - mL_{seg}$ represent the indices of the slow and fast time, respectively.

Similarly, the segmented echo signal can be expressed from (6) as

$$\begin{aligned} \tilde{s}_e(m,\tilde{n}) = s_e\left(\tilde{n}+(m-1)L_{seg}\right), \\ m = 1,2,\ldots,N_{seg}, \tilde{n} = 1,2,\ldots,L_{seg}+T_{d\max}. \end{aligned} \tag{9}$$

Based on the above segmentation, the FDPC corresponding to the m-th pulse is calculated as

$$\mathbf{X}[m,:]_{N_{seg}\times \tilde{L}_{seg}} = \mathcal{F}^{-1}\{\mathcal{F}^*\{\tilde{s}_r(m,\tilde{n})\}\cdot\mathcal{F}\{\tilde{s}_e(m,\tilde{n})\}\}, \tag{10}$$

where $\tilde{L}_{seg} = 2^{\lceil \log_2(L_{seg}+T_{d\max})\rceil}$, and $\lceil\rceil$ represents the operation of rounding up to the closest integer. Denote T_{int} as the integration time, then, we have $f_s T_{int} = L_{seg}N_{seg} = L$. Note that, upon performing the Fourier transform, each reference signal segment is zero-padded toward its end. Let f_r denote the pulse repetition frequency. To avoid velocity ambiguity, the maximum target Doppler must satisfy $f_{dmax} < f_r/2 = f_s/(2L_{seg})$. For phase-shift keying signals, the SNR loss of a target echo with Doppler f_d induced by the FDPC can be expressed as [36]

$$\text{Loss} = -20\text{log}_{10}\left|\frac{\sin\left(\frac{\pi f_d L_{seg}}{f_s}\right)}{L_{seg}\cdot\sin\left(\frac{\pi f_d}{f_s}\right)}\right|. \tag{11}$$

Since $\pi f_{d\max} \ll f_s$, it can be simplified as

$$\text{Loss} = -20\text{log}_{10}\left|\frac{\sin\left(\frac{\pi f_d L_{seg}}{f_s}\right)}{\frac{\pi f_d L_{seg}}{f_s}}\right| = -20\text{log}_{10}\left|\text{sinc}\left(\frac{f_d L_{seg}}{f_s}\right)\right|. \tag{12}$$

To ensure this SNR loss to be less than 1 dB, the largest Doppler frequency $f_{d\max}$ should satisfy:

$$\text{Loss} = -20\text{log}_{10}\left|\text{sinc}\left(\frac{f_{d\max}L_{seg}}{f_s}\right)\right| < 1\text{dB}. \tag{13}$$

As such, we obtain

$$\frac{f_{d\max}L_{seg}}{f_s} < 3.824 \quad \Rightarrow \quad L_{seg} < \frac{f_s}{3.824 f_{d\max}}. \tag{14}$$

After pulse compression, Doppler filtering is performed across the slow-time dimension to yield the CAF surface. The output of the d-th range bin Doppler filtering is expressed as

$$\mathbf{\Psi}[:,d]_{N_{seg}\times T_{d\max}} = \mathcal{F}\{\mathbf{X}[:,d]\odot\mathbf{w}\}, \tag{15}$$

where $\odot$ represents element-wise multiplication, and $\mathbf{w}$ is an $N_{seg}\times 1$ hamming window.

The Doppler shift resolution and the time bin width in the CAF map can be respectively obtained as

$$\begin{cases} \Delta f_d = 1/T_{int} = f_r/N_{seg} = f_s/(N_{seg}L_{seg}), \\ \Delta\tau \ \ = 1/f_s. \end{cases} \tag{16}$$

It can be readily shown that the computations involved in the block diagram amount to

$$\#\text{FDPC}-\text{CAF} = \underbrace{N_{seg}\tilde{L}_{seg}\left(\frac{3}{2}\log_2\tilde{L}_{seg}+1\right)}_{\text{pulse compression}} + \underbrace{T_{d\max}\tilde{N}_{seg}\left(1+\frac{1}{2}\log_2\tilde{N}_{seg}\right)}_{\text{Doppler filtering}} \tag{17}$$

complex multiplications, where $\tilde{N}_{seg} = 2^{\lceil\log_2(N_{seg})\rceil}$.

B. Range migration compensation utilizing CZT

When the signal is segmented into pulses, the pulse period should be sufficiently short so that the target range migration within a pulse is negligible. As such, the length of each segment satisfies [37]

$$\frac{L_{seg}}{f_s}v_{d\max} < \Delta R = \frac{\mathrm{c}}{2B} \quad \Rightarrow \quad L_{seg} < \frac{\mathrm{c}f_s}{2Bv_{d\max}}, \tag{18}$$

where ΔR , B and $v_{d\max}$ represent the range resolution, the bandwidth of the signal, and the maximum radial velocity of target, respectively. From (14) and (18), we have

$$L_{seg} < \min\left\{\frac{f_s}{3.824f_{d\max}}, \frac{\mathrm{c}f_s}{2Bv_{d\max}}\right\}. \tag{19}$$

Converting (9) to the frequency domain yields

$$S_e(m,f) = AS_m(f)\mathrm{e}^{-\mathrm{j}2\pi(f+f_c)\tau_0}\mathrm{e}^{-\mathrm{j}2\pi(f+f_c)a_\tau mT_r}, \tag{20}$$

where $S_m(f)$ is the Fourier transform of $\tilde{s}_r(m,\tilde{n})$. Following Fig. 2, we obtain the conjugate product of $S_e(m,f)$ and $S_r(m,f)$ as

$$S_{re}(m,f) = S_e(m,f)S_r{}^*(m,f) = A|S_m(f)|^2\mathrm{e}^{-\mathrm{j}2\pi(f+f_c)\tau_0}\mathrm{e}^{-\mathrm{j}2\pi(f+f_c)a_\tau mT_r}. \tag{21}$$

To better explain the range migration phenomenon described in (21), we draw the schematic diagram of the phase response with respect to the frequency in Fig. 3. The first exponential term in (21) implies the target location at the initial time, whereas the second term reflects the variation of signal phase with the slow time. Note that, however, because of the presence of f , the phase changes differently with respect to the slow time for different frequencies. According to Fig. 3, to guarantee in-phase stacking, the oblique phase-frequency response line should be flattened. This can be achieved by using the KT.

In our work, the following KT is adopted to rescale the slow-time dimension [23],

$$\tilde{m} = \frac{f_c+f}{f_c}m. \tag{22}$$

Substitute (22) into (21), we obtain

$$S_{re}(\tilde{m}, f) = A|S_m(f)|^2 \mathrm{e}^{-\mathrm{j}2\pi(f+f_c)\tau_0} \mathrm{e}^{-\mathrm{j}2\pi f_c a_\tau \tilde{m} T_r}. \tag{23}$$

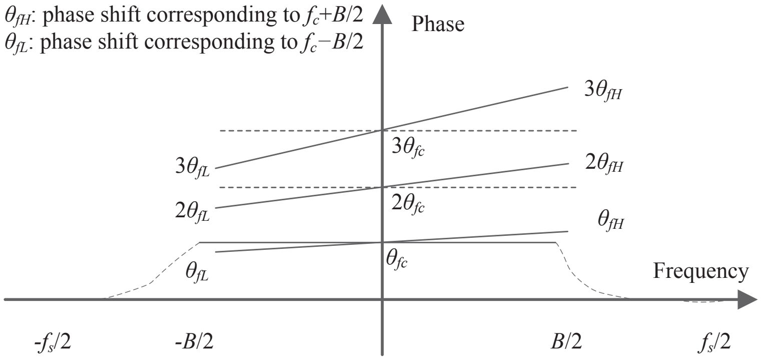

θ_fH: phase shift corresponding to f_c+B/2
θ_fL: phase shift corresponding to f_c−B/2
Phase
3θ_fH
3θ_fc
3θ_fL
2θ_fH
2θ_fL
2θ_fc
θ_fH
θ_fc
θ_fL
Frequency
-f_s/2
-B/2
B/2
f_s/2

Fig. 3. Phase response with respect to the frequency.

Therefore, the phase change rate of all frequencies is equal to that of the carrier frequency. Convert (23) to the time domain, we can obtain

$$\tilde{s}_{re}(\tilde{m}, \tilde{n}) = A \cdot s_m(\tilde{n} - \tau_0 f_s) \mathrm{e}^{-\mathrm{j}2\pi f_c \tau_0} \mathrm{e}^{\mathrm{j}2\pi f_d \tilde{m} T_r}, \tag{24}$$

where $s_m(\tilde{n}) = \mathrm{IFFT}\{|S_m(f)|^2\}$. (24) shows that the range migration has been compensated for.

KT has been realized by various methods, such as sinc interpolation, discrete Fourier transform (DFT), and CZT. In this paper, we use the CZT method, which requires the lowest computational complexity for large data sets. The CZT algorithm computes the z-transform along circular or spiral contours starting at any arbitrary point in the z-plane. Define a sequence $y(n)$ as a column of $S_{re}(m, f)$, which has N non-zero entries, $N < \infty$. Then, the CZT at point $z_k = AW^{-k}$ of this sequence is defined as [22]

$$\tilde{y}(k) = W^{k^2/2} \sum_{n=0}^{N-1} y(n) A^{-n} W^{n^2/2} W^{-(k-n)^2/2}, \qquad k = 0, 1, \ldots, M-1, \tag{25}$$

where M is an integer that represents the number of points of the complex spectrum to be analyzed, and A and W are arbitrary complex values expressed as [22]

$$\begin{cases} A = A_0 \mathrm{e}^{\mathrm{j}2\pi\theta_0}, \\ W = W_0 \mathrm{e}^{\mathrm{j}2\pi\varphi_0}, \end{cases} \tag{26}$$

with A_0, W_0, θ_0 and φ_0 denoting the vector radius of the initial sample z_0, the extensional rate of the spiral contour, the phase angle of z_0, and the angular spacing between adjacent samples, respectively.

For the purpose of PBR range migration compensation, the main steps of CZT-based KT are summarized in Table I.

TABLE I

CZT-BASED KT

Parameters:	$N,\ M,\ \tilde{L}\ ,\ A_0,\ W_0,\ \theta_0,\ \varphi_0,\ f_c,\ f$
	$N=M=N_{seg},\ \tilde{L}=2^{\lceil \log_2(N+M-1)\rceil}\ ,\ A_0=W_0=1,\ \theta_0=0,\ \varphi_0=(f_c+f)/(f_cM)$
CZT:	Define $g(n)=\begin{cases} y(n)W^{n^2/2}, & 0\le n\le M-1, \\ 0, & M\le n\le \tilde{L}-1, \end{cases}$
	and $h(n)=\begin{cases} W^{-n^2/2}, & 0\le n\le M-1, \\ W^{-(\tilde{L}-n)^2/2}, & M\le n\le \tilde{L}-1, \end{cases}$
	then $\tilde{y}(k)=\mathcal{F}^{-1}\{\mathcal{F}\{g(n)\}\cdot\mathcal{F}\{h(n)\}\}W^{k^2/2}, \qquad k=0,1,\ldots,M-1.$

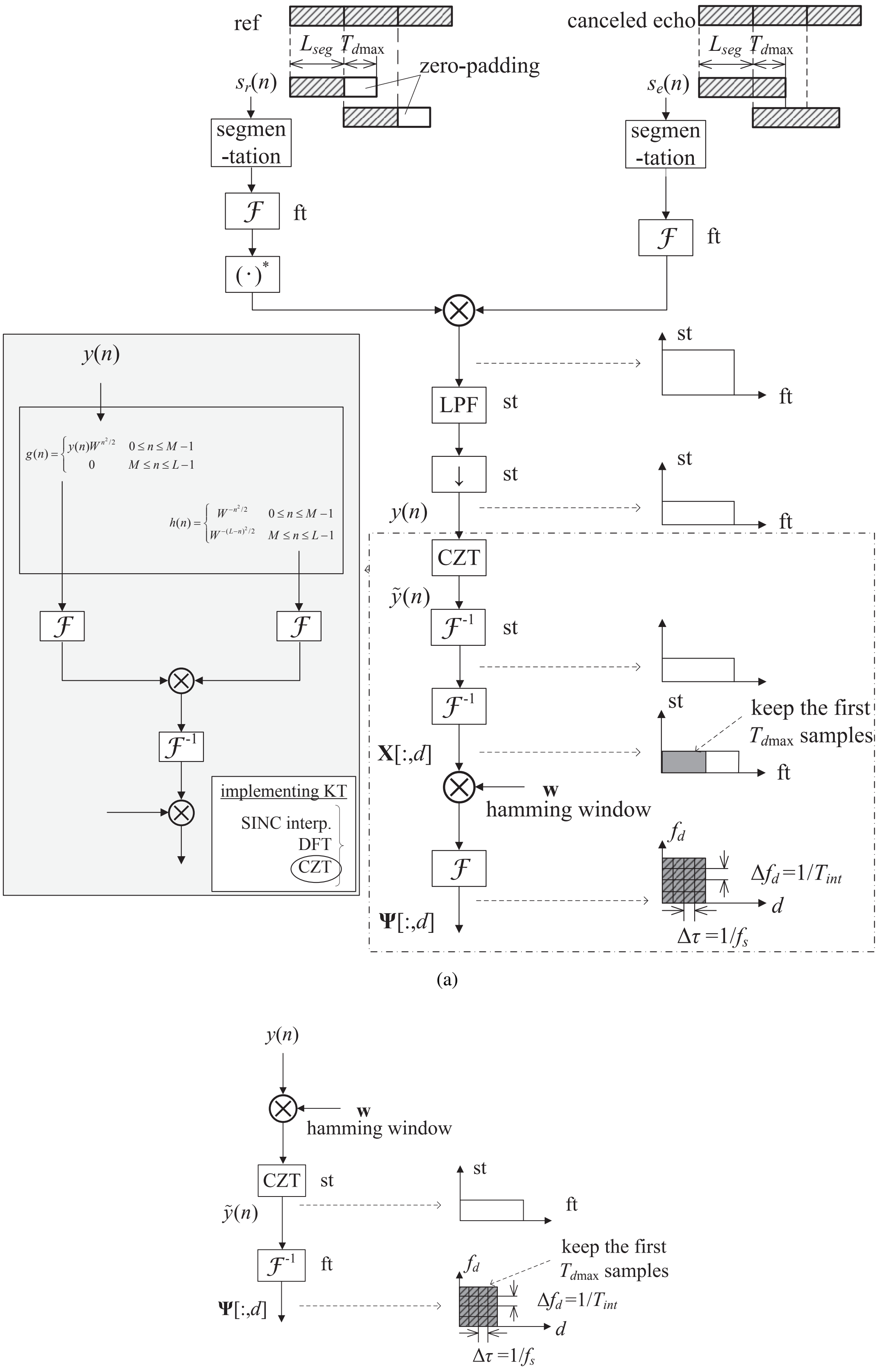
C. *Efficient coherent integration processor with range migration compensation*

To facilitate efficient implementation on a GPU platform, the computational complexity of the coherent integration processor is further lowered by down-sampling and algorithm pruning. As shown in (14), to reduce the SNR loss, the sampling frequency in the slow time is higher than $f_{d\max}$. To avoid redundant computations when directly performing FFT with respect to the slow time, we pass the signal through a low-pass filter (LPF) and decimate it prior to performing KT to a lower sampling rate, so that the computation complexity is reduced as the unambiguous Doppler frequency range becomes smaller. On the other hand, as described in Table I, the last step of CZT-based KT is an IFFT operation with respect to the slow time, while an FFT operation along the same slow-time dimension is performed in the following Doppler filtering stage. As a result, this FFT/IFFT pair can be pruned. Fig. 4 shows the updated algorithm flowchart.

The computation of the proposed FDPC-CZT-based CAF in Fig. 4 involves a total number of

$$\begin{aligned}\#\mathrm{FDPC{-}CZT} = &\underbrace{N_{seg}\tilde{L}_{seg}\left(\log_2\tilde{L}_{seg}+1\right)}_{\text{pulse compression}} + \underbrace{\frac{\tilde{L}_{seg}}{\tilde{D}}\left\{N_{seg}(N_L+4)+\tilde{L}(1+\frac{3}{2}\log_2\tilde{L})\right\}}_{\text{CZT}} \\ &+ \underbrace{\frac{N_{seg}}{2\tilde{D}}\tilde{L}_{seg}\log_2\tilde{L}_{seg}}_{\text{IFFT}} \end{aligned} \tag{27}$$

complex multiplications, where $\tilde{D}$ and N_L denote the decimation ratio and the order of the LPF, respectively.



(a)

(b)

Fig. 4. Modified signal processing flow of the coherent integration processor. (a) The algorithm flow chart. (b) Prune the redundant FFT/IFFT pair.

IV. Hardware implementation

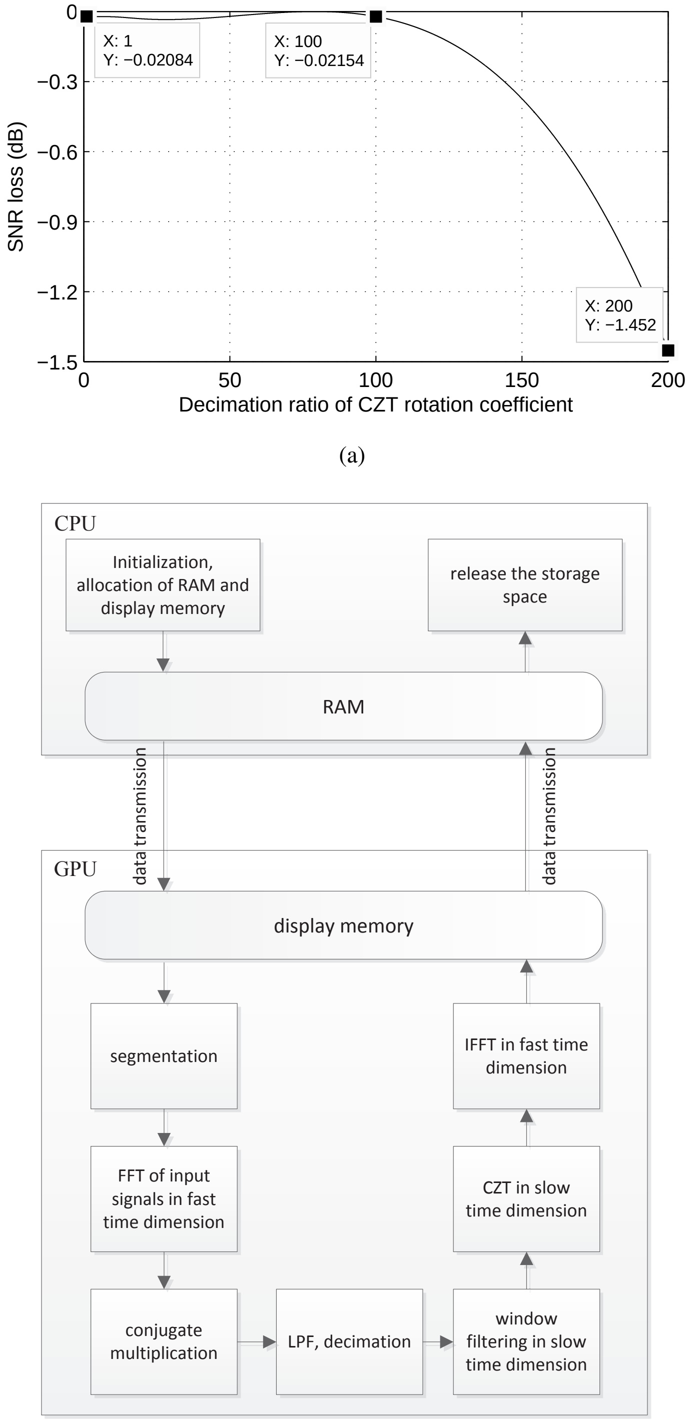
A. Efficient CZT implementation with storage of decimated rotation coefficient

When performing CZT with respect to the slow time, extensive computation of the rotation coefficients $W^{n^2/2}$ and $W^{-(k-n)^2/2}$ is required in each column, thereby necessitating considerable computing resources. As these rotation coefficients are determined once the numbers of segments and delay bins are selected, we can store the coefficients as a lookup table for later use. Toward this end, however, a large storage space is needed. For instance, if f_s =10 MHz, T_{int} =1.0 s, N_{seg} =2500, L_{seg} =4000, $M = N = N_{seg}/2$ =1250, $T_{d\max}$ =4000, the number of samples in the fast-time frequency domain is $\tilde{L}_{seg} =$ 8192, and $\tilde{L}$ in TABLE II is 4096. With a 1/2 decimation, the coefficients require a storage space of approximately 416 MB. This result assumes that each point occupies a storage space of 8 B and includes a storage space of 80 MB for coefficients $W^{n^2/2}$ of $N \times \tilde{L}_{seg}$ =10 million points, 256 MB for coefficients $h(n)$ of $\tilde{L} \times \tilde{L}_{seg}$ =32 million points, and 80 MB for coefficients $W^{k^2/2}$ of $N \times \tilde{L}_{seg}$ =10 million points.

On the other hand, the adopted graphics card GTX460 only has a display memory of 1 GB, of which 700 MB is available for use because some of the memory is preserved for the display processing. Therefore, the memory required for the entire computing process and the rotation coefficient storage makes the problem impractical. In addition, because the difference between the corresponding rotation coefficients employed by the adjacent columns in the slow-time dimension is negligible, we can decimate the coefficients to reduce the required storage space, i.e., several adjacent columns share the same rotation coefficient in the proposed design. The SNR loss induced by different decimation ratios is plotted in Fig. 5(a). It can be observed that, when the decimation ratio is less than 100, the performance degradation due to the decimation is negligible, and when the decimation ratio exceeds 100, the SNR degrades sharply. Thus, a decimation ratio of 100 is chosen in our work.

B. Compute unified device architecture programming

Due to the inherent data-parallelism of a coherent integration processing, it is suitable to exploit the compute unified device architecture (CUDA) GPU realization. In a CUDA paradigm, the CPU is regarded as a HOST, which is in charge of logical tasks and sequential computing, whereas the GPU is regarded as a DEVICE, which is responsible for the massive computationally intensive parallel operations. During the execution of a CUDA program, the HOST code mainly deals with system startup, allocation of the random access memory (RAM) and display memory, data transmission, call of DEVICE kernel function, and release of the memories. On the other hand, the DEVICE code mainly handles data processing.


0
−0.3
−0.6
−0.9
−1.2
−1.5
SNR loss (dB)
0
50
100
150
200
Decimation ratio of CZT rotation coefficient
X: 1
Y: −0.02084
X: 100
Y: −0.02154
X: 200
Y: −1.452
CPU
Initialization, allocation of RAM and display memory
release the storage space
RAM
data transmission
data transmission
GPU
display memory
segmentation
FFT of input signals in fast time dimension
conjugate multiplication
LPF, decimation
window filtering in slow time dimension
CZT in slow time dimension
IFFT in fast time dimension

(a)

(b)

Fig. 5. Proposed hardware architecture. (a) SNR loss induced by CZT rotation coefficient decimation. (b) Flow diagram of the proposed architecture.

The main steps of the CPU+GPU implementation of the proposed algorithm are listed as follows.

Step 1: Initialize the CPU program. This step includes the allocation of the RAM and display memory for the input signal and calculation result.

Step 2: Allocate space for the parameters configured in GPU, and transfer the reference and echo signals in the CPU RAM to the display memory.

Step 3: Segment the reference and echo signals in display memory, and perform FFT in the fast-time dimension.

Step 4: Calculate the conjugate multiplications with the programmed kernel functions, and proper Grid and Block numbers can be chose to optimize this step.

Step 5: In the slow-time dimension, multiply the outcome of the *Step 4* by low pass filter coefficients and then perform decimation. Other kernel functions are employed to achieve parallel computing.

Step 6: Multiply the outcome of the *Step 5* by the window filter coefficients in the slow-time dimension with the programmed kernel function of parallel vector multiplication.

Step 7: Perform CZT with respect to the slow time by using the stored rotation coefficients.

Step 8: Perform IFFT with respect to the fast time by using the cuFFT (the NVIDIA® CUDA FFT product, see [38]) library in parallel.

Step 9: Transfer the final result from the GPU display memory to the CPU RAM, and release the storage space.

A flow diagram of the above steps is illustrated in Fig. 5(b).

V. Experiments and analysis

In this section, the processing results using real data are presented to examine the performance of the proposed algorithm. In our experiment, signals of the Channel 33, a high-definition channel hosted by the China Central Television (CCTV), were employed. The signals adopt the Digital Multimedia Broadcast-Terrestrial/Handheld format, which is supported by the Chinese mandatory standard GB20600-2006 [39]. The key parameters are listed in Table II.

TABLE II

Parameters of transmitted signal used in the experiment

Carrier number	Carrier frequency	Polarization	Bandwidth	Transmitted power
1	674 MHz	horizontal	7.56 MHz	3 kW

The scene of data collection is shown in Fig. 6. The reference antenna was steered towards the CCTV Tower (approximately 23.8 km Northeast of Liangxiang Campus, Beijing Institute of Technology) whereas the surveillance antenna was pointed approximately toward Southeast. The reference antenna is a Yagi antenna whose gain is about 13 dB, and the surveillance antenna is a 48-element planar array whose combined antenna and array gain is about 20 dB. These antennas are installed on the roof of the Library building. Several departure and arrival routes for the Beijing Capital Airport are contained within the surveillance antenna pattern. The experiment environment is shown in Table III.

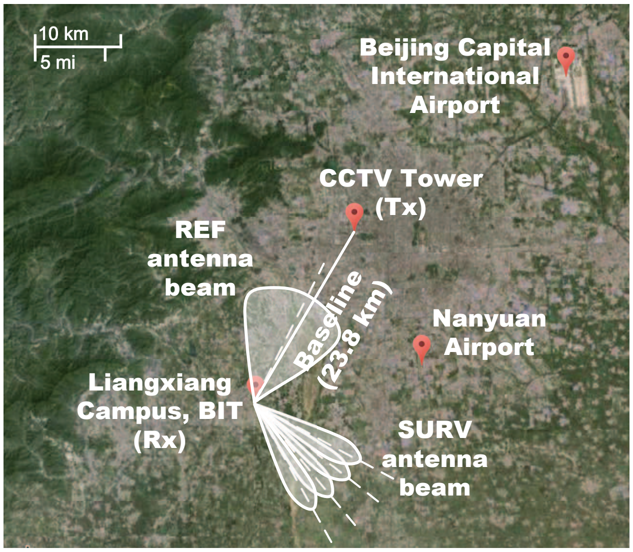
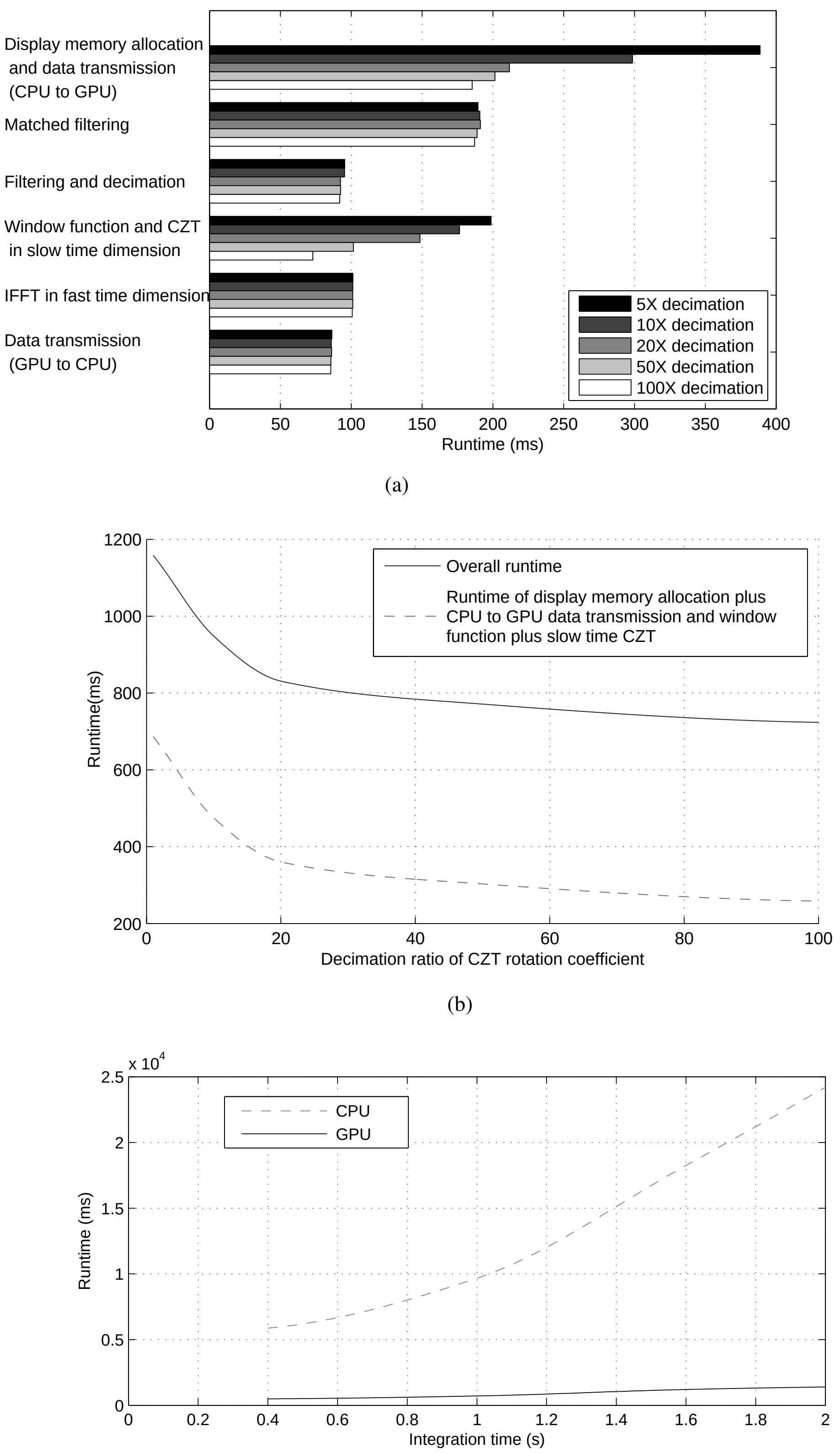


Fig. 6. Experimental geometry.

TABLE III

EXPERIMENT ENVIRONMENT

Hardware environment	
GPU model	GTX460
Display memory	1.0 GB
Bus interface	PCIE-x16
CPU model	Intel(R) Core(TM) 2 Duo E8400
CPU RAM	2.0 GB
Software environment	
Operation system	Window 7 32 bit
CPU compiling environment	VS2008
GPU compiling environment	CUDA 5.0
Compiler	Nvcc, gcc
Matlab version	R2012a



(a)

(b)

(c)

Fig. 7. Runtime statistics. (a) Runtime of each module with different decimation ratio of the CZT rotation coefficients. (b) The trend of runtime vs. the decimation ratio of CZT rotation coefficient. (c) Runtime comparison between CPU and GPU implement.

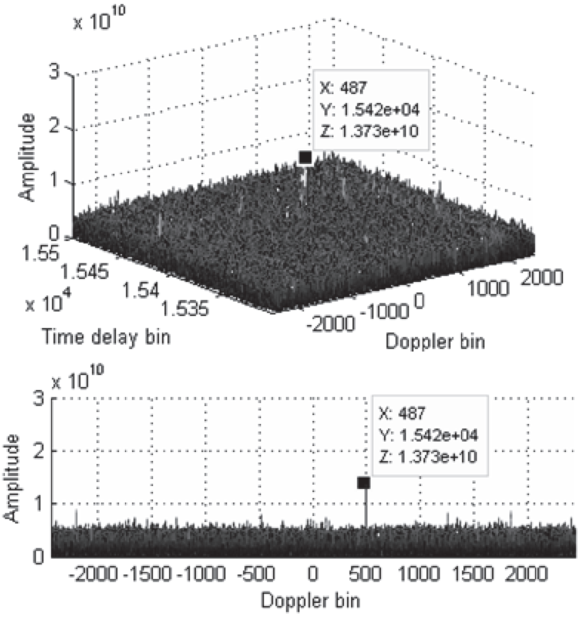
The runtime statistics is plotted in Fig. 7. In the experiment process, the baseband sampling rate is 10 MHz, and the integration time is 1.0 s. In addition, the number of segments is 2500, and there are 4000 range bins. It is clear from Fig. 7(a) that the alteration of the CZT coefficient decimation ratio mainly affects the runtime of display memory allocation plus CPU to GPU data transmission and slow-time CZT. Fig. 7(b) shows that, as the CZT decimation ratio increases, the runtime of the two modules (depicted by dashed line) decreases in a similar tendency as the overall runtime. This is explained by the fact that, when the CZT decimation ratio is small, a large amount of stored CZT coefficients will consume a considerable time resource to perform the transferring and accessing operations. Based on a compromise between the SNR loss and the computational complexity, we select the CZT coefficient decimation ratio to be 100. The procedure is further optimized by choosing a suitable dimension of the thread blocks and grids, and utilizing measures such as shared memory and host memory page locking [40]. The execution time is illustrated with respect to the integration time in Fig. 7(c). It is observed that, when integration time reaches 1.0 s, the execution time of the GPU is 713 ms, which increases the execution speed by a factor of 13 as compared to the CPU execution.

The real data experiment results using GPU program and MATLAB code are demonstrated in Fig. 8, where the target SNRs in Figs. 8(a)–8(d) are 17.42 dB, 21.20 dB, 21.17 dB and 21.16 dB, respectively. From Figs. 8(a) and 8(b), we can see that the proposed algorithm achieves a SNR gain of 3.78 dB compared to the conventional method. In addition, by comparing the results of Figs. 8(c) and 8(d), we can draw a conclusion that the GPU implementation has almost the same precision as the MATLAB simulation with CPU and, when the CZT coefficient decimation ratio is not large than 100, it has a little impact on the outcomes.

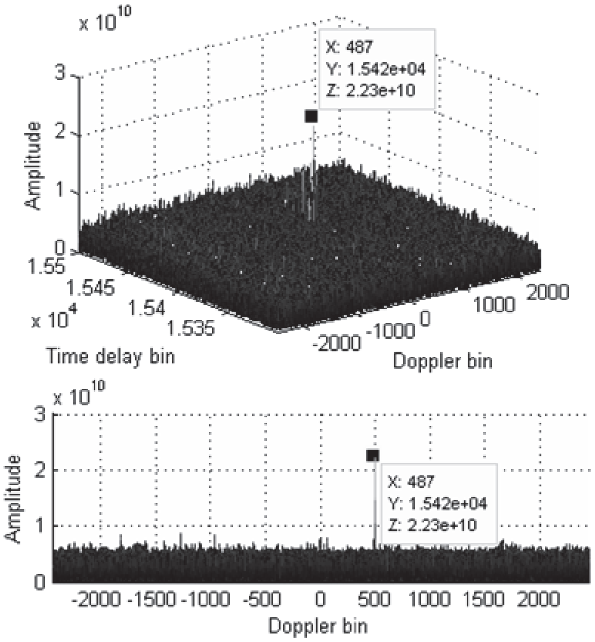
A comparison of the computational complexity between the proposed approach and the previous algorithms described in [15] is shown in Fig. 8(e). The corresponding simulation parameters are listed in Table IV, where D denotes the decimation rate of the zoom FFT. It is evident that the proposed algorithm significantly reduces the required computation complexity.

TABLE IV
SIMULATION PARAMETERS

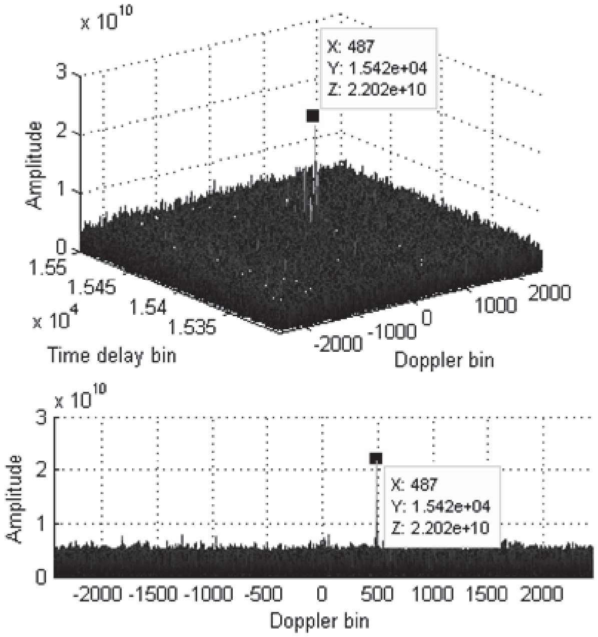
f_s	$T_{d\max}$	D	L_{seg}	$\tilde{D}$	N_L
10 MHz	5000	2000	2000	2	19


x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 1.373e+10
1.55
1.545
1.54
1.535
x 10^4
Time delay bin
-2000
-1000
0
1000
2000
Doppler bin
x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 1.373e+10
-2000 -1500 -1000 -500 0 500 1000 1500 2000
Doppler bin

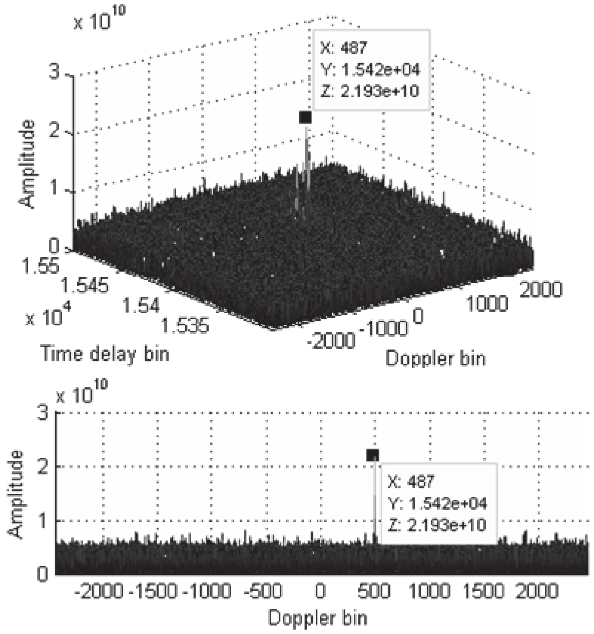
(a)


x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.23e+10
1.55
1.545
1.54
1.535
x 10^4
Time delay bin
-2000
-1000
0
1000
2000
Doppler bin
x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.23e+10
-2000 -1500 -1000 -500 0 500 1000 1500 2000
Doppler bin

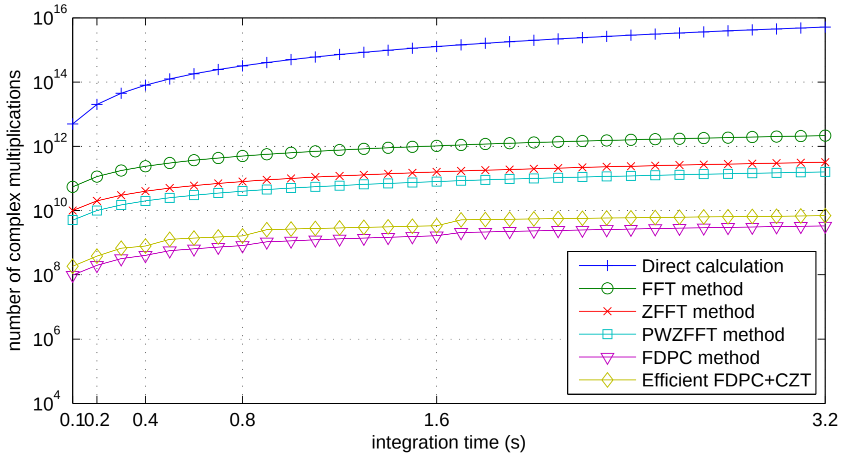
(b)


x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.202e+10
1.55
1.545
1.54
1.535
x 10^4
Time delay bin
-2000
-1000
0
1000
2000
Doppler bin
x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.202e+10
-2000 -1500 -1000 -500 0 500 1000 1500 2000
Doppler bin

(c)


x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.193e+10
1.55
1.545
1.54
1.535
x 10^4
Time delay bin
-2000
-1000
0
1000
2000
Doppler bin
x 10^10
3
2
1
0
Amplitude
X: 487
Y: 1.542e+04
Z: 2.193e+10
-2000 -1500 -1000 -500 0 500 1000 1500 2000
Doppler bin

(d)



(e)

Fig. 8. Real data experiment results. (a) CAF using the conventional algorithm without applying KT. (b) CAF using the proposed algorithm with no coefficient decimation. (c) GPU implementation of the proposed algorithm with no coefficient decimation. (d) GPU implementation of the proposed algorithm with the coefficient decimation ratio to be 100. (e) Comparison of computational complexity.

VI. Conclusions

We have considered the design of a coherent integration processor for a digital video broadcast-based passive bistatic radar (PBR). A long integration time results in two serious issues from a real-time implementation perspective, namely, a high runtime of coherent processing and a low integration gain due to the range walk through several range bins. In order to solve these problems, a fast coherent integration algorithm in the presence of range migration was proposed, by combining the frequency-domain pulse compression (FDPC) and decimated Keystone transform (KT) based on the chirp z-transform. The FDPC is capable of significantly reducing the computation of cross ambiguity function, which is the most exhausting module in the entire PBR signal processing, while the proposed simplified KT can satisfactorily compensate the range migration without alarmingly increasing the computational load. The efficient real-time GPU implementation of the proposed scheme was presented, which uses decimated coefficients to reduce the required storage space and, thus, further reduces the overall computation time. The effectiveness and validity of the proposed design has been verified through real data experiments.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grants No. 61172176, No. 61331021 and No. 61421001. Tao Shan would also like to thank the China Scholarship Council for the support.

REFERENCES

[1] Palmer, J. E., Harms, H. A., Searle, S. J., Davis, L. M.: 'DVB-T passive radar signal processing,' *IEEE Trans. Signal Process.*, 2013, **61**, (8), pp. 2116–2126

[2] Howland, P. E., Maksimiuk, D., Reitsma, G.: 'FM radio based bistatic radar,' *IEE Proc. Radar, Sonar Navig.*, 2005, **152**, (3), pp. 107–115

[3] Berger C. R., Demissie B., Heckenbach J., Willett P., Zhou S. L.: 'Signal processing for passive radar using OFDM waveforms,' *IEEE J. Sel. Topics Signal Process.*, 2010, **4**, (1) pp. 226–238

[4] Tan, D. K. P., Sun, H., Lu, Y., Lesturgie, M., Chan, H. L.: 'Passive radar using global system for mobile communication signal: theory, implementation and measurements,' *IEE Proc. Radar, Sonar Navig.*, 2005, **152**, (3), pp. 116–123

[5] Krysik, P., Samczynski, P., Malanowski, M., Maslikowski, L., Kulpa, K. S.: 'Velocity measurement and traffic monitoring using a GSM passive radar demonstrator,' *IEEE Aero. Electr. Syst. Mag.*, 2012, **27**, (10), pp. 43–51

[6] Colone, F., Falcone, P., Bongioanni, C., Lombardo, P.: 'Wifi-based passive bistatic radar: data processing schemes and experimental results,' *IEEE Trans. Aero. Electr. Syst.*, 2012, **48**, (2), pp. 1061–1079

[7] Chetty, K., Smith, G. E., Woodbridge, K.: 'Through-the-wall sensing of personnel using passive bistatic WiFi radar at standoff distances,' *IEEE Trans Geosci. Remote Sens.*, 2012, **50**, (4), pp. 1218–1226

[8] Griffiths, H.: 'Bistatics: introduction and historical background,' *NATO Research and Technology Organisation Lecture Series RTO-EN-SET-133 (Multistatic Surveillance and Reconnaissance: Sensor, Signals and Data Fusion)*, 2009

[9] Special issue on Passive Radar Systems, *IEE Proc. Radar, Sonar Navig.*, 2005, **152**, (3)

[10] Special issue on Passive Covert Radar, Part I, *IEEE Aero. Electr. Syst. Mag.*, 2012, **27**, (10)

[11] Special issue on Passive Covert Radar, Part II, *IEEE Aero. Electr. Syst. Mag.*, 2012, **27**, (11)

[12] Saini, R., Cherniakov, M.: 'DTV signal ambiguity function analysis for radar application,' *IEE Proc. Radar, Sonar Navig.*, 2005, **152**, (3), pp. 133–142

[13] Raout, J.: 'Sea target detection using passive DVB-T based radar,' Proc. Int. Conf. Radar, Adelaide, Australia, 2–5 September 2008, pp. 695–700

[14] Kulpa, K. S., Czekała, Z.: 'Masking effect and its removal in PCL radar,' *IEE Proc. Radar, Sonar Navig.*, 2005, **152**, (3), pp. 174–178

[15] Tao, R., Zhang, W. Q., Chen, E. Q.: 'Two-stage method for joint time delay and Doppler shift estimation,' *IET Radar Sonar Navig.*, 2008, **2**, (1), pp. 71–77

[16] Auslander, L., Tolimieri, R.: 'Computing decimated finite cross-ambiguity functions,' *IEEE Trans. Acoust., Speech, Signal Process.*, 1988, **36**, (3), pp. 359–364

[17] Tolimieri, R., Winograd, S.: 'Computing the ambiguity surface,' *IEEE Trans. Acoust., Speech, Signal Process.*, 1985, **33**, (4), pp. 1239–1245

[18] Palmer, J., Palumbo, S., Summers, A., Merrett, D., Searle, S., Howard, S.: 'An overview of an illuminator of opportunity passive radar research project and its signal processing research directions,' *Digit. Signal Process.*, 2011, **21**, (5), pp. 593–599

[19] Feng, Y., Shan, T., Zhuo, Z. H., Tao, R.: 'The migration compensation methods for DTV based passive radar,' IEEE Radar Conf., Ottawa, Canada, 29 April–3 May 2013

[20] Lu, G. Y., Bao, Z.: 'Compensation of scatterer migration through resolution cell in inverse synthetic aperture radar imaging,' *IEE Proc. Radar, Sonar Navig.*, 2000, **147**, (2), pp. 80–85

[21] Malanowski, M., Kulpa, K., Olsen, K. E.: ‘Extending the integration time in DVB-T-based passive radar,’ Proc. European Radar Conference (EuRAD), Manchester, UK, 12–14 October 2011, pp. 190–193

[22] Rabiner, L. R., Schafer, R. W., Rader, C. M.: ‘The chirp z-transform algorithm,’ *IEEE Trans. Audio Electroacoust.*, 1969, **17**, (2), pp. 86–92

[23] Perry, R. P., Dipietro, R. C., Fante, R. L.: ‘SAR imaging of moving targets,’ *IEEE Trans. Aero. Electr. Syst.*, 1999, **35**, (1), pp. 188–200

[24] Wang, Y., Li, J. W., Chen, J., Xu, H. P., Sun, B.: ‘A parameter-adjusting polar format algorithm for extremely high squint SAR imaging,’ *IEEE Trans Geosci. Remote Sens.*, 2014, **52**, (1), pp. 640–650

[25] Cantini, C., La Rosa, E., Lo Re, A., Di Lallo, A.: ‘Passive coherent locator signal processor on IBM Cell broadband engine (Cell BE),’ IEEE Radar Conf., Pasadena, CA, USA, 4–8 May 2009

[26] John, M., Inggs, M., Petri, D.: ‘Real time processing of networked passive coherent location radar system,’ *Int. J. Electr. Telecomm.*, 2011, **57**, (3), pp. 363–368

[27] Pidanic, J., Nemec, Z., Dolecek, R., Bezousek, P.: ‘Computing of bistatic cross-ambiguity function on GPU,’ IEEE Int. Symp. on Ind. Electr. (ISIE2013), Taipei, China, 28–31 May 2013

[28] Bernaschi, M., Di Lallo, A., Fulcoli, R., Gallo, E., Timmoneri, L.: ‘Combined use of graphics processing unit (GPU) and Central Processing Unit (CPU) for passive radar signal & data elaboration,’ Proc. Int. Radar Symp. (IRS2011), Leipzig, Germany, 7–9 September 2011, pp. 315–320

[29] Bernaschi, M., Di Lallo, A., Farina, A., Fulcoli, R., Gallo, E., Timmoneri, L.: ‘Use of a graphics processing unit for passive radar signal and data processing,’ *IEEE Aero. Electr. Syst. Mag.*, 2012, **27**, (10), pp. 52–59

[30] Blankenship, P., Hofstetter, E. M..: ‘Digital pulse compression via fast convolution,’ *IEEE Trans. Acoust., Speech, Signal Process.*, 1975, **23**, (2), pp. 189–201

[31] Willis, N. J.: ‘Bistatic radar’ (SciTech Publishing, Raleigh, NC, 2005, 2nd edn.)

[32] Tao, R., Wu, H. Z., Shan, T.: ‘Direct-path suppression by spatial filtering in digital television terrestrial broadcasting-based passive radar,’ *IET Radar Sonar Navig.*, 2010, **4**, (6), pp. 791–805

[33] Tao, R., Gao, Z. W., Wang, Y.: ‘Side peaks interference suppression in DVB-T based passive radar,’ *IEEE Trans. Aero. Electr. Syst.*, 2012, **48**, (4), pp. 3610–3619

[34] Colone, F., O’hagan, D. W., Lombardo, P., Baker, C. J.: ‘A multistage processing algorithm for disturbance removal and target detection in passive bistatic radar,’ *IEEE Trans. Aero. Electr. Syst.*, 2009, **45**, (2), pp. 698–722

[35] Liu, S. H., Shan, T., Tao, R., Zhang, Y. D., Zhang, G., Zhang, F., Wang, Y.: ‘Sparse discrete fractional Fourier transform and its applications,’ *IEEE Trans. Signal Process.*, 2014, **62**, (24), pp. 6582–6595

[36] Levanon, N., Getz, B.: ‘Comparison between linear FM and phase-coded CW radar,’ *IEE Proc. Radar, Sonar Navig.*, 1994, **141**, (4), pp. 230–240

[37] Weiss, L. G.: ‘Wavelets and wideband correlation processing,’ *IEEE Signal Process. Mag.*, 1994, **11**, (1), pp. 13–32

[38] ‘cuFFT: CUDA Toolkit Documentation’, http://docs.nvidia.com/cuda/cufft/index.html, accessed April 2015

[39] Standardization Administration of the People’s Republic of China Framing Structure, Channel Coding and Modulation for Digital Television Terrestrial Broadcasting System, GB20600–2006, 2006

[40] Fasih, A., Hartley, T.: ‘GPU-accelerated synthetic aperture radar backprojection in CUDA,’ Proc. IEEE Radar Conf., Arlington, USA, 10–14 May 2010, pp. 1408–1413