Fast Convolution Technique for Non-separable Oversampled Lapped Transforms

Kota HORIZUCHI* and Shogo MURAMATSU†

* Graduate School of Science and Tech., Niigata University, Niigata, Japan
E-mail:hori-ko@telecom0.eng.niigata-u.ac.jp
† Dept of Electrical and Electronic Eng., Niigata University, Niigata, Japan
E-mail:shogo@eng.niigata-u.ac.jp.

Abstract—In this study, an acceleration technique for non-separable oversampled lapped transforms (NSOLT) is proposed. NSOLT is a non-separable transform which simultaneously satisfies the oversampled, overlapped, symmetric, tight-frame, real-valued and compact-supported property. From the non-separability, NSOLT is valid for sparsely representing edges and textures with oblique directions as a linear generative model. In this paper, a fast convolution technique for NSOLT is proposed by using fast Fourier transform (FFT) and frequency-domain multirate processing. The analysis and synthesis processing for 2-D and 3-D data is evaluated through GPGPU implementation, and the processing speed of the proposed structure is compared with the conventional lattice-based one to show the effectiveness.

I. INTRODUCTION

With the development of vision technologies, displays and cameras tend to be capable of treating high definition pictures, such as 4K and 8K resolution TV. These devices are becoming mainstream and efficient image processing technology is required for such large-scale image data. Furthermore, imaging techniques, such as Magnetic Resonance Imaging (MRI) and Optical Coherence Tomography (OCT), have been used in the medical field. Image restoration technology for these large-scale volumetric images is also required[1][2].

In order to improve analysis, recognition and restoration performance of images, the use of sparse representation is proposed with redundant transform by a lot of researchers and its effectiveness is confirmed. In recent years, non-separable transforms have been studied to cope with the representation of diagonal edges and textures[3].

We have proposed an image restoration with non-separable oversampled lapped transform (NSOLT)[4]. NSOLT is a lattice-based redundant transform which simultaneously satisfies the symmetric, real-valued and compact-supported property, and is effective for sparse representation of edges and textures in oblique directions. So far we have confirmed that image restoration using NSOLT shows a high performance[5][6]. However, the computation amount of NSOLT is large, and it is an issue to take time for processing data-intensive images.

To solve this problem, we propose a parallel filterbank structure of NSOLT using fast Fourier transform (FFT) and attempt to improve the processing speed for data-intensive image. In this paper, we propose to exploit the symmetric and real-valued property of NSOLT and effectively utilize a real-valued FFTs.

In Section 2, we describe image representation using NSOLT. In Section 3, we overview the necessary knowledge for the proposed structure, and describe the proposed structure in Section 4. In Section 5, we evaluate the performance of the proposed structure, and conclude this study in Section 6.

II. NON-SEPARABLE OVERSAMPLED LAPPED TRANSFORM

In this section, we overview image representation using NSOLT.

A. Image Representation with Redundant Dictionary

An image can be represented by a linear combination of properly prepared atomic images.

Let $N$ and $M$ be the number of pixels (or voxels) and the number of atomic images, respectively. Then, an image $u \in \mathbb{R}^N$ can be expressed as

$$u = Dy,$$

where $D \in \mathbb{R}^{N \times M}$ is a rank-$N$ matrix consisting of $M$ vectorized atomic images and $y \in \mathbb{R}^M$ is a coefficient vector. The matrix $D$ is referred to as a "dictionary" for $\mathbb{R}^N$.

When $M = N$ and atomic images in $D$ are linearly independent from each other, image $u$ is uniquely determined from $y$. A dictionary of this case is referred to as a "non-redundant dictionary." Discrete cosine transform (DCT) and discrete wavelet transform (DWT) are examples of this type. In the case of $M > N$, $y$ representing $u$ is not determined uniquely. A dictionary of this type is referred to as a "redundant dictionary". Undecimated Haar transform is an example of this type. A redundant dictionary is effective in sparse representation of images that gives a better approximation with less coefficient. As a result, high quality image restoration can be achieved by a threshold processing, such as soft and hard shrinkage to the coefficients.

B. Lattice Structure of NSOLT

In previous studies, we have proposed non-separable oversampled lapped transforms (NSOLTs) as redundant dictionary $D$[4]. NSOLTs allow us to achieve overcomplete systems with non-separable, symmetric, real-valued, overlapped and compact-supported filters. NSOLT is composed of symmetric
and antisymmetric atomic images, and classified by the number of symmetric atomic images $p_s$ and antisymmetric ones $p_a$ as follows:

- Type-I : $p_s = p_a$,
- Type-II : $p_s \neq p_a$.

In this paper, we discuss only the Type-I construction. Fig. 1 shows the lattice structure of Type-I analysis NSOLT, where $p_s = p_a = 5$. The polyphase matrix of the structure is expressed by

$$ E(z) = \prod_{d=0}^{D-1} \left( \prod_{n=1}^{N_d} \begin{bmatrix} R_{n}^{(d)} & Q(z_d) \end{bmatrix} \right) \cdot R_0 E_0, $$

(2)

where $z = (z_0, z_1, ..., z_{D-1})^T$, $N_d \in \{0 \cup \mathbb{N}\}$ is the polyphase order of the $d$-th dimension.

$$ Q(z_d) = B_P \begin{bmatrix} I_p & \mathbb{O} \end{bmatrix} \begin{bmatrix} \mathbb{O} & z_d^{-1} I_p \end{bmatrix} B_P, $$

(3)

$$ R_{n}^{(d)} = \begin{bmatrix} I_p & \mathbb{O} \end{bmatrix} \begin{bmatrix} \mathbb{O} & U_{n}^{(d)} \end{bmatrix}, $$

(4)

and

$$ B_P = \begin{bmatrix} I_p & I_p \\ I_p & -I_p \end{bmatrix}, $$

(5)

where $U_{n}^{(d)} \in \mathbb{R}^{\frac{P}{2} \times \frac{P}{2}}$ is an arbitrary invertible matrices, $\mathbb{O}$ is null matrix, $I_p$ is the identity matrix of size $p \times p$. As the initial matrix $E_0(z)$, we adopt the product of $E_0 \in \mathbb{R}^{M \times M}$, denoting $D$-dimensional DCT, and

$$ R_0 = \begin{bmatrix} W_0 & \mathbb{O} \\ \mathbb{O} & U_0 \end{bmatrix} \begin{bmatrix} I_{\frac{P}{2}} & \mathbb{O} \\ \mathbb{O} & I_{\frac{M}{2}} \end{bmatrix} \in \mathbb{R}^{P \times M}, $$

(6)

where $P = \frac{P}{2} + \frac{D}{2}$, $W_0, U_0 \in \mathbb{R}^{\frac{P}{2} \times \frac{P}{2}}$ are arbitrary orthonormal matrices. When the parameter matrix, $W_{\ell}^{(d)}$, $U_{\ell}^{(d)}$, $W_0$ and $U_0$, are orthonormal, $E(z)$ becomes paraunitary. This paraunitary system corresponds to a tight-frame, with unit frame boundary (Perseval frame) [7]. When the analyzer satisfies the paraunitary property, a paraunitary synthesizer is obtained as

$$ R(z) = z^{-\tilde{\pi}} E^T(z^{-1}), $$

(7)

which is the polyphase matrix of synthesizer and $\tilde{\pi} = (N_0, N_1, ..., N_{D-1})^T \in (0 \cup \mathbb{N})^D$ is the polyphase order. Note that we adopt this synthesizer as tight-frame dictionary $D$.

**C. Design Example of NSOLT**

Using the lattice structure of NSOLT in Fig. 1, we can design a dictionary based on example-based learning by collecting the parameter matrices[8]. Fig. 2 shows an example set of learned atomic images. The design configuration is summarized in Table I.
D. Structure and Expression of Filterbank

The above-mentioned lattice structure is one of filterbank realization methods. An Filterbank system is constituted by a synthesizer and analyzer. Also, if it is in a tree structure, it is possible to analyze signals in multiresolution manner [9][10][11].

Fig. 3 is an example of tree structure of uniform filterbanks and Fig.4 is an example of non-uniform filterbank equivalent to the system in Fig. 3. In the case of non-uniform filterbank, it is possible to perform the processes in different tree levels in parallel[11][12][13].

E. Parallel Filterbank Structure of Hierarchical NSOLT

Regarding designed atomic images as filters iteratively applying the analysis-synthesis process and converting the structure to one in Fig. 4, it is possible to process hierarchical NSOLT using the parallel filterbank structure. By the non-separability of filters, it would be a problem to increase the amount of computation for multidimensional-convolution. However, choosing periodic extension as the boundary operation of images and using discrete Fourier transform (DFT) for the convolution, we can avoid this problem, where the fast Fourier transform (FFT) is available instead of DFT [14].

In this study, we discuss the parallel filtering implementation of NSOLT using FFT.
A. Circular Convolution using FFT

For filtering a multi-dimensional signal with the periodic extension, we can realize it through circular convolution using DFT, where FFT is applicable instead of DFT [15]. The circular convolution \( u_{k,t}(n) \) between input signal \( x(n) \) and FIR filter \( h_{k,t}(n) \) correspond to each of FFT \( Y_{k,t}(k) \), \( X(k) \) and \( H_{k,t}(k) \), which are related as follows:

\[
u_{k,t}(n) = h_{k,t}(n) \ast x(n) \iff U_{k,t}(k) = H_{k,t}(k)X(k),
\]

where operator \( \ast \) denotes the circular convolution.

In practical application of NSOLT, the input to FFT is real-valued array and the coefficient \( F \) satisfies

\[
F(-k) = F^\ast(k).
\]

Using this property, we can reduce the amount of data to be stored to a half to be stored. In addition, since filters are symmetric or antisymmetric real array, it is possible to express each FFT by using DCT and discrete sine transform (DST) so that the amount of data to be stored is reduced to a quarter [14][15].

B. Rate Conversion in DFT Domain

In previous works, it has been shown that rate conversion in the spatial domain corresponds to a coefficient operation of DFT [16][17]. Let \( M_d \) be the downsampling ratio and upsampling ratio and \( L \) is the DFT points. The downsampling process corresponds to folding-addition of DFT coefficient as

\[
Y(\ell) = \frac{1}{M_d} \sum_{n=0}^{M-1} U(nA_d + \ell),
\]

where \( Y(\ell) \) is the DFT coefficients after folding-addition and \( A_d = L/M_d \). The upsampling process corresponds to periodic-expansion of DFT coefficients as

\[
V_u(n) = J_u(([n])_B), 0 \leq n \leq M_d - 1
\]

where \( V_u \) is the DFT coefficients after periodic-extension, \( B = L/M_d, ([\cdot])_B \) is the remainder of the argument modulo \( B \). Using these relations, we can reduce the DFT points, leading to reduce the amount of computation.

IV. FFT-BASED REALIZATION OF NSOLT

In this section, we describe the parallel filterbank structure of NSOLT using FFT and the processing procedure in the proposed structure.

A. Summary of Proposed Structure

In this paper, we expand hierarchical NSOLT to the parallel filterbank structure. Fig. 5 shows the implementation of parallel filterbank structure of NSOLT in the FFT domain. In this structure, before performing the analysis and synthesis process, we prepare FFTs of atomic images in the tree structure. Since the input to FFT is assumed to be real, the data amount of the FFTs will be a half. Furthermore, atomic images of NSOLT are composed of real coefficients and are even or odd symmetric, the data amount to be stored will be a quarter. Compared with the lattice structure, although the proposed structure uses large memory capacity, it is possible to accelerate processing for large-scale data by using parallel computing devices, such as GPGPU.
B. Procedure of Proposed Structure

The processing procedure of the proposed structure in Fig. 5 is as follows:

[ Analyzer ]

Step 1: Calculate $X(k)$, the FFT of input $x(n)$,

Step 2: Calculate the element-wise products between $X(k)$ and $H_{k,t}(k)$, the FFT of the $k$-th analysis filter on the $t$-th tree,

Step 3: Fold and add the results in Step 2 as (11),

Step 4: Calculate $y_{k,t}(n')$ as output coefficients, the inverse FFT of results in Step 3.

[ Synthesizer ]

Step 1: Calculate $\hat{Y}_t(k')$, the FFT of input coefficients $\hat{y}_t(n')$,

Step 2: Periodically-extend $\hat{Y}_t(k')$ as (12),

Step 3: Calculate the element-wise products between $\hat{Y}_{k,t}(k)$ and $H_{k,t}(k)$, the prepared FFT of synthesis filters,

Step 4: Calculate $\hat{x}(n)$ as output, the inverse FFT of the sum of all results in Step 3.

V. PERFORMANCE EVALUATION

In this section, to verify the performance of the proposed structure, we compare the processing speed with that of the lattice structure.

A. Implementation Configuration and Specifications

We have implemented the model in Fig.5 using MATLAB R2016a, executed it on GPGPU compare the lattice and structure, we compare the processing speed with that of the lattice structure.

R2016a, executed it on GPGPU compare the lattice and structure. We have implemented the model in Fig.5 using MATLAB R2016a, executed it on GPGPU compare the lattice and structure.

B. GPGPU Overview

In this evaluation, we perform NSOLT in parallel processing using a GPGPU (General-Purpose computing on GPU) [18]. GPGPU is possible for a large amount of data to be processed simultaneously in parallel using a multiple processors [19].

Compared with the lattice structure, the acceleration using GPGPU is effective for the proposed structure, handling large array, performing independent computations of the neighboring pixels or voxels such as the element-wise coefficient products.

C. Execution Results and Discussion

Tables V and VI summarize the processing time and the speed gain for each image size and dimension. From the result, it is found the effectiveness of acceleration due to the parallel implementation. Furthermore, the speed gain increases as the image size is confirmed. Our conjecture is that this result is caused by the decrease of effectiveness of the parallel computing, due to an increase of the number of tree level.

VI. CONCLUSIONS

In this study, we proposed a parallel filterbank structure of hierarchical NSOLT with FFT domain implementation. The proposed structure was implemented on GPGPU, and evaluated the processing speed. The proposed structure was verified to be effective for a large amount of images.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number JP26420347.

REFERENCES


TABLE II

SPECIFICATIONS OF 2-D NSOLT

<table>
<thead>
<tr>
<th>Downsampling Ratio $M$</th>
<th>$2 \times 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td># Channels $P$</td>
<td>$4 + 4 = 8$</td>
</tr>
<tr>
<td>Data size of Filter</td>
<td>$10 \times 10$</td>
</tr>
<tr>
<td>Tree level $T$</td>
<td>3</td>
</tr>
</tbody>
</table>

TABLE III

SPECIFICATIONS OF 3-D NSOLT

<table>
<thead>
<tr>
<th>Downsampling Rate $M$</th>
<th>$2 \times 2 \times 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td># Channels $P$</td>
<td>$5 + 5 = 10$</td>
</tr>
<tr>
<td>Data size of Filter</td>
<td>$6 \times 6 \times 6$</td>
</tr>
<tr>
<td>Tree level $T$</td>
<td>4</td>
</tr>
</tbody>
</table>

TABLE IV

SPECIFICATION OF GPGPU

<table>
<thead>
<tr>
<th>GPGPU Device</th>
<th>NVIDIA Tesla C2075</th>
</tr>
</thead>
<tbody>
<tr>
<td># mounted GPU</td>
<td>1</td>
</tr>
<tr>
<td># CUDA’s core</td>
<td>448</td>
</tr>
<tr>
<td>Frequency of processor</td>
<td>1.15GHz</td>
</tr>
<tr>
<td>Capacity of the mounted memory</td>
<td>6GB</td>
</tr>
<tr>
<td>Size of Input Image (pixel)</td>
<td>Processing Time (sec) (Lattice Structure)</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>128 × 128</td>
<td>1.358</td>
</tr>
<tr>
<td>256 × 256</td>
<td>2.838</td>
</tr>
<tr>
<td>512 × 512</td>
<td>5.034</td>
</tr>
<tr>
<td>1024 × 1024</td>
<td>19.652</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size of Input Image (voxel)</th>
<th>Processing Time (sec) (Lattice Structure)</th>
<th>Processing Time (sec) (Parallel Filterbank Structure)</th>
<th>Speed Gain (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 × 64 × 32</td>
<td>2.375</td>
<td>1.387</td>
<td>129.29</td>
</tr>
<tr>
<td>128 × 128 × 32</td>
<td>9.427</td>
<td>3.223</td>
<td>292.49</td>
</tr>
<tr>
<td>256 × 256 × 32</td>
<td>26.254</td>
<td>4.423</td>
<td>593.58</td>
</tr>
</tbody>
</table>


