RECURSIVE ZAK TRANSFORMS AND WEYL-HEISENBERG EXPANSIONS

Andrzej K. Brodzik

Scientific Software, 50-113 Cambridge Rd., Woburn, MA 01801

ABSTRACT

We develop algorithms for computing block-recursive Zak transforms and Weyl-Heisenberg expansions, which achieve $p/logL$ and $(logM+p)/(logN+logL+1)$ multiplicative complexity reduction, respectively, over direct computations, where $p' = pM$, and $N - p'$ is the number of overlapping samples in subsequent signal segments. For each transform we offer a choice of two algorithms that is based on two different implementations of the Zak transform of the time-evolving signal. These two algorithm classes exhibit typical trade-offs between computational complexity and memory requirements.

1. INTRODUCTION

Many applications require repetitive evaluation of frequency content of a time-evolving signal [1,9,12]. Since the evaluation is typically performed over a sequence of overlapping signal segments, one can improve efficiency of the computation by taking advantage of the fact that the frequency description of the $(i+1)$-th segment f^{i+1} can be expressed in terms of the frequency description of the i-th segment f^i plus a correction term that carries information about the new samples. For example, take $\mathbf{f}^i$ to be a DFT of f^i, i.e.

$$\mathbf{f}^i(b) = \sum_{a=0}^{N-1} f^i(a) e^{2\pi iab/N},$$

$\mathbf{f}^{i+1}$ to be a DFT of the subsequent signal segment f^{i+1}, and f^{i+1} to be delayed by $p' = 1$ data samples with respect to f^i. Then f^{i+1} can be computed recursively by the formula

$$\mathbf{f}^{i+1}(b) = [\mathbf{f}^i(b) + (f^{i+1}(N-1) - f^i(0))] e^{-2\pi ib/N},$$

where $f^i(0)$ is the first sample of f^i, and $f^{i+1}(N-1)$ is the last sample of f^{i+1}. The algorithm can be easily extended to the case when $p' > 1$, and to other transforms, such as the Hartley transform or the DCT [8].

Here, we are interested in recursive algorithms for computing time-frequency transforms, particularly the finite Zak transform and the Weyl-Heisenberg expansion. Since a time-frequency space signal is generally known by its values on a $M \times L$ lattice, where $N = LM$, it is natural to consider a recursive algorithm, that computes time-frequency transforms of signal segments overlapping by one or more blocks of $p' = M$ samples. Such an algorithm for the case of a single block overlap was recently proposed by Lou *et al* [10]. In this work, we remove the single block overlap restriction, and develop a collection of general multi-block algorithms, that allow to compute Zak transforms and Weyl-Heisenberg expansions for the entire range of p values, i.e. for $1 \le p < L$. Moreover, we propose an approach alternative to the one presented in [10] that yields a parallel set of algorithms that are more efficient under the conditions described in sections 3 and 4.

Denote by $\mathcal{C}^N$ the N-dimensional space of N-tuples of complex numbers, and set $N = LM$, where L and M are integers. Take $f^0 \in \mathcal{C}^N$ to be the i-th segment of f, and $f^{p'} \in \mathcal{C}^N$ to be the $(i+1)$-th segment of f, delayed with respect to f^0 by p' samples.

Take

$$p' = pM, \quad 1 \le p < L, \tag{1}$$

so that $M \le p' < LM$, and write f^0 and $f^{p'}$ as column vectors

$$f^0 = \begin{bmatrix} f_0^0 \\ \vdots \\ f_{L-1}^0 \end{bmatrix}, \quad f_r^0 \in \mathcal{C}^M, \quad 0 \le r < L, \tag{2}$$

$$f^{p'} = \begin{bmatrix} f_0^{p'} \\ \vdots \\ f_{L-1}^{p'} \end{bmatrix}, \quad f_r^{p'} \in \mathcal{C}^M, \quad 0 \le r < L, \tag{3}$$

such that

$$f_r^{p'} = f_{r+p}^0, \quad 0 \le r < L - p. \tag{4}$$

We will call $f_r^{p'}$ the r-th block of $f^{p'}$, and f_r^0 the r-th block of f^0. Given conditions (1) and (4), we will call $f^{p'}$ the p-block update of f^0.

To compute the Zak transform and the Weyl-Heisenberg expansion of $f^{p'}$ given by (1-4) we will develop the Block Recursive Zak Transform (BRZT) algorithms and the Block Recursive Weyl-Heisenberg Expansion (BRWHE) algorithms. The algorithms will be given in two flavors, based on a different order of the recursive Zak transform algorithm constituent operations: the block replacement and the block shift. We will show that the two approaches lead to different trade-offs in terms of computational complexity and memory requirements. In effect, a complete framework for computing recursive time-frequency transforms will be developed, allowing user to select the optimal algorithm for a given application (segment length N and number of overlapping samples $N - p'$) and processing environment (clock speed and memory size).

2. BASIC FORMULAS

Zak transform is a fundamental tool in time-frequency analysis. It is essential in developing algorithms for computing Weyl-Heisenberg expansions, ambiguity functions, and

Wigner distributions [11]. It also plays an important role in signal extrapolation and filtering [5]. It is intimately related to the Fourier transform and, via the real Zak transform, to the Hartley transform [3].

The finite Zak transform (ZT) F of a signal f is defined as [13]

$$F(a,b) = \sum_{r=0}^{L-1} f(a+rM)e^{2\pi irb/L}, \quad 0 \le a < M, \quad 0 \le b < L, \tag{5}$$

where a and b are usually interpreted as the time and frequency variables, respectively. A detailed treatment of ZT is given in [7]. Here, we are interested mainly in the shift/periodicity properties of ZT.

Take $p' = pM, \quad 1 \le p < L$, and $q' = qL, \quad 1 \le q < M$. The Zak transform $F(a,b)$ satisfies the following relations

$$F(a+p',b) = e^{-2\pi ipb/L}F(a,b), \tag{6a}$$

$$F(a,b+q') = F(a,b), \quad a,b \in Z. \tag{6b}$$

The frequency shifted transform is identical with the original transform, while the time shifted transform is obtained by multiplying all values of the original transform by a constant in time phase factor. It follows that F is N-periodic in each variable and is completely determined by its values $(a,b) \in [0,M) \times [0,L)$. The time shift relation (6a) is the basis of the recursive algorithms developed in this paper.

3. BLOCK-RECURSIVE ZAK TRANSFORM

Consider f^0 and $f^{p'}$ as in (1-4). The finite Zak transforms F^0 of f^0 and $F^{p'}$ of $f^{p'}$ are

$$F^0(a,b) = \sum_{r=0}^{L-1} f_r^0(a)e^{2\pi irb/L}, \tag{7}$$

and

$$F^{p'}(a,b) = \sum_{r=0}^{L-1} f_r^{p'}(a)e^{2\pi irb/L}. \tag{8}$$

Here and throughout the rest of the paper $(a,b) \in [0,M) \times [0,L)$, unless otherwise indicated. Suppose we want to compute $F^{p'}$, and F^0 is known. Since $L-p$ terms of r.h.s. of (7) and (8) involve same blocks, we can express $F^{p'}$ in terms of F^0 and p blocks of $f_r^{p'}$.

For example, we can write $F^{p'}$ as

$$F^{p'}(a,b) = \sum_{r=0}^{L-p-1} f_{r+p}^0(a)e^{2\pi irb/L} + \sum_{r=L-p}^{L-1} f_r^{p'}(a)e^{2\pi irb/L}. \tag{9}$$

Changing limits of both summations and separating their common factor, we have

$$F^{p'}(a,b) = e^{-2\pi ipb/L} \times [\sum_{r=p}^{L-1} f_r^0(a)e^{2\pi irb/L} + \sum_{r=0}^{p-1} f_{L-p+r}^{p'}(a)e^{2\pi irb/L}]. \tag{10}$$

Adding $\sum_{r=0}^{p-1} f_r^0(a)e^{2\pi irb/L}$ to the first term of (10) and subtracting it from the second term of (10) yields

$$F^{p'}(a,b) = e^{-2\pi ipb/L} \times [F^0(a,b) + \sum_{r=0}^{p-1}(f_{L-p+r}^{p'}(a) - f_r^0(a))e^{2\pi irb/L}]. \tag{11}$$

Denote by $H^{0,p'}$ the Zak transform of the difference between the last p blocks of $f^{p'}$ and the first p blocks of f^0, i.e.

$$H^{0,p'}(a,b) = \sum_{r=0}^{p-1}(f_{L-p+r}^{p'}(a) - f_r^0(a))e^{2\pi irb/L}. \tag{12}$$

Then the Zak transform $F^{p'}$ can be computed as the time shift of $K^{p'}(a,b)$

$$F^{p'}(a,b) = K^{p'}(a+p',b) = K^{p'}(a,b)e^{-2\pi ipb/L}, \tag{13}$$

where

$$K^{p'}(a,b) = F^0(a,b) + H^{0,p'}(a,b). \tag{14}$$

The algorithm to compute $F^{p'}$ via (13-14) proceeds as follows

BRZT1 algorithm

- Compute $H^{0,p'}(a,b)$.
  This operation requires $(p-1)(N-M)$ multiplications and pM additions.
- Compute $K^{p'}(a,b) = F^0(a,b) + H^{0,p'}(a,b)$.
  This operation requires N additions.
- Compute $K^{p'}(a+p',b)$.
  This operation requires $N-M$ multiplications.

The overall computational complexity of the BRZT1 algorithm is $p(N-M)$ multiplications and $N+pM$ additions.

Alternatively, we can write $F^{p'}$ as

$$F^{p'}(a,b) = F^0(a+p',b) + H^{0,p'}(a+p',b), \tag{15}$$

where

$$F^0(a+p',b) = F^0(a,b)e^{-2\pi ipb/L} \tag{16}$$

and

$$H^{0,p'}(a+p',b) = H^{0,p'}(a,b)e^{-2\pi ipb/L} = \sum_{r=0}^{p-1}(f_{L-p+r}^{p'}(a) - f_r^0(a))e^{2\pi i(r-p)b/L}. \tag{17}$$

The algorithm to compute $F^{p'}$ via (15-17) proceeds as follows

BRZT2 algorithm

- Compute $F^0(a+p',b)$.
  This operation requires $N-M$ multiplications.
- Compute $H^{0,p'}(a+p',b)$.
  This operation requires $p(N-M)$ multiplications and pM additions.
- Compute $F^{p'}(a,b) = F^0(a+p',b) + H^{0,p'}(a+p',b)$.
  This operation requires N additions.

The overall computational complexity of the BRZT2 algorithm is $(p+1)(N-M)$ multiplications, or $N-M$ more than the BRZT1 algorithm, and $N+pM$ additions. In terms of complex multiplications the BRZT2 algorithm is more efficient than the direct approach, providing $p+1 < NlogL/(N-M)$.

4. BLOCK-RECURSIVE WEYL-HEISENBERG EXPANSION

Consider f^0 and $f^{p'}$ as in (1-4). The Weyl-Heisenberg expansion (WHE) of f^0 is given by

$$c_{m,n}(F^0) = \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{F^0(a,b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)}. \quad (18)$$

Suppose (18) is known, and we want to compute the WHE of $f^{p'}$,

$$c_{m,n}(F^{p'}) = \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{F^{p'}(a,b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)}. \quad (19)$$

Comment: The quotients $F^0(a,b)/G(a,b)$ in (18) and $F^{p'}(a,b)/G(a,b)$ and (19) require that either $G(a,b) \neq 0$ for all $(a,b) \in [0,M) \times [0,L)$, or the zero sets of $F^0(a,b)$ and $F^{p'}(a,b)$ contain the zero set of $G(a,b)$. When neither of the above conditions is met, the critically-sampled WHE, does not exist. A detailed treatment of the impact of zeros of $G(a,b)$ on existence and construction of WHE is given in [2] and [6].

The direct algorithm to compute $c_{m,n}(F^{p'})$ proceeds as follows

WHE algorithm

- Compute $F^{p'}(a,b)$.
  This operation requires $NlogL$ multiplications and additions.
- Compute the quotient $F^{p'}(a,b)/G(a,b)$.
  This operation requires N multiplications.
- Compute $c_{m,n}(F^{p'})$ by taking a 2D DFT of $F^{p'}(a,b)/G(a,b)$.
  This operation requires $NlogN$ multiplications and additions.

The overall computational complexity of WHE algorithm is $N(logN + logL + 1)$ multiplications and $N(logN + logL)$ additions.

Consider f^0 and $f^{p'}$ as in (1-4). A more efficient algorithm to compute $c_{m,n}(F^{p'})$ is obtained by employing one of the formulas for computing the BRZT.

For example, set

$$c_{m,n}(K^{p'}) = \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{K^{p'}(a,b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)}, \quad (20)$$

where $K^{p'}(a,b)$ is as in (14). Then

$$\begin{aligned} c_{m,n}(F^{p'}) &= \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{K^{p'}(a+p',b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)} \\ &= \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{K^{p'}(a,b)}{G(a,b)} e^{-2\pi i(ma/M+(n+p)b/L)} \\ &= c_{m,n+p}(K^{p'}) \end{aligned} \quad (21)$$

The algorithm to compute $c_{m,n}(F^{p'})$ proceeds as follows

BRWHE1 algorithm

- Compute $F^0(a,b) = K^0(a+p',b)$, where $K^0(a,b)$ is given by the previous recursion.
  This operation requires $N-M$ multiplications.
- Compute $K^{p'}(a,b)$ via the BRZT1 algorithm.
  This operation requires $(p-1)(N-M)$ multiplications and $N+pM$ additions.
- Compute the quotient $K^{p'}(a,b)/G(a,b)$.
  This operation requires N multiplications.
- Compute $c_{m,n}(K^{p'})$ by taking a 2D DFT of $K^{p'}(a,b)/G(a,b)$.
  This operation requires $NlogN$ multiplications and additions.
- Compute $c_{m,n}(F^{p'})$ by shifting $c_{m,n}(K^{p'})$ in n by p.

The overall computational complexity of the BRWHE1 algorithm is $N(logN + p + 1) - pM$ multiplications and $N(logN+1)+pM$ additions. The algorithm requires storing G, or N data points.

Alternatively, utilizing the second realization of $F^{p'}$, we can obtain (19) by computing the WHE of each of the two terms in (15) separately. First, we compute the WHE of $F^0(a+p',b)$

$$\begin{aligned} & c_{m,n}[F^0(a+p',b)] \\ &= \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{F^0(a+p',b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)} \\ &= \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{F^0(a,b)}{G(a,b)} e^{-2\pi i(ma/M+(n+p)b/L)} \\ &= c_{m,n+p}(F^0). \end{aligned} \quad (22)$$

Next, we will compute the WHE of $H^{0,p'}(a,b)$

$$c_{m,n}(H^{0,p'}) = \sum_{a=0}^{M-1}\sum_{b=0}^{L-1} \frac{H^{0,p'}(a,b)}{G(a,b)} e^{-2\pi i(ma/M+nb/L)}. \quad (23)$$

Set

$$h_r^{0,p'}(a) = f_{L-p+r}^{p'}(a) - f_r^0(a), \quad 0 \leq r < p, \quad (24)$$

so that

$$H^{0,p'}(a,b) = \sum_{r=0}^{p-1} h_r^{0,p'}(a) e^{2\pi irb/L}. \quad (25)$$

Then (23) can be expressed as

$$\begin{aligned} & c_{m,n}(H^{0,p'}) \\ &= \sum_{a=0}^{M-1}\sum_{r=0}^{p-1} h_r^{0,p'}(a) e^{-2\pi ima/M} \sum_{b=0}^{L-1} \frac{1}{G(a,b)} e^{-2\pi i(n-r)b/L} \\ &= \sum_{a=0}^{M-1} [\sum_{r=0}^{p-1} h_r^{0,p'}(a) X_{r,n}(a)] e^{-2\pi ima/M}, \end{aligned} \quad (26)$$

where

$$X_{r,n}(a) = \sum_{b=0}^{L-1} \frac{1}{G(a,b)} e^{-2\pi i(n-r)b/L} \quad (27)$$

is computed off-line. Since the WHE of $H^{0,p'}(a+p',b)$,

$$c_{m,n+p}(H^{0,p'}) = \sum_{a=0}^{M-1}\sum_{b=0}^{L-1}\frac{H^{0,p'}(a+p',b)}{G(a,b)}e^{-2\pi i(ma/M+nb/L)}, \quad (28)$$

is given by a shift of $c_{m,n}(H^{0,p'})$ in n by p, the WHE of $F^{p'}(a,b)$ can be obtained by adding (22) and (28)

$$c_{m,n}(F^{p'}) = c_{m,n+p}(F^0) + c_{m,n+p}(H^{0,p'}). \quad (29)$$

The algorithm to compute $c_{m,n}(F^{p'})$ proceeds as follows

BRWHE2 algorithm

- Pre-compute $X_{r,n}(a)$.
  This operation requires $pN(logL+1)$ multiplications and $pNlogL$ additions.
- Compute $h_r^{0,p'}(a)$.
  This operation requires pM additions.
- Compute $Y_n(a) = \sum_{r=1}^{p} h_r^{0,p'}(a)X_{r,n}(a)$.
  This operation requires pN multiplications and $(p-1)N$ additions.
- Compute $c_{m,n}(H^{0,p'})$ by taking a L M-point 1D DFT's of $Y_n(a)$, $0 \le n < L$.
  This operation requires $NlogM$ multiplications and additions.
- Compute $c_{m,n+p}(H^{0,p'})$ by shifting $c_{m,n}(H^{0,p'})$ in n by p.
- Compute $c_{m,n+p}(F^0)$ by shifting $c_{m,n}(F^0)$ in n by p, where $c_{m,n}(F^0)$ is given by the previous recursion.
- Compute $c_{m,n}(F^{p'}) = c_{m,n+p}(F^0) + c_{m,n+p}(H^{0,p'})$.
  This operation requires N additions.

The overall computational complexity of the BRWHE2 algorithm is $N(logM+p)$ multiplications and $N(logM+p)+pM$ additions. The algorithm requires storing $X_{r,n}(a)$, or pN data points. For $p=1$ the BRWHE2 algorithm is identical with the procedure given by Lou *et al* [10].

The BRWHE1 algorithm is more efficient than the WHE algorithm for $p < NlogL/(N-M)$. The BRWHE2 algorithm is more efficient than the WHE algorithm for $p \le logN$, and it is always more efficient than the BRWHE1 algorithm. For example, for $p=1$ and $L=M$, the BRWHE1 algorithm requires about 2/3 multiplications required by the WHE algorithm, while the BRWHE2 algorithm requires about 1/3 multiplications required by the WHE algorithm, given that N is sufficiently large. However, the BRWHE2 algorithm requires more memory than the BRWHE1 algorithm. For example, for $p=5$ and $N=2^{20}$, the BRWHE1 algorithm requires N memory locations and $26N$ multiplications, while the BRWHE2 algorithm requires $5N$ memory locations and $15N$ multiplications. In effect, the BRWHE1 algorithm is more efficient in terms of computational complexity, while the BRWHE2 algorithm needs less memory allocation, has a simpler dataflow, and does not require pre-computations.

5. SUMMARY

In this work we have developed a collection of algorithms for computing block recursive Zak transforms and Weyl-Heisenberg expansions. The algorithms are based on two fundamental implementations of the Zak transform of the time-evolving signal. The first implementation is obtained by application of block replacement followed by block shift to the Zak transform of the initial sequence, and yields BRZT1 and BRWHE1 algorithms. The second implementation is obtained by reversing the order of operations, and yields BRZT2 and BRWHE2 algorithms.

In the case of the Weyl-Heisenberg expansion, the first approach is advantageous in terms of memory requirements and algorithm structure complexity, while the second block recursive approach is more efficient in terms of multiplicative and additive complexity. In the case of the Zak transform, the two block recursive implementations have similar computational complexity.

In future work we will explore the case of sample-recursive time-frequency transforms, i.e., the case when $p' = pM$, $1/M \le p \le 1$, to obtain a complete framework for computing recursive time-frequency transforms [4].

6. REFERENCES

[1] S. Albrecht and I. Cumming, "Application of the Momentary Fourier Transform to SIFFT SAR Processing", *Proceedings IEEE-SP Symposium on Time-Frequency and Time-Scale Analysis*, Philadelphia, pp 517-520, 1998.

[2] M. An, A.K. Brodzik, I. Gertner and R. Tolimieri, "Weyl-Heisenberg Systems and the Finite Zak Transform", in *Signal and Image Representation in Combined Spaces*, eds Y. Zeevi and R. Coifman, pp 3-22, Academic Press, San Diego, 1998.

[3] A.K. Brodzik, "Signal extrapolation in the real Zak space", submitted.

[4] A.K. Brodzik, "Recursive time-frequency transforms", submitted.

[5] A.K. Brodzik and R. Tolimieri, "Extrapolation of band-limited signals and the finite Zak transform", *Signal Processing*, Vol. 80, No. 3, pp 413-423, March 2000.

[6] A.K. Brodzik and R. Tolimieri, "The Computation of Weyl-Heisenberg Coefficients for Critically Sampled and Oversampled Signals", *Proceedings IEEE-SP Symposium on Time-Frequency and Time-Scale Analysis*, Philadelphia, pp 272-275, 1994.

[7] A.J.E.M. Janssen, "The Zak Transform: A Signal Transform for Sampled Time-Continuous Signals", *Philips J. Res.*, Vol. 43, pp 23-69, 1988.

[8] K. J. Ray Liu and C-T Chiu, "Unified Parallel Lattice Structure for Time-Recursive Discrete Cosine/Sine/Hartley Transforms", *IEEE Trans. Signal Processing*, Vol. 41, No. 3, pp 1357-1377, March 1993.

[9] P.-C. Lo and Y.-Y. Lee, "Real-time implementation of the moving FFT algorithm", *Signal Processing*, Vol. 79, No. 3, pp 251-259, 1999.

[10] C. Lou, S. Joshi and J.M. Morris, "Parallel lattice structure of block time-recursive generalized Gabor transforms", *Signal Processing*, Vol. 57, No. 2, pp 195-203, 1997.

[11] R. Tolimieri and M. An, "Time-Frequency Representations", Birkhauser, Boston, 1998.

[12] M. Unser, "Recursion in short-time signal analysis", *Signal Processing*, Vol. 5, No. 3, pp 229-240, 1983.

[13] J. Zak, "Finite Translations in Solid State Physics", *Phys. Rev. Lett.*, Vol. 19, pp 1385-1397, 1967.