Embedded Implementation Model of 2-D Non-separable Oversampled Lapped Transforms for Video Processing

Keita IMAI[1] and Shogo MURAMATSU[2]
[1]Graduate School of Science and Technology, Niigata University
8050 2-no-cho Ikarashi, Nishi-ku, Niigata, 950-2181, Japan
[2]Department of Electrical and Electronic Engineering, Niigata University,
8050 2-no-cho Ikarashi, Nishi-ku, Niigata, 950-2181, Japan
E-mail : [1]keita@telecom0.eng.niigata-u.ac.jp, [2]shogo@eng.niigata-u.ac.jp

Abstract: This work proposes an embedded implementation model of 2-D Non-separable Oversampled Lapped Transform (NSOLT) for video processing. NSOLT satisfies the overlapping, symmetric, paraunitary, real-valued and compact-supported property. From these characteristics, NSOLT has successfully been applied to image restoration. Recent development of embedded vision technologies allows us to realize high performance image processing. Thus, the authors have proposed an embedded implementation model of NSOLT as a previous work. The existing model, however, cannot efficiently process successive frames of a video. This paper proposes a modified architecture effective for video processing and verifies the significance in terms of the processing speed and hardware resource usage through the synthesis reports.

Keywords—**Embedded implementation, Non-separable oversampled lapped transform, HW/SW co-implementation, Image processing**

1. Introduction

Currently, embedded vision techniques are significantly growing in many applications, such as video surveillance, robots, vehicles, medical instruments and enterprise servers; because embedded techniques allow us to save space, cost and energy. Image transforms have a crucial role in vision techniques because they are effective to represent images and videos and to restore contaminated ones. The applications include compression as well as denoing, inpainting and deblurring [1–3]. For instance, the discrete cosine transform (DCT) and the discrete wavelet transform (DWT) are well-known image transforms and adopted in various image coding standards, such as JPEG and JPEG2000 [4,5]. However, DCT and DWT are not suitable for representing diagonal textures and edges due to its separability. From this background, we have proposed NSOLT [6]. Compared with DCT and DWT, NSOLT can represent diagonal structures efficiently, because NSOLT is advantageous in term of sparse representation of images thanks of its non-separability and redundancy. As a result, NSOLT is effective for solving image restoration problems such as inpainting, denoising and deblurring [7].

As a previous work, we have already proposed an embedded implementation model of NSOLT. However, the previous model is applicable only to still images [8]. In this article, we propose to extend the existing model for video sequences. The proposed model improves the throughput by introducing pipeline processing. In order to verify the significance, the effectiveness of the proposed model is examined and compared with that of sequencial one in terms of throughput.

2. Review of NSOLT and Previous Model

This section briefly reviews NSOLT and the existing embedded implementation model.

2.1 Lattice Structures of NSOLT

NSOLTs are constructed by lattice structures which structurally guarantee the symmetric, real-valued and compact-supported property. In addition, NSOLT can be directional and Parseval tight-frame [6]. NSOLTs are categorized into two types:

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

where p_s and p_a denote the numbers of symmetric and anti-symmetric atomic images, respectively. This work deals only with the Type-I NSOLT.

When the number of channels P is even, it is possible to set $p_s = p_a = P/2$ and we can construct a Type-I NSOLT. Fig. 1 shows an example of Type-I lattice structure of analysis NSOLT. N_x and N_y in Fig. 1 are the polyphase orders of vertical and horizontal direction, respectively. $\mathbf{W}_0$, $\mathbf{U}_0$ and $\mathbf{U}_n^{\{d\}}$ are parameter matrices. These matrices determine the characteristics of the transform. Matrix $\mathbf{M}$ and $|\det \mathbf{M}| = M = M_x \times M_y$ denote the downsampling factor and ratio, respectively. $\mathbf{E}_0$ is an initial matrix, which is defined by the matrix representation of 2-D DCT.

2.2 Feedback Architecture with Transpostion Process

An embedded NSOLT can be modeled as hardware/software(HW/SW) co-implementation. This implementation method exploits the following two advantages: parallel processing of HW and flexibility of SW. Fig. 2 shows the embedded implementation architecture of NSOLT with a transposition buffer which we have proposed as a previous work [8]. Fig. 2 shows only the forward transform architecture. The inverse transform can be realized by the reverse direction of the forward transform. Module A and Module B denote the module to calculate DCT, $\mathbf{W}_0$, $\mathbf{U}_0$ and the module to calculate the order extension process in the polyphase domain, respectively. The dotted arrows in Fig. 2 are control signals from a control unit, which works to select the mode between the vertical and horizontal direction in the order extension process. The shaded and plain blocks indicate the hardware and software processes, respectively.

In general, non-separable transforms are not able to be separately processed in the vertical and horizontal direction. The process of NSOLT, however, can be separated since NSOLT is

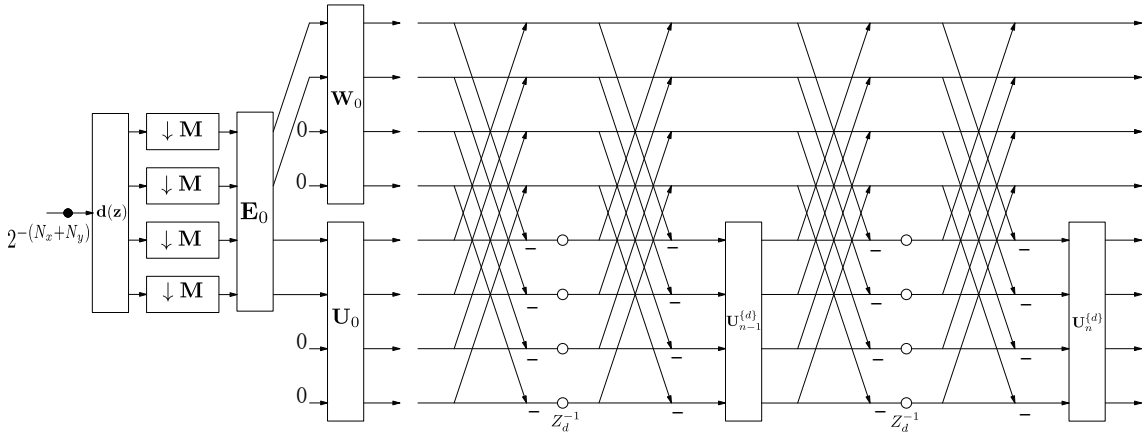

Figure 1. Example of Type-I lattice structure of forward NSOLT, where $\mathbf{M}$ denotes the downsampling factor, $|\det \mathbf{M}| = M = M_x \times M_y = 2 \times 2$ and $P = p_s + p_a = 4 + 4$.

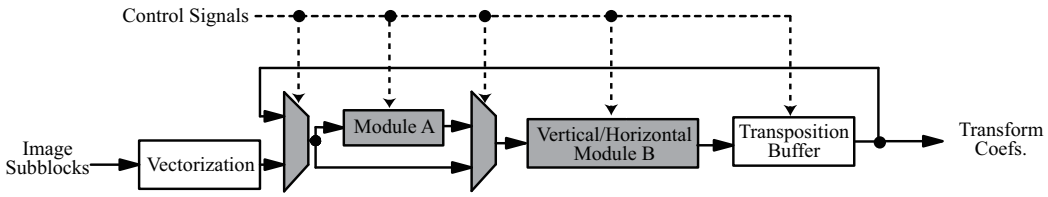

Figure 2. Feedback architecture of NSOLT with a transposition process, where Module A calculates 2-D DCT, $\mathbf{W}_0$ and $\mathbf{U}_0$, and Module B calculates the polyphase domain order extension process. Only the forward transform is illustrated.

factorized into the two directions in the polyphase domain [8]. In addtion, the lattice structure of NSOLT can process each subblock obtained by dividing the whole image. The hardware resources required for NSOLT is possible to be reduced by this characteristic [9, 10].

However, the previous model is designed for still images, not for videos, and the model does not take account of the data transfer between HW and SW.

3. Embedded Implementation Model for Video Processing

In this section, we propose to improve the previous embedded implementation model for video processing.

3.1 Analysis-Synthesis Model for Video Processing

In the previous work, the implementation model does not take the processing of successive video frames into account. Thus, in this work, we propose a novel embedded implementation model of NSOLT for video processing. We suggest to apply pipeline processing to the model. Fig. 3 shows the proposed model of the forward and inverse NSOLT. We designed the model in order to improve the throughput. The proposed model has a simple streaming architecture, where a plain model of the analysis-synthesis system is considered in order to concentrate on the evaluation of the processing speed. The process of Forward Module A and Inverse Module A are reverse to each other. Forward Module B and Inverse Module B are also in a similar relationship. The shaded and plain blocks indicate the hardware and software, respectively. The solid and dashed arrow indicate the data path and control signal.

The proposed model has three stages. A transposition buffer is placed between successive stages. In the followings, we summarize the role of each stage:

Stage A executes Forward Module A and Vertical Forward Module B for analysis process. The processed data is stored in a transposition buffer until the completion of all processing for one frame.

Stage B is composed of Horizontal Forward Module B for analysis process and Horizontal Inverse Module B for synthesis process. Horizontal Forward Module B takes the data output from the transposition buffer. Horizontal Inverse Module B takes the output from the analysis unit. In this plain model, we place no buffer between the analysis and synthesis two unit, and the data flows between the units without any storage. As well as Stage A, the output data from Stage B can also be stored in another transposition buffer until the completion of all processing for one frame.

Stage C executes Vertical Inverse Module B for synthesis process followed by Inverse Module A for the data output from the transposition buffer.

Because these three stages are composed of HW, they can run independently. Accordingly, each stage is able to work concurrently. The proposed model implements pipeline process by using this architecture, which is effective especially for video processing, where successive frames are processed continuously.

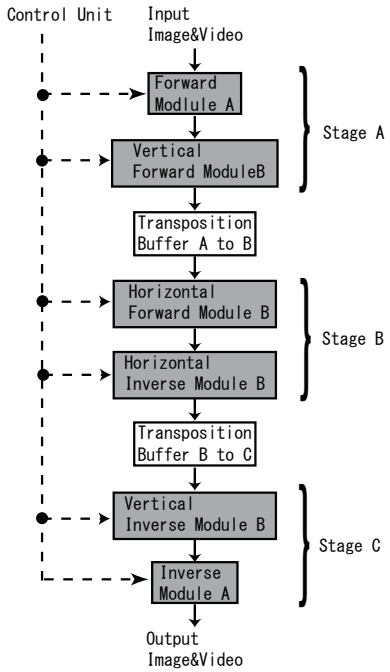

Figure 3. The proposed model of NSOLT, which consists of three stages.

3.2 Reduction of Data Transfer Frequency

We propose a method to reduce the data transfer frequency in order to improve the processing speed of each stage. In general, data transfer time between HW and SW is longer than processing time. Thus, we improve the data transfer frequency from the previous model. The previous model transfers the data between HW and SW in block-by-block. In contrast, the proposed model transfers all of data at once. The block division process of the proposed model is executed by HW after data transfer. Accordingly, the proposed model has less transfer frequency than the previous one. Consequently, we accelerate the processing speed of the proposed model.

3.3 Latency of Proposed Model

Fig. 4 shows the timing chart of the proposed model, where L_0 is the latency of Module A (the forward and inverse module have the same length), L_1 is the latency of Module B (the analysis and synthesis module have the same length), L is the latency of NSOLT to process all the three stages, and L' is the total time to complete the processes for multiple frames. Thus, each latency of Stage A and C is $L_0 + L_1$, the latency of Stage B is $2L_1$. Accordingly, the critical path, denoted by C [s/frame], of the proposed model results in

$$C = \max(L_0 + L_1, 2L_1).$$

We define the throughput, denoted by T[fps], by

$$T = C^{-1}.$$

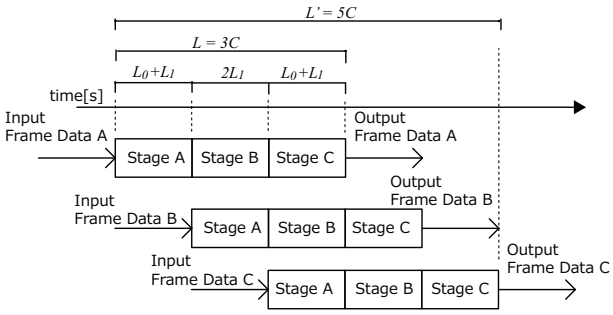

Figure 4. Timing chart of the proposed pipeline model, where L_x denotes latency and C is critical path. For simplicity, $L_0 + L_1$ and $2L_1$ are drawn with the same length. C varies depending on L_0 and L_1.

4. Performance Evaluation

This section evaluates the processing speed and HW resource requirement of the proposed model.

4.1 Evaluation Method

We implement the proposed model to verify the improvement of the throughput. The model is designed for Xilinx's All Programmable SoC in a HW/SW co-implementation manner by Xilinx SDSoC, where it runs with Linux [11]. The data type of the model is set to the single precision floating point. We used the parameters shown in Tabs. 1, 2 and 3. The throughput of the real-time processing is confirmed as follows:

$$T = n_c^{-1} \times f,$$

where n_c denotes the number of cycles per frame and f is the operating frequency on a target device. We verify the number of resource usage from the log file reported by the design tool.

4.2 Evaluation Results

Figs. 5 and 6 summarize the evaluation results. From the graphs, it is observed that the throughputs are increased progressively as the input image size becomes larger. In addition, all throughputs of the pipeline model are faster than those of sequencial one. According to Fig. 6, it is clear that the number of HW resources changes dependly on the input size.

4.3 Discussions

In general, as image size incenses, the usage rate of the resources also grows progressively. However, Fig. 6 shows that a 16×16-pixel image needs more resources than the 32×32-pixel one. Our conjecture is that the proposed model becomes inefficient for small images and this inefficiency has increased the resource usage.

5. Conclusions

In this article, we proposed an embedded implementation model of NSOLT for video processing. The proposed model was designed to improve the throughput with pipelining. Through evaluation of the proposed model, we verified the

Table 1. Parameters of the logic synthesis

Design tool	SDSoC 2015.4
Target device	XC7Z020-1CLG484(Zynq-7000)
Operating frequency of HW	142.86[MHz]
Operating frequency of SW	142.86[MHz]
Embedded OS	PetaLinux v2015.2.1 (Yocto 1.8)
Data Type	Single Precision Floating Point

Table 2. Parameters of NSOLT

Polyphase order	$N = N_y = N_x = 2$
Number of channels	$P = p_s + p_a = 4 + 4 = 8$
Decimation rate	$M = M_y \times M_x = 4$

Table 3. Parameters of input data

Image format	GrayScale
Bit depth	8bit
Image size	64×64

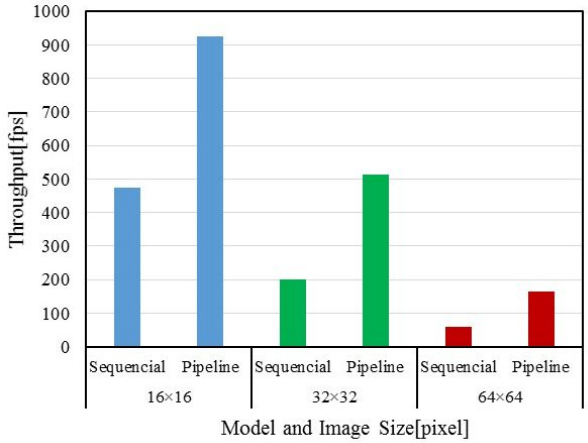

Figure 5. Results of thoughput.

improvement of the throughput and efficiency of the HW resource usage.

In future, we will further improve the processing speed and HW resource usage. We will consider implementing NSOLT in the cluster configuration and fixed-point. Furthermore, we will work on the implementation of the image processing applications, and the tree structure model.

Acknowledgement

This work was supported by KAKENHI (No.26420347).

References

[1] S. Mallat, "A Wavelet Tour of Signal Processing Secound Edition, " Academic Press, 1999.

[2] J. L. Stack, F. Murtagh, and J. M. Fadili, "Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity, " Cambridge University Press, 2010.

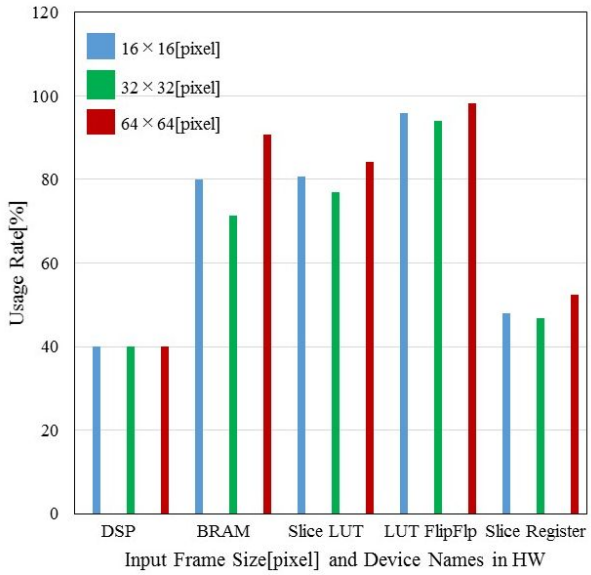

Figure 6. Results of resource usage rate with HW resources.

[3] M. N.Do and M. Vetterli, "The Contourlet Transform: An Efficient Directional Multiresolution Image Representation, " *Proc. of IEEE Trans. Image Process.*, vol. 14, No. 12, pp.2091-2106, Dec. 2005.

[4] V. Bhaskaran and K. Konstantindies, "Image and Video Compression Standards," *kluwer Academic Publishers*, 1997.

[5] D. S. Taubman and M. W. Marcellin, "JPEG2000, image compression fundamentals, standards and practic," *kluwer Academic Publishers*, 2002.

[6] S. Muramatsu and N. Aizawa, "Lattice Structure for 2-D Non-separable Oversampled Lapped Transforms," *Proc. of IEEE ICASSP*, pp. 5632-5636, May 2013.

[7] S. Muramatsu and N. Aizawa ,"Image Restoration with 2-D Non-separable Oversampled Lapped Transforms, " *Proc. of IEEE ICIP*, 15-18 Sept. 2013.

[8] K. Seino and K. Furuya and S.Muramatsu, "Transposition-based architecture of 2-D Non- 2-D Non-separable Oversampled Lapped Transforms, " *Proc. of APSIPA ASC*, Dec. 2014.

[9] S. Muramatsu and M. Hiki, "Block-wise implementation of directional GenLOT, " *Proc. of IEEE ICIP*, pp. 3977-3980, May 2009.

[10] K. Furuya and S. Hara and S.Muramatsu, "Boundary Operation of 2-D Non-separable Oversampled Lapped Transforms, " *APSIP ASP*, Nov. 2013.

[11] Xilinx, "SDSoC Environment User Guide Getting Started, " http://www.xilinx.com/support/documentation/sw_manuals/xilinx2015_2/ug1028-sdsoc-getting-started.pdf, Apr. 2016.