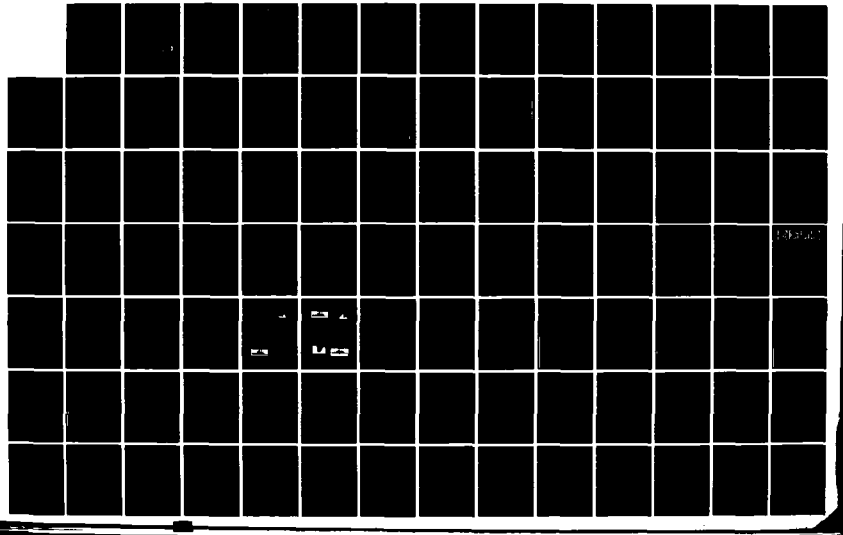
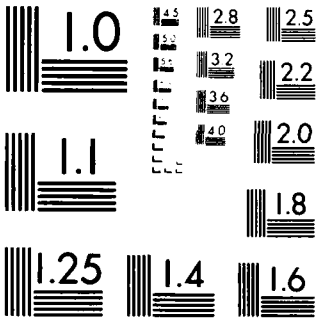
AD-A149 346 OPTICAL DATA PROCESSING FOR MISSILE GUIDANCE(U) 1/2
CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF ELECTRICAL
AND COMPUTER ENGINEERING D CASASENT 21 NOV 84
UNCLASSIFIED AFOSR-TR-84-1162 AFOSR-79-0091 F/G 17/7 NL





FINAL REPORT

OPTICAL DATA PROCESSING FOR MISSILE GUIDANCE

SUBMITTED BY:

David Casasent, Principal Investigator
Carnegie-Mellon University
Deparment of Electrical and Computer Engineering
Pittsburgh, PA 15213
Telephone: (412) 578-2464

SUBMITTED TO:

Air Force Office of Scientific Research
Bolling Air Force Base
Washington, D.C. 20332

ATTN: Lt. Colonel Robert Carter, Building 410

Date: 21 November 1984

Period Covered
30 September 1983 - 30 September 1984

FINAL REPORT

OPTICAL DATA PROCESSING

FOR MISSILE GUIDANCE

SUBMITTED BY:

David Casasent, Principal Investigator
Carnegie-Mellon University
Deparment of Electrical and Computer Engineering
Pittsburgh, PA 15213
Telephone: (412) 578-2464

SUBMITTED TO:

Air Force Office of Scientific Research
Bolling Air Force Base
Washington, D.C. 20332

ATTN: Lt. Colonel Robert Carter, Building 410

Date: 21 November 1984

Period Covered
30 September 1983 - 30 September 1984

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 84-1162 2. GOVT ACCESSION NO. AD-A149 346	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Optical Data Processing for Missile Guidance	5. TYPE OF REPORT & PERIOD COVERED Final Report Sept. 1983 - September 1984
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) David Casasent	8. CONTRACT OR GRANT NUMBER(s) AFOSR-79-0091
9. PERFORMING ORGANIZATION NAME AND ADDRESS Carnegie-Mellon University Dept. of Electrical and Computer Engineering Pittsburgh, PA 15213	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61102F, 2305 D1
11. CONTROLLING OFFICE NAME AND ADDRESS AFOSR/NE, Building 410 Bolling Air Force Base, D.C. 20332 (Lt. Col. Robert Carter)	12. REPORT DATE 21 November 1984
	13. NUMBER OF PAGES 136
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) same	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)
Unlimited

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
Unlimited

18. SUPPLEMENTARY NOTES
None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Acousto-optic, Correlators, Feature extractors, Fourier coefficient, Kalman filtering, Moments, Optical linear algebra processors, Optical pattern recognition, Pattern recognition, Synthetic discriminant functions, Systolic array processors.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
Research on optical data processing for missile guidance and robotics is described. Components addressed include acousto-optic cells. Pattern recognition work includes feature extraction (Fourier coefficients and moments) and correlation (using synthetic discriminant functions). All pattern recognition work concerns multi-class distortion-invariant pattern recognition. Optical linear algebra processors are addressed with attention to: algorithms, architectures, applications, Kalman filtering, system

fabrication, accuracy and performance, plus error source modeling and simulation.

Table of Contents

ABSTRACT

Research on optical data processing for missile guidance and robotics is described. Components addressed include acousto-optic cells. Pattern recognition work includes feature extraction (Fourier coefficients and moments) and correlation (using synthetic discriminant functions). All pattern recognition work concerns multi-class distortion-invariant pattern recognition. Optical linear algebra processors are addressed with attention to: algorithms, architectures, applications, Kalman filtering, system fabrication, accuracy and performance, plus error source modeling and simulation.

1. INTRODUCTION

During the past year (September 1983 - September 1984), our research in optical data processing for missile guidance has addressed many of the key issues and aspects of this technology. This research includes: real-time devices and components, new system architectures, new high-speed general purpose optical data processing techniques and systems, tests on new image data bases, basic studies of existing pattern recognition architectures, and new pattern recognition techniques, algorithms and concepts. As in past years, we have been quite faithful in reporting our AFOSR sponsored research in various journals and conference publications. Copies of the more relevant papers we have published over the past year are included as chapters of this report to provide complete documentation of each aspect of our work.

In Chapter 2, we provide a summary and overview of our research progress achieved over the past year. This work addresses five vital areas of optical data processing research:

1. real-time spatial light modulators (Section 2.2 and Chapter 3),
2. optical pattern recognition (Section 2.3 and Chapter 4),
3. optical feature extraction (Section 2.4 and Chapters 5-7),
4. optical correlation (Section 2.5 and Chapter 8), and
5. optical linear algebra processors (Section 2.6 and Chapters 9-14).

Topic (1) concerns the vital issue of real-time spatial light modulators. Topics (2)-(4) address pattern recognition for ATR using optical pattern recognition (OPR) techniques. In this work, we have been faithful to address vital problems such as multi-class distortion-invariant pattern recognition of military targets, the acquisition and importance of a large data base, and the effect of noise on the algorithm used. Topic (5) concerns the most attractive item in optical processing at present and a potentially quite general-purpose optical processor.

Details on the more salient results of our research are provided in Chapters 3-14. References are included in Chapter 15. In Chapter 16, we enumerate our AFOSR sponsored publications, the presentations given on this research at conferences and seminars during the past year, and the Master's and PhD students that this grant has supported.

During the past year, the principal investigator (PI) presented invited talks on our AFOSR sponsored research at various conferences including the Critical Review of Technology SPIE Conference on Optical Computing (SPIE, Los Angeles, CA, January 1984) and the DoD conference on Parallel Algorithms and Architectures for ATR (Leesburg, VA, July 1984) and various optical computing and robotics conferences during the past year. The PI has chaired conference sessions and seminars and served on the organizing committees for the following conferences and topics: SPIE (Robotics), IOCC (Optical Computing), ICALEO (Optical Data Processing). One of this major papers in 1984 was an invited paper on optical linear algebra processors for the July 1984 *Proc. IEEE Special Issue on Optical Computing.*

2. OVERVIEW AND SUMMARY

2.1 INTRODUCTION

Our five major research areas and our recent progress in each are highlighted in Sections 2.2 - 2.6. Details of each aspect of our thirteen work topics follows in Chapters 3 - 14.

2.2 SPATIAL LIGHT MODULATORS (ACOUSTO-OPTIC CELLS, CHAPTER 3)

Recently, our spatial light modulator research has emphasized acousto-optic cells. In Chapter 3, we discuss recent new work in this area [1]. We have considered the salient acousto-optic architectures (spectrum analyzers and correlators). The various acousto-optic cell and acousto-optic architecture component errors have been enumerated, grouped into different classes and combined into several new models. New performance measures for acousto-optic correlators and spectrum analyzers were defined and detailed (spectrum estimation, delay estimation, and detection). Each is an appropriate performance measure for a different application. General error-free formulae for each of these performance measures were derived and the performance obtained with each was described and quantified as a function of the various system parameters. Our future work in this area will include component error source effects on performance, the relationship of these models to optical linear algebra processors and tests on multi-channel acousto-optic cells.

2.3 OPTICAL PATTERN RECOGNITION REVIEWS (CHAPTER 4)

Our AFOSR optical pattern recognition research is at the forefront. Our paper [2] on coherent optical pattern recognition was included in the recent Critical Review of Technology series on optical computing. A more recent review [2] was the only optical pattern recognition paper at a DoD conference on parallel architectures and algorithms for ATR. For completeness and as an introduction and overview, we summarize recent coherent optical pattern recognition research. A full length journal paper on this topic with extensive references is expected to be in the *Optical Engineering - Special Issue on Optical Computing* [23] in January 1985 and will be included in our 1985 report. Chapter 4 reviews optical techniques for feature extraction and correlation, new algorithms, architectures and hybrid optical/digital concepts [2].

2.4 OPTICAL PATTERN RECOGNITION FEATURE EXTRACTION (CHAPTERS 5 - 7)

Two new optical feature extraction techniques are detailed: the use of new feature extractors and dimensionality reduction techiques on a wedge ring detector-sampled optically-produced feature space (Chapter 5) and a hierarchical two-level hybrid optical/digital moment feature processor (Chapters 6 and 7). Our earlier conference paper [4] on an optical Fourier coefficient feature space has been improved and expanded into a journal paper [5] for a special issue on robot vision. In Chapter 5 [4], this work is summarized. It includes four different dimensionality reduction and feature extraction techniques, a new classifier concept, quantitative data on the importance of amplitude versus phase Fourier coefficients (for pattern recognition, rather than image reconstruction) and the performance of each in the presence of noise. Experimental results for two letters and two vehicles with 25 images of each at different scale and in-plane rotational differences were obtained. In Chapter 6, our new hybrid optical/digital moment processor, a new hierarchical class estimator, and a new two-level classifier are detailed and results obtained on a set of over 300 robot objects (pipe parts) [6]. New quantitative and analysis data for our ship image data base will shortly be published [7]. The performance of the system on non-controlled imagery and the necessary pre-processing required are included [8] in Chapter 7. Our future work will involve laboratory optical Fourier coefficient research, new theoretical and optical laboratory work on chord distributions, fundamental work on training set selection, laboratory optical moment system fabrication, generic object recognition using optical feature extractors and synthetic filters. Our feature extraction work will continue to address distortion-invariant multi-class object recognition and performance in the presence of noise.

2.5 OPTICAL PATTERN RECOGNITION CORRELATORS (CHAPTER 8)

Our distortion-invariant multi-class multi-object correlator research emphasizes synthetic discriminant functions (SDFs). The basic SDF synthesis algorithms have now been unified [9] (Chapter 8). Our tests of projection SDFs on ship images with data on noise performance and guidelines for the selection of projection values are expected to appear [10] in a special journal issue on pattern recognition late this year. These results will be included in our 1985 Final Report. Three new types of SDFs have been devised and initial results with them have been obtained for a tank and APC image data base [11].

These details will be available shortly and will be included in our 1985 report together with initial results on linear functional (optimal linear discriminant functions) SDFs. Laboratory experimental data, system fabrication concepts and optical matched spatial filter work will be major future work issues together with various extensions of new SDFs and their applications to different correlation pattern recognition ATR data bases.

2.6 OPTICAL LINEAR ALGEBRA PROCESSORS (CHAPTERS 9 - 14)

This optical data processing application area has received very much recent attention.

Our recent work in this area has included extensions of previous LU and other direct matrix decomposition algorithms and architectures and new algorithms and architectures for back-substitution and the solution of triangular systems of LAEs (linear algebraic equations). Most recently, a parallel QR algorithm and its implementation were detailed by us [12,13]. This completes the major algorithm optical realization work on direct and indirect linear algebra solutions to systems of LAEs. A recent special issue of the *Proc. IEEE* on optical computing summarizes our architecture, algorithm, data flow and selected applications research on optical linear algebra processors. Chapter 9 details this work [14]. It is extremely noteworthy since one optical linear algebra processor system can achieve all necessary operations by format control.

A second vital aspect of optical linear algebra research that we initiated was the error source modeling and simulation of OLAP (optical linear algebra processor) architectures and algorithms [15]. Chapter 10 details this work [15] and our initial results using it in the comparison of direct and iterative solutions of LAEs on OLAPs. A third facet of our OLAP research has concerned specific applications. The operation chosen for major attention was Kalman filtering and the specific application of it was missile guidance and control. In [16] we first advanced the details of a general Kalman filter realization on one type of OLAP. Chapter 11 details this work fully [17]. A new architecture for Kalman filtering when noise statistics are known [18] was also devised. It is detailed in Chapter 12. New algorithms and optical architectures and accuracy issues concerning this and other applications will be available in 1985 as we unify our algorithm, architecture, modeling and simulation research on this application. We have

detailed the use of residue arithmetic in OLAPs [19] to achieve increased accuracy and have found other methods to be preferable to the use of residue arithmetic.

The major linear algebra operation required in Kalman filtering is the solution of a nonlinear quadratic matrix equation. We have devised a new algorithm to achieve this using a fixed number of iterations. We have quantified all operational parameters for the algorithm, simulated several solutions of it using different algorithms, assessed the effect of different optical system errors, the dominant optical system errors, and the effect of multiple errors as well as quantified the performance of the algorithm and provided a laboratory OLAP demonstration of it. This work is detailed [20] in Chapter 13.

The fourth and final aspect of our OLAP research has been attention to fabrication of an OLAP. We recently [21] clarified that the number of operations acheivable on our frequency-multiplexed processor is comparable to others and showed its equality and that it is preferable from a fabrication standpoint. We also detailed 4 - 5 different techniques for fabrication of such a system and provided the first initial laboratory experimental data on the performance and operation of an optical systolic processor. These results are highlighted in Chapter 14. In 1985, we expect significant laboratory OLAP results to emerge. Many applications for OLAPs exist. Reference [14] details several others and reference [22] discusses their use in pattern recognition.

3. TIME-INTEGRATING ACOUSTO-OPTIC CORRELATOR: ERROR SOURCE MODELING

Time-integrating acoustooptic correlator: error source modeling

David Casasent, Anastasios Goutzoulis, and B. V. K. Vijaya Kumar

The error sources present in a time-integrating acoustooptic correlator are considered. They are classified and modeled into three categories: input plane errors; frequency plane errors; and detector plane errors. To facilitate error analyses, performance measures are defined and quantified for an error-free system for detection and delay estimation applications.

I. Introduction

Optical signal processors provide real-time operations on high bandwidth and time–bandwidth product data of long duration and high center frequency. These features plus the rapidly maturing commercial availability of acoustooptic components are the major reasons optical signal processors (OSPs) employing acoustooptic devices have recently received considerable attention.[1–6] These acoustooptic systems offer a most attractive approach to signal processing problems in which data with high time bandwidths and variable codes must be processed.

Acoustooptic (AO) devices can be incorporated into various architectures. These OSP systems[1–4] can be divided into two general classes: (1) correlators and (2) spectrum analyzers. Both system classes can be realized by performing the necessary integration in space or in time.[1]

Time-integrating (TI) processors[7] have received considerable attention because they can accommodate extremely large time–bandwidth (TBW) product data and because many new and attractive TI algorithms[8] and architectures[9] exist. An important feature of TI processors is their ability to operate on signals with very large TBW product with the ability to change (on-line) the signal code being processed.

Despite the rapidly increasing use of AO devices, little attention (in the literature) has been given to the modeling of the components of such systems, to the effect various system parameters and component error sources have on the performance of these systems, and to the performance measures used to describe, analyze, and design such processors. In this paper, we advance the first such formulation for TI bulk AO correlators. In Sec. II, we briefly review the operation of a TI AO correlator. Our categorization and modeling of the various error sources are given in Sec. III, and their enumeration and origin are then presented in Sec. IV.

In Sec. V, we discuss the performance measures we chose to describe the accuracy and performance of the TI AO correlator. We consider two different correlator applications (detection and delay estimation) and employ different performance measures for each. The basic error-free analyses for a TI AO correlator for detection and delay estimation are then presented (Secs. VI and VII). These analyses provide the basic statistical framework for further analyses that include and quantify the effects of the various error sources. Such analyses will be the subject of future publications.

II. Signal Correlation with TI AO Correlators

The basic operation of a TI AO correlator is explained with the aid of Fig. 1. The signals to be correlated are $s_a(t)$ and $s_b(t)$. $s_b(t)$ is usually a delayed version of $s_a(t)$ and includes some additive noise $n(t)$. For linear intensity modulation[9] of the AO cells, the signals are added to two biases B_1 and B_2 and used to modulate the amplitude of an rf carrier. Thus the baseband electrical inputs to the laser diode (or other input point modulator) and the AO cell are

$$s_1(t) = [B_1 + s_a(t)], \tag{1}$$

$$s_2(t) = [B_2 + s_b(t)]. \tag{2}$$

The intensity of the data portion of the light leaving plane P_1 is proportional to

When this work was done all authors were with Carnegie-Mellon University, Department of Electrical & Computer Engineering, Pittsburgh, Pennsylvania 15213; A. Goutzoulis is now with Westinghouse Research & Development Laboratories, Pittsburgh, Pennsylvania 15235.

Received 12 March 1984.
0003-6935/84/183130-08$02.00/0.

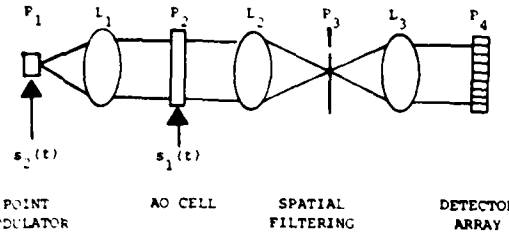


Fig. 1. Schematic diagram of a time-integrating acoustooptic correlator.

$$I_1(t) = B_2 + s_b(t). \tag{3}$$

This light beam is expanded by lens L_1 and uniformly illuminates the AO cell at plane P_2. Thus Eq. (3) also describes the light intensity incident on P_2. Note that it varies only in time and and not spatially. Lenses L_2, L_3 and the spatial filter at P_3 separate the undiffracted and diffracted orders, block the undiffracted order light, and image the first-order diffracted light onto a detector array at plane P_4. Denoting the detector's time constant of integration by T_I, the final detected output at P_4 is (including all bias terms)

$$I_4(\tau) = (1/T_I) \int_{-T_I/2}^{T_I/2} [B_2 + s_b(t)][B_1 + s_a(t - \tau)]dt, \tag{4}$$

where $\tau = x/v$, x is the direction of sound propagation in the AO cell, and v is the velocity of sound in the AO crystal. The second term in Eq. (4) is the modulation on the first-order term in the transmittance of P_2. Equation (4) can be further simplified to

$$I_4(\tau) = B + B_s + (1/T_I) \int_{-T_I/2}^{T_I/2} s_b(t)s_a(t - \tau)dt, \tag{5}$$

which is recognized as the desired correlation (last term) on a signal-independent bias B and a signal-dependent bias B_S with both temporal and spatial dependence.

Many AO architectures exist[9] that utilize amplitude rather than intensity modulation of the AO cell. In such cases, the detector output has the general form[9]

$$I_4(\tau) = B + B_S + (1/T_I)m \cos(2\pi f_0\tau) \int_{-T_I/2}^{T_I/2} s_b(t)s_a(t - \tau)dt, \tag{6}$$

where B is a bias, B_S is a signal-dependent bias with both temporal and spatial variation, m is a constant, and $\cos(2\pi f_0\tau)$ is a spatial carrier where f_0 is the frequency of a reference rf oscillator purposely included with the input signals. This electronic reference allows the correlation term to be separated from the bias terms by bandpass filtering the electrically readout version of Eq. (6).

The intensity and amplitude modulation modes for TI AO correlators have many well-known[9] advantages and disadvantages. Our initial objective is to model the various component error sources and system parameters so that our results are appropriate for both modulation modes. To achieve this, we consider the intensity modulation scheme, which for $B = B_S = 0$ is equivalent to the amplitude modulation scheme (after the necessary postprocessing filtering).

III. Error Source Modeling

In this section, we describe the mathematical models we use to describe the various types of component error in an AO TI correlator. The ideal model would express the system's output as a function of all the error parameters; however, because of the number of error sources and their nature, such a model cannot be analyzed statistically. Thus, we propose to model, study, and quantify several independent classes of errors and to determine the lower bound of the system's performance for each error class independently. We thus include three classes of error distinguished by whether their effects are modeled in the input, frequency, or output detector plane. Fortunately, there are only a few error sources that affect more than one error plane. We elaborate on these errors and the way to treat them in Sec. IV.

If the error to be considered occurs in the input plane P_2, it is directly mapped onto the output plane, and we thus describe it by including a multiplicative weighting function $w(\tau)$ in the processor's output, Eq. (5) becomes

$$I(\tau) = w(\tau)[B + B_S + (1/T_I) \int_{-T_I/2}^{T_I/2} s_b(t)s_a(t - \tau)dt]. \tag{7}$$

This class of error is quite unique, since it maps directly onto the output plane. The effect of this type of error is local rather than global and can thus be corrected by postdetection processing.

The second class of error are those which affect the frequency response of the system. (These concern the AO cell and the lenses.) Such error sources are obviously best modeled by a weighting function in the frequency plane. Because of the excellent quality of state-of-the-art lenses plus the fact that lens effects in optical processors have been studied in detail elsewhere,[10] we restrict our attention to the AO cell frequency response. With this in mind, we include the impulse responses $h_1(t)$ and $h_2(t)$ of the AO cell and the input point modulator. With these factors included, we can describe the output of the processor as

$$I(\tau) = B + B_S' + (1/T_I) \int_{-T_I/2}^{T_I/2} \iint_{-\infty}^{+\infty} h_2(\lambda_2)s_b(t - \lambda_2) \times h_1(\lambda_1)s_a(t - \tau - \lambda_1)d\lambda_1 d\lambda_2 dt, \tag{8}$$

where B_S' includes the effects of h_1 and h_2 on B_S. To obtain Eq. (8), we used the fact that the convolution of the input signal $s(t)$ with the system's impulse response $h(t)$ can be written as $\int h(\lambda)s(t - \lambda)d\lambda$. In writing Eq. (8), we also assumed that the AO cell is operated in a linear intensity mode, as is necessary for an AO correlator. We also note that intermodulation products in a TI correlator do not appear in specific spatial locations (as is the case for space-integrating spectrum analyzers), but rather they tend to be uniformly distributed over the output plane, and thus their effect is rather small.

For most statistical analyses, the form of Eq. (8) is not convenient. This is because extensive convolutions have to be evaluated in the time domain. Such a task is not trivial in digital simulators, since most signals of

interest are most commonly described in terms of their frequency domain characteristics. It is thus preferable that this class of e: or be studied in the frequency domain. This choi : is also convenient because the transfer functions $H(f) = \mathfrak{F}[h(t)]$ are easily measured for the two real-time devices in the system. To express Eq. (8) in the frequency domain, i.e., in terms of $H(f)$, we first form the expected value of Eq. (8) as below:

$$E|I(\tau)| = E|B| + E|B_S| + \frac{1}{T_I}\int_{-T_I/2}^{T_I/2}\iint_{-\infty}^{\infty} h_2(\lambda_2)h_1(\lambda_1) \times E|s_a(t-\tau-\lambda_1)s_b(t-\lambda_2)|d\lambda_1 d\lambda_2 dt. \quad (9)$$

For analytical simplicity, we consider the case of $\tau_0 = 0$ (without any loss of generality). Then the received signal $s_b(t)$ is simply $s_a(t) + n(t)$, and Eq. (9) becomes

$$E|I(\tau)| = B + E|B_S| + \frac{1}{T_I}\int_{-T_I/2}^{T_I/2}\iint_{-\infty}^{\infty} h_2(\lambda_2)h_1(\lambda_1) \cdot [E|s_a(t-\tau-\lambda_1)s_a(t-\lambda_2)| + E|s_a(t-\tau-\lambda_1)n(t-\lambda_2)|]d\lambda_1 d\lambda_2 dt. \quad (10)$$

Since the noise $n(t)$ is of zero-mean and is statistically independent from the signal $s_a(t)$, Eq. (10) simplifies to

$$E|I(\tau)| = B + E|B_S| + \frac{1}{T_I}\int_{-T_I/2}^{T_I/2}\iint_{-\infty}^{\infty} h_2(\lambda_2)h_1(\lambda_1) \cdot R_s(\tau+\lambda_1-\lambda_2)d\lambda_1 d\lambda_2 dt$$

$$= B + E|B_S| + \iint_{-\infty}^{\infty} h_2(\lambda_2)h_1(\lambda_1) \times R_s(\tau+\lambda_1-\lambda_2)d\lambda_1 d\lambda_2. \quad (11)$$

Expressing the signal autocorrelation function $R_s(\tau)$ in terms of its power spectral density $P_s(f)$, i.e.,

$$R_s(\tau) = \int_{-\infty}^{\infty} P_S(f)\exp(j2\pi f\tau)df, \quad (12)$$

Eq. (11) becomes

$$E|I(\tau)| = B + E|B_S| + \iiint_{-\infty}^{\infty} h_2(\lambda_2)\exp(-j2\pi f\lambda_2) \cdot h_1(\lambda_1)\exp(j2\pi f\lambda_1)P_S(f)\exp(j2\pi f\tau)d\lambda_1 d\lambda_2 df$$

$$= B + E|B_S| + \int_{-\infty}^{\infty} H_1^*(f)H_2(f)P_S(f)\exp(j2\pi f\tau)df, \quad (13)$$

where $H_2(f)$ and $H_1(f)$ are the transfer functions of the point modulator (or laser diode) and the AO cell, respectively. When formulated as in Eq. (13), the effect of all the frequency plane errors can be described by one transfer function $H(f) = H_1^*(f)H_2(f)$. This class of errors is global in nature and cannot be corrected by postdetection electronic processing and are thus appreciably different from the input plane weighting error sources whose effects were described by Eq. (7).

The third class of error are those which are best classified as detector plane errors. (They are due to the output detectors used.) Some of these detector errors can be considered as spatial response variations. They are best modeled by including them in our input plane weighting factor $w(\tau)$. The next detector plane error we consider is the output plane sampling (i.e., the fact that the output plane detectors are of finite size or length D in one dimension). This effect causes a spatial integration of the output over D followed by a temporal integration over T_I. We describe such detector effects by writing the observed output as an integral over space $(d\tau)$ followed by an integral in time (dt), i.e.,

$$I(k) = (1/T_I)\int_{-T_I/2}^{T_I/2}\int_{(k-1/2)D}^{(k+1/2)D} w_k(\tau) \times [B + B_S + s_a(t)s_b(t-\tau) + s_k(t)]d\tau dt, \quad (14)$$

where k is the detector element number, $w_k(\tau)$ is the spatial weighting function across the detector k, $s_k(t)$ is the noise of the kth detector element (this includes detector element cross talk) and where the integration over the 1-D detector area D describes the effect of the finite size of each detector element.

IV. Classification of Error Sources

In this section, we consider the origin of the specific system error sources that give rise to the three types of error we isolated in Section. III. We also discuss several other component errors present in an AO TI correlator and how to treat their effects.

In the case of input plane errors, we include the input optical beam profile, spatial variations in the AO cell response, and the nonuniform element-to-element response of the detector array. The input optical beam has a Gaussian rather than a plane-wave profile that can be described as[11]

$$W_{OB}(\tau) = \exp\left[-2\left(\frac{\tau-\tau_c}{W}\right)^2\right], \quad (15)$$

where the beam-taper coefficient W is the beamwidth at which the input light intensity is down by $\exp(-2)$ and τ_c denotes the center of the AO cell. This effect can be reduced by proper design of the collimation lens system L_1. In most practical situations (AO cells with 2–3-cm aperture), beam uniformities of 5% can be achieved without significant loss of input light. This corresponds to a worst-case weighting of 0.22 dB across the output. This can be reduced further by postdetection processing (since the errors are spatially fixed). Thus we ignore this effect in our future analysis.

The AO devices are the system components with the most significant input-plane errors. These errors include (1) beam walkoff (referring to the fact that the acoustic beam does not travel normal to the transducer as it propagates along the cell); (2) reflections from the sides of the cell (referring to the fact that the acoustic wave diffracts as it leaves the transducer and thus can strike the sides of the cell and suffer multiple reflections before reaching the end of the cell. This results in a nonuniform acoustic field and is particularly important when long AO delay lines are used); (3) near-field effects (referring to the Fresnel pattern resulting from the transducer excitation); (4) acoustic attenuation (referring to the fact that the acoustic field strength decreases exponentially within the cell with increasing distance from the transducer and as a function of frequency).

This last AO error source contributes to both the input plane and frequency plane errors. For a fixed signal bandwidth, we calculate the resulting weighting due to acoustic attenuation as a function of distance only and incorporate it into our $w(\tau)$ input plane weighting function in Eq. (7). To describe the model the frequency dependence of the acoustic attenuation α (i.e., $\alpha \propto f^2$), we include its effect in $H(f)$ for the AO cell.

Many of these error sources can be reduced or corrected for to various degrees. The beam walkoff can be minimized by proper AO cell design and accurate crystal cut. The sound reflections and near-field effects can be partially corrected by either careful AO cell design or optical spatial filtering. The effects of acoustic attenuation can be reduced (at one frequency) by electronic postprocessing and by a fixed optical mask whose transmittance compensates the acoustic attenuation's weighting. The last input plane error is the element-to-element nonuniform response in the detector array. State-of-the-art detector arrays have a uniformity of 90–95%. This corresponds to a maximum 0.46-dB spatial variation in the output plane intensity data. This weighting is generally negligible but is correctable if required for a given application.

Next we enumerate the frequency plane errors. In this category, we include the nonideal transfer function $H(f) = |H(f)| \exp[j\theta(f)]$ of the AO cells, where the magnitude $|H(f)|$ and phase $\theta(f)$ are functions of the frequency of the input signal. It is known[12] that $|H(f)|$ is the product of (1) the transducer's transfer function, (2) the shape of the acoustic interaction bandwidth, and (3) spatial frequency response terms due to dispersion and the finite AO cell aperture. The magnitude of the transfer function is also affected by the acoustic attenuation as noted earlier. The AO cell's phase response $\theta(f)$ is composed of (1) the transducer's phase response and (2) the optical phase within the cell. The transducer's phase is in general nonlinear[12] with a shape that depends mainly on the bonding techniques used. (A thin bond yields a quite nonlinear phase response, whereas a quarterwave bond yields a less nonlinear phase response but a poorer electrical-to-acoustical conversion efficiency for the transducer.) The optical phase effects are due to (1) off-axis acoustic beams that propagate in different directions due to beam walkoff and other effects (each off-axis beam will have a different phase) and (2) the finite transducer width. (This results in acoustic diffraction, which in turn results in off-axis beams.) With careful AO cell design and external optical filtering, the optical phase effects can be minimized, and the transducer's phase effects will dominate. We will assume this in future analyses.

In Ref. 12, exact expressions for $|h(f)|$ and models for $\theta(f)$ have been calculated. The $|H(f)|$ expressions are quite complex, and it is thus difficult to incorporate them in any statistical analysis. Thus the alternative we adopted was to measure $|H(f)|$ (from the optical Fourier transform of the light leaving the AO cell when it is driven by a linear FM signal) and to approximate it by a mathematical function. This alternative approach has yielded simpler expressions for $|H(f)|$ that can be incorporated in a statistical analysis. This class of errors is not correctable, and thus its effects on the system's performance will be studied in our future work.

The last class of error source are those due to the output correlation plane detectors. These include (1) the sampling and area integration due to the detector finite area D, (2) the spatial weighting function due to the trapezoidal[13] spatial response across each detector element, (3) the location of the output correlation peak within one detector element, (4) detector noise, and (5) cross talk between detectors. The effect of detector noise has been considered,[9] and the effect of finite detector area has been initially addressed.[14] The remaining detector error sources and the effects of all errors on our performance measures merit further research.

V. Performance Measures

Let us now discuss the performance measures which we will use in our error-free analyses (Secs. VI and VII) and in our future work. A correlator has two main purposes: (1) detection of the presence of a signal and (2) estimation of its location. These two different applications require different performance measures.

As detection performance measures, one should use probability of detection P_D, probability of false alarm P_{FA}, and probability of error P_e. The P_D is the probability that the correlation value at the peak $C(0)$ will exceed a threshold θ when the correlation is present. It is given by[15]

$$P_D = \frac{1}{\sqrt{2\pi \operatorname{var}[C(0)]}} \int_\theta^\infty \exp\left(\frac{-|x - E[C(0)]|^2}{2 \operatorname{var}[C(0)]}\right) dx, \quad (16)$$

where $E[C(0)]$ and $\operatorname{var}[C(0)]$ are the expected value and variance at the correlation peak and where $C(0)$ is modeled by a Gaussian random variable from central limit theorem arguments. P_D will be less than unity because of noise and because of the statistical nature of the signals. The presence of noise also results in a nonzero probability that the value of the correlation at the peak will exceed the threhsold when the signal is absent. This is the probability of false alarm. It can be described[15] as

$$P_{FA} = \frac{1}{\sqrt{2\pi \operatorname{var}[C(\tau)]}} \int_\theta^\infty \exp\left(\frac{-|x - E[C(\tau)]|^2}{2 \operatorname{var}[C(\tau)]}\right) dx, \quad (17)$$

where the mean $E[C(\tau)]$ and the variance $\operatorname{var}[C(\tau)]$ of the noise in the output plane can be estimated by evaluating $C(\tau)$ at $\tau \gg 0$. With no loss of generality, we assume the correlation peak occurs at $\tau = 0$. Assuming equal probabilities for the presence and absence of a signal, P_e is given by

$$P_e = \tfrac{1}{2}(1 + P_{FA} - P_D). \quad (18)$$

These three probabilistic performance measures can be easily expressed as a function of the correlation plane SNR values:

$$\mathrm{SNR}_1 = \frac{E^2[C(0)]}{\operatorname{var}[C(0)]}, \quad (19)$$

$$\mathrm{SNR}_2 = \frac{E^2[C(0)]}{\mathrm{var}[C(\tau)]\Big|_{\tau \gg 0}}. \tag{20}$$

which can easily be evaluated and experimentally measured. SNR_1 is the SNR at the peak (the conventional communications definition[15]), and SNR_2 is similar to the peak-to-sidelobe ratio (with the sidelobe or noise level measured at $\tau \gg 0$, far from the peak). Both of these SNR measures have been used previously,[16,17] and SNR_2 can be directly related to SNR_1. In terms of these SNRs, one can show

$$P_D = \frac{1}{\sqrt{2\pi E^2[C(0)]/\mathrm{SNR}_1}} \int_{\epsilon}^{\infty} \exp\left(\frac{-\mathrm{SNR}_1[x - E[C(0)]]^2}{2E^2[C(0)]}\right) dx, \tag{21}$$

$$P_{FA} = \frac{1}{\sqrt{2\pi E^2[C(0)]/\mathrm{SNR}_2}} \int_{\epsilon}^{\infty} \exp\left(\frac{-\mathrm{SNR}_2[x - E[C(\tau)]]^2}{2E^2[C(0)}\right) dx. \tag{22}$$

We now consider the second correlator application: delay estimation. The location of the correlation peak contains useful target location and signal synchronization information, and thus one refers to the target location or signal delay estimation performance of a TI AO correlator. In image correlations, this performance measure is referred to as registration error,[17] since it describes the accuracy with which the location of an object in an image is known or the accuracy to which two images can be registered. To develop an expression for the delay estimation error e, we define the exact peak location as τ_0, and we denote the estimated peak location by $\hat{\tau}_0$. Then

$$e = (\tau_0 - \hat{\tau}_0). \tag{23}$$

We will denote other observed parameters by a ($\hat{}$) (i.e., the observed correlation function is $\hat{C}$, whereas the exact or ideal correlation function is C). To analyze e, we must calculate its expected value and its variance. The first parameter determines if the estimator is biased (time bias), whereas the second provides us with the variation to be expected in calculating e. We thus use both $E[e]$ and $\mathrm{var}[e]$ as delay-estimation performance measures.

VI. Error-free Detection Analysis

In this section, we derive general expressions for the mean and variance of the correlation output. We then evaluate these expressions for a zero-mean Gaussian signal model with a Gaussian-shaped autocorrelation function. We then evaluate SNR_1 and SNR_2 and plot P_D and P_{FA} as functions of the basic TI AO system parameters.

For both active and passive signal processing, the transmitted $s_a(t)$ and received $s_b(t)$ signals to be correlated are denoted by

$$s_a(t) = s(t), \tag{24}$$

$$s_b(t) = s(t - \tau_0) + n(t), \tag{25}$$

where τ_0 is the time delay between the two signals, and $n(t)$ is additive noise. Assuming equal biases ($B_1 = B_2 = B$) for simplicity, Eq. (4) becomes

$$C(\tau) = B^2 + \frac{B}{T_I}\int_{-T_I/2}^{T_I/2} s(t - \tau_0)dt + \frac{B}{T_I}\int_{-T_I/2}^{T_I/2} n(t)dt + \frac{B}{T_I}\int_{-T_I/2}^{T_I/2} s(t - \tau)dt + \frac{1}{T_I}\int_{-T_I/2}^{T_I/2} s(t - \tau)n(t)dt + \frac{1}{T_I}\int_{-T_I/2}^{T_I/2} s(t - \tau)s(t - \tau_0)dt. \tag{26}$$

Since $s(t)$ and $n(t)$ are zero-mean signals and assuming that they are independent random processes, the expected value of the correlation peak (assumed to occur at $\tau_0 = 0$ with no loss of generality) becomes

$$E[\hat{C}(0)] = \frac{1}{T_I}\int_{-T_I/2}^{T_I/2} E[s(t)s(t)]dt = R_s(0), \tag{27}$$

where $R_s(\tau)$ is the signal autocorrelation function, and where we assumed that the bias term B^2 was subtracted from the output. In general, we find $E[\hat{C}(\tau)] = R_s(\tau)$, and thus (with bias subtraction) the estimator is unbiased or the average value of the estimated correlation equals the correlation we are trying to estimate. The variance of $\hat{C}(\tau)$ is easily found to be

$$\begin{aligned}\mathrm{var}[\hat{C}(\tau)] &= E[\hat{C}(\tau)]^2 - E^2[\hat{C}(\tau)] \\ &= (B^2/T_I^2)\int_{-T_I}^{T_I} (T_I - |z|)[2R_s(z) + R_n(z) \\ &\quad + (1/B^2)R_s^2(z) + (1/B^2)R_s(z + \tau)R_s(z - \tau) \\ &\quad + (1/B^2)R_s(z)R_n(z) + 2R_s(z - \tau)]dz,\end{aligned} \tag{28}$$

where the assumption that $s(t)$ is Gaussian distributed (this makes the third-order moments zero), and the fourth-order theorem for Gaussian random variables[18] were used.

We now consider the evaluation of our output SNR_0 measures in Eqs. (19) and (20) for the specific case of a Gaussian signal with a Guassian-shaped autocorrelation function. In this case, $R_s(z)$ has the form

$$R_s(\tau) = R_0 \exp(-\pi\beta^2\tau^2), \tag{29}$$

where β is the signal's 3-dB bandwidth (BW) and R_0 is the signal power. We assume that the noise has power R_n and has a statistical autocorrelation function that is of the same form as in Eq. (29). In this case, the input SNR is $\mathrm{SNR}_I = R_0/R_n$. A more complex SNR_I expression results[16] if this assumption is not valid. Substituting Eq. (29) into Eqs. (27) and (28), evaluating at $\tau = 0$, and assuming $T_I \gg 1/\beta$ (i.e., a signal time–bandwidth product $\mathrm{TBW} = T_I > 10$), we find

$$\mathrm{SNR}_1 = \frac{T_I\beta}{\frac{\sqrt{2}}{2}(2 + 1/\mathrm{SNR}_I) + \frac{1}{(\mathrm{SBR})^2}\left(4 + \frac{1}{\mathrm{SNR}_I}\right)}, \tag{30}$$

where $\mathrm{SBR} = \sqrt{R_0}/B$ is the signal-to-bias ratio. The SNR_2 expression is identical to Eq. (30) with $(2 + 1/\mathrm{SNR}_I)$ replaced by $(1 + 1/\mathrm{SNR}_I)$. We note that when SBR and SNR_I are infinite, SNR_2 is >SNR_1 by 3 dB.

From Eq. (30), we note that both SNR_0 measures increase as either $T_I\beta$, TBW, SNR_I, or SBR increase. These results are as expected. In Fig. 2, we plot SNR_2 vs TBW for several SBR values. These plots are important since they quantify the rather severe SNR_0 loss for SBR < ∞. SNR_0 is superior with AO cell amplitude

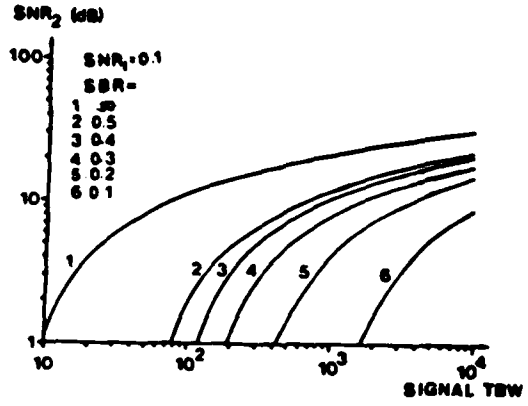
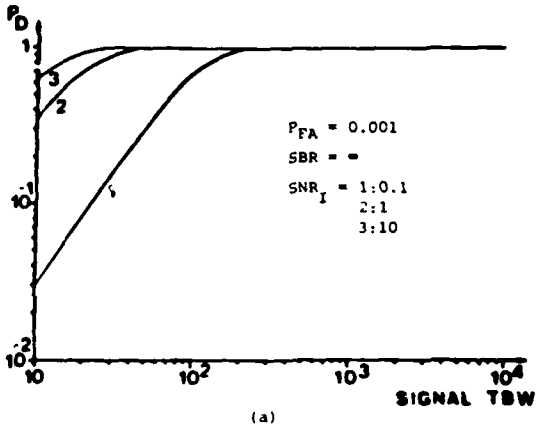


Fig. 2. Effect of SBR and signal TBW on output SNR_2 for a time-integrating correlator.

modulation than with intensity modulation. Here we quantify this effect and the effect of SBR on SNR_0. The case of no bias (i.e., SBR = ∞) corresponds to AO cell amplitude modulation. SBR = 0.5 corresponds to the best case one can obtain using AO cell intensity modulation. Comparing these two cases, we find an SNR_0 loss of 8 dB for most TBWs.

Expressions for P_D and P_{FA} are obtained by substituting Eq. (30) into Eqs. (21) and (22). We normalize to unity and consider $E[C(\tau)]$ to be zero for $\tau \gg 0$. In Fig. 3, we show P_D (for $P_{FA} = 0.001$) and P_{FA} (for $P_D = 0.999$) vs TBW for three SNR_I values (i.e., 0.1, 1.0, and 10.0). From these curves, we see that $P_D(P_{FA})$ increases (decreases) monotonically as TBW or SNR_I increases, thus improving the detection performance of the system. Such trends ar well known. In Fig. 4, we show P_D (for $P_{FA} = 0.001$) and P_{FA} (for $P_d = 0.999$) as a function of TBW and SBR. As expected, $P_D(P_{FA})$ increases (decreases) as SBR increases. For the best practical intensity modulation case (SBR = 0.5), both $P_D(P_{FA})$ are quite inferior to the P_D and P_{FA} values for SBR = ∞. For TBW > 5000, the difference is small.



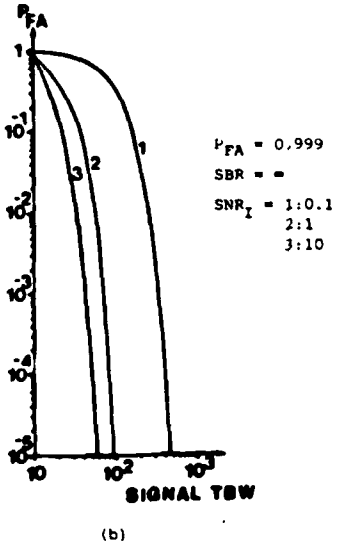
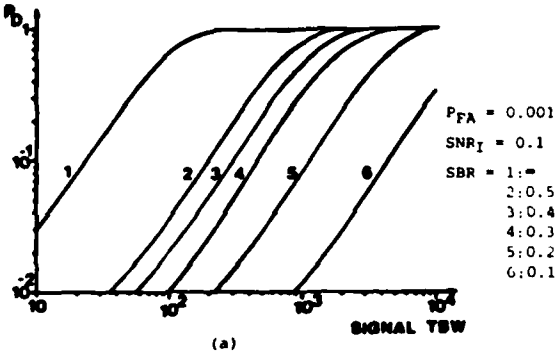


Fig. 3. Effect of input SNR_I and signal TBW on detection performance for a time-integrating correlator: (a) P_D (for $P_{FA} = 0.001$) and (b) P_{FA} (for $P_D = 0.999$).



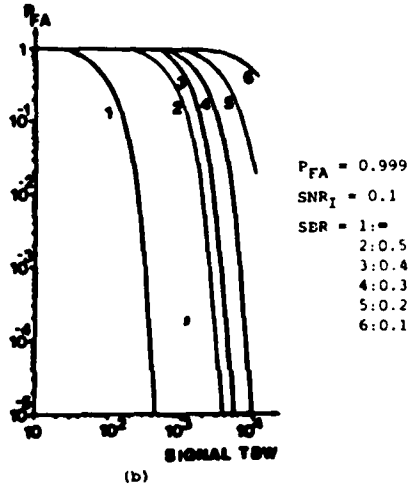


Fig. 4. Effect of SBR and signal TBW on the detection performance of a time-integrating correlator: (a) P_D (for $P_FA = 0.001$) and (b) P_{FA} (for $P_D = 0.999$).

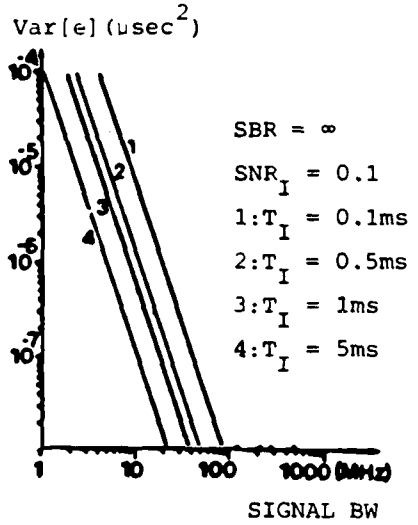


Fig. 5. Effect of integration time T_I and signal BW on the variance var[e] of the delay estimation error e for a time-integrating correlator.

For TBW < 1000, the difference is quite appreciable (i.e., for TBW = 450, P_D = 0.35 vs 0.999, and P_{FA} = 0.65 vs 0.00001). From these data, the TBW increase required to achieve a given performance for a given SBR can be found.

VII. Error-free Delay Estimation Analysis

We now analyze the delay estimation performance of the correlator with attention to the calculations of $E[e]$ and var[e], their relationships to measurable $\hat{C}(\tau)$ correlation data, and their dependence on system parameters.

To relate $\hat{C}(\tau)$ to e, we expand $\hat{C}(\tau)$ in a Taylor series around $\tau = \tau_0$. We consider $\hat{\tau}_0 \cong \tau_0$ and thus ignore terms higher than $(\hat{\tau}_0 - \tau_0)^2$ in our $\hat{C}(\tau_0)$ expansion. Setting the derivative $\hat{C}'(\tau)$ of $\hat{C}(\tau)$ equal to zero, we find

$$e = (\tau_0 - \hat{\tau}_0) = -\hat{C}'(\tau_0)/\hat{C}''(\tau_0), \tag{31}$$

where $'$ and $''$ denote first and second derivatives. From the gradient $\hat{C}'(\tau_0)$ and curvature $\hat{C}''(\tau_0)$ of the correlation at the peak, we can thus obtain e. We consider the case when the standard deviation of the curvature $\hat{C}''(\tau_0)$ of the correlation at the true peak is small. In this case, $\hat{C}''(\tau_0)$ is well approximated by its average value. We thus write

$$e \cong -\hat{C}'(\tau_0)/\overline{\hat{C}''(\tau_0)}, \tag{32}$$

where the average value $\overline{\hat{C}''(\tau_0)}$ is used, since it can be more easily evaluated[18] for our statistical signals.

To describe the delay estimation performance, we require the mean and variance of e in Eq. (32) in terms of measurable correlator parameters. To develop this, we denote the two parts of the correlation output in Eq. (26) (for an intensity modulation scheme) by $\hat{C}_s(\tau)$ [the autocorrelation of $s(t)$ or the last term in Eq. (26)] and $\hat{C}_n(\tau)$ [the terms remaining in Eq. (27) after bias subtraction, i.e., only the fifth term in Eq. (26)]. In terms of these quantities,

$$E[e] = |E[\hat{C}_s(\tau_0)]| + E[\hat{C}_n(\tau_0)]|/\overline{\hat{C}''(\tau_0)}. \tag{33}$$

From Eq. (33), we see that e is an unbiased estimate. This follows since C_s is a maximum at τ_0, and thus its gradient is zero at τ_0. The second term in the numerator of Eq. (33) is zero since it corresponds to the zero mean terms in Eq. (26). Next, we evaluate the variance of e, which (from the above) simplifies to

$$\text{var}[e] = |E[\hat{C}''(\tau_0)]|^{-2}[E|\hat{C}_s(\tau_0)|^2 + E|\hat{C}_n(\tau_0)|^2] \tag{34}$$

To evaluate Eq. (34) in terms of system parameters, we considered the case of a Gaussian-distributed signal and noise with a Gaussian-shaped autocorrelation function given by Eq. (29). For this case, a lengthy but straightforward analysis shows

$$\text{var}[e] = \frac{1}{\beta^3 T_I^2}\left[\frac{T_I}{\text{SNR}_I\sqrt{32\pi}} + \frac{1}{4\beta\pi^2} + \frac{1}{2\beta(\pi\text{SBR})^2}\left(1 + \frac{1}{\text{SNR}_I}\right)\right]. \tag{35}$$

where equal signal and noise bandwidths β and TBW ≥ 10 were assumed.

Equation (35) relates the delay estimation accuracy of a TI AO correlator to the various system parameters. To quantify the delay estimation performance and the effect of the various system parameters, we include Fig. 5. In Fig. 5, we show how var[e] varies with the bandwidth β for four different T_I values (0.1, 0.5, 1.0, and 5.0 msec). From this figure, we see that var[e] monotonically decreases as β or T_I increases, and, most important, this plot quantifies these variations. The main new feature in these data is that var[e] depends more on the signal bandwidth ($\Delta\text{var}[e] \propto 1/\beta^3$) than on the integration time ($\Delta\text{var}[e] \propto 1/T_I^2$). In retrospect, this might have been expected because of the well-known inverse dependence of the width of the correlation peak on the bandwidth of the signal. However, the dependence has now been quantified. In Fig. 6, we show var[e] vs bandwidth β for three SNR_I values (0.1, 1.0, and 10.0). As expected, var[e] decreases as SNR_I or β increases. Finally, in Fig. 7, we plot var[e] as a function of bandwidth β for two SBR values (0.1 and ∞). As we see, var[e] exhibits a negligible dependence on SBR.

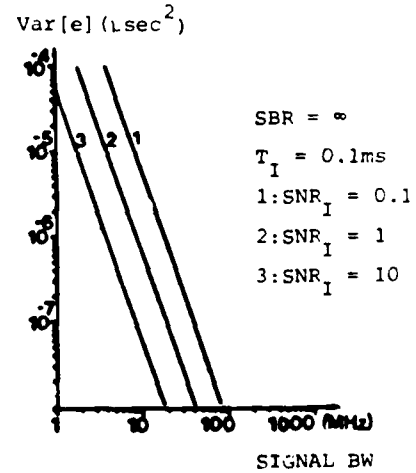


Fig. 6. Effect of SNR_I and signal BW on the variance var[e] of the delay estimation error e for a time-integrating correlator.

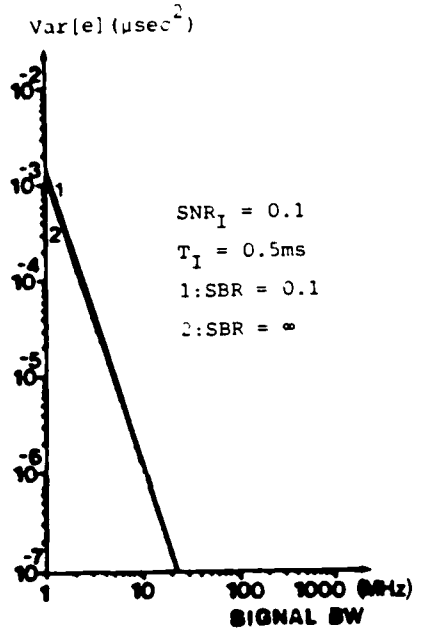


Fig. 7. Effect of SBR and signal BW on the variance var[ϵ] of the delay estimation error ϵ for a time-integrating correlator.

This is a rather important result since it demonstrates that AO correlators with amplitude (SBR = ∞) or intensity (SBR $\neq \infty$) modulation will yield equivalent delay estimation performance.

VIII. Summary

In this paper, we have modeled and classified the error sources present in a time-integrating acoustooptic correlator. We have shown that the error sources can best be grouped into three major error classes: input plane errors; frequency plane errors; and output detector plane errors. The effect of input plane errors was shown to be a weighting of the correlation output. These errors were quantified, and we noted that such errors are in general negligible and correctable. The frequency plane errors are uncorrectable and were noted to result in a complex frequency domain weighting. The major such error was found to be the nonideal transfer function of the acoustooptic cells. Most output detector plane errors are uncorrectable and were noted to consist of detector limitations such as sampling, area integration, and weightings.

We also suggested performance measures to describe the performance of a time-integrating acoustooptic correlator in different applications. For detection applications, we employ P_D, P_{FA}, and P_e and noted how to relate these measures to acoustooptic time-integrating correlator parameters and measurable correlation output SNRs. For delay estimation applications, we used the expected value and variance of the registration error. For these two correlator applications, we demonstrated how simple statistical analyses allow quantification of the effects of different system parameters. Using these formulations, we conducted error-free statistical analyses for both detection and delay estimation. Our analyses confirmed (and most important, quantitified) well-known trends. They also provided and quantified several new results. These included the effect of signal-to-bias ratio on both detection and delay estimation performance. For detection applications, we found that low SBR values significantly affect P_D and P_{FA}. For delay estimation applications, we found SBR effects to be rather negligible. These trends affect the selection of amplitude or intensity modulation modes for acoustooptic cells.

Our error-free statistical analyses provide valuable quantitative data. They also provide the base line performance levels against which to quantify the effect of the various component error sources noted. This present paper has laid the framework for the component error source analysis of acoustooptic correlators. A detailed error source analysis using these guidelines, models, and performance measures is the subject of future research.

The support of our basic research program by the Air Force Office of Scientific Research (grant AFOSR-79-0091 and agreement F49620-83-C-0100) and the support of this work by Teledyne Electronics are appreciated and gratefully acknowledged together with the assistance of Irving Cohen of Teledyne Electronics.

References

1. Special Issue on Acousto-Optic Signal Processing, Proc. IEEE **69** (1981).
2. Special Issue on Real-Time Signal Processing II, Proc. Soc. Photo-Opt. Instrum. Eng. **180** (1979).
3. Special Issue on Real-Time Signal Processing III, Proc. Soc. Photo-Opt. Instrum. Eng. **241** (1980).
4. Special Issue on Real-Time Signal Processing IV, Proc. Soc. Photo-Opt. Instrum. Eng. **298** (1981).
5. Special Issue on Active Optical Devices, Proc. Soc. Photo-Opt. Instrum. Eng. **202**, (1979).
6. Special Issue on Acousto-Optic Bulk Wave Devices, Proc. Soc. Photo-Opt. Instrum. Eng. **214** (1979).
7. R. A. Sprague and C. L. Koliopoulos, "Time Integrating Acoustooptic Correlator," Appl. Opt. **15**, 89 (1976).
8. J. M. Speiser and H. J. Whitehouse, Technical Report NUCTN 1355R, Naval Undersea Center, San Diego, Calif. (1974).
9. P. Kellman, "Time-Integrating Optical Signal Processors," PhD Thesis, Stanford U. (June 1979).
10. D. Casasent and T. K. Luu, "Performance Measurement Techniques for Simple Fourier Transform," Appl. Opt. **17**, 2973 (1978).
11. A. Yariv, *Introduction to Optical Electronics* (Holt, Rinehart & Winston, New York, 1976).
12. I. C. Chang and D. L. Hecht, "Characteristics of Acousto-Optic Devices for Signal Processors," Proc. Soc. Photo-Opt. Instrum. Eng. **241**, 129 (1980.
13. R. K. Hopwood, Proc. Soc. Photo-Opt. Instrum. Eng. **230**, 72 (1980).
14. B. V. K. Vijaya Kumar, D. Casasent, and A. Goutzoulis, "Fine Delay Estimation with Time Integrating Correlators," Appl. Opt. **21**, 3855 (1982).
15. H. L. Van Trees, *Detection, Estimation and Modulation Theory Part I* (Wiley, New York, 1968).
16. B. V. K. Vijaya Kumar and D. Casasent, "Space-Blur Bandwidth Product in Correlator Performance Evaluation," J. Opt. Soc. Am. **70**, 103 (1980).
17. H. Mostafavi and F. Smith, "Image Correlation With Geometric Distortions: parts I and II," IEEE Trans. Aerosp. Electron. Syst. **AES-14**, 487 (1978).
18. A. Papoulis, *Probability, Random Variables and Stochastic Processes* (McGraw-Hill, New York (1965).

4. PARALLEL COHERENT OPTICAL PROCESSOR ARCHITECTURES AND ALGORITHMS FOR ATR

PARALLEL COHERENT OPTICAL PROCESSOR ARCHITECTURES AND ALGORITHMS FOR ATR

David Casasent
Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

ABSTRACT

Parallel optical pattern recognition architectures for multi-class distortion-invariant autonomous target recognition (ATR) are described. Algorithms that utilize the parallel outputs and real-time processing features of optical systems are noted. Three hybrid optical/digital feature extraction techniques for ATR are described together with an optical correlation method that achieves multi-class shift-invariant distortion-invariant object identification. Initial results on selected military objects are included in the presentation. Brief remarks on optical systolic linear algebra processors are also advanced as they apply to the processing requirements for ATR.

1. INTRODUCTION

The real-time, parallel-processing, low size, weight and power dissipation advantages of optical pattern recognition (OPR) systems for ATR have long been recognized. Recently, several small size and weight real-time optical correlators have been fabricated and demonstrated.[1,2] Thus, the technology of OPR for ATR merits attention and discussion. In Section 2, we briefly review the classic Fourier transform (FT) and correlation operations of such architectures. Section 3 considers three different hybrid optical/digital feature extractors and Section 4 considers a new optical correlator. In all cases, these parallel architectures and algorithms achieve distortion-invariant multi-class recognition. Recent performance of one feature extraction system on non-controlled IR data[20] and recent performance of the correlator Section 4 in structured clutter have been obtained. Optical systolic linear algebra processors are then briefly noted in Section 5.

2. A REVIEW OF OPR

The system of Figure 1 is the classic OPR architecture. The FT G(u,v) of the input image g(x,y) in P_1 appears at P_2 with higher input spatial frequencies (u,v) appearing at radially increasing distances from the center of P_2. As the input translates, the intensity detected magnitude of the FT is shift-invariant. However, as the input object rotates, so does the FT. These features of a coherent optical system are exploited in all of our architectures to be described. In the full system of Figure 1, a transparency proportional to the conjugate FT $H^*(u,v)$ of a reference object can be recorded holographically at P_2. This is referred to as a matched spatial filter (MSF). The light distribution incident on P_2 is G(u,v) and the light leaving P_2 is $G(u,v)H^*(u,v)$. Thus passing one 2-D image plane through another achieves a 2-D point-by-point multiplication. This feature is likewise constantly exploited in OPR systems. The FT of this product of FTs is then formed at P_3, where the correlation of the two space functions f and h results.

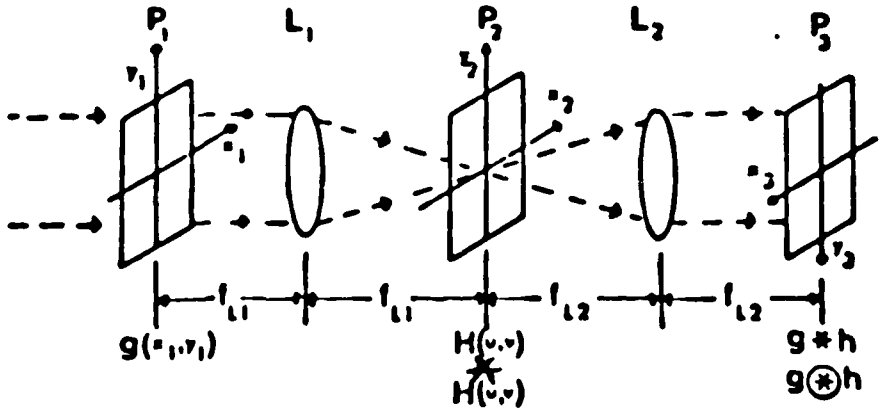


FIGURE 1 Coherent optical Fourier transform and correlation processor

3. OPTICAL FEATURE EXTRACTION

The classic approach to pattern recognition employs a training set of imagery from which features are extracted and subsequently operated upon to determine class and orientation estimates of input objects and the confidence of these estimates. A hybrid optical/digital architecture in which the image features are optically computed in parallel is shown schematically in Figure 2.

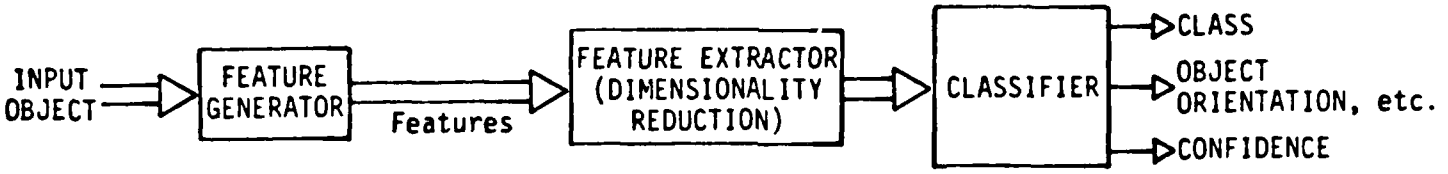


FIGURE 2 Simplified diagram of a hybrid optical/digital feature-space pattern recognition processor

Such an architecture is attractive because it can provide orientation information on the input object and because the same optical system can be used for different object classes. With the proper digital post-processor, distortion-invariance and multi-class recognition can be achieved. We now discuss three versions of parallel optically-computed features and the associated digital post-processor system required.

3.1. Fourier-Coefficient Feature-Space

The shift-invariance of the FT coupled with the change in scale and rotation of the FT pattern with changes in the scale and orientation of the input image can be utilized for feature-space pattern recognition. The anatomy of an optical FT dictates that a wedge ring detector (WRD) sampling at P_2 of Figure 1 provides data compression and dimensionality reduction in a Fourier coefficient feature-space plus scale (from wedge data) and rotation (from ring data) invariance. Many uses of this technique have been

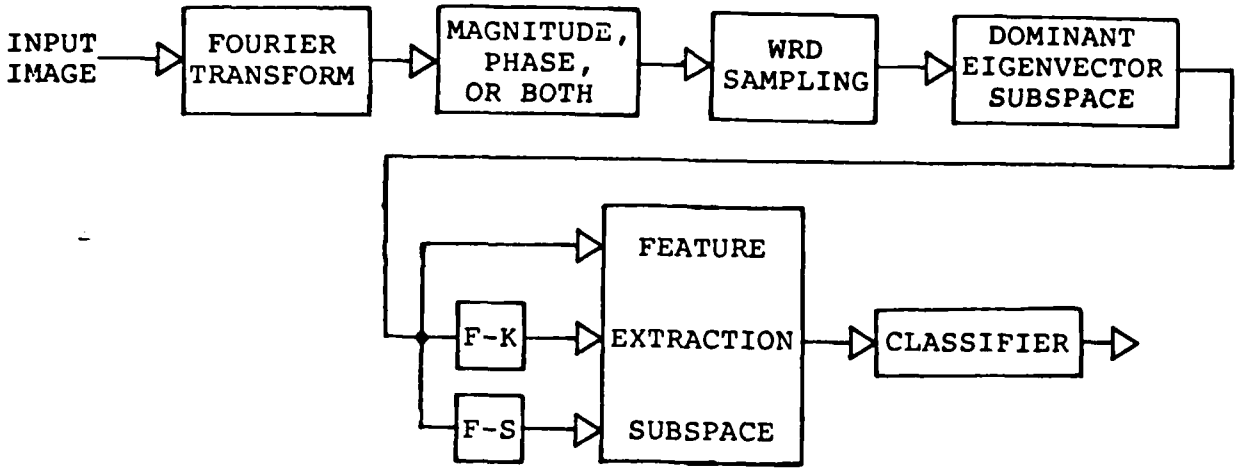


FIGURE 3 Block diagram of a hybrid optical/digital WRD-sampled Fourier-coefficient feature-space pattern recognition system

detailed.[3,4] The most recent work[2] used the system block diagram of Figure 3 in which the amplitude, phase and both the amplitude and phase of the FT were used as the observation space. WRD-sampling provided dimensionality reduction to 64 image features (32 wedges and 32 ring data elements). Feature extraction involves three techniques:

1. projection of the feature vector onto the dominant Karhunen-Loeve (KL)[5] eigenvectors per object class;

2. projection onto a Fukunaga-Koontz (FK)[6] discriminant vector for each class, with FK feature vectors calculated only from the dominant KL eigenvector; and

3. projection onto the Foley-Sammon (FS)[7] discriminant vector, calculated from the dominant KL eigenvectors only.

The three different feature extractors noted in Figure 3 were evaluated using two different data sets (50 different vehicles in two classes and 50 different letters of two types, each with different scales and orientations). The KL feature extractor was found to give good intra-class performance, but better Fisher ratio performance measures

resulted when the FK and FS unitary transformations were employed. Amplitude Fourier-coefficient features were found to be more robust in the presence of noise than were phase features. This is attributed to the concentration of the dominant Fourier-amplitude coefficients into a few WRD-samples, whereas phase Fourier-plane data is more evenly distributed over all WRD samples. An extensive tabulation and analysis of this data is available elsewhere.[8]. The performance obtained is not the major concern at present, rather the flexibility of digital analysis of Fourier-coefficients that are optically produced in parallel is the major message to be conveyed. These features are easily produced on the simplest coherent optical processor in parallel. Dimensionality reduction of these features is employed to simplify the digital post-processing required. Only simple vector inner product operations are needed in the post-processor, with computation of the discriminant functions and transformation matrices required being performed off-line on training set data.

3.2. Chord-Histogram Distribution Feature-Space

The chords of an object boundary define the object's shape and are useful image features.[9,10] Each chord is described by two parameters (its length r and angle θ). The distribution $h(r,\theta)$ of all chords thus defines the shape of the boundary. Denoting a boundary point on an object by $b(x,y) = 1$, then

$$g(x,y,r,\theta) = b(x,y)b(x+r\cos\theta, y+r\sin\theta) = 1 \tag{1}$$

defines a chord. The chord distribution is simply the integral

$$h(r,\theta) = \int g(x,y,r,\theta)dxdy. \tag{2}$$

Substituting $(\xi,\eta) = (r\cos\theta, r\sin\theta)$ into (2), the chord distribution is seen to be the

autocorrelation of the object's boundary. Optical systems easily perform the autocorrelation function on Figure 1 or in a joint transform correlator or from the FT of the magnitude of a FT. Since optical systems perform a correlation on the full grey-scale image rather than on just the object's boundary, a generalized chord distribution function can be obtained optically.[11] We WRD-sample this optical autocorrelation plane to simultaneously obtain the h(r) and h(θ) chord distributions and a reduced dimensionality feature space. These distributions provide invariance to object rotations and scales respectively.

The hybrid optical/digital system block diagram shown in Figure 4 uses the chord distributions optically generated in parallel together with a vector inner product of the observed feature vector and a Fisher discriminant vector $\underline{w}$ for feature extraction.[11] Comparison of the vector projection value to a threshold determined from the training set data determines the class of the input object. As before, the post-processor must perform only a vector inner product since calculation of the Fisher discriminant vector is performed off-line on training set data. The major multi-class databases on which most of the results noted were obtained consists of five different ship classes from a 90° depression angle with 36 images per class (at 10° aspect intervals). Extensive data[11] (summarized in Table 1) was obtained with the system of Figure 4.

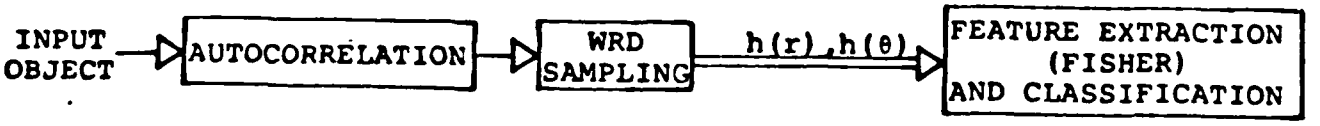


FIGURE 4 Block diagram of a hybrid optical/digital generalized chord histogram feature-space pattern recognition system

To compute the Fisher discriminant vector $\underline{w}$ with a reduced number of training set

Table 1: Test results obtained with a generalized chord feature space on the 72 images in the first 2 ship image classes

TEST NUMBER	NUMBER OF TR SET IMAGES USED PER CLASS	TR SET SELECTION REMARKS	NUMBER OF ERRORS	PERCENT CORRECT CLASSIFICATION
1	18	IMAGE EVERY 20°	0	100%
2	12	IMAGE EVERY 20° (±50° (BROADSIDE)	8	88.9%
3	12	IMAGE EVERY 30°	0	100%

images, we selected the 18 dominant WRD features and used 12-18 different training set images per class (tests 1-3 in Table 1). The results shown indicate that perfect class performance of all 72 images in the two ship classes tested can be obtained with as few as 12 training set images per class. Such excellent and correct recognition and classification of multiple object classes in the face of 3-D out-of-plane aspect distortions are typical of the performance that is possible with parallel optical feature extractors.

3.3. Moment Feature-Space

The geometrical intensity moments of an object f(x,y) are defined by

$$m_{pq} = \int\int f(x,y)x^p y^q dxdy. \tag{3}$$

These features are used in nearly all computer vision systems.[13] The moment feature vector $\underline{m}$ can be computed optically in parallel on the system of Figure 5.

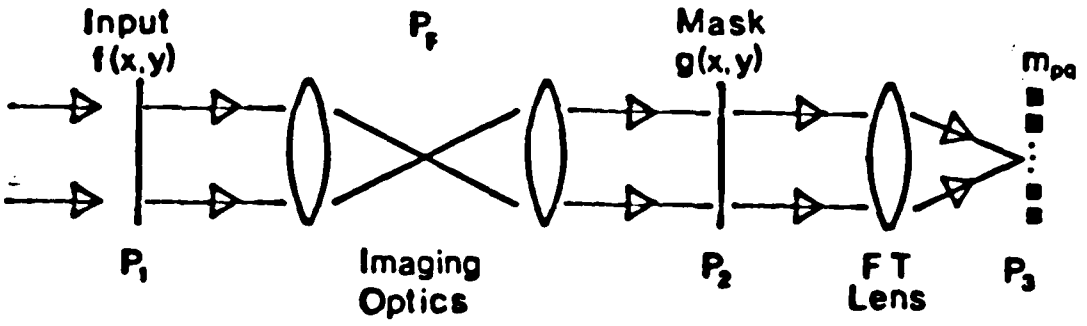


FIGURE 5 Schematic diagram of an optical moment-based feature generation system

With different monomial masks $g(x,y) = x^p y^q$ present on different spatial frequency carriers at P_2, the P_3 output pattern

$$\int f(x,y)g(x,y)dxdy \tag{4}$$

corresponds to the moments of the P_1 input f(x,y), each located at a spatially-different position in P_3. The parallel moment computer of Figure 5 is attractive because the computed moments can be corrected for various optical system errors in a simple matrix-vector post-processor.[12] The architecture of Figure 4 can be fabricated in a small size system occupying $330in^3$ or much less volume if needed.

The parallel set of observed moment features $\hat{\underline{m}}$ optically-computed in parallel are fed to the two-class classifier of Figure 6.

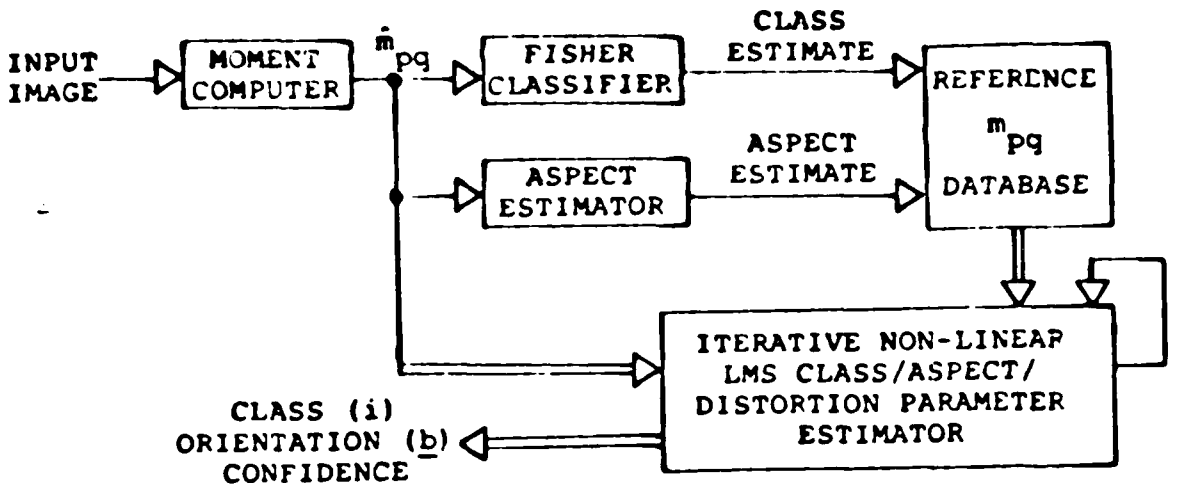


FIGURE 6 Full hybrid optical/digital moment feature-space two-level classifier pattern recognition system

In the first-level classifier,[14,15] the central moment ratio μ_{20}/μ_{02} is used to estimate the aspect ratio of the input object and a hierarchical node tree is used to provide class estimates. The node selection is automated from scatter plots onto a multi-dimensional Fisher space obtained from the μ_{pq} for the training set data. The branch selection is automatically determined from similar two-class Fisher projections. This first-level classifier reduces the number of aspect view classes that the second-level classifier must handle. It also allows the jointly Gaussian random variable nature of $\underline{m}$ features with respect to sampling to be employed in a Bayesian classifier. The discriminant function calculated in the second-level classifier for each aspect view class i is

$$g_i(\hat{\underline{m}}) = [\hat{\underline{m}} - \underline{m}_i(\underline{b})]^T \underline{\Sigma}_i^{-1} [\hat{\underline{m}} - \underline{m}_i(\underline{b})], \tag{5}$$

where $\underline{\Sigma}_i$ is the covariance matrix for class i, $\hat{\underline{m}}$ is the observed moment vector and $\underline{m}_i(\underline{b})$ denotes the reference moment vector for class i with distortions described by the distortion vector $\underline{b} = (x_0, y_0, a, b, R, \theta)$, where (x_0, y_0) are translations of the input object,

(a,b) are its horizontal and vertical scale changes, R is its range and θ is its in-plane rotation angle. For each aspect view class, (5) is evaluated for an initial $\underline{b}^0$ estimate obtained from the $\hat{\underline{m}}$, and a new $\underline{b}^k$ estimate at iteration k is calculated from the nonlinear estimator

$$\underline{b}^{k+1} = \underline{b}^k + [(\underline{J}^k)^T \underline{\Sigma}_i^{-1} \underline{J}^k]^T \underline{J}^T \underline{\Sigma}_i^{-1} [\hat{\underline{m}} - \underline{m}_i(\underline{b}^k)]. \quad (6)$$

Eq.(5) is then evaluated for this new $\underline{b}^k$ estimate and the process is continued. The aspect class i and the distortions $\underline{b}$ that yield the lowest $g_i(\hat{\underline{m}})$ define the object class, aspect and distortion parameter estimates. The $g_i(\hat{\underline{m}})$ value is a measure of the confidence of our estimate.

Excellent performance (over 90% correct class recognition has been obtained with this parallel algorithm and architecture for our 180 image ship database[15] and for a 32 image five-class pipe-part robotic database[14] and on non-controlled real infrared imagery.[20] In each case, 3-D aspect distortions of all aspects over all 360° were used. The algorithm in (5) and (6) requires 6500 operations per iteration and in general requires only six iterations. Thus, the full architecture of Figure 6 is quite parallel, efficient, automated, has a sound theoretical basis and has demonstrated excellent initial performance results.

4. DISTORTION-INVARIANT OPTICAL CORRELATORS

The optical correlator of Figure 1 has the multi-object, processing gain and performance and noise features noted earlier. To form a MSF at P_2 of Figure 1 that yields a distortion-invariant correlation, we employ a training set of images $\{f_n\}$ of different distorted versions of the object f in one class. We form the correlation matrix R_1 for this data set and restrict the filter function h(x,y) to be a linear combination of the training set data

$$h(x,y) = \Sigma\, a_n\, f_n(x,y). \quad (7)$$

The MSFs with distortion-invariant features that we discuss herein are referred to as synthetic discriminant functions (SDFs). Five different types of SDFs have been defined.[16] We now emphasize the algorithms to produce these SDFs and briefly discuss their performance. The SDF synthesis algorithm is computed off-line on training set data. Once synthesized, the SDF can be used on-line in the parallel correlator of Figure 1 with no additional computational overhead.

To produce a filter function h such that its correlation with all $\{f_n\}$ is a constant value of unity, i.e.

$$f_n \circledast h = 1, \tag{8}$$

the filter in (7) is defined by

$$\underline{a} = \underline{R}_1^{-1}\underline{u}, \tag{9}$$

where $\underline{u}$ is the unit vector. This is referred to as an equal correlation peak (ECP) SDF. It is useful for intra-class pattern recognition. To achieve inter-class discrimination with one object per class, we desire N filters h for an N-class problem such that

$$f_j \circledast h_i = \delta_{ij}. \tag{10}$$

These mutual orthogonal function (MOF) SDFs are defined by

$$\underline{a}_i = \underline{R}_2^{-1}\underline{u}_i, \tag{11}$$

where $\underline{R}_2$ is the full correlation matrix of the N object classes, $\underline{a}_i$ denotes the selection of coefficients for filter i, and $\underline{u}_i$ contains all zeroes with a single 1 in location i. The extension to intra-class MOF SDFs $h_j(x,y)$ follows directly. The filter function is now a sum over all N_1N training set images (N_1 images per class and N object classes). The coefficients of the filter $\underline{h}_j$ are

$$\underline{a}_j = \underline{R}_3^{-1}\underline{u}_j, \tag{12}$$

where $\underline{R}_3$ is the full $N_1N \times N_1N$ correlation matrix and $\underline{u}_j$ contains all zeroes except for N_1 ones in the locations corresponding to the class j training set images.

Another SDF that achieves inter-class discrimination and intra-class recognition is the multi-level nonredundant filter (NRF) SDF. In this filter, the correlation output is allowed to assume different levels,

$$f_n \circledast h = n, \tag{13}$$

where the value n of the correlation output determines the output class n. Synthesis of the simple SDF to satisfy (13) is defined by

$$\underline{a} = \underline{R}_4^{-1}\underline{u}_n, \tag{14}$$

where $\underline{u}_n = [1,...,1,2,...,2,...]^T$. To retain binary valued outputs, we can employ a K-tuple NRF SDF. For M-object classes, we require K filters, where $2^K \geq M$. For the four-class case, $K = 2$ and the two SDFs h_1 and h_2 are defined by the truth table

INPUT CLASS	OUTPUTS FOR EACH FILTER	
	h_1	h_2
1	0	0
2	0	1
3	1	0
4	1	1

(15)

Each filter is defined as a linear combination of all training set images as

$$h_1 = \Sigma\, a_n\, f_n, \quad h_2 = \Sigma\, b_n\, f_n. \tag{16}$$

The solution for the filters is given by the solution of

$$\underline{R}_5[\underline{a}\ \underline{b}] = [\underline{u}_1\ \underline{u}_2], \tag{17}$$

where $[\underline{u}_1\ \underline{u}_2]$ is the full vector extension of the right-hand side of (15).

The synthesis of all five types of SDFs described above is quite similar and other variants are obvious. Many test results have been obtained with these parallel algorithms and architectures on our ship image database[17,18] and on other military objects.[19] Excellent results have been obtained (over 90% correct object classification, even in the presence of noise and real-world clutter) in all cases. This represents the most attractive and promising technique for utilization of the full potential and parallel processing possible with coherent optical pattern recognition architectures. It is a practical and efficient processor and it achieves very high effective computation and image frame rates.

5. OPTICAL LINEAR ALGEBRA SYSTOLIC PROCESSORS

The optical processors described thusfar are quite powerful and appropriate for the parallel realization of various pattern recognition algorithms. The most intense topic in optical computing research at present is optical linear algebra processors.[21] These architectures provide the basic framework for a general-purpose optical processor capable of matrix-vector operations. This concept in parallel optical processing is the equivalent of the digital array processor in which arrays of data (matrices) are operated on in parallel.

The optical system of Figure 7 is one example of such a processor. In this system an array of input point modulators is imaged through separate regions of an acousto-optic (AO) cell. With the input data representing a vector and the contents of the AO cell being a matrix (N vectors each on a separate temporal input carrier), the light leaving the AO cell is the product of the input vector and matrix. The output lens forms the sum of each vector product by spatial integration and the matrix-vector product appears on the linear detector array in parallel.

Various realizations of this processor are detailed elsewhere.[22] By frequency, time and space-multiplexing, format control of the inputs to the system can be used to achieve all of the fundamental operations in linear algebra. This flexible and general-purpose processor can achieve in excess of 10 GOPs per second. Alternate architectures (with multi-channel AO cells) allow digital accuracy (32 bit) processing to be achieved with this processor at comparable data rates.

ACKNOWLEDGMENTS

The support of the research reported herein by the Air Force Office of Scientific Research (Grant AFOSR-79-0091 and F49620-83-C-0100) and the Internal Research and Development Funds of General Dynamics - Pomona Division (Agreement No. 539004) is gratefully acknowledged.

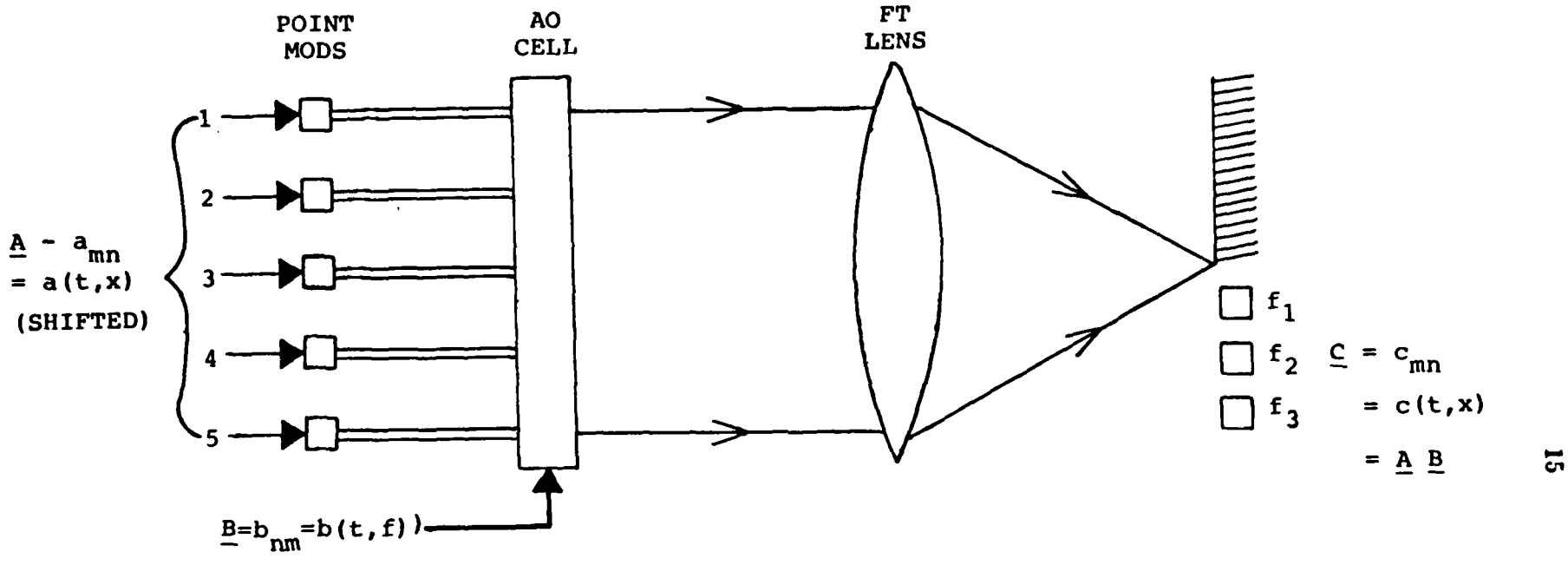


FIGURE 7 Schematic diagram of a general-purpose optical linear algebra processor

REFERENCES

1. J. Upatnieks, "Portable Real-Time Coherent Optical Correlator", Applied Optics, Vol. 22, p. 2798-2803 (September 1983).

2. J.G. Duthie and J. Upatnieks, "Compact Real-Time Coherent Optical Correlators", Optical Engineering, Vol. 23, pp. 7-11 (January/February 1984).

3. G.G. Lendaris and G.L. Stanley, "Diffraction-Pattern Sampling for Automatic Pattern Recognition", Proc. IEEE, Vol. 58, pp. 198-216 (February 1979).

4. H. Kasdan and D. Mead, "Out of the Laboratory and into the Factory: Optical Computing Comes of Age", Proc. EOSD, pp. 248-256 (1975).

5. S. Watanabe, 4th Prague Conference on Information Theory (1965).

6. K. Fukunaga and W.L.G. Koontz, "Application of the Karhunen-Loeve Expansion to Feature Selection and Ordering", IEEE, Trans. Comp., Vol. C-19, p. 311-317 (April 1970).

7. D.H. Foley and J.W. Sammon, "An Optimal Set of Discriminant Vectors", IEEE, Trans. Comp., Vol. C-24, pp. 381-389 (March 1975).

8. D. Casasent and V. Sharma, "Feature Extractors for Distortion-Invariant Robot Vision", Opt. Engrg., Vol. 23 (November 1984).

9. G. Tenery, "A Pattern Recognition Function of Integral Geometry", IEEE Trans. Mil. Electron., Vol. ME-7, pp. 196-199 (1963).

10. D.J.H. Moore and D.J. Parker, "Analysis of Global Pattern Features", Pattern Recognition, Vol. 6, pp. 149-164 (1974).

11. D. Casasent and W-T. Chang, "Generalized Chord Transformation for Distortion-Invariant Optical Pattern Recognition", Applied Optics, Vol. 22, pp. 2087-2094 (July 1983).

12. D. Casasent, R.L. Cheatham and D. Fetterly, "Optical System to Compute Intensity Moments: Design", Applied Optics, Vol. 21, pp. 3292-3298 (September 1982).

13. G.J. Agin, "Computer Vision Systems for Industrial Inspection and Assembly", Computer, Vol. 13, pp. 11-20 (May 1980).

14. D. Casasent and R.L. Cheatham, "Hierarchical Pattern Recognition Using Parallel Feature Extraction", Proc. Am. Soc. Mech. Engrs. Intl Computers in Engineering Conference and Exhibit' (August 1984).

15. D. Casasent and R.L. Cheatham, "Hierarchical Fisher and Moment-Based Pattern Recognition", Proc. Soc. Photo-Opt. Instr. Engrs., Vol. 504 (August 1984).

16. D. Casasent, "Unified Synthetic Discriminant Function Computational Formulation", Applied Optics, Vol. 23, pp. 1620-1627 (May 1984).

17. D. Casasent and V. Sharma, "Shift-Invariant and Distortion-Invariant Object Recognition", Proc. Soc. Photo-Opt. Instr. Engrs., Vol. 442, pp. 47-55 (August 1983).

18. D. Casasent, W. Rozzi and D. Fetterly, "Deterministic Synthetic Discriminant Functions: Performance", Optical Engineering, Vol. 23 (November 1984).

19. D. Casasent, W-T. Chang and D. Fetterly, "Deterministic Synthetic Discriminant Functions for Pattern Recognition", Proc. Soc. Photo-Opt. Instr. Engrs., Vol. 507 (August 1984).

20. D. Casasent and R.L. Cheatham, "Optical Moment-Based Feature Extraction: Real Image Test Results", Optics Communications (Submitted April 1984).

21. D. Casasent, "Acousto-Optic Linear Algebra Processors: Architectures, Algorithms and Applications", Proc. IEEE, Vol. 72 (July 1984).

22. Proc. IEEE, Special Issue on Optical Computing, Vol. 72 (July 1984).

5. FOURIER TRANSFORM FEATURE-SPACE STUDIES

SPIE, VOL. 449
CAMBRIDGE
NOV. 1983

FOURIER-TRANSFORM FEATURE-SPACE STUDIES

David Casasent and Vinod Sharma

Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

ABSTRACT

A hierarchial multi-level feature-space pattern recognition system is described. Multi-class distortion-invariant object identification is the purpose of this study. Attention is given to dimensionality reduction (to simplify computations) and to the use of non-unitary transformations (to achieve discrimination). A Fourier transform feature space is used. However, our basic hierarchial concepts, our theoretical analysis, and our general conclusions are applicable to other feature spaces. The use of intensity versus phase features is studied and the performance of our system in the presence of noise is studied. Quantitative experimental data on 2 two-class pattern recognition databases are provided.

1. INTRODUCTION

Distortion-invariant multi-class pattern recognition is considered using a feature space. Feature extraction, dimensionality reduction, discrimination and classification are addressed. A simplified block diagram of our hierarchial pattern recognition system is shown in Figure 1. We begin with a Fourier transform feature space, since such a representation is well-known [1] to allow significant data compression. We extract the amplitude, phase or both from the Fourier transform plane. As the first dimensionality reduction technique, we wedge ring detector (WRD) sample the Fourier transform plane data [2]. This reduces the dimensionality of the feature space to 64. Next, we compute the dominant eigenvectors of the WRD-sampled autocorrelation matrix. This reduced subspace is calculated using a Karhunen-Loeve (K-L) transformation [3] or implemented by new efficient techniques [4] for computing the dominant eigenvectors and eigenvalues of a large matrix. This completes the dimensionality reduction steps in our system. To provide discrimination, we employ two non-unitary transformation: the Fukunaga-Koontz (F-K) [5] and the Foley-Sammon (F-S) [6]. Our classifier selects the best subspace (based on the probability of error) from the K-L, F-K and F-S feature vectors.

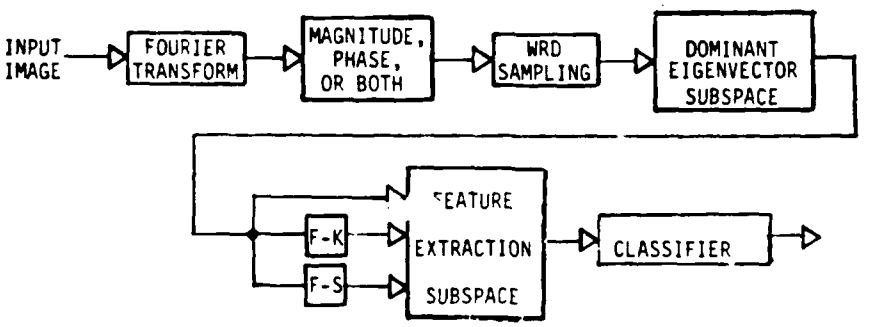


FIGURE 1. General Fourier Transform (etc.) Feature-Extraction Pattern Recognition System Block Diagram.

In Section 2, we review and highlight our two levels of dimensionality reduction and we discuss how we achieve distortion-invariance. Our two discrimination algorithms are detailed in Section 3. Brief theoretical remarks on the use of Fourier transform plane intensity or amplitude features and on the noise performance of a feature extractor are then advanced in Section 4. The database used and our five image tests on dominant eigenimage feature vectors are summarized in Section 5. Our more extensive 25 image per class database tests are presented in Section 6. These results include a comparison of the performance of our system for five different discrimination vectors, comparison of the performance of amplitude and phase Fourier transform features, and a comparison of the classifiers and feature extractors in the presence of noise.

2. DIMENSIONALITY REDUCTION AND DISTORTION-INVARIANCE

If the input image or object is 256 x 256 pixels, its dimensionality is $n = 256^2$. The Fourier transform plane for such an object still has a dimensionality of n. This is quite prohibitive for subsequent feature-extraction, matrix transformations, or other similar operations. Thus, we consider dimensionality reduction techniques.

As the first level of dimensionality reduction, we sample the Fourier transform plane with a WRD. If an optical system is used to produce the Fourier transform, a commercial WRD detector exists [2]. This units consists of 32 wedge-shaped detector elements in one-half of a circular detector and 32 annular-shaped detector elements in the other half of the detector plane. This device thus provides 64 WRD outputs. One can also digitally model such a device, of course. The ring detector elements provide rotation-invariance, whereas the wedge detector elements provide scale-invariance (if the values of the wedge-ring detector element readings are properly normalized for object energy). This WRD-sampling, plus the training of our system on different distorted images provides distortion-invariance to our algorithm. The WRD-sampling also provides a dimensionality reduction from n to 64, i.e. the Fourier transform plane feature vectors $\{x_i'\}_{i=1}^{n}$ and $\{y_i'\}_{i=1}^{n}$ are converted to WRD feature vectors $\{x_i'\}_{i=1}^{64}$ and $\{y_i'\}_{i=1}^{64}$

As the second level of dimensionality reduction, we apply a K-L transformation [3] to the autocorrelation matrix formed from the WRD feature vectors for each separate object class. The autocorrelation matrix is formed from the 64 element x_i' vectors for each of the training set images x in class one and a second matrix is formed from the corresponding y_i' vectors of images in class two. The eigenvalues and eigenvectors of each matrix are calculated and tabulated. We retain the dominant n_x and n_y eigenimages per class. In general, $n_x = n_y = 1,2,3$. In our experiments, we retained only the dominant eigenimage for each class.

To use these dominant eigenimages for pattern recognition and classification, we would compute z_i' (i=1...64) for an unknown input z, project it onto the dominant eigenimages or eigenvectors KL-1 and KL-2 (for class one and two respectively), and select the class of the unknown input based upon which projection value is larger. In practice, we calculate the dominant eigenvectors using newer and more efficient algorithms [4].

3. NON-UNITARY TRANSFORMATIONS

The K-L or dominant eigenvector transformation (Section 2) represents a considerable compression of data and simplifies feature extraction and classifier decisions. The dominant eigenvectors represent each class well in the optimally compressed manner, however there is no assurance that those features which represent each class well will be optimal for discriminating one class from another. Thus, dominant eigenimages are useful for intra-class pattern recognition, but not necessarily for inter-class discrimination. In a hyperspace feature vector and discriminant vector description, unitary transformations do not change the distances of points or vectors in hyperspace. To achieve discrimination or inter-class pattern recognition, non-unitary transformations represent an attractive approach. These transformations can increase interclass distances and hence provide improved discrimination. We pursue this approach rather than employing more eigenvectors, since the latter approach would only further increase the dimensionality and computational complexity of our processing.

F-K Transformation

The first non-unitary transformation we consider is the F-K transformation [5]. To describe the steps in this algorithm, we first define P_i as the a'priori probability for class i and R_i' as the autocorrelation matrix for class i. We form the autocorrelation matrices R_1 and R_2 for each class, where $R_i = P_i R_i'$, and we form the full autocorrelation matrix $R = R_1 + R_2$. We then determine the transformation matrix T that diagonalizes R, i.e.

$$TRT^T = T(R_1 + R_2)T^T = I. \tag{1}$$

By this transformation, we have orthogonally decomposed the full R_1+R_2. Next, we apply T to R_1 and R_2, i.e., we form new matrices for each class given by TR_1T^T and TR_2T^T.

These new correlation matrices have two attractive features:

(a) The eigenvectors $\psi_i^{(1)}$ and $\psi_i^{(2)}$ of TR_1T^T and TR_2T^T are equal

(b) The eigenvalues $\lambda_i^{(1)}$ and $\lambda_i^{(2)}$ associated with $\psi_i^{(1)}$ and $\psi_i^{(2)}$ are related by

$$\lambda_i^{(1)} = 1 - \lambda_i^{(2)}. \tag{2}$$

From (2), we see that the dominant eigenvectors of the transformed class one matrix are the least-dominant eigenvectors for the transformed class two matrix. Thus, those eigenvectors which represent class one the least represent class two the best (in the new F-K transformed feature space). From (2), these operations have separated the data in the two classes. Thus, we will select two ψ_i with the largest $|\lambda_i^{(1)} - 0.5|$ values. We will denote these two eigenvectors ψ_i (as defined above) of the transformed autocorrelation matrices by FK-1 and FK-2. To use these new discriminant vectors to determine the class of an unknown input image $\underline{z}$, we transform $\underline{z}$ to a new $\underline{T}\,\underline{z} = \underline{z}'$. This transforms the data input to the new FK space. We then project $\underline{z}'$ onto an FK discriminant vector ψ_i and calculate $\psi_i^T z' = d$. If d $\lesseqgtr$ our threshold, we select class one or class two for the class of the input object. We normalize the FK eigenvectors and refer to the projections onto the FK directions 1 and 2 (corresponding to FK-1 and FK-2). We note that FK-1 and FK-2 do not refer to discriminant vectors for classes one and two, rather they refer to the two most dominant eigenvectors of the transformed full autocorrelation matrix of both classes.

Foley-Sammon Transformation

In the F-S nonunitary transformation [6], we find a linear discriminant vector $\underline{w}$ that is a linear combination of the x_i and y_i vectors in our two-class training set. The vector $\underline{w}$ is selected to maximize the Fisher ratio [7]:

$$F(\underline{w}) = \frac{(\text{Dif of Means of Projections})^2}{\text{Sum of Scatter of Projections}} = \frac{\left|m_1 - m_2\right|^2}{s_1^2 + s_2^2} = \frac{\underline{w}^T \underline{S}_B \underline{w}}{\underline{w}^T \underline{S}_W \underline{w}}, \tag{3}$$

where $\underline{S}_B$ is the *between-class scatter matrix* and $\underline{S}_W$ is the within-class scatter matrix [7]. The solution for $\underline{w}$ that maximizes (3) is

$$\underline{w} = \underline{S}_w^{-1}(\underline{m}_1 - \underline{m}_2), \tag{4}$$

where $\underline{m}_1$ and $\underline{m}_2$ are the vector means of the two classes. To use $\underline{w}$ for an unknown input $\underline{z}$, we form $\underline{w}^T\underline{z} = d$ and compare the projection value to the threshold T

$$T = (m_1 + m_2)/2. \tag{5}$$

If d $\gtrless$ T, we select class one or class two for the class of the unknown input image represented by the vector $\underline{z}$.

4. INTENSITY OR PHASE FOURIER TRANSFORM FEATURES

One particular aspect of our Fourier transform feature-space study is to determine if the intensity or the phase of the Fourier transform features provides better performance. As the basic theoretical justification for the performance and use of our algorithm, we represent the intensity or phase of the wedge ring detected Fourier transform output for an image as a random process. We have extensively investigated the theoretical basis for this and the conditions to be satisfied for the resultant $\{x_i\}$ and $\{y_i\}$ features to be validly represented as random vectors. We have shown that the Fourier transform of an analog or discrete image can be represented by a random process $\{x(t)\}$ if: $\{x(t)\}$ is separable, has finite expected value, is continuous in the mean and in probability. Under these conditions, the resultant analog or digital Fourier transform is an n-dimensional random vector. We have also shown that the Fourier transform intensity and phase are continuous and yield real random vectors. Finally, the WRD-sampled intensity and phase Fourier transform features are found to be random variables (since the sum of random variables is a random variable).

Considerable work [8,9] exists on the representation of image data by the intensity or phase of the Fourier transform. In general, the conditions under which the Fourier transform phase features are adequate is less restrictive than the conditions for which the Fourier transform magnitude features are adequate. If the zeroes of the z-transform of a sequence occuring in reciprocal pairs lie on the unit circle, the phase of the FT is adequate. The intensity of the FT is adequate if the z-transform does not contain reciprocal pole-zero pairs, poles outside the unit circle, or zeroes inside the unit circle.

5. DATABASES AND INITIAL RESULTS

The four image databases used are summarized in Table 1. They include scaled and rotated images of the letters A and B and of hand-drawn tanks and trucks. For each of these two object classes, we used a set of five images per class and a set of 25 images per class. Various scaled and rotated views were included in each of these image sets. The specific distorted object views included in each case are detailed in Table 1.

TABLE 1. Summary of Experimental Image Databases Used

TEST SETS	FIVE-IMAGE DATA BASE		25-IMAGE DATA BASE	
	SCALES	ROTATIONS	SCALES	ROTATIONS
A and B	0.9,1.0,1.1	0°,10° (for 0.9,1.1 scales)	0.8,0.9,1.0, 1.1,1.2	±10°,±5°,0° (for each scale)
Hand-Drawn Tank/Truck	0.9,1.0,1.1	0°,10° (for 0.9,1.1 scales)	0.8,0.9,1.0, 1.1,1.2	±10°,±5°,0° (for each scale)

In Table 2, we list all of the eigenvalues for the dominant eigenvectors for the five-image database for all four object types and for both intensity and phase Fourier transform features. As seen, the dominant eigenvectors for intensity FT features is approximately 70 times the second dominant eigenvector. Using phase FT features, the dominant eigenvector is considerably less dominant (in general). The eigenvalue for the dominant eigenvector for A obtained from FT phase data is exceptionally low (0.67). From the low (0.67) eigenvalue associated with the dominant eigenvector for Fourier transform phase features for the letter A, we expect low projection values and hence more errors in our pattern recognition of letters using phase features. In general, the dominance of the eigenimage in this data can be attributed to the fact that the image database consists of scaled and rotated (in-plane rotation) images rather than different aspect views of each object. In such distorted images, there is no appreciable new information present in each object representation in our databases investigated.

TABLE 2. Eigenvalues (e-v) of WRD Fourier Transform Eigenvectors (Five-Image Databases).

WRD FEATURES	TRUCK		TANK		A		B	
	INT	PHASE	INT	PHASE	INT	PHASE	INT	PHASE
e-v 1	0.983	0.992	0.983	0.886	0.99	0.67	0.99	0.95
e-v 2	0.17×10^{-1}	0.78×10^{-2}	0.166×10^{-1}	0.983×10^{-1}	0.71×10^{-2}	0.24	0.13×10^{-1}	0.43×10^{-1}
e-v 3	0.82×10^{-4}	0.28×10^{-3}	0.21×10^{-3}	0.117×10^{-1}	0.84×10^{-4}	0.72×10^{-1}	0.49×10^{-3}	0.186×10^{-2}
e-v 4	0.81×10^{-5}	0.115×10^{-3}	0.64×10^{-4}	0.236×10^{-2}	0.47×10^{-4}	0.11×10^{-1}	0.28×10^{-4}	0.77×10^{-3}
e-v 5	0.49×10^{-6}	0.11×10^{-4}	0.11×10^{-5}	0.138×10^{-2}	0.65×10^{-5}	0.38×10^{-2}	0.17×10^{-4}	0.70×10^{-4}

The projections of all five images per class on the dominant eigenimage for each class were tabulated. Our results show that the projections of all images in one class on the dominant eigenimage for that class were larger than the projections on the dominant eigenimage of the other class. However, the differences (for intensity FT features) were quite small (e.g. 0.99 versus 0.96). The projections on the second dominant eigenimages (for intensity FT features) gave lower projection values than those on the dominant eigenvector. In many cases, larger projections were obtained for the wrong images. Thus, for intensity FT features, the second dominant eigenvector should not be included. These results support our earlier observation that the eigenvectors generally provide intra-class recognition rather than inter-class discrimination. A comparison was also made of the dominant eigenimages obtained from the intensity-only and phase-only WRD Fourier transform features. The phase features provided larger differences of the projections onto the dominant eigenimages of each class (on the average) for the five-image tank versus truck images. However, for the five-image letter (A and B) database, phase features gave many errors. This was expected and is attributed to the small eigenvalue associated with the dominant eigenvector for A. Had we retained two dominant

eigenvectors, better phase feature performance for the case of letters coule be expected.

6. INITIAL EXPERIMENTAL RESULTS

6.1 Non-Unitary Transformations

All of the results included in this section were obtained on our more extensive database of 25 object images per class. In Figure 2, we show the scatter plots for the projections of all tank and truck images onto the dominant eigenimages for tanks and for trucks respectively. As seen, all images can be separated and correctly classified from either projection alone. However, all projection values (even those on the dominant eigenimage of the other class) are quite large (all projection values are above 0.95). Only five points (X) are shown for the 25 training images corresponding to different scaled and rotated truck images. All five rotated images for each scale factor α yielded identical projection values. This verifies the good rotation - invariance of our WRD features and our training set used. The variation in the projection values due to scale differences can be attributed to the normalization technique used (each eigenimage was normalized only within one class) and to the significantly larger number of pixels present in the tank object compared to the truck object (800 versus 280 pixels).

In Figures 3 and 4, similar data are shown for the projections onto the two dominant FK discriminant vectors (Figure 3) and onto the best FS vector (Figure 4). These data in Figures 2-4 were obtained from intensity-only WRD Fourier transform features. Note the significantly different axes scales in Figures 2-4. To compare which feature extraction technique (dominant eigenvectors, FK vector, FS vector) yields the best performance, we computed a separation measure

$$\text{SEPARATION MEASURE} = \frac{\text{DIFFERENCE OF MEANS OF PROJECTIONS PER CLASS}}{\text{SUM OF STANDARD DEVIATIONS PER CLASS}} \qquad (6)$$

for five different discriminant vectors for the tank/truck and A/B image sets. The results (for intensity-only WRD Fourier transform features) are summarized in Table 3. As shown, both dominant eigenimages (KL-1 and KL-2) perform quite well (even though they only achieve intra-class compression). This can be attributed to the a'priori existence of different wedge and ring Fourier transform features for the two object classes and to the distortion-invariance and lack of information loss incurred by wedge ring detection sampling of the Fourier transform plane. For the tank/truck images, the performance of both F-K features is comparable, whereas the performance of the F-S vector is slightly better. For the case of the letters A and B, the non-unitary transformations achieve considerable improvement (by approximately a factor of 2). Thus, in some cases, non-unitary transformations will improve performance. The results are quite data dependent. These non-unitary transformations do not degrade performance and in general improves performance. Thus, such feature-extraction techniques appear to be merited in all instances.

6.2 Noise Performance

To further test and compare our different feature-extraction approaches, we added noise to the image data, recalculated the projections and the associated separation measures. The results are summarized in Table 4 for our 25 feature vectors, for five different amounts of noise and for both databases. For the tank and truck image data, very little difference occurs as the noise level is varied. This can be attributed to the fact that only several wedge and ring elements dominate the feature vectors. Since the noise is evenly distributed over all wedge and ring feature elements, its effect on the dominant feature elements is reduced and noise has less of an effect. For the case of letter recognition and classification, the separation measure decreases as the noise level is increased. This is the general trend we would expect. The amount of decrease is generally the same for all five discrimination vectors. The difference in the case of letters and tanks/trucks can be attributed to the fact that letters have more structure and hence Fourier transform plane information is more uniformly distributed over all of the wedge and ring sampling elements. Hence, the effects of noise is more fully transferred to such a feature-space. It should be noted that in all cases, good performance was obtained.

6.3 WRD Fourier Transform Phase versus Intensity Features

Similar tests were performed for the case of phase-only WRD Fourier transform features. For the tank/truck data, the separation measure for phase features was found to be better (by 10-65%) than for intensity features. For the A and B letter images, the phase features sometimes provided better separation measures, but in general gave worse results. This can be attributed to the small 0.66 eigenvalue of the dominant phase eigenvector for the letter A. Tests were also performed using combined intensity and phase features. For the tank/truck images, 10-90% better separation measures were obtained with phase features. For the

FIGURE 2. Intensity-Only WRD Fourier Transform Features Projected onto Dominant Tank/Truck Eigenimages (for 25 Image Database). ϕ_1=Dominant Truck Eigenimage; ϕ_2= Dominant Tank Eigenimage; •=Tank Images; X=Truck Images.

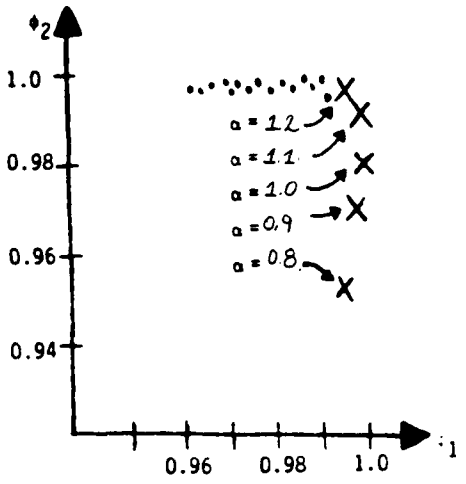


FIGURE 3. Intensity-Only WRD Fourier Transform Feature Projections for Tank/Truck Images onto FK Vectors (25 Images/Class).

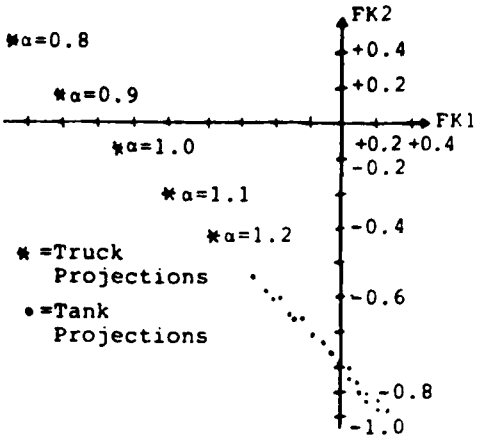


FIGURE 4. Projections of 25 Image/Class Tank/Truck Data on Best Foley-Sammon Vector.

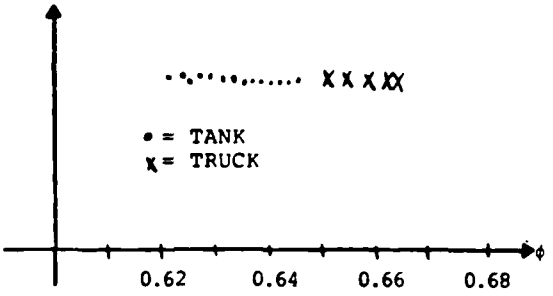


TABLE 3. Separation Measures for Different Intensity Feature-Extraction Techniques.

DATA / PROJ VECTOR	TRUCK AND TANK	A AND B
DOMINANT CLASS 1 EIGENIMAGE	4.13	7.09
DOMINANT CLASS 2 EIGENIMAGE	2.89	5.98
FK-1	3.91	12.1
FK-2	3.88	12.1
F-S	4.50	11.9

TABLE 4. Noise Performance of Intensity WRD Fourier Transform Features for Different Feature Extraction Techniques (Separation Measure Tabulated).

IMAGES	TRUCK AND TANK					A AND B				
STANDARD DEVIATION OF NOISE	0	0.1	0.2	0.3	0.4	0	0.1	0.2	0.3	0.4
DISCRIMINANT VECTOR										
KL (CLASS 1)	4.2	4.3	4.3	4.3	4.1	8.5	9.3	7.7	4.6	2.7
KL (CLASS 2)	2.9	2.9	2.9	2.9	2.8	6.9	6.6	5.0	3.3	2.4
FK-1	3.9	3.9	3.9	3.9	3.85	13.5	16.4	12.3	6.2	3.8
FK-2	3.9	3.9	3.9	3.9	3.8	13.5	16.4	12.3	6.2	3.7
F-S	4.6	4.7	4.7	4.7	4.7	13.3	17.0	13.9	6.7	4.0

letters A and B, the separation measure was never significantly better and often was significantly worse than when intensity-only features were used.

The performance of phase features with noise were also tested. For the tank/truck images, the separation measure decreased as the noise increased (by a factor of 0.3 to 10.0). This was quite significant and worse than the intensity feature results which showed negligible variation with noise. Similarly large reductions in the separation measure (by factors of 0.3 to 10.0) were obtained for the case of the letters A and B as the noise was increased. These losses were much larger than for the intensity features.

The conclusions reached from this limited testing are that phase features provide better separability. Better performance can be expected for the case of letters if the second dominant eigenvector is included. The noise performance of phase features appears to be worse. This can be attributed to the more uniform distribution of Fourier transform phase over all of the WRD features (compared to the concentration of the intensity features in fewer WRD elements).

8. SUMMARY AND CONCLUSIONS

In this paper, we have addressed a hierarchial multi-level general feature-space pattern recognition system for multi-class distortion-invariant object recognition. Attention was given to dimensionality reduction and its importance and success were demonstrated. The Fourier transform plane was found to allow significant dimensionality reduction. Wedge ring detector Fourier transform sampling and Karhunen-Loeve (KL) or dominant eigenvector calculations were found to allow considerable reduction and compression with little information loss. To provide discrimination, non-unitary transformations were used and found to either improve discrimination (or to provide negligible loss in performance). The Fukunaga-Koontz (FK) and Foley-Sammon (FS) non-unitary transformations were considered. Both perform comparably, with the FS technique being somewhat better.

Quantitative experimental data and excellent performance were obtained on various image databases. The dominant eigenimage performed quite well if it was very dominant. When it was not dominant, non-unitary transformations helped performance considerably. If several of the feature elements are dominant, noise performance improves. This provides further motivation for reducing the number of feature elements and for devising schemes in which only several features are dominant. Our theoretical contributions on random vector modeling, noise performance and sample matrix calculations are quite general and useful in many other feature-extraction problems. Our study of intensity and phase Fourier transform features found phase features to be preferable, but that phase features generally perform poorer in the performance of noise (since they are more uniformly distributed in Fourier transform space).

ACKNOWLEDGMENT

The support of this research by the Air Force Office of Scientific Research (AFOSR-79-0091) is gratefully acknowledged.

REFERENCES

1. H.C. Andrews, IEEE, Trans. Comp., p. 1045 (1971).
2. H. Kasdan and D. Mead, EOSD, p. 248 (1976).
3. S. Watanabe, 4th Prague Conference on Information Theory (1965).
4. H. Murakami and B.V.K. Vijaya Kumar, IEEE, PAMI, p. 511 (1982).
5. K. Fukunaga and W.L.G. Koontz, IEEE, Trans. Comp., p. 311 (1970).
6. D. Foley and J. Sammon, IEEE, Trans. Comp., p. 281 (1975).
7. R.O. Duda and P.E. Hart, "Pattern Classification and Scene Analysis", John Wiley and Sons (1973).
8. A.V. Oppenheim and J.S. Lin, Proc. IEEE, p. 529 (1981).
9. M.H. Hayes, J.S. Lin and A.V. Oppenheim, IEEE, ASSP, p. 672 (1980).

6. HIERARCHICAL PATTERN RECOGNITION USING PARALLEL FEATURE EXTRACTION

HIERARCHICAL PATTERN RECOGNITION USING PARALLEL FEATURE EXTRACTION

David Casasent and R. Lee Cheatham
Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

ABSTRACT

A hierarchical feature extraction pattern recognition technique is described and experimental test data is presented. The multi-level system estimates the class of the object and its aspect view in level one. A nonlinear iterative least squares estimator comprises the level two processor. A moment-based feature extractor is used. The level one system allows the classifier to use features that are jointly Gaussian random variables. Experimental results on a set of pipe images are presented.

1. INTRODUCTION

Feature extraction is a major computationally efficient approach to pattern recognition. In this paper we consider the use of a moment-based feature extractor for distortion-invariant object identification and classification. Moments were selected as the feature space to be used because of four unique aspects that these features exhibit:

(a) They can be computed in parallel [1].
(b) They allow easy correction after computation for various system computational errors [2].
(c) They provide position, orientation and scale information on the object [3].
(d) They are jointly-Gaussian random variables (JGRVs) [4] and hence allow use of a Bayesian classifier [5] and do not require a training set of imagery.

Our concern is to be able to recognize and classify objects in multiple classes independent of geometrical distortions due to the object's orientation and view angle and to estimate the distortions. The former is needed for object recognition and the latter for object control (i.e. by a robot).

In Section 2, we briefly review how the moment vector $\underline{m}_i(\underline{b})$ for an object in class i can be computed for different object distortions, described by $\underline{b}$, from a reference $\underline{m}_i(\underline{b}^0)$ vector (Section 2.1). We then review (Section 2.2) the conditions under which moment features are JGRVs and the simplified classifier that results (Section 2.3). To utilize the JGRV property of moments, a new classifier is required. This two-level classifier is summarized in Section 3 and demonstrated for a new database of pipe parts in Section 4.

2. MOMENT FEATURE SPACE

A moment feature space is most attractive for many reasons, several of which were noted in Section 1. In this section, we expand upon several of the less-detailed properties of moments, especially those aspects that apply to our new moment-based classifier.

2.1 Distortion Parameter Effects

For our application, we consider $\underline{b} = (a,b,x_0,y_0,\theta)$ where the elements of $\underline{b}$ contain the horizontal (a) and vertical (b) scale of the object, its translation (x_0, y_0) and in-plane rotation (θ) with respect to a reference $\underline{b}^0$ vector. Computing $\underline{m}_i(\underline{b})$ from $\underline{m}_i(\underline{b}^0)$ for different distortions described by $\underline{b}$ involves a simple matrix multiplication. For intensity changes by a factor k,

$$\hat{m}_{pq} = k\, m_{pq} . \tag{1}$$

For scale changes,

$$\hat{m}_{pq} = (1/a)^{p+1}(1/b)^{q+1} m_{pq} . \tag{2}$$

For translations,

$$\hat{m}_{pq} = \sum_{i=0}^{p} \sum_{j=0}^{q} \binom{p}{i} \binom{q}{j} (x_0)^i (y_0)^j m_{p-i,q-j} . \tag{3}$$

For in-plane rotations

$$\hat{m}_{pq} = \sum_{i=0}^{p} \sum_{j=0}^{q} \binom{p}{i} \binom{q}{j} (\cos\theta)^{p-i+j} \cdot (\sin\theta)^{q-j+i} m_{p+q-i-j,i+j} . \tag{4}$$

For reflections about the y axis (x axis reflections are similar),

$$\hat{m}_{pq} = (-1)^q m_{pq}. \quad (5)$$

Thus, new $\hat{\underline{m}}$ moments are easily calculated from the nominal $\underline{m}$ moments for different distortions.

2.2 Moment Statistics

Finite spatial-sampling of the object causes variations in $\underline{m}$. It is fairly straightforward to show that the statistics of these sampled moments are good estimates of the true moments and that the moment features are JGRVs. The number of object pixels required to satisfy the Gaussian pdf assumption is much less than the number of object pixels needed for recognition and classification. Thus, a JGRV model for moment features is quite valid. However, this is only true for those distortions $\underline{b}$; specifically the moments are JGRVs with respect to scale, translation and in-plane rotation θ, but not for out-of-plane rotations ϕ. Thus, with respect to the distortions in $\underline{b}$, the moments are JGRVs. Similarly, one cannot devise deterministic linear transformations as in Section 2.1 for the variations in $\underline{m}$ with ϕ. Thus, for all of the above reasons, different aspect views (ϕ) of each object must be considered as separate classes. We refer to these as view classes to distinguish them from object classes (different objects specifically).

2.3 JGRV Classifier

The conventional Bayesian classifier [6] that minimizes the probability of an incorrect view-class i estimate can be used with conventional assumptions (such as JGRV features) to obtain the discriminant function

$$g_i(\underline{x}) = (\underline{x}-\underline{\mu}_i)^T \underline{\Sigma}^{-1} (\underline{x}-\underline{\mu}_i), \quad (6)$$

where $\underline{\mu}_i$ and $\underline{\Sigma}_i = \underline{\Sigma}$ are the mean vector and covariance matrix for class i. In most cases, $\underline{\mu}_i$ and $\underline{\Sigma}_i$ must be estimated from training sets of imagery. When the measured feature vector $\underline{x}$ is a moment vector $\underline{m}$, only one object view per class is needed to measure $\underline{\mu}_i$ and $\underline{\Sigma}_i$. The class i that minimizes $g_i(\underline{x})$ is the best class estimate. The discriminant function in (6) is the Mahalanobis distance. If $\underline{\Sigma} = \underline{I}$, (6) becomes a Euclidean distance measure or nearest-neighbor classifier. This assumes that all moments are independent and that the expected variations of all moments are equal.

3. NEW CLASSIFICATION ALGORITHM

3.1 Overview

To utilize the classifier in (6), each view-class must be treated as a separate class i. For 9 objects and 36 aspect views per object (10° increments from a fixed depression angle), there are i = 324 view-classes to be searched. In Section 3.2, we describe our second-level classifier which solves (6) for the view-class i and the distortion parameter vector $\underline{b}$. Its parameters are discussed in Section 3.3. To reduce the number of view-classes i to be searched, a hierarchical first-level classifier is used in which estimates of the object class (Section 3.4) and the aspect view (Section 3.5) of the object are obtained and passed to the second-level classifier where the final decision on the view-class i and the distortion parameters $\underline{b}$ is made, and the confidence of our estimates are provided. A block diagram of the classifier is provided in Figure 1.

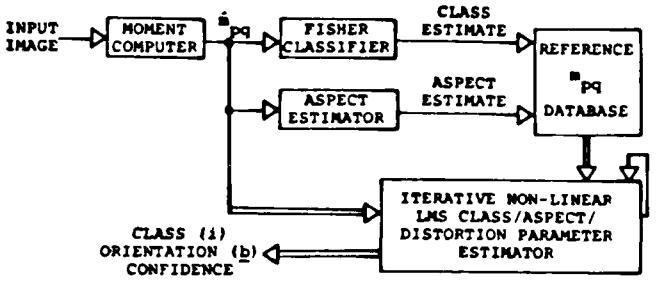


FIGURE 1 Block diagram of our Two-Level Moment-Based Classifier.

3.2 Iterative Second-Level Distortion-Parameter Estimator

The second-level classifier is described first to provide added motivation for the first-level-classifier. We desire to combine the ease with which $\underline{m}_i(\underline{b})$ can be calculated for new $\underline{b}$ vectors and the classifier in (6) to estimate $\underline{b}$ for the input object and to provide final estimates of the object class and aspect view. The basic concept is to vary i and $\underline{b}$ to minimize $\underline{e}_i = \hat{\underline{m}} - \underline{m}_i(\underline{b})$, where $\hat{\underline{m}}$ is the measured moment vector of the input object. The square error measure $E_i = \underline{e}_i^T \underline{\Sigma}^{-1} \underline{e}_i$ where $\underline{\Sigma}^{-1}$ is the weighting matrix used. To minimize E_i with respect to $\underline{b}$, an iterative algorithm is used since $\underline{m}(\underline{b})$ is a nonlinear function of $\underline{b}$. The algorithm is of the general form

$$\underline{b}^{k+1} = \underline{b}^k + a^k \underline{r}^k, \quad (7)$$

where $\underline{b}^k$ is the $\underline{b}$ estimate at iteration k and $\underline{b}^{k+1}$ is a point in an r-dimensional space a distance a^k in the direction $\underline{r}^k$ from the present $\underline{b}^k$. We expand $\underline{m}_i(\underline{b})$ in a Taylor series about the present $\underline{b}^k$ point as

$$\underline{m}_i(\underline{b}) = \underline{m}_i(\underline{b}^k) + \underline{J}^k(\underline{b}-\underline{b}^k), \quad (8)$$

where $\underline{J}$ is the Jacobian of $\underline{m}_i(\underline{b})$ with respect to $\underline{b}$ at the k-th iteration. Substituting $\underline{e}_i$ and (8) into E_i, solving for the minimum $\underline{b}$ (by setting $\nabla E_i(\underline{b}) = 0$), we obtain

$$\underline{b}^{k+1} = \underline{b}^k + [(\underline{J}^k)^T \underline{\Sigma}^{-1} \underline{J}^k]^{-1} (\underline{J}^k)^T \underline{\Sigma}^{-1} [\hat{\underline{m}} - \underline{m}_i(\underline{b})]. \quad (9)$$

Eq.(9) is the nonlinear iterative algorithm used in our second-level classifier to estimate $\underline{b}$. For each i, (9) is repeated and we calculate

$$\Delta g_i = [g^k(i,\underline{b}) - g^{k-1}(i,\underline{b})]/g^k(i,\underline{b}), \quad (10)$$

where $g_i(\underline{b}) = E_i$. The iterations k are continued until Δg_i is less than a convergence threshold T. The algorithm in (9) is the Gauss-Newton formulation, which degenerates to the Newton algorithm with appropriate assumptions [7].

3.3 Second-Level Classifier Parameters

The iterative algorithm in (9) and the second-level classifier requires selection of various parameters. These are summarized in Table 1 and discussed below. The convergence threshold T determines when the iterations k are stopped and how small (10) becomes. $T = 10^{-4}$

corresponds to a difference of 0.01% in (10). The confidence value C is a measure of the confidence of our estimates. It is obtained by measuring the distances d_1 and d_2 between the input $\hat{\underline{m}}$ vector and the two closest $\underline{m}_i(\underline{b})$ vectors and defined as $C = 100[1 - d_1/d_2]$, where $d_1 < d_2$. Calculation of $\underline{J}$ is simplified by evaluating it for $\underline{b}$ with $(x_0,y_0) = (0,0)$ and $(a,b) = (1,1)$, i.e. assuming the presently calculated distortion is correct. This is equivalent to viewing each iteration as an update of the prior $\underline{b}^k$ rather than the initial $\underline{b}^0$ estimate. This greatly simplifies calculation of $\underline{J}$ and (9) at each iteration.

TABLE 1 Second-Level Classifier Parameters

SYMBOL	PARAMETER	REMARKS
T	Convergence Threshold	Typically 0.1
C_2	Confidence Value	$C_2 = 100[1 - d_1/d_2]$
J	J Calculation	$\underline{b} = [1,1,0,0]^T$
Σ^{-1}	Σ^{-1} Calculation	$\underline{\Sigma}^{-1} = \underline{I}$ or $\underline{\Sigma}^{-1} = \underline{W}\,\underline{W}^T$
$\underline{b}^0$	(a,b,x_0,y_0)	$(\hat{m}_{00}^{\frac{1}{2}}, \hat{m}_{00}^{\frac{1}{2}}, -\hat{m}_{10}/\hat{m}_{00}, -\hat{m}_{01}/\hat{m}_{00})$

Calculation of $\underline{\Sigma}^{-1}$ is quite difficult since the exact $\underline{\Sigma}$ matrix is quite ill-conditioned. The two choices considered in our system are $\underline{\Sigma} = \underline{I}$ and $\underline{\Sigma}^{-1} = \underline{W}\,\underline{W}^T$. The choice $\underline{\Sigma} = \underline{I}$ weights all features equally and assumes independent features. In our first-level classifier, we calculate a multiclass Fisher projection matrix $\underline{W}$. In a Fisher space, $\underline{\Sigma}^{-1} = \underline{W}\,\underline{K}^{-1}\underline{W}^T$, where $\underline{K}$ is the covariance matrix of the Fisher features. With $\underline{K} = \underline{I}$, our second $\underline{\Sigma}^{-1}$ choice is obtained. The second $\underline{\Sigma}^{-1}$ estimate contains some information about the object separation of the reference set. Initial estimates of the distortion parameters in $\underline{b}^0$ are obtained directly from the measured $\hat{m}_{pq}$ as listed in Table 1.

3.4 First-Level Classifier: Object Class Estimates

To reduce the number of view-classes i for which the second-level classifier of (9) and (10) and Table 1 must be used, object class estimates are obtained in the first-level classifier. This is achieved by a hierarchical classifier. To provide invariance to the distortions in $\underline{b}$, the central moments $\underline{u}_i$ (normalized for translation and scale) for each $\underline{m}_i$ are computed. To select the best subsets at each node in the hierarchical node tree, a multclass Fisher projection matrix $\underline{W}$ of size m x (c-1) is computed, where m is the number of moments and c is the number of object classes. $\underline{W}$ is chosen to maximize the total between class scatter to total within-class scatter (with respect to the overall sample mean). The $\underline{u}$ normalized central moments are then transformed into this multidimensional Fisher space as new transformed feature vectors $\underline{y} = \underline{W}^T\underline{u}$. In practice, the $\underline{u}$ are projected onto only the two dominant Fisher feature vectors.

An example of such a multiclass projection for 9 objects (Section 4) is shown in Figure 2. In this classifier, all aspect views of each object (36 for the case shown) are viewed as different versions of each object to be clustered and separated from aspect views of the other objects. The various data points in each cluster correspond to scatter due to 36 aspect views per object class. From this multiclass Fisher projection, the object subsets (these may be multiple classes) that are best separated at node 0 in our hierarchical tree are determined. A new two-class Fisher discriminant vector $\underline{w}$ is then computed that optimizes the Fisher ratio for these two object subsets. The corresponding $\underline{w}^T\underline{u}_i$ projections for the two class subsets at a later node in the tree are shown in Figure 3. As seen from this data, the two object subsets (represented by the symbols 0 and 1) can easily be separated. As seen, the simple linear $\underline{w}$ discriminant defined by Fisher feature 1 for this node achieves this. Thus, our class estimator proceeds by forming multiclass Fisher projections of the available reference imagery and from this selecting the subsets to be separated at each node in the tree. A different multiclass case is considered at each node and a different two-class Fisher discriminant vector $\underline{w}$ is then calculated for use at each node. These procedures are performed on available reference imagery prior to classification and thus need not be performed in real-time. During classification, only the simple vector inner product $\underline{w}_n^T\underline{u}_i$ must be calculated for each node n. A confidence $C_1 = 35$ for the object class estimator is used at each node. If $C_1 < 35$, both subsets at that node are passed. C_1 is similar to C_2 but in Fisher space.

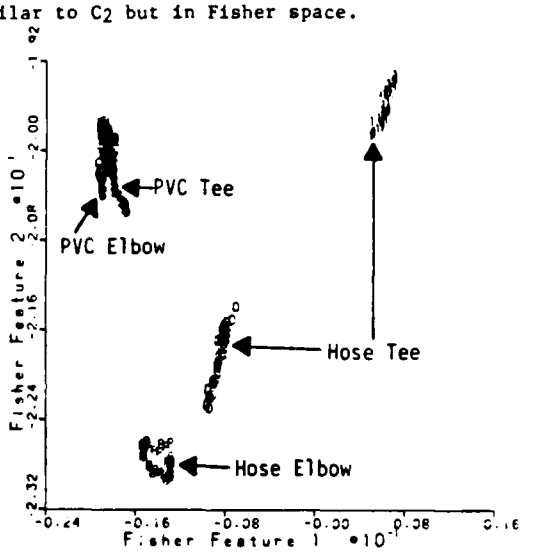


FIGURE 2 Multiclass Node-0 Fisher Projection for the Database of 9 Pipe Parts.

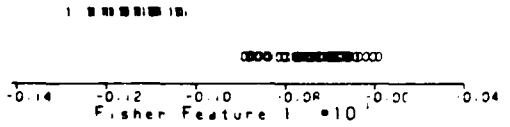
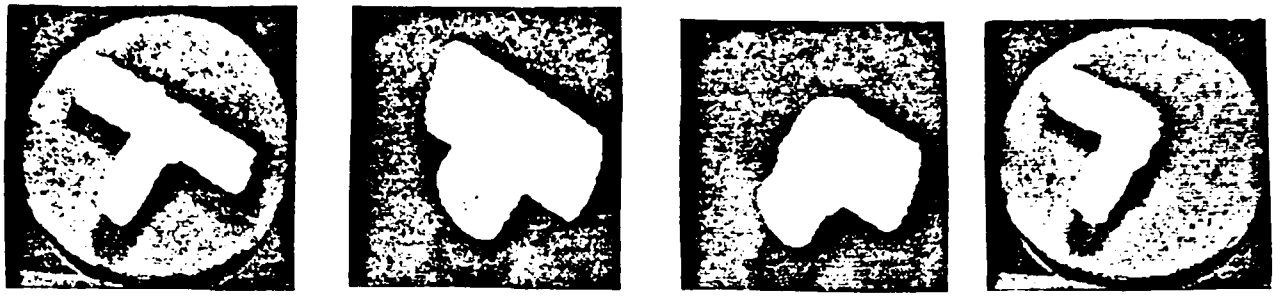


FIGURE 3 Two-Class Vector Inner Product $\underline{w}^T\underline{u}_i$ Fisher Projection at Node-2 for the Database of 9 Pipe Parts.



(a) Hose Tee (Class 0) (b) PVC Tee (Class 1) (c) PVC Elbow (Class 2) (d) Hose Elbow (Class 3)

FIGURE 4 Representative Images of Objects in the Four Main Classes.

TABLE 2
Effect of First-Level Classifier
(9 Aspect References Every 40° Used, Tested Against all 324 Images)

TEST NUMBER	CONDITIONS ON FIRST-LEVEL CLASSIFIER	AVERAGE NUMBER OF VIEW CLASSES PASSED TO SECOND-LEVEL	PERCENT CORRECT OUT OF 324
1	FIRST-LEVEL ENABLED	10.3	97.5
2	ASPECT-RATIO ESTIMATOR NOT USED	23.8	97.5
3	OBJECT-CLASS ESTIMATOR NOT USED	34.7	72.2
4	FIRST-LEVEL DISABLED	81.0	71.0

TABLE 3
Effect of Convergence Threshold on Number of Iterations Required

TEST NUMBER	ITERATION UPDATE INCREASED LINEARLY OVER N ITERATIONS	CONVERGENCE THRESHOLD RANGE	PERCENT CORRECT OUT OF 324	AVERAGE NUMBER OF ITERATIONS
5	N = 25	$T = 10^{-4} - 10^{-1}$	98.2	17 - 13
6	N = 5	$T = 10^{-4} - 10^{-1}$	98.2	6.3 - 5.7
7	N = 25	T = 0.5 - 1.0	98.2	2

TABLE 4
Effect of Covariance Matrix Estimate Used

TEST NUMBER	COVARIANCE ESTIMATE	PERCENT CORRECT
8	4 Fisher Vectors	93.9
9	2 Fisher Vectors	92.8
10	Identity	90.0

3.5 First-Level Classifier: Aspect Estimates

Once the object classes (that are sufficiently acceptable to be checked further in the second-level classifier) have been selected, those aspect views of each such object class (that should be included in the view classes i in the second-level classifier) are estimated from $\hat{m}_{pq}$. The use of moment features provides a quite convenient method. Specifically, we estimate the aspect ratio (ratio of its length to height) of the input object as $\hat{A} = u_{20}/u_{02}$, where $u_{20} = m_{20}-m_{10}^2/m_{00}$ and $u_{02} = m_{02}-m_{01}^2/m_{00}$. The aspect ratios A for all reference objects in the estimated object class(es) are calculated and $K = \hat{A}/A$ is formed. The aspect view with K closest to one and all aspect views within a factor $T_A = 1.5$ (the aspect threshold) of this are passed as a member of the view class i to be further processed by the second-level classifier. The value A can also be used to omit object class estimates (from Section 3.4) with no aspect ratio in the proper range. If a lower T_A value (closer to 1.0) is used, the number of aspect views per object class passed to the second-level classifier can be restricted to 1 or 2 with excellent final classification performance.

4. DATABASE RESULTS

4.1 Database

The new database used consisted of four different classes of pipe parts (Figure 4). Two different types of hose tees, four different types of PVC tees and two different types of PVC elbows were included (9 different pipe objects in four classes). 512 x 512 pixel digitized images of each of the 9 objects were obtained from a 50° depression angle at 10° aspect view increments (36 aspect views per object). These 324 images were reduced to 128 x 128 pixels and binarized.

4.2 Hierarchical Node Tree

The multiclass Fisher projections in Figure 2 show the scatter of the different pipe parts. From such plots, the subsets used at each node in the tree were chosen. Figure 5 shows the level-one classifier hierarchical node tree. A two-class Fisher discriminant vector is computed for each node and used to determine the subset choice at each node. An example of the scalar projections at node 2 was shown in Figure 3.

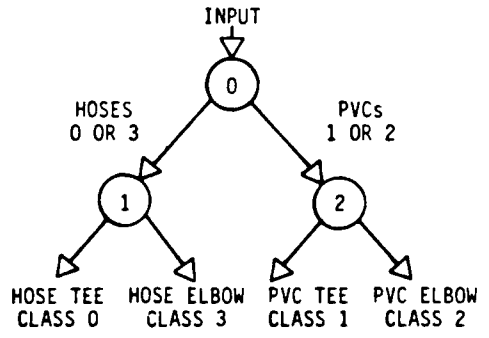


FIGURE 5 Hiearchical Node Tree for Class Estimation in the Level-One Classifier.

4.3 Experimental Results

Our extensive simulation tests are summarized in Tables 2-4. The nominal values used for the various classifier parameters were: convergence threshold $T = 10^{-4}$, confidence threshold $C_2 = 0$ (this forces a decision to be made for each input), covariance matrix $\underline{\Sigma} = \underline{I}$, $C_1 = 35$, $T_A = 1.5$. The reference set contains 9 aspects per class at 40° intervals. Unless otherwise noted, these conditions are used in each test. In Table 2, the overall performance obtained with and without the first-level classifier used is shown. As seen, over 97% correct object classification can be obtained (tests 1 and 2). If the aspect ratio estimator in the first-level classifier is not used (test 2), no performance change results, however about 2.4 times more view classes i must be checked in the second-level classifier. If the object-class estimator is not used (test 3), the number of view classes i to be checked in the second-level classifier is 3.5 times larger and performance is 25% poorer. Without the first-level classifier (test 4), performance is comparable to test 3 but all 81 view classes (9 reference aspects for each of 9 objects) must be searched. Thus, the first-level classifier both improves performance and reduces the number of computations needed. The object-class estimator controls performance and both the aspect and object-class estimators reduce computations.

In separate tests, various reference image sets with different numbers of aspect views (i.e. only four aspect views in one quadrant per object class) were used and achieved comparable results. In Table 3, the number of iterations required in the second-level classifier is quantified and the effect of varying the convergence threshold T is investigated. As seen, varying T has negligible effect on the percent of the images correctly identified. Smaller T values result in fewer iterations required. However, we suspect that better $\underline{b}$ estimates will result if more iterations are used. In Table 4, the effect of various covariance matrix $\underline{\Sigma}^{-1}$ estimates are considered. The identity matrix is found to perform adequately with only a few percent better accuracy resulting when different Fisher vectors are used to calculate $\underline{\Sigma}^{-1}$.

5. SUMMARY AND CONCLUSIONS

A new classifier using a moment-based feature space has been described. The second-level classifier is optimal and uses the JGRV property of the features with respect to distortions contained in $\underline{b}$. A hierarchical first-level classifier was included to improve performance and reduce the computational load on the second-level classifier. A new organized procedure for selecting the node structure, the subsets per node and the discriminant vector per node was advanced. A multiclass and conventional two-class Fisher discriminant technique was advanced to automate this procedure. Experimental verification and quantification of all aspects of both levels of the classifier were obtained for a pipe part database. Excellent results were obtained. This appears to be a most attractive and viable feature space pattern recognition system with many unique properties.

ACKNOWLEDGMENTS

The support of this research by the Air Force Office of Scientific Research (Grant AFOSR-79-0091) is gratefully acknowledged.

REFERENCES

(1) D. Casasent, R.L. Cheatham and D. Fetterly, "Hybrid Optical/Digital Moment-Based Robotic Pattern Recognition System", Proc. SPIE, Vol. 360, pp. 105-111, August 1982.

(2) D. Casasent, R.L. Cheatham and D. Fetterly, "Optical System to Compute Intensity Moments: Design", Applied Optics, Vol. 21, pp. 3292-3298, September 1982.

(3) M.R. Teague, "Image Analysis via the General Theory of Moments", J. of the Opt. Soc. of Am., Vol. 70, No. 8, pp. 920-930, August 1980.

(4) D. Casasent, J. Pauly and D. Fetterly, "Infrared Ship Classification Using a New Moment Pattern Recognition Concept", Proc. SPIE, Vol. 302, pp. 126-133, August 1981.

(5) R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, NY, 1973.

(6) K. Fukunaga, Introduction to Statistical Pattern Recognition, John Wiley and Sons, NY, 1972.

(7) R. Schmidt, Advances in Nonlinear Parameter Optimization, Lecture Notes in Control and Information Sciences, Vol. 37, Springer-Verlag, Berlin, 1982.

7. IMAGE SEGMENTATION AND REAL-IMAGE TESTS FOR AN OPTICAL MOMENT-BASED FEATURE EXTRACTOR

IMAGE SEGMENTATION AND REAL-IMAGE TESTS FOR AN OPTICAL MOMENT-BASED FEATURE EXTRACTOR

David CASASENT and R. Lee CHEATHAM
Carnegie-Mellon University, Department of Electrical and Computer Engineering, Pittsburgh, PA 15213, USA

Received 10 May 1984

A two-level classifier has been designed for use in a moment-based hybrid optical/digital processor. The simulation performance of this pattern recognition system using real IR input test images of ships and reference moments obtained from ship models is described with emphasis given to the preprocessing operations required.

1. Introduction

The use of optical processors to compute image features for feature-based pattern recognition has recently received renewed interest. The optically-computed image features thus far considered include Fourier coefficients [1–3], chord histogram distributions [4,5], and geometrical moments [6–8]. In this paper, a moment-based feature extractor and classification algorithm for pattern recognition is considered (section 2) and its performance in the classification of ship imagery (section 3) is addressed. Specific attention is given to classification of real input imagery (section 5) and the image preprocessing required (section 4).

2. Optical computation of the geometrical moments

The optical system considered to generate the moments of an input object [7] consists of an input plane P_1 (in which the input image is placed) imaged onto a moment generating mask at plane P_2. The monomials $x^p y^q$ up to fifth-order ($p + q \leq 5$) are recorded on the P_2 mask each spatially multiplexed using a different spatial frequency for each carrier. The optical Fourier transform of the light distribution leaving P_2 is detected on 21 multiple parallel output detectors in the P_3 output plane and contains the moments

$$m_{pq} = \iint f(x,y)\, x^p y^q \, dx \, dy \qquad (1)$$

of the P_1 input pattern $f(x, y)$ as detailed in [7].

These optically-generated image features are used as inputs to a digital feature-based classifier which then determines the object class and the orientation, scale and aspect of the input object. The details of this classifier are provided elsewhere [8] and are not germaine to our present discussion, however several remarks on the classifier follow for completeness. The optically-calculated input moment vector $\bar{m}$ is projected by the first-level classifier in the digital section onto a multi-dimensional Fisher feature space [9]. From the location of the projection vector, initial estimates of the input object class are made. From the ratio of the normalized second-order moments μ_{20} and μ_{02}, an estimate of the aspect ratio or aspect angle of the input object is made. These estimates are used to select reference vectors $\bar{m}_i(\theta)$ for class i and aspect θ from storage against which $\bar{m}$ is compared. The final decision on the object class and the geometrical location of the input object is made in a second-level classifier implementing a nonlinear least-squares solution as detailed in [8]. Our present concern is the preprocessing required on real images before their moments $\bar{m}$ can be reliably extracted.

0 030-4018/84/$03.00

3. Database

As our reference database we used 180 images of five types of ships with 36 images available per ship (at 10° intervals around each ship, from a 90° depression angle). This reference database was obtained from ship models under controlled conditions. Each image contains 128 X 32 pixels with about 2000 pixels corresponding to the ship (for the broadside view) and less than 200 ship pixels (for the bow and stern views). The moments of 4 images per class (10°, 30°, 50° and 80°, where 0° is the bow view and 90° is the broadside view) constituted our reference $\boldsymbol{m}_i(\theta)$ database. As test data, we used various real images of the class 2 ship (the Leahy). A typical image is shown in fig. 1. It shows the ship in water with a sky and shoreline background. We used 256 X 128 pixel images with 8 bits of gray scale for the real ships in our tests. The horizon (separating the water and the sky background) is seen and the depression viewing angle for the real images is 80° (rather than 90°, as in the reference imagery). The real image (from bottom to top) contains four regions: (1) water, (2) the hull of the ship and some water, (3) the superstructure of the ship with a water background, and (4) the sky and shoreline at the top of the image. In section 4, we detail the preprocessing used to extract the ship from the background and in section 5, we discuss the classification performance obtained on such imagery.

4. Image preprocessing

Feature-extraction pattern recognition algorithms require that one object location within the input field-of-view be extracted before the features are computed. These operations are most commonly referred to as segmentation and also involve noise removal and filling in of holes on the object [10]. Care should be taken to employ only simple image preprocessing operations that are not computationally expensive. Thus, we used mainly histogram operations (since they require only simple tallies of image pixel levels) to aid in threshold selections. A wealth of such methods exist, but their specific implementations are quite problem-dependent. In our case, we used context information (the water is below the ship, the sky is above the ship and the deck line and horizon are nearly horizontal due to the sensor system used) to greatly simplify the ship segmentation. Our approach is quite novel in the techniques employed to select separate thresholds for the different image regions and dynamically select these regions based on the scene information. Such methods are of use in feature extractors for diverse applications.

Fig. 1. Typical ship test image (the guided-missile cruiser, the Leahy, ship class 2).

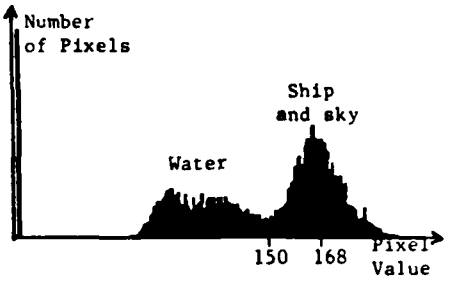


Fig. 2. Bimodal gray-level histogram of fig. 1.

As step 1, we formed the gray-level histogram of fig. 1 (see fig. 2). It was bimodal as expected extending from 0 to 255 (8 bits). A broad peak exists at low pixel values (corresponding to the water and noise, which is low in intensity in fig. 1) and a sharper peak is centered at the high 175 pixel level (corresponding to the ship and the sky, whose pixel values are larger in fig. 1). A well-defined valley at pixel level 150 exists. Thus, at step 2, we thresholded the image at 150 (with all pixel values below 150 set to zero and all pixel values above 150 set to one). The resultant binary image is shown in fig. 3.

At step 3, the image in fig. 3 is used to estimate the location of the four image regions defined in section 3. To achieve this, a horizontal or row-projection histogram of fig. 3 is formed. This is a graph (fig. 3) of

Fig. 3. Binary version of fig. 1 thresholded from the bimodal gray-level histogram of fig. 2.

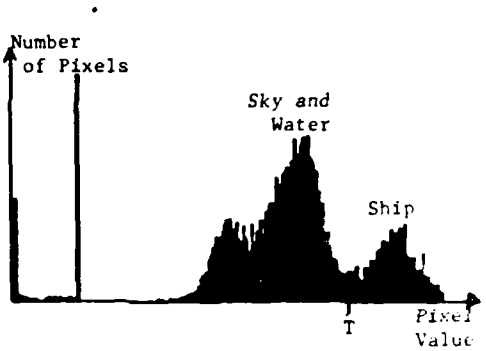


Fig. 5. Gray-level histogram of the gray-scale image in fig. 1 after subtraction of the means of the sky and water from the appropriate image rows.

the number of pixels with value equal to one in each row of fig. 3. From fig. 4, the different image regions can be identified. The region to the right of row C (with zero-valued pixels) is the water below the ship. The flatter region just to the left of row C is the hull. The region between row B and where the hull occurs contains the ship's superstructure (plus water background). The sky and shoreline lie in the region to the left of row A. Between rows A and B is a transition region between the sky and water which contains the horizon region with some sky, water and ship superstructure. Row A and C are easily defined and located. Row B was located from the sum of first differences for consecutive row values as the inflection point in the histogram. These procedures are all automated and require only simple computations.

In step 4, the values for rows A, B and C from fig. 4 are used to extract the sky only (top row to row A) and water only (row A to the bottom row) region of the *original gray-scale image*. Specifically, the average pixel values in these two image regions are calculated. This involves only a simple sum of the pixel levels in the proper rows of fig. 1. In step 5, the mean-value of the sky and shoreline region is subtracted from the rows above A in fig. 1, the mean value of the water region is subtracted from the rows below C in fig. 1, and a linear combination of the mean of the water and sky is subtracted from the rows between A and B. This produces an image with the ship pixels on a positive bias and with the water and sky regions on a zero bias. In step 6, the gray-level histogram of this image is formed. As shown in fig. 5, it has an obvious bimodal structure with a very apparent threshold level or valley point at pixel value V_T.

In step 7, all pixels in the image with gray-level values below V_T in fig. 5 are set to zero. This removes the

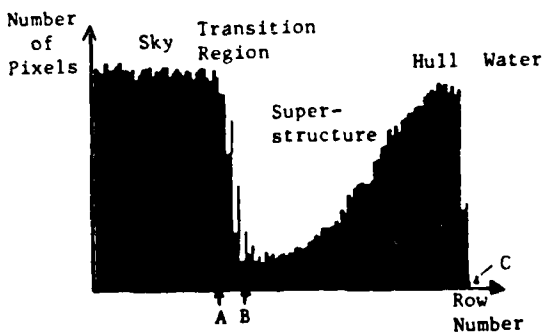


Fig. 4. Horizontal projection histogram of the binary image of fig. 3. The sky, ship, superstructure and water regions of the image are noted.

Fig. 6. Segmented ship image produced using the threshold level V_T found from fig. 5.

sky, shoreline and water and thus extracts the ship. If the gray-levels above V_T are retained, a gray-scale segmented ship image results. If levels above V_T are set to unity, a binary segmented ship image results (fig. 6). Simple median filtering or other local convolution operations can be used to suppress miscellaneous noise pixels remaining in the background and to fill in holes on the target object.

5. Image classification

The moments $\hat{m}$ of the image in fig. 6 were computed and fed to our digital first-level Fisher projection class estimator. This first-level classifier omitted class 1 and 3 ships as possible class matches. The second-level classifier returned class 2 as the most-likely object class. This classifier also provides confidence levels for each possible ship class (classes 2, 4 and 5) passed by the first-level classifier. The class 4 ship, another guided-missile cruiser, had the second-best confidence but it was quite worse than that of the best (and correct) class 2 match. The correct aspect angle (70°) and scale (50%) of the input object are also provided by the classifier.

6. Summary and conclusion

A necessary aspect of feature extractors for pattern recognition is the image preprocessing required. A novel digital segmentation preprocessing procedure of quite general use was detailed for a ship pattern recognition scenario. Such operations are essential if optical or digital feature extraction processors are to achieve good performance. The successful classification of a real input image using moment features and a unique two-level classifier was demonstrated. Similar results were obtained for other real images.

Acknowledgement

The support of this research by the Air Force Office of Scientific Research (Grant AFOSR-79-0091 and F49620-83-C-0010) is gratefully acknowledged.

References

[1] G.G. Lendaris and G.L. Stanley, Proc. Inst. Electrical and Electronic Engrs. 58 (1979) 198.
[2] H. Kasdan and D. Mead, Proc. Elec. Opt. Sys. Des. Conf. 248 (1976).
[3] D. Casasent and V. Sharma, Opt. Eng. 23 (1984).
[4] D. Casasent and W-T. Chang, Appl. Opt. 22 (1983) 2087.
[5] D.G. Nichols, Optics Comm. 43 (1982) 168.
[6] J.A. Blodgett, R.A. Athale, C.L. Giles and H.H. Szu, Optics Lett. 7 (1982) 7.
[7] L. Cheatham, D. Casasent and D. Fetterly, Appl. Opt. 21 (1982) 3292.
[8] D. Casasent, J. Pauly and D. Fetterly, Proc. Soc. Photo-Opt. Instr. Engrs. 302 (1981) 126.
[9] R. Duda and P. Hart, Pattern analysis and scene classification (John Wiley and Sons, New York, 1973).
[10] W.K. Pratt, Digital image processing (John Wiley and Sons, New York, 1978).

8. UNIFIED SYNTHETIC DISCRIMINANT FUNCTION COMPUTATIONAL FORMULATION

Unified synthetic discriminant function computational formulation

David Casasent

A most attractive approach to distortion-invariant pattern recognition uses a synthetic discriminant function (SDF) as the matched spatial filter in a correlator. In this paper, we (1) provide a general basis function and hyperspace description of SDFs, (2) advance a derivation showing the generality of the correlation matrix observation space that we use in our filter synthesis, and (3) detail a unified SDF filter synthesis technique for five different types of pattern recognition problem.

I. Introduction

Correlators are well known to be powerful systems and architectures that can recognize multiple occurrences of an object in the presence of noise. Optical systems using holographic matched spatial filters (MSFs)[1] easily perform the correlation function. However, the performance of a correlator rapidly degrades as distortions are present in the input image. Various approaches have been advanced in recent years to achieve distortion-invariant pattern recognition. None has yet demonstrated such performance while retaining the shift-invariant feature of a correlator. If shift-invariance is not required, a correlation approach is still preferable to feature extraction techniques for distortion-invariant pattern recognition because the processing gain (PG) of a correlator allows more input noise to be present.

In this paper, we detail a generalized method to achieve multiobject shift-invariant and distortion-invariant pattern recognition using a correlator. This technique uses a synthetic discriminant function (SDF) to form the MSF for use in a correlator. The SDF synthesis technique achieves the distortion-invariance, whereas the use of an MSF correlator provides the PG and shift-invariance. This SDF is similar to averaged filters[2,3] and generalized matched filters.[4] However, the filter synthesis and computational technique we use[5] are most general. In Sec. II, we discuss this pattern recognition problem and SDF synthesis using a modified hyperspace description. In Sec. III, we describe our filter synthesis in terms of general 2-D basis functions, and we show that a correlation matrix observation space results directly and yields a SDF synthesis technique. In Sec. IV, we detail the synthesis of five different types of SDF for different pattern recognition applications (using our general filter synthesis description). As we show, all SDFs can be derived from the same basic matrix–vector equation.

We restrict attention to the use of a conventional correlator (modified to use an MSF of an SDF). In such an architecture, the positions of the output correlation peaks denote the locations of the objects in the input field of view. This differs from coded-phase processors[6,7] in which the location of the output peak determines the class of the input object. Such processors are not shift-invariant and require that only one object be present in the input field of view. The SDF concept we advance can be viewed as an extension and reformulation of the use of correlators with multiple MSFs (one per object class) and multiple correlation outputs. As noted in Ref. 8, one can obtain better performance from a multichannel correlator by forming a linear combination of the multiple correlation outputs (compared to the performance that results if we simply select the single correlation output with the largest peak value). Our filter synthesis technique forms one MSF that is a linear combination of the MSFs of the different object classes being considered. However, we form this filter in the image plane and then by a conjugate Fourier transform construct the MSF. This approach might appear to differ only slightly from others. However, as we show (Secs. II and III), it is much more general (since synthesis of a SDF directly in the MSF Fourier transform plane restricts the basis functions used to be Fourier coefficients or exponentials etc.), and it is also much easier to compute (as we show in Sec. III).

The author is with Carnegie-Mellon University, Department of Electrical & Computer Engineering, Pittsburgh, Pennsylvania 15213.

Received 21 November 1983.
0003-6935/84/101620-08$02.00/0.

II. Hyperspace SDF formulation

A hyperspace description of an SDF is the most conventional pattern recognition approach.[9] This approach to SDF descriptions was first advanced in Ref. 10. Conventional pattern recognition uses a feature vector representation in which each input image is described by its projections on different scalar image features.[9] Each input image is then described by a feature vector in a hyperspace (whose axes are the scalar image features considered to be of importance). In Fig. 1, we show a typical representation of how one might desire two classes of data to be displayed in a simplified two-axis hyperspace. Objects in the two classes (represented by Xs and Os, respectively, in Fig. 1) should be widely separated, and objects within each class should cluster in a small region in this display space. In conventional pattern recognition, the basis functions or object features (the axes of the hyperspace) are usually scalar features. In our description, we consider the use of a hyperspace with 2-D spatial basis functions as the axes or object features. This will clearly greatly increase the power of such a pattern recognition representation. The major problem is the selection of the object features (the axes of the hyperspace or equivalently the basis functions) to achieve the desired separation of different classes and the clustering of data within each class. Most techniques that have been suggested to achieve this are rather *ad hoc*. However, our approach is automatic (as will be shown).

We thus consider an advanced and modified hyperspace in which the basis functions are 2-D spatial image functions rather than scalars. We retain the same basic concepts used in conventional hyperspace feature-space pattern recognition. For example, if a line or a hyperplane (shown in Fig. 1) can be drawn that separates the two image classes, the normal to this plane from the origin defines the discriminant function to be used. In conventional pattern recognition with scalar basis functions, an input object is described by its features, and these comprise the elements of the feature vector that describes the input object. When this feature vector is projected onto the discriminant feature vector, a decision on the class of the input object is made (based upon the value of the projection).

In our modified hyperspace formulation, we retain the major element of conventional pattern recognition, the concept of basis functions, discriminant functions, etc. However, in our formulation, each of these now becomes a 2-D spatial function. Since the basis functions are 2-D, so is the discriminant function and so is the input image in our representation. We can simply project the 2-D input image onto the SDF as in the conventional case of image and feature vectors. However, the result will be valid for only one location of the object in the input field of view. To see this, recall that our basis functions are 2-D spatial functions; thus each shifted version of an object corresponds to a new point in our hyperspace. All these points (for one object) lie on the surface of a hypersphere (since shifted versions of an object have the same energy). Clearly, a conventional hyperspace description becomes very complicated if shifted versions of the input object are required to be recognized. Thus, in our modified hyperspace description, we retain the simplicity of a single vector representation of an object and the definition of the discriminant function as the normal from the origin to the discriminant hypersurface separating regions containing different object classes. To provide shift-invariance, we correlate (a 2-D spatial correlation) the discriminant function with the input image and use the hyperspace concept only to synthesize the discriminant function to be used. Since any shifted version of an object can be used to synthesize a MSF in a correlator, we need select only one shifted version of each object class in our hyperspace representation and for our discriminant function synthesis. We select the specific shifted version used for each object class based upon maximum common information concepts as we detailed in Ref. 11. A maximum common information SDF then results.

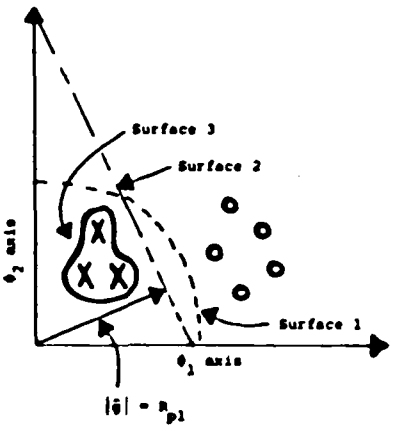


Fig. 1. Simplified two-axis hyperspace description of distortion-invariant multiclass shift-invariant pattern recognition using a feature vector and discriminant vector hyperspace concept.

The selection of the specific shifted version of each object class to be used can often be simply achieved by colocating the centroids of all the objects. In specific cases, small shifts from the centroid-centered images are needed if optimum performance is to be achieved. In general, sufficient performance results from the use of centroid-shifted objects alone.[11] Since this and other pattern recognition techniques employ training sets of data for the different object classes, such flexibility in the selection of the images used in the hyperspace description is quite valid and appropriate. The general approach is described in the simple system diagram of Fig. 2. We use several different images of each object class for filter synthesis and to perform the hyperspace diagram. These images can and usually are different geometrically distorted views of each object class. These are referred to as the image training set. They are used to determine the basis function to be used, to select the discriminant hypersurface, and hence to define the discriminant function itself. The training set of images is chosen to provide a valid statistical repre-

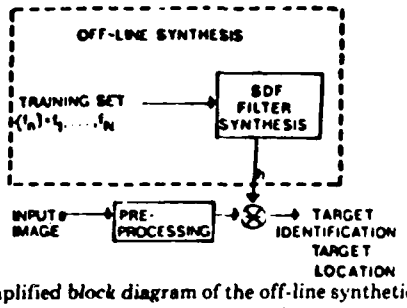


Fig. 2. Simplified block diagram of the off-line synthetic discriminant function synthesis from training set data and the use of such filters for on-line correlation of the SDF with unknown input imagery.

sentation of each object class. The SDF algorithm itself provides the discrimination (as we will detail in Sec. III).

This entire filter synthesis operation is performed off-line on training set images. From such computations, an SDF $h(x,y)$ is produced, which is then correlated with new input imagery in a MSF correlator as shown in Fig. 2. This new input imagery is referred to as test imagery. These test images are not members of the training set of images. For generality, a preprocessing box is included in Fig. 2. This can perform edge enhancement, median filtering, or similar operations. In general, this preprocessing function can be omitted or restricted to quite simple operations (because of the processing gain of a correlator).

III. Correlation Observation Space

In this section, we discuss an automated technique to select the basis functions and the SDF described in Sec. II. The general SDF formulation we employ uses a correlation observation space. To justify the generality of this technique, we devote this section to a derivation of it as the most general set of features to be used in synthesizing and computing a SDF MSF for use in a correlator. Our formulation uses a general set of basis functions and involves an automated technique to select them. To develop our general SDF synthesis technique, we consider N training set images of an object in class one. These N images can represent different distorted versions of this one object. For simplicity, we consider the synthesis of an equal correlation peak (ECP) SDF. This filter function $h(x,y)$ has the property that the correlation output of $h(x,y)$ and all images $\{f_n(x,y)\}$ in class one equals a constant (unity is chosen for this constant), i.e.,

$$f_n(x,y) \circledcirc h(x,y) = 1. \tag{1}$$

In Sec. IV, we extend the basic algorithm in this section to other pattern ern recognition applications and other types of SDF. When the different images $\{f_m(x,y)\}$ are different geometrically distorted versions of one object $f(x,y)$, this ECP SDF is appropriate for an intraclass pattern recognition problem (recognition of any distorted version of an object using a single filter function).

To develop formally an algorithm for synthesis of a filter function $h(x,y)$ that satisfies Eq. (1), we describe each training set image as a linear combination of a basis function set $\phi_m(x,y)$, i.e.,

$$f_n(x,y) = \sum_m a_{nm}\phi_m(x,y) \tag{2}$$

This follows directly from our hyperspace description in Sec. II. We place no specific restrictions on the basis function set; i.e., we do not assume a Fourier coefficient basis function set as in Ref. 4 or the use of circular harmonics as in Ref. 12. The desired SDF is described as another linear combination of the same basis function set. This is compatible with the conventional hyperspace description in Fig. 1 and Sec. II, extended to the case of 2-D basis functions, i.e.,

$$h(x,y) = \sum_m b_m\phi_m(x,y). \tag{3}$$

Assuming an orthonormal set of basis functions (as is conventional in pattern recognition), we can substitute Eq. (2) into Eq. (1) and rewrite the ECP SDF condition in Eq. (1) as

$$f_n(x,y) \circledcirc h(x,y) = \mathbf{f}_n \cdot \mathbf{h} = \sum_n a_{nm}b_m = 1. \tag{4}$$

Next we note that since $h(x,y)$ is a linear combination of the $\phi_m(x,y)$ and so is $f_n(x,y)$, we can write $h(x,y)$ as a linear combination of the input training set of images $\{f_n(x,y)\}$; i.e., we first write out several of the terms in Eq. (3):

$$h(x,y) = b_1\phi_1(x,y) + b_2\phi_2(x,y) + \ldots = \sum_m b_m\phi_m(x,y). \tag{5}$$

From Eq. (2), we can write the basis function set $\phi_m(x,y)$ as a linear combination of the training set of images $f_n(x,y)$ as

$$\phi_m(x,y) = \sum_n d_{mn}f_n(x,y). \tag{6}$$

Substituting Eq. (6) into Eq. (3), we obtain

$$h(x,y) = b_1\sum_n d_{1n}f_n(x,y) + b_2\sum d_{2n}f_n(x,y) + \ldots \tag{7}$$

$$= e_1f_2(x,y) + e_2f_2(x,y) + \ldots \tag{7a}$$

$$= \sum_m e_mf_m(x,y). \tag{7b}$$

In Eq. (7a), we have grouped all coefficients of f_1, f_2, etc. together and have denoted them by e_1, e_2, etc. The final result in Eqs. (7b) and (3) are equivalent; one describes the SDF in terms of the basis functions [Eq. (3)], and the other [Eq. (7b)] describes them in terms of the original training set of images.

We now consider how to determine the e_m in Eq. (7b) to satisfy our ECP SDF criteria in Eq. (1) or (4). For notational simplicity, we describe all images [the SDF $h(x,y)$ and the training set images] by vectors $\mathbf{h}$ and $\mathbf{f}_n$ or $\mathbf{f}_m$, respectively. This notation and description follow directly from the hyperspace model advanced in Sec. II. We denote the correlation of two such vector functions by the vector inner product, which we write simply as $\mathbf{f}_n \cdot \mathbf{h}$. Since we use a correlator for our final object classification, and since any shifted version of an

image can be used as the MSF in a correlator, there is no loss of generality in this simplified formulation.

With these preliminaries, the ECP SDF requirement in Eq. (1) is now written as

$$\mathbf{f}_n \cdot \mathbf{h} = 1. \tag{8}$$

Substituting Eq. (7b) into Eq. (8) for **h**, rearranging terms, and defining r_{mn} as the elements of the correlation matrix **R**, Eq. (8) becomes

$$\mathbf{f}_n \cdot \mathbf{h} = \mathbf{f}_n \cdot \left[\sum_m c_m \mathbf{f}_m\right] = \sum_m c_m (\mathbf{f}_n \cdot \mathbf{f}_m) = \sum_m c_m r_{nm} = 1. \tag{9}$$

In matrix-vector form, we rewrite Eq. (9) as

$$\mathbf{Rc} = \hat{u}, \tag{10}$$

where $\hat{u}$ denotes the unit vector and where the elements of the vector **c** are the c_m in Eq. (7b) or (9). The solution for the ECP SDF $h(x,y)$ defined by Eq. (7b) that satisfies (1) is thus given by the solution to Eq. (10), i.e.,

$$\mathbf{c} = \mathbf{R}^{-1}\hat{u} \tag{11}$$

From this general formulation, we have shown that a correlation matrix observation space directly results as an ideal feature space from which to compute the required coefficients for a linear combination filter such as an SDF. We note that this formulation used a general basis function set $\phi_m(x,y)$, but that in our algorithm no specific choice for the basis function set was required. Thus, to synthesize an SDF, we simply form the correlation matrix of the training set of data, invert it, and multiply it by the appropriate vector $\hat{u}$. This discriminant function formulation is thus automatic and does not require *ad hoc* selection of certain basis functions or input features. We first advanced the fundamentals of this unified correlation matrix observation space description in Ref. 5. This present description is a revised and more general version of our original algorithm for synthesis of an averaged filter[2,3] with the removal of any specific requirements or selection techniques for the basis functions used. In Sec. IV, we develop a general formulation along the general description in Eq. (11) for the synthesis of several different SDFs for different pattern recognition applications.

Many techniques exist by which a general basis function set can be obtained. In Refs. 2, 3, 10, and 11, we used a Gram-Schmidt procedure to select orthogonal basis functions. In Refs. 6 and 7 a Fukunaga-Koontz and Foley-Sammon technique is employed. In Refs. 10 and 13, Karhunen-Loeve transforms were suggested for similar problems. In Ref. 14, singular value decomposition techniques were described. Our present algorithm can accommodate any of these methods, but by our new generalized description, we require no specific basis function selection. However, these prior techniques are useful as intermediate steps in performing the required correlation matrix inversion in Eq. (11). No specific guidelines for matrix inversion techniques are advanced in this present paper, since we desire to retain a general description. However, we note that if **R** is singular, we employ a generalized inverse, and if the dimension of **R** is large, we use several new computationally efficient methods such as on-line dominant-image calculation[15] and orthogonal-hyperplane-projection methods.[16]

IV. Generalized SDF Synthesis

In this section, we describe five general SDFs and detail their synthesis in the form of Eq. (11) and their use for different pattern recognition problems and applications. The computational ease with which these useful pattern recognition filters can be fabricated is quite attractive. The filters to be described include a more unified and general description (Sec. IV.A) of the ECP SDF of Sec. III (for intraclass pattern recognition), a mutual orthogonal function (MOF) SDF (Sec. IV.B) for M-class interclass pattern recognition, an MOF SDF for two-class and multiclass interclass discrimination as well as intraclass recognition (Sec. IV.C), a new multiclass MOF SDF using several SDFs (Sec. IV.D), and another new simple nonredundant filter (NRF) SDF for intraclass and interclass pattern recognition (Sec. IV.E).

A. Equal Correlation Peak SDFs for Intraclass Pattern Recognition

The general formulation for the ECP SDF satisfying the condition in Eq. (1) can be described (for a training set of N_1 images and an associated $N_1 \times N_1$ correlation matrix $\mathbf{R}_1$) as

$$\mathbf{a} = \mathbf{R}_1^{-1}\hat{u}_1 = \mathbf{R}_1^{-1}[1, \ldots, 1]^T, \tag{12}$$

where the unit vector $\hat{u}_1$ has N_1 elements (all of which are unity). The elements of **a** are the weighting coefficients in the linear combination SDF:

$$h(x,y) = \sum_n a_n f_n(x,y), \tag{13}$$

where the $\{f_n(x,y)\}$ training set images associated with the correlation matrix **R** are different distorted versions of the same object. This SDF filter $h(x,y)$ when used in a correlator is thus capable of intraclass distortion-invariant pattern recognition (i.e., recognition of different distorted views of one class of object). Such an ECP SDF yields the same correlation output for all distorted views of one object as required by Eq. (1). In other extensions of this general SDF synthesis algorithm to other pattern recognition applications (beyond intraclass object recognition), we begin by describing the SDF as a linear combination of the training set of images. As shown in Sec. III, such a formulation emerges directly from our hyperspace description.

B. Mutual Orthogonal Function SDF for Interclass Pattern Recognition

Next we consider an interclass pattern recognition problem (the discrimination between and recognition of M different objects in M different classes). In this initial example, we assume one image per object class, and we consider only interclass discrimination rather than intraclass recognition of distorted versions of each object. We describe our training set of M images (one

per object class) by $\{f_m(x,y)\}$, and we denote the $M \times M$ correlation matrix of this training set of data by $\mathbf{R}_2$. For this problem, we desire to produce M SDFs $h_m = h_1, h_2, \ldots$, so that

$$f_j(x,y) \circledcirc h_i(x,y) = \delta_{ij}, \tag{14}$$

i.e., $f_j(x,y) \circledcirc h_i(x,y)$ is unity only for filter i and image class $j = i$. Thus only one of the M SDFs $\{h_m(x,y)\}$ yields an output of unity, whereas all the $M - 1$ other SDFs yield zero outputs. The filter with the unity output thus determines the class of the input object.

Following our general procedure in Secs. III and IV.A, we describe each of these M SDFs as a different weighted linear combination of all M training set images $f_m(x,y)$, i.e.,

$$h_1(x,y) = \sum_m a_m f_m(x,y), \quad h_2(x,y) = \sum_m b_m f_m(x,y), \ldots,$$

$$h_M(x,y) = \sum_m M_m f_m(x,y), \tag{15}$$

or in general

$$h_p(x,y) = \sum_m p_m f_m(x,y), \tag{16}$$

where each summation in Eqs. (15) and (16) is over all M training set images. Following our earlier general approach, we can write the coefficients **a**, **b**, etc. in Eq. (15) that satisfy Eq. (14) as

$$\mathbf{a}\mathbf{R}_2 = \hat{u}_a = [1,0,0,0,\ldots,0]^T, \tag{17a}$$

$$\mathbf{b}\mathbf{R}_2 = \hat{u}_b = [0,1,0,0,\ldots,0]^T, \tag{17b}$$

$$\mathbf{c}\mathbf{R}_2 = \hat{u}_c = [0,0,1,0,\ldots,0]^T, \text{etc.} \tag{17c}$$

Each $\hat{u}_m$ vector in Eqs. (17) has M elements and contains $M - 1$ zeros and one 1. The location of the single 1 is different in each of the vectors. For $\hat{u}_a$, the first element is a 1; for $\hat{u}_b$, the second element is a 1; etc. The elements of the different vectors **a**, **b**, etc. in Eqs. (17) are the linear coefficients in the corresponding SDF equations in (15). Thus the M SDFs in Eq. (15) that satisfy Eq. (14) are described by

$$\mathbf{a} = \mathbf{R}_2^{-1}\hat{u}_a, \quad \mathbf{b} = \mathbf{R}_2^{-1}\hat{u}_b, \quad \mathbf{c} = \mathbf{R}_2^{-1}\hat{u}_c, \text{etc.} \tag{18}$$

As seen by inspection of the $\hat{u}_m$ vectors in Eq. (17), filter h_1 (described by **a**) yields unity output for the image f_1 of class one and zero for all other image classes. Filter h_2 (described by **b**) yields unity output for the image f_2 of class two and zero for all other $M - 1$ image classes, etc. This is as required by Eq. (14).

Since all these filter functions are mutually orthogonal, we refer to this type of SDF as a mutual orthogonal filter (MOF) SDF. The problem formulation advanced above is similar to that of the generalized matched filter as described in Ref. 4 and the decorrelation matrix filter synthesis described in Ref. 10. However, our present formulation is in terms of our general description using the correlation matrix of the training set of data.

In Ref. 4, the filter function was synthesized from Fourier coefficients of each object and is thus the frequency-domain filter synthesis dual of the earlier[8] use of a linear combination of multiple correlation-plane outputs. Instead of forming a linear combination of M multiple correlation-plane outputs (each correlation using an MSF of one class of object), the M MSFs are synthesized so that only one of the M correlation-plane outputs yield a peak value near the maximum. Our algorithm for synthesis of such filters in Eqs. (18) describes such generalized matched filter synthesis using the original images rather than the Fourier coefficients of each image. In Ref. 4, only interclass discrimination was discussed (rather than intraclass recognition of distorted views of an object). In Sec. IV.C, we will extend this MOF SDF to include both intraclass and interclass object recognition.

In Ref. 10, a Gram-Schmidt basis function selection technique was used to assemble a Gram-Schmidt correlation matrix. In this approach, the first basis function contains information only associated with the first image f_1; the second basis function contains only the new image information present in the second image f_2 that is not also present in the first image f_1; the third basis function contains the new information present in f_3 but not previously included in f_1 and f_2; etc. If the first row and column of this Gram-Schmidt matrix is set equal to zero (in Ref. 10, this was achieved by multiplying this matrix by a decorrelation matrix), all training set image information associated with f_1 is removed, and the correlation of the filter synthesized from this reduced matrix will yield zero output when correlated with f_1. Extensions of this technique to the decorrelation of the training set of data for the other object classes follow directly.

Our present formulation in Eqs. (17) and (18) is much simpler and more easily implemented, and it is cast in the same general matrix–vector form as that of Eqs. (11) and (12). We refer to the SDF in Eqs. (17) and (18) as an interclass MOF SDF or simply as an MOF SDF.

C. MOF SDF Synthesis for Intraclass and Interclass Pattern Recognition

We now combine our ECP intraclass SDF (Sec. IV.A) and our MOF interclass SDF (Sec. IV.B) formulations to describe the synthesis of an MOF SDF for both intraclass and interclass pattern recognition. We describe the algorithm for a three-class problem, for a two-class problem, and then we generalize to the case of an M-class problem. This type of SDF is appropriate for pattern recognition applications in which the input object can be a member of several classes and when different distorted versions of the input object can be expected. In such a case, we must insure interclass discrimination between objects of different classes and intraclass recognition of distorted versions of one object as members of the same object class. We consider a three-class pattern recognition problem with N_1 images of one object $\{f_{a_i}(x,y)\}$ of class a, N_2 images of the class b object $\{f_{b_i}(x,y)\}$, and N_3 images of a class c object $\{f_{c_i}(x,y)\}$ used as the training set. As before, each of these training sets of objects consists of different distorted versions of one object.

We desire to synthesize three filter functions $h_a(x,y)$, $h_b(x,y)$, and $h_c(x,y)$ that satisfy

$$f_{mi}(x,y) \circledcirc h_m(x,y) = \delta_{nm}, \tag{19}$$

for all members i of each separate object class $m = n$. We describe the SDFs for our three-class problem as linear combinations of the entire training set of data $\{f_n(x,y)\} = \{f_{ai}(x,y), f_{bi}(x,y), f_{ci}(x,y)\}$, i.e.,

$$h_a(x,y) = \sum_n a_n f_n(x,y), \quad h_b(x,y) = \sum_n b_n f_n(x,y), \quad h_c(x,y) = \sum_n c_n f_n(x,y), \tag{20}$$

where all summations in (20) are over the entire $N = N_1 + N_2 + N_3$ training set of images. By directly extending our results in Secs. IV.A and IV.B, we form the full $(N_1 + N_2 + N_3) \times (N_1 + N_2 + N_3)$ correlation matrix $\mathbf{R}_3$. By analogy with Eq. (17), we can then describe the three SDFs in Eqs. (20) subject to the conditions in Eq. (19) by the vectors $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ (each of dimensionality $N_1 + N_2 + N_3$) as

$$\mathbf{a}\mathbf{R}_3 = \hat{u}_a = [1, \ldots, 1; 0, \ldots, 0; 0, \ldots, 0]^T, \tag{21a}$$

$$\mathbf{b}\mathbf{R}_3 = \hat{u}_b = [0, \ldots, 0; 1, \ldots, 1; 0, \ldots, 0]^T, \tag{21b}$$

$$\mathbf{c}\mathbf{R}_3 = \hat{u}_c = [0, \ldots, 0; 0, \ldots, 0; 1, \ldots, 1]^T \tag{21c}$$

In Eqs. (21), $\hat{u}_a$, $\hat{u}_b$, and $\hat{u}_c$ each contain $N_1 + N_2 + N_3$ elements with only the first N_1, the central N_2 or the last N_3 elements being 1, respectively, and with all other elements being zero. The matrix–vector constraints in Eqs. (21) thus correspond to those in Eq. (19), i.e., $\mathbf{a}$, and hence $h_a(x,y)$ is required to have a unity correlation output for the N_1 objects in class a and zero for the other training set images. Similar remarks follow for $\mathbf{b}$ and $\mathbf{c}$ and equivalently for the associated filters $h_b(x,y)$ and $h_c(x,y)$.

The three SDFs in Eqs. (20) are thus defined by

$$\mathbf{a} = \mathbf{R}_3^{-1}\hat{u}_a, \quad \mathbf{b} = \mathbf{R}_3^{-1}\hat{u}_b, \quad \mathbf{c} = \mathbf{R}_3^{-1}\hat{u}_c \tag{22}$$

analogously to Eqs. (18). They thus satisfy a three-class intraclass and interclass pattern recognition problem. The extension to M-classes with N training set images per class results in an increase in the size of the correlation matrix (to $MN \times MN$) and an increase in the dimensionality of the coefficient vectors to MN. We refer to such filters as intraclass and interclass MOF SDFs or simply as MOF SDFs.

For smaller problems such as two-class pattern recognition applications requiring intraclass distortion-invariance, considerable simplifications are possible.[5] We can use a single SDF $h(x,y)$ described by

$$\mathbf{a} = \mathbf{R}_4^{-1}\hat{u}_a, \tag{23}$$

where $\hat{u}_a = [1, \ldots, 1; 0, \ldots, 0]^T$ contains N ones and N zeros and where $\mathbf{R}_4$ is the two-class correlation matrix of dimensionality $N_1 + N_2$. If the correlation peak output is above (or below) an 0.5 threshold level (halfway between the two required zero and one output levels), we choose class one (or class two) for the object class. Since a zero correlation output or a correlation output below 0.5 can also result when no input object is present, this approach in Eq. (23) is appropriate only in the restricted applications. We can modify Eq. (23) using $\mathbf{u}_a = [1, \ldots, 1; -1, \ldots, -1]^T$. In this case, if the correlation peak value is above 0.5 (or below −0.5), we select class one (or class two) as the object class. Such an approach is attractive for digital correlators but not for conventional optical correlators using intensity detectors. (Such correlators provide unipolar correlation outputs only.) For now, we only note that for two-class intraclass pattern recognition problems, a single SDF in general suffices if two different correlation plane detection threshold levels are used with the class of the input object determined from the correlation peak value. Such multilevel nonredundant filter (NRF) SDFs are discussed further in Sec. IV.D.

D. Multilevel NRF SDFs

We now generalize our three-class intraclass and interclass pattern recognition example in Sec. IV.C to the use of a single SDF. This SDF $h(x,y)$ is required to give outputs of 1, 2, and 3 for objects in classes one, two, and three, respectively.[5] (Other appropriate constants can be selected.) We refer to such a filter as a multilevel nonredundant filter (NRF) SDF. The filter requirement is described by

$$f_n(x,y) \circledcirc h(x,y) = n, \tag{24}$$

where the correlation output $n = 1$ for objects $\{f_1(x,y)\}$ in class one, $n = 2$ for objects $\{f_2(x,y)\}$ in class two, etc. With N_1, N_2, and N_3 training set images for the three classes, respectively, $h(x,y)$ is described by

$$h(x,y) = \sum_m a_m f_m(x,y), \tag{25}$$

where $f_m(x,y) = \{f_{1i}(x,y), f_{2i}(x,y), f_{3i}(x,y)\}$ contains all $N_1 + N_2 + N_3$ training set images and where the summation in Eq. (25) is over $N_1 + N_2 + N_3$. The vector $\mathbf{a}$ that describes the $h(x,y)$ that satisfies Eq. (24) is

$$\mathbf{a} = \mathbf{R}_3^{-1}\hat{u}_3, \tag{26}$$

where $\mathbf{R}_3$ is the $(N_1 + N_2 + N_3) \times (N_1 + N_2 + N_3)$ correlation matrix and where $\hat{u}_3 = [1, \ldots, 1; 2, \ldots, 2; 3, \ldots, 3]^T$ contains N_1 ones, N_2, twos, and N_3 threes. Extensions of this multilevel NRF SDF to more classes are straightforward. However, more stringent detector plane requirements and reduced performance can be expected as the number of restrictions placed on such a single SDF filter are increased. Thus, if this technique is to be used for more than three or four classes of data, more advanced preprocessing and image training set selection techniques should be considered.[17] Such issues will be addressed in subsequent journal papers.

E. *K*-tuple NRF SDFs

As implied in our discussion in Sec. IV.D, a single SDF for multiclass intraclass and interclass recognition is possible conceptually but may not yield acceptable performance when noise and other issues are included. The specific nature of the distortions to be considered and the nature of the different object classes to be dis-

tinguished plus the type and amount of noise to be expected will determine the performance obtained. If a large number of object classes must be considered, the conventional MOF SDFs (Secs. IV.B and IV.C) would require the use of M SDFs (for an M-class problem) and the scanning of M 2-D output correlation planes. For such pattern recognition problems, an alternate technique we refer to as a K-tuple NRF SDF technique may be more appropriate.

In this SDF algorithm, we consider an M-class pattern recognition problem and the use of only K SDFs, where K is chosen to satisfy $M \le 2^K$. In this case, we denote the K correlation outputs at the same (x,y) spatial location in all K correlation planes by $c_1, c_2, \ldots, c_K$. We consider the case of binary correlation plane threshold levels of 0 or 1 as in Secs. IV.B and IV.C. For each correlation plane point, we thus have a K-tuple binary vector $\mathbf{c}$. Using conventional binary Boolean algorithm encoding, each $\mathbf{c}$ vector can thus represent up to 2^K different numbers. In our present M-class pattern recognition problem, this means that the corresponding $\mathbf{c}$ value can determine to which of $M = 2^K$ classes the input object lies (the object at the associated position in the input plane). We refer to this as the K-tuple binary-level NRF SDF or simply as a K-tuple NRF SDF.

The formal description of such SDFs is best presented only for a $K = 2$ filter SDF case, i.e., a four-class pattern recognition problem. Generalizations beyond this case should follow directly, but formally writing the associated matrix–vector equations results in unneeded notational complexity that will not further advance understanding of the basic concepts. We thus consider only an M = four-class intraclass pattern recognition problem and the use of $K = 2$ SDFs. We describe the training sets of data for these four object classes by $|f_{1i}(x,y)|$, $|f_{2i}(x,y)|$, etc. We assume that there are N_1 training set images in class one, N_2 images in class two, etc. For notational simplicity, we assume $N = N_1 = N_2 = N_3 = N_4$ or $4N$ total training set images. We describe the full $4N$ training set of images by $|f_n(x,y)|$, the associated correlation matrix by $\mathbf{R}$ (it is of dimensionality $4N \times 4N$), and the two SDFs by $h_a(x,y)$ and $h_b(x,y)$.

We require the two correlation outputs of the general input image $f(x,y)$ and the two filters $h_a(x,y)$ and $h_b(x,y)$ to satisfy the truth table in Table I. The four possible combinations of the binary (0 and 1) correlation plane outputs are used to denote in which of the four classes the input object lies. We use correlation plane levels of 0 and 1 with no loss of generality. Although a zero output level can correspond to no object, this is easily altered by selecting any other nonzero coefficient for the desired correlation plane output level in our filter synthesis equation (as noted at the end of Sec. IV.D). We denote the two SDFs by

$$h_a(x,y) = \sum_n a_n f_n(x,y), \qquad h_b(x,y) = \sum_n b_n f_n(x,y), \tag{27}$$

where the summations in Eqs. (27) are over the $4N$ training set images. In matrix–vector form, we write the truth table in Table I as

Table I. Truth Table for a K-tuple Nonredundant SDF. The Case of $M = 4$ Classes and $K = 2$ Filters is Shown.

Object class \ Filter out	$h_a(x,y)$	$h_b(x,y)$
1	0	0
2	0	1
3	1	0
4	1	1

$$|R| \begin{bmatrix} a_1 & b_1 \\ \vdots & \vdots \\ a_{4N} & b_{4N} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \cdots & \\ 0 & 1 \\ \cdots & \\ 1 & 0 \\ \cdots & \\ 1 & 1 \end{bmatrix} \tag{28a}$$

or

$$\mathbf{R}[\mathbf{ab}] = [\hat{u}_1 \hat{u}_2], \tag{28b}$$

where $\mathbf{R}$ is $4N \times 4N$, $\mathbf{a}$ and $\mathbf{b}$ define $\mathbf{h}_a$ and $\mathbf{h}_b$ in Eqs. (27) and where the right-hand side vector in Eqs. (28) consists of two column vectors $\hat{u}_1$ and $\hat{u}_2$ with N element pairs equal to (0,0), N element pairs equal to (0,1), etc. The solution to Eqs. (28) for the vectors $\mathbf{a}$ and $\mathbf{b}$ that define $h_a(x,y)$ and $h_b(x,y)$ in Eqs. (27) to satisfy Table I or Eq. (28) is thus

$$[\mathbf{ab}] = \mathbf{R}^{-1}[\hat{u}_1 \hat{u}_2]. \tag{29}$$

A variation of this K-tuple NRF formulation was first advanced in Ref. 18 for coherent optical systems and then extended to noncoherent optical correlators in Ref. 19. Neither of these formulations used a correlation matrix observation space to describe synthesis of the required filter, however.

V. Summary and Conclusion

From Sec. IV, we have shown and detailed how five different types of synthetic discriminant function for different pattern recognition problems (intraclass recognition, interclass discrimination, and both intraclass and interclass object identification) can be formulated as the same general matrix–vector equation. Specifically, the vectors that describe the SDFs equal the inverse of a correlation matrix R times a control vector (containing ones, zeros, or other similar constants). Inspection of Eqs. (11) and (12), (17) and (18), (21) and (22), (23), (26), and (28) and (29) shows that all expressions for such a filter computation and synthesis are of the same general matrix–vector linear algebraic equation form.

In Sec. III, we provided a general description of a distortion-invariant matched spatial filter and showed for general basis functions that all such MSF pattern recognition problems involve inversion of a correlation matrix and multiplication by a different external vector. The size of and the elements of the correlation matrix and the elements chosen for the external vector differ for specific pattern recognition applications, but the same general format is retained throughout all types of SDF for different applications. In Sec. II, we described the philosophy and details of SDF synthesis in terms of a modified hyperspace and feature-vector system.

This hyperspace description (using 2-D basis functions and 2-D discriminant functions), our derivation of the use of a correlation matrix observation space (independent of the basis functions used), and the general unification and detailed description of five different types of SDF are the original contributions in this paper. Other variations of these concepts (such as combinations of multilevel K-tuple nonredundant filter SDFs) are obvious but were not detailed. Initial experiments[20] have showed that the SDFs described herein give most excellent performance even in the performance of noise.[17] These results and the issue of training set selection and the theoretical basis for the performance found using such filters will be the subject of future journal papers. Our intent in this paper was to provide the initial details and foundations of a unified SDF filter synthesis technique for multiclass distortion-invariant shift-invariant pattern recognition.

The support of this research by grants from the Air Force Office of Scientific Research (AFOSR-79-0091) and the Internal Research and Development Funds of General Dynamics Pomona (313614) is gratefully acknowledged as is the original work by Charles Hester on this topic and many fruitful technical discussions with B. V. K. Vijaya Kumar.

References

1. A. VanderLugt, IEEE Trans. Inf. Theory **IT-10**, 139 (1964).
2. C. Hester and D. Casasent, Proc. Soc. Photo-Opt. Instrum. Eng. **201**, 77 (Aug. 1979).
3. C. F. Hester and D. Casasent, Appl. Opt. **19**, 1758 (1980).
4. H. J. Caulfield, Appl. Opt. **19**, 3877 (1980).
5. D. Casasent, V. Sharma, and B. V. K. Vijaya Kumar, Proc. Soc. Photo-Opt. Instrum. Eng. **360**, 136 (Aug. 1982).
6. J. H. Leger and S. H. Lee, Appl. Opt. **21**, 274 (1982).
7. J. H. Leger and S. H. Lee, J. Opt. Soc. Am. **72**, 556 (1982).
8. H. J. Caulfield and W. T. Maloney, Appl. Opt. **8**, 2354 (1969).
9. R. Duda and P. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973); Y. T. Chien, *Interactive Pattern Recognition* (Marcel Dekker, New York, 1978).
10. C. Hester and D. Casasent, Proc. Soc. Photo-Opt. Instrum. Eng. **302**, 108 (1981).
11. C. Hester and D. Casasent, Proc. Soc. Photo-Opt. Instrum. Eng. **292**, 25 (1981).
12. Y-N. Hsu and H. H. Arsenault, Appl. Opt. **21**, 4016 (1982).
13. J. Duvernoy, Opt. Commun. **42**, 386 (1982).
14. J. B. Nelson, Appl. Opt. **20**, 8 (1981).
15. H. Murakami and B. V. K. Vijaya Kumar, IEEE Trans. Pattern. Anal. Machine Intell. **PAMI-00**, 511 (1982).
16. B. V. K. Vijaya Kumar, Appl. Opt. **22**, 1445 (1983).
17. D. Casasent and V. Sharma, Proc. Soc. Photo-Opt. Instrum. Eng. **442**, 47 (Aug. 1983).
18. J. Flenret and H. Maitre, Opt. Commun. **17**, 64 (1976).
19. B. Braunecker, R. Hauck, and A. W. Lohmann, Appl. Opt. **18**, 2746 (1979).
20. D. Casasent and V. Sharma, in *Proceedings, International Optical Computing Conference*, IEEE Cat. CH1880-4/83 (Apr. 1983).

9. ACOUSTO-OPTIC LINEAR ALGEBRA PROCESSORS: ARCHITECTURES, ALGORITHMS AND APPLICATIONS

Acoustooptic Linear Algebra Processors: Architectures, Algorithms, and Applications

DAVID CASASENT, FELLOW, IEEE

Invited Paper

Architectures, algorithms, and applications for systolic processors are described with attention to the realization of parallel algorithms on various optical systolic array processors. Systolic processors for matrices with special structure and matrices of general structure, and the realization of matrix–vector, matrix–matrix, and triple-matrix products and such architectures are described. Parallel algorithms for direct and indirect solutions to systems of linear algebraic equations and their implementation on optical systolic processors are detailed with attention to the pipelining and flow of data and operations. Parallel algorithms and their optical realization for LU and QR matrix decomposition are specifically detailed. These represent the fundamental operations necessary in the implementation of least squares, eigenvalue, and SVD solutions. Specific applications (e.g., the solution of partial differential equations, adaptive noise cancellation, and optimal control) are described to typify the use of matrix processors in modern advanced signal processing.

I. Introduction

Optical processors have long intrigued researchers and data processors because of their parallelism, high computational rates, small size and weight, and their low power dissipation and cost. Most optical processors have been special-purpose systems performing Fourier transforms and correlations. However, in the past three years, more general-purpose optical processors have emerged that perform matrix–vector and various linear algebra operations. These optical linear algebra processor architectures exhibit pipelining and both local and global interconnections and are generally referred to as optical systolic array processors. In this paper, several architectures and various algorithms for the use of such systems in various applications are reviewed. Because of the parallel nature of these optical systolic array processors, parallel linear algebra algorithms are essential and the flow of data and operations in the system as well as input and output issues merit attention.

Many systolic [1], wavefront [2], and concurrent [3] parallel digital architectures have been suggested in which most processing elements are standard and in which each processing element is always kept active as data flow across the element array. Conventional algorithms (e.g., the International Mathematical and Statistics Library, IMSL) are appropriate for uniprocessors but not for systolic array architectures. A wealth of research on algorithms for multiprocessors and parallel algorithms suitable for systolic processors exist [4]. However, systolic architectures are often devised to implement different algorithms, and the algorithm and system design for complex operations is often complicated by the requirement to utilize only local communication and yet maintain efficiency in the systolic array. Thus even within the digital systolic array community, appropriate algorithms for systolic architectures is a current area of intensive research. No attempt will be made to review digital systolic architectures and algorithms. Rather, attention will be focused only on optical systolic processors and parallel algorithms thus far developed for such systems. Since research on appropriate algorithms and architectures for optical systolic processors is still in the formulation stage, and since different algorithms and implementations detailed are appropriate for each different optical systolic architecture proposed, I shall concentrate on the basic linear algebra operations of matrix–vector, matrix–matrix, and triple-matrix multiplications, plus matrix inversion, direct and indirect solutions of systems of linear algebraic equations (LAEs), and matrix decomposition. These represent the basic linear algebra operations required for more advanced problems such as least squares, eigenvalue, and singular value decomposition (SVD) algorithms needed in advanced modern signal processing.

Optical systolic processors represent an attractive general-purpose system for performing various matrix–vector and linear algebra operations with the high speed and parallelism of the optics being fully utilized. Since many image, pattern recognition, and signal processing problems can and generally are formulated as matrix–vector problems, such optical processors represent a general-purpose and flexible system in which one optical processor can solve a wide variety of problems in many different applications. By examples and specific case studies, these features will be shown.

Many different optical matrix–vector processors have been described as far back as 1965. A survey of these

Manuscript received March 1, 1984; revised March 22, 1984. This work was supported by the NASA Lewis Research Center under Grant NAG-3-5, the Air Force Office of Scientific Research under Grant AFOSR-79-0091, and the NASA Langley Research Center under Grant NAG-1-409, as well as the contractors of Unicorn Systems Incorporated.

The author is with the Department of Electrical and Computer Engineering, Carnegie-Mellon University, Pittsburgh, PA 15213, USA.

0018-9219/84/0700-0831$01.00 ©1984 IEEE

architectures is available [5]. Several of the current systolic architectures advanced in the past two years are reviewed in Section II. These are separated into architectures for matrices with special structure and those for general matrices. How these various architectures achieve the basic operations of matrix–vector and matrix–matrix multiplication is detailed. The solution of systems of linear algebraic equations is a central problem in engineering and computational mathematics. Thus the basic indirect (Section III) and direct (Section IV) parallel algorithms for this fundamental operation are reviewed and optical implementations for each are detailed, with attention to pipelining and flow of data and operations in the system. The extension of these basic operations to advanced problems such as least square, eigenvector, and SVD solutions are then briefly reviewed (Section V). Three specific applications are then briefly discussed to detail how the basic operations (the vector inner product, matrix–vector multiplication, and the solution of systems of LAEs) and matrices with special structure arise. The applications chosen include: the solution of partial differential equations (Section VI-A), adaptive noise filtering (Section VI-B), and optimal control requiring the solution of quadratic matrix equations (Section VI-C). Accuracy and performance issues are then addressed together with a summary and conclusion in Section VII.

II. Optical Linear Algebra Processor Architectures

A plethora of optical matrix–vector and systolic architectures have been described in the past several years. These include: the original Naval Ocean Systems Center [6], Stanford [7], and Carnegie-Mellon University [8] systems, beam modulator systems using change-coupled device (CCD) shift register detector readout [9], beam modulator systems without CCD shift register readout [10], banded and Toeplitz matrix acoustooptic (AO) systems [10], iterative AO systolic architectures [10], vector outer product systems using time-integrating detectors and crossed AO cells or two-dimensional (2-D) spatial light modulators (SLMs) [11], an engagement-mode processor using multichannel AO cells [12], frequency-multiplexed AO processors [13], an engagement-mode (RUBIC) cube processor using 2-D SLMs [14], and architectures combining one-dimensional (1-D) and 2-D SLMs [15]. Many optical systolic architectures for improved accuracy and performance have also been described. These include: accurate vector outer product processors [16], an accurate RUBIC cube processor [17], architectures using 1-D and binary 2-D SLMs [18], and an accurate engagement-mode processor using multichannel AO cells (the systolic AO binary convolver, SAOBIC) [19]. Several of these architectures are reviewed elsewhere in this issue [20], [21].

In this present paper, only analog optical systolic processors using single-channel AO cells are considered (such systems are readily available with present component technology). All of the systems and algorithms described in this paper can be extended fairly directly to use multichannel AO cells and binary 2-D SLMs. Such extensions increase the number of operations performed per second. It appears best to use the added dimension of such systems to achieve improved system accuracy (as accomplished, for example, in the SAOBIC architecture [19]) rather than increased computational rates [22]. The method by which such advanced optical systolic processors achieve digital accuracy uses data encoding and the basic algorithm for digital multiplication by convolution first described in [23] (and first applied to optical architectures in [24]). Examples of several such architectures are described elsewhere in this issue [20], [21].

Initial laboratory demonstrations have been provided for several of the architectures and algorithms described [6]–[8], [18], [22], [25]. Many of the optical systems noted above produce 2-D output data in parallel. Use of such systems in most applications requires advanced detector readout methods with parallel A/Ds, detectors, and parallel high-speed post-processing logic. In incorporating such 2-D output systems into the algorithms described, parallel readout of one row or column of the 2-D output is assumed. In the block diagrams that will be used to describe the various algorithms and architectures, a generic optical systolic processor that performs one matrix–vector multiplication every bit time T_B is assumed (with a parallel linear input and a parallel linear output array). With associated modifications, most of the algorithms described can be implemented on the various optical systolic array processors (with associated modifications to the data flow and computational time, depending upon the specific processor used).

A. Systems for Matrices with Special Structure

As the first class of AO systolic processors, we consider systems suitable for matrices with special structure. The system of Fig. 1 consists of N point modulators whose

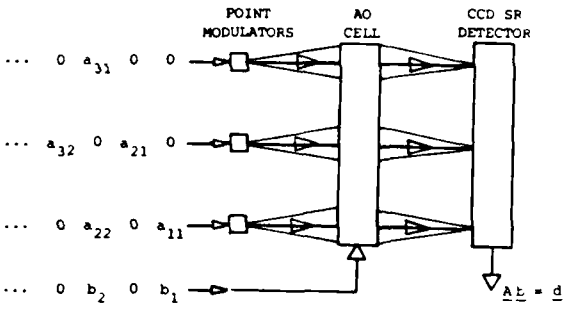


Fig. 1. Simplified schematic of a banded-matrix optical systolic processor using CCD shift register (SR) detector readout (adapted from [9]).

outputs are imaged through different regions of an AO cell and onto different output detectors. AO cells and conventional AO architectures are detailed elsewhere [25], but a simplistic description is included herein for completeness. Electrical data fed to an AO cell are converted to an acoustic wave which travels the length of the cell and introduces spatial and temporal variations in the dielectric constant of the acoustic material. When the data reach the end of the cell, they are absorbed. When the cell is illuminated with light, the amplitude or intensity of the light leaving the cell is modulated spatially in proportion to the strength of the acoustic field in the cell (i.e., proportional to the strength of the input electrical signal), and the light leaves the cell at an angle proportional to the spatial frequency of the acoustic signal (i.e., proportional to the frequency of the input electrical signal). These two properties of AO cells will be employed in the various architec-

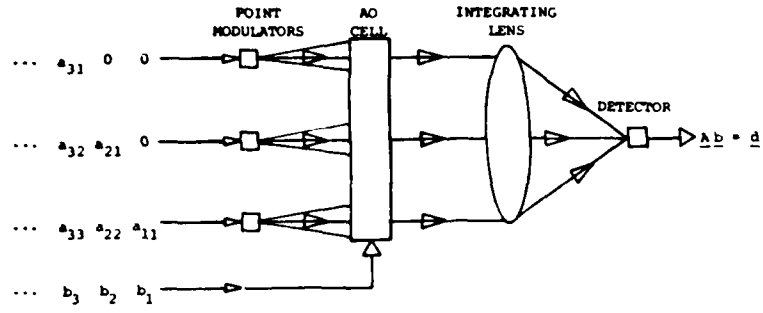


Fig. 2. Simplified schematic of a banded-matrix and Toeplitz-matrix optical systolic processor [10] with only a single detector. This architecture exhibits local and global interconnections and performs a vector inner product as its basic operation.

tures described. For simplicity, we will omit the details of the single-sideband filtering required in such architectures.

With respect to Fig. 1, the elements of the vector are fed time-sequentially to the AO cell. Each vector element is assigned a given time slot in the input electrical signal and the electrical power in each time slot is proportional to the desired vector element value. We denote the length of the AO cell in time by T_A and the time duration of each data packet or vector element by T_B. For simplicity, we assume $N = T_A/T_B$ data packets or pulses can be present in the cell (in practice, some time spacing, i.e., a guard band, will be required between data pulses). The parameters N and T_B are set by the time–bandwidth product (TBWP) of the AO cell, TBWP $= T_A W_A = 1000$ being a typical value, where W_A is the bandwidth of the cell. We consider the use of the system of Fig. 1 to form the matrix–vector product $\boldsymbol{Ab}$

$$\begin{bmatrix} a_{11} & & & & & \\ a_{21} & a_{22} & & & & \\ a_{31} & a_{32} & a_{33} & & & \\ & a_{42} & a_{43} & a_{44} & & \\ & & a_{53} & a_{54} & a_{55} & \ddots \\ & & & \ddots & \ddots & \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \end{bmatrix} \quad (1)$$

where the matrix $\boldsymbol{A}$ is banded. We denote matrices and vectors by bold face upper and lower case letters, respectively. The bandwidth of the matrix is the number of nonzero diagonals (three in (1)).

The matrix–vector product in (1) can be accomplished optically on the system of Fig. 1 by feeding the vector $\boldsymbol{b}$ to the AO cell time-sequentially as shown and the three diagonals of the matrix as time-histories to the three input point modulators. The data modulation on the electronic time-history input signals for this case is shown in Fig. 1 (time increases from right to left in the figure). New data enter the system every T_B with zero-valued data packets of duration T_B placed between each vector and matrix element. When the input point modulators are pulsed on, the light intensity leaving the AO cell is the point-by-point product of the input data to the point modulators and the associated RF input to the AO cell. These point-by-point products are collected on separate output detectors. The contents of the output CCD detector array are then shifted down by one and at the next T_B, the new point-by-point products are added (by charge accumulation) to the shifted data previously present on the detectors. It is easily shown that the time-history output from the single channel on the linear detector CCD shift-register (SR) array is seen to be the desired matrix–vector product $\boldsymbol{Ab} = \boldsymbol{d}$ in (1). This first AO systolic processor architecture was described by Caulfield *et al.* [9] earlier. At each point in the AO cell (opposite an input point modulator), the system performs a multiplication, and on the associated output detector this scalar product is added to a prior value obtained from the neighboring local element. In this sense, this architecture is the optical equivalent of a conventional digital systolic architecture [1].

Another AO systolic architecture for multiplication of a banded matrix by a vector, described by Casasent [10], [13], is shown in Fig. 2. In this system, the nonzero elements of each *row* of the matrix are fed one row at a time in parallel to the input point modulators and the vector data are fed time-sequentially to the AO cell. Each T_B, the elements of the input matrix row are multiplied by the corresponding elements of the vector $\boldsymbol{b}$, and the sum of these scalar products is produced on a single output detector. The integrating lens shown achieves this summation of partial products. Thus each T_B, this architecture multiplies one row of the matrix by the associated elements of the vector $\boldsymbol{b}$ and thus one element of the $\boldsymbol{Ab} = \boldsymbol{d}$ vector is produced each T_B. Other techniques to perform banded matrix–vector multiplications exist that avoid the need for the CCD SR detector readout required in the beam-modulator architecture of Fig. 1. The system of Fig. 2 is the simplest since it requires only one output detector. Such architectures are quite attractive for banded matrices since the number of input point modulators required only need equal the bandwidth of the matrix, and only one output detector is necessary. Such processors exhibit the local interconnection feature of digital systolic processors together with a global interconnection feature unique to optical systems (i.e., addition of all separate element products by use of an integrating lens). The system of Fig. 2 thus performs one vector inner product each T_B of time. In designing optical systolic array architectures, the unique global interconnection features, which have served optical processing systems very well in the past, should not be abandoned, and thus optical systolic architectures should not attempt to emulate the various digital systolic architectures that are proposed. The subsequent optical systolic architectures to be described make use of this philosophy.

Let us next consider an optical systolic processor to multiply a vector by a Toeplitz matrix (the elements along a diagonal are constant in a Toeplitz matrix, i.e., the elements

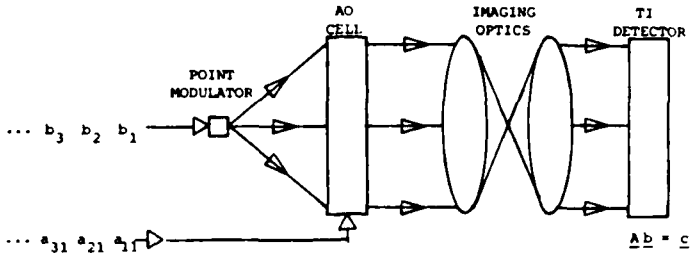


Fig. 3. Simplified schematic of a Toeplitz-matrix optical systolic processor [10] for matrices with large bandwidths

of each row of the matrix are the same shifted by one position). In this case, the architecture of Fig. 2 again suffices [10]. Now, the nonzero elements of one row of the matrix are *fixed* inputs (constant with time) to the input point modulators, and the vector data are fed to the AO cell. Each T_B, one row of the matrix is multiplied by the corresponding vector elements (the AO cell achieves the time-delay shift required to align the matrix elements with the proper vector elements automatically) and summed (by the integrating lens) to produce one element (a vector inner product) of the final matrix–vector product on a *single* output detector every T_B. The number of input point modulators required and the TBWP needed for the AO cell are determined by the size of the matrix.

An alternate architecture devised by Casasent [10] for the multiplication of a vector $\boldsymbol{b}$ by a Toeplitz matrix $\boldsymbol{A}$ is shown in Fig. 3. In this system, the elements of $\boldsymbol{b}$ are fed time-sequentially to *one* input point light modulator whose output uniformly illuminates an AO cell fed with the data $\boldsymbol{a}$ *in one column of the Toeplitz matrix* $\boldsymbol{A}$. The light distribution leaving the AO cell at each T_B is a scalar–vector product, i.e., $b_n\boldsymbol{a}_n$ (where the input to the point modulator b_n is the associated element of the vector $\boldsymbol{b}$, and $\boldsymbol{a}_n$ contains all of the elements of column n of the matrix $\boldsymbol{A}$, properly apertured). This $b_n\boldsymbol{a}_n$ product is imaged onto a linear output detector array. At the next T_B, $b_{n+1}\boldsymbol{a}_{n+1}$ is formed and added to the previous scalar–vector product. Thus after NT_B (the integration time of the detector) the entire matrix–vector product $\boldsymbol{Ab} = \boldsymbol{d}$ is present on the output detectors. This Toeplitz matrix–vector product is thus achieved as

$$\boldsymbol{Ab} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_n][b_1 \cdots b_n]^T = b_1\boldsymbol{a}_1 + b_2\boldsymbol{a}_2 + \cdots = \boldsymbol{a} * \boldsymbol{b} \tag{2}$$

where $\boldsymbol{a}_n$ is the nth column of the matrix $\boldsymbol{A}$ and b_n is element n of the vector $\boldsymbol{b}$. Since all columns of $\boldsymbol{A}$ are shifted versions of each other, the matrix–vector product is simply the convolution of the elements of $\boldsymbol{b}$ and the elements of one column $\boldsymbol{a}$ of $\boldsymbol{A}$. We denote this by $\boldsymbol{a} * \boldsymbol{b}$ in (2) (where $*$ denotes convolution). This formulation of a matrix–vector product as a convolution is also employed in many high-accuracy digital optical systolic processors.

Persons familiar with AO signal processors will recognize the system of Fig. 2 as a space integrating AO correlator and the system of Fig. 3 as a time integrating acoustooptic correlator. These architectures have existed and have been used for correlation signal processing for many years [25]. Thus in retrospect, optical signal processors have used systolic architectures for many years (but under different names). The architecture of Fig. 3 is preferable for Toeplitz matrix applications when the bandwidth of the matrix is large, and the architecture of Fig. 2 is preferable when the length of the vector is large. Since convolution is commutative, the roles of the matrix and vector can be reversed in either architecture, as the application or system fabrication merits.

B. Systems for General Matrices

The architecture of Fig. 1 (or an associated architecture with parallel readout detectors) can be extended to handle general matrices at a significant increase in computational difficulty. The architectures of Figs. 2 and 3 are most appropriate for matrices with special structure (banded or Toeplitz). Special techniques for circulant matrices (as arise in FFTs) and other matrix structures are also possible and follow directly from conventional linear algebra. In this subsection, we discuss optical systolic processors suitable for the multiplication of a general matrix by a vector (i.e., when the matrix has no specific structure). The two major architectures considered are an AO modulator and an AO modulator–deflector. The modulator system is analogous to that of Fig. 1 but with separate detectors with parallel readout (and a rearrangement of the method for feeding data to the system). This system is described elsewhere [20] and is thus not detailed here. Rather, the frequency-multiplexed modulator–deflector architecture of Fig. 4 is described. In this latter system [13], M input point modulators are imaged through M spatially separated regions of an AO cell, and the Fourier transform of the light distribution leaving the AO cell is formed in the back focal plane of the lens where it is sensed by a linear output detector array with parallel outputs. This system is thus topologically identical to that of Fig. 2 with the addition of parallel output detectors. For simplicity, only five input point modulators are shown in Fig. 4.

We describe the operation of the optical systolic processor in Fig. 4 for the case when M signals (vectors), each of length N and each on a separate temporal frequency, are present simultaneously in the AO cell. We refer to this as frequency-multiplexing of the input AO cell data. When the N input point modulators are pulsed on in parallel, the associated input vector multiplies all M vectors in the AO cell. This produces the elements of M separate vector inner products. Each vector inner product will leave the AO cell at a different angle (proportional to the temporal frequency used for each of the M input signal vectors to the AO cell). The Fourier transform lens thus forms each of these M vector inner products on M separate output plane detec-

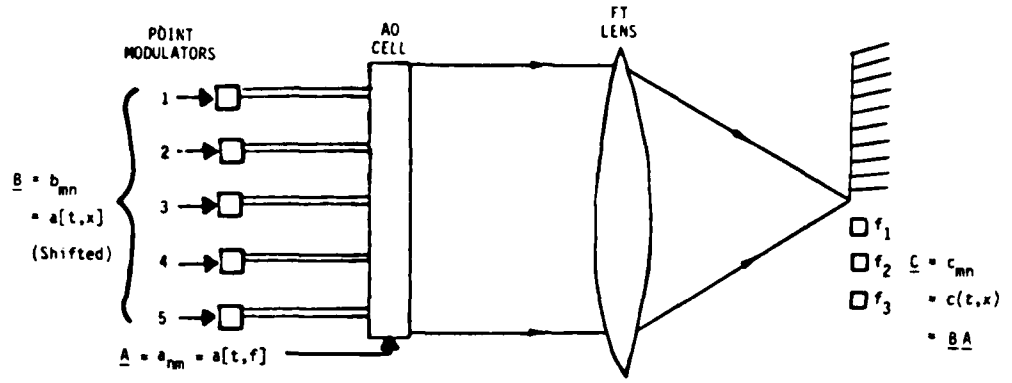


Fig. 4. Simplified schematic of a frequency-multiplexed general-matrix acoustooptic (AO) systolic processor [13] with both local and global interconnections and with a matrix–vector multiplication as its basic operation.

tors, i.e., the system performs a matrix–vector multiplication in parallel each T_B of time. During the next T_B, the matrix data in the AO cell shift up by T_B. At this time, N input point modulators (spatially shifted up by one) are pulsed on with new vector data and a new matrix–vector product (with the same matrix as before) is formed in parallel on the output detectors. After MT_B, a matrix–matrix product has been formed as M matrix–vector multiplications (one per T_B). For square matrices ($M = N$), N^2 is the required TBWP of the AO cell. The motivation for this architecture devised by Casasent *et al.* [13] was that an AO modulator architecture requires $N' = \text{TBWP}$ input point modulators and output detectors to fully utilize the processing capability of the AO cell. In the AO modulator–deflector system of Fig. 4, the bit time T_B and AO cell bandwidth W_A are traded ($T_A W_A = N^2 = \text{TBWP}$ is fixed). The resultant frequency-multiplexed architecture is easier to fabricate, can use larger T_B intervals, and performs a more intensive basic operation (matrix–vector multiplication versus a vector inner product) each T_B. For an $N \times N$ matrix–matrix multiplication, we require an AO cell with $\text{TBWP} = N^2$, $2N - 1$ input point modulators, and N detectors. If the input point modulators are pulsed on with new vector data faster than every T_B, higher computation rates are possible, and the modulator and deflector architectures achieve the same computation rate, with the associated need to feed input data and collect output data at a faster rate.

We now discuss how all of the basic linear algebra operations required can be accomplished on the system of Fig. 4 by various data encoding choices. If the matrix $\boldsymbol{A}$ is fed to the point modulators one row at a time in parallel, i.e., with its elements a_{mn} time and space multiplexed as $a[t, x]$, and the vector $\boldsymbol{b}$ is fed time-sequentially to the AO cell, i.e., with its elements b_n encoded as $b[t]$, then the matrix–vector product $\boldsymbol{Ab} = \boldsymbol{c}$ is formed one element c_m at a time as $c(t)$ on a single output detector. In this case, the degenerate system of Fig. 2 is adequate (or Fig. 4 without frequency multiplexing and with only a single output detector used). In the case when $\boldsymbol{A}$ is fed one column in parallel per T_B to the AO cell, i.e., its elements a_{mn} are encoded as $a_{mn} = a[f, t]$ and all elements b_n of $\boldsymbol{b}$ are fed in parallel to the point modulators as $b_n = b[x]$ (i.e., space-multiplexed), then the matrix–vector product $\boldsymbol{Ab} = \boldsymbol{c}$ is formed in parallel in space on the output detectors, i.e., as $c_m = c(x)$. In effect, the AO cell converts time to space (i.e., $t \rightarrow x$), and the Fourier transform lens behind the AO cell converts temporal frequencies to spatial coordinates (i.e., $f \rightarrow x$). Next, we consider matrix–matrix multiplication. If $\boldsymbol{A}$ is fed to the AO cell one row at a time as $a_{nm} = a[t, f]$ and $\boldsymbol{B}$ is fed to the input point modulators as $b_{mn} = b[t, x]$, the matrix product $\boldsymbol{BA}$ is produced one row at a time in parallel on the output detectors. With the opposite encoding, $a_{mn} = a[f, t]$ and $b_{nm} = b[x, t]$, the matrix product $\boldsymbol{AB}$ is produced one column at a time in parallel on the output detectors. Reference [20] provides a tutorial description of the frequency-multiplexed architecture of Fig. 4 for those readers less versed in optical Fourier transforms.

In Table 1, we summarize the various operations that result from the different possible encoding choices. The two matrix–matrix multiplication techniques, with the product matrix fed back to the AO cell and a new matrix $\boldsymbol{C}$ fed to the input point modulators, produces the triple-matrix products $\boldsymbol{CBA}$ or $\boldsymbol{ABC}$. Various other operations can be performed on this system using these basic functions. These are noted under applications in Table 1. Subsequent sections will detail each of these. They are included in Table 1 at this time for completeness. In general, these operations are accomplished by feeding back the output of the system to the AO cell or the input point modulators. In subsequent sections, we will assume that the matrix to be processed can be accommodated in half of the AO cell and that one row or column of it can be accommodated by half of the input point modulators. If this is not the case, matrix partitioning techniques are required. Such issues are not included at present to simplify description of the algorithms and architectures.

Other encoding schemes are, of course, possible, but have thus far not been found useful. These include: a) $\boldsymbol{A} = a[t, f]$ and $\boldsymbol{b}[x]$, which yields $\boldsymbol{A}^T\boldsymbol{b}$ in parallel; b) $\boldsymbol{A} = a[t, f]$ and $\boldsymbol{B} = b[x, t]$, which yields $\boldsymbol{A}^T\boldsymbol{B}$; etc.

III. Parallel Systolic Indirect Algorithms for the Solution of Systems of Linear Algebraic Equations

A wealth of literature exists on various algorithms for the solution of systems of linear algebraic equations (LAEs), where we wish to find the vector solution

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b} \tag{3}$$

Table 1 Format Control for Flexibility and Data Flow in Fig. 4

Operation	Notation	AO Cell	Point Modulators	Applications
Matrix–Vector Multiplication	Ab	$b = b[t]$	$A = a[t, x]$	Banded Matrix Multiplication Solve Banded Matrix Problems Solve Triangular Matrix Problems (Feedback to AO Cell or Point Modulators)
Matrix–Vector Multiplication	Ab	$b = b(t)$	$A = a(t)$ (one row)	Solve Toeplitz Matrix Problem (Feedback to AO Cell)
Matrix–Vector Multiplication	Ab	$A = a(t)$ (one column)	$b = b(t)$ (serially)	Solve Toeplitz Matrix Problem (Feedback to Point Modulators)
Matrix–Vector Multiplication	Ab	$A = a[t, t]$	$b = b[x]$	Solve Systems of LAEs (Feedback to Point Modulators)
Matrix–Matrix Multiplication	BA	$A = a[t, t]$	$B = b[t, x]$	Triple Matrix Product CBA LU Matrix Decomposition Direct LAEs Solution by LU or QR Least Squares Solution by LU or QR (Feedback to AO)
Matrix–Matrix Multiplication	AB	$A = a[t, t]$	$B = b[x, t]$	Triple Matrix Product ABC (Feedback to AO) QR Matrix Decomposition Without Vector Outer Product Processor (Feedback to Point Modulators and AO Cell)

to the matrix–vector equation

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{4}$$

where all vectors and matrices are assumed to be of order N (i.e., the order of the system of equations) throughout this paper. This one problem is central to many image and signal processing applications, and thus we detail various solutions for LAEs in this and the next section. The material in Sections III and IV draws heavily on several surveys of operations achievable on optical systolic processors [26], [27] and associated journal literature [8], [28]–[30]. As detailed by Rice [31] and others, the two major classes of solutions to systems of LAEs are direct (matrix decomposition) and indirect (iterative). Direct algorithms are discussed separately in Section IV.

Four linear iterative algorithms to solve the LAEs $\boldsymbol{Ax} = \boldsymbol{b}$ can easily be identified. These solutions emerge from the additive splitting of the coefficient matrix into

$$\boldsymbol{A} = \boldsymbol{D} - \boldsymbol{L} - \boldsymbol{U} \tag{5}$$

where $\boldsymbol{D}$ is a diagonal and nonsingular matrix, $\boldsymbol{L}$ is lower triangular (elements only on and below the main diagonal), and $\boldsymbol{U}$ is upper triangular. The four iterative algorithms then become [26], [27], [32], [33]: the Richardson algorithm (also called simultaneous-displacement or semi-iterative, depending upon whether ω is constant with j)

$$\boldsymbol{x}(j+1) = \boldsymbol{x}(j) - \omega\boldsymbol{A}\boldsymbol{x}(j) + \omega\boldsymbol{b} \tag{6}$$

the Jacobi algorithm

$$\boldsymbol{x}(j+1) = \left[\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})\right]\boldsymbol{x}(j) + \boldsymbol{D}^{-1}\boldsymbol{b} \tag{7}$$

the Gauss–Seidel algorithm

$$\boldsymbol{x}(j+1) = \left[(\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{U}\right]\boldsymbol{x}(j) + (\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{b} \tag{8}$$

and the successive overrelaxation (SOR) algorithm

$$\boldsymbol{x}(j+1) = \left\{(\boldsymbol{D}-\omega\boldsymbol{L})^{-1}\left[(1-\omega)\boldsymbol{D} + \omega\boldsymbol{U}\right]\right\} \cdot \boldsymbol{x}(j) + \omega(\boldsymbol{D}-\omega\boldsymbol{L})^{-1}\boldsymbol{b}. \tag{9}$$

In (6) and (9), ω is an acceleration or scaling parameter that regulates the rate of convergence and appropriately scales the eigenvalues to insure convergence. The choice of one of these four iterative algorithms depends on many factors that are highly application and problem dependent (e.g. convergence of the algorithm, the dynamic range of the matrices, the number of iterations required, and the ease of implementation). The Gauss–Seidel algorithm in (8) is equivalent to the SOR algorithm in (9) when $\omega = 1$. Convergence of the algorithms in (7)–(9) requires that $\boldsymbol{A}$ have various specific properties [32], [33]. In (6), $\boldsymbol{A}$ must be completely stable or unstable; in (7), $\boldsymbol{A}$ must be strongly diagonally dominant; and in (8) and (9), $\boldsymbol{A}$ must be positive definite (i.e., have only positive eigenvalues). Calculation of ω in (9) imposes other matrix conditions [32], [33]. We have chosen to concentrate on the Richardson algorithm in (6) because selection of ω for stability and convergence is quite easy, because convergence is insured when $\boldsymbol{A}$ (or $-\boldsymbol{A}$) is stable (i.e., when all eigenvalues lie strictly in the left (right)-half plane), and because its optical implementation is easy to detail. The selection of ω and stopping criteria are discussed later.

To understand how such iterative algorithms are implemented using matrix–vector processors, let us consider the use of (6) to solve $\boldsymbol{Ax} = \boldsymbol{b}$. At iteration one, we use our initial estimate $\boldsymbol{x}(0)$ of $\boldsymbol{x}$ and form $\boldsymbol{Ax}(0)$ (this requires a matrix–vector multiplication). We then subtract this from the vector $\boldsymbol{b}$, multiply the result by ω, and add $\boldsymbol{x}(0)$ to the result. This produces the right-hand side of (6) and hence our next $\boldsymbol{x}(j=1)$ estimate, which is then fed back to the input of the system. We then repeat the above operations with $\boldsymbol{x} = \boldsymbol{x}(1)$. These iterations continue until $\boldsymbol{x}(j+1) \cong \boldsymbol{x}(j)$. Stopping criteria for iterative algorithms are discussed below. When $\boldsymbol{x}(j+1) \cong \boldsymbol{x}(j)$, we see that (6) reduces to $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{x} = \boldsymbol{x}(j)$, and thus the resultant $\boldsymbol{x}$ is the solution $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$ to $\boldsymbol{Ax} = \boldsymbol{b}$. This iterative algorithm thus requires successive matrix–vector multiplications and vector addition at each iteration. The basic element of a processor to iteratively solve LAEs is thus a matrix–vector multiplier

Let us now consider the use of the systems of Figs. 2–4 to realize the iterative solution in (6) to systems of LAEs. For the banded-matrix architecture of Fig. 2, the matrix is fed to the input point modulators and the vector to the AO cell and a new vector element is produced at the output detector each T_B. The aperture time $T_A = NT_B$ for the AO cell is often such that N is much larger than the bandwidth of banded matrices. Thus the input point modulators can be located at appropriate positions near the upper end of the AO cell and the lower portion of the AO cell can be used to store the calculated vector elements as they are produced. In this case, newly produced vector data (one element per T_B) at the output are used to produce the right-hand side of (6) and are then immediately fed to the new data slot available at the bottom of the AO cell. Such an architecture is shown in Fig. 5 for the system of Fig. 2 (a degenerate case of Fig. 4 with one output detector). This system requires only one detector, a one-channel resistive subtractor and adder, and a single operational amplifier [10]. To iteratively solve a set of Toeplitz LAEs, the systems of Figs. 2 or 3 can be used as the basic block with feedback to the AO cell [10].

Next, we consider the solution of LAEs with general matrices without special structure. In this case, the architecture of Fig. 4 (or similar ones that produce one matrix–vector multiplication per T_B) is considered as the basic element in the system. For the specific system of Fig. 4, the vector $x(j)$ data are fed in parallel to the input point modulators and the matrix A data are fed to the AO cell as $a[f, t]$. After the matrix data have been loaded into the AO cell (after a latency time NT_B), the elements of the vector $x(j)$ are applied to the input point modulators. Immediately, the matrix–vector product $Ax(j)$ is produced in parallel on the output detectors. This output matrix–vector product is read out in parallel, operated upon by the N elements of the vector b, etc., to produce the right-hand side of (6) in dedicated analog hardware. The new $x(j + 1)$ vector data are then fed back to the input point modulators in parallel. At the next T_B bit time, the next iteration occurs. Thus, one iteration occurs every T_B, and the data flow in the system is such that the processor is kept fully active, i.e., the data-handling requirements of the output vector data produced and the input vector data required are simultaneously satisfied by the feedback arrangement in a system which is easily synchronized. The associated architecture and data flow for this algorithm are shown in Fig. 6. The system of Fig. 5 employs the data encoding scheme in line one of Table 1, and the system of Fig. 6 uses the encoding noted in line two of Table 1. After NT_B of time, the system of Fig. 6 has performed N iterations of the Richardson algorithm in (6). If the problem is such that the iterations will not converge sufficiently in this amount of time, then the matrix contents of the AO cell are constantly recycled; i.e., as matrix data reach the upper end of the cell, the associated new matrix data are immediately reentered into the bottom of the cell. The length of the cell need only satisfy $T_A = NT_B$ (where N is the size of the matrix to be processed), and only N input point modulators are required [34]. In the case where the aperture of the cell satisfies $T_A = 2NT_B$, new matrix or vector data can be entered into the bottom of the cell as required for the next operation following the solution of the LAEs. As we briefly discuss in Section V, the solution of LAEs is rarely the only operation to be performed in advanced modern signal processing. As further

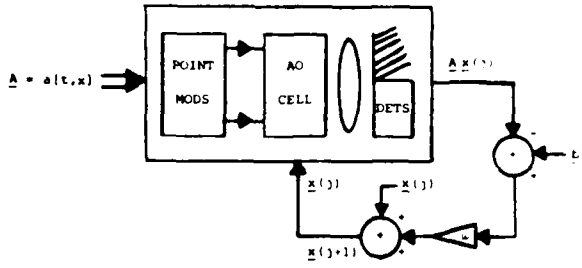


Fig. 5. Simplified schematic of an optical systolic processor [10] to solve banded-matrix LAEs by indirect algorithms

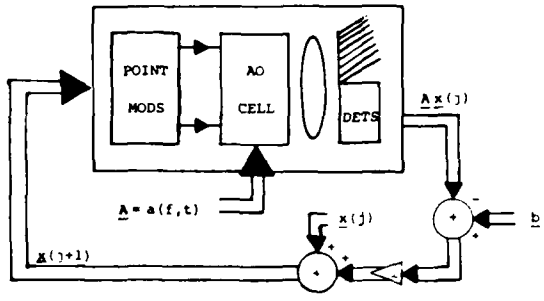


Fig. 6. Simplified schematic of an optical processor [13] to solve LAEs with general matrices using indirect algorithms

extensions of iterative algorithms, we note that (6) and the associated processors can also be realized for the case when A, b, and ω are time-varying functions of the iteration index j. This extension allows the general LAEs solution presented to be extended to time-varying stochastic gradient-following algorithms in adaptive filtering and signal processing.

Experimental demonstrations of the algorithm in Fig. 4 for a Toeplitz matrix in a deconvolution application [10] and for the system of Fig. 5 for a full matrix [34] have been reported. Iterative algorithms require attention to stability, convergence, the choice of ω, and the stopping criteria. These issues are problem dependent but easily obtained given the available *a priori* problem specifics. Iterative algorithms appear essential for the solution of eigensystems and for singular value decomposition, as noted in Section V, and thus much future work on such algorithms is expected. In general, the successful use of iterative algorithms requires slight application-dependent algorithm and matrix modifications. Specific examples will be detailed in future publications. However, several general guidelines are advanced below.

Let us now address the general guidelines for parameter selection in indirect algorithms. We will consider selection of ω and the stopping criteria. For such analyses, we believe that one should utilize deterministic engineering techniques and digital simulation rather than formal mathematical analyses (since formal analyses are valid only "in-the-limit") and develop tight upper bounds from analytical models to characterize convergence of the algorithm [27]. We will first develop the general expressions for the case of the iterative algorithm in (6) and then separately address selection of ω and the stopping criteria. We first note that the right-hand side of (6) weights the difference between the calculated solution $x(j)$ at iteration j and the weighted error $\omega[Ax(j) - b]$ in the exact solution. The

algorithm successfully reduces this weighted error. This is best seen by writing the computational error $\mathbf{e}(j)$ at the jth iteration as the difference between the calculated solution $\mathbf{x}(j)$ at the jth iteration and the exact solution $\mathbf{x}^* = \mathbf{A}^{-1}\mathbf{b}$. The error vector is thus

$$\mathbf{e}(j) = \mathbf{x}(j) - \mathbf{x}^*. \tag{10}$$

The error vectors on successive iterations can be related by

$$\mathbf{e}(j+1) = [\mathbf{I} - \omega\mathbf{A}]\mathbf{e}(j). \tag{11}$$

After j Richardson iterations, the error $\mathbf{e}(j)$ is related to the initialization error $\mathbf{e}(0)$ by

$$\mathbf{e}(j) = [\mathbf{I} - \omega\mathbf{A}]^j\mathbf{e}(0). \tag{12}$$

To facilitate selection of a fixed number J of iterations, we require a tight upper bound on the norm of the computational error in (12). The classic upper bound is

$$\|\mathbf{e}(j)\| \leq \|[\mathbf{I} - \omega\mathbf{A}]\|^j\|\mathbf{e}(0)\|. \tag{13}$$

Let us now consider the choice of ω and then return to evaluating (13). It is well known [35] that for the eigenvalues of $[\mathbf{I} - \omega\mathbf{A}]$ to be less than unity, one uses

$$\omega \cong 1/\lambda_{max} \tag{14}$$

and that λ_{max} of $\mathbf{A}$ is bounded by the Euclidean norm as

$$\lambda_{max} \leq \|\mathbf{A}\| = \left[\sum_{m=1}^{M}\sum_{n=1}^{N} a_{mn}^2\right]^{1/2}. \tag{15}$$

Since the upper bound in (15) is weak, we select

$$\omega = p/\|A\| \tag{16}$$

where p is a problem-dependent constant greater than two. In our algebraic Riccati equation (ARE) solutions (Section VI-C), we have emperically [8], [36] selected $p = 3$ and consistently achieved excellent performance in over ten cases investigated.

If $\mathbf{A}$ is symmetric and ω satisfies (14), the norm $\|\mathbf{I} - \omega\mathbf{A}\|$ in (13) is well approximated [33] by the spectral radius (the largest eigenvalue, in absolute value) of $[\mathbf{I} - \omega\mathbf{A}]$, i.e., by $(1 - 1/C)$, where the condition number $C = \lambda_{max}/\lambda_{min}$ (the ratio of the largest to smallest eigenvalues of $\mathbf{A}$, in absolute value). Substituting this into (13), we obtain

$$\|\mathbf{e}(j)\| \leq [1 - 1/C(\mathbf{A})]^j\|\mathbf{e}(0)\| \cong \exp[-j/C(\mathbf{A})]\|\mathbf{e}(0)\|. \tag{17}$$

Equation (17) describes the performance (convergence) of the Richardson algorithm for j iterations. We see that it is determined by C and thus from an estimate of C we can fix the number of iterations at a constant number J to achieve a given accuracy or error in (17). Selection of a fixed J is quite problem-dependent. In our work in Section VI-C we found

$$J = 3.0C \tag{18}$$

has yielded excellent performance for those cases considered. In specific cases, i.e., if $\mathbf{A}$ is the covariance matrix, C can often be approximated by the ratio of the strengths of the signals expected [37]. Since the exact answer is rarely known and in some cases an estimate of C is not easily obtained, one can simply continue the iteration until the norm of the difference between successive iterates is below a preset error threshold [3a]. Goodman and Song [38] have shown that this approach is also helpful in reducing the effect of noise in an optical iterative processor. As noted at the outset, the effective use of iterative algorithms is very problem-dependent and merits further research.

IV. Parallel Systolic Direct Algorithms for the Solution of Systems of LAEs

The iterative solution of LAEs (Section III) was an obvious choice for the initial optical matrix–vector processors using fixed 2-D spatial masks to store the matrix data spatially [8]. However, with the advent of systolic processors using AO cells, a different algorithm philosophy (direct solutions to LAEs) emerged [39], [40]. Since the matrix data in systolic processors shift through the AO cell one row or column per T_B, a new vector row or column of a matrix must be fed to the AO (or other) transducer every T_B (i.e., the matrix in an optical systolic processor must be updated each T_b). Research on direct algorithms for the optical solution of LAEs and for matrix decomposition is thus quite new. However, parallel algorithms for LU [28], QR [29], and Cholesky [28] matrix decomposition and parallel algorithms for the solution of triangular LAEs [30] and general LAEs using optical systolic processors have been detailed and published during the past year. In this section, we summarize this research. In Section V, we discuss various possible extensions of these initial matrix decomposition algorithms to more advanced optical systolic processors.

The general philosophy in matrix decomposition solutions to LAEs is to convert the given $\mathbf{Ax} = \mathbf{b}$ LAE problem into a simpler one where the new matrix has specific structure that allows the solution of a simpler matrix–vector equation (by easily implemented techniques such as forward or backward substitution). Most matrix decomposition techniques are variants of Gaussian elimination. The two conditions used in the various Gaussian-elimination-based algorithms are that: 1) the elements of any row of the matrix $\mathbf{A}$ can be multiplied by a nonzero real number, and 2) a constant multiple of any row can be added to the associated elements of any other row. All matrices produced by these operations are equivalent. The two classic direct LAE solutions are LU (triangular) and QR (orthogonal) matrix decomposition. We thus discuss these two algorithms and their optical systolic processor realization in this section. In Section V, we discuss advanced applications of other decomposition algorithms to other modern signal processing problems.

In LU decomposition, the matrix A in the original $\mathbf{Ax} = \mathbf{b}$ problem is decomposed into a lower $\mathbf{L}$ and upper $\mathbf{U}$ triangular matrix, where the diagonal elements of $\mathbf{L}$ are all "1" and the resulting decomposition is unique. Thus the original $\mathbf{Ax} = \mathbf{b}$ problem (where $\mathbf{x}$ is unknown) becomes (substituting $\mathbf{A} = \mathbf{LU}$) two problems. First, one can solve $\mathbf{LUx} = \mathbf{b}$ for $\mathbf{y} = \mathbf{Ux}$ and second solve $\mathbf{Ux} = \mathbf{y}$ for $\mathbf{x}$, where $\mathbf{U}$ and $\mathbf{y}$ are known from the LU decomposition and the first triangular system solution in step one. Each of these subproblems requires the solution of a lower or upper triangular system of equations. This is trivial on-line in dedicated digital hardware. In QR matrix decomposition, the matrix $\mathbf{A}$ is factored or decomposed into an orthogonal matrix $\mathbf{Q}$ (such that $\mathbf{Q}^T = \mathbf{Q}^{-1}$ or $\mathbf{QQ}^T = \mathbf{I}$) and an upper triangular matrix $\mathbf{R}$. In this case, the original $\mathbf{Ax} = \mathbf{b}$ LAE problem reduces to $\mathbf{Rx} = \mathbf{Q}^{-1}\mathbf{b} = \mathbf{Q}^T\mathbf{b} = \mathbf{b}'$, i.e., the solution of another simplified triangular system of equations $\mathbf{Rx} = \mathbf{b}'$ as

before. Achieving the matrix decomposition in either of these algorithms was recognized early [39] as the major computational step in such algorithms. We thus first detail how to achieve LU and QR decomposition (Sections IV-A and IV-B) on optical systolic processors. Then, we address an optical solution to the triangular system of equations that results (Section IV-C), and finally (Section IV-D), we detail a full direct systolic solution in N matrix–matrix multiplication steps. The major reasons for interest in direct versus indirect solutions to LAEs is that the number of iterations required in an indirect solution (Section III) is not easily quantified and is thus highly problem-dependent. Conversely, direct solutions require a fixed number of steps N (the order of the matrix). The parallel aspects of direct LAE solutions must be properly advanced and not implemented on systolic architectures as in the conventional linear algebra descriptions. Our algorithms and architectures will demonstrate such parallel guidelines and the best use of matrix decomposition with systolic processors.

A. Optical Systolic Realization of LU Matrix Decomposition [-8]

All matrix decomposition solutions to $\boldsymbol{Ax} = \boldsymbol{b}$ involve multiplying $\boldsymbol{A}$ and $\boldsymbol{b}$ by a decomposition matrix $\boldsymbol{P}_0$ (this yields $\boldsymbol{P}_0\boldsymbol{A} = \boldsymbol{A}_1$ and $\boldsymbol{P}_0\boldsymbol{b} = \boldsymbol{b}_1$), multiplying $\boldsymbol{A}_1$ and $\boldsymbol{b}_1$ by a matrix $\boldsymbol{P}_1$, etc. After N such matrix–matrix multiplications, one obtains a matrix $\boldsymbol{PA}$ and a vector $\boldsymbol{Pb} = \boldsymbol{b}'$, where $\boldsymbol{P} = \boldsymbol{P}_{N-1} \cdots \boldsymbol{P}_m \cdots \boldsymbol{P}_0$. In matrix decomposition, each $\boldsymbol{P}_m\boldsymbol{A}_m$ multiplication only affects columns (or rows) m through N of $\boldsymbol{A}_{m-1}$.

In LU decomposition, $\boldsymbol{P}_m$ is chosen to force the elements below the diagonal in the mth column of $\boldsymbol{A}_m$ to be zero. On successive cycles, we require a matrix–matrix multiplication and a matrix–vector multiplication. These operations are combined (since the same matrix is used in both) into the multiplication of the matrix $\boldsymbol{P}_m$ by the augmented matrix $[\boldsymbol{A}_m;\boldsymbol{b}_m]$. Each successive cycle of the system thus requires a matrix–matrix multiplication, calculation of one column of $\boldsymbol{P}_m$, and assembly of the $\boldsymbol{P}_m$ matrix. On each successive cycle, we produce: one row of the upper triangular matrix $\boldsymbol{U}$, one element of the new $\boldsymbol{b}'$ vector (from the matrix–matrix multiplication), and one column of $\boldsymbol{P} = \boldsymbol{L}^{-1}$ (from the calculations of the $\boldsymbol{P}_m$ matrix). This is achieved as detailed below. $\boldsymbol{P}_m$ is an identity matrix except for column m whose elements $p_{k\,m}^{(m)}$ are well-known and easily calculated [28] functions of the elements of the mth row of $\boldsymbol{A}_{m-1}$, i.e.,

$$p_{k\,m}^{(m)} = a_{k\,m}^{(m-1)} / a_{m\,m}^{(m-1)} \tag{19}$$

where superscripts denote the step or matrix–matrix multiplication number. If $\boldsymbol{A}$ is neither strictly diagonally dominant nor positive-definite, pivoting (i.e., interchanging of rows of the matrix) is necessary to insure that (19) is less than unity.

The data flow for use of a systolic processor for LU decomposition is shown in Fig. 7. The matrix $\boldsymbol{A}$ augmented by the vector $\boldsymbol{b}$ (i.e., $[\boldsymbol{A}_m;\boldsymbol{b}_m]$) is fed to the AO cell, multiplied by $\boldsymbol{P}_m$ to yield the new augmented matrix $[\boldsymbol{A}_{m+1};\boldsymbol{b}_{m+1}]$. After each matrix–matrix multiplication, one row of the final $\boldsymbol{U}$ matrix and one element of the final $\boldsymbol{b}'$ vector are produced (i.e., the first row of $\boldsymbol{A}_{m+1}$ and the first element of $\boldsymbol{b}_{m+1}$ are in final converted form). One additional row

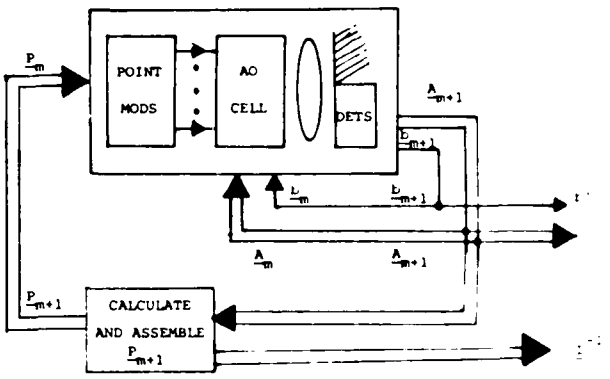


Fig. 7. Simplified schematic of an optical systolic processor [28] to perform LU matrix decomposition of a general matrix by Gaussian elimination.

and column of the augmented matrix just calculated $[\boldsymbol{A}_{m+1};\boldsymbol{b}_{m+1}]$ is thus not altered or needed in each subsequent matrix–matrix multiplication. Thus we remove one row and column of the matrix product produced at each cycle and reduce the order of each subsequent matrix–matrix multiplication by one. The remaining elements of $\boldsymbol{A}_{m+1}$ and $\boldsymbol{b}_{m+1}$ are fed back to the AO cell and the processor as they are produced (one row at a time in parallel). The new elements to be calculated in $\boldsymbol{P}_{m+1}$ require only the elements of one column $m+1$ of the new $\boldsymbol{A}_{m+1}$ matrix as in (19). Since one row of $\boldsymbol{A}_{m+1}$ is produced in parallel each T_B, the column elements of $\boldsymbol{A}_{m+1}$ needed to compute $\boldsymbol{P}_{m+1}$ are available one element each T_B (from the same output detector). As each element of the proper $\boldsymbol{A}_{m+1}$ column is produced, the element in the corresponding column of $\boldsymbol{P}_{m+1}$ is calculated during T_B and stored.

These calculations in (19) are performed in special-purpose analog hardware in the box noted in Fig. 7. These operations are easily achieved during $1T_B$ since only one element per row of $\boldsymbol{P}_{m+1}$ must be computed. Similarly, storage of these $\boldsymbol{P}_{m+1}$ values and formatting the $\boldsymbol{P}_{m+1}$ matrix (an identity matrix except for one column) for input to the point modulators one row at a time in parallel is easily achieved [28]. The operations that the special-purpose analog hardware must perform are inversion of the first element of the appropriate column of $\boldsymbol{A}_{m+1}$ during the first T_B. At subsequent T_B times, this element is multiplied by the new column elements of $\boldsymbol{A}_{m+1}$ to generate the appropriate new elements in the new column in $\boldsymbol{P}_{m+1}$ as defined in (19). Each new row of $\boldsymbol{P}_{m+1}$ defines a row of $\boldsymbol{L}^{-1}$ which is available as an output as shown in Fig. 7. After each matrix–matrix multiplication, one column of $\boldsymbol{L}^{-1}$, one row of $\boldsymbol{U}$, and one element of $\boldsymbol{b}'$ are computed. After N matrix–matrix multiplications, the full LU decomposition has been achieved and $\boldsymbol{L}^{-1}$, $\boldsymbol{U}$, and $\boldsymbol{b}'$ have been produced. Since the order of the matrix is reduced by one on each cycle, and (assuming one matrix–vector multiplication each T_B) the matrix decomposition (including NT_B of latency time to load the cell) thus requires a total time

$$[N + (N-1) + (N-2) + \cdots + 2]\,T_B = [(N^2 + N - 2)/2]\,T_B \tag{20}$$

or (for large N) $\cong N^2/2T_B$, during which (for large N) approximately $N^3/3$ multiplications and additions are performed.

B Parallel Optical Systolic Realization of QR Matrix Decomposition [29]

QR decomposition or orthogonal matrix factorization ($\boldsymbol{A} = \boldsymbol{QR}$) can be accomplished [41], [42] by modified Gramm–Schmidt orthogonalization, Householder plane reflections [41], or Givens plane rotations [43]. The first method requires transposing a matrix and performing two matrix–matrix multiplications to produce one column of $\boldsymbol{Q}$ and $\boldsymbol{R}$. The last method requires one matrix–matrix multiplication to produce one *element* of $\boldsymbol{Q}$ and $\boldsymbol{R}$. Thus the Householder *QR* decomposition appears to be the most practical and most easily paralleled algorithm for *QR* factorization, since it produces one column and row of the final decomposed $\boldsymbol{A}_m$ matrix in each step. The basic steps in this operation are similar to those in Fig. 7 for *LU* decomposition, i.e., successive multiplication of $\boldsymbol{A}$ and $\boldsymbol{b}$ by a decomposition matrix $\boldsymbol{P}_0$ to produce $\boldsymbol{A}_1$ and $\boldsymbol{b}_1$, multiplication of this matrix and vector by $\boldsymbol{P}_1$ to produce $\boldsymbol{A}_2$ and $\boldsymbol{b}_2$, etc. The decomposition matrices in *QR* decomposition are different from those in *LU* decomposition. Each successive matrix–matrix multiplication produces one row of the final upper triangular matrix $\boldsymbol{R}$. After N such matrix–matrix multiplications, we obtain $\boldsymbol{PA} = \boldsymbol{R}$ and $\boldsymbol{Pb} = \boldsymbol{b}'$, where $\boldsymbol{P} = \boldsymbol{Q}^{-1} = \boldsymbol{Q}^T$. *QR* decomposition yields, after each matrix–matrix multiplication, one row of $\boldsymbol{R}$ and one row and column of $\boldsymbol{Q}^T$ as in *LU* decomposition. One row and column of the computed augmented matrix $[\boldsymbol{A}_m; \boldsymbol{b}_m]$ are not needed on subsequent matrix–matrix multiplications. Thus the order of the matrix–matrix multiplications can be reduced by one on each subsequent cycle. This will represent a considerable reduction in the computational time and system performance as we note in Section IV-D. The full $\boldsymbol{A}_m$ matrix after cycle m will have the structure

$$\boldsymbol{A}_m = \left[\begin{array}{c|c} \boldsymbol{R}_m & \boldsymbol{V}_m \\ \hline 0 & \boldsymbol{W}_{N-m} \end{array}\right] \tag{21}$$

where $\boldsymbol{R}$ is an upper triangular matrix. On subsequent steps, only $\boldsymbol{W}$ is changed ($\boldsymbol{W}_{N-m}$ denotes that $\boldsymbol{W}$ for $\boldsymbol{A}_m$ is of order $N - m$). To calculate the next $\boldsymbol{P}_{m+1}$ decomposition matrix, only the elements of the first column of $\boldsymbol{W}$ are needed. We denote the first column of $\boldsymbol{W}$ by the column vector $\boldsymbol{w}_{m+1}$. The equation to generate $\boldsymbol{P}_m$ is [41]

$$\boldsymbol{P}_m = \boldsymbol{I} - k_m \boldsymbol{u}_m \boldsymbol{u}_m^T. \tag{22}$$

The vector $\boldsymbol{u}_m$ in (22) is the same as $\boldsymbol{w}_m$ except for the first element, which is

$$u_1 = w_{m,m} + \ell_m \operatorname{Sign}(w_{m,m}) \tag{23}$$

where ℓ_m is the norm or the vector inner product of $\boldsymbol{w}_m$, i.e.,

$$\ell_m^2 = \sum_{i=1}^{N-m} w_{m+i,m}^2 \tag{24}$$

and the constant k_m is

$$k_m = \left[\ell_m^2 + \ell_m |w_{m,m}|\right]^{-1}. \tag{25}$$

The steps in a Householder *QR* decomposition thus involve 1) calculation of $\boldsymbol{P}_m$ (this requires a vector outer product), and 2) the matrix–matrix and matrix–vector multiplications $\boldsymbol{P}_m \boldsymbol{A}_m = \boldsymbol{A}_{m+1}$ and $\boldsymbol{P}_m \boldsymbol{b}_m = \boldsymbol{b}_{m+1}$. The second operation is performed on the system of Fig. 4 using the data encoding in the last line of Table I. Since calculation of $\boldsymbol{P}_m$ in (23) requires only one column of the new $\boldsymbol{A}_m$ matrix, we generate $\boldsymbol{A}$ one column at a time rather than one row at a time (using the data encoding noted). Step one is clearly the critical operation in terms of data flow and efficient processor use. Since one column of $\boldsymbol{A}$ can be produced in parallel on the system, the vector outer product operation can be performed on the optical system of Fig. 4 by several methods [29], [39]. If $\boldsymbol{A}$ is symmetric, we can [29] operate with the transposed $\boldsymbol{A}^T$ matrix and utilize the symmetry of $\boldsymbol{A}$, $\boldsymbol{A}_m$, and $\boldsymbol{P}_m$. For a nonsymmetric $\boldsymbol{A}$, one can [39] produce one column $\boldsymbol{u}$ of $\boldsymbol{W}$ in parallel and (in N cycles) compute the vector outer product in (22), the norm from the trace of $\boldsymbol{u}_m \boldsymbol{u}_m^T$, and evaluate k_m. Both of these approaches require intermediate data storage and leave the processor inactive during fill times of the AO cell. More attractive data flow results if the optical system in Fig. 8 (using two crossed-point modulator arrays) is used to perform the vector outer product. This vector outer-product system of Fig. 8 is detailed elsewhere [55]. It involves

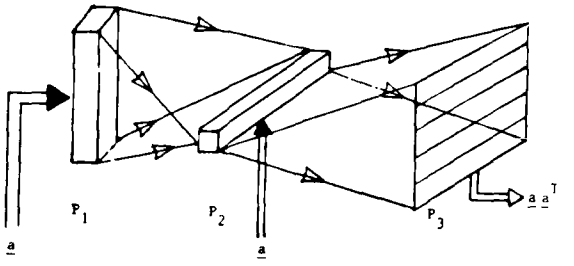


Fig. 8. Simplified schematic of an optical systolic vector outer product processor (adapted from [55]).

imaging the modulator at plane P_1 horizontally onto the output plane P_3, with P_1 compressed vertically and expanded horizontally as shown to uniformly illuminate P_3, and with P_2 imaged horizontally onto the P_3 output plane. More attractive data flow within the optical matrix multiplier results if rows are fed in parallel with data encoding achieved as in the *LU* processor of Fig. 7.

The use of the combined architectures of Figs. 4 and 8 for *QR* matrix decomposition is shown in Fig. 9. In this architecture, one row of $\boldsymbol{A}_m$ is produced in parallel on the output detectors. The first row produced (one row of $\boldsymbol{R}$) is an output and the remaining rows are reinserted into the AO cell. When the first column of $\boldsymbol{A}_m$ has been produced, it is fed in parallel (or sequentially as it is produced) to the outer-product processor of Fig. 8, which produces $\boldsymbol{a}_m \boldsymbol{a}_m^T$ in parallel. This 2-D symmetric output in Fig. 8 is read out one line at a time in parallel, ℓ_m^2 is calculated from the trace of $\boldsymbol{a}_m \boldsymbol{a}_m^T$ (with parallel output detectors along diagonals), k_m is formed, then $\boldsymbol{P}_{m+1}$ is assembled one row at a time in parallel in simple analog hardware and fed to the input point modulators. Thus after one cycle, the necessary rows and columns of $[\boldsymbol{A}_{m+1}; \boldsymbol{b}_{m+1}]$ are in the AO cell and the first row of the new $\boldsymbol{P}_{m+1}$ matrix is available at the input point modulators. The next cycle can thus begin immediately.

After each matrix–matrix multiplication, one row of $\boldsymbol{Q}^T$ and $\boldsymbol{R}$ and one element of $\boldsymbol{b}'$ is formed. Data flow in this system is ideal, and full advantage is made of the reduced matrix order on each cycle. Assuming a negligible time to

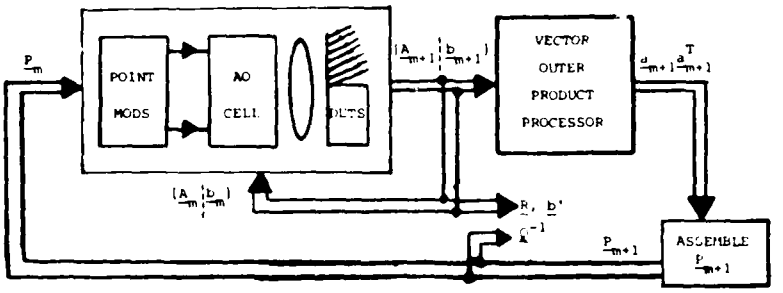


Fig. 9. Simplified schematic of an optical systolic processor [29] to perform *QR* matrix decomposition on a general matrix.

produce the vector outer product and assemble each row of P_{m+1} (this is realistic), the system of Fig. 9 performs a *QR* decomposition in the same $(N^3/2)T_B$ time as in (20).

C. Parallel Optical Systolic Processors for the Solution of Triangular LAEs [30]

In the architecture of Fig. 7, we detailed how all of the matrices and vectors associated with *LU* decomposition could be calculated in parallel, i.e., $\boldsymbol{PA} = \boldsymbol{U}$, $\boldsymbol{Pb} = \boldsymbol{b'}$, and $\boldsymbol{P} = \boldsymbol{L}^{-1}$. In the conventional second and third steps in a direct *LU* decomposition solution to a system of LAEs, one solves $\boldsymbol{Ly} = \boldsymbol{b}$ for $\boldsymbol{y}$ and then $\boldsymbol{Ux} = \boldsymbol{y}$ for $\boldsymbol{x}$. The latter triangular LAEs can easily be solved by back or forward substitution in dedicated hardware and on digital systolic processors. Ghosh and Casasent [30] have also noted that such triangular systems can be solved optically and have detailed the solution for the $\boldsymbol{Ly} = \boldsymbol{b}$ case (the $\boldsymbol{Ux} = \boldsymbol{y}$ case follows directly as noted in Section IV-D). The system of Fig. 10 achieves this and demonstrates the general algorithm and architecture for an optical systolic processor to solve a triangular system of equations. The data flow for the case of a lower triangular system of equations is described. We assume that one row of $\boldsymbol{L}$ and one element of $\boldsymbol{b'}$ are produced in parallel on each cycle. If these elements are fed to the input point modulators and a one-channel adder as in Fig. 10(b), the solution $\boldsymbol{x} = \boldsymbol{L}^{-1}\boldsymbol{b'}$ is obtained sequentially from the system. The algorithm used is [30]

$$x_m = \left(b'_m - \sum_{n=1}^{m-1} \ell_{mn} x_n\right)(1/\ell_{mm}) \qquad (26)$$

where ℓ_{mm} is the associated diagonal element of $\boldsymbol{L}$. For $\boldsymbol{L}$, its diagonal elements are unity in *LU* decomposition and thus $1/\ell_{mm} = \ell_{mm}$ involves no additional calculations. The data flow for this case is shown in Fig. 10(b) and the interconnections between Figs. 7 and 10(b) are shown in Fig. 10(a). This is the most efficient method for a direct solution of LAEs as we discuss in the following section.

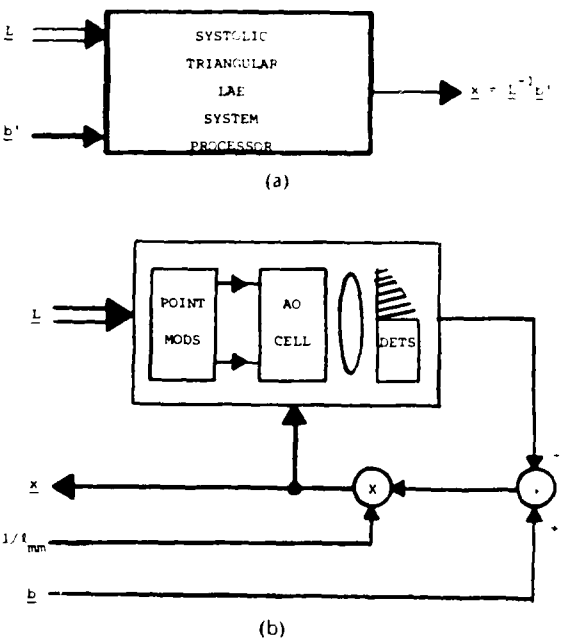


Fig. 10. Simplified (a) and detailed (b) schematic for an optical systolic processor [30] to solve triangular LAEs.

D. Parallel Optical Systolic Processor for the Direct Solution of LAEs

Conceptually, it is easiest to view an *LU* decomposition solution of a system of LAEs as a sequence of successive matrix–matrix multiplications with the augmented matrix $[\boldsymbol{A}_{m+1} \mid \boldsymbol{b}_{m+1}]$. Such an architecture is shown in Fig. 11 for the implementation of the Gauss–Jordan [41] algorithm for the direct solution of a system of LAEs. In this algorithm and architecture, N matrix–matrix multiplications of $\boldsymbol{P}_m$ by

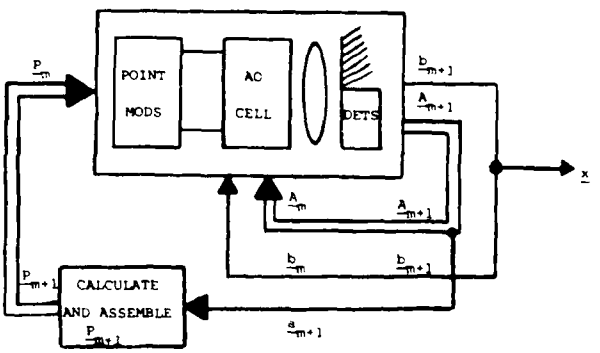


Fig. 11. Simplified schematic of an optical processor for the direct solution of a system of LAEs (a combined version of Figs. 10 and 7 or 9).

the augmented matrix are performed to convert $\boldsymbol{A}$ to an upper triangular matrix $\boldsymbol{U}$. N additional matrix–matrix multiplications are then performed to diagonalize the upper triangular matrix and to properly condition the augmented vector $\boldsymbol{b}$. These last N matrix–matrix multiplications accom-

plish the back-substitution algorithm as depicted in Fig. 10. In the final augmented matrix, $\boldsymbol{A}$ is an identity matrix and the final $\boldsymbol{b}$ vector is the desired solution vector $\boldsymbol{x}$ to the original $\boldsymbol{Ax} = \boldsymbol{b}$ problem. The detailed steps in this Gauss–Jordan algorithm are detailed elsewhere [42], [44]. The required feedback and data flow for such an algorithm are shown in Fig. 11.

In this architecture and algorithm, $2(N-1)$ successive matrix–matrix multiplications are performed (full $N \times N$ matrices are required in this case). This architecture is attractive because of the simplified schematic diagram that results, as in Fig. 11. However, this algorithm requires $2(N-1)$ matrix–matrix multiplications (with all matrices $N \times N$). The execution time for such an algorithm requires $(N-1)T_B$ setup time plus $(2N-2)NT_B$ of time for the matrix multiplications, for a resultant calculation time of

$$(N-1)T_B + (2N^2 - 2N)T_B \cong 2N^2 T_B. \tag{27}$$

This is significantly longer than the time required using a combined version of Figs. 7 and 10 as we now detail. First, we note that $\boldsymbol{L}^{-1}$ and $\boldsymbol{b}$ are available as outputs in Fig. 7. We could form $\boldsymbol{L}^{-1}\boldsymbol{b} = \boldsymbol{y}$ (i.e., the solution $\boldsymbol{y}$ of $\boldsymbol{Ly} = \boldsymbol{LUx} = \boldsymbol{b}$) by a simple matrix–vector multiplication. However, the system of Fig. 7 has already produced $\boldsymbol{L}^{-1}\boldsymbol{b} = \boldsymbol{b}'$ as an output and thus, by the use of an augmented matrix, we have already solved the first lower triangular system of LAEs. Hence, if we feed the stored $\boldsymbol{U}$ and $\boldsymbol{b}'$ data produced from Fig. 7 back to the point modulators and AO cell, we can compute $\boldsymbol{x} = \boldsymbol{U}^{-1}\boldsymbol{b}'$ directly in an additional NT_B of time using the system of Fig. 10(b) and the algorithm

$$x_m = \left(b'_m - \sum_{n=m+1}^{N} u_{mn}x_n\right)(1/u_{mm}) \tag{28}$$

where u_{mn} denotes the elements of $\boldsymbol{U}$.

Similar remarks apply for QR decomposition as achieved in Fig. 9. In this case, the original $\boldsymbol{Ax} = \boldsymbol{b}$ problem is converted to $\boldsymbol{Rx} = \boldsymbol{Q}^T\boldsymbol{b} = \boldsymbol{b}'$, where $\boldsymbol{R}$ and $\boldsymbol{b}'$ are outputs from Fig. 9. Thus this upper triangular system of equations can be solved in an additional NT_B as noted above for the final $\boldsymbol{x}$ solution. Thus in a direct solution implemented as above with augmented matrices, the size of the matrix is reduced by one on each cycle, and $\boldsymbol{U}$ or $\boldsymbol{R}$ and the new $\boldsymbol{b}'$ are stored. Then, $\boldsymbol{U}$ or $\boldsymbol{R}$ are fed to the point modulators and $\boldsymbol{b}'$ to a serial adder (as in Fig. 10), and in an additional NT_B after LU decomposition the final solution is produced. The total time for the LU or QR versions of such an algorithm is thus the time in (20) plus NT_B or

$$(N^2/2)T_B + NT_B \cong (N^2/2)T_B \tag{29}$$

for N large. This is one-fourth the time required in (27), and storage is a factor of N less. Propagation of the full matrix data can introduce and accumulate additional errors, since the many 0 and 1 matrix elements in such a full matrix propagation may not be identically 0 and 1. Thus it is preferable to use the architecture of Fig. 11 as in Figs. 7 or 9 and on the last cycle to feed $\boldsymbol{U}$ or $\boldsymbol{R}$ to the input point modulators. Such an architecture will thus achieve a full direct LU or QR solution to a system of N LAEs in $N^2T_B/2$ of time.

In closing this section, we note that most of the proposed optical systolic processors (Section II) can implement the various algorithms (Figs. 5–11) with different degrees of parallelism and with associated detector complexity (depending upon the precise architecture). Specific issues of the detector system readout and the data and operational flow must be detailed for each system to assess the best architecture to use for a given application and operation.

V. Extensions of Basic Algorithms and Operations

The matrix–vector, matrix–matrix, and triple-matrix multiplication operations (Section II), plus the matrix decomposition methods (Section IV) and the techniques for solving LAEs (Sections III and IV) represent the basic operations required in advanced linear algebra processors. Speiser [5] has defined the fundamental needs of a modern real-time matrix or systolic processor to include the above operations plus the solution of eigensystems, singular value decomposition (SVD), and least squares solution. In this section, these latter operations, other matrix decomposition algorithms, and various extensions of the basic operations are briefly discussed. Emphasis is given to those methods that are most suitable for parallel implementation on optical systolic processors, to those methods for which an optical systolic processor implementation has thus far been detailed, and to the most stable and preferred algorithms. For reasons of space, all of these discussions must be quite brief. The complexity of different problems and operations is usually described by the number of multiplications and additions required for problems of size N. The problems of most concern are of order N^3. In Table 2, problems of different order, the name given to each, and examples of each are provided.

Table 2 Complexity Measures for Different Problems (Adapted from [5])

Order	Name	Examples
N	Scalar	Inner Product, IIR Filter
N^2	Vector	Linear Transforms, Fourier Transform, Convolution, Correlation, Matrix–Vector Products
N^3	Matrix	Matrix–Matrix Products, Matrix Decomposition, Solutions of Eigensystems or LAEs or Least Squares Problems

In many cases, the system of LAEs is overdetermined (i.e., there are more equations than unknowns). In this case $\boldsymbol{Ax} = \boldsymbol{b}$ is a matrix with M rows and N columns, where $M > N$. The conventional least squares solution to minimize $\|\boldsymbol{Ax} - \boldsymbol{b}\|^2$ results in the classical Gauss–normal equation $\boldsymbol{A}^T\boldsymbol{Ax} = \boldsymbol{A}^T\boldsymbol{b}$. One can solve this as LAEs where the matrix is $\boldsymbol{A}^T\boldsymbol{A}$ (a square matrix) and the vector is $\boldsymbol{A}^T\boldsymbol{b}$. This is not attractive since the new matrix has a condition number that is the square of the one in the original problem. This will significantly increase the effect of any computational errors. Modern signal processing solutions to least squares problems employ matrix decomposition by LU, QR, and SVD methods. If a QR decomposition is performed ($\boldsymbol{A} = \boldsymbol{QR}$), the Gauss–normal equations become $\boldsymbol{x} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}$ or $\boldsymbol{Rx} = \boldsymbol{Q}^T\boldsymbol{b} = \boldsymbol{b}'$, and thus from the $\boldsymbol{R}$ matrix and the $\boldsymbol{b}'$

vector (produced for example in Fig. 9), only the solution of an upper triangular system is required (this can be achieved on the system of Fig. 10 as detailed in Fig. 11). Similar remarks apply to an LU decomposition. In adaptive beam forming (Section VI), calculation of the adaptive weights can be formulated as a constrained least squares problem. In solving this, the constraints are first removed and a conventional least squares problem results which can be solved as above. Such advanced adaptive noise cancellation algorithms using direct least squares techniques are attractive since they provide better convergence than the gradient-based algorithm noted in Section VI. They require the matrix decomposition algorithms described in Section IV and below.

Solutions of eigensystems is a second major problem that arises in modern signal processing algorithms. The newest beam-forming and direction-finding algorithms for high-resolution performance require the solution of a symmetric eigensystem for each resolved temporal frequency [45] for $\boldsymbol{x}$ and λ. The most popular algorithms for eigensystem solutions involve the Jacobian method [46], SVD [67], Householder or Givens transformations [41] (to calculate selected eigenvalues), and the QR algorithm [41]. In the QR algorithm, similarity transformations are applied ($\boldsymbol{A}$ is transformed into $\boldsymbol{B} = \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}$, where $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same eigenvalues and the eigenvectors $\boldsymbol{y}$ of $\boldsymbol{B}$ are related to the eigenvectors $\boldsymbol{x}$ of $\boldsymbol{A}$ by $\boldsymbol{T}\boldsymbol{y} = \boldsymbol{x}$). Using QR decomposition, the matrix $\boldsymbol{Q}$ is calculated such that $\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{D}$ (where $\boldsymbol{D}$ is approximately diagonal, with small off-diagonal elements). This is achieved by successive matrix decompositions and matrix multiplications, i.e., at step m we decompose $\boldsymbol{A}_m = \boldsymbol{Q}_m\boldsymbol{R}_m$ and form a new matrix $\boldsymbol{A}_{m+1} = \boldsymbol{R}_m\boldsymbol{Q}_m = \boldsymbol{Q}_m^T\boldsymbol{A}_m\boldsymbol{Q}_m$. This procedure is repeated recursively until $\boldsymbol{Q}_m^T\boldsymbol{A}_m\boldsymbol{Q}_m$ is approximately diagonal. The final matrix is $\boldsymbol{Q} = \boldsymbol{Q}_0\boldsymbol{Q}_1 \cdots \boldsymbol{Q}_{m-1}$. The high accuracy achieved with such orthogonal transformations makes their general use most attractive.

The optical realization of QR solutions to eigensystems has been detailed [29] and follows directly from Fig. 9 and the above steps. Shift algorithms [41] can be used to greatly reduce the number of matrix multiplications needed. If the matrix $\boldsymbol{A}$ is full and not symmetric, similarity transforms can only reduce $\boldsymbol{A}$ to a Hessenberg matrix (an upper triangular matrix with one additional diagonal below the main diagonal). Standard decomposition methods exist to reduce Hessenberg matrices to tridiagonal form and symmetric matrices to tridiagonal (using the QR algorithm) or bidiagonal (using SVD) matrices. The optical realization of several of these methods have been detailed [29], [47]. The preferable solutions appear to be to use QR techniques to reduce $\boldsymbol{A}$ to tridiagonal form or one-sided SVD techniques to reduce $\boldsymbol{A}$ to bidiagonal form, and to then calculate the eigenvalues of these simplified matrices. Such methods are the subject of active current research [29], [48]–[50]. Such eigensystem solutions are preferable to power methods for which several optical realizations [51], [52] have been described. These represent other fruitful areas for future optical systolic processor research.

Let us next advance several remarks on other triangular and orthogonal matrix decompositions and then briefly discuss SVD. Many triangular factorization techniques are possible besides LU decomposition. These include LDU, LDL^T, and LL^T (optical realizations of each of these are quite straightforward). LU and LDU decomposition require that $\boldsymbol{A}$ be only nonsingular. If $\boldsymbol{A}$ is symmetric and positive-definite (as often occurs in signal processing), then LDL^T and LL^T decomposition are quite attractive since they avoid the need for pivoting in the calculations required to compute the new $\boldsymbol{P}_m$. The optical realization of Cholesky (LL^T or LDL^T) decomposition [53] has been detailed [28] and follows directly from Fig. 7. LDL^T decomposition is the preferable choice in such cases since it avoids the need to form the square root, as required in LL^T decomposition. Orthogonal matrix factorizations are preferable for general matrices since they are numerically stable, since there is no need for pivoting (as can be required in LU decomposition of general matrices), and because fast shifted QR algorithms (for eigensystem solution) exist.

As noted in Section IV, numerically stable QR decomposition can also be achieved by modified Gramm–Schmidt [53] and Givens techniques [53]. The optical realization of these methods can be directly realized by a sequence of matrix multiplications with calculations of the elements of the next transformation matrix required after each matrix product is formed. The Householder technique appears to be the most parallel, stable, and easiest algorithm to realize optically as quantified in Section IV-D. For digital systolic processors, Givens techniques are presently the most popular and attractive ones (this is due to the architectural differences between most digital and optical systolic processors).

As noted earlier, SVD is a powerful and useful technique for least squares, eigensystems, and high-resolution direction-finding problems. Although this is a complicated decomposition algorithm, it provides estimates of the condition number of the matrix and the number of signals present. In SVD, the matrix $\boldsymbol{A}$ is factored into three matrices, $\boldsymbol{A} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{Q}^T$, where $\boldsymbol{P}$ and $\boldsymbol{Q}$ are orthogonal matrices and $\boldsymbol{D}$ is a diagonal matrix. The singular values of $\boldsymbol{A}$ are the elements of $\boldsymbol{D}$. When applied to a least squares problem, the SVD solution $\boldsymbol{x} = \boldsymbol{Q}\boldsymbol{D}^{-1}\boldsymbol{P}^T\boldsymbol{b}$ is easily calculated once the SVD has been performed. Thus far, the only optical realization of SVD described [47] used 2-D modulators in an optical matrix–vector processor. Extension to optical systolic architectures appears to be rather straightforward.

A review of the wealth of linear algebra algorithms in modern signal processing is beyond the scope of this paper. However, the selected algorithms noted above and the selected applications discussed in Section VI provide a good introduction and overview to the role of optical systolic processors in modern signal processing. A summary of attractive solutions for various mathematical problems is provided in Table 3. This table is the compilation of many references [5], [26], [27], [41], [53]. The matrix solutions given are by no means complete and represent what presently appear to be the better approaches for optical systolic processors. Several general signal processing applications are then noted in Table 4. For each, an attractive problem formulation and one candidate solution are noted. As before, various other problem formulations and candidate solutions are possible beyond those listed in Table 4. However, the methods listed appear at present to be among the most attractive ones for realization on optical systolic processors.

Table 3 Attractive Solutions for Various Problems

Problem	Matrix Features	Attractive Solution
Solution of Time-Dependent PDEs [29]	Banded	Finite Differences Implicit or Explicit
Deconvolution [10] (Stationary)	Toeplitz	Time or Space Integrating Processors with Feedback
Solution of Systems of LAEs $\boldsymbol{Ax} = \boldsymbol{b}$	None	Direct by QR
	Diagonally Dominant	Direct by LU
	Stable	Indirect
	Symmetric Positive Definite	Direct by Cholesky
Eigensystem Solution $\boldsymbol{Ax} = \lambda \boldsymbol{x}$	Real-Symmetric or Complex Hermitian Symmetric	Householder Direct Decomposition Iterative QR Algorithm
[41], [53]	Symmetric Nonnegative Definite	Direct SVD Reduction to Bidiagonal
Symmetric Generalized Eigensystem Solutions $\boldsymbol{Ax} = \lambda \boldsymbol{Bx}$ [45]	$\boldsymbol{A}$ and $\boldsymbol{B}$ are Real Symmetric $\boldsymbol{B}$ is Positive-Definite	Unitary Transforms or LL^T
Least Squares Solutions $\|\boldsymbol{Ax} - \boldsymbol{b}\|^2$	$\boldsymbol{R} = \boldsymbol{A}^T\boldsymbol{A}$ is Positive-Definite	Direct Matrix Decomposition
	$\boldsymbol{A}^T\boldsymbol{A}$ is Nonsingular	QR Decomposition
	$\boldsymbol{A}^T\boldsymbol{A}$ is Singular	SVD

Table 4 One Possible Problem Formulation and Solution for Selected Specific Applications

Application	Attractive Problem Formulation	Candidate Solutions	Reference
High Resolution Direction Finding	Symmetric Eigensystem	SVD	[45]
State Estimation	Kalman Filter	Recursive Least Squares (Square-Root Formulation)	[66]
Adaptive Noise Cancellation	Constrained Least Squares	Triangular or Orthogonal Decomposition	[37]

VI. Selected Applications for Optical Systolic Processors

A wealth of physical, signal processing, and control problems require various linear algebra operations and the solutions of diverse matrix equations. Brief discussions of several applications are now advanced. These are drawn from available optical systolic processing literature and are chosen and intended to demonstrate different points and features: 1) solutions of partial differential equations (PDEs) with emphasis on matrix structure and implicit and explicit solutions (Section IV-A), 2) radar and sonar applications with attention to simple adaptive filtering and the need to handle complex-valued data (Section VI-B), and 3) optimal control with attention to the solution of a nonlinear matrix equation on an optical systolic processor (Section VI-C).

A. Solution of PDEs

PDEs are the standard mathematical models for many physical problems and distributed systems in applied mechanics. For steady-state PDEs (e.g., elliptical equations), spatial discretization leads directly to LAEs which can be solved by the indirect (Section III) or direct (Section IV) solution methods noted earlier. Time-dependent PDEs represent another major class of mathematical models. Discretization of such equations can yield implicit or explicit solutions as we now demonstrate [29].

We consider the diffusion equation as a second-order PDE example

$$\partial u(x,t)/\partial t = c^2 \partial^2 u(x,t)/\partial x^2 \tag{30}$$

to be solved for $u(x,t)$ with boundary conditions $u(x,0) = f(x)$ for $0 \leqslant x \leqslant L$ and $u(0,t) = u(L,t) = g(t)$ for $0 \leqslant t \leqslant T$. We discretize time and space into increments $\Delta t = T/(N+1)$ and $\Delta x = L/(J+1)$, and denote discrete points in time and space by $n\Delta t$ and $j\Delta x$. If we apply single differencing in time and space to both sides of (30), we obtain

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = c^2 \left[\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \right] \tag{31}$$

where superscripts denote time increments and subscripts denote space increments. Rearranging (31), we obtain

$$u_j^{n+1} = \lambda u_{j+1}^n + (1 - 2\lambda) u_j^n + \lambda u_{j-1}^n, \quad \text{for } n \geqslant 0, 1 \leqslant j \leqslant J \tag{32}$$

where $\lambda = c^2 \Delta t/(\Delta x)^2$. An alternate formulation results if we apply double differencing to the space derivative on the right-hand side of (30). In this case, we obtain

$$(1 + 2\lambda) u_j^{n+1} - \lambda u_{j+1}^{n+1} - \lambda u_{j-1}^{n+1} = \lambda u_{j+1}^n + (1 - 2\lambda) u_j^n + \lambda u_{j-1}^n. \tag{33}$$

Let us now consider and compare the use of (32) or (33)

to solve for $u(x,t)$ as $u_j^{(1)}$ (at $t = 1\Delta t$ for all $1 \leq j \leq J$), then $u_j^{(2)}$ (at $t = 2\Delta t$ for all j), etc. From (32), calculation of $u_j^{(n+1)}$ for all j requires a simple matrix–vector multiplication $\boldsymbol{u}^{(n+1)} = \boldsymbol{A}\boldsymbol{u}^{(n)}$, where $\boldsymbol{A}$ is tridiagonal with elements λ, $(1 - 2\lambda)$, and λ along the three diagonals (where $\boldsymbol{u}^{(n)}$ is known from boundary conditions or from the calculations at the prior $t = n\Delta t$ time step). However, for the single-differencing approximation to be a good approximation, Δx must be small, and for stability $0 \leq \lambda \leq 0.5$ is necessary, and thus a large number of very small time steps Δt are needed to produce accurate results. Hence, such explicit solutions (which initially look quite attractive because they require a matrix–vector multiplication to obtain the data at the next time step) can, in practice, require many small Δx samples and many small Δt time steps, and thus a significant number of matrix multiplications.

Let us next consider the Crank–Nicholson algorithm as formulated in (33). In this case, $\boldsymbol{u}^{(n+1)}$ is calculated from $\boldsymbol{u}^{(n)}$, which is known from boundary conditions or calculations at the prior $t = n\Delta t$ time step, by solving the LAEs $\boldsymbol{A}\boldsymbol{u}^{(n+1)} = \boldsymbol{b}^{(n)}$. Thus the explicit solution in (32) requires only a matrix–vector multiplication, whereas the implicit solution in (33) requires the solution of a system of LAEs at each time step. An implicit solution is still attractive because it is unconditionally stable and has second-order accuracy, and because the number of matrix multiplications in the explicit solution may be quite large. Since the matrix is tridiagonal, solutions to such LAEs can become quite simple using, for example, the system of Fig. 2. In many cases, the coefficient c in (30) is constant or is slowly varying with time, and hence so is the matrix $\boldsymbol{A}$. In such cases, matrix decomposition or direct solutions are quite attractive, since the matrix decomposition need be performed only once, thereafter the simplified triangular systems solution can be used with a different exogenous vector. This would require only a matrix–vector multiplication at each time step, as in the explicit algorithm in (32).

Trapezoidal, Runge–Kutta, and other difference approximations in space and time are also possible. These will yield different forms for (32) and (33), but with similar matrix structures and conclusions. In general, all discretizing methods will yield implicit or explicit solutions with banded matrices. Thus many physical problems directly result in matrix–vector problems with quite structured matrices. Deconvolution applications [10] are yet another case when structured matrices result. In such cases, the received signal $s(t)$ is the convolution of the original signal $a(t)$ with the impulse response $h(t)$ of the transmitting medium. In terms of discrete signal samples, $c_m = \sum h_{m-n} a_n$ (where the summation is over the range of sample points). The matrix–vector form is $\boldsymbol{c} = \boldsymbol{H}\boldsymbol{a}$ (where the matrix $\boldsymbol{A}$ is Toeplitz and contains elements h_{m-n}). Thus to recover $\boldsymbol{a}$ given $\boldsymbol{h}$ and $\boldsymbol{c}$ requires the solution of a system of LAEs with a Toeplitz matrix. The Toeplitz matrix structure will exist for linear shift-invariant distortions, and its bandwidth will depend upon the length of the impulse response function. In such cases, the architectures of Figs. 2 and 3 with appropriate feedback as in Figs. 5 or 6 can be employed. A variety of applications thus exist for structured matrix and LAE solutions. The best solution, algorithm, and architecture depend upon the specific problem and application. However, implementation methods for the basic algorithms and architectures have been described (Sections II–IV).

B. Simplified Adaptive Noise Cancellation

One of the original motivations and applications for optical matrix–vector processors was adaptive phased-array radar processing [54]. This application [55] introduced the original iterative optical matrix–vector algorithm in (6). We briefly consider the calculations required to obtain the set of adaptive weights $\boldsymbol{w}$ for an adaptive phased array to steer the antenna in a direction defined by the vector $\boldsymbol{s}$ and to null the noise field defined by the covariance matrix $\boldsymbol{M}$. This problem is the basis for much of radar and sonar beam forming. In the simple case of a linear array of N evenly spaced antenna elements, the received signal $v_n(t)$ at antenna element n is multiplied by an appropriate weight w_n. For the full antenna, the output signal for one set of weights is

$$s(t) = \sum_{n=1}^{N} w_n v_n(t).$$

The weights defined by the vector $\boldsymbol{w}$ are chosen to control the antenna beam pattern $E(\theta)$, where θ is the angle at which the beam is steered. In general, $\boldsymbol{w}$ is complex-valued and varies with time in conjunction with the noise environment. $\boldsymbol{w}$ is chosen to null the noise sources within the antenna's field of view at the desired angles and frequencies, and to produce a peak at the desired steering direction. Hence, a least squares formulation is appropriate as noted in Section V.

Various adaptive control loops are possible to achieve this. Their operation and convergence are detailed elsewhere [54]. In vector notation, the dynamic behavior of one type of adaptive control loop and weight vector is described by

$$\tau\dot{\boldsymbol{w}}/G + (\boldsymbol{M} + I\boldsymbol{G})\boldsymbol{w} = \boldsymbol{s}^* \tag{34}$$

where τ is the time constant of a low-pass filter and G is the gain within the control loop. The covariance matrix has elements $m_{ij} = \langle v_i^*(t) v_j(t) \rangle$, where $\langle - \rangle$ denotes a time average. Assuming $G \gg 0$, then in steady state ($\dot{\boldsymbol{w}} = 0$), (34) reduces to $\boldsymbol{M}\boldsymbol{w} = \boldsymbol{s}^*$, and the set of adaptive weights $\boldsymbol{w}$ is given by the solution $\boldsymbol{w} = \boldsymbol{M}^{-1}\boldsymbol{s}^*$ to a set of LAEs. The various algorithms described in Sections III and IV and the various optical systolic architectures in Section II are suitable for solution to such problems. Least squares, SVD, and eigensystem solutions (Section V) are the most attractive present methods for such beam-steering problems using new algorithms noted in Section V.

Matrix inversions will arise in various applications. At this point, we note that the architecture and algorithm depicted in Figs. 9 or 11 can perform matrix inversion if the augmented vector $\boldsymbol{b}$ is replaced by the identity matrix $\boldsymbol{I}$ of order equal to that of $\boldsymbol{A}$. In this case, the system solves the matrix–matrix equation $\boldsymbol{B}\boldsymbol{A} = \boldsymbol{I}$ for $\boldsymbol{B} = \boldsymbol{A}^{-1}$ by a direct algorithm. A parallel iterative algorithm [13] for matrix inversion on the system of Fig. 6 is also possible by modifying the algorithm in (6). To develop this algorithm, we consider the solution of a general matrix–matrix equation $\boldsymbol{C} = \boldsymbol{A}\boldsymbol{B}$ for $\boldsymbol{B} = \boldsymbol{A}^{-1}\boldsymbol{C}$ by a new indirect method. The conventional iterative algorithm is rewritten as

$$\boldsymbol{B}_{i+1} = \omega\left(\boldsymbol{A}'\boldsymbol{B}_i + \boldsymbol{C}\right) \tag{35}$$

where $\boldsymbol{A}' = (\boldsymbol{I}/\omega - \boldsymbol{A})$. Calculation of $\boldsymbol{A}'$ is trivial and (35) is

considerably simpler to implement than the conventional matrix extension of (6).

Extensions of adaptive phased-array radar processing to provide multidimensional adaptation in velocity or time as well as angle or space have also been described [56]. The matrix vectorizing methods discussed in Section VI-C are quite useful in such extensions. Calculation of antenna weights also introduces the issue that the vector will have complex-valued elements. Various methods for handling matrices and vectors with bipolar-valued and complex-valued elements have been described. These include space, time, frequency, or wavelength multiplexing. The most popular general methods for representing complex-valued data are by a four-tuple representation (the positive- and negative-valued real and imaginary parts) or a three-tuple representation (each complex number is represented by its three projections on the 0°, 120°, and 240° axes in the complex plane).

Other classical signal processing operations can also be described as matrix operations, some as vector outer products rather than matrix–vector multiplications. In signal processing applications, the matrix used is generally the covariance matrix. It will be real, symmetric, and nonnegative definite (for real random vectors) or Hermitian symmetric and nonnegative definite (for complex vectors). Hence, different matrix properties will result in different applications, and appropriate decomposition algorithms utilizing the matrix features should be employed. Calculation of the ambiguity function [57] of two signals is a classical signal processing operation. In its discrete form, it can be described [44] as the product of three matrices (one matrix being Toeplitz and another being diagonal). The required cross-ambiguity function can thus be calculated on a matrix processor by performing the indicated triple-matrix product. Detailed algorithms and optical systolic processors for these and other advanced signal processing functions described here and in Section V are the subject of current research.

C. Optimal Control, State Estimation, and Kalman Filtering

State estimation and Kalman filtering applications are among the most demanding ones for which advanced highly parallel optical systolic processors with very high computational rates are needed. The basic operations required in Kalman filtering are well known [58]. They include triple-matrix products, matrix inversions, and the solutions of nonlinear matrix equations [13]. Algorithms and architectures to achieve all of these operations (except the last one) have been described earlier in this paper. We thus now advance a new algorithm for solving nonlinear matrix equations on optical systolic processors. This will enable our repertoire of operations achievable on an optical systolic processor to include all of the basic operations needed for state estimation and Kalman filtering.

For the case when the noise statistics are known, a simple two-channel optical systolic processor design has been advanced for steady-state Kalman filter computations [59]. In the more important case of a fully adaptive Kalman filter, the sequence of operations necessary and the required processor architecture, as well as the flow of operations, is far more complicated. The full solution to this problem for an extended Kalman filter has been detailed [60], [61] for an air-to-air missile guidance controller. In this case, a Newton–Raphson solution was employed to solve the nonlinear matrix equation, and calculation of the Jacobian was achieved by an efficient digital table lookup method or by a new optical systolic processor as described in the references noted. A more general optical systolic method to solve nonlinear quadratic matrix equations is described below [36], [62], [63].

The specific application considered is the solution of a linear quadratic regulator (LQR) problem of modern control engineering, in which the control signals $\boldsymbol{u}(t)$ that minimize a quadratic cost-performance index for the general linear system model

$$d\boldsymbol{x}/dt = \boldsymbol{F}\boldsymbol{x}(t) + \boldsymbol{G}\boldsymbol{u}(t) \tag{36}$$

are desired. The solution is

$$\boldsymbol{u}(t) = -\boldsymbol{K}\boldsymbol{x}(t) \tag{37}$$

where the LQR feedback gain matrix $\boldsymbol{K}$ is computed as

$$\boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{G}^{T}\boldsymbol{S} \tag{38}$$

and the symmetric matrix $\boldsymbol{S}$ is the solution of the algebraic Ricatti equation (ARE)

$$\boldsymbol{S}\boldsymbol{F} + \boldsymbol{F}^{T}\boldsymbol{S} - \boldsymbol{S}\boldsymbol{L}\boldsymbol{S} + \boldsymbol{Q} = 0 \tag{39}$$

where $\boldsymbol{L} = \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^{T}$. Selection of this application was motivated by the availability of all of the necessary matrices for the F100 turbofan jet engine, thus allowing specific quantitative data to be obtained and used. The key step in the calculation of $\boldsymbol{u}(t)$ in (37) is solving the quadratic matrix equation (39) for $\boldsymbol{S}$. Hence, we concentrate on one solution method recently developed [63]. This will result in an optical systolic system realization of earlier algorithms [36], [62] devised for an optical matrix–vector processor using a 2-D light modulator.

The solution $\boldsymbol{S}$ to (39) is devised beginning from the classical Newton–Raphson algorithm. Substituting the ARE into the Newton–Raphson solution, the iterative algorithm

$$\boldsymbol{S}(k)\boldsymbol{F}(k) + \boldsymbol{F}^{T}(k)\boldsymbol{S}(k) = -\boldsymbol{S}(k-1)\boldsymbol{L}\boldsymbol{S}(k-1) + \boldsymbol{Q} \tag{40}$$

results, where k denotes the iteration index and where

$$\boldsymbol{F}(k) = \boldsymbol{F} - \boldsymbol{L}\boldsymbol{S}(k-1). \tag{41}$$

This is known as the Kleinman algorithm [64]. Noting that the right-hand side of (40) is known from the value $\boldsymbol{S}(k-1)$ at the prior $(k-1)$ iteration, we see that (40) is linear in $\boldsymbol{S}$. the Kleinman algorithm has thus converted the nonlinear quadratic matrix equation in (39) into the linear equation in $\boldsymbol{S}$ in (40). We also note that (40) has the form of the Lyapanov equation and that solutions to this equation using the Kronecker or tensor product and vectorization exist [65]. To convert (40) to LAEs, we vectorize the matrix on the right-hand side of (40) by lexographically ordering the matrix elements. The resultant column vector is denoted by $\boldsymbol{y}(k)$. The vectorized column vector associated with $\boldsymbol{S}(k)$ is denoted by $\boldsymbol{x}(k)$. Equation (40) can now be described by the system of LAEs

$$\boldsymbol{H}(k)\boldsymbol{x}(k) = \boldsymbol{y}(k) \tag{42}$$

where $\boldsymbol{H}(k)$ is a matrix with specific block structure as detailed elsewhere.

The steps in solving for the matrix $\boldsymbol{S}$ in the form of the vector $\boldsymbol{x}$ in (42) thus involve at step k: a) evaluation of $\boldsymbol{F}(k)$

in (41), the right-hand side $y(k)$ of (40), and formatting of H in (40), and b) the solution of the LAEs in (42) for $x(k)$. The steps in a) involve simple matrix–vector and matrix–matrix multiplications. For the steps in b), indirect or direct algorithms (Sections III and IV) can be used. For the case when an indirect solution to (42) is used, the solution x for S in (39) can be described by the two-loop iterative algorithm

$$x(r+1,k) = [I - \omega(k)H(k)]x(r,k) + \omega(k)y(k). \tag{43}$$

For a fixed k, (40) is the Richardson algorithm in (6) used to solve (39) for $x(k)$. We use r to denote iterations in the Richardson algorithm solution to the LAEs in (42). $H(k)$ and $y(k)$ are then updated and new LAEs in (42) are obtained. We denote the iterations of the Kleinman algorithm by the index k. The solution described to a quadratic matrix equation thus employs an inner iterative loop (implementing the Richardson algorithm for solutions to LAEs) and an outer loop (implementing the Kleinman algorithm, to update the LAEs). These iterations continue until the solution $x(k+1) \equiv x(k) = x$ is obtained. Direct LAE solutions to (42) and other solutions (more complicated to formulate) to (40) are of course possible. The algorithm described above is one example of the class of operations and algorithms possible on optical systolic processors for advanced signal processing applications.

VII. Discussion, Summary, and Conclusions

The accuracy and performance of any optical processor is always an issue of concern. If the performance of the analog architectures described is not sufficient, they can be extended to digital–optical architectures as noted earlier and elsewhere. In all instances, the error source modeling and performance measures used merit attention. The conventional roundoff error analysis available for many digital linear algebra algorithms is not appropriate for optical systolic processors whose errors (such as spatial nonuniformities in the input, AO cell and detector planes, plus detector noise) are considerably different in nature. Initial modeling of such error sources in optical processors has been accomplished [25], and the results are applicable to both analog or digital optical systolic processors. The appropriate performance measure used is also of concern. This will depend on the purpose of the processor. For general systolic array processors, the average or maximum error in any one element of the computed matrix or vector is one performance measure. In specific applications, different performance measures can be defined. For the LQR example defined in Section VI-C, the accuracy of the closed-loop poles of a controlled system matrix is an appropriate performance measure (since these poles describe the transient response of the closed-loop system). In some cases, such as many adaptive noise cancellation applications, the set of adaptive weights may only need to be computed to 1-percent accuracy or so. Such issues merit attention in both analog and digital optical systolic processors.

The algorithms selected and used can also significantly affect the performance obtained in both analog and digital optical systolic processors. Parallel algorithms are essential, and not all algorithms have yet seen parallel realizations. The specific algorithm used must often be selected to match the specific optical systolic architecture. In all instances, robust and stable algorithms are essential and specific attention should be given to selecting algorithms that do not increase the condition number (and hence the accuracy requirements) of the original problem.

The solutions of LAEs, least-squares problems, and eigensystems are essential problems in signal processing. The major direct and indirect algorithms to solve LAEs were noted. For general-purpose processors, direct algorithms are often preferable since the number of iterations and processing time required is known. For specific applications, indirect algorithms are acceptable. Direct algorithms appear to require more precision at each multiplication step than do indirect solutions; however, they will then also provide more accurate results. In general, it is necessary to employ improved algorithms and attention to specific applications to fully address such issues. Iterative solutions to nonlinear equations, eigensystems, and large matrices are still the preferable and often the only approach.

In this paper, many optical systolic architectures have been reviewed and several architectures detailed. Attention was given to architectures for matrices with specific structure (banded, Toeplitz, and triangular), and to matrices with general structure. The solution of LAEs, least squares problems, and eigensystems were selected as the most fundamental problems. It is quite significant that one optical systolic architecture can achieve all of the basic operations required. Efficient digital systolic architectures have thus far required a new mesh connection for different functions. The use of various indirect and direct algorithms and associated optical systolic architectures to realize each were described and discussed. Several specific applications were detailed to demonstrate the many diverse linear algebra problems and operations that emerge. These included the solution of partial differential equations, adaptive noise cancellation, and the basic operations required in state estimation and Kalman filtering.

The field of optical systolic processors is quite young and active. Many architectures, parallel algorithms, and systems with potentially high computational rates above 10^9 multiplications per second have been suggested. In several instances, prototype systems have been fabricated, and in other instances commercially available architectures are being fabricated. Considerable system fabrication, algorithm, and application-directed research remains. All present indicators promise a bright future for this newest topic in optical computing. Optical systolic array processors achieve the flexibility and general-purpose features (that have escaped prior systems), the accuracy and performance (that have eluded prior approaches to optical computers), and such architectures can be fabricated with available components (at competitive cost, size, weight, and power dissipation specifications).

Acknowledgment

The author sincerely acknowledges his present and prior Ph.D students whose research was summarized and whose work contributed significantly to the present state of the art of optical systolic processors (deserving specific mention are Dr. M. Carlotto, Dr. A. Ghosh, and J. Jackson). He also acknowledges the many fruitful discussions on linear algebra with Prof. C. P. Neuman of the Department of Electrical and Computer Engineering at Carnegie-Mellon University.

References

[1] H. T. Kung and C. E. Leiserson, in *Introduction to VLSI*, C. A. Mead and L. A. Conway, Eds. Reading, MA: Addison-Wesley, 1980.

[2] S. Y. Kung, R. J. Gal-Ezer, and K. S. Arun, "Wavefront array processors: Architecture, language, and applications," in *Proc. Conf. on Advanced Research in VLSI*, P. Penfield, Jr., Ed., MIT, Jan. 1982.

[3] H. M. Ahmed, J.-M. Delosme, and M. Morf, "Highly concurrent computing structures for matrix arithmetic and signal processing," *IEEE Computer*, vol. 15, Jan. 1982.

[4] A. Sameh and D. Kuch, "On stable parallel linear system solvers," *J. ACM*, vol. 25, pp. 81–91, Jan. 1978.
See also D. Heller, "A survey of parallel algorithms in numerical linear algebra," *SIAM Rev.*, vol. 20, pp. 740–777, 1978.

[5] K. Bromley, J. Speiser, and H. Whitehouse, "Signal processing: algorithms, architectures, and applications," SPIE Short Course Notes, Jan. 1984.

[6] M. A. Monahan, K. Bromley, and R. P. Bocker, "Incoherent optical correlators," *Proc. IEEE*, vol. 65, pp. 121–129, Jan. 1977.

[7] J. W. Goodman, A. R. Dias, and L. M. Woody, "Fully parallel, high-speed incoherent optical method for performing discrete Fourier transforms," *Opt. Lett.*, vol. 2, pp. 1–3, Jan. 1978.

[8] D. Casasent and M. Carlotto, "A microprocessor-based fiber-optic iterative optical processor," *Appl. Opt.*, vol. 21, pp. 147–152, Jan. 1982.

[9] H. J. Caulfield, W. T. Rhodes, M. J. Foster, and S. Horvitz, "Optical implementation of systolic array processing," *Opt. Commun.*, vol. 40, pp. 86–90, Dec. 1981.

[10] D. Casasent, "Acousto-optic transducers in iterative optical vector-matrix processors," *Appl. Opt.*, vol. 21, pp. 1859–1865, May 1982.

[11] R. A. Athale and W. C. Collins, "Optical matrix-matrix multiplier based on outer product decomposition," *Appl. Opt.*, vol. 21, pp. 2089–2092, June 1982.

[12] P. Guilfoyle, "Acousto-optic engagement matrix processor," in *Bragg Signal Processing and Output Devices*, B. Markevitch, Ed. (*Proc. SPIE*, vol. 352, Aug. 1982), pp. 2–8.

[13] D. Casasent, J. Jackson, and C. Neuman, "Frequency-multiplexed and pipelined iterative optical systolic array processors," *Appl. Opt.*, vol. 22, pp. 115–124, Jan. 1983.

[14] R. Bocker, H. J. Caulfield, and K. Bromley, "Rapid unbiased bipolar incoherent calculator cube," *Appl. Opt.*, vol. 22, pp. 804–807, Mar. 1983.

[15] R. A. Athale and J. N. Lee, "Optical systems for efficient triple-matrix-product processing," *Opt. Lett.*, vol. 8, pp. 590–592, Nov. 1983.

[16] R. A. Athale, W. Collins, and P. Stilwell, "High accuracy matrix multiplication with outer product optical processor," *Appl. Opt.*, vol. 22, pp. 368–370, Feb. 1983.

[17] R. P. Bocker, S. R. Clayton, and K. Bromley, "Electro-optical matrix multiplication using 2's complement arithmetic for improved accuracy," *Appl. Opt.*, vol. 22, pp. 2019–2021, 1983.

[18] R. Athale, H. Hoang, and J. N. Lee, "High accuracy matrix multiplication with a magnetooptic spatial light modulator," in *Real-Time Signal Processing VI*, K. Bromley, Ed. (*Proc. SPIE*, vol. 431, Aug. 1983), pp. 187–193.

[19] P. S. Guilfoyle, "Systolic acousto-optic binary convolver," *Opt. Eng.*, vol. 23, pp. 20–25, Jan./Feb. 1984.

[20] W. Rhodes and P. Guilfoyle, "Acoustooptic algebraic processing architectures," this issue, pp. 820–830.

[21] R. Athale and J. N. Lee, "Optical processing using outer-product concepts," this issue, pp. 931–941.

[22] D. Casasent and J. Jackson, "Fabrication considerations for acousto-optic systolic processors," *Proc. Soc. Photo Opt. Instr. Eng.*, vol. 465, Jan. 1984.

[23] H. J. Whitehouse and J. M. Speiser, in *Aspects of Signal Processing*, pt. 2, G. Tacconi, Ed. (Proc. NATO Advanced Study Institute). Boston, MA: D. Reidel, 1976, pp. 669–702.

[24] D. Psaltis, D. Casasent, D. Neft, and M. Carlotto, "Accurate numerical computation by optical convolution," in *1980 Int. Optical Computing Conf.*, W. T. Rhodes, Ed. (*Proc. SPIE*, vol. 232, Apr. 1980), pp. 151–156.

[25] *Proc. IEEE* (Special Section on Acousto-Optic Signal Processing), vol. 69, pp. 48–118, Jan. 1981.

[26] D. Casasent and C. Neuman, "An iterative optical processor: Selective survey of operational achievable," in *Proc. NASA Langley Conf. Optical Information Processing*, NASA Langley Conf. Publication 2207 (NTIS), pp. 105–118, Aug. 1981.

[27] C. Neuman and D. Casasent, "Optical systolic solutions of linear algebraic equations," in *Optical Information Processing Conf. II*, NASA Langley Research Center, Hampton, VA, Aug. 1983.

[28] D. Casasent and A. Ghosh, "LU and Cholesky decomposition on an optical systolic array processor," *Opt. Commun.*, vol. 46, pp. 270–273, July 1983.

[29] ——, "Direct and implicit optical matrix-vector algorithms," *Appl. Opt.*, vol. 22, pp. 3572–3578, Nov. 1983.

[30] A. Ghosh and D. Casasent, "Triangular system solutions on an optical systolic processor," *Appl. Opt.*, vol. 22, pp. 1795–1976, June 1983.

[31] J. R. Rice, *Numerical Methods, Software and Analysis*. New York: McGraw-Hill, 1983.

[32] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1962, pp. 56–59.

[33] D. M. Young, *Iterative Solution of Large Linear Systems*. New York: Academic Press, 1971.

[34] D. Casasent, A. Ghosh, and C. Neuman, "Direct and indirect optical solutions to linear algebraic equations: Error source modeling," in *Real-Time Signal Processing VI*, K. Bromley, Ed. (*Proc. SPIE*, vol. 431, Aug. 1983), pp. 201–208.

[35] G. F. Franklin and J. D. Powell, *Digital Control of Dynamic Systems*. Reading, MA: Addison-Wesley, 1980.

[36] C. Neuman, D. Casasent, and R. Baumbick, "An electro-optical processor for the optimal control of the F100 aircraft engine," *Proc. Elec. Opt. Sys. Des.*, pp. 311–320, Nov. 1981.

[37] R. Monzingo and T. Miller, *Introduction to Adaptive Arrays*. New York: Wiley-Interscience, 1980.

[38] J. Goodman and M. S. Song, "Performance limitations of an analog method for solving simultaneous linear equations," *Appl. Opt.*, vol. 21, pp. 502–506, Feb. 1982.

[39] D. Casasent and A. Ghosh, "Optical linear algebra," in *Advances in Optical Information Processing*, G. M. Morris, Ed. (*Proc. SPIE*, vol. 388, Jan. 1983).

[40] *Advances in Optical Information Processing*, G. M. Morris, Ed. (*Proc. SPIE*, vol. 388, Jan. 1983).

[41] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*. Oxford, England: Clarendon, 1965.

[42] G. Strang, *Linear Algebra and its Applications*. New York: Academic Press, 1980.

[43] W. Gentleman, "Least squares computation by Givens transformations without square roots," *J. Inst. Math. Appl.*, vol. 12, pp. 329–336, 1973.

[44] R. P. Bocker, "Optical digital RUBIC (rapid unbiased bipolar incoherent calculator) cube processor," *Opt. Eng.*, vol. 23, pp. 26–33, Jan./Feb. 1984.

[45] R. O. Schmidt, "New mathematical tools in direction finding and spectral analysis," in *Real-Time Signal Processing VI*, K. Bromley, Ed. (*Proc. SPIE*, vol. 431, Aug. 1983), pp. 7–19.

[46] H. R. Schwarz, H. Rutishauser, and E. Steifel, *Numerical Analysis of Symmetric Matrices*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[47] B. V. K. Vijaya Kumar, "Singular value decomposition using iterative optical processors," *Appl. Opt.*, vol. 22, pp. 962–963, Apr. 1983.

[48] A. R. Gourlay and G. A. Watson, *Computational Methods for Matrix Eigenproblems*. London, England: Wiley, 1973.

[49] D. E. Heller and I. C. F. Ipsen, "Systolic networks for orthogonal decompositions, with applications," Rep. No. CS-81-18, Aug. 1981, Dep. Computer Sci., Pennsylvania State Univ. (also in P. Pennfield, in *Proc. Conf. on Advanced Research in VLSI*. Dedham, MA: Artech House, 1981).

[50] F. T. Luk, "Computing the singular value decompositions on the Illiac IV," *ACM Trans. Math. Software*, vol. 6, pp. 524–539, 1980.

[51] H. J. Caulfield, D. Dvore, J. W. Goodman, and W. Rhodes, "Eigenvector determination by noncoherent optical methods," *Appl. Opt.*, vol. 20, pp. 2263–2265, July 1981.

[52] B. V. K. Vijaya Kumar and D. Casasent, "Eigenvector determination by iterative optical methods," *Appl. Opt.*, vol. 20, pp. 3707–3710, Nov. 1981.

[53] G. W. Stewart, *Introduction to Matrix Computations*. New

York: Academic Press, 1973
[54] W. F. Gabriel, "Adaptive arrays—An introduction," *Proc. IEEE*, vol. 64, pp. 239–272, 1976.
[55] D. Psaltis, D. Casasent, and M. Carlotto, "An iterative optical processor for adaptive phased array radar processing," in *Real-Time Signal Processing III*, T. T. Tao, Ed. (*Proc. SPIE*, vol. 180, Apr. 1979), pp. 114–120.
[56] D. Casasent and M. Carlotto, "Multidimensional adaptive radar array processing using an iterative optical matrix-vector processor," *Opt. Eng.*, vol. 21, pp. 814–821, Sept. 1982.
[57] P. M. Woodward, *Probability and Information Theory, with Applications to Radar*. London, England: Pergamon, 1953.
[58] R. Brown, *Introduction to Random Signal Analysis and Kalman Filtering*. New York: Wiley, 1983.
[59] I. Jackson and D. Casasent, "A state estimation Kalman filter using optical processing: Noise statistics known," *Appl. Opt.*, vol. 23, pp. 376–378, Feb. 1984.
[60] D. Casasent, C. Neuman, and J. Lycas, "Optical Kalman filtering for missile control," presented at ICALEO'83, Los Angeles, CA, Nov. 1983.
[61] D. Casasent, C. Neuman, and J. Lycas, "Optical Kalman filtering for missile guidance," *Appl. Opt.*, July 1984.
[62] D. Casasent, C. Neuman, and M. Carlotto, "An electro-optical processor for optimal control," in *Control and Communication Technology in Laser Systems*, K. Yong, Ed. (*Proc. SPIE*, vol. 295, Aug. 1981), pp. 176–183.
[63] D. Casasent and A. Ghosh, "A quadratic matrix solution for optical linear algebra processors," to be published.
[64] D. L. Kleinman, "On a iterative technique for Riccati equation computation," *IEEE Trans. Automat. Contr.*, vol. AC-13, pp. 114–115, Feb. 1968.
[65] N. Puri and W. Gruver, "Optimal control design via successive approximations," in *Proc. Joint Automatic Control Conf.* (Univ. of Pennsylvania, Philadelphia, PA), pp. 335–345, 19[illegible].
[66] T. Kailath, *Linear Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981.
[67] J. C. Nash, *Compact Numerical Methods for Computers*. New York: Halsted Press (Wiley), 1979.

10. DIRECT AND INDIRECT OPTICAL SOLUTIONS TO LINEAR ALGEBRAIC EQUATIONS: ERROR SOURCE MODELING

431-25

DIRECT AND INDIRECT OPTICAL SOLUTIONS TO LINEAR ALGEBRAIC EQUATIONS: ERROR SOURCE MODELING

David Casasent, Anjan Ghosh and Charles P. Neuman

Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

ABSTRACT

Direct and indirect solutions to linear algebraic equations (LAEs) are considered with attention to the use of optical acousto-optic (AO) systolic array processors. Specific attention is given to error sources in one AO systolic processor. A case study of an LAE solution is conducted. The first error source model for an optical systolic array processor is advanced. Using this and digital computer modeling, a direct solution is found to be less sensitive to various optical system error sources than is an indirect solution. Acoustic attenuation is found to be the dominant error source in the AO systolic array processor considered. Related error source remarks on different bipolar data representation schemes and on optical versus digital solutions to a triangular system of equations are also advanced.

1. INTRODUCTION

Optical linear algebraic processors are currently receiving considerable attention [1-16]. These architectures vary from simple optical systems that compute matrix-vector products [1-2] to iterative optical processors [3-4] that solve matrix-vector equations or LAEs. Newer architectures using AO light modulators [5-7] are more attractive and can be fabricated with presently available components. These architectures [5-7] and more advanced ones using 2-D CCD-addressed liquid crystals [8] represent yet another class of optical linear algebra systems known as optical systolic array processors. This paper focuses on only the discussion of one specific architecture. We have selected the frequency-multiplexed AO architecture [7] for our specific case study in this paper. Extensions of this frequency-multiplexed AO architecture have been described for the optical solution of: nonlinear matrix equations [9-10], LAE solutions by matrix-decomposition [11-13] and the solution of the resultant lower or upper triangular system of equations [14].

In this paper, we consider only AO systolic processors and specifically only the frequency-multiplexed optical system (this decision is made because the architecture allows more flexibility in the data format possible and in the operations achievable on the system). In this paper, we concentrate on various possible optical and digital solutions to LAEs. Attention is specifically given to the error sources present in optical systems. This subject has not received attention previously. Other techniques to achieve increased accuracy by encoding of the data to be processed using various methods are not addressed (such architectures generally result in a significant reduction in the number of operations possible per second and in an increased complexity in the output detector array). Similarly, vector-outer product optical processors are not addressed (since they require the readout of an entire 2-D output matrix of data every bit time T_B).

In Section 2, we briefly review the AO frequency-multiplexed architecture and several of the different operations that it can achieve. Attention is given to iterative (or indirect) and direct (specifically matrix-decomposition) solutions to LAEs. When direct techniques are used, the final step required is the solution of a triangular system of equations. In Section 2, we note that this is also possible both optically and digitally. In Section 3, we advance the first error source and component model for an optical systolic array processor using AO devices. In Section 4, we discuss how this model is incorporated into a digital simulator to model and analyze the effects of the different error sources present in such advanced data processors. We also advance initial remarks on the effects of different data encoding schemes for representation of bipolar data (with attention to the effect that optical system and component error sources and noise have on the resultant performance and accuracy). In Section 5, we present initial results obtained for an optical direct and indirect solution of a system of LAEs. We also consider a hybrid optical and digital direct solution to an LAE problem. We quantify the dominant system and component error sources found and the performance and accuracy achievable. Conclusions, guidelines and a summary are then advanced in Section 6.

2. FREQUENCY-MULTIPLEXED AO SYSTOLIC PROCESSOR

The basic frequency-multiplexed systolic AO array processor (SAOP) [7] to be considered is shown schematically in Figure 1. It consists of a linear array of point modulators

imaged through separate spatial regions of an AO cell with the Fourier transform of the resultant data collected on an output linear detector array. The point modulator inputs can be time and space multiplexed and the AO cell inputs can be time and frequency-multiplexed. This enables this system to perform matrix-matrix multiplications with one matrix-vector product (one column or row of a matrix-matrix product) produced in parallel every bit time T_B. The bit time T_B is the time required for the AO cell data to propagate between two spatially adjacent regions of the AO cell. This time T_B also represents the rate at which new data can be fed in parallel to the AO cell and to the linear point modulator input array.

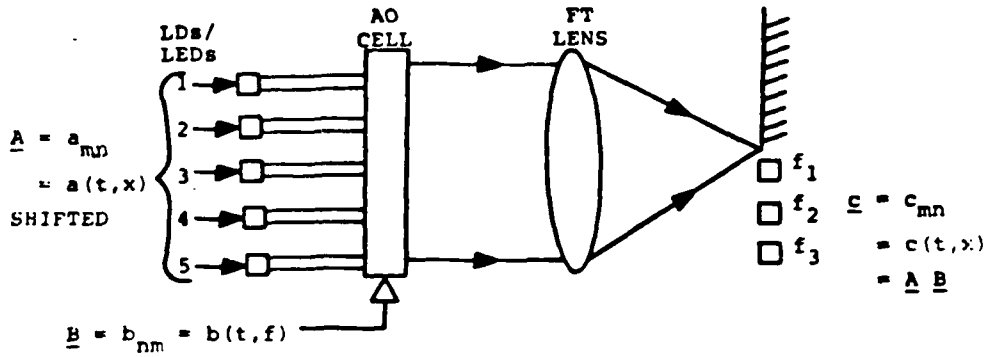


FIGURE 1 Schematic diagram of a frequency-multiplexed AO systolic array processor.

To describe the operation of this system most simply, we consider its use in the calculation of the 3 x 3 matrix-matrix product

$$\begin{matrix} & \text{SPACE} \\ \text{TIME} & \end{matrix}\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \begin{matrix} & \text{FREQUENCY} \\ \text{TIME} & \end{matrix}\begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = AB = C = \begin{matrix} & \text{SPACE} \\ \text{TIME} & \end{matrix}\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}. \tag{1}$$

For now, we consider the case when time and space multiplexing of the rows and columns of $\underline{A}$ and time and frequency-multiplexing of the rows or columns of $\underline{B}$ is performed, as noted in (1). After $3T_B$, the entire $\underline{B}$ matrix is present in the lower $3T_B$ of the AO cell. Point modulator inputs 3-5 are now pulsed on with the first row of $\underline{A}$. The first row of the matrix-matrix product $\underline{A}\,\underline{B} = \underline{C}$ is then produced immediately on the output detector array in parallel. At the next T_B, the data input to the AO cell is shifted up by T_B. We now pulse on point modulators 2-4, with the data input being the next row of the matrix A and immediately obtaining the second row of $\underline{C}$ at the output of this system. This procedure is repeated until all rows of the matrix-matrix product have been produced.

We now briefly describe several other data formats and applications of this basic matrix-matrix or matrix-vector processor architecture. In general, with 2N - 1 LEDs and with an AO cell with a time-aperture $T_A = (2N-1)T_B$ we form N vector inner products on N element vectors every T_B (all in parallel). As we have previously shown [7,12-16], pipelining and the flow of data and operations is quite ideal in this system architecture.

2.1 INDIRECT (ITERATIVE) SOLUTIONS OF LAEs

For an indirect or interative solution to the LAE equation

$$\underline{A}\underline{b} = \underline{c} \tag{2}$$

for

$$\underline{b} = \underline{A}^{-1}\underline{c}, \tag{3}$$

we prefer the iterative Richardson algorithm [4,6,7,15]. In this case, we use the basic optical matrix-vector multiplication system in Figure 1 in conjunction with a parallel analog adder and feedback of the output directly into the AO cell. This configuration, described in [7], realizes the iterative algorithm

$$\underline{b}(j+1) = \underline{b}(j) - \omega\,\underline{A}\underline{b}(j) + \omega\underline{c}, \tag{4}$$

where j denotes the iterative index or time-step, and where ω is the acceleration parameter.

which is selected as described in [4]. When $\underline{b}(j+1) = \underline{b}(j)$, equation (4) converges to (2) and the output $\underline{b}$ is the desired solution in (3). To achieve this [7], we frequency-multiplex the rows of $\underline{A}$ and time-sequentially multiplex the columns of $\underline{B}$. As we will see later, recycling of $\underline{A}$ within an AO cell of length $T_A = NT_B$ is preferable to the use of a longer AO cell length $T_A = (2N-1)T_B$.

2.2 BANDED MATRIX AND TRIANGULAR SYSTEM SOLUTION

If we feed the vector b output back to the AO cell and if one row of $\underline{A}$ is fed to the LEDs in parallel at one time [14], then the same architecture in Figure 1 is ideal for the solution of banded matrix problems and a triangular LAE solution.

2.3 DIRECT SOLUTIONS (MATRIX-DECOMPOSITION)

If we frequency-multiplex the columns (rather than the rows) of the matrix $\underline{A}$ and if we feed one column (rather than one row) of the matrix $\underline{B}$ to the LEDs in parallel, then we form the matrix-matrix product $\underline{B}\,\underline{A}$ at the output (rather than the matrix-matrix product $\underline{A}\,\underline{B}$) [7]. This data encoding approach is ideal [12,13] for matrix-decomposition algorithms (the basic step in a direct solution of an LAE). In [12] and [13], we detail this matrix-decomposition procedure for the cases of LU (Gauss elimination), QR, and Cholesky decompositions.

2.4 DATA FLOW

Moreover, we showed earlier that the pipelining and data flow in such an approach is attractive (the same remarks apply to the indirect algorithms described in Section 2.1). Specifically, every bit time T_B, one time-slot of data leaves the AO cell and a new time-slot of data must be entered into the AO cell. With the aperture time T_A of the AO cell properly chosen for a given problem, we find that the parallel output detected data can be operated upon and fed back immediately to the AO cell input. Thus, in the realization of all of the algorithms we describe, the output data are immediately fed back into the system as they are produced.

2.5 MATRIX INVERSION

The data encoding in Section 2.1 is also appropriate to allow matrix inversion on this system. This aspect of this processor was fully detailed in [7]. It is thus not discussed in further detail here.

3. ERROR SOURCE MODEL

Our component error source model is summarized in Table 1. We consider calculation of the matrix-vector product $\underline{A}\,\underline{b} = \underline{c}$. We separate all component errors into: input plane, AO cell plane and detector plane errors, and we denote each by a separate superscript as noted in Table 1. We denote the spatial coordinates of the input plane and the AO cell by the subscript i and the frequency coordinate of the AO cell and the output detectors by the subscript j.

ERROR SOURCE	NOTATION
Spatial Errors	Subscript i
Frequency Errors	Subscript j
Input Plane Errors	Superscript 1
AO Cell Errors	Superscript 2
Detector Plane Errors	Superscript 3
INPUT PLANE ERRORS	
Point Modulator	
Spatial Gain	$1 + \delta_{i1}^{(1)}$
Nonuniform Response	$1 + \delta_{i2}^{(1)}$
Coupling (Spatial)	$1 + \delta_{i3}^{(1)}$

ERROR SOURCE	NOTATION
AO CELL PLANE ERRORS	
Amplifier Errors	$1 + \delta^{(2)}$
Spatial Response	$1 + \delta_i^{(2)}$
AO Transfer Function	$H(f_j)$
Acoustic Attenuation	$\exp(-\alpha x)$
DETECTOR PLANE ERRORS	
Spatial Response	$1 + \delta_j^{(3)}$
Dark Current	d_j
Time-Varying Noise	$n_j(t)$

TABLE 1. SAOP ERROR SOURCE MODEL

For the input plane, we note that the light intensity incident on the AO cell with all errors included can be described by the factor

$$\hat{b}_i = b_i\left(1 + \delta_{i1}^{(1)} + \delta_{i2}^{(1)} + \delta_{i3}^{(1)}\right), \tag{5}$$

where $\underline{b}$ is the point-modulator vector input data. Similarly, the space i and frequency j transmittance of the AO cell for the matrix element a_{ji} is described by

$$\hat{a}_{ji} = a_{ji}\left(1 + \delta^{(2)} + \delta_i^{(2)}\right)H(f_j)\exp(-\alpha x_i). \tag{6}$$

Likewise, the actual detector plane output $\hat{s}_j$ (the observed output) is related to the exact s_j value and the other error source parameters by

$$\hat{s}_j = s_j(1 + \delta_j^{(3)}) + d_j + n_j(t). \tag{7}$$

Combining all of these factors in (5) - (7), we note that in general all of the error sources must be small (this is realistic and necessary to obtain reasonable accuracy in such a processor). In this case, we can describe the observed detector output $\hat{e}_j$ in terms of the exact inputs a_{ji} and b_i as

$$\hat{e}_j = \sum_i a_{ji}b_i(1+\psi_i)(1+\delta_j)H(f_i)e^{-\alpha x_i} + d_j + n_j(t), \tag{8}$$

where

$$\psi_i = \delta_{i1}^{(1)} + \delta_{i2}^{(1)} + \delta_{i3}^{(1)} + \delta^{(2)} + \delta_i^{(2)}. \tag{9}$$

For a (2 x 2) matrix, the observed outputs $[\hat{e}_1, \hat{e}_2]^T$ are related to the various component error sources in Table 1 by

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1+\delta_1^{(3)} & 0 \\ 0 & 1+\delta_2^{(3)} \end{bmatrix}}_{\text{DET SPAT ERRORS}} \underbrace{\begin{bmatrix} H(f_1) & 0 \\ 0 & H(f_2) \end{bmatrix}}_{\text{AO FREQ RESP}} \underbrace{\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}}_{\text{DATA MATRIX}} \underbrace{\begin{bmatrix} 1+\psi_2 & 0 \\ 0 & 1+\psi_1 \end{bmatrix}}_{\text{SPAT ERRORS INPUT}} \underbrace{\begin{bmatrix} e^{-\alpha x_2} & 0 \\ 0 & e^{-\alpha x_1} \end{bmatrix}}_{\text{AO ATTN}} \underbrace{\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}}_{\text{DATA } \underline{b}} + \underbrace{\begin{bmatrix} d_1 \\ d_2 \end{bmatrix}}_{\text{DET DARK I}} + \underbrace{\begin{bmatrix} n_1(t) \\ n_2(t) \end{bmatrix}}_{\text{DET NOISE}} \tag{10}$$

If we assume that the acoustic attenuation α is small, then (10) reduces to

$$\begin{bmatrix} \hat{e}_1 \\ \hat{e}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} c_1 \\ c_2 \end{bmatrix}}_{\text{EXACT}} + \underbrace{\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix}}_{\text{SPATIAL}} + \underbrace{\begin{bmatrix} n_1 \\ n_2 \end{bmatrix}}_{\text{TEMPORAL}}. \tag{11}$$

This latter formulation in (11) is attractive because it shows that spatial and temporal errors can be separated in such an AO systolic processor. This is useful, since all spatial errors can then be reduced to any desired level by applying the associated fixed correction factors to the input point light-modulator input and to the output detector elements. In closing our remarks on error sources, we noted that the new spatially-multiplexed bipolar-data representation scheme we advanced earlier [7] is very attractive since it does not result in the magnification of system errors. The various biasing and scaling techniques previously proposed to accommodate bipolar data in such an optical matrix-vector processor result in a magnification of any residual system errors (by a factor equal to the dynamic range of the matrix). In general, such errors rapidly become quite intolerable.

4. SIMULATION OF SAOP ERROR SOURCES

To determine the dominant error sources, to quantify the degree to which the various error sources must be reduced, and to quantify the performance to be expected as a function of all

of the various system parameters, digital simulation techniques are essential and were employed. In this section, we discuss several of the details associated with our digital modeling and simulation of the SAOP system error sources and model noted in Section 3. From (10), we note that the SAOP system and component errors are multiplicative and are a matrix cascade. This is distinguished from the error source analysis and modeling we conducted for the fixed-mask iterative optical processor (IOP) system. In the case of the IOP system, we found the error sources of this architecture to be additive [10], rather than multiplicative. We also note that for the matrix-matrix multiplication required in the LU decomposition (one approach to the direct solution of LAEs), the matrix cascade of errors is reversed since row-wise multiplication (rather than column-wise multiplication) is employed (see Section 2). However, the same basic results are expected for both systems. We also note that we assume (in our analysis) that the residual spatial errors (for the input, AO cell and detector system) are reduced to a significantly low level, but are present even after correction. Our intent is to quantify the amount to which correctable spatially-fixed errors must be reduced and the amount of time-varying detector noise and acoustic attenuation that is allowable in such a processor.

In our digital modeling, we represent residual spatial errors (input, AO, and detector plane) by Gaussian random variables with 3σ maximum deviations equal to the fractional residual error remaining after corrections. These residual errors are included as fixed-multiplicative factors that we apply to the point-source inputs and the detector outputs at each matrix-vector multiplication. Detector plane temporal errors (noise versus dark current spatial variations) are also modeled by similar Gaussian random variables applied to each vector output produced on the detectors. However, a different seed-value is used to produce uncorrelated noise that is added to each matrix-vector output product at each T_B time to appropriately model detector system time-varying noise. This approach models the time-varying detector noise and distinguishes the fixed spatial errors from the time-dependent noise errors. Acoustic attenuation effects are handled by directly including the necessary exponential attenuation factor into the input data to the AO cell (and the associated transmittance of the AO cell). Acoustic attenuation is dispersive. However, our initial tests included only a fixed attenuation α which can thus be transferred to the point-modulator input plane (and subsequently corrected to the degree necessary).

As performance measures, we use three quantities. First, the Euclidean norm of the error in the calculated vector $\underline{b}$ is used; i.e.,

$$||\Delta b||_{avg} = [\sum_i (\hat{b}_i - b_i^*)^2]^{1/2}. \tag{12}$$

This error measure corresponds to the average error in the calculated vector output $\underline{\hat{b}}$. If we divide $||\Delta b||$ by the norm of the exact $\underline{b}^*$ vector, then

$$||\Delta b||_{avg}^{\%} = \{100 \times (||\Delta b||/||b^*||)\} \tag{13}$$

defines the average percent error in the elements of the calculated vector. In (12) and (13), $||(\ )||$ denotes the Euclidean norm of the corresponding vector, $\underline{\hat{b}}$ denotes the measured value of the vector, and $\underline{b}^*$ denotes the exact value of the associated vector.

The second performance measure we use is the maximum error in any single element of a calculated vector. This corresponds to a very worst-case error. This performance measure is described analytically by

$$|\Delta b|_{max}^{\%} = \max_i \{100(\hat{b}_i - b_i^*)/b_i^*\}. \tag{14}$$

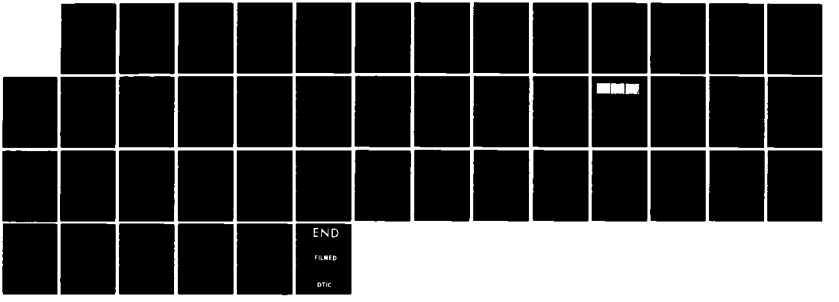
This error measure is an extremely worst-case one. The final error-measure we considered was the maximum error in the closed-loop poles of the resultant system. This error measure is simply defined by

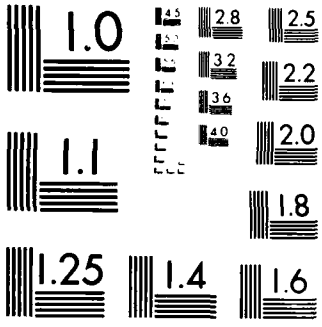
$$|\Delta\lambda|_{max}^{\%} = \max_i \{100(\hat{\lambda}_i - \lambda_i^*)/\lambda_i^*\}, \tag{15}$$

where $\hat{\lambda}_i$ denotes the calculated poles and λ_i^* denotes the location of the exact pole values. This particular error-source measure is most appropriate for optimal control applications. It is also most appropriate to provide a specific case study and application of a situation in which a large error in one element of the vector output does not appreciably affect the net performance of the system.

In general, different performance measures are appropriate for different problems and applications. Attention to the worst-case element error in (14) is not an appropriate measure of an optical LAE solution for many specific cases and applications. When the applica-

AD-A149 346 OPTICAL DATA PROCESSING FOR MISSILE GUIDANCE(U) 2/2
CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF ELECTRICAL
AND COMPUTER ENGINEERING D CASASENT 21 NOV 84
UNCLASSIFIED AFOSR-TR-84-1162 AFOSR-79-0091 F/G 17/7 NL





tion can be specifically defined, other performance measures are more appropriate than this worst-case one. For certain specific case-studies and applications, we note the fact that the following performance measures are appropriate for the various indicated applications: locations of the closed-loop poles of this system (this is appropriate for control applications), SNR (this is appropriate for adaptive filtering applications), and symbol error rate (this is appropriate for communication applications). For our present studies, we use the three performance measures noted above: (12) [average error], (14) [maximum percentage error in any element of the computed vector], and (15) [the maximum percent error in the location of the closed-loop poles of this system].

5. INITIAL EXPERIMENTAL RESULTS

The purposes of our initial simulations were: (1) quantification of the amount of allowable residual spatial errors in the input and detector planes (these are entered as percentages in our table); (2) quantification of the amount of allowable time-varying detector noise (this time-varying detector noise parameter is also entered as a percentage in our tables of data presented); and (3) quantification of the amount of AO cell acoustic attenuation possible (this is entered in units of dB/cm in our tables of data. Only the results for 0.1dB/cm or approximately 6% spatial errors across the AO cell are shown and considered in the data presented. This was necessitated by the fact that the use of larger α values yielded unacceptable performance and no convergence for the algorithm in many instances). Data for the three performance measures in (12), (14) and (15) are given in the tables.

The specific LAE solution case-study used arose from the final outer-loop solution of an LQR (linear quadratic regulator) design with the algebraic Ricatti equation for an F100 aircraft with three states and three controls [9,10]. This situation corresponds to an LAE of order N = 9. The matrix associated with this matrix-vector solution has no specific structure and is essentially full. It is characterized by a condition number C = 2.48 and a dynamic range = 47.7. The acceleration parameter $\omega = 1/\lambda_{max} = -0.044$ was selected as we described earlier [4,6]. For this specific problem $||\underline{b}^*|| = 0.46$ is the average output plane value. This parameter can be used to express the average error $||\underline{\Delta b}||$ as a percentage of the indicated system performance.

Our tests were intended to quantify the component performance of an indirect (or iterative) solution using the Richardson algorithm and a direct solution (using LU decomposition). For the iterative solution, we used J = 10 iterations. This value was determined from four times the condition number of the matrix as described in [16].

TABLE 2. ERROR SOURCE EFFECTS IN AN INDIRECT LAE SOLUTION

TEST NUMBER	RESIDUAL SPATIAL ERRORS (%) INPUT	DET	TIME-VARYING DETECTOR NOISE (%)	AO CELL ATTENUATION dB/cm	ERROR MEASURES $\|\|\underline{\Delta b}\|\|$	$\|\underline{\Delta b}\|_{max}$ (%)	$\|\Delta\|_{max}$ (%)
1	0	0	0	0	0.28×10^{-3}	0.39	0.26×10^{-3}
2	0	0	1	0	0.71×10^{-3}	2.8	0.52×10^{-2}
3	1	1	0	0	0.37×10^{-2}	1.7	0.27
4	1	1	1	0	0.39×10^{-2}	1.36	0.26
5	1	1	1	0.1	0.16	61.8	15.8
6	0	0	0	0.1	0.16	60.4	16.1

In Table 2, we show the results of an indirect solution for six different sets of system and component errors. The error and noise-free results in Test 1 were obtained with 36-bit digital accuracy. As seen, excellent accuracy was obtained in these experiments (the performance obtained is limited by the finite word-length and number of iterations performed). Test 2 shows the effects of 1% detector noise error-source alone. The accuracy obtained is better than 0.1% even though one element of the matrix-vector output was in error by 2.8%. The effects of 1% spatial input errors and 1% spatial output errors (Test 3) alone show that better than 1% accuracy is still possible on such a system. However, the maximum error in one element of the computed matrix is 1.7%. In Test 4, both spatial errors and detector

noise were present. The results of these tests confirmed the implications advanced by our earlier findings on similar tests performed on our IOP optical matrix processor. Specifically, we found that the presence of both spatial and temporal errors did not appreciably affect the accuracy obtained in this system (compared to the case when only one type of error was present). In Test 6, we used the value $\alpha = 0.1$dB/cm for the acoustic attenuation present. From the results obtained and from the results in Test 5 (when acoustic attenuation and all other error sources are present), we see that acoustic attenuation is the dominant error source effect and that the α value used must be significantly reduced if we are to obtain adequate performance from such a matrix-vector processor.

TABLE 3. ERROR SOURCE AND NOISE EFFECTS ON DIRECT AND INDIRECT AND OPTICAL AND DIGITAL TRIANGULAR SYSTEM SOLUTIONS TO LAEs.

TEST NO. (BACK SUBST)	RESIDUAL SPATIAL ERRORS (%)		TIME-VARYING DETECTOR NOISE (%)	AO CELL ATTENUATION dB/cm	ERROR MEASURES		
	INPUT	DET			$\|\|\Delta b\|\|$	$\|\Delta b\|_{max}$ (%)	$\|\Delta\|_{max}$ (%)
1 INDIR	0	0	0	0	0.28×10^{-3}	0.39	0.26×10^{-3}
2 DIRECT	0	0	0	0	0.11×10^{-5}	0.18×10^{-2}	0.10×10^{-4}
? (VAX)	0	0	1	0	0.8×10^{-3}	1.27	0.88×10^{-2}
4 (SAOP)	0	0	1	0	1.0×10^{-3}	1.47	0.22×10^{-1}
5 (SAOP)	1	1	0	0	1.0×10^{-2}	6.0	0.34
6 (SAOP)	1	1	0	0.1	1.0×10^{-2}	8.8	0.39
7 (SAOP)	1	1	1	0.1	1.1×10^{-2}	9.2	0.37
8 (VAX)	1	1	1	0.1	1.4×10^{-2}	8.7	0.57

In Table 3, we compare direct and indirect solutions to LAEs. The error and noise-free results in Tests 1 and 2 show that better accuracy appears to be obtainable with a direct solution. However, this is misleading since, if the number of iterations J were increased to 50, then both algorithms would yield similar error and noise-free performance. In Tests 3 through 8, we included various amounts of spatial errors, temporal noise, and acoustic attenuation. As seen, a direct algorithm yields better accuracy and performance than an indirect algorithm. Specifically, 0.5-2% accuracy is obtained (the maximum error in one element is 8-9%) even with 1% input spatial error, 1% output spatial error, 1% detector noise and 0.1dB/cm acoustic attenuation all present. We also note from this data that in a direct solution, acoustic attenuation is no longer necessarily the dominant error source. Further tests on various implementations of the direct LAE solution were also conducted and are included in Table 3. These involve performing the matrix decomposition optically followed by the solution of the resultant triangular system of equations digitally to 36-bit accuracy (this is denoted by VAX in parentheses in Table 3) or optically using our triangular system solutions algorithm [14] (this is denoted by SAOP in parentheses in Table 3). Comparing the results of the VAX tests (3 and 8) and the SAOP tests (4-7), we find negligible difference in performance whether the triangular system was solved optically or digitally. This is expected due to the nature of the simpler vector inner product calculations required in a triangular system solution.

6. SUMMARY, CONCLUSION AND GUIDELINES

In this paper, attention was given to one optical systolic array architecture, the frequency-multiplexed SAOP. The flexibility possible in formatting data in this architecture was noted together with examples of how the same architecture can be used for many different

operations. This flexibility plus the reduced component requirements (fewer point light modulators and detectors and lower per-channel data rates are required to achieve performance comparable to other architectures) appeas s to make this system more attractive than others. Hence, we restricted attention to it. Extensions of this basic architecture using multi-channel AO cells are also obvious and direct. In this paper, we also advanced the first component error-source model for an optical systolic processor and we noted that many system and component errors are spatially-fixed and hence are correctable. To quantify the level to which various errors and noise must be reduce" and to quantify the performance expected, we detailed the digital model for the SAOP architecture and its system and component error and noise sources. By simulation, we tested an indirect algorithm solution and found that acoustic attenuation was the dominant error source, that 1% accuracy could be obtained; but to do this the acoustic attenuation must be below 0.1dB/cm. We compared direct and indirect solutions and found that direct solutions yielded significantly better accuracy and performance. Finally, we compared direct solutions in which the triangular system of equations that resulted was solved optically and digitally. Negligible difference was found if either approach was used. Using a direct solution, accuracy and performance approaching 1% appears to be possible using realistically achievable component quality and detector noise.

Further tests, experimental verification, and more general analyses and trends and quantification for other specific applications is necessary before definitive general conclusions should be advanced. However, qualitative explanations for all of the results obtained have been advanced and thus the trends observed appear to be representative of a general matrix-vector LAE problem solution.

ACKNOWLEDGMENT

The support of various portions of this research by NASA Lewis Research Center (Grant Number NAG 3-5), the Air Force Office of Scientific Research (Grant AFOSR 79-0091) and by private contractors of Unicorn Systems Incorporated is gratefully acknowledged.

REFERENCES

1. M. Monahan, et al, Proc. IEEE, 65, 121 (1977).
2. J.W. Goodman, et al, Optics Letters, 2, 1 (January 1978).
3. D. Psaltis, et al, Optics Letters, 4, 348 (November 1979).
4. M. Carlotto and D. Casasent, Applied Optics, 21, 147 (1982).
5. H.J. Caulfield, et al, Optics Communication, 40, 86 (1981).
6. D. Casasent, Applied Optics, 21, 1859 (May 1982).
7. D. Casasent, J. Jackson and C. Neuman, Applied Optics, 22, 115 (January 1983).
8. R. Bocker, et al, SPIE, 388 (January 1983).
9. D. Casasent, et al, SPIE, 295, 176 (August 1981).
10. D. Casasent, et al, Proc. EOSD, 311 (November 1981).
11. D. Casasent and A. Ghosh, SPIE, 388 (January 1983).
12. D. Casasent and A. Ghosh, "LU and Cholesky Decomposition on an Optical Systolic Array Processor", Optics Communication, 46, pp. 270-273 (July 1983).
13. D. Casasent and A. Ghosh, "Direct and Implicit Optical Matrix-Vector Algorithms", Applied Optics (Accepted 1983).
14. A. Ghosh and D. Casasent, "Triangular System Solutions on an Optical Systolic Processor", Applied Optics, 22, 1795 (June 1983).
15. D. Casasent and C. Neuman, Proc. "Optical Data Processing for Aerospace Applications", NASA Conference Publication 2207 (NTIS, Springfield, Virginia, 1981).
16. C. Neuman and D. Casasent, Proc. "Optical Information Processing Conference II", NASA Langley Research Center (August 1983).

11. OPTICAL KALMAN FILTERING FOR MISSILE GUIDANCE

Reprinted from Applied Optics, Vol. 23, page 1960, June 15, 1984

Optical Kalman filtering for missile guidance

David Casasent, Charles P. Neuman, and John Lycas

Optical systolic array processors constitute a powerful and general-purpose set of optical architectures with high computational rates. In this paper, Kalman filtering, a novel application for these architectures, is investigated. All required operations are detailed; their realization by optical and special-purpose analog electronics are specified; and the processing time of the system is quantified. The specific Kalman filter application chosen is for an air-to-air missile guidance controller. The architecture realized in this paper meets the design goal of a fully adaptive Kalman filter which processes a measurement every 1 msec. The vital issue of flow and pipelining of data and operations in a systolic array processor is addressed. The approach is sufficiently general and can be realized on an optical or digital systolic array processor.

I. Introduction

A multitude of optical systolic array processors[1-5] have recently been proposed. These processors comprise a broad class of optical linear algebra processors. Numerous engineering applications of these processors have been described, including adaptive phased array radar,[6] optimal control,[7,8] and Kalman filtering.[3,9] In this paper, we detail the realization of a discrete-time extented Kalman filter (EKF) for air-to-air missile guidance using optical systolic array processors. This application provides a specific case study of the use of an optical systolic linear algebra processor in a full problem application. This case study leads to a novel discrete-time EKF algorithm with sufficient parallelism for realization on an optical or digital systolic array processor. Our approach results in a novel algorithm and novel operations that are possible on optical systolic processors. We realize an EKF because the missile and target are modeled by linear differential equations in Cartesian coordinates, whereas the measurement model is nonlinear. Linearizing the nonlinear measurement equation about the most recent relative motion estimates results in an EKF. Discretization of the continuous-time Kalman filter leads to a novel discrete-time algorithm. Such a discrete-time algorithm is essential for realization on optical or digital systolic processors.

We note recent publications on systolic Kalman filters, which appeared as our work was reaching completion. A steady-state analysis of finite word-length effects, roundoff-error propagation, stability and estimation sensitivity is detailed in Ref. 10 for a systolic Kalman filter architecture. Our fully adaptive optical systolic Kalman filter (which processes a measurement every 1 msec) incorporates the automatic updating of the Kalman filter gain and covariance matrix of the error of state estimation and thus differs appreciably from this work. Extended Kalman filter algorithms for optical implementation are proposed in Ref. 11, but implementation details are not provided. In this paper, we detail the design and realization of a complete discrete-time EKF optical systolic array processor.

A. Motivation

Proportional navigation guidance (PNG)[12] is the traditional guidance law used for air-to-air missiles. In this controller, noisy measurements of the target's position and velocity are fed to the PNG computer, which estimates the line-of-sight rate and calculates the missile acceleration for the steering autopilot, which is then applied to the missile's actuators (the fins) to control the missile's position and velocity. These estimates are fed back to the PNG computer, new target measurements are taken, and the process is repeated. Of the PNG assumptions, removing the assumption of a constant relative missile-to-target velocity provides the largest improvement,[13] especially for the case of evasive targets. For advanced guidance laws to be practical, enhanced target motion estimates are required. Modern filtering algorithms, such as the Kalman filter, can provide such estimates. The Kalman filter provides the optimum estimate (in a least-mean square or maximum likelihood sense). Such algorithms use the kinematics and dynamics of the missile and the target,

The authors are with Carnegie-Mellon University, Department of Electrical & Computer Engineering, Pittsburgh, Pennsylvania 15213.

Received 20 January 1984.
0003-6935/84/121960-07$02.00/0.

plus the statistics of the noise in the measurements and in the dynamic process disturbances.

B. Overview

In Sec. II, we highlight the missile–target and measurement models, and review the conventional continuous-time Kalman filter and EKF formulations. A novel discrete-time EKF is then introduced (in Sec. III) and the linear algebra operations required in each step are defined. The major operation required is the solution of a quadratic matrix equation. In Sec. IV, we review optical systolic processors with attention to one specific architecture[3] and to the variety of achievable operations by format control. A new optical system solution to a quadratic matrix equation is then advanced in Sec. V and the required operations are noted. These include a new optical systolic system for calculation of the Jacobian matrix. In Sec. VI, the realization of all operations required in our EKF is summarized, and the load time and calculation time for each step in our algorithm are detailed. Our full system architecture is advanced, the critical time path is isolated, and the processing time required for our EKF is quantified. Our summary and conclusions are then advanced in Sec. VII.

II. Continuous-Time EKF for Dynamic Systems

In this section, we highlight the missile–target and measurement models and review the continuous-time Kalman filter and EKF formulations.

A. Missile–Target Model

The linear dynamic system model for the missile and target is described by the matrix–vector differential equation

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{u}(t) + \mathbf{w}(t), \tag{1}$$

where $\mathbf{x}(t)$ = (9 × 1) missile–target state vector (the state variables are the Cartesian coordinates of the relative target-to-missile position and velocity and the target acceleration);
$\mathbf{F}$ = (9 × 9) missile–target state matrix;
$\mathbf{u}(t)$ = (9 × 1) missile acceleration control vector; and
$\mathbf{w}(t)$ = (9 × 1) missile and target dynamic disturbance vector; $\mathbf{w}(t)$ is modeled as a zero-mean Gaussian white-noise uncorrelated vector with covariance matrix $\mathbf{Q}$, i.e., $\mathbf{w}(t) \sim N[\mathbf{0},\mathbf{Q}]$.

B. Target Measurement Model

We assume that a passive sensing system estimates the elevation (ϕ) and azimuth (θ) of the target.[14] These polar coordinates are related to the relative Cartesian position coordinates by a nonlinear transformation. We denote the relationship between the measured quantities and the relative spatial coordinates by the elements h_1 and h_2 of a vector $\mathbf{h}$. The target measurement model is thus described by the nonlinear vector algebraic equation

$$\mathbf{z}(t) = \mathbf{h}[\mathbf{x}(t)] + \mathbf{w}_s(t). \tag{2}$$

We model the sensor noise vector $\mathbf{w}_s(t)$ by a zero-mean Gaussian white-noise vector with covariance matrix $\mathbf{R}$. In Eq. (2) we note that $\mathbf{z}(t)$ is the measurement; $\mathbf{h}[\mathbf{x}(t)]$ is nonlinearly related to the state vector $\mathbf{x}$ (since Cartesian coordinates rather than polar coordinates are used). The angles ϕ and θ are the directly measurable quantities (because of the sensors used and the techniques available). We chose Cartesian coordinates since the target–missile model is linear and its propagation is easier to realize. Since the measurement model in Eq. (2) is nonlinear, we linearize the nonlinear function $\mathbf{h}[\mathbf{x}]$ by the matrix–vector product $\mathbf{H}(t)\mathbf{x}(t)$, where $\mathbf{H}$ is the gradient of $\mathbf{h}$. This approximation leads to the EKF (Sec. III).

C. Continuous-Time Extended Kalman Filter

The objective of the extended Kalman filter (EKF) is to produce an estimate $\hat{\mathbf{x}}(t)$ of $\mathbf{x}(t)$ for each ϕ and θ measurement $\mathbf{z}(t)$. We linearize $\mathbf{h}(\mathbf{x})$, about the most recent estimate $\hat{\mathbf{x}}(t)$ of the relative spatial coordinates, and approximate $\mathbf{h}[\hat{\mathbf{x}}(t)]$ by the matrix–vector product $\mathbf{H}(t)\hat{\mathbf{x}}(t)$, where the gradient matrix is

$$\mathbf{H}(t) \triangleq \left.\frac{\partial \mathbf{h}[\mathbf{x}(t)]}{\partial \mathbf{x}(t)}\right|_{\mathbf{x}(t)=\hat{\mathbf{x}}(t)}.$$

We thus realize an EKF from the conventional KF. The three steps and three equations which define an EKF follow[15]:

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{F}\hat{\mathbf{x}}(t) + \mathbf{K}(t)[\mathbf{z}(t) - \mathbf{h}[\hat{\mathbf{x}}(t)]] + \mathbf{u}(t), \tag{3}$$

$$\dot{\mathbf{P}}(t) = \mathbf{F}\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T + \mathbf{Q}(t) - \mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t)\mathbf{H}(t)\mathbf{P}(t), \tag{4}$$

$$\mathbf{K}(t) = \mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t). \tag{5}$$

The state-estimate equation in (3) propagates the estimate $\hat{\mathbf{x}}(t)$ of the state vector. From Eq. (3) we see that the next state estimate is the weighted sum of the output of the process or system model, plus the innovations process (the difference between the measurement $\mathbf{z}$ and the nonlinear transformation $\mathbf{h}$ of the state vector), plus the control vector $\mathbf{u}$. The second term in Eq. (3) is an estimate of the measurement noise. If the system noise increases, the present measurement is weighted more heavily (i.e., the KF gain $\mathbf{K}$ is large). We note that the feedback of the current state estimate (including the KF gain) is included via the control vector and the iteration implied in Eq. (3). Thus, the system model is updated with the current state-estimate information from the KF.

The error covariance matrix $\mathbf{P}$, which is a measure of the uncertainty of the $\hat{\mathbf{x}}$ estimate, is defined by

$$\mathbf{P}(t) = E\{[\hat{\mathbf{x}}(t) - \mathbf{x}(t)][\hat{\mathbf{x}}(t) - \mathbf{x}(t)]^T\}. \tag{6}$$

Propagation of $\mathbf{P}$ is the major aspect of the KF and is defined by Eq. (6). The KF gain $\mathbf{K}$ is defined by the matrix product in Eq. (5). From Eqs. (4)–(6), we see that $\mathbf{P}$ increases if the system noise ($\mathbf{Q}$) increases (this is logical). Thus, if $\mathbf{Q}$ increases, $\mathbf{K}$ increases and hence we weight the present measurement more. If the measurement noise $\mathbf{R}$ increases, $\mathbf{K}$ decreases, and we

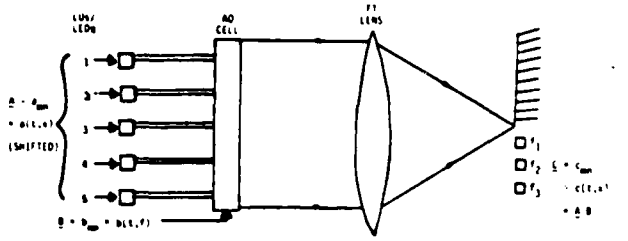


Fig. 1. Schematic diagram of a frequency-multiplexed acoustooptic systolic array processor.[3]

weight the system more heavily. Thus, the next estimate relies on the last estimate (if **R** is large) and relies on the present measurement **z** (if **R** is small).

The operations required in the EKF thus include: calculation of the KF gain in Eq. (5), propagation of the error covariance matrix $\mathbf{P}(t)$ according to Eq. (4), and propagation of the state estimate $\hat{\mathbf{x}}(t)$ according to Eq. (3).

III. Discrete-Time Extended Kalman filter

In this section we develop our new discrete-time EKF algorithm. The need for a discrete-time EKF arises because of the systolic processor realization (which requires pulsed data). Any time-sampled processor (digital or optical) requires a discrete-time formulation. We used the forward Euler method to discretize the state and update equations in (1) and (3) and the trapezoidal rule to discretize the error covariance matrix propagation equation in (4). These algorithms decouple the update equations. The resultant discrete-time EKF algorithm becomes

$$\mathbf{K}_k = \mathbf{P}_k \mathbf{H}_k^T \mathbf{R}_k^{-1} \tag{7}$$

$$\hat{\mathbf{x}}_{k+1} = [\mathbf{I} + T\mathbf{F}]\hat{\mathbf{x}}_k + T\mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k)] + T\mathbf{u}_k, \tag{8}$$

$$\mathbf{P}_{k+1}\mathbf{M}_{k+1}\mathbf{P}_{k+1} + [\mathbf{P}_{k+1}\mathbf{L}^T + \mathbf{L}\mathbf{P}_{k+1}] + \mathbf{C}_k = 0. \tag{9}$$

Our discrete-time EKF algorithm is thus novel and differs from prior applications of discretization schemes[16] (e.g., explicit algorithms and Runge-Kutta methods) to Kalman filtering. In all our equations, the subscript k refers to the time-index or the KF iteration count. Equation (7) defines the KF gain, Eq. (8) characterizes the state-estimate update, and Eq. (9) describes the error covariance update. In Eq. (8), T is the measurement sampling interval. The matrices which we introduced in Eq. (9) are

$$\mathbf{M}_{k+1} = \mathbf{H}_{k+1}^T \mathbf{R}_{k+1}^{-1} \mathbf{H}_{k+1},$$

$$\mathbf{L} = [(1/T)\mathbf{I} - \mathbf{F}], \tag{10}$$

$$\mathbf{C}_k = \mathbf{P}_k \mathbf{M}_k \mathbf{P} - \mathbf{P}_k[(1/T)\mathbf{I} + \mathbf{F}]^T - [(1/T)\mathbf{I} + \mathbf{F}]\mathbf{P}_k - 2\mathbf{Q}.$$

Equations (7)–(10) outline the steps and identify the linear algebraic operations (matrix–vector, matrix–matrix, and matrix–matrix–matrix multiplication) which are required in each iteration to realize our discrete-time EKF. The solution of the symmetric quadratic matrix equation in (9) is the major computational operation required. In Sec. V we detail our new solution to the symmetric quadratic matrix equation in (9) for this application.

IV. Optical Linear Algebra Systolic Array Processors

Numerous optical systolic array processors have been described and some have been analyzed.[1-5] The acoustooptic (AO) optical systolic array processor we chose to detail for this EKF application is the frequency-multiplexed system shown schematically in Fig. 1.[3] The system consists of a linear input array of point modulators, each imaged through a different region of an AO cell, and the Fourier transform (FT) of the light leaving the AO cell formed on the output linear detector array. This specific system was chosen because it is the most documented and analyzed one and because its flexibility leads to the realization of a spectrum of linear algebraic operations by format control of the input data. In this section we summarize the operations heretofore documented that are required for our present EKF application, and we detail how each is realized on this system.

The inputs to the point modulators, light emitting diodes (LEDs) or laser diodes (LDs), are space (x) and time (t), while the inputs to the AO cell are time (t) and frequency (f). The time variable is converted to space as the contents of the AO cell travel across the aperture in time. We achieve the matrix–vector (MV) multiplication $\mathbf{Ab} = \mathbf{c}$ on this system as

$$\begin{matrix} \overset{x,t}{\longrightarrow} \\ \left| \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \right. \\ \underset{f}{\downarrow} \end{matrix} \quad x \left| \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right. = x \left| \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} \right. . \tag{11}$$

AO cell — point modulators — detector outputs

To see how the operations described in Eq. (11) occur, we define the bit time T_B as the time it takes data in the AO cell to move from the region illuminated by the input point modulator N to the region illuminated by the point modulator $N + 1$. We consider a (3×3) matrix example as in Eq. (11). At time $1T_B$, we load the first column of the matrix into the AO cell with each element present on a different frequency (f). At times $2T_B$ and $3T_B$, we load the second and third columns of the matrix into the AO cell. At $3T_B$, the full matrix is present in the AO cell, with its columns opposite point modulators 3, 4, and 5, respectively. At this point, the elements of the vector **b** are fed in parallel to the point modulators 3–5. Each element of **b** multiplies the corresponding columns of the matrix **A**. The output FT sums the proper elements of the product in each row; and on the output detectors, the MV product **Ab** appears in parallel. For an $(N \times N)$ element matrix, the MV product appears in parallel in zero time (after a load time NT_B, during which the matrix is loaded into the AO cell).

Matrix–matrix (MM) multiplication for a (3×3) example is realized as

$$\begin{matrix} \overset{x,t}{\longrightarrow} \\ \left| \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \right. \\ \underset{f}{\downarrow} \end{matrix} \quad x \begin{matrix} \overset{t}{\longrightarrow} \\ \left| \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{21} & b_{22} & b_{23} \end{bmatrix} \right. \end{matrix} = x \begin{matrix} \overset{t}{\longrightarrow} \\ \left| \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \right. \end{matrix} . \tag{12}$$

AO cell — point modulators — detector outputs

This MM multiplication is a direct extension of the MV product in Eq. (11), repeated N times. At each successive T_B of time (after the load time), N vector inner products of N-element vectors are formed in parallel on the N output detectors. A MM product thus requires NT_B of time (plus NT_B of time to load the matrix into the AO cell). One row of the MM product emerges in parallel on the output detectors every T_B time interval.

Matrix–matrix–matrix (MMM) multiplication is a further extension of the MM multiplication in Eq. (12). We realize $\mathbf{ABD} = \mathbf{E}$ by forming $\mathbf{AB} = \mathbf{C}$, feeding $\mathbf{C}$ to the AO cell as it is formed (one row at a time), and then producing $\mathbf{CD} = \mathbf{E}$ one row at a time in parallel. As we have shown,[3] operations and data flow ideally in this architecture (i.e., as one row of the matrix is produced, one T_B of the cell becomes vacant, and we immediately feed the row of the MM product produced to the vacant slot in the AO cell).

Iterative MV algorithms for the solution of linear algebraic equations can also be realized on this system. To solve $\mathbf{Ab} = \mathbf{c}$, we feed one iterate $\mathbf{b}_k$ of $\mathbf{b}$ to the point modulators and $\mathbf{A}$ to the AO cell. We form $\mathbf{Ab}_k$ at the output, subtract $\mathbf{c}$, add $\mathbf{b}_k$ to the result, and feed this sum back to the point modulators as the next $\mathbf{b}_{k+1}$ iterative input. To solve $\mathbf{Ab} = \mathbf{c}$ for $\mathbf{b}$, we thus realize the Richardson algorithm,[3,6,17]

$$\mathbf{b}_{k+1} = \omega(\mathbf{Ab}_k - \mathbf{c}) + \mathbf{b}_k, \tag{13}$$

where ω is the acceleration parameter chosen for stability.[6] When $\mathbf{b}_k \cong \mathbf{b}_{k+1}$, Eq. (13) produces the solution to $\mathbf{Ab} = \mathbf{c}$; i.e.,

$$\mathbf{b} = \mathbf{A}^{-1}\mathbf{c}. \tag{14}$$

The operations in Eqs. (11)–(13) realize all the steps required to implement our discrete-time EKF with the exception of the solution of the symmetric quadratic matrix equation in (9). Our prior approaches to solving a quadratic matrix equation used two iterative loops to implement the Kleinman and Richardson algorithms, respectively.[7] In Sec. V we introduce a new optical solution to the symmetric quadratic matrix problem and detail its implementation. This novel algorithm has many advantageous features compared with those which we previously reported.[7]

V. Optical Systolic Algorithms

In this section, we apply the Newton-Raphson algoroithm to introduce an optical systolic solution to the symmetric quadratic matrix equation in (9) and highlight the efficient calculation of the Jacobian matrix.

A. Optical Systolic Solution to a Quadratic Matrix Equation

The optical solution we introduce to solve the symmetric quadratic matrix equation

$$\mathbf{G}_k = \mathbf{P}_{k+1}\mathbf{M}_{k+1}\mathbf{P}_{k+1} + |\mathbf{P}_{k+1}\mathbf{L}^T + \mathbf{L}\mathbf{P}_{k+1}| + \mathbf{C}_k = 0 \tag{15}$$

for $\mathbf{P}_{k+1}$ is the Newton-Raphson[18] algorithm. We write Eq. (15) as the lexographically ordered vector $\mathbf{g}(p_1 \ldots p_{81}) = 0$, where $\mathbf{p}_n$ contains the $N^2 = 81$ elements of $\mathbf{P}_n$.

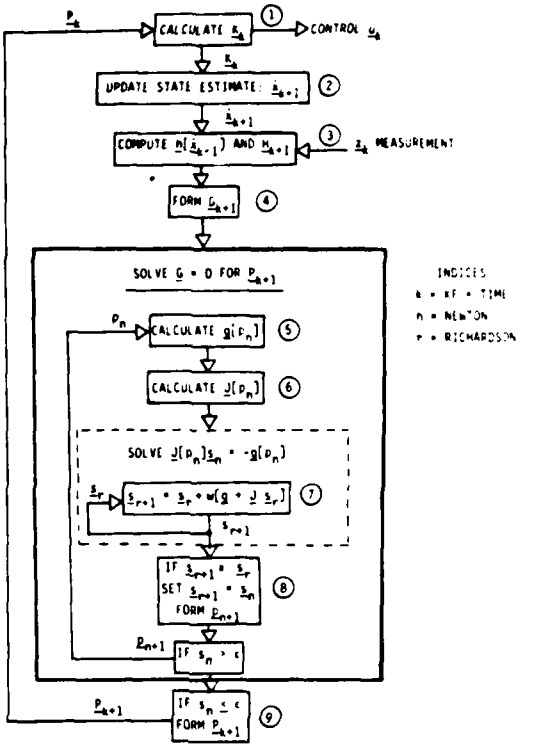


Fig. 2. Triple-nested discrete-time EKF algorithm.

We thus convert Eq. (15) into a system of $N^2 = 81$ simultaneous quadratic equations (for a 9-state problem). The elements $(p_1 \ldots p_{81})$ of $\mathbf{P}_{k+1}$ are the desired solution. The Newton-Raphson algorithm to solve Eq. (15) is[18]

$$\mathbf{p}_{n+1} = \mathbf{p}_n - \mathbf{J}[\mathbf{p}_n]^{-1}\mathbf{g}[\mathbf{p}_n], \tag{16}$$

where the Jacobian matrix $\mathbf{J}[\mathbf{p}_n]$ is defined by

$$J(i,j) = \partial g_i/\partial p_j|_{\mathbf{p}_n} \text{ for } i,j = 1, \ldots, 81. \tag{17}$$

The Jacobian in Eq. (17) is thus an $(N^2 \times N^2) = (81 \times 81)$ matrix. We chose this algorithm because it is quadratically convergent and is a single-step algorithm (i.e., $\mathbf{J}[\mathbf{p}_n]$ is computed from the nth iterate $\mathbf{p}_n$ alone and not from prior iterates of $\mathbf{p}$).

The four steps in each iteration of the algorithm in Eq. (16) and the operations required to implement it are:

(i) Calculate the constituent matrices in Eq. (15) and form the N^2 vector $\mathbf{g}[\mathbf{p}_n]$.

This step requires matrix–matrix and matrix–matrix–matrix multiplication.

(ii) Calculate the $N^2 = 81$ elements of the Jacobian matrix $J(i,j)$ in Eq. (17). In Sec. V.B we describe a novel optical or digital approach for this step.

(iii) Modify Eq. (16) and solve Eq. (15) in the form.

$$\mathbf{J}(\mathbf{p}_n)\mathbf{s}_n = -\mathbf{g}[\mathbf{p}_n] \tag{18}$$

for $\mathbf{s}_n$, where

$$\mathbf{s}_n = \mathbf{p}_{n+1} - \mathbf{p}_n \tag{19}$$

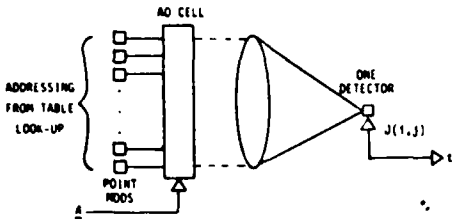


Fig. 3. Optical random-access AO storage/multiplier/summer for calculation of the Jacobian.

To solve the system of linear algebraic equations in (18), we apply the iterative Richardson algorithm[17] in Eq. (13) in the form

$$\mathbf{s}_{r+1} = \mathbf{s}_r + \omega[\mathbf{g}(\mathbf{p}_n) + \mathbf{J}(\mathbf{p}_n)\mathbf{s}_r]. \tag{20}$$

(iv) Compute $\mathbf{p}_{n+1}$ from Eq. (19).

This formulation is attractive since it circumvents the need to invert the Jacobian matrix in Eq. (16).

We illustrate our discrete-time EKF algorithm in the block diagram of Fig. 2. Our algorithm thus incorporates three nested iterative loops. In the innermost loop (index r), we apply the Richardson algorithm in Eq. (20) to solve Eq. (18) for $\mathbf{s}_{r+1}$. The iterations in Eq. (20) continue until $\mathbf{s}_{r+1} \simeq \mathbf{s}_r$. When the iterative Richardson algorithm converges, we set $\mathbf{s}_{r+1} = \mathbf{s}_n$ and form $\mathbf{p}_{n+1}$ from Eq. (19). If $|\mathbf{s}_n| > \varepsilon$, we begin a new Newton-Raphson loop (index n), calculate $\mathbf{g}[\mathbf{p}_n]$ in Eq. (15) and $\mathbf{J}[\mathbf{p}_n]$ in Eq. (17). We then repeat the Richardson iterative algorithm in Eq. (20) for the new $\mathbf{g}[\mathbf{p}_n]$ vector and Jacobian $\mathbf{J}[\mathbf{p}_n]$ matrix. This two-loop Newton-Raphson/Richardson iterative procedure is repeated until $|s_n| \leq \varepsilon$, at which point we set $\mathbf{P}_{n+1} = \mathbf{P}_{k+1}$. We now return to the KF loop (index k) and calculate a new $\mathbf{K}_k$ and $\hat{\mathbf{x}}_{k+1}$. The next measurement $\mathbf{z}_k$ can then be accepted. In Sec. VI we detail our full system architecture and quantify the processing time for our EKF.

B. Efficient Calculation of the Jacobian

The calculation of the Jacobian matrix is a crucial step in a Newton-Raphson solution. We have developed an efficient technique for **J** calculation using the table look-up method of Blackburn[19] (developed for the solution of the algebraic Riccati equation) and modified the table look-up method for an optical or digital systolic parallel processor and its application in our discrete-time EKF algorithm. We rewrite the Jacobian matrix in Eq. (17) as

$$\mathbf{J} = \mathbf{A}^T \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{A}^T, \tag{21}$$

where

$$\mathbf{A} = \mathbf{M}_{k+1}\mathbf{P}_{n+1} + \mathbf{L}, \tag{22}$$

and $\otimes$ denotes the Kronecker product; i.e.,

$$\mathbf{M} \otimes \mathbf{P} = (m_{ij}\mathbf{P})_{ij} \tag{23}$$

Equations (21) and (23) illustrate that the Kronecker product reorders the elements of **A**. Thus, calculation of **J** can be simply achieved by addressing the proper elements of **A**.

For an N-state problem, **A** in Eq. (22) is $(N \times N)$ and the full-order Jacobian **J** in Eq. (21) is $(N^2 \times N^2)$. Because the matrices **M**, **P**, **C**, and **G** and the quadratic matrix equation in (15) are symmetric, there are $N(N+1)/2 = 45$ unknown elements in **P** (rather than $N^2 = 81$). Thus, we can reduce the size of the Jacobian from (81×81) to (45×45). This simplifies the calculation of **J** and reduces the number of elements of **A** that must be addressed. The number of states or dimension ($N = 9$) of the problem and the symmetry of the matrices determine which elements of **A** form which elements of **J**. Since N is fixed, the same elements of **A** are addressed in each Newton-Raphson iteration. Although $\mathbf{P}_n$ and hence $\mathbf{A}_n$ change with the index n, the elements addressed remain the same. Thus, we need only form a new $\mathbf{A}_n$ matrix and use the same processor to calculate the new $\mathbf{J}_n$ matrix from this $\mathbf{A}_n$ matrix. Since $\mathbf{s}_n$ is a combination of $\mathbf{p}_n$ (the lexographically ordered elements of $\mathbf{P}_n$), **S** is symmetric (i.e., $s_{ij} = s_{ji}$). By applying this property and the fact that **J** multiplies $\mathbf{s}_n$ in Eq. (18), we can delete and combine redundant rows of **J** and the corresponding elements of $\mathbf{s}_n$. We find that only 405 elements (or one-fifth of the 2025 elements of the reduced-order **J**) are nonzero and must be calculated. The system of Fig. 3 can compute **J** using this table look-up technique and the aforementioned algorithm. In this system, the matrix **A** is fed to the AO cell. At successive instants of time, the proper point modulators are pulsed-on. This accesses the correct elements of **A**. By varying the strength of the point modulators, different weights or multiplictions of an element of **A** can be achieved. By pulsing-on two point modulators simultaneously, the sum of two elements of **A** can be produced. By this technique and architecture, **J** can easily be produced (one element at a time). The point modulators addressing each $\mathbf{T_B}$ are determined from a look-up table (and this table is fixed for an Nth order problem).

VI. Systolic EKF Architecture and Processing Time

For our AO cell we assume an aperture time $T_A = 35$ μsec, which is divided into 100 time slots; i.e., $T_B = 350$ nsec (for a 3-MHz data rate per channel). Calculation of **J** in Fig. 3 (which incorporates 81 point modulators) requires $T_J = 20.7$ μsec using this system. The nonlinear functions $\mathbf{h}[\hat{\mathbf{x}}]$ and $[\mathbf{H}]$ require evaluation of the arctangent and magnitude and are thus best formed in nonlinear analog modules. The calculation times for **h** and **H** (using conventional available off-the-shelf analog modules) are $t_h = 20$ μsec and $t_H = 30$ μsec, re-

Table I. Component Requirements and Performance for the Linear Algebraic Operations Required in Our EKF Algorithm

Size	Multiplication operation	Number of point modulators	Number of frequencies	Remarks	Load time (T_B)	Calculation time (T_B)	Symbol
(9 × 9)	Bipolar M-V	19	9		$2N-1$	2	MV
(9 × 9)	Bipolar M-M	35	9	Extend number of point modulators	$2N-1$	$2N$	MM
(9 × 9)	Bipolar MM + MM	53	9		$4N-1$	$2N$	MM + MM
(45 × 45)	Bipolar M-V	91	9	Partition (no partial sums)	$2N-1$	$\frac{2N^2-17N+9}{9}$	BPMV

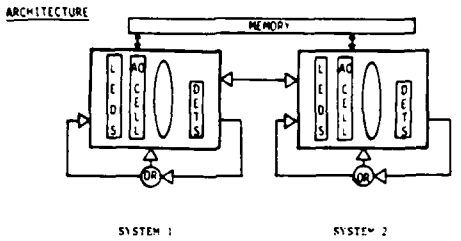


Fig. 4. Optical systolic discrete-time EKF processor architecture.

spectively. Faster computation of all these parameters is possible with a different AO cell, a higher data rate per channel and different analog modules. The performance goal (a 1-kHz measurement sample rate) can easily be achieved with these present components and the parameters noted above. Consequently, additional effort was not directed toward improving further the speed obtainable.

In the design of the optical system in Fig. 1, we restricted the number of frequencies to be a maximum of ten (to simplify the electronic support required) and we assumed an AO cell with a time–bandwidth product of 1000 maximum (this value is compatible with present off-the-shelf components). In Table I we list the linear algebraic operations which are required, and the system design parameters and performance specifications associated with each operation. The first three operations must be performed on (9 × 9) matrices. To represent bipolar data, we use 18 elements (space or time-multiplexed) to represent 9 bipolar values. The number of point modulators required in each case is noted (in all cases only 9 frequencies are used). The time to load the data into the AO cell of the system and the calculation time (once the data are loaded) are noted separately in units of the bit time T_B and as a function of the dimension N of the matrix. In general, the load time does not enter into the full processing time, since the operations can be pipelined to allow new data to be loaded as calculations proceed. The symbols used for each operation are noted in the last column of Table I.

In Fig. 4 we display the architecture of our final EKF design. We employ two of the optical systolic processors depicted in Fig. 1. The first system uses 35-input point modulators and the second uses 91-input point modulators. Each system employs 9 multiplexed

Table II. Operations and Timing for Our EKF Processor (*Defines Critical Calculation Time Path)

System 1			System 2		
Calculation	Operation	Load/calculation time	Calculation	Operation	Time
$*\mathbf{K}_k$	M-M	$18T_B$	$\mathbf{C}_k$	M-M	$18T_B$
$*T\mathbf{K}_k[\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k)]$	M-V	$2T_B$		M-M	$18T_B$
$\hat{\mathbf{x}}_{k+1}$	M-V + M-V	$19T_B$		M-M + M-M	$36T_B$
$\mathbf{h}(\hat{\mathbf{x}}_k)$	NL	20 μsec			
$*\mathbf{H}_{k+1}$	NL	30 μsec			
$*\mathbf{M}_{k+1}$	M-M-M	$22T_B$			
$\mathbf{P}_{k+1}$	(Newton-Raphson) M-M	$18T_B$	$*\mathbf{s}_n$	(Richardson) M-M	$18T_B$
				Jacobian	20.7 μsec
				BPMV	$455T_B$

frequencies within the AO cell. This allows optimal use of parallelism in the operations required in our EKF algorithm.

In Table II we compile the operations and load and calculation times for the sequence of steps (for each time-sample k) in our optical systolic discrete-time EKF processor (Fig. 4). The asterisks in the table label the critical time paths in our processor.

We summed the load and calculation times delineated in Table II and found the total calculation time for one time-sample k in our EKF processor to be

$$T_k = 42T_H + 30\ \mu\text{sec} + n[18T_H + 20\ \mu\text{sec} + r\ 455T_B]. \quad (24)$$

where n and r denote, respectively, the number of Newton-Raphson and Richardson algorithm iterations required. For the bit time $T_B = 350$ nsec, the computation time is

$$T_k = 44.7 + n[26 + 159r]\ \mu\text{sec}. \quad (25)$$

The largest time is spent in the innermost Richardson loop. [The solution of the symmetric quadratic matrix equation in (15) for $\mathbf{P}_{k+1}$ consumes 95% of the computation time.] We have thus optimized our system, the data flow, and the data format to reduce the computational time for this specific bipolar MV operation (in Tables I and II). From our initial simulations, we found $n = 2$ or 3 and $r = 2$ to be adequate to achieve convergence of our algorithm. On substituting these values into Eq. (25), we find the processing time for one EKF sample to be $T_k = 0.73$ msec or $T_k = 1.08$ msec. These correspond to a measurement data rate of $1/T_k = 1.36$ kHz or 0.926 kHz. These data rates are adequate for our present goal of ~1-msec data measurements. As we noted at the outset of this section, faster calculation times are possible with different system and component parameters choices.

VII. Summary and Conclusions

In this paper we have considered a specific problem in optical linear algebra, the realization of a Kalman filter (for an air-to-air missile engagement) and have developed a new discrete-time extended Kalman filter algorithm for this application. We have structured the steps in this algorithm in terms of basic linear algebra operations. We then detailed how each operation could be realized on an optical systolic array processor, and the pipelining and data flow between all steps. A specific frequency-multiplexed optical systolic processor was chosen since it allows the flexibility of performing different operations with different data encoding and format control techniques. A new triple-nested algorithm was devised to solve the discrete-time extended Kalman filter problem for this application. This algorithm also involved a new optical technique to evaluate the Jacobian matrix. The performance achieved in this design was found to be adequate for our intended measurement sampling rate. This represents a new application of systolic array processors (optical or digital). This paper represents an important case study in how individual operations must be pipelined and nested to solve an overall system problem.

The support of our optical systolic processor research by NASA Lewis Research Center (grant NAG-3-5) and the support of the Air Force Office of Scientific Research (grant AFOSR-79-0091) for missile guidance applications of this technology are gratefully acknowledged.

References

1. H. J. Caulfield *et al.*, "Optical Implementation of Systolic Array Processing," Opt. Commun. **40**, 86 (1981).
2. D. Casasent, "Acoustooptic Transducers in Iterative Optical Vector-Matrix Processors," Appl. Opt. **21**, 1859 (1982).
3. D. Casasent, J. Jackson, and C. Neuman, "Frequency-Multiplexed and Pipelined Iterative Optical Systolic Array Processors," Appl. Opt. **22**, 115 (1983).
4. R. P. Bocker, H. J. Caulfield, and K. Bromley, "Rapid Unbiased Bipolar Incoherent Calculator Cube," Appl. Opt. **22**, 804 (1983).
5. R. A. Athale and W. C. Collins, "Optical Ma[illegible]ultiplier Based on Outer Product Decomposi[illegible]," Appl. Opt. **21**, 2089 (1982).
6. M. Carlotto and D. Casasent, "Microprocessor-Based Fiber-Optic Iterative Optical Processor," Appl. Opt. **21**, 14[illegible]
7. D. Casasent, C. P. Neuman, and M. Carlotto, "An Electro-Optical Processor for Optimal Control," Proc. Soc. Photo-Opt. Instrum. Eng. **295**, 176 (1981).
8. C. P. Neuman, D. Casasent, and R. Baumbick, "An Electro-Optical Processor for the Optimal Control of F100 Aircraft Engines," in *Proceedings Electro-Optical Systems Design Conference* (Industrial & Scientific Conference Management, Chicago, 1981), pp. 311–320.
9. J. Jackson and D. Casasent, "State Estimation Kalman Filter Using Optical Processing: Noise Statistics Known," Appl. Opt. **23**, 376 (1984).
10. R. H. Travassos, "Real-Time Implementation of Systolic Kalman Filters," Proc. Soc. Photo-Opt. Instrum. Eng. **431**, 97 (1983).
11. W. A. Roemer and P. S. Maybeck, "An Optically Implemented Multiple-Stage Kalman Filter Algorithm," Proc. Soc. Photo-Opt. Instrum. Eng. **431**, 221 (1983).
12. R. L. Barron *et al.*, "A New Class of Guidance Laws for Air-To-Air Missiles," in *Proceedings, Third Meeting of the Coordinating Group on Modern Control Theory, Part 1* (Oct. 1081), pp. 20–21.
13. T. L. Riggs and P. L. Vergez, "Advanced Air-To-Air Missile Guidance Using Optimal Control and Estimation," Report on Contrast AFATL/DLMA, AFATL-TR-81-56 (June 1981).
14. P. S. Maybeck, *Stochastic Models, Estimation, and Control, Vol 1*, (Academic, New York, 1979).
15. A. Gelb, *Applied Optimal Estimation* (MIT Press, Cambridge, Mass., 1974).
16. A. E. Bryson, Jr., and Y. C. Ho, *Applied Optical Control* (Blaisdell, Waltham, Mass., 1969).
17. R. S. Varga, *Matrix Iterative Analysis* (Prentice-Hall, Englewood Cliffs, N.J., 1962).
18. R. E. Bellman and R. E. Kalaba, *Quasilinearization and Nonlinear Boundary-Value Problems* (Elsevier, New York, 1965).
19. T. R. Blackburn, "Solution of the Algebraic Matrix Riccati Equation via Newton-Raphson Iteration," AIAA J. **6**, 951 (1968).

12. A STATE ESTIMATION KALMAN FILTER USING OPTICAL PROCESSING: NOISE STATISTICS KNOWN

Reprinted from Applied Optics, Vol. 23, page 376, February 1, 1984

State estimation Kalman filter using optical processing: noise statistics known

James Jackson and David Casasent

Carnegie-Mellon University, Department of Electrical & Computer Engineering, Pittsburgh, Pennsylvania 15213.
Received 18 August 1983.
0003-6935/84/030376-03$02.00/0.

Kalman filtering and state estimation are major techniques used in many control and signal processing applications.[1,2] A Kalman filter produces the optimum least mean square or maximum-likelihood estimate of the state of a linear system driven by additive noise. In Kalman filtering, a system or process model with additive noise and a sensor measurement system with additive noise are assumed. The Kalman filter provides estimates of the state of the system, the accuracy of the most recent estimate, and the control for the system. This is achieved, assuming that the system and measurement noise are known (zero-mean Gaussian statistics are assumed), by recursive curve-fitting to estimate the state of the system. The next state estimate is a linear combination of the prior control and the prior estimate and the uncertainty measurement of the sensor's noise. Depending upon the amount of process and sensor noise, more weight (through the Kalman filter gain matrix) is given to the present estimate or the present measurement.

In Ref. 3, we described a frequency-multiplexed acoustooptic (AO) processor and detailed how it was capable of performing all the individual operations (matrix–matrix–matrix multiplication, matrix inversion, etc.) required for Kalman filtering. However, the data flow and organization of all required operations were not detailed. In this Letter, we consider specifically a simpler Kalman filter state estimation problem. We assume that the measurement vector $\mathbf{z}_k$ is received serially and is sampled at regular intervals. We now consider the problem of calculating the state estimation vector $\hat{\mathbf{x}}_k$ and the extrapolated state estimation vector $\bar{\mathbf{x}}_{k+1}$, assuming that the system's noise statistics are known. The data flow is found to be quite ideal in the iterative optical processor we devised for this Kalman filtering state estimation problem.

In Table I, we list the discrete-time Kalman filter equations. For more details and a derivation of these equations, Refs. 1–3 should be consulted. We assume equally spaced time-sampled intervals kT_s, where k is the iterative time index. We assume that the system noise vector $\mathbf{w}$ and the measurement noise vector $\mathbf{v}$ are uncorrelated and Gaussian-distributed and that the noise statistics ($\mathbf{Q}$ and $\mathbf{R}$) and the system model ($\mathbf{\Phi},\mathbf{\Gamma},\mathbf{H}$) are known. Thus the error covariance matrix $\mathbf{P}$ and the extrapolated error covariance matrix $\mathbf{M}$ can be precomputed, and hence the Kalman gain matrix $\mathbf{K}_k$ can be precomputed and stored for each input time sample.

With these assumptions, we now consider the state of the filter and system and the calculations required after receipt of a new measurement sample $\mathbf{z}_k$ at time kT_s. From the previous Kalman filter cycle, we have an extrapolated state estimate $\bar{\mathbf{x}}_k$, and from the known noise statistics we have precomputed the Kalman gain matrices. In many cases, the noise statistics (and the matrices describing the system model) change sufficiently slowly that the storage requirements and updating requirements for $\mathbf{K}_k$ and the other necessary system and noise matrices ($\mathbf{\Phi}_k$, $\mathbf{\Gamma}_k\mathbf{H}_k$, $\mathbf{Q}_k$, and $\mathbf{R}_k$) are not excessive. When a new measurement sample $\mathbf{z}_k$ is obtained at time kT_s, the new state vector estimate $\hat{\mathbf{x}}_k$ must be calculated from Eq. (1c), and then the new extrapolated state estimate vector $\bar{\mathbf{x}}_{k+1}$ must be evaluated using Eq. (1d). Thus, with known noise statistics and a known system model, the required Kalman filter state estimate calculations required for each new input sample are simply Eqs. (1c) and (1d) of Table I. Combining these equations, we describe the simplified Kalman filter and the calculations required by

$$\bar{\mathbf{x}}_{k+1} = (\mathbf{\Phi}_k - \mathbf{\Phi}_k\mathbf{K}_k\mathbf{H}_k)\bar{\mathbf{x}}_k + \mathbf{\Phi}_k\mathbf{K}_k\mathbf{z}_k + \mathbf{\Gamma}_k\bar{\mathbf{w}}_k. \quad (2a)$$

$$\bar{\mathbf{x}}_{k+1} = \mathbf{A}_k\bar{\mathbf{x}}_k + \mathbf{B}_k\mathbf{z}_k + \mathbf{\Gamma}_k\bar{\mathbf{w}}_k. \quad (2b)$$

Since the matrices $(\mathbf{\Phi}_k - \mathbf{\Phi}_k\mathbf{K}_k\mathbf{H}_k)$ and $\mathbf{\Phi}_k\mathbf{K}_k$ can be precomputed, we denote the associated matrices required in Eq. (2a) by $\mathbf{A}_k$ and $\mathbf{B}_k$ as noted in Eq. (2b) to simplify notation. If we had included the assumption of zero-mean system noise ($\bar{\mathbf{w}}_k = 0$), the equation would simplify even further.

Table I. Discrete-Time (k) Kalman Filter Equations

Description	Defining Equations
System model	$\mathbf{x}_{k+1} = \mathbf{\Phi}_k\mathbf{x}_k + \mathbf{\Gamma}_k\mathbf{w}_k$ (1a) $\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k$ (1b)
State estimate	$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\bar{\mathbf{x}}_k)$ (1c)
Extrapolated state estimate	$\bar{\mathbf{x}}_{k+1} = \mathbf{\Phi}_k\hat{\mathbf{x}}_k + \mathbf{\Gamma}_k\bar{\mathbf{w}}_k$ (1d)

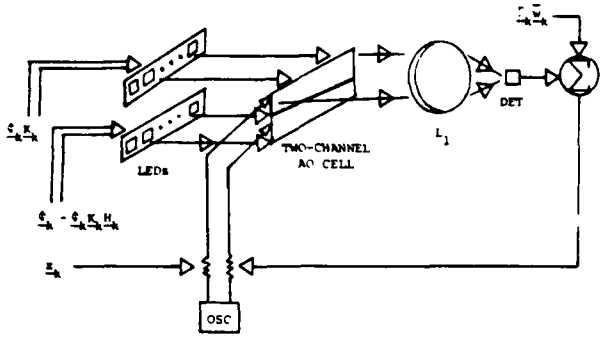


Fig. 1. Schematic diagram of an optical systolic processor for Kalman filter state estimation.

We consider the use of optical systolic array processors employing acoustooptic (AO) transducers to perform the necessary computations in Eqs. (2). Only single-channel optical systolic array processor architectures[3,4] have been described thus far in the literature. This class of optical linear algebra processors is quite general purpose. Attention, however, must be given to the flow of data and operations in such systems; and for high computational efficiency one must avoid analog-to-digital conversion and the storage of intermediate data results. To perform efficiently the calculations necessary in Eqs. (2), we found that a multichannel optical systolic array processor yielded the best results. One realization with such a system is shown in Fig. 1.

This architecture is a new two-channel iterative optical systolic array processor. In the system of Fig. 1, two linear LED arrays are imaged onto two separate channels of an AO cell. This forms the separate product of the corresponding input LED data and the contents of the AO cell. Since all the data in the AO cell are present at the same frequency, all the light distribution leaving both channels of the AO cell will be deflected in the same direction and will thus be focused by lens L_1 at the same horizontal location in the output plane. Lens L_1 also vertically integrates and focuses the light leaving both channels of the AO cell. The size of the detector is chosen to collect all this light. Thus the summation of the total light distribution leaving both channels of the system is formed on the single-output photodetector (DET). The upper AO cell channel is fed with the measurement vector $\mathbf{z}_k$, and the lower AO cell channel is fed with the prior state estimate $\bar{\mathbf{x}}_k$. The upper LED array is fed with one row of the matrix $\mathbf{A}_k$ in parallel, and the lower LED array is similarly fed sequentially with the rows of $\mathbf{B}_k$. Subsequent rows of $\mathbf{A}_k$ and $\mathbf{B}_k$ are entered every T_s in parallel. For an $N \times N$ matrix, the leftmost N LEDs (1 to N) are addressed with the first row of $\mathbf{A}_k$ and $\mathbf{B}_k$ at time step T_s. At $2T_s$, LEDs 2 to $N+1$ are addressed with the next row of $\mathbf{A}_k$ and $\mathbf{B}_k$. Since the data in the AO cell move horizontally by a time step T_s, it is necessary to stagger the LEDS being addressed at each kT_s time as detailed in Ref. 3.

At each T_s time step, the output from the photodetector will be one element of the matrix-vector product and vector summation given by the first two terms in Eq. (2a) or (2b). After NT_s, the vector data have reached the end of the AO cell, and the entire matrix-vector product has been produced as the time-history output from the detector. As each element of the output vector is produced, the corresponding element of $\mathbf{\Gamma}_k \bar{\mathbf{w}}_k$ [the last term in Eq. (2)] is added to it (using a simple resistor adder) to produce one element of the new state update vector $\bar{\mathbf{x}}_{k+1}$. As this vector is produced, it is fed back directly into the lower AO cell. The aperture time of the AO cell is chosen to be $(2N-1)T_s$, and $(2N-1)$ LEDS are used. Thus, after NT_s, a new extrapolated state estimate vector $\bar{\mathbf{x}}_{k+1}$ has been produced and loaded into the cell (together with the new measurement vector $\mathbf{z}_k+1$). The above iterative cycle can then be immediately repeated on the new sampled data. As seen, data flow in such a system is ideal. (As soon as an output is produced, it is loaded directly into the newly vacant slot at the transducer end of the AO cell.) The time history of the output from the adder is the new extrapolated state estimate, which can then be used for various control applications and other on-line adaptive processing functions depending upon the application. Many variations of this basic architecture are possible, such as the use of a linear CCD shift register detector readout system as in Ref. 4, frequency-multiplexing of the LED or AO cell data as in Ref. 1. These different systems may be preferable for specific applications such as when the number of states is large, but the input data sampling rate is slow. The system of Fig. 1 appears to be the best general solution at present.

We now briefly consider the extension of this system to allow it to operate on bipolar-valued matrix and vector data. Many possibilities exist. The one we have found to be most attractive is a direct extension of the system of Fig. 1. We frequency-multiplex the inputs to the AO cell (and thus use both its bandwidth and delay time). For matrices and vectors with bipolar values, we enter the positive $\bar{\mathbf{x}}_k^+$ and negative $\bar{\mathbf{x}}_k^-$ parts of the vector $\bar{\mathbf{x}}_k = \bar{\mathbf{x}}_k^+ - \bar{\mathbf{x}}_k^-$ into the lower channel of the AO cell in parallel on two separate frequencies. We separate the positive and negative parts of the input matrices and time multiplex the LED outputs (first pulsing them on with the positive-valued matrix data and then with the negative-valued matrix data). A similar time and frequency division multiplexing is used for the measurement data and the upper AO cell channel. With the LEDs pulsed at twice the input data rate to the AO cell, the system thus operates properly with no reduction in the input data rate it can handle. At the output, we determine the magnitude and sign of $\bar{\mathbf{x}}_{k+1}$ and appropriately feed this data back to the AO cell iteratively as before. The sign of $\bar{\mathbf{x}}_{k+1}$ is determined from the time slot in the detector output with a nonzero value. From the sign of $\bar{\mathbf{x}}_{k+1}$, we select the multiplexed frequency input to the AO cell to be used. The data in such an architecture still pipeline ideally from the output detector back to the AO cell. Extensions of this technique to handling complex-valued data by the use of three frequencies to encode complex data by their projections on the 0, 120, and 240° axes in complex space[6] also follow directly.

These techniques for handling bipolar data increased the required LED source modulation rate by a factor of 2 above the input data sampling rate. (However, the number of LEDs required is not increased.) More important, the bipolar data handling technique requires a quadruple increase in the AO cell's bandwidth and in its time–bandwidth product (a factor of 2 due to the two frequencies used and a factor of 2 due to the doubling of the LED modulation rate). Despite these disadvantages, this technique is more appropriate than the use of various biasing methods or the use of two cycles to process bipolar data (one cycle for positive data and one for negative data) because of the complicated detector postprocessing that results and/or the intermediate data storage required. Thus the reduced amount of data shuffling and postprocessing that results with the technique described above seems to make it preferable. If the dimension of the matrix–vector problem becomes too large (i.e., if the entire vector will not fit into ½ of the AO cell at one time), matrix-partitioning techniques on several simple system modifications are required. These are simple conceptually and require using

more than two AO cell channels plus time-division multiplexing of the LED inputs and the single detector output. We will detail such issues in a later publication.

The use of multichannel AO cells (together with proper time-division multiplexing of the inputs and outputs of the system) represents a major extension to this class of optical systolic array processor. Their applications to Kalman filtering, state estimation, and handling bipolar and complex-valued data appear to be quite significant.

The support of the Air Force Office of Scientific Research (grant AFOSR 79-0091) and NASA Lewis Research Center (grant NAG-3-5) for this aspect of our optical data processing research program is gratefully acknowledged.

References

1. A. Bryson and Y. C. Ho, *Applied Optimal Control* (Blaisdell, Waltham, Mass. 1969).
2. B. Anderson and J. Moore, *Optimal Filtering* (Prentice-Hall, Englewood Cliffs, N.J., 1979).
3. D. Casasent, J. Jackson, and C. P. Neuman, Appl. Opt. **22**, 115 (1983).
4. H. J. Caulfield *et al.*, Opt. Commun. **40**, 86 (Dec. 1981).
5. D. Casasent, Appl. Opt. **21**, 1859 (1982).
6. J. W. Goodman and L. M. Woody, Appl. Opt. **16**, 2611 (1977).

13. ITERATIVE SOLUTIONS TO NONLINEAR MATRIX EQUATIONS USING A FIXED NUMBER OF STEPS

ITERATIVE SOLUTIONS TO NONLINEAR MATRIX EQUATIONS USING A FIXED NUMBER OF STEPS

D. Casasent, A. Ghosh* and C.P. Neuman

Carnegie-Mellon University
Department of Electrical and Computer Engineering, Pittsburgh, Pennsylvania 15213

*Present Address: AT&T Bell Laboratories, Allentown, Pennsylvania 18103

ABSTRACT

An iterative algorithm for the solution of a quadratic matrix equation (the algebraic Ricatti equation) is detailed. This algorithm is unique in that it allows the solution of a nonlinear matrix equation in a finite number of iterations to a desired accuracy. Theoretical rules for selection of the operation parameters and number of iterations required are advanced and simulation verification and quantitative performance on an error-free processor are provided. An error source model for an optical linear algebra processor is then advanced, analyzed and simulated to verify and quantify our performance guidelines. A comparison of iterative and direct solutions of linear algebraic equations is then provided. Experimental demonstrations on a laboratory optical linear algebra processor are included for final confirmation. Our theoretical results, error source treatment and guidelines are appropriate for digital systolic processor implementation and for digital-optical processor analysis.

1. INTRODUCTION

Optical linear algebra processors (OLAPs) represent a most general and attractive use of the parallelism and real-time processing features of optical systems [1]. The frequency-multiplexed acousto-optic (AO) processor [2,3] of Figure 1 represents a most general-purpose OLAP architecture with ease of fabrication [4] and competitive computational rates [2,4]. In this architecture (Figure 1), N point modulator inputs are imaged through N separate regions of an AO cell. These individual regions are separated by T_B of time (for propagation of the acoustic wave) and by a physical distance d_B. In [2], the use of this processor in iterative algorithms, direct LU and QR matrix decomposition algorithms, and triangular system solutions was detailed.

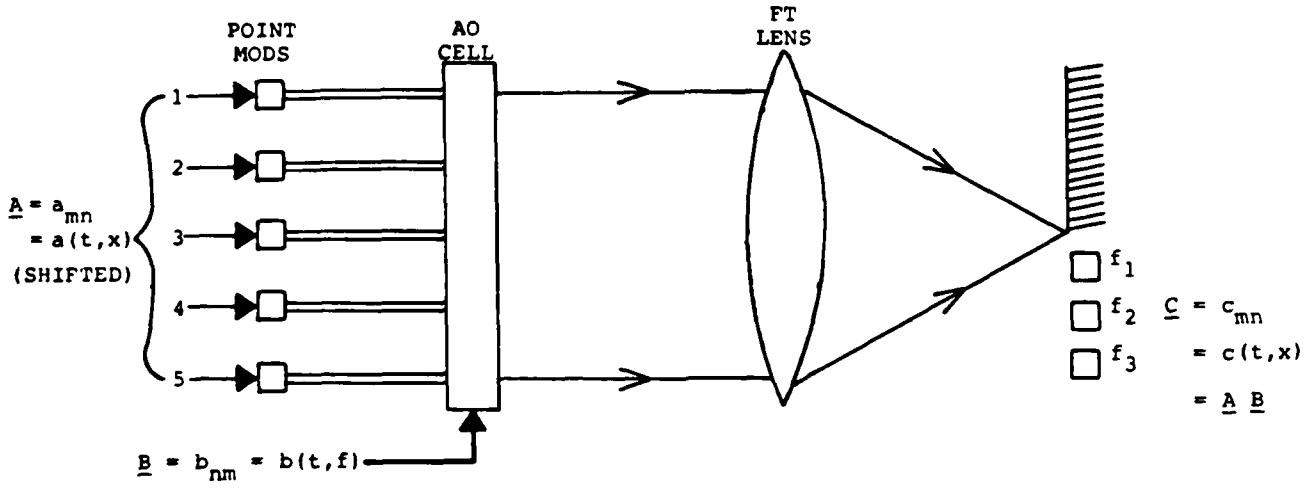


FIGURE 1
Simplified schematic of a frequency-multiplexed optical linear algebra processor [3]

In this paper, we consider the use of this processor for the solution of a nonlinear matrix equation (Section 2). The specific application chosen is the solution of the algebraic Ricatti equation (ARE). This nonlinear equation is similar to the expressions to be solved in Kalman filtering and other advanced modern signal processing algorithms. An iterative solution is necessary for such problems and for eigensystem solutions. Our proposed nonlinear ARE solution is quite unique since it requires a finite number of steps to achieve a specific accuracy and performance. In Section 3, we summarize selection of the operational parameters for such an iterative algorithm and the theoretical basis for our choice of

the fixed number of iterations to be used. Section 4 presents initial error-free simulation data. In Section 5, we advance our error source model. In Section 6, we review our iterative and direct solutions to systems of linear algebraic equations (LAEs). This represents the fundamental operation required in advanced linear algebra algorithms. Section 7 contains simulation data to quantify the dominant error sources and the accuracy expected from such algorithms. We conclude in Section 8 with the experimental verification and quantification of our theoretical results. Our summary and conclusions are then advanced in Section 9.

2. NONLINEAR MATRIX SOLUTION

In reference [5], we detailed a solution to the linear quadratic regulator control problem to minimize a quadratic performance index for a linear system. Computation of the regulator feedback gain matrix $\underline{K}$ that defines the optimal controls $\underline{u}$ involves the solution of the ARE

$$\underline{S}\,\underline{F} + \underline{F}^T\underline{S} - \underline{S}\,\underline{L}\,\underline{S} + \underline{Q} = \underline{0} \tag{1}$$

for $\underline{S}$. To achieve this, we used the Kleinman algorithm [5] and the solution of the vectorized Lyapanov equation to format the solution of (1) as a solution of the set of LAEs

$$\underline{H}(k)\underline{s}(k) = \underline{y}(k), \tag{2}$$

where $\underline{s}$ and $\underline{y}$ are the vectorizations of $\underline{S}$ and $\underline{S}\,\underline{L}\,\underline{S} - \underline{Q}$ respectively and $\underline{H}$ is a Kronecker formatted matrix. This system of LAEs must be solved successively with different matrices $\underline{H}$ and vectors $\underline{y}$ with the results of one cycle used to compute the matrix $\underline{H}$ and vector $\underline{y}$ for the next cycle. To achieve this, we employ a two-loop iterative algorithm described by

$$\underline{s}(r+1,k) = [\underline{I} - \omega(k)\underline{H}(k)]\underline{s}(r,k) + \omega(k)\underline{y}(k). \tag{3}$$

In solving (2) using (3), we solve (2) for one outer loop iteration k, update $\underline{H}$ and $\underline{y}$ and solve the next LAE. This procedure continues until $\underline{s}$ is of sufficient accuracy. The algorithm in (3) implies an iterative solution for each LAE. Direct solutions are also possible as we discuss in Sections 6 and 7. The indices r and k in (3) refer to Richardson (inner) and Kleinman (outer) loop iterations respectively.

3. OPERATIONAL PARAMETER SELECTION

In an iterative algorithm such as (3), various operational parameters must be selected. The initial selection $\underline{s}(0,0)$ for $\underline{S}$ and the choice $\underline{s}(0,k)$ for each LAE solution are required. For $\underline{s}(0,0)$, we use $\underline{0}$ to insure outer loop convergence (a stability matrix). For $\underline{s}(0,k)$, we use the obvious choice of the prior $\underline{s}(0,k-1)$ estimate. The acceleration parameter ω in (3) is chosen to be $\omega = n/\lambda_{max} \approx 3/||\underline{H}(k)||$. This insures inner loop convergence [2,5]. Stopping the inner loop ierations (index r) for each LAE solution and stopping the number of outer loop iterations (index k) is a major decision.

In reference [5], we derived bounds for the inner loop error, the outer loop error and their coupling. From this analysis, we derived the selection of a fixed number of inner loop iterations R to solve each LAE given by

$$R = nC = C \log \alpha \approx 1.5C \text{ to } 3.0C. \tag{4}$$

where $||\underline{x}^*(0)-\underline{x}^*(1)|| \leq \alpha$ and $[1 - 1/C]^R \approx \exp(-n) < 1/\alpha$ is chosen. This follows from our analysis of the error in an iterative solution (due to a fixed number of iterations R), which showed that the norm of such an error is

$$||\underline{s}(r,k) - \underline{s}^*|| = ||\underline{I} - \omega\underline{H}(k)||^r = (1 - 1/C(k))^r, \tag{5}$$

where C is the condition number of $\underline{H}$. Since r is expected to increase with C, we set $r = nC$ and thus select n such that the error between the computed solution $\underline{s}$ and the exact solution $\underline{s}^*$ in (5) is as small as is required. For the fixed number of outer loop iterations K, we use $K = 5$ or 6, which can be theoretically derived (and appropriately modified) for other applications with matrices with specific features. These iterative operational parameter selections are summarized in Table 1.

4. ERROR-FREE SIMULATION RESULTS

The performance measures we adopted to assess performance of the algorithm in Section 2 implemented using the operational parameters in Table 1 are the maximum percent error in any element of the matrix $\underline{K}$ (i.e. $\Delta K_{max}\%$) and the maximum error in the location of the closed-loop poles of the system ($\Delta\lambda_{max}\%$). We expect $\Delta K >> \Delta\lambda$ and note that $\Delta\lambda$ is the more

appropriate error measure for this specific application and that similar error measures should be used to evaluate the performance of other specific case studies. In Figures 2 and 3, we show the variation of these two error measures with the number of outer loop iterations k for a fixed number of inner loop iterations for two case studies. These case studies are the fifth (Figure 2) and third (Figure 3) order models of an F100 engine. As seen from the data for these two case studies, the use of a fixed number of iterations results in a monotonic decrease in the solution error with the ΔK error being approximately ten times that of the $\Delta\lambda$ error. From these results, we conclude that the use of a fixed number of iterations can yield adequate results when the number of iterations is properly chosen. Our parameter selection guidelines in Table 1 have thus all been verified and discussed.

TABLE 1
Operational Parameter Selection Guidelines [5]

SYMBOL	PARAMETER	PREFERRED CHOICE
$\underline{s}(0,0)$	Initial Initialization	$\underline{s}(0,0) = \underline{0}$
$\underline{s}(0,k)$	k-th Kleinman Loop Initialization	$\underline{s}(0,k) = \underline{s}(0,k-1)$
R	Number of Inner Loop Iterations	R = 1.5C to 3.0C
K	Number of Outer Loop Iterations	K = 5 - 6
$\omega(k)$	Acceleration Parameter	$\omega(k) = 3/\|\|\underline{H}(k)\|\|$

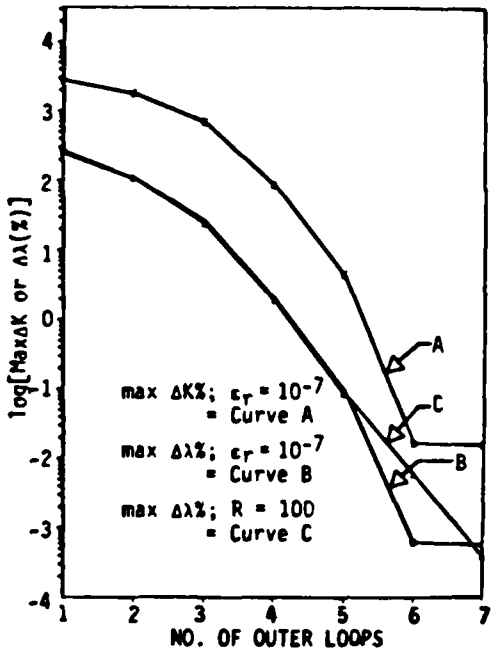


FIGURE 2
Variation of the error measures $\Delta K_{max}(\%)$ and $\Delta\lambda_{max}(\%)$ with the number of outer-loop iterations K for different inner-loop iteration stopping criteria for the fifth-order HPG3 F100 model

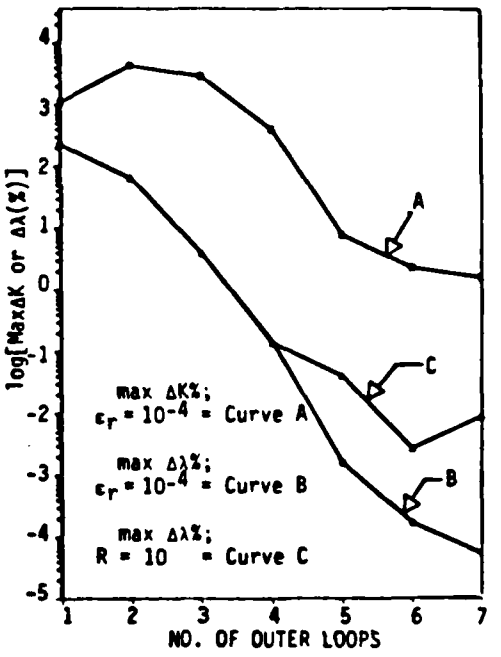


FIGURE 3
Variation of the error measures $\Delta K_{max}(\%)$ and $\Delta\lambda_{max}(\%)$ with the number of outer-loop iterations K for different inner-loop iteration stopping criteria for the third-order HPG3 F100 model

5. ERROR SOURCE MODEL

In earlier publications [7,8], we detailed the first system and component error source model for an OLAP and the general issue of errors in such an architecture. In this section, we review this OLAP error source model. In this model, we distinguish input, AO cell and detector plane errors separately. Spatial errors include: input and detector response variations and errors in the interconnections between the input modulators and the AO cell, and detector dark current. The spatial variations are fixed (time-independent) and are

correctable to small residual levels as required (by adjusting the gain of the input point modulators, the detector amplifiers, and the input matrix and vector data). Detector noise is the only time-varying error source considered. Acoustic attenuation produces a deterministic exponential variation of the data in the AO cell. This effect is dispersive, but its frequency dependence is not included in our present model. Acoustic attenuation can be corrected at one frequency and is thus an input spatial error. The product of an input matrix $\underline{A}$ and vector $\underline{b}$ thus yields a final output $\underline{\hat{d}}$ given by

$$\underline{\hat{d}} = \left[\begin{array}{c}\text{Detector}\\ \text{Spatial}\\ \text{Response}\\ \text{Variations}\end{array}\right]\left[\begin{array}{c}\text{AO Cell}\\ \text{Frequency}\\ \text{Response}\\ \text{Variations}\end{array}\right]\left[\begin{array}{c}\text{Data}\\ \text{Matrix}\\ \underline{A}\end{array}\right]\left[\begin{array}{c}\text{AO Cell}\\ \text{Attenuation}\end{array}\right]\left[\begin{array}{c}\text{Point Mod}\\ \text{Response and}\\ \text{Interconnection}\\ \text{Variations}\end{array}\right]\left[\begin{array}{c}\text{Data}\\ \text{Vector}\\ \underline{b}\end{array}\right] + \left[\begin{array}{c}\text{Detector}\\ \text{Dark}\\ \text{Current}\end{array}\right] + \left[\begin{array}{c}\text{Time-}\\ \text{Varying}\\ \text{Detector}\\ \text{Noise}\end{array}\right]. \quad (6)$$

As seen, the different types of system and component variations are described by error matrices that multiply the input data vector or input matrix data. Thus, the system errors are described by the corresponding variations in the data matrix and vector. The detector dark current and noise appear additively in the output vector as shown in Eq.(6).

6. DIRECT AND INDIRECT SOLUTIONS OF LAEs

The solution of a system of LAEs, $\underline{A}\ \underline{x} = \underline{b}$ is the fundamental operation required in most linear algebra processors and signal processing applications. Thus, we concentrate on this function. The two major types of LAE solutions are direct or matrix decomposition solution and an iterative or indirect solution.

The preferable iterative algorithm is [2,9]

$$\underline{x}(r+1) = \underline{x}(r) + \omega[\underline{b} - \underline{A}\,\underline{x}(r)], \quad (7)$$

where ω is an acceleration parameter chosen to insure convergence. The iterations (described by the iterative index r) continue until $\underline{x}(r) = \underline{x}(r+1)$. Then, (7) reduced to $\underline{A}\ \underline{x} = \underline{b}$ and the system's output $\underline{x}$ is the desired solution. To implement (7) on the system of Figure 1, the matrix data is fed to the AO cell one column at a time in parallel with the rows of the matrix frequency-multiplexed, i.e. with the matrix elements a_{mn} encoded in time and frequency as $a(f,t)$ and with the vector data $\underline{x}$ spatially-multiplexed as $x(x)$ and fed in parallel to the input point modulators. The matrix-vector product $\underline{A}\ \underline{x}$ is formed, operated upon in analog or digital post-processing electronics to produce the right-hand side of (7) and hence the new $\underline{x}$ iterate input to the point modulators. Thus, the detector output is fed back to the input point modulators. The length of the AO cell NT_B is chosen to be just as sufficient to accommodate the matrix data. Each T_B, as one column of the matrix leaves the AO cell, it is reintroduced into the bottom of the cell. This recycling of the matrix data is more efficient for system fabrication and reduces the effects of acoustic attenuation.

In direct solutions, the matrix $\underline{A}$ and the vector $\underline{b}$ are multiplied by a decomposition matrix $\underline{P}_1$ to generate new $\underline{A}_1$ and $\underline{b}_1$. Each such matrix-matrix and matrix-vector multiplication generates one row of the final $\underline{A}'$ matrix and one element of the final $\underline{b}'$ vector. After each matrix-matrix multiplication, the order of the matrix and vector are reduced by one and the new reduced order $\underline{A}_1$ and $\underline{b}_1$ are multiplied by a new $\underline{P}_2$. This procedure is repeated N-1 times (for an N x N matrix) and yields a new upper-triangular matrix $\underline{U}$ and a new vector $\underline{b}'$. This simplified upper-triangular system of equations $\underline{U}\ \underline{x} = \underline{b}'$ is then easily solved by back-substitution. The matrix-decomposition can be realized either as an LU decomposition (this is the technique we use when the matrix is positive-definite or diagonally-dominant, as is the case here, since pivoting is then not required) or as a QR orthogonal decomposition (this technique is more general and stable, but is more difficult to realize). The detailed implementation of LU [2,10] and QR [2,11] decomposition and back-substitution [2,12] have been described elsewhere. To implement the Gaussian-elimination algorithm (LU) used in the present application on the system of Figure 1, we feed one row of the matrix $\underline{A}$ to the AO cell in parallel (with the columns of $\underline{A}$ frequency-multiplexed, i.e. with the elements a_{mn} of $\underline{A}$ frequency and time encoded as $a(t,f)$) and with one row of the decomposition matrix $\underline{P}_j$ fed to the input point modulators in parallel (with the elements p_{mn} of $\underline{P}$ time and space encoded as $p(t,x)$). To facilitate data flow and for speed, we simultaneously operate on $\underline{A}$ and $\underline{b}$ by using an augmented matrix. One row of the augmented matrix $\underline{A}'$ is produced in parallel as $a'(t,x)$ on the output detector during each of the N cycles. The new $\underline{P}_j$ matrix is easily calculated from the elements of the j-th column of the augmented matrix in dedicated electronics.

7. SYSTEM ERROR EFFECTS ON THE SOLUTION OF LAEs

The direct solution requires an AO cell of twice the length of the matrix, but achieves the decomposition in a fixed number of steps. However, as noted in Section 3, iterative

algorithms can be operated with a fixed number of iterations to achieve a given desired accuracy and iterative algorithms are essential [2] for eigen-systems solutions and the solution of nonlinear matrix equations such as the ARE [5] and in Kalman filtering [13]. In our new results (Sections 7 and 8), we compare [6] the performance of direct and iterative algorithms in the solution of the LAEs that arise in a specific ARE solution for the F100 engine. The two cases considered are third and fifth-order F100 models. These give rise to 9 x 9 and 25 x 25 matrices. Bipolar data is handled by space-multiplexing [3] and this doubles the size of the matrices and vectors required. For the third-order problem, C = 2.48, the dynamic range is 47.7 and from (5), j = 10 iterations are required to solve each LAE. For the fifth-order problem, C = 56.9, the matrix dynamic range is 1117 and from (5), j = 100 iterations are required to solve each LAE. We consider three solutions: an iterative algorithm, direct LU Gaussian-elimination with the back-substitution performed optically and direct Gaussian-elimination with the back-substitution performed digitally with high accuracy. We consider two problems: the solution of $\underline{A}_5\underline{x}_5 = \underline{b}_5$ for the fifth and last outer loop in (2) and (3) for the solution of the ARE in (1) with $\underline{A}_5$ and $\underline{b}_5$ digitally calculated exactly, and the solution of all five LAEs for all outer loop iterations.

TABLE 2
Performance of Three Algorithms for Two Data Sets in the Solution of One System of LAEs

ALGORITHM	TEST NO.	F100 DATA SET	RESP. VARIATIONS Point Mods (%)	RESP. VARIATIONS Dets (%)	ACOUSTIC ATTEN. (dB / cm)	DET RMS NOISE (%)	$\|\Delta x\|$ (%)	$\Delta\lambda_{max}$ (%)
(I) Iterative	1	3	1	1	0.1	0.6	2.49	0.2×10^{-3}
	2	5	1	1	0.001	0.06	4.31	1.3
(II) LU and Optical Back-Substitution	3	3	1	1	0.1	0.6	2.39	0.52
	4	5	1	1	0.1	0.006	9.77	0.93
(III) LU and Digital Back-Substitution	5	3	1	1	0.1	0.6	3.04	0.33
	6	5	1	1	0.1	0.006	6.78	0.71

TABLE 3
Performance of Three Algorithms for Two Data Sets in the Solution of the Nonlinear ARE

ALGORITHM	TEST NO.	F100 DATA SET	RESP. VARIATIONS Point Mods (%)	RESP. VARIATIONS Dets (%)	ACOUSTIC ATTEN. (dB / cm)	DET RMS NOISE (%)	$\|\Delta x\|$ (%)	$\Delta\lambda_{max}$ (%)
(I) Iterative	7	3	1	1	0.1	0.6	2.98	0.77
	8	5	1	1	0.001	0.06	5.24	1.62
(II) LU and Optical Back-Substitution	9	3	1	1	0.1	0.6	4.56	0.72
	10	5	1	1	0.1	6×10^{-4}	11.34	1.44
(III) LU and Digital Back-Substitution	11	3	1	1	0.1	0.6	4.12	0.5
	12	5	1	1	0.1	6×10^{-4}	10.17	1.17

In Table 2, we show the results for the solution of the single fifth set of LAEs. Our results for the full set of five LAEs, i.e. the full ARE solutions are included in Table 3. Data sets 3 and 5 refer to the third and fifth-order F100 matrix problems respectively. The performance measures used in evaluating each approach are the average norm $||\Delta x||$ of the error in the calculated vector $\underline{x}$ and the maximum error $\Delta\lambda_{max}$ in the location of the closed-loop poles of the final system. The spatial, detector noise, and acoustic attenuation errors noted earlier were selected to produce approximately equal output errors for each error source treated separately.

In Tests 1 and 2, we see that our theoretical operational parameters (Table 1) are also valid when noise and system errors are present. Comparing the results for Algorithm I and II, we see that acoustic attenuation is the dominant error source for an iterative algorithm and detector noise dominates the performance of a direct algorithm. This is expected

because of the cyclic data flow of the matrix in the AO cell during the iterative algorithm. This alters C for the matrix. In the direct algorithm, detector noise on one cycle is fed back to both the inputs and the AO cell and thus changes the noise distribution and its effects accumulate. Also, detector noise affects the small vector elements and this effect also compounds on successive cycles. From the results of Algorithms II and III, we see that optical back-substitution yields comparable performance to digital back-substitution. This is expected, since the operations required in back-substitution are only vector inner products and only N-1 of these are required. This is a substantially lower computationally intensive set of operations than those required in the matrix decomposition. Thus, the accuracy of the matrix decomposition determines the final accuracy in our results. Comparing the results for data sets 3 and 5 and the corresponding data in Tables 2 and 3, we see that the larger matrix size and the increased number of steps required in the ARE versus the LAE solution causes the required accuracy to increase for direct algorithms more than for iterative algorithms (e.g. a lower acoustic attenuation constant α is noted to be required for the iterative ARE solution than for a direct LAE solution). We have derived a theoretical expression [6]

$$\alpha < (1/2.3LC) \tag{7}$$

for the amount of acoustic attenuation α in dB/cm allowed for convergence of an iterative algorithm, where L is the length of the AO cell in cm. From the last two columns in both tables, we see that $\Delta\lambda_{max}$ errors are significantly less than Δx errors as expected. The results in Tables 2 and 3 are in agreement with the theoretical guidelines in (7). From Test 1 and all other tests, we find that spatial errors are additive and that for small errors the percent performance scaled with the magnitude of the error. In Tables 2 and 3 and in (7), we assume that each T_B of the AO cell corresponded to 1mm and we assumed new input data to the point modulators in the AO cell to be introduced every T_B. To achieve more practical α levels, closer spacing of data packets in the cell is necessary. This can easily be obtained by scaling the values given in Tables 2 and 3. Operation of the input point modulators at a higher rate than the AO cell data [2] can also improve the α and detector noise values found in Tables 2 and 3. These initial test results are intended to provide guidelines for the efficient use of various algorithms, efficient solutions to linear and nonlinear matrix equations, and quantitative data on performance expected. Our theory, guidelines, and modeling are also appropriate for digital-optical linear algebra architectures.

8. REAL-TIME LABORATORY EXPERIMENTS

In Figure 2, we show the nine outputs from a laboratory system to iteratively solve the fifth set of LAEs for the third-order F100 model (Test 1, Table 2). The outputs are shown after 80, 400 and 640 iterations. The laboratory system used a fixed 2-D photographic mask for the matrix in place of the AO cell and 2-D space-multiplexing in place of frequency-multiplexing. To accomodate bipolar data, the matrix and vector were biased positive. This increased C to 120. The laboratory system was operated at a 10MHz data rate per channel. To facilitate easy monitoring of the system, we used $\omega = -0.125$. The number of iterations $j = nC$ required for 0.6% accuracy was calculated from (3) to be 613 iterations. Our experimental value of 640 iterations at which convergence occurred is thus in excellent agreement with theory. In the laboratory system, the mask errors were ±7.2% and these dominated other spatial system errors. The detector noise was measured as 0.4%. With these errors included in our simulator, the solution vector $\underline{x}$ was calculated, compared to the ideal theoretical $\underline{x}^*$ value and to the $\underline{\hat{x}}$ vector calculated on the laboratory system. The locations of the closed-loop poles of the system in each case were calculated and compared. The results in Table 4 show excellent agreement (0.5% accuracy or better) in the location of the poles and with the nature of the poles preserved (e.g. complex-conjugate pole pairs).

TABLE 4
Comparison of the Closed-Loop Poles Computed Theoretically and Using the Optical Laboratory System

THEORETICAL POLE LOCATIONS	OPTICAL LABORATORY COMPUTED POLES	% ERROR
-20.45 + j6.26	-20.74 + j5.88	0.5
-20.45 - j6.26	-20.74 - j5.88	0.5
-4.53	-4.53	10^{-3}

(a) 80 ITERATIONS (b) 400 ITERATIONS (c) 640 ITERATIONS

FIGURE 2
The nine photo-detectors outputs from a fixed mask OLAP at selected cycles in the iterative solution of the system of LAEs $\underline{A}_5\underline{x}_5 = \underline{b}_5$ that arise in the final loop of the solution of the nonlinear ARE

9. SUMMARY AND CONCLUSION

We have detailed a two-loop solution to the nonlinear ARE. In the iterative solution, a fixed number of iterations can be employed to achieve a given performance accuracy. A direct solution of each LAE can also be employed, however the iterative solution is faster ($100T_B$ vs. $975T_B$). Selection of the operational parameters for the two-loop algorithm were theoretically derived, verified by noise-free simulations and shown to be appropriate when system noise and errors were present. The implementation of direct and iterative solutions of LAEs on a frequency-multiplexed OLAP was detailed. A theoretical analysis of both algorithms showed that acoustic attenuation was the dominant error source in iterative algorithms and detector noise dominated direct algorithms. Our simulations verified these theoretical predictions and quantified the performance obtained with each. Our theoretical values for the amount of acoustic attenuation allowed to permit convergence of an iterative algorithm was verified by simulations. We confirmed and quantified by simulations that optical back-substitution yields comparable performance to its digital realization. Experimental verification on a laboratory system was obtained. The guidelines, and theory provided are appropriate for various other systolic processors (optical and digital) and for high-accuracy digital-optical linear algebra processors. Our nonlinear matrix solution using a fixed number of iterations is appropriate for realization on any linear algebra processor.

ACKNOWLEDGMENTS

The support of this research by the NASA Lewis Research Center (Grant NAG-3-5), the Air Force Office of Scientific Research (Grant AFOSR-79-0091) and in part by NASA Langley Research Center (Grant NAG-1-409) are gratefully acknowledged.

REFERENCES

1. Proc. IEEE, Special Issue on Optical Computing, Vol. 72, July 1984.
2. D. Casasent, "Acousto-Optic Linear Algebra Processors: Architectures, Algorithms and Applications", Proc. IEEE, Special Issue on Optical Computing, Vol. 72, pp. 831-849, July 1984.
3. D. Casasent, J. Jackson, C.P. Neuman, Applied Optics, Vol. 22, pp. 115-124, January 1983.
4. D. Casasent, J. Jackson, Proc. SPIE, Vol. 465, January 1984.
5. D. Casasent, A. Ghosh, C.P. Neuman, "A Quadratic Matrix Algorithm for Linear Algebra Processors", Submitted, J. of the Opt. Soc. of Am. A, April 1984.
6. A. Ghosh, D. Casasent, C.P. Neuman, "Performance of Direct and Iterative Algorithms on an Optical Systolic Processor", Submitted, Applied Optics, September 1984.
7. D. Casasent, A. Ghosh, C.P. Neuman, Proc. SPIE, Vol. 431, pp. 201-208, August 1983.
8. D. Casasent, A. Ghosh, "Optical Linear Algebra Processors: Noise and Error Source Modeling", To be submitted, Optics Letters, August 1984.
9. M. Carlotto, D. Casasent, Applied Optics, Vol. 21, pp. 147-152, January 1982.
10. D. Casasent, A. Ghosh, Optics Communications, Vol. 46, pp. 270-273, July 1983.
11. D. Casasent, A. Ghosh, Applied Optics, Vol. 22, pp. 3572-3578, November 1983.
12. A. Ghosh, D. Casasent, Applied Optics, Vol. 22, pp. 1795-1796, June 1983.
13. D. Casasent, C.P. Neuman, J. Lycas, Applied Optics, Vol. 23, pp. 1960-1966, July 1984.

14. FABRICATION CONSIDERATIONS FOR ACOUSTO-OPTIC SYSTOLIC PROCESSORS

SPIE-JAN'82
405-14

FABRICATION CONSIDERATIONS FOR ACOUSTO-OPTIC SYSTOLIC PROCESSORS

David Casasent and James Jackson

Carnegie-Mellon University
Department of Electrical and Computer Engineering
Pittsburgh, Pennsylvania 15213

ABSTRACT

The number of multiplications per second and fabrication issues associated with several different acousto-optic systolic processors are discussed and the flexibility in the operations achievable by format control are briefly reviewed. Emphasis is given to the effects of divergence of the optical input beam. Various input sources and interconnection schemes are considered. These include: fiber and GRIN optics, multi-channel acousto-optic cells and individually collimated laser diodes. Quantitative theoretical and experimental data are provided. A new architecture using spatial-multiplexing of the input sources and frequency-multiplexing of the acousto-optic cell data is described and used for handling bipolar and complex-valued matrix and vector elements.

1. INTRODUCTION

Optical matrix-vector processors [1,2] represent a most general-purpose class of optical system. Optical systolic array processors, especially those using acousto-optic (AO) cells [3-5,13] represent very practical systems that can be fabricated with present technology. Many interested people feel that the optics community should fabricate an optical systolic array processor rather than continue paper studies of such systems. In this paper, we address several fabrication and architectural issues associated with AO systolic array processors. In Section 2, we provide a quantitative assessment of the performance (in terms of mults/secs) possible on two different basic AO systolic processors. A new architecture using a multi-channel AO cell is described for use in cases when a higher computational rate is required. Other more advanced multi-channel AO systolic processors have been advanced elsewhere [13] and their use with digitally-encoded data for higher accuracy has also been described. In Section 3, we briefly review some of the different operations required in linear algebra and how all of the basic operations needed are possible (via format control) on the same generic optical systolic array processor.

Our initial remarks and comparisons of different architectures (Sections 2 and 3) are also flavored with practical fabrication considerations. In Section 4, we specifically address and quantify the effects of optical beam divergence on the performance of various input to AO cell interconnection techniques (Section 5) suitable for a wide variety of AO systolic array processors. In Section 6, we address several architectural issues associated with handling bipolar and complex-valued matrix and vector data. A new optical systolic array architecture is advanced for such practical applications. Our summary and conclusions then follow in Section 7.

The computational rate of an optical systolic array processor is the most discussed performance parameter of such systems. However, the flow and pipelining of operations and data in these systems is of equal importance [3], as is the ease of fabrication and the flexibility of a given architecture (Section 3). Another vital factor is that the operations possible on a given architecture must be properly arranged to solve a given problem. This generally involves much more than simply a matrix-vector multiplication. Examples of the detailed linear algebra operations required in various applications are available in the literature. The examples thusfar published include adaptive phased array radar [6], Kalman filtering [3,7], and optimal control [8]. The need for parallel algorithms suitable for optical architectures [9,10] is also of vital concern. The accuracy of optical linear algebra processors is yet a final issue requiring attention in many applications.

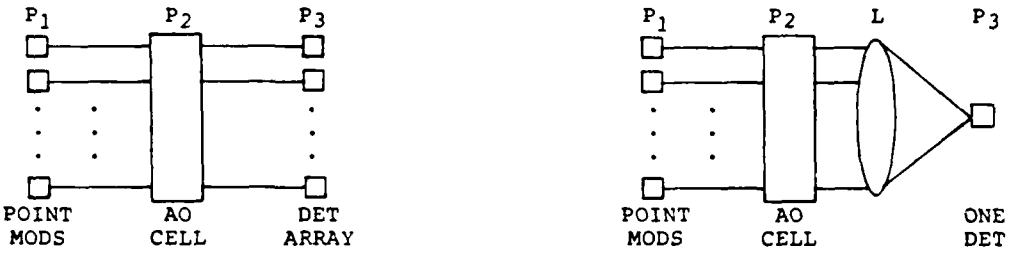
In this present paper, we restrict attention to AO-based systolic array processors, since they are the most easily fabricated architectures. We further consider only vector inner product (VIP) and matrix-vector (M-V) architectures, since such systems have 1-D output detector arrays. The high data rates from optical matrix processors are such that optical linear algebra systems requiring 2-D detector output arrays (such as vector outer product systems and certain matrix-matrix processors) pose severe output detector fabrication requirements. Specifically, a 2-D parallel readout detector array with high data readout rates is required. We also consider in this present paper only optical architectures operating on analog data. These systems represent those architectures with the highest throughput. By the use of multi-channel AO cells and various architectural changes, these basic systems we consider can be extended to operate on digital and other encoded data. Other authors have

addressed various approaches to achieving high-accuracy optical systolic array processors using various architectures and data encoding schemes. In this present paper, we will also consider only optical systems capable of operating on general matrices with no special structure. This class of system represents the most general-purpose architecture. Different architectures [3,4] are suitable for matrix problems with special matrix structure.

Our results are sufficiently general to be applicable to many AO systolic processors. The new architectures we describe can be extended (by the use of multi-channel cells and additional linear modulator arrays) to handle encoded data (for applications where higher accuracy is required). The computational rates possible from all optical systolic processors is so large that one dimension of the multiplexed systems shown can easily be used for data encoding. In such cases, the performance of the system is only reduced by a factor of 16-32 and still yields a quite significant number of mults/sec with a significantly more accurate system with fewer dynamic range constraints.

2. COMPUTATIONAL RATES

As noted in Section 1, the computational rate (mults/sec) possible is a favorite criteria (but not the panacea) for comparing optical systolic array processors. Following the terminology and motivation in [11], we now briefly compare the performance obtainable from the two generic classes of optical systolic array processors. Rhodes [11] distinguished between two types of AO systolic processors by the manner in which the AO cell was used. He refers to these AO operating modes as a modulator (Figure 1) and a deflector/modulator (Figure 2). Both approaches are self-explanatory.



(A) Integrating Detector System. (B) Single Detector Architecture.

FIGURE 1: Two Basic Acousto-Optic Modulator Vector Inner Product Processors.

The architectures in Figure 1 perform the basic operation of a VIP with one vector fed to the AO cell and the other vector fed to the input point modulators. The output from the system is the VIP. In the system of Figure 1B, the full VIP appears on one detector. In the system of Figure 1A, the product of each corresponding element of each vector is formed on separate detectors. The output detectors in Figure 1A can accumulate data or their contents can be shifted and added. These operations can be used in performing matrix-vector multiplications on a VIP processor. In the system of Figure 2, data is fed to the AO cell time and frequency-multiplexed, i.e. the cell contains 2-D or matrix data and the basic operation of the system is a matrix-vector multiplication.

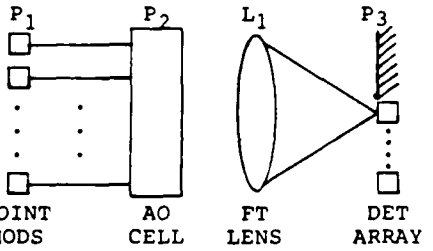


FIGURE 2: Frequency-Multiplexed Modulator/Deflector Acousto-Optic Matrix-Vector Processor.

We denote the transit time in the AO cell between two adjacent spatially-illuminated regions by the bit time T_B and the full aperture time of the cell by T_A. In all cases, efficient use of the system requires a cell with $T_A = 2NT_B$ and $2N-1$ point modulators (where N is the order of the vector or the matrix). To see this, recall that NT_B of time is required to load data into the cell and that NT_B of the cell's aperture time is required for this data. After $1T_B$ of time, the entire contents of the cell are no longer useable (since one element of the vector has now left the cell). We could recycle this element into the bottom

of the cell, but this requires additional complexity, memory and complicates data flow and feedback in general applications. Thus, we consider operation of all systems by initially (at $1T_B$) pulsing on the bottom N point modulators with one input vector, forming one VIP. Then (at time $2T_B$) pulsing on the point modulators 2 to N+1 with new vector data, etc. In this way, we multiply the vector in the cell by N different vectors before data reaches the end of the cell. Every T_B, we input new data to the cell and thus maintain full throughput in the system. In all systems, we thus assume $T_A = 2NT_B$. The architectures of Figures 1A and 1B thus perform one VIP on N element vectors every T_B or N VIPs in $T = NT_B$ (where $T_A=2T$).

The operation of the system of Figure 2 can most easily be described by viewing the contents of the AO cell as N_1 vectors (each of length N_1) with each vector on a separate frequency carrier. The data leaving the AO cell in the system of Figure 2 thus consists of N_1 VIPs on N_1 element vectors. Since the data leaving the AO cell is Fourier transformed onto the output plane in Figure 2, proper frequency-multiplexing and arranging of data can allow each of these separate N_1 VIPs to be produced in parallel on N_1 separate output detectors. When the input data to this system is properly multiplexed, a full matrix-vector multiplication is performed each T_B (this is compared to one VIP per T_B for the architectures in Figure 1). We will denote the time bandwidth product (TBWP) of the AO cell as TBWP = 2N. Thus, the systems in Figure 1 can operate on N element vectors, whereas the system of Figure 2 can operate on an $N_1 \times N_1$ element matrix (where $N_1^2 = N$). Furthermore, T_B for the systems of Figure 1 satisfies $T = NT_B$ or $N = T_A/2T_B$, whereas the system of Figure 2 requires $T = T_BN_1$ or a larger T_B since $N_1 < N$, specifically $N_1^2 = N$.

To quantitatively compare the performance and fabrication issues for these architectures, we assume an AO cell with T_A = 40μsec (T = 20μsec) and TBWP = 2000 (i.e., N = 1000). The systems in Figure 1 can thus perform

$$N \text{ mults}/T_B = 1 \text{ VIP}/T_B, \text{ or } N^2 \text{ mults}/T = 1 \text{ M-V mult}/T \tag{1}$$

i.e., one VIP every T_B or one matrix-vector (M-V) multiplication every T. This results in

$$1000^2/20\mu\text{sec} = 5 \times 10^{10} \text{ mults/sec} = 50 \text{ GOPS}. \tag{2}$$

The system of Figure 2 performs

$$N^{\frac{1}{2}} \text{ VIPs}/T_B = 1 \text{ M-V mult}/T_B = N^{\frac{1}{2}}N^{\frac{1}{2}} \text{ mults}/T_B = N \text{ mults}/T_B = NN^{\frac{1}{2}} \text{ mults}/T, \tag{3}$$

where $T = N^{\frac{1}{2}}T_B$ was used. This yields a computation rate for the system of Figure 2 of

$$1000(32)/20\mu\text{sec} = 1.6 \times 10^9 \text{ mults/sec}. \tag{4}$$

It is possible to pulse on each point modulator, in Figure 2, $N^{\frac{1}{2}} = 32$ times per T_B. In this case, (4) becomes

$$5.1 \times 10^{10} \text{ mults/sec}, \tag{5}$$

and thus both architectures can achieve the same performance for the same I/O data rate (as expected).

However, let us consider the hardware and fabrication requirements of these architectures to achieve the computation rate in (5). The architectures of Figure 1 require 2000 point modulators all packed very densely and all addressed in parallel at

$$T_B = 20\mu\text{sec}/1000 = 20\text{nsec}, \text{ or at } 50\text{MHz}. \tag{6}$$

This is a quite high data rate (and precludes A/D and D/A conversion, at a large number of bits). The very large number of sources required represents a considerable fabrication achievement. Conversely, to achieve the same performance, the system of Figure 2 requires only 64 point modulators, 32 detectors and a much lower bit time

$$T_B = (T_A/2)/N^{\frac{1}{2}} = 0.625\mu\text{sec}, \text{ or } 1.6\text{MHz}. \tag{7}$$

As seen, the bit time is significantly reduced, as is the data rate 1.6MHz versus 50MHz at which data must be fed in parallel to each channel. Thus, the ease of fabrication (64 vs. 2000 point modulators) for the frequency-multiplexed modulator/deflector matrix-vector processor of Figure 2 is quite attractive compared to the VIP architectures using AO modulators in Figure 1.

Should a given application require a higher computational rate above 50 GOPS, a multi-channel AO cell can be used (Figure 3). Practical considerations dictate that the number of multiple channels (rows of the matrix $\underline{A}$) will be less than the TBWP of each channel (the number of columns of the matrix $\underline{A}$). Frequency-multiplexing and matrix partitioning are thus quite essential to redistribute the TBWP of the multi-channel AO cell. If we have M_2 AO cell channels (each $2N_1T_B = T_A$ long), M_1 frequencies, $2N_1$ point modulators, and M_1 output arrays of M_2 detectors each, the architecture of Figure 3 realizes M_1 matrix-vector multiplications per T_B (where the matrix is $M_2 \times N_1$, i.e. one matrix per frequency, and the vector is of length N_1) i.e.

$$M_1 \text{ M-V mults/}T_B = M_1M_2N_1 \text{ mults/}T_B. \tag{8}$$

Assuming reasonable parameters: $2N_1M_1 = \text{TBWP} = 2000$, $M_1 = N_1 = 32$ and $M_2 = 100$, we find

$$10^5 \text{ mults/}T_B = 10^5/0.625\mu\text{sec} = 1.6 \times 10^{11} \text{ mults/sec} \tag{9}$$

or with 32 pulses per T_B, we obtain

$$5.1 \times 10^{12} \text{ mults/sec.} \tag{10}$$

This is equivalent to 3200 VIPs every 0.6μsec. The computation rate in (9) or (10) can be achieved with only 64 point modulators and a data rate of 1.6MHz. With vertically-oriented detector arrays, the system of Figure 3 can perform M_2 matrix-vector multiplications on $M_1 \times N_1$ matrices.

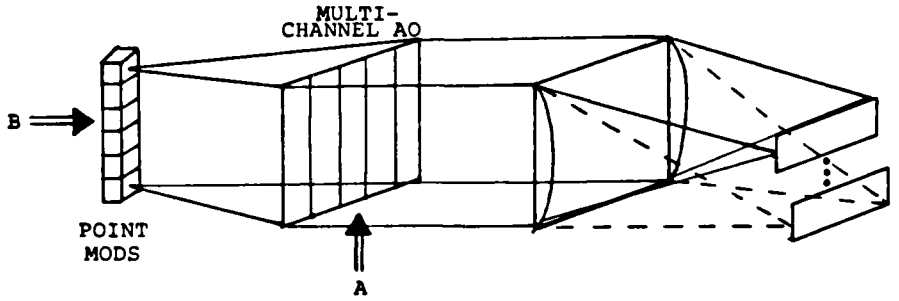


FIGURE 3: Multi-Channel AO Modulator/Deflector Frequency-Multiplexed AO systolic processor.

Many variations of the basic architecture of Figure 3 are possible and obvious. The input point modulators can be replaced by a second multi-channel AO cell. The second dimension of either modulator can be used to encode the data in digital or other representations. Alternatively, the system can be made to perform M_1 correlations per channel (this achieves M_1 digital multiplications on N_1-bit words). Finally, partial products can be accumulated by time integration on the output detectors. All of these techniques provide methods to increase the accuracy of the system and reduce dynamic range requirements (at the expense of a reduced number of mults/sec). Since few applications require the large number of operations possible in (9) and (10), such tradeoffs appear quite attractive and realistic. A detailed analysis of the architecture of Figure 3 (or similar ones) shows that such architectures are only appropriate for matrix-matrix multiplication, operation on partitioned larger-order matrices, or similarly more complicated linear algebra operations.

3. FORMAT CONTROL AND DATA FLOW FOR FLEXIBILITY

As noted in Section 1, given practical problems require far more complex operations than a VIP or even a M-V multiplication. In Table 1, we list various operations, the associated matrix and vector formatting of the data to the AO cell and the point modulators, plus where the output produced is fed back to achieve more complicated operations and applications. Each of these operations has been fully detailed in different publications, e.g., the solution of banded matrices [4], the solution of triangular systems of equations [12], general linear algebraic equation (LAE) solution by iterative or indirect algorithms [5], matrix-matrix-matrix (M-M-M) multiplication [5], matrix decomposition [9,10] for direct LAE solutions, etc. An attractive feature of the architecture of Figure 2 and all of the operations noted in Table 1 is that the data and operations flow ideally with no dead time in the system. From these brief remarks, data flow and format control are seen to provide considerable flexibility.

TABLE 1. Format Control or Data Flow for Flexibility and Data Flow.

OPERATION	NOTATION	ENCODING (ROW, COL)		FEEDBACK TO	APPLICATION
		AO CELL	POINT MODS		
M-V Multiplication	$\underline{A}\,\underline{b}$	$\underline{b} = b(t)$	$\underline{A} = a(t,x)$ (1 row per T_B)	AO	Solve Banded M-V and Triangular M-V (One Detector)
M-V Multiplication	$\underline{A}\,\underline{b}$	$\underline{A} = a(f,t)$ (1 col per T_B)	$\underline{b} = b(x)$	Point Modulators	Solve LAE
M-M Multiplication	$\underline{B}\,\underline{A}$	$\underline{A} = a(t,f)$	$\underline{B} = b(t,x)$	AO	MMM = $\underline{C}\,\underline{B}\,\underline{A}$
M-M Multiplication	$\underline{A}\,\underline{B}$	$\underline{A} = a(f,t)$	$\underline{B} = b(x,t)$	AO	MMM = $\underline{A}\,\underline{B}\,\underline{C}$ M Decomposition M Inversion Solve M Eqn

4. OPTICAL BEAM DIVERGENCE CONSIDERATIONS

As seen in Section 2, the bit time T_B is a key parameter affecting the system computation rate. As we will show below, T_B also quite significantly affects fabrication. The center-to-center spacing T_B of packets of data in the AO cell should be largely filled with the information packet (a fill ratio of 0.5 is quite practical). If we denote the physical size along the AO cell (associated with T_B) by d_B, we find

$$d_B = v_s T_B, \tag{11}$$

where v_s is the velocity of sound in the AO cell. For T_B = 0.1μsec (a 10MHz data rate per channel), d_B = 62μm (for a TeO_2 AO cell) and d_B = 657μm (for a LNB AO cell). These quantitative parameters significantly affect fabrication of the system. For N_1 = 200 and TeO_2 with T_A = 40μsec, the above d_B parameters are appropriate. Larger N_1 values (or cells with lower T_A values) will require quite smaller d_B values and will thus introduce quite significant practical fabrication problems.

Even with the above d_B values, typical point modulators have physical sizes or center-to-center spacings d_S larger than the required d_B. Thus, a demagnification of the input point modulator array (by a factor M) is required when imaging the input sources onto the AO cell, i.e. we require

$$d_B = d_S/M. \tag{12}$$

Another vital and practical fabrication issue of concern is the divergence $\pm\theta_D$ of the input light incident on each T_B packet of data in the AO cell. It is well-known that the divergence θ_D of the input light affects the frequency resolution of an AO spectrum analyzer. In the frequency-multiplexed deflector system, this affects the spacings Δf and center frequency f_C of the data. In the modulator architecture, this affects the spacing T_B of the data bits or packets. In the following paragraphs, we quantify the effect of θ_D on the performance of AO systolic processors with specific attention to different point modulator choices and different point modulator to AO cell interconnection techniques.

All AO modulators require a separation of the zero and first-order beams. This separation is $2\theta_B$ (where θ_B is the Bragg angle). If the input light has a divergence θ_D, the zero and first-order beams will also have a divergence θ_D. Thus, separation of the two beams requires

$$\theta_D \leq \theta_B. \tag{13}$$

Since θ_B satisfies

$$2\theta_B = \lambda_0/\Lambda = \lambda_0 f_c/v_s, \tag{14}$$

where λ_0 is the optical wavelength, Λ is the acoustic wavelength and f_c is the center frequency of the AO cell. Thus, we require

$$\theta_D \leq \lambda_0 f_c/v_s. \tag{15}$$

Thus, as θ_D increases, a larger f_C is required. Acoustic attenuation effects now increase. To quantify these issues, we note that for a LNB AO cell at $\lambda_0 = 820nm$

$$2\theta_{D(max)} = 2.3° \text{ to } 6.1° \tag{16}$$

as f_C varies from 300 to 800MHz. For TeO_2, as f_C varies from 40-100MHz, we require

$$2\theta_{D(max)} = 3.0° - 7.6°. \tag{17}$$

Next, we consider the effects of a divergence $\pm\theta_D$ or $\Delta\theta = 2\theta_D$ on the frequency resolution Δf and the minimum bit separation T_B. A beam divergence θ is equivalent to a spread Δf in the input RF frequency where

$$\Delta f = 2(M\theta_D')v_s/\lambda_0 = 2\theta_D d_B/\lambda_0 T_B, \tag{18}$$

where θ_D' is the divergence of the source and $M\theta_D' = \theta_D$ is the divergence of the optical beam as it enters the AO cell. This effect in (18) limits the Δf between multiplexed frequencies. Similar effects appear to be present on the minimum T_B allowed. In the conventional processors of Figure 1 (using the AO cell as a modulator), the nominal T_B is set by $T_A/2 = T = NT_B$, i.e. N packets of data can be used (where $2N = TBWP$). However, when θ_D is included, the interaction length L, the size of the AO transducer, the thickness of the AO cell and the Bragg sensitivity all enter. In general, it appears that the TBWP or the number N of bit times T_B allowed in this system is affected similarly by the presence of a $\Delta\theta$. Specifically, the Δf increase reduces the number of resolvable frequencies to $BW/\Delta f$ (where BW is the bandwidth of the device) and this correspondingly reduces the number of bit times allowable. Thus, when θ_D is present, the number of bit times allowed in the AO modulator architectures is also reduced. Since the frequency-multiplexed architecture uses a larger T_B, it is far less susceptible to this effect than are the AO modulator architectures where N (and hence the computation rate) are directly reduced as θ_D effects are included.

To quantify the magnitude of this effect, we note that for $T_B = 0.1\mu sec$ and $\lambda_0 = 0.82\mu m$, we find for LNB that $\theta_D = 1° = 17.45mrad$ requires a $\Delta f = 250MHz$ and for $\theta_D = 3mrad$ we require $\Delta f = 47MHz$. For TeO_2 (slow shear), $\theta_D = 17.45mrad$ corresponds to $\Delta f = 25.2MHz$ and $\theta_D = 3mrad$ corresponds to $\Delta f = 4.5MHz$. For LNB and TeO_2 cells with typical bandwidths, a large divergence angle of 1° thus has quite severe effects. As noted earlier, these effects on T_B are comparable. The θ_D and T_B effects are less significant for the frequency-multiplexed AO modulator/deflector architecture however.

5. SOURCES, INTERCONNECTIONS AND EXPERIMENTAL RESULTS

One attractive technique for demagnifying a linear array of point modulator sources (LEDs or laser diodes) onto the AO cell, while maintaining low divergence θ_D at the cell, is shown in Figure 4. As depicted in this figure, the point modulators are first focused into fiber optics using graded index (GRIN) optical elements (G1). The fiber optic (FO) interconnections allow the source spacings (which are generally quite large) to be reduced to the center-to-center spacing of the GRIN elements (G2) placed at the opposite end of the FO assembly as shown in Figure 4. The primary purpose of the G1 elements is to provide high-coupling efficiency from the point modulators to the fibers. The primary purpose of the FO link is to increase the packing density of the sources and to reduce the center-to-center source size. The G2 elements have the function of producing well collimated separate optical light channels incident on the AO cell. The GRIN optical element we have used have an 0.29 pitch and a 1mm diameter (for G1) and an 0.25 pitch, a 1mm center-to-center spacing with an active optical output beam diameter of 0.4mm and a $f_L = 1.1mm$ (for G2). Such an interconnection system provides parallel output beams from G2 from separate input point modulators with a corresponding $d_B = 1.0mm$ and an active beam diameter of 0.4mm. These parameters are quite compatible with the requirements for several of our AO cell systems. Additional beam reducing optics can be included between the G2 outputs and the light to the AO cell (as shown in Figure 4) to further reduce the center-to-center spacing (if this is required by the AO cell and its T_B or d_B value).

The portion of each 1.0mm diameter GRIN lens that contains light is the active source size from G2. This is set by $f_L = 1.1mm$ of G2 and N.A. = 0.19 of the fiber. With multimode fibers, with a 50μm core, the core diameter d sets the divergence from G2. For such a system, we find an active source diameter (at the input to the AO cell) of

$$d_S' = 2(N.A.)f_L = 400\mu m \tag{19}$$

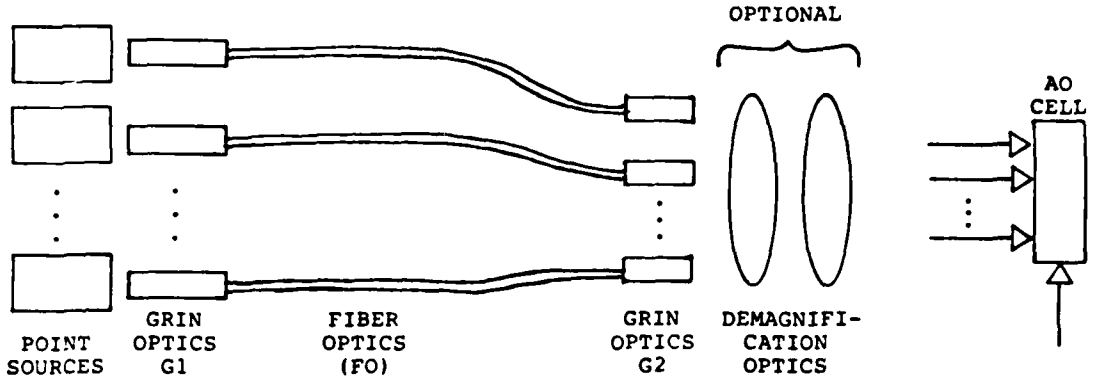


FIGURE 4: GRIN Optics Source and AO Cell Coupling Using Fiber Optics (FO).

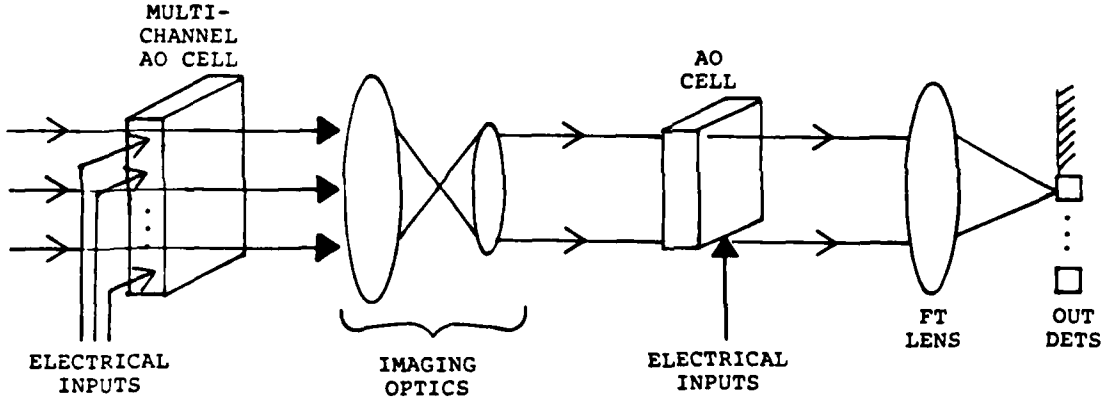


FIGURE 5: Multi-Channel Point Modulator AO Input System Architecture.

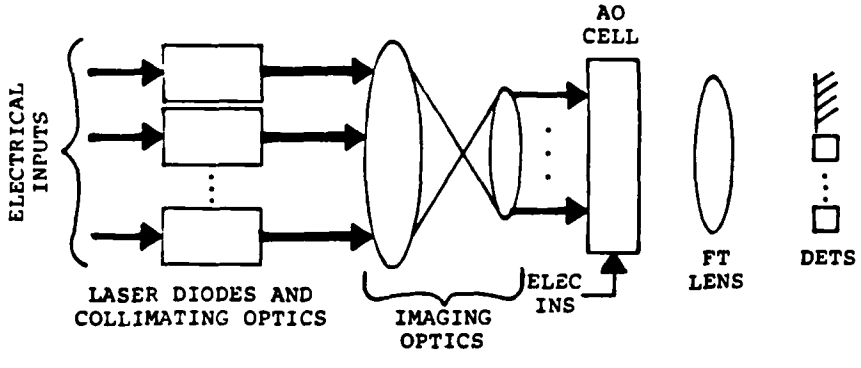


FIGURE 6: Architecture Using Separate Laser Diodes and Individual Collimating Optics.

and a beam divergence given by

$$\tan\theta_D \approx \theta_D = (D/2)/f_L = 25\mu m/1.1mm = 22mrad. \quad (20)$$

For a single-mode fiber (SMF) with a 6μm core, we obtain

$$d_S' = 350\mu m, \text{ and } \theta_D = 3mrad. \quad (21)$$

Our experimental tests have verified all of the above theoretical parameters of the two indicated interconnection architectures. Other experiments we performed verified the associated theoretical Δf associated with the various given $\Delta\theta$ values noted above.

Other possible linear point source alternatives include the use of a multi-channel AO cell (Figure 5) and the use of laser diodes with separate individual collimating optics (Figure 6). Each of these architectures represents most attractive alternatives that are appropriate for various applications. The multi-channel AO input cell architecture requires demagnification optics. It has the advantages of a very low divergence angle; however its performance is generally limited by the optical and electrical isolation achievable between the separate AO channels. The use of such an input to an optical matrix-vector processor is thus probably restricted (within the near-term) to systems employing data encoding for reduced dynamic range and improved accuracy. As multi-channel AO devices mature, such systems may become more appropriate for analog matrix-vector applications. The use of separate laser diodes with individual collimating optics for each source is quite attractive since several such units are commercially available. The divergence angle obtainable from such systems appears to be adequate to allow simple beam-reducing optics to be employed (without the need for the GRIN-FO-GRIN system in Figure 4).

6. BIPOLAR AND COMPLEX-DATA HANDLING

The issue of handling bipolar and complex-valued data in optical systolic processors has often not been detailed. In Figure 7, we show a new architecture that is appropriate for such data. For the input point sources, we spatially-multiplex two linear arrays of point modulators. With such an arrangement, we can represent bipolar data by inputing positive valued vector elements on one input array and negative valued vector elements on the other input array. Thus, which input array contains non-zero elements will determine whether the input data is positive or negative valued. For complex-valued data, three linear input arrays would be employed. As shown in Figure 7, the light from each input array passes through the AO cell at a different angle and hence the matrix-vector product of the corresponding input vector and the matrix within the AO cell appears on a separate linear output detector array (in a different vertical location). We now direct attention to the data input to the AO cell in Figure 7. In this figure, we show three multiplexed frequency inputs to the AO cell. These can be used to represent complex-valued data (by encoding such data with its projections on the 0°, 120° and 240° projections in the complex plane). For the architecture shown, a bipolar input vector is multiplied by a complex-valued matrix and the corresponding matrix-vector product is formed on separate linear output detector arrays. The post-processing required to convert this output data for feedback to the system in a compatible form is quite simple.

7. SUMMARY AND CONCLUSION

In this paper, we have described many practical fabrication issues associated with optical systolic array processors. The performance (mults/sec) of several different architectures have been quantified and compared. A frequency-multiplexed architecture was shown to require greatly simplified fabrication and to yield equivalent performance to that achieved on other architectures. We noted that by format control, many different linear algebraic operations were possible on the same architecture and that all such operations provide quite ideal data and operational flow. The effects of the bit element size at the AO cell and the divergence of the optical beam entering the AO cell were noted and quantified. Three new architectures were suggested that appear appropriate for fabrication of a realistic and practical optical systolic array architecture. These systems include the detailed issues of source size, source spacing, and fill-ratio, as well as the details of the source-to-AO-cell coupling, and the aforementioned issues of bit size and divergence. Finally, a new spatial and frequency-multiplexed architecture was described to allow handling of complex-valued and bipolar matrix and vector data.

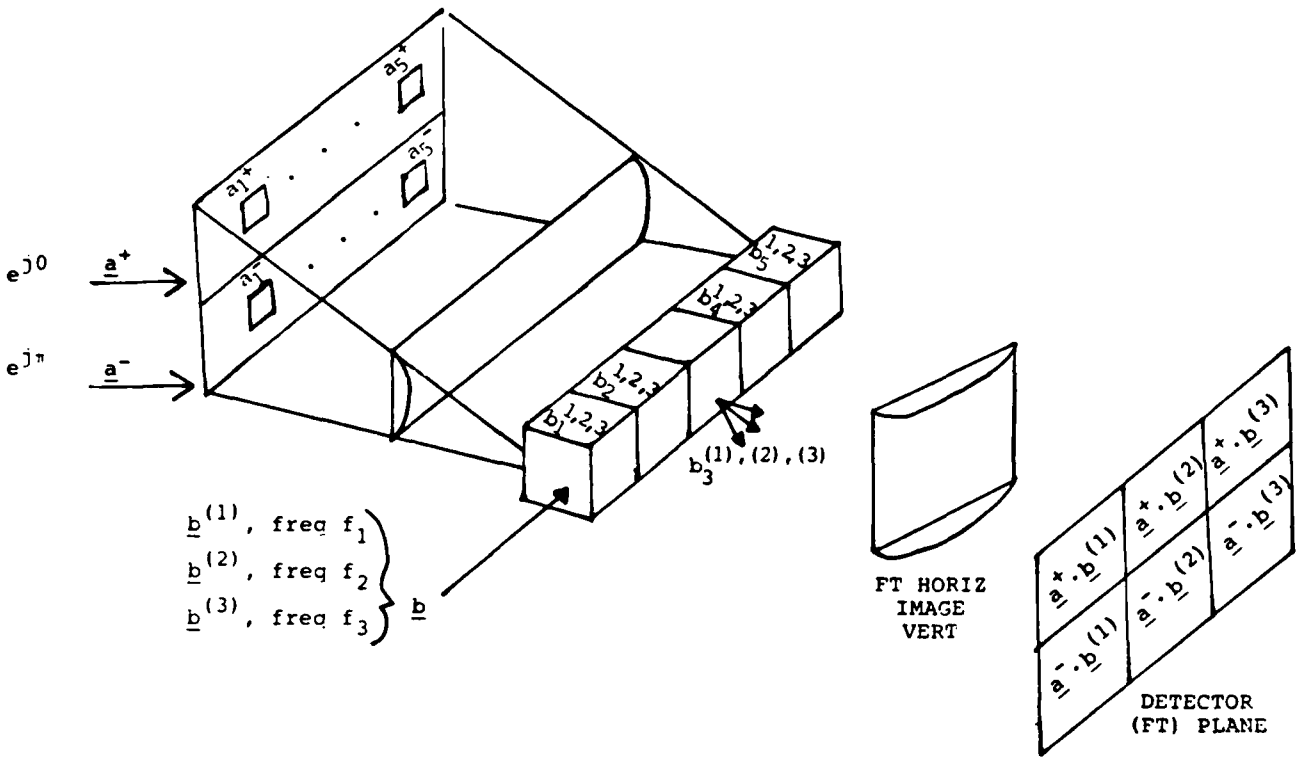


FIGURE 7: Bipolar and Complex-Valued Data Handling Using Space and Frequency-Multiplexing.

ACKNOWLEDGMENT

The support of NASA Lewis Research Center (Grant NAG-3-5) and NASA Langley Researh Center (Grant NAG-1-409), plus the support of independent contractors to Unicorn Systems Incorporated for the practical fabrication issues noted herein is gratefully acknowledged.

REFERENCES

1. J.W. Goodman, et al, Optics Letters, 2, 1 (1978).
2. M. Carlotto and D. Casasent, Applied Optics, 21, 147-152 (January 1982).
3. H.J. Caulfield, et al, Optics Communications, 40, 86 (1981).
4. D. Casasent, Applied Optics, 21, 1859-1865 (May 1982).
5. D. Casasent, J. Jackson and C.P. Neuman, Applied Optics, 22, 115-124 (January 1983).
6. D. Casasent and M. Carlotto, Optical Engineering, 21, 814 (1982).
7. D. Casasent, C.P. Neuman and J. Lycas, ICALEO'83 Conference, Los Angeles (November 1983).
8. D. Casasent, C.P. Neuman and M. Carlotto, SPIE, 295, 176-183 (August 1981).
9. D. Casasent and A. Ghosh, Applied Optics, 22, 3572-3578 (November 1983).
10. D. Casasent and A. Ghosh, Optics Communications, 46, 270-273 (July 1983).
11. W. Rhodes, SPIE, 431 (August 1983).
12. A. Ghosh and D. Casasent, Applied Optics, 22, 1795-1796 (June 1983).
13. P. Guilfoyle, SPIE, 362, 2 (1982).

15. REFERENCES

1. D. CASASENT, A. Goutzoulis and B.V.K. Vijaya Kumar, "Time-Integrating Acousto-Optic Correlator: Error Source Modeling", Applied Optics, Vol. 23, pp. 3130-3137, September 1984.

2. "Parallel Coherent Optical Processor Architectures and Algorithms for ATR", Conference on *Parallel Algorithms and Architectures*, Leesburg, VA, July 1984.

3. D. CASASENT, "Coherent Optical Pattern Recognition: A Review", Optical Engineering, *Special Issue*, Vol. 24, January 1985.

4. D. CASASENT and V. Sharma, "Fourier Transform Feature-Space Studies", Proc. SPIE, Vol. 449, pp. 2-8, November 1983 (CASASENT, Sharma).

5. D. CASASENT and V. Sharma, "Feature Extractors for Distortion-Invariant Robot Vision", Optical Engineering, Vol. 23, pp. 492-498, October 1984.

6. D. CASASENT and R.L. Cheatham, "Hierarchical Pattern Recognition Using Parallel Feature Extraction", Proc. ASME, August 1984.

7. R.L. Cheatham and D. CASASENT, "Hierarchical Fisher and Moment-Based Pattern Recognition", Proc. SPIE, Vol. 504, August 1984.

8. D. CASASENT and R.L. Cheatham, "Image Segmentation and Real-Image Tests for an Optical Moment-Based Feature Extractor", Optics Communications, Vol. 51, pp. 227-230, September 1984.

9. D. CASASENT, "Unified Synthetic Discriminant Function Computational Formulation", Applied Optics, Vol. 23, pp. 1620-1627, May 1984.

10. D. CASASENT, W. Rozzi and D. Fetterly, "Projection Synthetic Discriminant Function Performance", Submitted, Optical Engineering, Vol. 23, pp. 716-720, November 1984.

11. W.T. Chang, D. CASASENT and D. Fetterly, "SDF Control of Correlation Plane Structure for 3-D Object Representation and Recognition", Proc. SPIE, 507, August 1984.

12. D. CASASENT and A. Ghosh, "Direct and Implicit Optical Matrix-Vector Algorithms", Applied Optics, Vol. 22, pp. 3572-3578, November 1983.

13. D. CASASENT and A. Ghosh, "Direct and Implicit Optical Matrix-Vector Algorithms: Addendum", Applied Optics, Vol. 23, p. 1450, May 1984.

14. D. CASASENT, "Acousto-Optic Linear Algebra Processors: Architectures, Algorithms and Applications", Proc. IEEE, Special Issue on Optical Computing, Vol. 72, pp. 831-849, July 1984.

15. D. CASASENT, A. Ghosh and C.P. Neuman, "Direct and Indirect Optical Solutions to Linear Algebraic Equations: Error Source Modeling", Proc. SPIE, Vol. 431, pp. 201-208, August 1983.

16. D. CASASENT, C.P. Neuman and J. Lycas, "Optical Kalman Filtering for Missile Guidance",

ICALEO'83, Laser Institute of America, Vol. 41, pp. 70-78, Los Angeles, California, November 1983.

17. D. CASASENT, C.P. Neuman and J. Lycas, "Optical Kalman Filtering for Missile Guidance", Applied Optics, Vol. 23, pp. 1960-1966, July 1984.

18. J. Jackson and D. CASASENT, "A State Estimation Kalman Filter Using Optical Processing: Noise Statistics Known", Applied Optics, Vol. 23, pp. 376-378, February 1984.

19. J. Jackson and D. CASASENT, "Optical Systolic Array Processor Using Residue Arithmetic", Applied Optics, Vol. 22, pp. 2817-2821, September 1983.

20. D. CASASENT, A. Ghosh and C.P. Neuman, "Iterative Solutions to Nonlinear Matrix Equations Using a Fixed Number of Steps", Proc. SPIE, Vol. 495, August 1984.

21. D. CASASENT and J. Jackson, "Fabrication Considerations for Acousto-Optic Systolic Processors", Proc. SPIE, Vol. 465, pp. 104-112, January 1984.

22. D. CASASENT, "Linear Algebra Techniques for Pattern Recognition: Feature Extraction Case Studies", Proc. SPIE, Vol. 431, pp. 263-269, August 1983.

23. D. CASASENT, "Coherent Optical Pattern Recognition: A Review", Optical Engineering, 24, Special Issue, January 1985.

16. PUBLICATIONS AND PRESENTATIONS

16.1 PUBLICATIONS (AFOSR SUPPORTED, 1979-DATE)

Publications from 30 September 1979 - 30 September 1980 on work performed under AFOSR-79-0091 are listed in Section 16.1.1. Publications during 30 September 1980 - 30 September 1981 follow in Section 16.1.2, and publications in FY82 and FY83 continue in Sections 16.1.3 and 16.1.4. New publications from September 1983 - September 1984 follow in Section 16.1.5. A list of presentations at conferences, companies, and seminars on our AFOSR research conducted during the prior year then follow.

16.1.1 PUBLISHED PAPERS UNDER AFOSR SUPPORT (30 SEPTEMBER 1979 - 30 SEPTEMBER 1980)

1. "Photo-DKDP Light Valve in Optical Data Processing", Applied Optics, 18, 3307-3314, October 1979 (Casasent, Luu).
2. "Coherent Optical Pattern Recognition", Nikkei Electronics, 150-181, October 1979 (in Japanese) (Casasent).
3. "Optical Data Processing for Advanced Missile Guidance Needs", AIAA, October 1979 (Casasent).
4. "Spread Spectrum Optical Signal Processors", Proc. EOSD, 333-342, October 1979 (Casasent, Psaltis).
5. "Space Blur Bandwidth Product in Correlator Performance Evaluation", JOSA, 70, 103-110, January 1980 (Kumar, Casasent).
6. "Optical Image Processing", EOSD, Tokyo, January 1980 (in Japanese) (Casasent).
7. "Optical Signal Processing", EOSD, Tokyo, January 1980 (in Japanese) (Casasent).
8. "Beyond Matched Filtering", Opt. Engr., 19, 152-156, March 1980 (Caulfield et al).

9. "Multivariant Technique for Multi-Class Pattern Recognition", Applied Optics, 19, 1758-1761, June 1980 (Psaltis, Casasent).

10. "Optical Fourier Transform Techniques for Advanced Fourier Spectroscopy", Applied Optics, 19, 2034-2037, June 1980 (Casasent, Psaltis).

11. "Nonlinear t-E Curve Effects in an Optical Correlator", Opt. Commun., 34, 4-6, July 1980 (Kumar, Casasent).

12. "Correlation of Images with Random Contrast Reversals", SPIE, 238, 156-165, July 1980 (Barniv, Mostafavi, Casasent).

13. "A Laser Diode Lensless MSF-HOE Correlator", Applied Optics, 19, 2653-2654, August 1980 (Caimi et al).

16.1.2 PUBLISHED PAPERS UNDER AFOSR SUPPORT (30 SEPTEMBER 1980 - 30 SEPTEMBER 1981)

14. "Hybrid Processor to Compute Invariant Moments for Pattern Recognition", Opt. Lett., 5, 395-397, September 1980 (Casasent, Psaltis).

15. "Optical Word Recognition, Case Study in Coherent Optical Pattern Recognition", Opt. Engr., 19, 716-721, September 1980 (Casasent et al).

16. "Lensless Matched Spatial Filter Correlator Experiments", Opt. Commun., 34, 311-315, September 1980 (M. Shen et al).

17. "HOE/Lensless Matched Spatial Filter Correlator Experiments", Opt. Commun., 34, 316-320, September 1980 (M. Shen et al).

18. "A Laser Diode/Lensless MSF Optical Pattern Recognition System", EOSD, 46-52, November 1980 (Casasent et al).

19. "Optical Pattern Recognition: Matched Spatial Filter Processors", EOSD, 33-39, November 1980 (Casasent).

20. "Optical Pattern Recognition: Beyond Matched Spatial Filtering", EOSD, 39-47, March 1981 (Casasent).

21. "Pattern Recognition: A Review", IEEE Spectrum, 28-33, March 1981 (Casasent).

22. "Processing Flexibility by Hybrid Optical/Digital Techniques", Proc. Workshop of Future Directions in Optical Data Processing, Texax Tech. Rept., 1 March 1981, 17-23 (Casasent, Kumar).

23. "Beyond Holographic Matched Filtering", Israel Journal of Technology, 18, 255-260, March 1981 (Casasent).

24. "Binarization Effects in a Correlator with Noisy Input Data", Applied Optics, 20, 1433-1438, April 1981 (Kumar, Casasent).

25. "Correlation of Images with Random Contrast Reversals", SPIE, 238, 156-165, July 1980 (Barniv, Mostafavi, Casasent).

26. "Image Quality Effects in Optical Correlators", SPIE, 310, 183-192, August 1981 (Casasent, Eiva, Kumar).

27. "Multisensor Image Registration: Experimental Verification", SPIE, 292, 160-171, August 1981 (Barniv, Casasent).

28. "Intra-Class IR Tank Pattern Recognition Using SDFs", SPIE, 292, 25-33, August 1981 (Hester, Casasent).

29. "Inter-Class Discrimination Using SDFs", SPIE, 302, 108-116, August 1981 (Hester, Casasent).

16.1.3 PUBLISHED PAPERS UNDER AFOSR SUPPORT (30 SEPTEMBER 1981 - 30 SEPTEMBER 1982)

30. "An Iterative Optical Processor: Selective Survey of Operations Achievable", Proceedings NASA Langley Conference on Optical Information Processing, Publication 2207, August 1981, 105-118 (Casasent, Neuman).

31. "A Review of Optical Signal Processing", IEEE Commun., 40-48, September 1981 (Casasent).

32. "Optical Signal Processing II: Applications, Systems and New Techniques", EOSD, 41-47, September 1981 (Casasent).

33. "The Soviet Priz Spatial Light Modulator", Applied Optics, 20, 3090-3092, September 1981 (Casasent, Caimi, Khomenko).

34. "A Laser Diode/HOE Pattern Recognition System", Acta Optica Sinica, 1, 401-410, September 1981 (Casasent et al).

35. "Eigenvector Determination by Iterative Optical Methods", Applied Optics, 20, 3707-3710, November 1981 (Kumar, Casasent).

36. "A New Soviet BSO Light Modulator for Optical Data Processing", Proc. EOSD, 297-303, November 1981 (Casasent, Caimi).

37. "A Correlator for Optimum Two-Class Discrimination", Proc. EOSD, 321-330, November 1981 (Casasent et al).

38. "Test and Evaluation of the Soviet Prom and Priz Spatial Light Modulators", Applied Optics, 20, 4215-4220, December 1981 (Casasent, Caimi, Khomenko).

39. "A Microprocessor-Based Fiber-Optic Iterative Optical Processor", Applied Optics, 21, 147-152, January 1982 (Carlotto, Casasent).

40. "Principal Component Imagery for Statistical Pattern Recognition Correlators", Opt. Engr., 21, 43-47, January/February 1982 (Kumar, Casasent).

41. "Adaptive Phased Array Radar Processing Using an Optical Matrix-Vector Processor", SPIE, 341, May 1982 (Casasent, Carlotto).

42. "New Research in Holographic Pattern Recognition", Proc. SPIE, 353, 6-11, August 1982 (Casasent).

43. "Synthetic Discriminant Functions for 3-D Object Recognition", Proc. SPIE, 360, 136-142, August 1982 (Casasent, Kumar, Sharma).

44. "Multidimensional Adaptive Radar Array Processing Using an Iterative Optical Matrix-Vector Processor", Opt. Engr., 21, 814-821, September 1982 (Casasent, Carlotto).

16.1.4 PUBLISHED PAPERS UNDER AFOSR SUPPORT (30 SEPTEMBER 1982 - 30 SEPTEMBER 1983)

45. "Advanced Acousto-Optic Signal Processors", Proc. SPIE, 352, 50-58, August 1982 (Casasent).

46. "A Fisher Discriminant Approach to Distortion-Invariant Pattern Recognition Using Autocorrelations", Lasers and Electro-Optics, 34, 18-23, September 1982 (Casasent, Chang).

47. "Realization of a Sobel Operator by Coherent Optical Techniques", Lasers and Electro-Optics, 34, 24-30, September 1982 (Chen, Casasent).

48. "Applications of the Priz Light Modulator", Applied Optics, 21, 3846-3854, November 1982 (Casasent, Caimi, Petrov, Khomenko).

49. "Frequency-Multiplexed and Pipelined Iterative Optical Systolic Array Processors", Applied Optics, 22, 115-124, January 1983 (Casasent, Jackson, Neuman).

50. "Optical Linear Algebra", SPIE, 388, January 1983 (Casasent, Ghosh).

51. "Nonlinear Local Image Preprocessing Using Coherent Optical Techniques", Applied Optics, 22, 808-814, March 1983 (Casasent, Chen).

52. "Performance of Synthetic Discriminant Functions for Infrared Ship Classification", IOCC Conference, Boston, Massachusetts, April 1983, IEEE Cat. No. CH1880-4/83, SPIE Vol. 422, pp. 193-196 (CASASENT, Sharma).

53. "Guidelines for Efficient Use of Optical Systolic Array Processors", IOCC Conference, Boston, Massachusetts, April 6-8, 1983, IEEE Cat. No. CH1880-4/83, SPIE Vol. 422, pp. 209-213 (CASASENT).

54. "Recent Advances in Optical Signal Processing", CLEO Conference, May 17-20, 1983, Baltimore, Maryland (CASASENT).

55. "Developments in Acousto Optic Signal Processing", Trends and Perspectives in Signal Processing, Vol. 3, No. 2, pp. 1-6, June 1983 (CASASENT).

56. "Generalized Chord Transformation for Distortion-Invariant Optical Pattern Recognition", Applied Optics, 22, pp. 2087-2094, July 1983 (CASASENT, Chang).

57. "LU and Cholesky Decomposition on an Optical Systolic Array Processor", Optics Communications, 46, pp. 270-273, July 1983 (CASASENT, Ghosh).

58. "Direct and Indirect Optical Solutions to Linear Algebraic Equations: Error Source Modeling", Proc. SPIE, 431, pp. 201-208, August 1983 (CASASENT, Ghosh, Neuman).

59. "Linear Algebra Techniques for Pattern Recognition: Feature Extraction Case Studies", SPIE, 431, pp. 263-269, August 1983 (CASASENT).

60. "Shift-Invariant and Distortion-Invariant Object Recognition", SPIE, 442, pp. 47-55, August 1983 (CASASENT, Sharma).

16.1.5 PAPERS PUBLISHED AND SUBMITTED UNDER AFOSR SUPPORT

(SEPTEMBER 1983 - SEPTEMBER 1984)

61. "Fourier Transform Feature-Space Studies", Proc. SPIE, 449, pp. 2-8, November 1983 (CASASENT, Sharma).

62. "Direct and Implicit Optical Matrix-Vector Algorithms", Applied Optics, 22, pp. 3572-3578, November 1983 (CASASENT, Ghosh).

63. "Optical Kalman Filtering for Missile Guidance", ICALEO'83, Laser Institute of America, 41, pp. 70-78, Los Angeles, California, November 1983, (CASASENT, Neuman, Lycas).

64. "Recent Advances in Optical Pattern Recognition", Proc. SPIE, 456, January 1984 (CASASENT, Fetterly).

65. "Fabrication Considerations for Acousto-Optic Systolic Processors", Proc. SPIE, 465, pp. 104-112, January 1984 (CASASENT, Jackson).

66. "A State Estimation Kalman Filter Using Optical Processing: Noise Statistics Known", Applied Optics, 23, pp. 376-378, February 1984 (Jackson, CASASENT).

67. "Unified Synthetic Discriminant Function Computational Formulation", Applied Optics, 23, pp. 1620-1627, May 1984 (CASASENT).

68. "Direct and Implicit Optical Matrix-Vector Algorithms: Addendum", Applied Optics, 23, p. 1450, May 1984 (CASASENT, Ghosh).

69. "Acousto-Optic Linear Algebra Processors: Architectures, Algorithms and Applications", Proc. IEEE, Special Issue on Optical Computing, 72, pp. 831-849, July 1984 (CASASENT).

70. "Optical Kalman Filtering for Missile Guidance", Applied Optics, 23, pp. 1960-1966, July 1984 (CASASENT, Neuman, Lycas).

71. "Time-Integrating Acousto-Optic Correlator: Error Source Modeling", Applied Optics, 23, pp. 3230-3237, September 1984 (CASASENT, Goutzoulis, Kumar).

72. "Acousto-Optic Processor for Adaptive Radar Noise Environment Characterization", Accepted for publication, Applied Optics, 1984 (Goutzoulis, CASASENT, Kumar).

73. "Feature Extractors for Distortion-Invariant Robot Vision", Optical Engineering, 23, pp. 492-498, October 1984 (CASASENT, Sharma).

74. "Projection Synthetic Discriminant Function Performance", Optical Engineering, 23, pp. 716-720, November 1984 (CASASENT, Rozzi, Fetterly).

75. "A Quadratic Matrix Algorithm for Linear Algebra Processors", Submitted, IEEE Trans. SMC, Submitted, August 1984 (CASASENT, Ghosh, Neuman).

76. "Image Segmentation and Real-Image Tests for an Optical Moment-Based Feature Extractor", Optics Communications, 51, pp. 227-230, September 1984 (CASASENT, Cheatham).

77. "Hierarchical Pattern Recognition Using Parallel Feature Extraction", Proc. ASME, August 1984 (CASASENT, Cheatham).

78. "Hierarchical Fisher and Moment-Based Pattern Recognition", Proc. SPIE, 504, August 1984 (Cheatham, CASASENT).

79. "Iterative Solutions to Nonlinear Matrix Equations Using a Fixed Number of Steps", Proc. SPIE, 495, August 1984 (CASASENT, Ghosh, Neuman).

80. "SDF Control of Correlation Plane Structure for 3-D Object Representation and Recognition", Proc. SPIE, 507, August 1984 (Chang, CASASENT, Fetterly).

81. "Iterative Optical Vector-Matrix Processor", SPIE, 373, 111-116, February 1981 (Carlotto, CASASENT).

82. "Optical Linear Algebra Processors: Noise and Error Source Modeling", Optics Letters, Submitted September 1984, (CASASENT, Ghosh).

16.2 SEMINARS, CONFERENCES, ETC. PRESENTATIONS OF AFOSR RESEARCH (1 SEPTEMBER 1983 - 30 SEPTEMBER 1984)

October 1983

1. Washington, D.C., "Acousto-Optic Research Possibilities".

2. DARPA - Washington, D.C., "Advanced Optical Pattern Recognition Algorithms, Architectures, and Systems".

3. Carnegie-Mellon University, Sophomore Seminar - Pittsburgh, Pennsylvania, "Optical Information Processing".

November 1983

4. SPIE Conference - Cambridge, Massachusetts, "Fourier Transform Feature-Space Studies".

5. SPIE Conference - Cambridge, Massachusetts, "Direct and Implicit Optical Matrix-Vector Algorithms".

6. Laser Institute of America Conference - Los Angeles, California, "Optical Kalman Filtering for Missile Guidance".

7. VOIS Inc. - Binghamton, New York, "Optical Pattern Recognition".

8. Carnegie-Mellon University, ECE Department - Pittsburgh, Pennsylvania - "Optical Information Processing".

9. Carnegie-Mellon University, Presented to NASA Lewis - Pittsburgh, Pennsylvania, "Optical Linear Algebra".

December 1983

10. Stanford University - Stanford, "Optical Systolic Processors".

11. Chevron Oil Field Research Co. - La Habra, California, "Optical Information Processing".

12. University of California at Santa Barbara - Santa Barbara, California, "Optical Information Processing".

January 1984

13. SPIE Conference - Los Angeles, California, "Recent Advances in Optical Pattern Recognition".

14. Teledyne Electronics - Newbury Park, California, "Optical Signal Processing".

15. SPIE Conference - Los Angeles, California, "Fabrication Considerations for Acousto-Optic Systolic Processors".

February 1984

16. Polytechnic Institute - Brooklyn, New York, "Optical Processing for Robotics".

17. Robotics Institute, Carnegie-Mellon University - Pittsburgh, Pennsylvania, "Optical Information Processing".

March 1984

18. Washington, D.C., "Optical Data Processing".

19. Carnegie-Mellon University, Professional Education Program - Pittsburgh, Pennsylvania, "Optical Pattern Recognition".

20. Carnegie-Mellon University, Professional Education Program - Pittsburgh, Pennsylvania, "Optical Signal Processing".

April 1984

21. Carnegie-Mellon University, Professional Education Progra- Pittsburgh, Pennsylvania, "Optical Information Processing".

May 1984

22. Air Force Office of Scientific Research - Washington, D.C., "Optical Information Processing".

23. NASA Langley - Hampton, Virginia, "Optical Linear Algebra".

June 1984

24. Carnegie-Mellon University, Presented to Westinghouse R & D - Pittsburgh, Pennsylvania, "Center for Excellence in Optical Data Processing".

August 1984

25. SPIE Conference - San Diego, California - "Iterative Solutions to Nonlinear Matrix Equations Using a Fixed Number of Steps".

26. SPIE Conference - San Diego, California - "Hierarchical Fisher and Moment-Based Pattern Recognition".

27. SPIE Conference - San Diego, California - "SDF Control of Correlation Plane Structure for 3-D Object Representation and Recognition".

September 1984

28. Philips Laboratories - Briarcliff, NY - "Optics and Pattern Recognition in Robotics".

29. Optical Society of America - Pittsburgh, PA, "CMU Center for Excellence in Optical Data Processing".

30. Carnegie-Mellon University - Pittsburgh, PA, "Signals and Systems Research in ECE".

31. Westinghouse Corporation - Baltimore, MD, "Center for Excellence in Optical Data Processing".

16.3 THESES SUPPORTED BY AFOSR FUNDING (SEPTEMBER 1980 - SEPTEMBER 1984)

1. Hiroyasu Murakami, M.S. Dissertation, "Matched Filter Statistical Correlator (February 1981).

2. Saulius Eiva, M.S. Dissertation, "Image Quality Effects in Optical Correlators" (May 1981).

3. Charles Hester, PhD Dissertation, "Synthetic Filters for Multi-Class Pattern Recognition" (May 1981).

4. Yair Barniv, PhD Dissertation, "Multi-Sensor Image Registration" (May 1981).

5. Mark Carlotto, PhD Dissertation, "Iterative Electro-Optic Matrix Processor" (May 1981).

6. Andrew Sexton, M.S. Dissertation, "Digital Analysis of Space-Variant Optical Processors" (July 1981).

7. Bernard Szymanski, M.S. Dissertation, "A Computer-Controlled Film Recorder for Optical Processing" (July 1983).

8. Vinod Sharma, PhD Dissertation, "Design and Analysis of Algorithms for Distortion-Invariant Object Recognition" (January 1985).

9. R. Lee Cheatham, PhD Dissertation, "Moment-Based Object Recognition Using a Two-Level Classifier" (April 1984).

10. Anjan Ghosh, PhD Dissertation, "Performance Evaluation of Optical Linear Algebra Processors" (April 1984).

11. Eugene Pochapsky, M.S. Dissertation, "The Simulation of Optical Pattern Recognition Systems" (August 1984).

12. William Rozzi, M.S. Dissertation, "New Distortion-Invariant Correlator Research" (Expected in December 1984).

13. Bruce Thomas, M.S. Dissertation, "Moments for Distortion Parameter Estimation" (Expected in December 1984).

14. Wen-Thong Chang, PhD Dissertation, "Shift-Invariant and Distortion-Invariant Pattern Recognition Techniques" (Expected in February 1985).

16.4 PATENT DISCLOSURES (SEPTEMBER 1980 - SEPTEMBER 1984)

1. Multiple-Invariant Space-Variant Pattern Recognition System.

2. Pattern Recognition by Invariant Moments.

3. Synthetic Discriminant Functions for Multi-Class Pattern Recognition.

4. Equalization and Coherent Measure Correlator.

5. Multi-Variant Technique for Multi-Class Pattern Recognition.

END

FILMED

2-85

DTIC