Parallel implementation of optical symbolic substitution logic using shadow-casting and polarization

Ahmed Louri

We propose a parallel implementation method for optical symbolic substitution logic. The method uses shadow-casting principles for the efficient implementation of fundamental operations required in symbolic substitution logic; namely, image replication, spatial shifting, and combination. The use of light polarization allows for the implementation in parallel of several substitution rules without replicating the input image. The distinctive features of the method include light efficiency, flexibility, cascadability, and programmability. *Key words:* Optical symbolic substitution logic, shadow-casting, polarization, parallel processing, light efficiency, programmability, cascadability.

I. Introduction

The escalating demands for processing power and speed in a wide range of numerical and symbolic applications are placing stringent demands on computer system design. It is generally agreed that significant improvements in computer performance in the future can only be achieved through exploitation of parallelism at all machine organization levels (architecture and algorithm designs). A key issue in the design of parallel processing systems is their ability to provide adequate support for interprocessor and processor memory communications. As it turns out, communications (interprocessor and processor memory) can be the deciding and limiting factor in cost and performance of parallel processing machines. However, conventional electronic technology seems to be reaching its fundamental physical limits and therefore is unable to provide adequate architectural support for high speed and massively parallel processing.[1,2]

Optics, due to its inherent parallelism, high temporal–spatial bandwidths, and noninterfering communications, has the potential of breaking through the performance barriers faced by conventional technology and is therefore under serious consideration for implementing future high performance parallel computers.[3–5] As a result, several optical computing techniques and architectures, with varying degrees of computational efficiency and design complexity have recently emerged (see, for example, Ref. 6). Extensive research and laboratory experiments are under way to assess the validity and merits of these novel techniques. Among the wide variety of proposed techniques, optical symbolic substitution logic (SSL)[7] and optical array logic[8] (shadow-casting) have gained wide popularity among the research community.

We introduce here a hybrid computation method that combines SSL, shadow-casting principles, and the parallelism of light polarization. First, architectural merits and implementation requirements of SSL are discussed, then a new implementation method based on shadow-casting principles and light polarization is proposed for its efficient realization. The main distinctive features of the new method are simultaneous implementation of several substitution rules, energy efficiency, cascadability, and, most important, programmability.

II. Computational Merits and Implementation Requirements of SSL

This section is intended to highlight briefly the basic concept of SSL, its applicability range, and its optical implementation requirements.

A. Basic Principles of SSL

Symbolic substitution logic[9] is a pattern transformation design technique for performing digital logic optically. It uses both the temporal–spatial bandwidths and the high connectivity of optics for constructing digital optical computing systems. The motivation behind this computing technique is the exploitation of

---

The author is with University of Arizona, Department of Electrical & Computer Engineering, Tucson, Arizona 85721.

Received 18 December 1989.

0003-6935/91/050540-09$05.00/0.

the massive fine grain parallelism and the regular and space-invariant connectivity of optics. In optics, it is relatively easy to move and operate on optically encoded data in a regular fashion rather than in a random manner. Hence, SSL is one possible way to perform computations with constant fanin, constant fanout, and regular (space-invariant) interconnections. In this method, data are encoded as spatial patterns and operators are seen as pattern transformation rules or substitution rules. In its operation, SSL consists of two pattern processing steps. The first step is a recognition phase whereby all the occurrences of a search pattern (representing the left-hand side of a substitution rule) are simultaneously searched in the input plane. This is followed by a substitution phase whereby a different pattern (representing the right-hand side of the substitution rule) is substituted in all the locations where the search pattern is found.

B. Applicability

Thus far, SSL has been proposed for a wide range of applications, including digital logic and arithmetic operations,[9–12] signal and image processing,[13–16] massively parallel computing,[17] and symbolic artificial intelligence computing.[18–20] This rapid spread of SSL has generated a series of comparisons.[21,22] In Ref. 21 it was concluded, among other things, that SSL is not a valid model for parallel computations because of the difficulty of applying it to applications that exhibit global communications (i.e., numerical transforms, sorting, and searching). The argument went even further by claiming that mesh connected architectures (one of which is SSL) are not efficient architectures for parallel processing.

As described above, the essence behind SSL is compatibility with optics characteristics, that is, the easy (hardware) support of parallel, regular space-invariant network topologies. Therefore, it is evident that SSL will be a local communication oriented computing model. The real question is whether parallel architectures designed around local communication network topologies are efficient. In my view, the answer is yes because an optimal architecture (in terms of communications and processing power) is application dependent. Not all real world computing applications require multiple instruction control of multiple data (MIMD) computations and global communications. There is an abundance of applications where single instruction control of multiple data (SIMD) computing and local communications are the most suitable. Data parallel computing, where the parallelism comes from the simultaneous operation across large sets of data rather than from multiple threads of control, has been proposed as the most suitable class for SIMD fine grained parallel processing.[23] Some areas of this class that are still overburdening existing electronic technologies include image processing, radar signal processing, image analysis, low level vision processing (pattern recognition and classification phases), optimization processes (branch and bound algorithms), partial differential equations (finite element analysis, numerical integrations), and some artificial intelligence problems such as production systems, mathematical resolution, and unification problems.

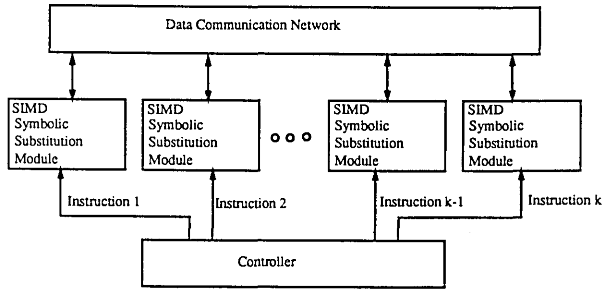


Fig. 1. Block diagram of a parallel architecture formed by multiple control of multiple single-instruction-stream multiple data (MIMSIMD) symbolic substitution modules.

The proliferation of commercial mesh connected computers is yet another evidence that SIMD computing is a well-established and accepted field of parallel processing. Some of these machines include the NASA massively parallel processor (MPP),[24] the ICL DAP,[25] the CLIP,[26] the GRID,[27] the AAP,[28] and to some extent the Connection Machine[29] (at the low level, every sixteen processing elements are mesh connected and form a cluster; at the high level, the clusters communicate via a router network which is configured as a Boolean n-cube). Granted that SSL may not be an efficient computing model for all-purpose parallel computing, it is still a viable and a better alternative model for SIMD and multiple control of multiple SIMD modules (MIMSIMD). This latter is a subset of the MIMD computation model. Multiple control of multiple SIMD can be achieved by operating several SIMD modules on different operation streams as shown in Fig. 1.

C. Implementation

The author agrees with the conclusion in Ref. 21 relating to power breakdown of the current optical implementations of SSL. The most popular algorithm for implementing SSL is based on additive logic and consists of applying a thresholding (nonlinear operation) to a composite of shifted copies of the input plane.[9] The algorithm is simple and general. However, its known implementation methods[9,30] are power inefficient and lack computational flexibility (namely, architectural flexibility, cascadability, and programmability).

In its operation, and assuming bright pixel recognition, the additive logic algorithm consists of designating a reference pixel in the search pattern, replicating the input plane as many times as there are bright pixels in the search pattern, shifting the replicated images in such a manner that their associated bright pixel is aligned with the reference pixel, superimposing them, and thresholding the resulting image. The substitution phase is functionally similar to the recognition process except for thresholding. The replication of images constitutes a major source of power loss. The situation is even worse for implementing several rules in parallel. In this latter case, the input image is replicated into as many copies as there are rules to be processed in parallel. Then each rule splits the image according to the number of dark/bright pixels in its search/replacement pattern. It should be noted that there are other novel optical implementation methods under investigation which are not based on additive logic.[31–34] The computational merits of these methods are yet to be determined. A critical study is under way to determine the performance and complexity of the various implementation methods that have been proposed for SSL.[35]

This paper is an attempt to contribute to the ongoing efforts in finding power efficient implementation means for SSL that are also architecturally flexible, programmable, and cascadable. In this vein, the rest of the paper introduces a parallel implementation method based on shadow-casting principles and light polarization.

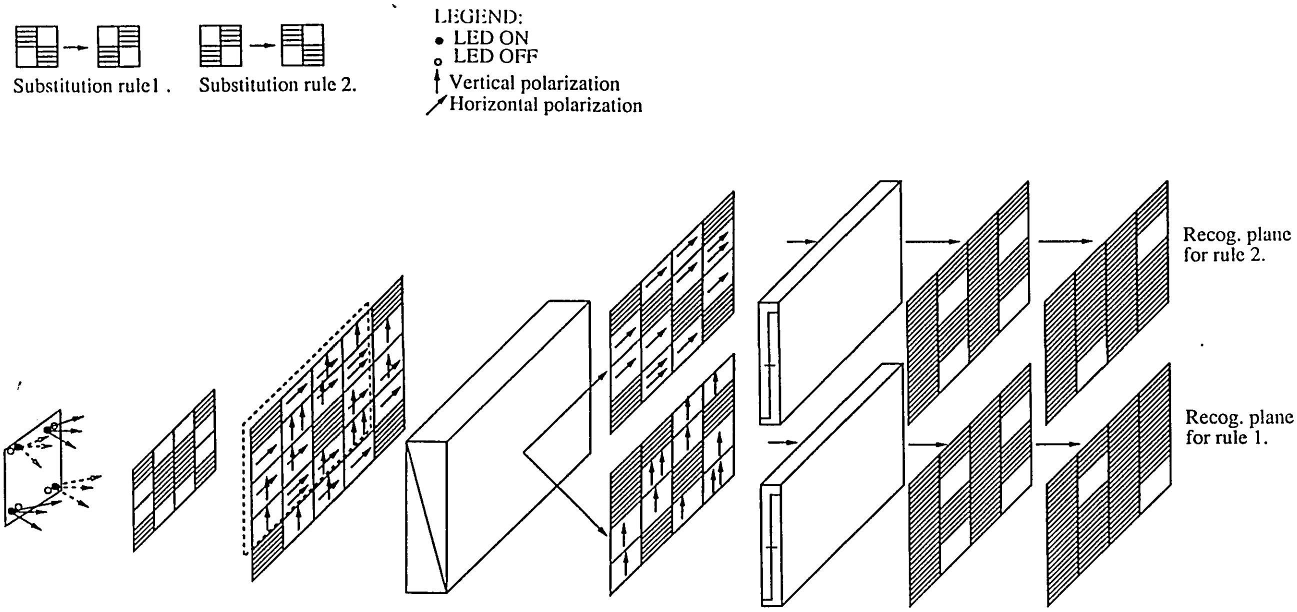
III. Parallel Implementation of SSL

In Ref. 36 a power efficient and flexible method was presented for implementing SSL using the additive logic algorithm. The basic idea was to use shadow-casting principles[8] to provide input replication, spatial shifting, and combination. A somewhat similar idea to the one presented in Ref. 36 was reported independently elsewhere.[37] A shadow-casting system is composed of a source plane (a set of LED arrays), an input plane, and an output plane or screen.[8] Diverging light beams from the LEDs pass through the input plane and produce overlapped shadows of the input plane on the screen. By choosing the spacing between the LEDs and distance from the source plane to the input plane and from input plane to the screen, one can obtain a number of replicated and shifted copies of the input plane superimposed on the screen. The number of replicas and the amount of shift are a function of the ON–OFF switching states of the LEDs. The states of the LEDs are in turn dictated by the structure of the search and replacement patterns of the substitution rules. In Ref. 36, the method was shown to implement SSL systems with both single rail as well as dual rail coding.

A. Simultaneous Implementation of Two Substitution Rules

The shadow-casting system, as proposed by Tanida and Ichioka,[8] was extended by Li *et al.*[38] to include polarized input pixel coding and output mask transparency, to permit the generation of multivariable logic as well as multiple valued logic functions. Recently, Karim *et al.*[39,40] presented several systematic algorithms for designing complex arithmetic logic units using the extended shadow-casting scheme. We explore here the polarization of the source plane (the use of polarized LEDs) for the parallel implementation of SSL. Using the two orthogonal polarization states of light, several substitution rules can be implemented simultaneously. In the following, I first describe the parallel implementation of two substitution rules, then extend it to the implementation of four rules in the next section. Without loss of generality, I discuss substitution rules in which the search and replacement patterns consist of 2 × 2 pixels.

Figure 2 illustrates the parallel implementation of the recognition phase of two substitution rules. The



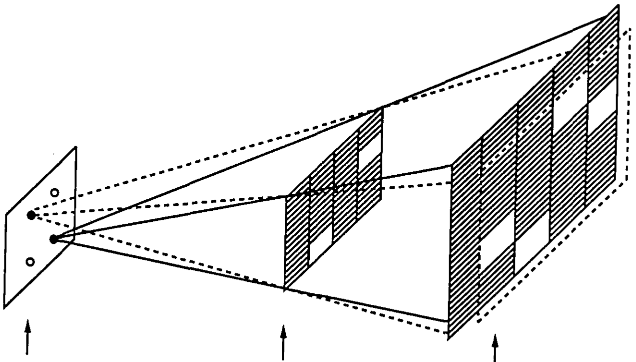
(a) Source plane. (2 orthogonally polarized LED arrays)
(b) Input image.
(c) Superimposed image.
(d) Wollaston prism.
(e) Two copies with distinct polarization.
(f) Thresholding optical AND-gate arrays.
(g) Two masks.
(h) Two recognition planes: one for each substitution rule.

Fig. 2. Parallel implementation of the recognition phase of two substitution rules.

source plane consists of 2 × 2 orthogonally polarized LED arrays [Fig. 2(a)]. Each element of the 2 × 2 array is a pair of orthogonally polarized LEDs that are physically positioned at near coincident points in the source plane. This is equivalent to having two independent LED arrays occupying the same space and able to radiate horizontally and vertically polarized light that simultaneously passes through the input plane. The vertically polarized LED array is responsible for implementing substitution rule 1, while the second LED array is horizontally polarized and is responsible for substitution rule 2. The vertical and horizontal states of polarization are represented by a vertical bar and a horizontal bar, respectively. The reference pixel (that will later indicate the presence and location of the search pattern) is chosen to be the lower right corner of the search pattern. Each LED array (vertically and horizontally polarized) provides multiple shadowgrams of the input plane such that all the pixels of its associated search pattern overlap in the reference pixel.

The input plane is encoded in light intensity (opaque/transparent coding) as shown in Fig. 2(b). The configuration of the LED arrays produces distinct, shifted copies of the input plane onto the screen. The superimposed image (projected on the screen) consists of pixels containing two horizontal polarizations, pixels containing two vertical polarizations, pixels containing one polarization (vertical or horizontal), and pixels containing both polarizations (indicated by a cross). Pixels containing two vertical polarizations indicate the presence and locations of the search pattern of substitution rule 1. Pixels containing two horizontal polarizations indicate the presence and locations of the search pattern of substitution rule 2. There may be some erroneous pixels (due to pattern overlap) that will be discarded later through masking operations. Thus, the superimposed image constitutes, in effect, two recognition planes sharing the same physical space. This image is then passed through a Wollaston prism.[5] The Wollaston prism consists of two birefringent wedges with their crystal axes orthogonal to each other and also orthogonal to the principal beam direction. The Wollaston prism deflects the two states of polarization in opposite directions, hence producing two physically separate recognition planes. The upper plane contains horizontally polarized pixels, while the lower plane contains vertically polarized pixels [Fig. 2(e)]. It should be noted that there is no power loss in generating these two separate planes. The Wollaston prism splits the image according to the state of polarization only.

The next step in the recognition phase is a thresholding operation through optical AND gate arrays [Fig. 2(f)]. The thresholding operation will make all the pixels with two identical polarizations bright and all the other pixels will remain dark. The thresholded planes are passed through an optical mask [Fig. 2(g)] whose transparent pixels coincide with the location of the reference pixel in the thresholded image. The purpose of the mask is to filter out erroneous pixels.

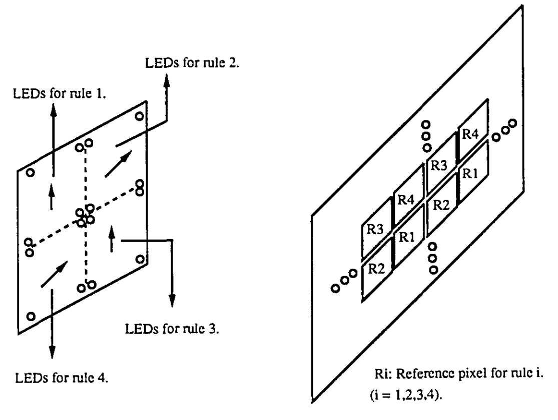


(a) Unpolarized LED array.

(b) Recognition plane for substitution rule 2.

(c) Output plane.

Fig. 3. Implementation of the substitution phase using shadowcasting.



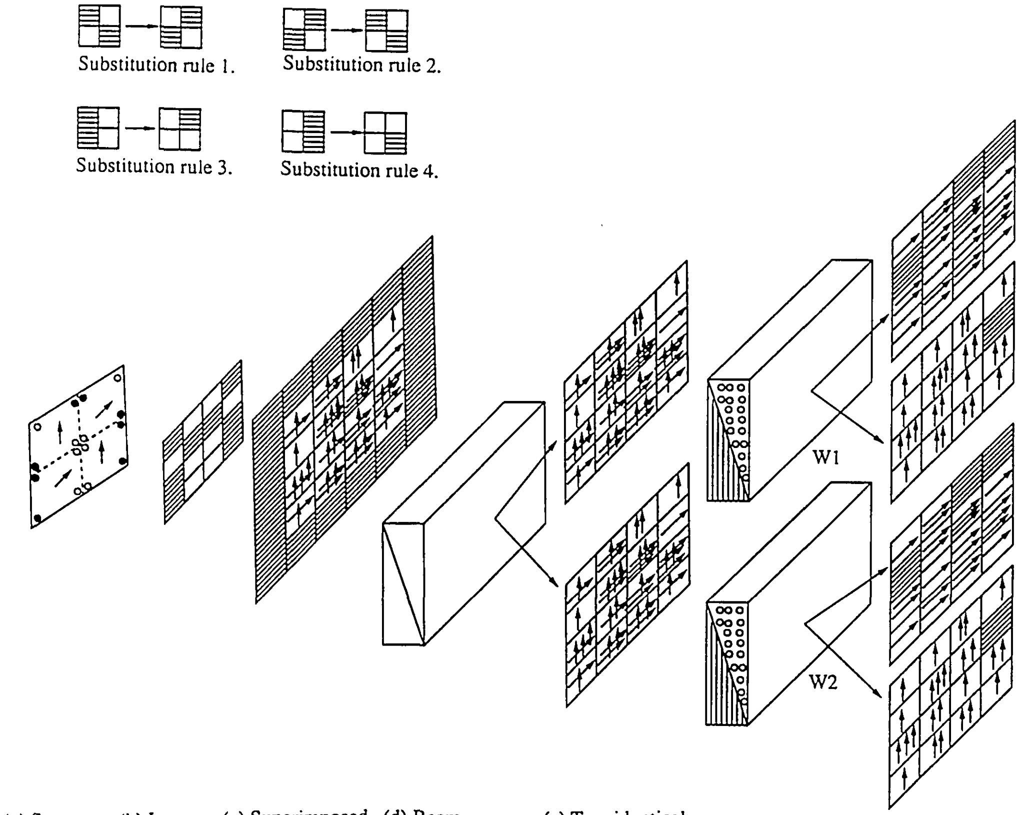
(a) Four LED arrays.

(b) Locations of the reference pixels on the superimposed image.

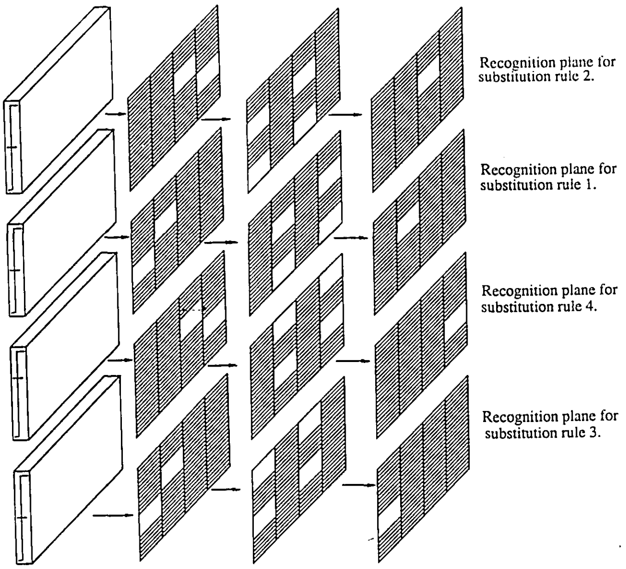
Fig. 4. Source plane configuration and reference pixel arrangement for the parallel implementation of four substitution rules.

Thus, the masked planes contain bright pixels only in the locations of the input plane where the search patterns are found. At this end, we get two recognition planes, one for each substitution rule being implemented [Fig. 2(h)].

Next is the substitution phase. Since we have two separate recognition planes, the substitution of the replacement patterns can also proceed in parallel. In additive logic, the substitution phase consists of replicating the recognition plane as many times as there are bright pixels in the replacement pattern and shifting them so as to scribe the replacement pattern in all the bright locations of the recognition plane. The shifted copies are then combined (ORed) to produce the final output plane. The optical setup to accomplish this is shown in Fig. 3 (the figure shows an optical setup for a single substitution rule). The unpolarized LED array configuration in Fig. 3(a) provides a superimposed image of shifted replicas of the recognition plane. The ON–OFF state of each LED is dictated by the placement of the bright pixels in the substitution pattern. Thus, for a parallel implementation, two distinct LED arrays are required. The replicas are shifted and su-



(a) Source plane. (b) Input image. (c) Superimposed image. (d) Beam splitter. (e) Two identical copies. (f) Two Wollaston prisms.



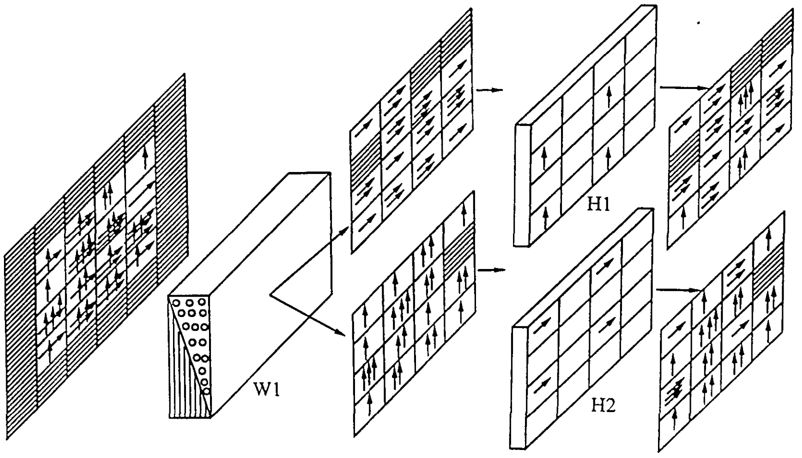
(f) Optical AND-gates arrays. (g) Four distinct masks. (h) Four recognition planes.

Fig. 5. Optical setup for the parallel recognition of four search patterns.

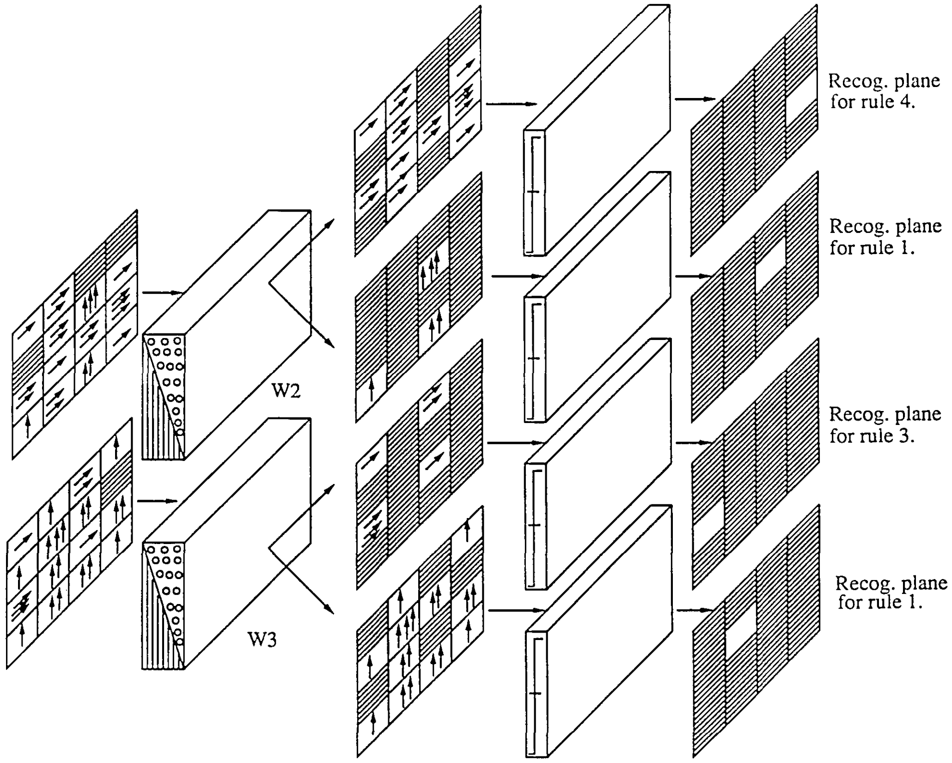
perimposed with the net result that the substitution pattern is scribed in all the occurrences of the search pattern [Fig. 3(c)].

B. Simultaneous Implementation of Four Substitution Rules

Figure 4 shows the source plane and the arrangement of the screen for the simultaneous recognition of four search patterns. The source plane consists of sixteen LEDs arranged in a square array. The LEDs are organized into four 2 × 2 arrays. Each 2 × 2 LED array implements one substitution rule. The geometric configuration is chosen such that each 2 × 2 LED array provides a distinct pixel on the screen where all four patterns of a search pattern can overlap. These pixels are chosen to be the reference pixels of the four



(a) Superimposed image. (b) Wollaston prism. (c) Halfwave plates.



(d) Two Wollaston prisms. (e) Four optical AND-gate arrays. (f) Four recognition planes.

Fig. 6. Modified optical setup for the parallel recognition of four search patterns without any power splitters.

substitution rules to be implemented. Thus, the reference pixels for LED arrays 1, 2, 3, and 4 are the lower right corner, lower left corner, upper right corner, and upper left corner, respectively [Fig. 4(b)]. Two of the LED arrays will be polarized vertically and the other two horizontally [Fig. 4(a)]. The ON–OFF states of each 2 × 2 LED array are dictated by the placement of the bright pixels of the search pattern of the substitution rule associated with it.

The complete four rule recognition setup is shown in Fig. 5. The LED arrays produce a composite image on the screen [Fig. 5(c)] whose pixels contain various states of polarization. This image is duplicated via an unpolarizing beam splitter [Fig. 5(d)] into two copies which are passed through two Wollaston prisms, *W*1 and *W*2 [Fig. 5(f)]. Thus the formation of four images, each with only one type of polarization. These images are impinged on four optical AND gate arrays that produce a bright pixel in all locations having three vertical or three horizontal bars. The outputs of the AND gate arrays are passed through four distinct masks [Fig. 5(h)]. Each mask is associated with one substitution rule, and therefore its transparent pixel coincides with the reference pixel of the corresponding

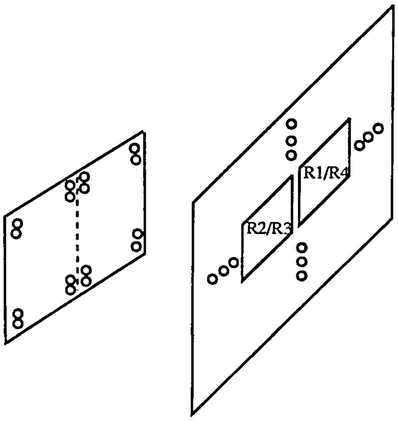


Fig. 7. Alternative optical setup for the parallel recognition of four search patterns which requires only two distinct masks.

substitution rule. The outputs of the masks constitute four recognition planes [Fig. 5(i)]. Each plane indicates the presence and locations of one search pattern. The substitution phase proceeds with four independent channels; one for each recognition plane. The final output is obtained by optically combining the outputs of each substitution channel.

In the setup of Fig. 5, the superimposed image is split via a beam splitter into two copies, each with half of the power of the original superimposed image. This splitting can be avoided if pixel-addressable halfwave plates are used. Since we are using only the two orthogonal polarization states of light, each pair of LED arrays shares the same polarization. The basic idea for the modified scheme is to introduce halfwave plates to produce a physical separation of the recognition planes of the substitution rules that share the same state of polarization.

The modified setup is shown in Fig. 6. The front end (source plane, input plane, and screen) is similar to the setup described above. Instead of splitting the superimposed image into two identical copies, we first pass it through a Wollaston prism, *W*1 [Fig. 6(b)], that splits it into two copies, each holding only one type of polarization. In this example, substitution rules 1 and 3 are associated with vertical polarization and substitution rules 2 and 4 are associated with horizontal polarization. The upper halfwave plate, *H*1 [Fig. 6(c)], inverts the polarization state of the reference pixels of substitution rule 2, while the polarization state of reference pixels of substitution rule 4 are left unchanged. Similarly, the lower halfwave plate, *H*2, inverts the polarization state of reference pixels of substitution rule 3 and does not affect the reference pixels of substitution rule 1. The output of each halfwave plate is an image with both vertical and horizontal polarizations. These images are passed through two Wollaston prisms, *W*2 and *W*3, to generate four images with no power loss [Fig. 6(d)]. In addition, only one in four pixels of the halfwave plates needs to be switched on. This will reduce the demand for extra power to operate the halfwave plates. The four images are then thresholded with optical AND gate arrays and masked as in the previous method.

Another alternative method for four-rule implementation is the extension of the two-rule method presented in Sec. III.A. The source plane would be sixteen LED arrays arranged as shown in Fig. 7; basically, two LED arrays, where each element of the array is in turn two orthogonally polarized LEDs, placed adjacent to each other in the horizontal (or vertical) direction. This setup maps two substitution rules onto the same reference pixel in the output plane (Fig. 7). The advantage of this setup is a reduced number of customized masks. By arranging the source plane as depicted in Fig. 7, only two types of mask are required at the output.

IV. Summary and Conclusions

Despite the space-invariant nature of SSL, it is an attractive and promising model for fine grain parallel processing. Its computational power stems from the fact that it is compatible with optics capabilities. SSL is based on a few basic processing steps (that is, image replication, shifting, and combination) that can be effectively implemented in optics.

Because of its inherent space-invariant connections and regular communications, SSL may not be efficient for implementing general-purpose scalar computations with a low degree of parallelism (e.g., scalar arithmetic), or parallel computations with irregular and global communication patterns. SSL is more suitable for regular and structured computations that exhibit massive data parallelism and heavy local communications. This is not a negative point for SSL since there is an abundance of these applications in real world computing and current (electronic) systems are failing to provide the required levels of performance.

The major problems facing SSL at this time are power requirements and lack of computational flexibility (in terms of programmability and architectural flexibility) of its optical implementation schemes. Part of the problem, namely, the lack of optical gate arrays and spatial light modulators with faster switching time and significantly reduced switching energy, afflicts all digital optical computing systems. Much work is currently being done in this area, and I believe this will be overcome in the near future. The other part of the power problem is due to the fundamental operations required in SSL (beam splitting, shifting, and combining). The implementation method introduced here is an attempt to limit the power loss from the basic operations and provide enough flexibility so as to make the architecture easily programmable and cascadable. Summarizing, the features of the proposed method are as follows:

(1) Energy efficiency: no optical image splitters or analyzers are needed, only polarization beam splitters are used. In addition, more than one substitution rule is implemented in parallel.

(2) Hardware flexibility: the same optical setup can implement different substitution rules by simply

changing (if necessary) the geometric configuration of the source plane.

(3) Programmability: by controlling the switching configuration of the LED arrays, the optical setup can be dynamically reconfigurable to implement different functions. Thus an external control memory that stores microcodes, corresponding to different computable functions, can be used to control the source plane.

(4) Cascadability: the input and output planes are of the same format; therefore, the output image can be used as input to the same system for feedback processing or to a subsequent stage for pipelined computations.

The limitations of the proposed implementation method can be seen as architectural and technological. The architecture is based on the shadow-casting concept and therefore is limited by the principles of geometrical optics. These limits have been studied by Tanida and Ichioka.[8] The technological limitations depend on the nonlinear optical devices to be used such as the polarization devices for thresholding and pixel-addressable halfwave plates for filtering.

This research was supported by NSF grant MIP-8909326. The author would like to acknowledge the fruitful discussions with Drs. Miles Murdocca and Jun Tanida. The constructive comments of the referees were very much appreciated.

References

1. K. C. Saraswat and F. Mohammadi, "Effect of Scaling of Interconnections on the Time Delay of VLSI Circuits," IEEE Trans. Electron Devices **ED-29**, 645–650 (1982).
2. G. M. Amdahl, "Tempered Expectations in Massively Parallel Processing and Semiconductor Industry," in *Technical Digest, Second International Conference on Supercomputing*, Santa Clara, CA (1987).
3. A. A. Sawchuk and T. C. Strand, "Digital Optical Computing," Proc. IEEE **72**, 758–779 (1984).
4. A. Huang, "Architectural Considerations Involved in the Design of an Optical Digital Computer," Proc. IEEE **72**, 780–787 (1984).
5. A. W. Lohmann, "What Classical Optics Can Do for the Digital Optical Computer," Appl. Opt. **25**, 1543–1549 (1986).
6. H. J. Caulfield and G. Gheen, Eds., *Milestone Series, Selected Papers on Optical Computing*, Proc. Soc. Photo-Opt. Instrum. Eng. **1142** (1989).
7. A. Huang, "Parallel Algorithms for Optical Digital Computers," in *Proceedings, IEEE Tenth International Optical Computing Conference* (1983), pp. 13–17.
8. J. Tanida and Y. Ichioka, "Optical Logic Array Processor Using Shadowgrams," J. Opt. Soc. Am. **73**, 800–809 (1983).
9. K.-H. Brenner, A. Huang, and N. Streibl, "Digital Optical Computing with Symbolic Substitution," Appl. Opt. **25**, 3054–3060 (1986).
10. K. Hwang and A. Louri, "Optical Multiplication and Division using Modified Signed-Digit Symbolic Substitution," Opt. Eng. **28**, 364–373 (1989).
11. K.-H. Brenner, M. Kufner, and S. Kufner, "Highly Parallel Arithmetic Algorithms for a Digital Optical Processor Using Symbolic Substitution Logic," Appl. Opt. **29**, 1610–1618 (1990).
12. A. Louri, "Throughput Enhancement for Optical Symbolic Substitution Computing Systems," Appl. Opt. **29**, 2979–2981 (1990).
13. S. D. Goodman and W. T. Rhodes, "Symbolic Substitution Applications to Image Processing," Appl. Opt. **27**, 1708–1714 (1988).
14. K. S. Huang, B. K. Jenkins, and A. A. Sawchuk, "Image Algebra Representation of Parallel Optical Binary Arithmetic," Appl. Opt. **28**, 1263–1278 (1989).
15. A. K. Cherri and M. A. Karim, "Uses of Optical Symbolic Substitution in Image Processing: Median Filters," in *Technical Digest, Topical Meeting on Optical Computing*, Vol. 9 (Optical Society of America, Washington, DC, 1989).
16. M. J. Murdocca, "Digital Optical Computing with One-Rule Cellular Automata," Appl. Opt. **26**, 682–688 (1987).
17. A. Louri, "A Parallel Architecture and Algorithms for Optical Computing," Opt. Commun. **72**, 27–37 (1989).
18. A. D. McAulay, "Optical Prolog Computer Using Symbolic Substitution," Proc. Soc. Photo-Opt. Instrum. Eng. **881**, 223–229 (1988).
19. G. Eichmann and S. Basu, "Parallel Optical Syntactic Pattern Recognizers," Appl. Opt. **26**, 1859–1865 (1987).
20. D. P. Casasent and E. C. Botha, "Multifunctional Optical Processor Based on Symbolic Substitution," Opt. Eng. **28**, 425–433 (1989).
21. F. Kiamilev *et al.*, "Programmable Optoelectronic Multiprocessors and Their Comparison with Symbolic Substitution for Digital Optical Computing," Opt. Eng. **28**, 396–409 (1989).
22. T. J. Cloonan, "Performance Analysis of Optical Symbolic Substitution," Appl. Opt. **27**, 1701–1707 (1988).
23. W. D. Hillis and G. L. Steele, Jr., "Data Parallel Algorithms," Commun. ACM **29**, 1170–1183 (1986).
24. K. E. Batcher, "Design of a Massively Parallel Processor," IEEE Trans. Comput. **C-29**, 836–840 (1980).
25. S. F. Reddaway, "DAP—a Distributed Array Processor," in *Proceedings, First Annual IEEE/ACM Symposium on Computer Architecture*, Miami, FL (1973).
26. M. J. Duff, "*CLIP 4: Special Computer Architecture for Pattern Recognition*," X. Fu and X. Ichikawa, Eds. (CRC Press, Cleveland, 1982).
27. I. N. Robinson and W. R. Moore, "A Parallel Processor Array Architecture and Its Implementation in Silicon," in *Proceedings, IEEE Custom Integrated Circuit Conference* (1982), pp. 41–45.
28. T. Kondo, T. Nakashima, M. Aoki, and T. Sudo, "An lsi Adaptive Array Processor," IEEE J. Solid-State Circuits **SSC-18**, 147–156 (1983).
29. W. D. Hillis, *The Connection Machine* (MIT Press, Cambridge, 1985).
30. K.-H. Brenner, "New Implementation of Symbolic Substitution Logic," Appl. Opt. **25**, 3061–3064 (1986).
31. E. Botha, D. Casasent, and E. Barnard, "Optical Symbolic Substitution Using Multichannel Correlators," Appl. Opt. **27**, 817–818 (1988).
32. K.-H. Brenner, A. W. Lohmann, and T. M. Merklein, "Symbolic Substitution Implemented by Spatial Filtering Logic," Opt. Eng. **28**, 390–395 (1989).
33. F. T. S. Yu, C. Zhang, and S. Jutamulia, "Application of One-Step Holographic Associative Memories to Symbolic Substitution," Opt. Eng. **27**, 399–402 (1988).
34. H.-I. Jeon, M. A. G. Abushagur, A. A. Sawchuk, and B. K. Jenkins, "Digital Optical Processor Based on Symbolic Substitution Using Holographic Matched Filtering," Appl. Opt. **29**, 2113–2125 (1990).
35. A. Louri and A. D. Post, "Digital Optical Computing Techniques: Computational Merits and Implementation Requirements," Submitted to AO.
36. A. Louri, "Efficient Implementation Method for Symbolic Substitution Logic Based on Shadow-Casting," Appl. Opt. **28**, 3264–3267 (1989).

37. W. Xue, L. X. Chen, C. F. Li, and Q. S. Hu, "Symbolic Substitution Using Shadow-Casting," Proc. Soc. Photo-Opt. Instrum. Eng. **963**, 653–656 (1989).
38. Y. Li, G. Eichmann, and R. R. Alfano, "Optical Computing Using Hybrid Encoded Shadow Casting," Appl. Opt. **25**, 2636–2638 (1986).
39. M. A. Karim, A. A. S. Awwal, and A. K. Cherri, "Polarization-Encoded Optical Shadow-Casting Logic Units: Design," Appl. Opt. **26**, 2720–2725 (1987).
40. A. A. S. Awwal and M. A. Karim, "Multiprocessor Design Using Polarization-Encoded Optical Shadow-Casting," Appl. Opt. **29**, 2107–2112 (1990).