The Order of Computation for Finite Discrete Gabor Transforms

Richard S. Orr, *Member, IEEE*

Abstract—**Equations for the continuous-parameter Gabor transform are presented and converted to finite discrete form suitable for digital computation. A comparative assessment of the computational complexity of several algorithms that execute the finite discrete equations is given, with results in the range $O(P^2)$ to $O(P \log_2 P)$, where P is the number of input data points being transformed. The most efficient of the reviewed methods, which uses the Zak transform as an operational calculus, performs Gabor analysis and synthesis transforms with complexity of the same order as a fast Fourier transform (FFT).**

I. Introduction

THOUGH Gabor representations promise to become more widespread in signal and image processing through judicious exploitation of their potential advantages relative to classical and short-time Fourier analysis [1], ultimate realization of these advantages will depend on the ability to make accurate and efficient digital computation of the transforms. In this paper we focus on the computational complexity of finite discrete algorithms for the forward and inverse Gabor transforms.

Section II introduces the basic formulation of the continuous-parameter Gabor transform, including a short discussion on the relationship of contemporary Gabor theory to that originated by Gabor [2]. Section III presents the various formulas for the both the forward (analysis) and inverse (synthesis) transforms as derived by three different techniques: 1) the method of biorthogonal functions; 2) the Zak transform; and 3) deconvolution of the sampled short-time Fourier transform (STFT). In order to map the Gabor formulas into constructs suitable for digital computation, we address, in Section IV, the problem of finding finite discrete replacements for them. A finite discrete version of each Section III formula is presented, with details of the proof referenced.

In Section V, the computational complexity of each finite discrete algorithm is evaluated, using the number of complex multiplies and divides as the unit of measure. It is shown that for various rearrangements of these formulas, the computational costs can differ significantly. Results are summarized in Section VI in a tabulation of the computational complexity of each algorithm.

Manuscript received May 7, 1991; revised September 1, 1991. This work was supported by the Advanced Research Projects Agency of the Department of Defense and monitored by the Air Force Office of Scientific Research under Contract F49620-90-C-0016.

The author is with Atlantic Aerospace Electronics Corporation, Greenbelt, MD 20770.

IEEE Log Number 9203357.

II. Gabor Representation

A. The Fundamental Formulas

A Gabor representation of a time function $f(t)$ is a series expansion of the following form:

$$f(t) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{-\infty} a_{m,n} w(t - nT) \exp(j2\pi mt/T). \tag{1}$$

The ingredients of this expansion, which we designate as the inverse, or *synthesis* transform, are 1) a set of complex weights, $\{a_{m,n}\}$, $m, n \in \mathbb{Z}$, called the *Gabor coefficients*, and 2) a set of time and frequency translates of a single function, $w(t)$, the *window*. The translates of $w(t)$ form the *Gabor basis* $\{w_{m,n}(t)\}$, as follows:

$$w_{m,n}(t) = w(t - nT) \exp(j2\pi mt/T); \qquad m, n \in \mathbb{Z}. \tag{2}$$

The grid of time-frequency points, $(t_n, f_m) = (nT, m/T)$, over which $w(t)$ is displaced in forming $\{w_{m,n}(t)\}$—called the *von Neumann* or *Gabor lattice*—defines a unit area cell of dimension $T \times (1/T)$ in the time-frequency plane. Stated another way, the time-frequency plane density of basis functions is one per unit area for any $T > 0$. This unit density is a distinguishing characteristic of Gabor representations, and it is in fact the most sparse that can give rise to complete expansions. In this regard, the Gabor expansion may be compared to its generalization, the Weyl-Heisenberg expansion, which can use any density of basis functions ≥ 1 [3].

In general, the $\{w_{m,n}(t)\}$ are nonorthogonal, and (1) is a nonorthogonal expansion, causing one to take some pains to assure accuracy and stability of algorithms computing the transformation. A Gabor expansion can be orthogonal, but this is not required or always desirable. A simple example in which the transformation is orthogonal is the case where $w(t)$ is a square pulse on $[0, T)$, in which case the transform provides Fourier series analysis on contiguous intervals. Conditions for orthogonality of the Gabor basis have recently been found in terms of the autoambiguity function of the window—see Tolimieri and

Orr [3]—and a general construction of orthogonal Gabor expansions has recently been developed by Coifman *et al.* [4].

B. Gabor's Original Expansion

It is important to place the Gabor expansion defined above in proper relation to the one originally proposed by Gabor, especially as it relates to computational matters. In his seminal work [2], Gabor's choice of window function was the Gaussian pulse, which we denote by $g(t)$, and define in notation different from Gabor's:

$$g(t) = \left(\frac{1}{\sigma}\right)^{1/2} \exp\left[-\frac{\pi}{2}\left(\frac{t}{\sigma}\right)^2\right] \tag{3}$$

where σ is the rms pulsewidth (see (6)). The normalization in (3) is that the window has unit energy, or $L^2(\mathbb{R})$ norm,

$$\|g\|^2 = \int_{-\infty}^{\infty} dt\, |g(t)|^2 = 1 \tag{4}$$

but this is not necessary for the expansion to hold. Gabor chose the Gaussian pulse for several advantageous properties. First of all, it is an eigenfunction of the Fourier transform operator, i.e., its Fourier transform is again Gaussian:

$$\begin{aligned} G(f) &= \int_{-\infty}^{\infty} dt\, g(t) \exp(-j2\pi ft) \\ &= (2\sigma)^{1/2} \exp[-2\pi(\sigma f)^2]. \end{aligned} \tag{5}$$

In addition, the Gaussian pulse is the unique function that achieves equality in the Heisenberg uncertainty relation. If one starts with the time and bandwidth equations for a unit energy signal,

$$\begin{aligned} (\Delta t)^2 &= \int_{-\infty}^{\infty} dt\, |tf(t)|^2 \\ (\Delta f)^2 &= \int_{-\infty}^{\infty} dv\, |vF(v)|^2 = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt \left|\frac{df(t)}{dt}\right|^2 \end{aligned} \tag{6}$$

and applies Schwarz' inequality to the product $(\Delta t \Delta f)^2$, $(\Delta t \Delta f)^2$ will be minimized when the two integrands in (6) are proportional:

$$|tf(t)| = \left|k \frac{df(t)}{dt}\right|. \tag{7}$$

The Gaussian is the unique solution to (7), yielding for any $\sigma > 0$ the minimum achievable product

$$\Delta f \Delta t = \frac{1}{4\pi}. \tag{8}$$

Gabor exploited the $\Delta f\, \Delta t$ product of the Gaussian function to minimize the area of what he called the "logon," or elementary cell in the time-frequency plane occupied by the window. Wanting to synthesize waveforms from time- and frequency-shifted Gaussians that contain the most "information" per unit area, he envisioned placing a grid on the time-frequency plane with an arbitrary time step Δt and frequency step $\Delta f = 1/\Delta t$, which defines a unit area cell, and placing in the (m, n)th cell one of his "elementary signals," a Gaussian pulse, having time and frequency shifts $n\Delta t$ and $m\Delta f$ units from the origin, respectively. He allowed a complex coefficient $a_{m,n}$ to multiply the (m, n)th pulse, identifying with the real and imaginary parts of the coefficient two quadrature components carrying one "datum" each. The resultant waveform given by the superposition of all these weighted cell components thus contains two items of data per unit cell, which he believed to be the maximum packing density of information onto the waveform.

Because of his synthesis viewpoint, Gabor was apparently not motivated to investigate the representation impact of the relationship between the cell dimensions and the structure of the signal being represented. He is not known to have published anything on the inverse problem, the analysis of a waveform into its coefficients, and it was not until 1980 that Bastianns first gave the inverse formula for the Gaussian case [5]. It was here, despite the earlier treatment by Auslander and Tolimieri [6], that it first became evident in the engineering literature that the Gaussian pulse, for all its other niceties, could lead to numerically troublesome expansions, characterized by Gabor coefficients whose magnitudes are not square summable. It was Bastiaans who proposed the generalization (1) in which an (almost) arbitrary function could replace the Gaussian in the window role. There is today an operational calculus of this general Gabor representation, mediated by the Zak transform and the theory of frames, that provides much of the mechanism required to gain insight into the computational stability issues. In this language, the problem with the Gaussian window is recognized as being due to the zero of its Zak transform. Extensive use of the Zak transform is made in arriving at the various formulations of the Gabor equations.

III. The Continuous Parameter Formulas

In (1) we have a synthesis formula that tells how to recover a function from its expansion coefficients. There is a corresponding need for analysis formulas that generate those coefficients from the data signal. We investigate three rather different looking analytic means to compute Gabor transforms, given the window function and the von Neumann grid: the biorthogonal method, the Zak transform method, and the deconvolution method. Each is discussed below in turn.

A. The Biorthogonal Method

In the biorthogonal method, one uses both the basis functions $\{w_{m,n}(t)\}$ and a related set of biorthogonal functions $\{b_{m,n}(t)\}$,

$$b_{m,n}(t) = b(t - nT) \exp(j2\pi mt/T); \qquad m, n \in \mathbb{Z} \tag{9}$$

generated from a function $b(t)$ having the property

$$\langle w_{m,n}|\, b_{p,q}\rangle = \int_{-\infty}^{+\infty} dt\, w(t - nT)b^*(t - qT) \cdot \exp\,[j2\pi(m - p)t/T] = \delta_{n-q}\delta_{m-p}. \tag{10}$$

If such $b(t)$ exists, (10) permits analysis of f into its coefficients by inner products of $f(t)$ and the $\{b_{m,n}(t)\}$ according to the formula

$$a_{m,n} = \langle f|b_{m,n}\rangle = \int_{-\infty}^{+\infty} dt\, f(t)\, b^*_{m,n}(t) \tag{11}$$

as shown by the following development:

$$\begin{aligned}\langle f|b_{m,n}\rangle &= \int_{-\infty}^{\infty} dt f(t)\, b^*_{m,n}(t) = \int_{-\infty}^{\infty} dt \sum_{p=-\infty}^{\infty}\sum_{q=-\infty}^{\infty} \cdot a_{p,q} w_{p,q}(t)\, b^*_{m,n}(t) \\ &= \sum_{p=-\infty}^{\infty}\sum_{q=-\infty}^{\infty} a_{p,q} \int_{-\infty}^{\infty} dt\, w_{p,q}(t)\, b^*_{m,n}(t) \\ &= \sum_{p=-\infty}^{\infty}\sum_{q=-\infty}^{\infty} a_{p,q}\delta_{m-p}\delta_{n-q} = a_{m,n}. \end{aligned} \tag{12}$$

Interchange of the integral and summations in (12) requires uniform convergence of the sum. This formulation suggests the interpretation of a Gabor coefficient engine as a bank of time- and Doppler-offset filters.

B. The Zak Transform Method

A second analysis formula is based upon the Zak transform, a time-frequency mapping given by

$$Z_f(v, \tau) = \sum_{k=-\infty}^{\infty} f(kT + \tau) \exp\,(-j2\pi kvT). \tag{13}$$

The interpretation of the Zak transform as a time-offset DFT will be useful in the sequel. Taking Zak transforms on both sides of (1) yields the following relation among the Zak transforms of f and w, and the Gabor coefficients [7], [8]:

$$Z_f(v, \tau) = Z_w(v, \tau) \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} a_{m,n} \cdot \exp\,[j2\pi(m\tau/T - nvT)]. \tag{14}$$

The $\{a_{m,n}\}$ are found formally by inverting the two-dimensional Fourier series on the right:

$$a_{m,n} = \int_0^T d\tau \int_0^{1/T} dv\, \frac{Z_f(v, \tau)}{Z_w(v, \tau)} \cdot \exp\,[j2\pi(-m\tau/T + nvT)]. \tag{15}$$

It is now clear what is required to determine Gabor coefficients via the Zak transform method. The integrand $Z_f(v, \tau)/Z_w(v, \tau)$ must have a Fourier series that converges to the periodic extension of Z_f/Z_w on $[0, 1/T) \times [0, T)$, which requires that Z_f/Z_w and its first partials be piecewise continuous over the integration region. If Z_w has a zero at some interior point and Z_f does not, Z_f/Z_w approaches $\pm\infty$ at that point and cannot be continuous in any interval containing it. It is also true that if w is an $L^2(\mathbb{R})$ function for which Z_w has a zero, the corresponding b will not be $L^2(\mathbb{R})$. All these hold for the Gaussian window. Another way to state the consequence of the zero is that there are unit norm functions in $L^2(\mathbb{R})$ for which the maximum magnitude of any Gabor coefficient can be arbitrarily small. The basis functions are on the borderline of being incomplete. When these conditions obtain, Riemann sum approximations to (15) may not provide numerically accurate formulas. Stabilization of this computational process is possible, but is not discussed in this paper.

A synthesis formula for f is evident in (13) using its interpretation as a time-offset DFT. In this inversion we first recover $Z_f(v, \tau)$ via (14) and compute its Fourier coefficients. The inversion is:

$$f(kT + \tau) = T \int_0^{1/T} dv\, Z_f(v, \tau) \exp\,(j2\pi kv\tau); \quad 0 \le \tau < T,\ k \in \mathbb{Z}. \tag{16}$$

C. The Deconvolution Methods

The third method exploits a relationship derived from (14) among the Gabor coefficients, the sampled short-time Fourier transform (STFT) of f, $\{\langle f|w_{m,n}\rangle\}$, and the Zak transform of the window. We also need the readily verified observation that the Zak transform is an isomorphism, preserving inner products to within a constant scale factor [8]:

$$\langle Z_f(v, \tau)|Z_g(v, \tau)\rangle = \frac{1}{T}\langle f(t)|\, g(t)\rangle \tag{17}$$

where the inner product of Zak transforms is taken to be

$$\langle Z_f(v, \tau)|Z_g(v, \tau)\rangle = \int_0^{1/T} dv \int_0^T d\tau\, Z_f(v, \tau)Z^*_g(v, \tau). \tag{18}$$

Multiplying both sides of (14) by $Z^*_w(v, \tau)$,

$$Z_f(v, \tau)Z^*_w(v, \tau) = |Z_w(v, \tau)|^2 \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} a_{m,n} \cdot \exp\,[j2\pi(m\tau/T - nvT)] \tag{19}$$

and computing the Fourier coefficients of both expressions reduces (19), after simplification, to the convolution [7]

$$\langle f|w_{m,n}\rangle = \sum_{p=-\infty}^{\infty}\sum_{q=-\infty}^{\infty} a_{p,q}\langle w|w_{m-p,n-q}\rangle. \tag{20}$$

As an aside, we note that (20) is the root of a familiar comparison between a Gabor representation and a sam-

pled short-time Fourier transform of some function f, when each uses window w. The sampled STFT is represented as a "blurred" version of the Gabor coefficients through the two-dimensional convolution with a kernel that is the sampled ambiguity function of the window function. To the extent that a window function chosen for spectral decomposition is as smooth and locally concentrated as possible, within the limits imposed by the Balian–Low theorem [9], its ambiguity function resembles the point spread function (impulse response) of a well-designed optical system in which slight input image blurring occurs due to finite aperture.

Extraction of the Gabor coefficients according to (20) requires deconvolution of the sampled STFT. After Fourier analysis of both sides of (19), we find that the deconvolution can be carried out in the transform domain:

$$a_{m,n} = \int_0^T d\tau \int_0^{1/T} dv \, \frac{1}{|Z_w(v,\tau)|^2} \sum_{r=-\infty}^{\infty} \sum_{s=-\infty}^{\infty} \langle f | w_{r,s} \rangle \cdot \exp\{j2\pi[(r-m)\tau/T - (s-n)vT]\}. \quad (21)$$

Thus the algorithm, given the sampled STFT, is to take its DFT, divide by $|Z_w(v,\tau)|^2$, and extract the Gabor coefficients by the inverse 2-D DFT of the ratio. Recovery of f follows directly once the Gabor coefficients are obtained.

Upon first look the deconvolution algorithm may seem unnecessarily complicated. It is useful, however, if one wants to digitally process, by Gabor means, data that has originally been taken in analog form as an STFT. The deconvolution formula also yields the generalized inverse for Gabor expansions, that is, the coefficients that lead to the minimum L^2 norm error between a function (not in the space spanned by the Gabor basis) and its Gabor representation.

These three techniques, the method of biorthogonal functions, Zak transforms, and deconvolution of the sampled STFT, exhaust the formal structures under consideration. In the following section we convert the equations of each method to formulations that are amenable to digital computation.

IV. The Finite Discrete Equations

A. The Role of Periodization and Sampling

There are several possible approaches to making the Gabor equations discrete and finite. The most straightforward is to truncate the time functions to compact support and sample them at a rate capturing their significant behavior. A more elegant process that better preserves some of the relationships between the discrete Gabor equations and the original transforms accomplishes the time truncation by periodization before sampling. The equations for the latter are summarized in Orr [13]. From the point of view of computation, it makes little difference which approach is selected, since the resulting equations have common computational requirements. We adopt the periodized, sampled approach here as the model for computation analysis. By periodizing and sampling $f(t)$, we reduce the Gabor transform of f to a discrete linear mapping that relates the samples of the periodized f (having period P samples) to a periodic $M \times N$ array of Gabor coefficients, where M is the number of Gabor frequency cells, N the number of time cells and $MN = P$. We find that the samples of the window function are also periodized as a result of the processes imposed on the signal.

Many of the results presented in this section have been summarized without proof in [11].

B. The Discrete Biorthogonal Equations

All the equations in which either the Gabor coefficients or the original signal f are expressed directly in terms of the window function or its biorthogonal counterpart are considered to be elements of the biorthogonal method. We first periodize f

$$f_{NT}(t) = \sum_{k=-\infty}^{\infty} f(t + kNT) \quad (22)$$

to a multiple of the Gabor time step T, and sample $f_{NT}(t)$ at the Mth harmonic of $1/T$, i.e., $t_p = pT/M$, yielding

$$f_p^{(P)} \equiv f_{NT}(pT/M) = \sum_{k=-\infty}^{+\infty} f(pT/M + kNT), \quad 0 \le p \le P-1. \quad (23)$$

The notation $f_p^{(P)}$ will be used throughout; it represents the pth sample of f periodized to period P samples. By formally inserting the Gabor expansion of $f(t)$ into (23) and regrouping the terms, we eventually find

$$f_p^{(P)} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} a_{m,n}^{(M,N)} w_{p-Mn}^{(P)} \exp(j2\pi mp/M) \quad (24)$$

where

$$w_p^{(P)} = \sum_{k=-\infty}^{\infty} w_{p+Pk} \quad (25)$$

and

$$a_{m,n}^{(M,N)} = \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} a_{m+Mp,n-Nq}. \quad (26)$$

Equation (24) can be written more compactly by defining

$$w_{m,p-Mn}^{(P)} = w_{p-Mn}^{(P)} \exp(j2\pi mp/M) \quad (27)$$

to yield

$$f_p^{(P)} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} a_{m,n}^{(M,N)} w_{m,p-Mn}^{(P)}. \quad (28)$$

Equation (28) shows exactly how the discrete finite representation is related to the original continuous-parameter Gabor formulation: *The set of periodized samples of f is expressed as a weighted sum of the periodized samples of the window function w, where the weighting coefficients are the doubly periodized Gabor coefficients of f.* In making computations with the discrete transform, it is gen-

erally not possible to "back out" the initial Gabor coefficients $\{\alpha_{m,n}\}$ from the periodic array $\{a_{m,n}^{(M,N)}\}$, or to reconstruct the aperiodic $f(t)$ from the $\{f_p^{(P)}\}$, due to the aliasing that accompanies periodization. But, as in any other problem using sampled data, one can use the controllable parameters, i.e., the periodization interval (N) and the sampling rate (M), to make the sampled and continuous parameter versions closely approximate.

The conditions under which the $\{a_{m,n}^{(M,N)}\}$ computed from (28) should closely resemble the $\{\alpha_{m,n}\}$ of (1) are that 1) any frequency aliasing induced into f or the original $\{\alpha_{m,n}\}$ by sampling is made negligible, which necessitates choosing the sampling rate M/T to sufficiently exceed the bandwidth of f; and 2) f, w and the lattice are such that at most the first N Gabor coefficients in (1) are significant at any frequency up to M/T Hz, that is, the interval of periodization should be great enough that within it, the periodized f and w closely resemble their aperiodic versions. When we think these conditions are satisfied, we can replace (24) by (29) below, a sampled, truncated approximation to (1), and proceed as though (29), in which periodization is ignored, were an equality, and neglect the differences between it and (24):

$$f(pT/M) \approx \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} a_{m,n} w[(p - Mn)T/M] \cdot \exp(j2\pi mp/M). \quad (29)$$

If, however, one is dealing with a window function whose Zak transform vanishes, or nearly so, then the periodized Gabor coefficient sum (26) will be slow to converge, and the correspondence between the periodic and aperiodic coefficients can be anticipated to be poor.

Given the development above, it should not be surprising that (11), which expresses the Gabor coefficients as inner products of f and the biorthogonal functions, is transformed by periodization and sampling into the following expression:

$$a_{m,n}^{(M,N)} = \frac{T}{M}\sum_{k=0}^{P-1} f_k^{(P)} b_{m,n}^{*(P)}(k) = \frac{T}{M}\langle f^{(P)} | b_{m,n}^{(P)}\rangle \quad (30)$$

where

$$b_{m,n}^{(P)}(k) = \sum_{q=-\infty}^{\infty} b_{m,n}\left(\frac{kT}{M} + qNT\right) \quad (31)$$

is the periodized and sampled biorthogonal function.

C. The Discrete Zak Transform Equations

In order to find a finite discrete version of (14), which relates the Zak transforms of the signal and window to the Gabor coefficients, sampling and periodization must be applied. One finds that

$$Z_f\left(\frac{q}{NT}, \frac{pT}{M}\right) = Z_w\left(\frac{q}{NT}, \frac{pT}{M}\right)\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} a_{m,n}^{(M,N)} \cdot \exp[j2\pi(mp/M - nq/N)] \quad (32)$$

from which the solution for the coefficients follows directly:

$$a_{m,n}^{(M,N)} = \frac{1}{MN}\sum_{p=0}^{M-1}\sum_{q=0}^{N-1}\frac{Z_f(q/NT, pT/M)}{Z_w(q/NT, pT/M)} \cdot \exp[-j2\pi(mp/M - nq/N)] \quad (33)$$

when Z_w does not vanish on the von Neumann lattice.

We shall also need the finite discrete analog of the recovery of f from its Zak transform (13), which is

$$f_p^{(P)} = \frac{1}{N}\sum_{q=0}^{N-1} Z_f\left(\frac{q}{NT}, \frac{pT}{M}\right)\exp(j2\pi qp/N). \quad (34)$$

That is, the finite DFT of the sampled Zak transform is the sampled, periodized f.

D. The Discrete Sampled STFT Equations

The basis for this approach is (32). Multiplying both sides by $Z_w^*(q/NT, pT/M)$ and applying the convolution theorem, (32) becomes the following analog of (20):

$$\langle f^{(P)} | w_{m,n}^{(P)}\rangle = \sum_{p=0}^{M-1}\sum_{q=0}^{N-1} a_{m,n}^{(M,N)}\langle w^{(P)} | w_{m-p,n-q}^{(P)}\rangle \quad (35)$$

where the finite discrete inner product is

$$\langle f^{(P)} | g^{(P)}\rangle = \sum_{p=0}^{P-1} f_p^{(P)}(g_p^{(P)}) \quad (36)$$

and the convolution is circular with period MN. As in the continuous parameter case, the Gabor coefficients can be deconvolved from (35) in the frequency domain:

$$a_{m,n}^{(M,N)} = \sum_{p=0}^{M-1}\sum_{q=0}^{N-1}\frac{1}{|Z_w(q/NT, pT/M)|^2}\sum_{r=-\infty}^{\infty}\sum_{s=-\infty}^{\infty} \cdot \langle f^{(P)} | w_{r,s}^{(P)}\rangle \exp\{j2\pi[p(r - m)/M - q(s - n)/N)]\}. \quad (37)$$

One recovers $f_p^{(P)}$ using (28).

E. Notation

In the following we write $N = P^{\alpha}$ and $M = P^{1-\alpha}$ to emphasize dependence of the results upon both the size of the data set (P) and the shape of the time-frequency lattice (α). Superscript notation pertaining to periodization and sampling is henceforth dropped, since for computation complexity purposes it does not matter whether finitization of the representation has been achieved by periodization or truncation.

V. Algorithms and Complexity Analysis

Explicit algorithms for each formula in Section IV are presented next. We evaluate algorithm complexity by counting the number of complex multiplies and divisions required. A P-point, 1- or 2-D FFT will be said to require $P \log_2 P$ complex multiplies, as in the case where P is a power of 2 [12]. We will not take pains to more finely

tailor the coefficient of $P \log_2 P$ to specific algorithms, or account for differences in the number of real multiplies and/or divides needed to implement the complex multiplies and divides. As a result, slight revisions of the formulas presented here are possible. For each method, the coefficient, or *analysis*, algorithm is presented prior to the reconstruction, or *synthesis*, algorithm. Methods are denoted by the letters B (biorthogonal), Z (Zak), or D (deconvolution), and the total computation for an algorithm is given by $C(*|*)$, where the first argument is the method and the second is A or S (analysis or synthesis).

A. Method B1—Multiply and Add

This is the most naive model, presented for reference only as a worst case. It corresponds to the pre-FFT method of computing finite Fourier series.

1) Analysis: To obtain the $\{a_{m,n}\}$ via (30) by forming and then adding the summand products requires only one multiply per term, assuming that the array $\{b_{m,0}(k)\}$ is precomputed. There are P terms in the sum for each (m, n) and P values of (m, n), making the computation

$$C(\text{B1}|\text{A}) = P^2. \tag{38}$$

2) Synthesis: Reconstruction in this case is via (28). For each triplet (p, m, n) there is one multiply, and there are $PMN = P^2$ such triplets. The result is again

$$C(\text{B1}|\text{S}) = P^2. \tag{39}$$

B. Method B2—FFT-Based

1) Analysis: Equation (11) is actually a DFT that can be computed more efficiently by FFT. In this case we factor the biorthogonal terms into their time- and frequency-translation components and first compute the signal-biorthogonal products

$$h_{p,n} = f_p b^*_{p-Mn}; \qquad 0 \le p \le P-1, \quad 0 \le n \le N-1. \tag{40}$$

There are $NP = P^{1+\alpha}$ multiplies required to compute all the $\{h_{p,n}\}$. Compute these first and follow with a P-point FFT for each n:

$$a_{m,n} = \sum_{p=0}^{P-1} h_{p,n} \exp(j2\pi mp/P). \tag{41}$$

There are N FFT's requiring $P^{1+\alpha} \log_2 P$ operations, making the total count

$$C(\text{B2}|\text{A}) = P^{1-\alpha}(\log_2 P + 1). \tag{42}$$

Method B2 actually produces more than is required, because it overresolves in frequency, producing data at the Fourier resolution limit of $1/NT$ Hz, not the Gabor-cell $1/T$ Hz resolution. Each FFT yields P output points, of which only every Nth point is desired Gabor coefficient. Despite this, the algorithm is a considerable improvement over the naive algorithm, B1.

2) Synthesis: There is no directly corresponding result for the synthesis algorithm, since it is cast as a double rather than single summation. This, however, leads to an efficient method developed from (24) by factoring the triple product of the coefficient, window and complex exponential such that the summations over m are performed first. Write (24) as

$$f_p = \sum_{n=0}^{N-1} w_{p-Mn} \sum_{m=0}^{M-1} a_{m,n} \exp(j2\pi mp/M) \tag{43}$$

and execute the sums over m as FFT's. Each M-point FFT accomplishes the sum over m for one value of n and M values of p. Because the sum is periodic in m with period M, each FFT actually yields the values for all p. Thus the effort required to obtain all the sums on m is $N[M \log_2 M] = (1-\alpha)P \log_2 P$ multiplies. Add to these computations N multiplies for each value of P, another $P^{1+\alpha}$ operation, for a total of

$$C(\text{B2}|\text{S}) = P[P^\alpha + (1-\alpha)\log_2 P]. \tag{44}$$

Comparing (44) to (42), we find the synthesis algorithm to be better. The asymmetry in the answers suggests that the analysis algorithm has further room for improvement. Below, the approach that led to (44) is applied to the analysis algorithm with good results.

C. Method B3—Expurgated FFT-Based

1) Analysis: The B2 analysis algorithm can be improved by reorganizing (30) as a double summation, recasting the computation in lower dimensional FFT's; for details, see [14]. The result is

$$a_{m,n} = \sum_{r=0}^{M-1} \left[\sum_{q=0}^{N-1} h_{qM-r,n} \right] \exp(j2\pi mr/M). \tag{45}$$

In (45), N multiplies are needed for each instance of a sum on q; since there are MN values of (r, n), all sums are obtained in $N^2M = P^{1+\alpha}$ multiplies. To complete the calculation, N M-point FFT's accomplish the sum over r, i.e., $NM \log_2 M$ steps for the FFT portion of the computation. The multiplies and the FFT's are sequential, summing to

$$C(\text{B3}|\text{A}) = P[P^\alpha + (1-\alpha)\log_2 P]. \tag{46}$$

The computation for this algorithm is exactly the same as for synthesis via B2, and in fact the two carry out exactly the same set of operations. The FFT portion of this computation is smaller than a full FFT by a coefficient of $(1-\alpha)$, but the dominant term for large P will generally be $P^{1+\alpha}$, resulting from the point-by-point multiplications of data and the biorthogonal functions.

2) Synthesis: The synthesis algorithm for this method is that given in B2. When compared step by step to the B3 analysis algorithm, one finds that each exactly inverts the other. The author is unaware of any algorithm using the biorthogonal functions that has lower computational complexity.

D. Method B4—Matrix

1) Analysis: Balart [15] has formulated the discrete Gabor problem in matrix terms, expressing (29) in the

form $f = Ax$, where f is the P-vector of signal points, x is a P-vector of Gabor coefficients ordered with time as the slower varying index, and A the matrix

$$A = \begin{bmatrix} A_{00} & 0 & \cdots & & 0 \\ A_{10} & A_{11} & & & \\ \vdots & A_{21} & & & \\ A_{l0} & & \ddots & & \\ \vdots & \ddots & & A_{N-2,N-2} & 0 \\ A_{N-1,0} & \cdots & A_{N-1,n} \cdots & A_{N-1,N-2} & A_{N-1,N-1} \end{bmatrix}. \tag{47}$$

Each subblock of A is an $M \times M$ matrix that is a product of an $M \times M$ diagonal matrix containing values of the window function and the $M \times M$ Fourier rotation matrix. The system is solved by inverting A taking advantage of its special structure. In [14], [15] it is shown that this can be accomplished in

$$C(\text{B4}|\text{A}) = P[P^{\alpha} + (1 - \alpha) \log_2 P]. \tag{48}$$

2) Synthesis: The synthesis algorithm works by using the inverse set of operations, and has then the same complexity:

$$C(\text{B4}|\text{S}) = P[P^{\alpha} + (1 - \alpha) \log_2 P]. \tag{49}$$

The synthesis simply accomplishes the multiplication of the Gabor coefficients and the windows in an efficient way, and requires the same amount of computation as the best of the other biorthogonal-based methods.

E. Method Z—Zak Transform

1) Analysis: This algorithm has recently been exposited by Auslander *et al.* [7]. It uses the finite discrete Zak transform implied by (34) to implement (33). In the first step, $Z_f(\bullet, pT/M)$ is computed as the N-point FFT

$$Z_f(q/NT, pT/M) = \sum_{k=0}^{N-1} f_{p+Mk} \exp(-j2\pi kq/N). \tag{50}$$

Transforms are taken for each of the M values of p, permitting the set of Zak transforms to be computed in $MN \log_2 N = \alpha P \log_2 P$ multiplies.

The coefficient algorithm (33) first computes the above finite discrete Zak transform of the signal, Z_f, and divides by Z_w. The division takes P operations, since the Z_w array can be precomputed. The final step is a 2-D $M \times N$ FFT on the ratio array, taking $P \log_2 P$ multiplies. The total operation count is thus

$$C(\text{Z}|\text{A}) = P[(1 + \alpha) \log_2 P + 1]. \tag{51}$$

For the first time we see an algorithm that will run at the same order of speed as the FFT of comparable size.

2) Synthesis: This algorithm was first suggested to the author by M. Wengrovitz of Atlantic Aerospace Electronics Corporation. It first recovers Z_f according to (32) by a 2-D Fourier inversion of the Gabor coefficients ($P \log_2 P$ multiplies), followed by P point-by-point multiplications by Z_w. These steps are completed in $P(\log_2 P + 1)$ operations. Extraction of f from its recovered Zak transform is then accomplished according to (34). This recovery takes M N-point FFT's, which was assessed at $\alpha P \log_2 P$ multiplies above. These operations add to total computational cost

$$C(\text{Z}|\text{S}) = P[(1 + \alpha) \log_2 P + 1]. \tag{52}$$

As was the case in methods B2 and B3, the best synthesis algorithm inverts the analysis by performing the identical operations in the reverse order.

F. Method D1—Deconvolution of the STFT

In this algorithm we assume that the input is the sampled STFT, from which either the function f or its Gabor coefficients can be obtained.

1) Analysis: We wish to carry through the computations indicated in (37) to get the Gabor coefficients. If we assume that the sampled STFT is given, the first step is to compute its FFT, which takes $P \log_2 P$ steps. Division by the sampled $|Z_w|^2$ adds P more steps, and the inverse FFT another $P \log_2 P$, for a total of

$$C(\text{D1}|\text{A}) = P(2 \log_2 P + 1) \tag{53}$$

operations, where D1|A stands for the analysis computation excluding the evaluation of the sampled STFT's.

2) Synthesis: To recover f from its sampled STFT, we write the inner product of f and a basis function in terms of their Zak transforms:

$$\langle f^{(P)} | w_{m,n}^{(P)} \rangle = \frac{1}{N} \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} Z_f(q/NT, pT/M) \cdot Z_w^*(q/NT, pT/M) \cdot \exp[-j2\pi(mp/M - nq/N)] \tag{54}$$

and implement it as a Fourier transform. In fact, (54) is no more than (33), the right-hand side of which has the roles of b and w interchanged thus:

$$\frac{1}{Z_w(v, \tau)} \equiv TZ_b^*(v, \tau) \Rightarrow TZ_w^*(v, \tau). \tag{55}$$

Calculation of Z_f from (54) first requires a 2-D P-point FFT, which yields the $Z_f Z_w^*$ product array, followed by P divisions by Z_w^*. The map to f takes $\alpha P \log_2 P$ steps (see the method Z discussion), resulting in a total computation of

$$C(\text{D1}|\text{S}) = P[(2 + \alpha) \log_2 P + 1]. \tag{56}$$

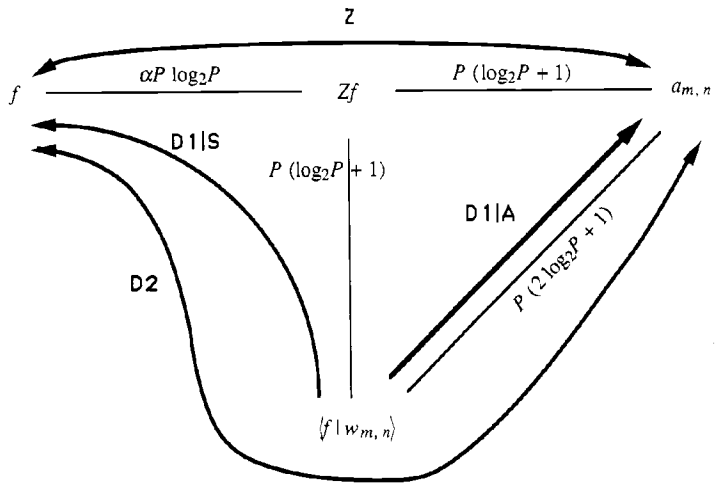


Fig. 1. Relationships among the fundamental quantities in the Zak-based methods.

The Method D1 results are asymmetric because the starting point is intermediate between the function and its Gabor coefficients, although, as we can see, it is closer to the coefficients in terms of required operations.

G. Method D2—STFT-Based

Here we consider the computations required in passing through the sampled STFT en route between the function and its Gabor coefficients.

1) Analysis: We need to start with the complexity of computing the STFT from f. Since the computation is an inner product, it could be executed in a variety of ways. Done according to the biorthogonal prescription in (30), where we replace $b_{m,n}^{(P)}$ with $w_{m,n}^{(P)}$, the calculation can be organized as in method B3, equation (45), and accomplished in $P[P^{\alpha} + (1 - \alpha) \log_2 P]$ steps. But the Zak transform provides a faster method, reversing, in effect, the flow evaluated immediately above. First compute Z_f in $\alpha P \log_2 P$ multiplies. Then follow (54); construct the $Z_f Z_w^*$ products and follow with an FFT, taking $P(\log_2 P + 1)$ multiplies. The final step is the map to the Gabor coefficients, shown in (53) to require $P(2 \log_2 P + 1)$ operations. The sum total, then is

$$C(\text{D2}|\text{A}) = P[(3 + \alpha) \log_2 P + 2]. \quad (57)$$

2) Synthesis: Obtaining f from the Gabor coefficients can be done by reversing the analysis argument. Consequently the computation is the same:

$$C(\text{D2}|\text{A}) = P[(3 + \alpha) \log_2 P + 2]. \quad (58)$$

Methods Z and D, both dependent upon the Zak transform, are tightly intertwined. In Fig. 1 we illustrate their interrelationships via a "state" diagram that shows the direct computational paths connecting the function, its Zak transform, its sampled STFT, and the Gabor coefficients. Each path is labeled with its computational complexity, and the paths corresponding to the D1 and D2 algorithms are marked. This graphic should make it easier for the reader to sort out the operations pertaining to each algorithm.

VI. Summary

In this paper we have demonstrated several ways to calculate finite discrete Gabor transforms. Table I summarizes the complexity results. For reference, the first entry is the standard complexity of the FFT on a P-point dataset. Method B1 represents the pre-FFT approach and would only be used today in very special circumstances. Method B2 illustrates that straightforward use of FFT's to replace the DFT's improves upon B1, but achieves an asymmetric result in which the analysis algorithm is of greater complexity than the synthesis algorithm. This defect suggests that there is further room for improvement, and it is remedied in both an algorithm (B3) that implements the result as a single stage of decimation in frequency and one (B4) that treats the Gabor expansion as a linear system with a fast inverse resulting from the highly structured form of the defining matrix. Algorithms B3 and B4 appear to have the minimal achievable complexity of any that uses straightforward application of the FFT in a biorthogonal method.

None of the biorthogonal-based method achieve the $O(P \log_2 P)$ complexity of the FFT. By contrast, both methods Z and D, both relying on the Zak transform as an operational calculus, do achieve $O(P \log_2 P)$ with coefficients no greater than 2 and 4, respectively, demonstrating that Gabor analysis can be carried out with the order of efficiency of an FFT.

Each of the algorithms discussed is subject to tailored enhancements under appropriate restrictions on the signal or window. For the most part these modifications result in coefficient improvement only, and are not discussed in this paper.

TABLE I
COMPARISON OF GABOR ALGORITHM COMPLEXITY AND THE FFT

Method	Complexity (Analysis)	Complexity (Synthesis)
Standard FFT	$P \log_2 P$	$P \log_2 P$
	Gabor Methods	
B1. Multiply and add	P^2	P^2
B2. FFT-based	$P^{1+\alpha}(\log_2 P + 1)$	$P[P^\alpha + (1-\alpha)\log_2 P]$
B3. Expurgated FFT-based	$P[P^\alpha + (1-\alpha)\log_2 P]$	$P[P^\alpha + (1-\alpha)\log_2 P]$
B4. Matrix	$P[P^\alpha + (1-\alpha)\log_2 P]$	$P[P^\alpha + (1-\alpha)\log_2 P]$
Z. Zak transform	$P[(1+\alpha)\log_2 P + 1]$	$P[(1+\alpha)\log_2 P + 1]$
D1. Deconvolution	$P[2\log_2 P + 1]$	$P[(2+\alpha)\log_2 P + 1]$
D2. STFT-based	$P[(3+\alpha)\log_2 P + 2]$	$P[(3+\alpha)\log_2 P + 2]$

$$0 \le \alpha \equiv \frac{\log_2 N}{\log_2 P} \le 1$$

P total number of sampled data points,

M number of Gabor frequency resolution cells,

N number of Gabor time resolution cells,

P MN.

ACKNOWLEDGMENT

The author would like to thank the anonymous referees of this paper, whose comments have contributed to a more compact and better focused work.

REFERENCES

[1] L. Auslander, C. Buffalano, R. Orr, and R. Tolimieri, "A comparison of the Gabor and short-time Fourier transforms for signal detection and feature extraction in noisy environments," *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 1348, pp. 230-247, Nov. 1990.

[2] D. Gabor, "Theory of communication," *J. Inst. Elect. Eng.*, vol. 93, pp. 429-459, 1946.

[3] R. Tolimieri and R. Orr, "Poisson summation, the ambiguity function, and the theory of Weyl-Heisenberg frames," *IEEE Trans. Inform. Theory*, accepted for publication pending revision.

[4] R. Coifman, Y. Meyer, and V. Wickerhauser, *Wavelet Analysis and Signal Processing*. New Haven, CT: Yale University 1991 (preprint).

[5] M. J. Bastiaans, "A sampling theorem for the complex spectrogram, and Gabor's expansion of a signal in Gaussian elementary signals," *Opt. Eng.*, vol. 20, no. 4, pp. 594-598, July/Aug. 1981.

[6] L. Auslander and R. Tolimieri, *Abelian Harmonic Analysis, Theta Functions and Function Algebras on a Nilmanifold*. New York: Springer-Verlag, 1975.

[7] L. Auslander, I. Gertner, and R. Tolimieri, "The discrete Zak transform application to time-frequency analysis and synthesis of nonstationary signals," *IEEE Trans. Signal Processing*, vol. 39, no. 4, pps. 825-835, Apr. 1991.

[8] A. J. M. Janssen, "The Zak transform: A signal transform for sampled time-continuous signals," *Phillips J. Res.*, vol. 43, pp. 23-69, 1988.

[9] J. Benedetto, C. Heil, and D. Walnut, *Remarks on the Proof of the Balian-Low Theorem*, MITRE Corp., 1990.

[10] W. Rudin, *Real and Complex Analysis*. New York: McGraw-Hill, 1974, p. 210.

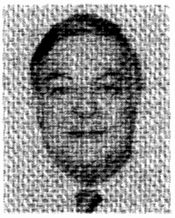
[11] R. S. Orr, "Computational assessment of Gabor representations," in *Proc. IEEE ICASSP '91*, vol. 3, May 1991, pp. 2217-2220.

[12] A. V. Oppenheim, and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

[13] R. S. Orr, "Derivation of the finite discrete Gabor transform by periodization and sampling," *Signal Processing*, accepted for publication.

[14] R. S. Orr and R. Balart, "Fast computation of Gabor coefficients using window and biorthogonal function-based methods," Atlantic Aerospace Electronics Corporation intern. memo.

[15] R. Balart, "Matrix reformulation of the Gabor transform," *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 1565, July 1991; also in *Opt. Eng.*, vol. 31, no. 6, pp. 1235-1242, June 1992.



Richard S. Orr (M'92) was born November 18, 1940 in Pittsburgh, PA. He received the S.B., S.M., E.E., and Ph.D. degrees from the Massachusetts Institute of Technology, in 1962, 1963, 1970, and 1973, respectively, all from the Department of Electrical Engineering.

Since January 1988 he has been at the Atlantic Aerospace Electronics Corporation (AAEC), Greenbelt, MD, where he is currently Director of the Advanced Concepts Group in the Processor Technology and Products Organization. His professional interests are in the theory and application of time-frequency and time-scale representations for signal and image processing, advanced signal processing for detection/extraction/characterization of signals, and digital spread spectrum communications. From May 1977 until January 1988, he was Technical Director of the Systems Engineering Operations, Stanford Telecommunications, Inc. (STel), Reston, VA, where he was engaged in engineering design and analysis of Government communications, navigation, and surveillance systems. During 1973-1977, he was a Member of the Technical Staff of the M.I.T. Lincoln Laboratory, working in analysis and simulation of air traffic control surveillance and guidance systems. From 1964 to 1972, at Sanders Associates, Bedford, MA, and Nashua, NH, he was engaged in the design and analysis of signals and signal processing systems for radar. He has taught extension courses in communications theory and has lectured on spread spectrum communications at George Washington University.

Dr. Orr is a member of Tau Beta Pi and the Signal and Image Processing Working Group of the SPIE.