INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 9, Number 2, Pages 401–409

VECTOR-MATRIX MULTIPLICATION IN TERNARY OPTICAL COMPUTER

YI JIN, XIAN-CHAO WANG, JUN-JIE PENG, MEI LI, ZHANG-YI SHEN, AND SHAN OU-YANG

Abstract. This paper proposes a new means to complete the optical vector-matrix multiplication (OVMM). The OVMM is implemented on a novel optical computing architecture, ternary optical computer (TOC) by using modified signed-digit (MSD) number system. In addition, this study realizes optical addition in three steps, independent of the length of the operands, by four transformations in MSD, and discuses the multiplication of two MSD numbers in detail. A preliminary experimentation about the OVMM is performed on TOC. It illustrates that the proposed method for OVMM is feasible and correct.

Key words. parallel, modified signed-digit, no-carry addition, basic operating unit, partial sum

1. Introduction

Optical vector-matrix multiplication (OVMM) was proposed by Lee et al. in 1970[1]. Since then, many optical means have been put forward to implement OVMM[2-23]. However, most of them are analog computing systems only using light intensity, so their apparatus and experimental conditions are very sensitive to noise. In order to improve the accuracy, some algorithms and technologies were applied to OVMM, such as digital multiplication by analog convolution[7, 12], twos complement arithmetic[7, 11], error-correcting codes [6, 9], time-and-space integrating[12, 23], and digital partitioning[9]. In 2009, Li et al realized binary OVMM in ternary optical computer(TOC)[22]. However, Li's work was short of application because every element in the matrix and vector was only one bit.

On the other hand, numerical redundant representations have been used in no-carry addition or other arithmetic operations in the past several decades. It is most worth to mention that in those redundant representations, the modified signed-digit (MSD) number system is often used in optical computing[24, 25, 26]. The MSD, using three digits, -1, 0, and 1 with radix-2, has been proved to be well suited for realizing no-carry addition and other arithmetic operations. In the number system, the carry propagation occurs only between the neighboring positions in the addition operation and the results can be obtained through three logical steps by four logic operations. Therefore MSD is a parallel computing means. This paper will propose a method for immplementing OVMM via MSD number system in TOC.

The remainder is organized as follows. Section 2 discusses the main technological foundations. Section 3 is the theory of the proposed method. Section 4 gives an experiment to verify the method. And the last section is a summary.

Received by the editors October 19, 2009 and, in revised form, March 22, 2010.

2000 *Mathematics Subject Classification.* 65Y05, 65Y10, 68Q10, 68W10.

This work was supported by the National Natural Science Foundation of China(61073049), the Ph.D. Programs Foundation of the Ministry of Education of China(20093108110016), the Shanghai Leading Academic Discipline Project(J50103), the Research Project of Excellent Young Talents in the Universities in Shanghai(B.37-0108-08-002).

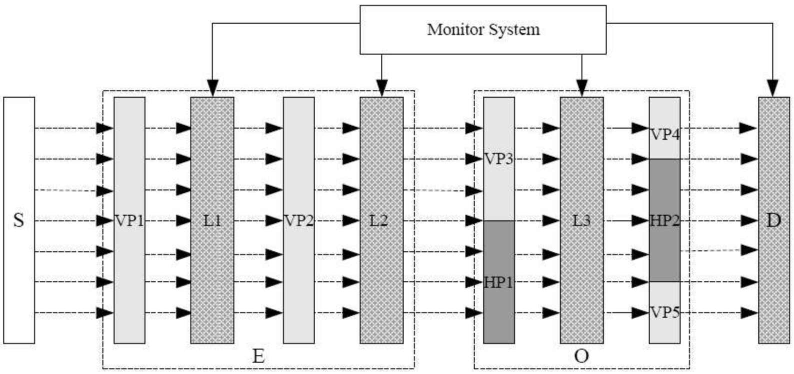


Figure 1. Architecture of the optical part in TOC.

2. Foundations

2.1. Optical Core of TOC Experimental System. The principle and architecture of TOC was proposed by Jin[28, 29]. In TOC, three steady states of light, no intensity light (NIL), horizontal polarization light (HPL) and vertical polarization light (VPL), are used to represent tri-valued information. In 2007 an experimental TOC system was built in Shanghai University. In the experimental system there is an optical core made up of a light source, 5 pieces of vertical polarizer (VP1, VP2, VP3, VP4 and VP5), 2 pieces of horizontal polarizer (HP1 and HP2), 3 monochromatic liquid crystal arrays (L1, L2 and L3) and 1 photoreceptor array (D), shown in Figure 1. The core can be divided functionally into four components,a light source (S) , an encoder (E, including VP1, L1, VP2 and L2), a processor (O, containing VP3, HP1, L3, VP4, HP2, VP5 and L3), and a decoder (D).

The optical processor (O) has four partitions from top to down, which are called respectively VV, VH, HV, and HH, shown in Figure 1. Each partition has the same structure: two pieces of polarizer holding a liquid crystal array in between, seeming like a sandwich. If the partition has a vertical polarizer in the front and a horizontal one in the back, it is called a VH partition. Other namings of the partitions follow similar rules.

2.2. Decrease-Radix Design Principle. In 2007, reference[30] proposed the decrease-radix design principle (DRDP). With this principle, each of n^{n^2} n-valued arithmetic unit without carry or borrow can be constructed by combining several simplest hardware units if there exists a special one (it is called D state in our work) included in the n physical states used to represent information. The sort of the simplest hardware units is $n^2\,(n-1)$, and they are called the basic operating-units(BOUs).

Applying the DRDP to TOC, we can get the results as follows:

- Any two-input tri-valued logic processor can be reconfigured by no more than 6 BOUs at any moment. And the total of the two-input tri-valued logic processors is 19683.
- There are 18 kinds of BOUs in total.
- Each BOU is made of a liquid crystal pixel and a piece of polarizer on each side of it(see “O” in Figure 1).

2.3. MSD Addition. MSD was proposed by Avizienis in 1961[27], and was first used on optical computing by Draker in 1986. A number x can be represented in MSD by the following equation:

$$x = \sum_i x_i 2^i, \qquad x_i \in \{\bar{1}, 0, 1\}, \tag{1}$$

where $\bar{1}$ denotes -1. In this paper, only MSD integer arithmetic operations are discussed. A number has several expression forms in MSD. For example,
$(4)_{10} = (100)_{2 \text{ or MSD}} = (1\bar{1}00)_{\text{MSD}} = (1\bar{1}\bar{1}00)_{\text{MSD}}$,
and
$(-4)_{10} = (\bar{1}100)_{\text{MSD}} = (\bar{1}1100)_{\text{MSD}}$.

From the example, we can know that MSD is a number system with redundancy. The redundancy provides the possibility of no-carry addition in MSD. No-carry addition can be implemented by using four logical operations, called T, W, T′, and W′ transformations, whose truth tables are shown in Table 1. The MSD addition can be explained by an example as follows.

T	$\bar{1}$	0	1	W	$\bar{1}$	0	1	T´	$\bar{1}$	0	1	W´	$\bar{1}$	0	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	0	$\bar{1}$	0	1	0	$\bar{1}$	$\bar{1}$	0	0	$\bar{1}$	0	$\bar{1}$	0
0	$\bar{1}$	0	1	0	1	0	$\bar{1}$	0	0	0	0	0	$\bar{1}$	0	1
1	0	1	1	1	0	$\bar{1}$	0	1	0	0	1	1	0	1	0

TABLE 1. *Truth tables of four transformations used in MSD addition.*

Assume two MSD numbers x and y, $x = 1\bar{1}0\bar{1}10$, $y = 100\bar{1}0000$.

Step 1: Carry out logical operations T and W on x and y, and obtain their results, t and w, in parallel.
By T, $t = 101\bar{1}0\bar{1}10\phi$.
By W, $w = \phi\bar{1}0\bar{1}001\bar{1}0$.
Where ϕ is a padded zero.

Step 2: Carry out logical operations T′ and W′ on t and w, and obtain their results, t' and w', in parallel.
By T′, $t' = 000\bar{1}00100\phi$.
By W′, $w' = \phi1\bar{1}100\bar{1}0\bar{1}0$.

Step 3: Carry out logical operation T on t' and w', and obtain the final sum s of x and y.
$s = 01\bar{1}00000\bar{1}0$.

From the example, we can find that the no-carry addition can be implemented by four transformations in three logic steps which are independent of the length of operands. And the four kinds of logical operation units can be reconfigured by a large number of BOUs in TOC at any moment by DRDP. So the no-carry addition can be implemented by hardware in TOC.

2.4. MSD Multiplication. If $a = a_{n-1} \ldots a_1 a_0$ and $b = b_{n-1} \ldots b_1 b_0$, their product p is governed by the following equation:

$$p = ab = \sum_{i=0}^{n-1} p_i 2^i = \sum_{i=0}^{n-1} ab_i 2^i \tag{2}$$

It is clear that if b_i is 0, p_i is also 0; if b_i is 1, p_i is a; and if b_i is $\bar{1}$, p_i is -a. So all $p_i (i = 0, \ldots, n-1)$ can be written out immediately from a_i and b_i in parallel by using a transformation. This transformation is also a tri-valued logic operation and marked with M. And then, the multiplication will be completed by adding up all $ab_i2^i(i = 0, \ldots, n-1)$. Thus the multiplication will be implemented in parallel by hardware of TOC because all additions can be achieved by using the MSD additions in Section 2.3. An example is used to describe the operating process.

M	$\bar{1}$	0	1
$\bar{1}$	1	0	$\bar{1}$
0	0	0	0
1	$\bar{1}$	0	1

TABLE 2. *Truth tables of M transformation.*

Assume $a = (14)_{10} = (1110)_{\mathrm{MSD}}$, $b = (9)_{10} = (101\bar{1})_{\mathrm{MSD}}$ then

$$p = ab = \sum_i p_i 2^i = \sum_i ab_i 2^i = (1110)_{\mathrm{MSD}} \times (101\bar{1})_{\mathrm{MSD}}$$

$$= (1110)_{\mathrm{MSD}} \times \bar{1} + (1110)_{\mathrm{MSD}} \times 1\phi + (1110)_{\mathrm{MSD}} \times 0\phi\phi + (1110)_{\mathrm{MSD}} \times 1\phi\phi\phi$$

$$= \overline{111}0 + 1110\phi + 0\phi\phi + 1110\phi\phi\phi.$$

Where the padded ϕs indicate $(\times\ 2^i)$. In other words,

$$p = ab = \sum_i p_i 2^i = \sum_i ab_i 2^i$$

$$= (p_0 \times 2^0 + p_1 \times 2^1) + (p_2 \times 2^2 + p_3 \times 2^3)$$

$$= (\overline{111}0 + 1110\phi) + (0\phi\phi + 1110\phi\phi\phi).$$

And then, the additions in two brackets can be completed via MSD means in parallel. After this, the last addition will be also completed by MSD addition. Because the TOC has a large number of data-bits, all sums ($p_i2^i + p_{i+1}2^{i+1}$) can be completed in different regions of data-bits of optical processor in one machine instruction.

3. MSD Vector-Matrix Multiplication

If α is a $1\times N$ row vector and β is a $N \times N$ matrix , their product ψ is also a row vector and expressed as follows:

$$\psi = \alpha\beta = (\psi_1, \psi_2, \ldots, \psi_N) = (\sum_{i=1}^{N} \alpha_i\beta_{i1}, \sum_{i=1}^{N} \alpha_i\beta_{i2}, \ldots, \sum_{i=1}^{N} \alpha_i\beta_{iN}). \tag{3}$$

Obviously, from (3), all products $\alpha_i\beta_{ij}(i, j = 1, \ldots, N)$ can be computed with the MSD multiplication and adding them up can be achieved with the MSD addition. Therefore the vector-matrix multiplication can be completed via a MSD means. The process of computing one of the vector-inner-products (VIPs) is shown in Figure 2. In Figure 2, $\psi^i_{j,k}$ is the jth partial sum of Step i in computing the VIP $\psi_k(k = 1, \ldots, N)$. And q in Figure 2 is the maximum step-order in computing partial sums. It is easy to understand that q is equal to $\log_2 N - 1$ or the greatest integer which is less than $\log_2 N$. The broken-line boxes are padded 0 when the total of partial sums is odd in each step of computing the next-step partial sums.

From the Figure 2, it can be seen that it is uncorrelated to compute all partial sums in each step. Therefore, all partial sums in the same step can be computed in parallel on a supercomputer or an optical computer. On the other hand, the TOC experimental system has three hundred data-bits and next-generation TOC system will have thousands of data-bits. Consequently, all partial sums in a computing step can be computed in parallel for normal task by utilizing these data-bits of

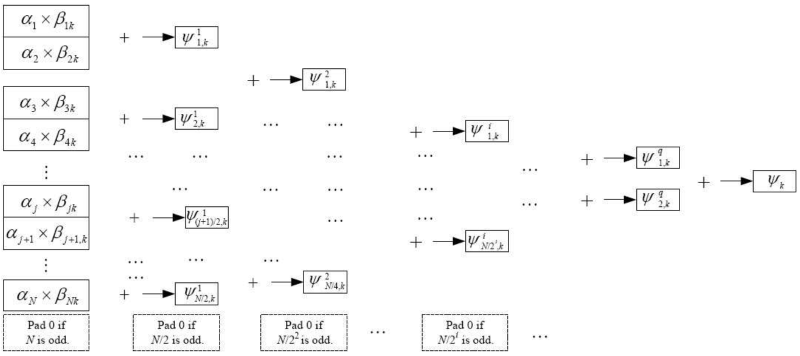


FIGURE 2. The process of computing a VIP, that is, the element ψ_k of vector-matrix product.

TOC. So the vector-matrix multiplication can be completed in $\lceil \log_2 N \rceil$ steps if all the $\alpha_i\beta_{ij}(i,j = 1,\ldots,N)$ are available.

4. Details of Implementing OVMM on TOC and the Experimental Results

The technique of computing a vector-matrix multiplication by TOC can be described through an experiment as follows. In the experiment the vector α and matrix M are as follows:

$$\alpha = \begin{pmatrix} \alpha_1 & \alpha_2 \end{pmatrix} = \begin{pmatrix} 3 & 1 \end{pmatrix}_{10} = \begin{pmatrix} 11 & 1\overline{1} \end{pmatrix}_{\text{MSD}},$$

$$\beta = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix} = \begin{pmatrix} -1 & 2 \\ -2 & -3 \end{pmatrix}_{10} = \begin{pmatrix} \overline{1}1 & 10 \\ \overline{1}0 & \overline{1}\overline{1} \end{pmatrix}_{\text{MSD}},$$

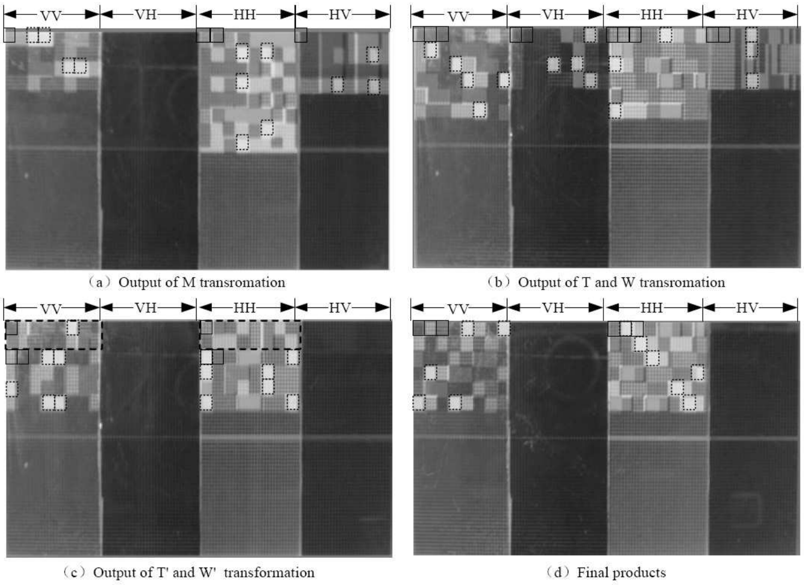
and

$$\psi = \alpha\beta = \begin{pmatrix} 11 & 1\overline{1} \end{pmatrix}_{\text{MSD}} \times \begin{pmatrix} \overline{1}1 & 10 \\ \overline{1}0 & \overline{1}\overline{1} \end{pmatrix}_{\text{MSD}}.$$

The process of computing all the VIPs can be divided into two sequential parts. The former one computes all products of $\alpha_i\beta_{ij}$ and the latter one computes all VIPs $\psi_j(j = 1, 2)$ from the products.

4.1. Computing the Products of α_i and β_{ij} in TOC. In the experiment there are four products of α_i and β_{ij}, that is, $\alpha_1\beta_{11}$, $\alpha_2\beta_{21}$, $\alpha_1\beta_{12}$, and $\alpha_2\beta_{22}$. As α_i and $\beta_{ij}(i,j = 1, 2)$ are two bits, each product needs two 4-bit M. In other words, eight 4-bit M units are needed in total. Because the TOC has enough data-bits, the eight 4-bit M units will be structured in different areas of the optical processor in TOC(see Section 2.1). After all data being input, the eight 4-bit M units will be completed in one operation clock. Therefore, it is a parallel computing which is very different from electric computer.

Following M transformation, MSD addition is activated to compute the products of α_i and β_{ij}. For each β_{ij} being two bits, each product only needs one addition. Consequently, independent four additions will be needed. And each addition sequentially needs a 4-bit T and a 4-bit W, a 4-bit T′ and a 4-bit W′, and a 4-bit T. In other words, computing all products successively needs four 4-bit T and four 4-bit W, four 4-bit T′ and four 4-bit W′, and four 4-bit T. These additions can be achieved in different areas of the optical processor of TOC in parallel in three operation clocks. The concrete steps taken to compute these products are as follows.



（a） Output of M transromation

（b） Output of T and W transromation

（c） Output of T' and W' transformation

（d） Final products

FIGURE 3. Output of optical processor in each step taken to compute the products of corresponding elements. To observe,real-line boxes mark the BOUs used by the first bit in each part, broken-line boxes mark the BOUs which light can pass through(LPBOU), and each BOU consists of 16 neighboring pixels. If the LPBOU is in VV or HV, the output is VPL which denotes 1; if the LPBOU is in VH or HH, the output is HPL which denotes $\bar{1}$; if there is no LPBOU, the output is NIL which denotes 0.

Step 1: In the O, construct eight 4-bit M units. Input the operands to corresponding M units. Run these M units on the optical processor. Get the output from the O. They are displayed in Figure 3(a). The results of the M units are 0011, $0\bar{1}\bar{1}0$, 0000, $0\bar{1}10$, 0000, 0110, $00\bar{1}1$, and $0\bar{1}10$. Store the results into registers and recover data bits used by these M units.

Step 2: In the O, construct four 4-bit T units and four 4-bit W units. Input the results of Step 1 to corresponding T and W units. Run these T and W units on the optical processor. Get the outputs from the O. They are displayed in Figure 3(b). The four group results of T are $0\bar{1}01$,$0\bar{1}10$, 0110 and $0\bar{1}01$, and those of W are $010\bar{1}$, $01\bar{1}0$, $0\bar{1}\bar{1}0$ and $010\bar{1}$.

According to the MSD addition, each group results of T transformation need padding one zero as its least significant bit. We get $0\bar{1}010$, $0\bar{1}100$, 01100, and $0\bar{1}010$ after padding zeros. And the new results are $\bar{1}010$, $\bar{1}100$, 1100, and $\bar{1}010$ after the frontal zero of each group being canceled. Store the results into registers and recover the data bits used by these T and W units.

Step 3: In the O, construct four 4-bit T′ units and four 4-bit W′ units. Input the results of Step 2 to corresponding T′ and W′ units. Run these T′ and W′ units on the optical processor. Get the outputs from the O. They are displayed in Figure 3(c). The results of T′ are 0000, 0100, 0000 and 0000, and those of W′ are $\bar{1}11\bar{1}$, $\bar{1}0\bar{1}0$, $10\bar{1}0$ and $\bar{1}11\bar{1}$. Similarly, the results of T′

are altered into 0000, 1000, 0000, and 0000. Store the results into registers and recover data bits used by these T′ and W′ units.

Step 4: In the O, construct four 4-bit T units. Input the results of Step 3 to corresponding T units. Run these T units on the optical processor. Get the outputs from the O. They are displayed in Figure 3(d). The results of T are $\bar{1}11\bar{1}$, $00\bar{1}0$, $10\bar{1}0$ and $\bar{1}11\bar{1}$. Store the results into registers and recover the data bits used by these T units.

Now, all products of α_i and β_{ij} have been computed in four steps. They are, respectively, as follows:

$$\alpha_1\beta_{11} = (\bar{1}11\bar{1})_{\mathrm{MSD}} = (-3)_{10}, \qquad \alpha_2\beta_{21} = (00\bar{1}0)_{\mathrm{MSD}} = (-2)_{10},$$

$$\alpha_1\beta_{12} = (10\bar{1}0)_{\mathrm{MSD}} = (6)_{10}, \qquad \alpha_2\beta_{22} = (\bar{1}11\bar{1})_{\mathrm{MSD}} = (-3)_{10}.$$

4.2. Computing the VIPs in TOC. According to Section 3, there are two VIPs, ψ_1 and ψ_2, to be computed in the experiment. And

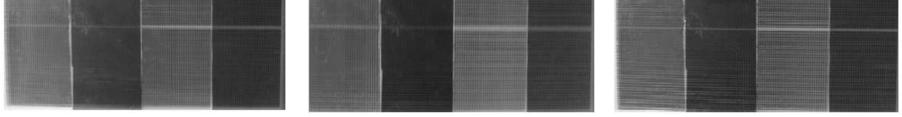
$$\psi_1 = \alpha_1\beta_{11} + \alpha_2\beta_{21}, \qquad \psi_2 = \alpha_1\beta_{12} + \alpha_2\beta_{22}.$$

Therefore, independent two MSD additions will be conducted in the O. The steps are given as follows.

Step 5: For each addition, construct a 4-bit T unit and a 4-bit W unit. Input the products of Step 4 to corresponding T and W units. Run these T and W units on the optical processor. Get the outputs from the O. They are shown in Figure 4(a). The results of T and W are $\bar{1}10\bar{1}$, $010\bar{1}$ and $1\bar{1}01$, $0\bar{1}01$, respectively. After being padded with zeros, the results are $\bar{1}10\bar{1}0$, $010\bar{1}0$ and $01\bar{1}01$, $00\bar{1}01$. Store the results into registers and recover the data bits used by these T and W units.

Step 6: For each addition, construct a 5-bit T′ unit and a 5-bit W′ unit. Input the results of Step 5 to corresponding T′ and W′ units. Run these T′ and W′ units on the optical processor. Get the outputs from the O. They are shown in Figure 4(b). The results of T′ and W′ are 01000, 00000 and $\bar{1}0\bar{1}\bar{1}1$, $01\bar{1}\bar{1}1$, respectively. Store the results into registers and recover data bits used by these T′ and W′ units.

Step 7: For each addition, construct a 5-bit T unit. Input the results of Step 6 to corresponding T units. Run these T units on the optical processor. The final outputs of O are displayed in Figure 4(c).



(a) T and W transromation (b) T′ and W′ transromation (c) The two VIPs

FIGURE 4. Outputs of optical processor in each step taken to compute VIPs from the products of corresponding elements.

From the last outputs of O, the results are $00\bar{1}\bar{1}1$, $01\bar{1}\bar{1}1$. That is

$$\psi = \alpha\beta = (\psi_1 \quad \psi_2) = (00\bar{1}\bar{1}1 \quad 01\bar{1}\bar{1}1)_{\mathrm{MSD}} = (-5 \quad 3)_{10}.$$

The experiment demonstrates the feasibility and correctness of the proposed OVMM on TOC.

5. Conclusions

This paper has implemented a method for OVMM by five transformations—T, W, T′, W′ and M—in MSD number system on TOC experiment platform. In the OVMM, seven steps are needed to compute a 1×2 row vector and a 2×2 matrix whose elements are all 2 bits. The OVMM has two obvious advantages as follows:

- It obtains the partial sums and VIPs in parallel.
- It is easily expanded to thousands of bits.

The operating speed isn't discussed in detail because of the limitation of experimental condition. At the same time, the utilization of hardware is also not considered here.

References

[1] R. A. Heinz, J. O. Artman and S. H. Lee, Matrix Multiplication by Optical Methods, Appl. Opt. 9(1970), 2161-2168.

[2] J. W. Goodman, A. R. Dias and L. M. Woody, Fully parallel, high-speed incoherent optical method for performing discrete Fourier transforms,Opt. Lett. 2(1978), 1-3.

[3] E. P. Mosca, R. D. Griffin, F. P. Pursel and J. N. Lee, Acoustooptical matrix-vector product processor: implementation issues, Appl. Opt. 28(1989), 3843-3850.

[4] R. P. Bocker, Optical digital RUBIC (rapid unbiased bipolar incoherent calculator) cube processor. Opt. Eng. 23(1984), 26-32

[5] C. J. Perlee and D. P. Casasent, Effects of error sources on the parallelism of an optical matrix-vector processor. Appl. Opt. 29(1990), 2544-2555.

[6] S. A. Ellett, J. F. Walkup and Thomas F. Krile, Error-correction coding for accuracy enhancement in optical matrix-vector multipliers, Appl. Opt. 31(1992), 5642-5653.

[7] N. Q. Ngo and L. N. Binh, Fiber-optic array algebraic processing architectures, Appl. Opt. 34(1995), 803-815.

[8] W. T. Rhodes and P. S. Guilfoyle, Acousto-optic algebraic processing architectures, Proc. IEEE 72(1984), 820-830.

[9] S. Cartwright, New optical matrix-vector multiplier. Appl. Opt. 23(1984), 1683-1684.

[10] D. Casasent and S. Riedl, Direct finite-element solution on an optical laboratory matrix-vector processor, Opt. Commun. 65(1988), 329-333.

[11] A. P. Goutoulis, Systolic time-integrating acousto-optic binary processor, Appl. Opt. 23(1984), 4095-4099.

[12] E. J. Baranoski and D. P. Casasent, High-accuracy optical processors: a new performance comparison, Appl. Opt. 28(1989), 5351-5357.

[13] G. Eichmann, Y. Li, P. P. Ho and R. R. Alfano, Digital optical isochronous array processing, Appl. Opt. 26(1987), 2726-2733.

[14] K. Al-Ghoneim and D. Casasent, High-accuracy pipelined iterative-tree optical multiplication, Appl. Opt. 33(1994), 1517-1527.

[15] P. Le, D. Y. Zang and C. S. Tsai, Integrated electrooptic Bragg modulator modules for matrix-vector and matrix-matrix multiplications, Appl. Opt. 27(1988), 1780-1785.

[16] N. Goto, Y. Kanayama and Y. Miyazaki,Integrated optic matrix-vector multiplier using multifrequency acoustooptic Bragg diffraction, Appl. Opt. 30(1991), 523-530.

[17] C. K. Gary, Matrix-vector multiplication using digital partitioning for more accurate optical computing, Appl. Opt. 31(1992), 6205-6211.

[18] P. Yeh and A. E. T. Chiou, Optical matrix-vector multiplication through four-wave mixing in photorefractive media, Opt. Lett. 12(1987), 138-140.

[19] M. Gruber, J. Jahns and S. Sinzinger, Planar-integrated optical vector-matrix multiplier, Appl. Opt. 39(2000), 5367-5373.

[20] S. Zhang and M. A. Karim, Real-time digital optical matrix multiplication with a joint-transform correlator, Appl. Opt. 38(1999), 399-408.

[21] S. A. Ellett, T. F. Krile and J. F. Walkup, Throughput analysis of digital partitioning with error-correcting codes for optical matrix-vector processors, Appl. Opt. 34(1995), 6744-6751.

[22] M. LI, Y. Jin and H. C. He, A New Method for Optical Vector-Matrix Multiplier, International Conference on Electronic Computer Technology (ICECT 2009), 2009, 191-194.

[23] D. Casasent and B. K. Taylor, Banded-matrix high-performance algorithm and architecture, Appl. Opt. 24(1985), 1476-1480.

[24] B. L. Draker, R. P. Bocker, M. E. Lasher, R. H. Patterson and W. J. Miceli, Photonic Computing Using the Modified Signed-Digit Number Representation, Opt. Eng. 25(1986), 038-043.

[25] A. K. Cherri and M. S. Alam, Algorithms for optoelectronic implementation of modified signed-digit division, square-root, logarithmic, and exponential functions, Appl. Opt. 40(2001), 1236-1243

[26] G. Q. Li, F. Qian, H. Ruan and L. R. Liu, Compact parallel optical modified-signed-digit arithmetic-logic array processor with electron-trapping device, Appl. Opt. 38(1999), 5039-5045

[27] A. Avizienis, Signed-digit number representations for fast parallel arithmetic, IRE Trans. Electron. Comp. EC-10(1961), 389-400.

[28] Y. Jin, H. C. He and Y. T. LÜ, Ternary Optical Computer Principle,Sci. China (Ser. F), 46(2003), 145-150.

[29] Y. Jin, H. C. He and Y. T. LÜ, Ternary Optical Computer Architecture, Physica Scripta, 2005, 98-101.

[30] J. Y. Yan, Y. Jin and K. Z. Zuo, Decrease-radix design principle for carrying/borrowing free multi-valued and application in ternary optical computer, Sci. China(Ser. F), 51(2008), 1415-1426.

School of Computer Engineering and Science and High Performance Computing Center, Shanghai University, Shanghai 200072, China
E-mail: yijin@shu.edu.cn,jjie.peng@shu.edu.cn,ou-yang-shan@163.com

School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China

School of Mathematics and Computational Science, Fuyang Normal College, Fuyang 236041, China
E-mail: wxcdx@126.com

College of Computer Science & Engineering, Northwestern Polytechnical University, Xi'an 710072, China
E-mail: plumlee@nwpu.edu.cn

School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China

Computer Science Department, Luoyang Normal University, LuoYang 471022, China
E-mail: willian_shen1@163.com