EFFICIENT 3-D PARALLEL FIR FILTERING ALGORITHM

*M. Aziz and S. Boussakta**

School of Engineering & Technology
De Montfort University, Leicester, LE1 9BH, UK
Email: maziz@dmu.ac.uk
*School of Electrical, Electronics and Computer Engineering
University of Newcastle upon Tyne, Newcastle, NE1 7RU
Email: s.boussakta@ncl.ac.uk

ABSTRACT

In this paper we present an efficient 3-D digital filtering method based on a new 3-D parallel filtering algorithm. This method is suitable for high resolution / high speed 3-D image and video processing. The proposed 3-D parallel filtering algorithm is highly efficient due to the elimination of overlapping segments overhead in the block-filtering method, and the boundary conditions in parallel filtering applications. It also solves the problem of limited efficiency of direct FIR filtering when the system impulse response is large, and enhances the overall memory distribution of the parallel system by segmenting both the 3-D input data and the impulse response of the system into smaller independent subsections that can be simultaneously processed. Finally, the algorithm's performance is assessed and analyzed.

1. INTRODUCTION

Digital filtering is an important application in communications and digital signal processing, and is widely used in many other applications in science and engineering [1-8]. However, digital filtering is known to be computationally intensive, and for many applications special hardware is utilised to achieve the required level of performance in acceptable time or real time [4-7] constraint. In fact, the demand is ever increasing for high-performance, flexible and reliable processors, as many commercially viable applications in science and engineering are identified requiring more processing power than that available form the single processor chip [10-12].

In [9] Agarwal and Burrus extended the block convolution method [8] to multidimensional convolution operation, which was later parallelised and mapped onto multiprocessor platform in 1-D & 2-D [10-12]. However, because of the overlapping segments in the block convolution algorithm [13] additional computational overhead was incurred and for efficient performance the relative size of the input data had to be restricted [13-15]. In [2] Aziz and Boussakta developed a new parallel filtering algorithm in 2-D and proposed efficient fast convolution algorithms using number theoretic transforms (NTTs) achieving high speeding factors compared with the number of parallel processors used as well as exact FIR filtering free from rounding and truncation errors.

In this paper a direct 3-D parallel FIR filtering algorithm is presented and its performance is assessed and analyzed for real time multiprocessor system [16]. The performance results reflects the highly parallel nature of the proposed algorithm, which eliminates the problem of overlapping segments in the block filtering method and the boundary conditions in the parallel filtering implementation [14-15].

2. 3-D PARALLEL FILTERING ALGORITHM DEVELOPMENT AND ANALYSIS

The new 3-D parallel filtering algorithm is presented in Figure 1. This algorithm differs from the block filtering method by segmenting both the input data and the impulse response of the system into smaller sub-blocks that have no overlapping regions, by using 3-D decimation by 2 algorithms in the input stage.

The 3-D parallel algorithm is divided into three main stages. The first stage is the 3-D input and impulse response decimation stage. The second stage is the 3-D parallel filtering stage of the different segments of the inputs, using a SIMD (single instruction multiple data [4]) parallel configuration, for optimal performance. The third stage is the final filtering output reconstruction stage. In the next sub-sections the mathematical modelling of the three stages are presented.

2.1 3-D Parallel input stage

The input stage of the 3-D parallel algorithm uses decimation by 2 in 3-D, producing 8-subsections for each 3-D input, see Figure 1. The subsections generated constitute exactly the same samples as the input block and hence no samples are lost due to the decimation process, and the resulting subsections are independent and not overlapping.

The input decimation stage involves the decimation of two data blocks, i.e. the 3-D input block $x(n_1, n_2, n_3)$ of size $(N \times N \times N)$ and the 3-D impulse response $h(n_1, n_2, n_3)$ of size $(M \times M \times M)$, into 8-segments each of sizes $\left(\frac{N}{2} \times \frac{N}{2} \times \frac{N}{2}\right)$ and $\left(\frac{M}{2} \times \frac{M}{2} \times \frac{M}{2}\right)$ respectively.

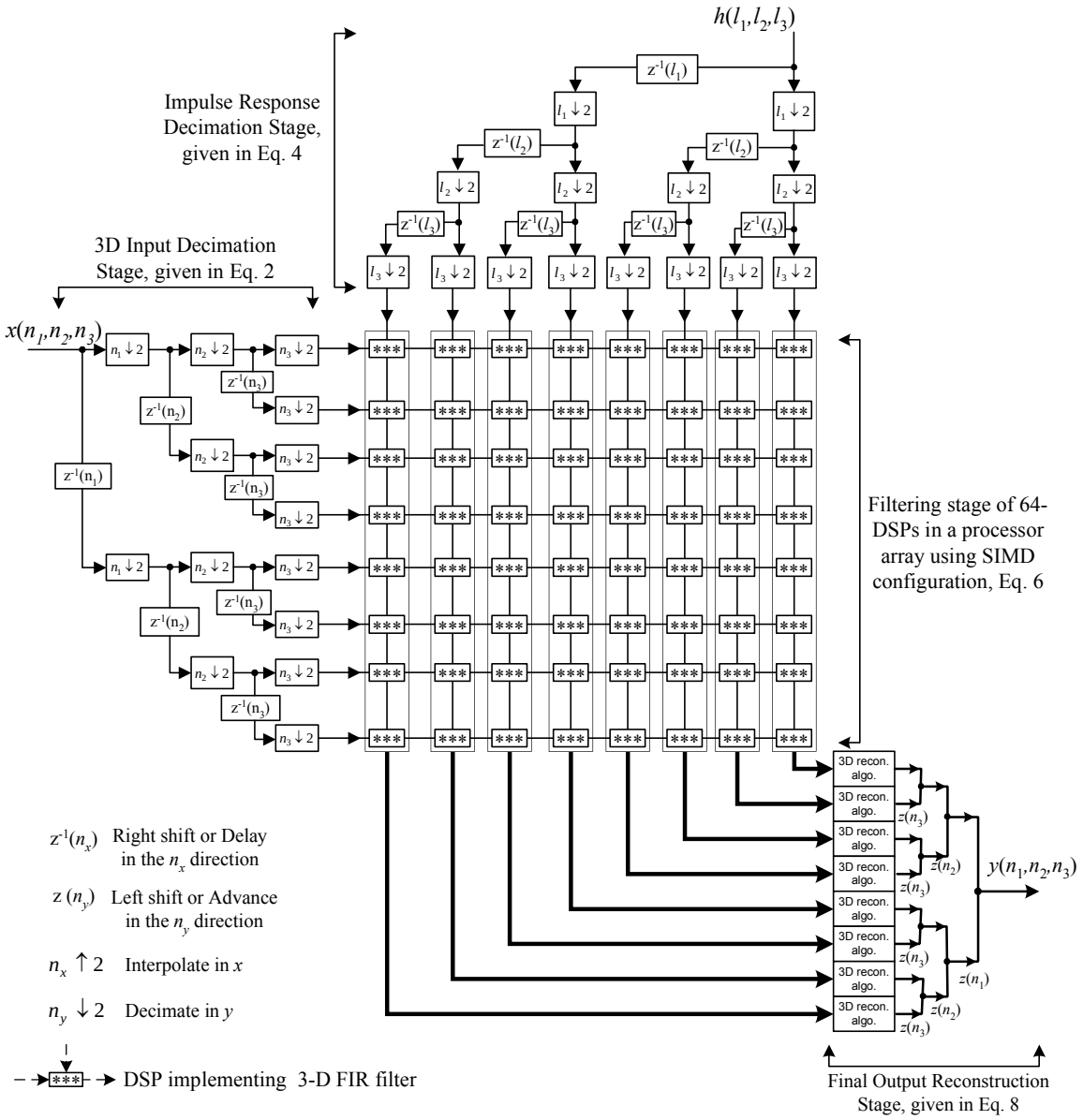

Figure 1 - The 3-D Parallel Filtering Algorithm

The 3-D input decimation process is defined as:

$$x_{i_1 i_2 i_3}(n_1,n_2,n_3)\Big|_{i_1,i_2,i_3=0,1} = x(2n_1+i_1,2n_2+i_2,2n_3+i_3) \quad (1)$$

where $0 \le n_1,n_2,n_3 \le N/2-1$ and $i_1,i_2,$ and i_3 take eight binary values, i.e. 000, 001, …, 111. So that the segments of $x(n_1,n_2,n_3)$ are defined as:

$$\begin{aligned} x_0 &= x_{000}(n_1,n_2,n_3) = x(2n_1,2n_2,2n_3) \\ x_1 &= x_{001}(n_1,n_2,n_3) = x(2n_1,2n_2,2n_3+1) \\ &\vdots \\ x_7 &= x_{111}(n_1,n_2,n_3) = x(2n_1+1,2n_2+1,2n_3+1) \end{aligned} \quad (2)$$

The impulse response of the system $h(n_1,n_2,n_3)$ is also decimated by 2, producing 8-segments, defined as:

$$h_{i_1 i_2 i_3}(n_1,n_2,n_3)\Big|_{i_1,i_2,i_3=0,1} = h(2n_1+i_1,2n_2+i_2,2n_3+i_3) \quad (3)$$

where $0 \le n_1,n_2,n_3 \le M/2-1$ and $i_1,i_2,$ and i_3 takes the binary values, i.e. 000, …, 111. So that the 8-segments of the impulse response are defined as:

$$\begin{aligned} h_0 &= h_{000}(n_1,n_2,n_3) = h(2n_1,2n_2,2n_3) \\ h_1 &= h_{001}(n_1,n_2,n_3) = h(2n_1,2n_2,2n_3+1) \\ &\vdots \\ h_7 &= h_{111}(n_1,n_2,n_3) = h(2n_1+1,2n_2+1,2n_3+1) \end{aligned} \quad (4)$$

2.2 3-D Parallel FIR filtering stage

The second stage is the 3-D FIR filtering of the different segments shown in Eq's 2 and 4 above. This is achieved by direct convolution [8-9]. However, on the 3-D parallel algorithm the 3-D filtering is processed in parallel using 64 sub-filtering equations for the different segments of the 3-D inputs. These parallel sub-filters are expressed by the 3-D convolution equation as:

$$\begin{aligned} y_{ij}(n_1,n_2,n_3) &= x_j(n_1,n_2,n_3) *** h_i(n_1,n_2,n_3)\big|_{i,\,j=0,\ldots,7} \\ &= \sum_{m_1=0}^{(N/2)-1}\sum_{m_2=0}^{(N/2)-1}\sum_{m_3=0}^{(N/2)-1} x_j(m_1,m_2,m_3)\times \\ &\quad h_i(n_1-m_1,n_2-m_2,n_3-m_3) \end{aligned} \tag{5}$$

where $0 \le (n_1,n_2,n_3) \le N/2+M/2-1$ are the output dimensions of each sub-filter. In general the 64 sub-filters carried out on the multiprocessor array are given by:

$$\begin{bmatrix} y_{00} & y_{01} & \cdots & y_{07} \\ y_{10} & y_{11} & \cdots & y_{17} \\ \vdots & \vdots & \ddots & \vdots \\ y_{70} & y_{71} & \cdots & y_{77} \end{bmatrix} = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_7 \end{bmatrix} *** \begin{bmatrix} x_0 & x_1 & \ldots & x_7 \end{bmatrix} \tag{6}$$

From Eq 6, it is clear that the 3-D filtering stage is computationally intensive. Hence, the parallel filtering stage is configured in a SIMD configuration for optimal performance [4], in order to achieve simplified synchronization between processors with minimum interprocessor communications.

2.3 3-D Parallel output reconstruction stage

The third stage is the output reconstruction stage. The 3-D parallel output stage can be viewed as a parallel to serial conversion stage. In this stage the reconstruction of the output $y(n_1,n_2,n_3)$ is carried out in two steps, using the 64 decimated sub-output segments given by Eq. 6. In the first step using interpolations and shifts (delays), producing 8-sub outputs $\tilde{y}_{000}, \tilde{y}_{001}, \tilde{y}_{010}, \tilde{y}_{011}, \tilde{y}_{100}, \tilde{y}_{101}, \tilde{y}_{110}$, and $\tilde{y}_{111}$, these are defined as:

$$\tilde{y}_{i_1i_2i_3}(n_1,n_2,n_3) = \sum_{i_1=0}^{1}\sum_{i_2=0}^{1}\sum_{i_3=0}^{1}\left[y_{i_1i_2i_3}(2n_1+i_1, 2n_2+i_2, 2n_3+i_3)\right] \tag{7}$$

where $y_{i_1i_2i_3}(n_1,n_2,n_3)$ are the sub-outputs produced from the different processing elements. However, the final output $y(n_1,n_2,n_3)$ is then reconstructed in the second stage from the intermediate outputs $\tilde{y}_0, \tilde{y}_1, \ldots, \tilde{y}_7$, as follows:

$$\begin{aligned} y(n_1,n_2,n_3) &= \sum_{i_1=0}^{1}\sum_{i_2=0}^{1}\sum_{i_3=0}^{1}\left[\tilde{y}_{i_1i_2i_3}(n_1+i_1,n_2+i_2,n_3+i_3)\right] \\ &= \tilde{y}_0(n_1,n_2,n_3) + \tilde{y}_1(n_1,n_2,n_3+1) \\ &\quad + \tilde{y}_2(n_1,n_2+1,n_3) + \tilde{y}_3(n_1,n_2+1,n_3+1) \\ &\quad + \tilde{y}_4(n_1+1,n_2,n_3) + \tilde{y}_5(n_1+1,n_2,n_3+1) \\ &\quad + \tilde{y}_6(n_1+1,n_2+1,n_3) + \tilde{y}_7(n_1+1,n_2+1,n_3+1) \end{aligned} \tag{8}$$

where $0 \le n_1,n_2,n_3 \le N+M-1$ are the final output dimensions, and $i_1, i_2,$ and i_3 are the shifts in the 3-D output array.

3. PARALLEL ALGORITHM EFFICIENCY AND PERFORMANCE RESULTS

In this section the performance and speeding factor for the 3-D parallel filtering algorithm is calculated on a multiprocessor DSP system (the ASP-P15 Quad-DSP card [16]), consisting of 4-SHARC DSP processors (ADSP21060), and benchmarked against a single SHARC-DSP processor system implementing 3-D FIR filtering. The simulation is carried out with test data sets of size (64×64×64) each, and the performance results are given in Table 1, where the computational time for the parallel system is calculated and compared to that of the single processor system and the speeding factor is calculated accordingly.

Input Size	(64×64×64)×(64×64×64)
Filter on 4-DSPs	150.1
Filter on 1-DSP	826.3
Speeding Factor	5.5

Table 1 - Performance results for the 3-D parallel filtering algorithm

The performance of the 3-D parallel algorithm is found to be very high, more than the number of DSP processor used, i.e. 4-DSPs. This is due to the following reasons:

- By using the 3-D parallel algorithm, the input is decimated into smaller sections that take advantage of the internal memory of the processing devices. This makes data fetching and storing very efficient on the DSP devices
- The interprocessor communications for the 3-D parallel algorithm is very low, due to the elimination of the boundary conditions in the filtering method. This significantly improves the overall performance of the system
- For the single DSP filtering all data are stored in the external memory of the system with added wait states. This effectively adds extra processing clocks for synchronizing the internal processor hardware

with the external memory in every fetching and storing operations

- In fact, the extensive use of external memory in the single processor system proves to be a major factor in restricting the system performance

4. CONCLUSION

In this paper we have presented an efficient 3-D parallel FIR filtering algorithm. The 3-D algorithm involves the decomposition of two 3-D input blocks into smaller independent segments, which are easier to manage and can be processed in parallel without interprocessor communications. The mathematical derivations of the parallel algorithm stages are given and the performance assessed using real parallel DSP card.

Furthermore, the 3-D parallel filtering algorithm eliminates the overlapping segments in the block filtering method and the boundary conditions in parallel filtering implementation, as both inputs are segmented into independent subsections that can be simultaneously processed. These features along with the elimination of interprocessor communication overhead and the efficient use of the internal memory banks of the processing elements (PE) allow the 3-D parallel algorithm to achieve high speeding factors that can exceed the number of processors used as shown in Table 1.

5. ACKNOWLEDGMENT

The authors have the pleasure in acknowledging the financial support of the EPSRC, under Grant No. GR/S 98160.

REFERENCES

[1] E. Gelenbe, “Multiprocessor performance”, Wiley series in parallel computing, (ISBN 0-471-92392-3), 1989.

[2] M. Aziz and S. Boussakta, “A hybrid parallel algorithm for digital image filtering applications”, Proc. of IEEE Inter. Conf. on Elec. Circ. And Systems, ICECS-2000 Conf., Lebanon, 2000, pp. 591-594.

[3] A. S. Jahangir, “Parallel beam mapping algorithm to compute radiation dose in 3-D treatment planning”, Ninth IEEE symposium on computer based medical systems, 1996, pp.24-29.

[4] A. Downton and D. Crookes, “Parallel architectures for image processing”, Electronics & Communications Engineering Journal, 1998, pp. 139-151.

[5] D. Moldovan, “Parallel processing: from applications to systems”, Morgan Kaufmann Publishers, Inc. (ISBN 1-55860-254-2), 1993.

[6] S. L. Hurt and A. Rosenfield, “Noise reduction in three dimensional digital images”, Pattern recognition, vol. 17, No. 4, 1984, pp. 407- 421.

[7] M. L. C. Hamilton, “The application of multiprocessor DSP to machine vision”, IEE Colloquium on Multiprocessor DSP Applications, Algorithms and Architectures, 1995, pp. 8/1 – 8/5.

[8] Gold, B., and Rader, C. M., “Digital processing of signals”, McGraw Hill, New York, pp. 171-172, 1969.

[9] R. C. Agarwal, and C. S. Burrus, “fast one-dimensional digital convolution by multidimensional techniques”, IEEE Trans. Acoust. Speech Signal processing, ASSP-22, 1974, pp. 1-10.

[10] M. Aziz, and S. Boussakta, “Parallelisation of the Overlap-Add Block-Filter Algorithm for Image Processing”, XI European Signal Proc. Conf., EUSIPCO’02 Toulouse, Vol. II, 2002, pp. 399-402.

[11] M. Aziz, and S. Boussakta, “Parallelisation of the 1-D Block Filter Algorithm to Run on Multiple DSPs”. IEEE Inter. Conf. on Elec. Circ. And Systems, ICECS’02 Dubrovnik, 2002, pp. 943-946.

[12] M. Aziz, S. Boussakta, and D. C. McLernon, “High performance 2-D parallel block filtering system for real-time imaging applications using the SHARC ADSP21060”, Elsevier Science, Real Time imaging journal, vol. 9 (2), 2003, pp. 151-161.

[13] X. Li, G. Qian, Block size considerations for multidimensional convolution and correlation, IEEE Trans. Signal Process. Vol. 40, 1992, pp. 1271-1273.

[14] A. D. Cenzo, “Transform length for correlation and convolution”, IEEE Trans. Acoust. Speech Signal processing, ASSP-35, 1987, pp. 698-699.

[15] S. R. Wang, and P. Siy, “Parallel-decomposition algorithm for discrete computational problems and its application in developing an efficient discrete convolution algorithm”, IEE Proc. Vis. Image and Signal Process. Vol. 142 (1), 1995, pp. 40-46.

[16] “ASP-P15 User Manual”. TRANSTECH parallel DSP systems, available on (www.transtech-dsp.co.uk/sharc/asp-p15.htm).