OCTOBER 1975

CR-137758

CONTRACT NAS 2-8327

SATELLITE ON-BOARD PROCESSING FOR EARTH RESOURCES DATA

FINAL REPORT

(NASA-CR-137758) SATELLITE ON-BOARD PROCESSING FOR EARTH RESOURCES DATA Final Report (Wintek Corp., Lafayette, Ind.) 179 p HC $7.00 CSCL 05B

N76-16592

Unclas
G3/43 09579

PREPARED FOR

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

AMES RESEARCH CENTER

MOFFETT FIELD, CALIFORNIA

PRICES SUBJECT TO CHANGE

REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U. S. DEPARTMENT OF COMMERCE
SPRINGFIELD, VA. 22161

PREPARED BY

WINTEK Corp.

902 NORTH NINTH STREET

LAFAYETTE, INDIANA

47904

N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM THE BEST COPY FURNISHED US BY THE SPONSORING AGENCY. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE.

CR 137758
OCTOBER 1975 CONTRACT NAS 2-8327

Contributors:

R. E. Bodenheimer
R. C. Gonzalez
J. N. Gupta
K. Hwang
R. W. Rochelle
J. B. Wilson
P. A. Wintz

Project Director:

P. A. Wintz
WINTEK CORPORATION

Technical Monitor:

Edgar Van Vleck
NASA/ARC

SATELLITE ON-BOARD PROCESSING FOR EARTH RESOURCES DATA

*

FINAL REPORT

PREPARED FOR

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

AMES RESEARCH CENTER

MOFFETT FIELD, CALIFORNIA

PREPARED BY

WINTEK Corp.

902 North Ninth Street
Lafayette, Indiana 47904

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

NOTE:

A 28-page summary report is also available as:

Satellite On-Board Processing for Earth Resources Data--
Summary Report, CR 137757, NASA/ARC, Moffett Field, California,
94035.

0 INTRODUCTION

Most of the past effort in the field of earth resources data processing has been research oriented. Earth resources imagery has been provided by NASA to a number of researchers who have processed the data in various ways in order to determine what, if any, useful information could be extracted from the given images. These experiments have demonstrated that useful information can indeed be extracted from aircraft and satellite multispectral scanner imagery of the earth's surface. Economic studies have indicated potential cost effective systems based on these techniques. Consequently, it is anticipated that during the 1980-1990 decade earth resources satellites will be designed and flown for specific purposes, i.e., to monitor severe weather systems, to monitor water pollution, to survey and monitor world food production, etc. In these applications it may be more cost effective to process the data on-board the satellite and transmit the data products directly to the users rather than transmit the raw data to a ground processing station for generating the data products and then distributing the data products to the users via another satellite system.

The purpose of this study was to investigate the feasibility of an on-board earth resources data processor launched during the 1980-1990 time frame. Since about five years are required to design, build, check out, and launch such a system, a 1980 system would be based on 1975 technology, and a 1990 system would be based on 1985 technology.

In order to determine the feasibility of on-board processing we must first define the on-board processor. This requires that we define both the technology available for use in the design and the computational requirements required of the processor. The computational requirements depend on the algorithms that the processor must implement which in turn, depend on the data products that must be extracted from the data to satisfy the users. Consequently, in order to determine the feasibility of on-board data processors we must start with a study of projected user applications to define the data format (data throughput rate, number of spectral bands, etc.) and the information extraction algorithms the processor must implement. Based on these constraints and the constraints imposed by the available technology we can design some on-board processors and evaluate their feasibility. The study plan is summarized in Figure 0(1).

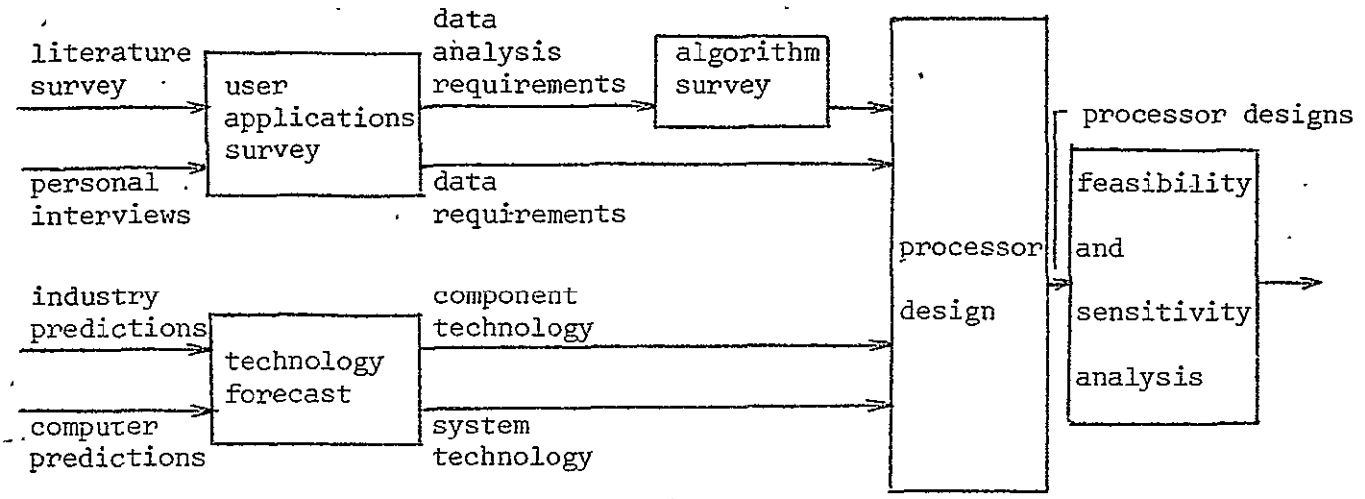


Figure 0 (1) Study Plan

A brief description of the content of each of the succeeding sections of this report follows.

Section 1. In this section we present the results of a survey of earth resources user applications and their data requirements, earth resources multispectral scanner sensor technology, preprocessing algorithms for correcting the sensor outputs and for data bulk reduction, and a candidate data format to be used in subsequent sections.

Section 2. This section contains the computational requirements required to implement the data analysis algorithms, a review of some computer architectures and organizations, a design of some computer architectures capable of handling the algorithm computational requirements, and a discussion of the on-board processor environmental effects.

Section 3. The ability of the on-board processors designed in Section 2 to implement the algorithms described in Section 1 in real time for the required throughput data rates depends on the components that will be available at the time of system design. The lead-time required for design, procurement, fabrication, checkout, and launch is about 5 years, so that 1980-1990 launches will utilize 1975-1985 technology. Consequently, we require accurate component and system technology forecasts for the next 10 years.

Section 4. This section identifies the pertinent performance parameters, isolates the independent and necessary parameters, and relates these parameters to the system requirements for each of the user requirements discussed in the

preceding sections. This will allow us to determine the feasibility of on-board processing for each user type in the 1980-1990 time frame and to perform a tradeoff analysis to determine the sensitivity of our results to each of the important system parameters.

Section 5. This section contains an overview of the entire study reported in detail in the preceding sections. Significant results and conclusions are discussed, and recommendations for future actions by NASA are presented.

Table 0(1) List of Symbols

Abbreviation	Definition
ATS	Advanced Technology Satellite
ALU	Arithmetic Logic Unit
cm	Centimeter
CPU	Central Processing Unit
CCD	Charge Coupled Device
CMOS	Complimentary Metal-Oxide Semiconductor
ERTS	Earth Resources Technology Satellite
EIF0V	Effective Instantaneous Field of View
EAROM	Electrically Alterable Read Only Memory
GSFC	Goddard Space Flight Center
HML	Hardware Maximum Likelihood
HTLU	Hardware Table Look-Up
HTC	Hybrid Technology Computer
IR	Infrared
I/O	Input/Output
IC	Integrated Circuit
I^2L	Integrated Injection Logic
IMP	Interplanetary Monitoring Platform
JTL	Josephson Tunneling Logic
km	Kilometer
LARS	Laboratory for Applications of Remote Sensing
LSI	Large Scale Integration
max	Maximum
ML	Maximum Likelihood
MTBF	Meantime Between Failure
MSI	Medium Scale Integration
MH_z	Megahertz
MOS	Metal-Oxide Semiconductor
MOSFET	Metal-Oxide Semiconductor Field Effect Transistor
m	Meter
µm	Micrometer
µP	Microprocessor
µPML	Microprocessor Maximum Likelihood
µPTLU	Microprocessor Table Look-Up--
µs	Microsecond
mv	Millivolt
mw	Milliwatt
min	Minimum
MIMD	Multiple Instruction Stream-Multiple Data Stream
MISD	Multiple Instruction Stream-Single data Stream
MSS	Multispectral Scanner
ns	Nanosecond
NASA	National Aeronautics and Space Administration
nm	Nautical Mile
n-MOS	n-Channel Metal Oxide Semiconductor
OPAU	Optimal Pipelined Arithmetic Unit
OEM	Original Equipment Manufacturer
pixel	Picture Element
pJ	Picojoule
PE	Processing Element
PROM	Programmable Read Only Memory
RAM	Random Access Memory
ROM	Read Only Memory
sec	Seconds
SPAU	Serial Pipelined Arithmetic Unit
SOS	Silicon On Sapphire
SIMD	Single Instruction Stream-Multiple Data Stream
SISD	Single Instruction Stream-Single Data Stream
SSI	Small Scale Integration
TLU	Table Look-Up
TTL	Transistor Transistor Logic
SUMC	Ultra Reliable Modular Computer
V	Volt
W	Watt

1 EARTH RESOURCES ALGORITHMS AND DATA SETS

In this section we present the results of a survey of earth resources user applications and their data requirements, earth resources multispectral scanner sensor technology, preprocessing algorithms for correcting the sensor outputs and for data bulk reduction, and a candidate data format to be used in subsequent sections.

Section 1.1 contains the results of the user requirements survey and their projected data needs in the 1980-1990 decade. The survey is based on existing literature and on personal interviews with earth resources experimenters. A survey of existing algorithms for carrying out the user requirements was also conducted. The maximum likelihood, perceptron, table look-up and clustering algorithms were examined in detail.

Section 1.2 deals with present-day and projected state-of-the-art technology relative to electro mechanical and solid-state scanners and their characteristics.

Section 1.3 contains a discussion of preprocessing algorithms for radiometric, gain, and offset corrections. Preprocessing algorithms for reducing the data bulk passed to the on-board processor using data compression and redundancy removal techniques are surveyed and analyzed.

In Section 1.4 a candidate data format is developed. This is used in later parts of the study as a baseline format for designing on-board computer architectures.

1.1 USER REQUIREMENTS

1.1.1 Applications Survey

The purpose of this section is to provide a brief survey of application areas which are most likely to be affected by remote sensing and automatic data interpretation techniques.

Two types of image-related information will be considered in the following discussion: (1) spectral information, and (2) spatial information. Spectral information is that resulting from the intensity response of a scene in the spectral bands of a multispectral scanner. Spatial information is the relationship between features in a scene. For example, the automatic classification of image elements in a lake can be easily carried out using spectral information, while the shape of the lake is best determined using spatial information.

The discussion in this section is of a general nature. Specific data requirements for each of the application areas discussed are covered in Section 1.1.2.

1.1.1.1 Agriculture (A)

Agricultural applications are receiving more attention by more investigators than almost any other application of Earth observations from space. The most important application in this area is that related to agricultural food production. The utilization of remotely-sensed data for crop acreage and production prediction will continue to play a central role in the design of sensors and processing hardware because of the social implications of this particular application.

Data processing functions related to food production include such varied tasks as:

- A1. Agricultural census
- A2. Plant species identification
- A3. Plant stress (due to insects, drought, or moisture)
- A4. Soil conservation practices
- A5. Crop yield estimates

The most challenging single technical problem associated with multispectral data analysis in crop investigations appears to be in the area of developing adequate automated machine techniques (i.e., the utilization of sample data to design automatic recognition devices). This problem will be discussed in Section 1.4.

1.1.1.2 Coastal-Zone Studies (C)

The monitoring of the physical as well as the biological environment in the coastal-zone regions is of great importance in preserving the quality of life in these regions. Data processing procedures related to this application must be able to support the following functions:

- C1. Mapping of shorelines
- C2. Mapping of shoals
- C3. Wetlands inventory
- C4. Bathymetry determination
- C5. Bottom topography studies
- C6. Mean high/low water line determination
- C7. Pollution detection

The wide range of possible features of interest, in terms of their spatial and spectral properties, requires that the minimum-sized objects of interest, with their attendant scene contrasts, be stipulated to permit an effective specification of required resolution, sampling rate, and quantization levels. Resolution and sample rate is largely determined by the minimum feature size and quantization level is determined by the multispectral analysis techniques used for identification and classification.

1.1.1.3 Forestry (F)

This application is in many respects similar to the agricultural problem. It is generally expected that a remote sensing system based primarily on multispectral analysis which will perform most agricultural survey problems will also be capable of solving many of the forestry survey problems. Some of the most important applications of remote sensing in forestry are:

F1. Forest-nonforest delineation
F2. Forest typing
F3. Detection of forest fires
F4. Plant stress detection

These applications span a wide spectrum of sensor requirements. For example, forest-nonforest delineation can be accomplished with fairly coarse resolution, while individual tree counts and classification would require resolution in the order of one meter or less. It is doubtful that satellites in the foreseeable future will possess the data requirement capabilities to solve the latter problem.

1.1.1.4 Geography (G)

Some of the major applications of remote sensing to geography are:

G1. Land-use change
G2. Earth resources location
G3. Delineation of urban/rural areas
G4. Detailed urban structure
G5. Traditional map preparation

Of these, the ones related to the inventory and classification of man's activities are receiving the most attention. Because most of the above application areas involve not only spatial and spectral signature investigations but also generic pattern recognition, the data processing

requirements tend to be very complex. Also, for political reasons, there will likely be a need for archival copies of portions of the imagery rather than summary data expressed in numerical form. These requirements limit the applicability of on-board data processing to these areas.

1.1.1.5 Geology (L)

Some of the most important applications of remote sensing to geology are:

L1. Structural Geology (faults, folds, lineaments)
L2. Geomorphology (landform classification)
L3. Lithologic mapping
L4. Geologic hazards
L5. Landslides
L6. Volcano studies

The information required for most geological investigations will include both spatial and spectral information. However, in some cases only spatial information will be of use. Examples of this would be cases where the spectral information existing in the mineralogy of the viewed area is obscured by vegetation or snow. In other cases, spatial information may be secondary as, for example, in the case of relatively featureless terrain.

The potential exists for partially automated analysis of the complex interrelated spectral and spatial information which is of significance in geological surveys. For example, certain lineaments and various other patterns could, in principle, be detected automatically. Some limited spatial frequency analysis has been attempted for geological investigations. Although this type of analysis has the potential for automation, it only represents a limited sector of all image interpretation which is of interest in geological studies. Automating the geological survey problem will undoubtedly be much more challenging than automating other applications of remote sensing such as crop surveys because of the frequent importance of simultaneous processing of both spatial and spectral information.

1.1.1.6 Hydrology (H)

Because of the importance of water-resources data acquisition to many users and government agencies, present and anticipated Earth resources

missions will find significant application of remote sensing to this area. Some of the high priority hydrologic applications are:

H1. Delineation of land-water boundaries

H2. Delineation of hydrologically-related terrain hectares

H3. Hydrodynamics, including floods, reservoirs, and estuaries

H4. Water quality evaluation

H5. Snow cover and run-off evaluation

Black-and-white infrared photography and near-infrared scanner imagery have proven to be extremely effective in detecting water surrounded by land. In infrared and near-infrared imagery, water appears black because of its absorption of solar energy in these wavelengths. The high contrast with the surroundings makes it possible to easily detect bodies of water such as lakes, rivers, streams, and reservoirs. Coupled with the repetitive coverage offered by a spacecraft, such a remote sensing technique could be utilized to monitor changes in the boundaries of surface water.

In the infrared and the near-infrared bands, healthy green vegetation reflects strongly and can be readily distinguished from water. Sensors operating in these ranges can therefore be useful in detecting water/vegetation interfaces. One particularly notable application of this feature is the delineation of wetlands areas. Wetlands are difficult or nearly impossible to map by conventional ground survey methods or by regular black-and-white or color photography.

It is evident that considerable variation in data processing requirements exist for the various hydrology applications listed above. Some can be expected to be accomplished by gross evaluation of the presence or absence of water. Such examples may relate to evaluation of regional water resources and various other large-scale water inventory applications. In these examples no firm requirement exists for precise, geometric location data or for high spatial-frequency data content. Hence, these gross evaluation applications appear to be prime candidates for on-board data processing.

Other hydrology applications, however, may require somewhat precise location data as well as spatial or textural data. Among these applications are flood hazard evaluation and precision mapping of submerged land forms. These applications will generally require the use of spatial and spectral data.

1.1.1.7 Meteorology (M)

Two of the principal areas where remote sensing is expected to yield results of metereological importance are:

M1. Cloud cover survey

M2. Prediction and assessment of natural disasters

The information required for cloud cover survey is the three-dimensional structure of the cloud cover of the globe. At first glance, the scene radiance present in the features of interest (clouds) seems to lend itself well to a fairly narrow dynamic range or nonlinear quantization, as most cloud cover images commonly published are relatively bright with respect to the underlying land or ocean surface. When the total range of lighting conditions (from a few hundred to nearly 10^4 foot-lamberts) and the apparent demand by some users for good radiometric resolution over the full range are considered, this prospect may diminish in importance. The relatively low spatial resolution required does offer good potential for optimization of resolution, quantization level, and sample rate over the very wide scan field. However, as the spatial resolution requirements are specified at the point of swath contiguity, existing sensors produce excessive spatial resolution at the nadir point which might profitably be traded for radiometric accuracy. Some of the relatively simple applications are appropriate for near-term automated data processing. One of these would be that of percent cloud cover evaluation. This can be accomplished by multispectral analysis and area integration.

Disasters resulting from hurricanes, tornadoes, flash floods, etc., could be predicted more accurately and damage assessed more quickly by employing remote-sensing technology. A new insight into the behavior of severe weather has been gained from the analysis of ATS satellite time-lapse photography, radar, and motion picture photography of tornado cloud tops taken from aircraft flying at about 15 km. A telltale "cloud turret" rises from the cirrus cloud shield, punctures the tropopause, collapses, and falls back into the smooth top of the cirrus shield. Then, another turret pops up. This tornado associated phenomena is shortlived, lasting 8 to 20 minutes.

These new scientific findings point the way to use remote-sensing technology for the observation of severe weather. An area scanner capability to view the whole earth every 20 minutes needs to be added to a

geostationary satellite. This area scanner would be pointed toward an active weather area (approximately 800 x 1000 km) and view that specific area every 2 or 3 minutes. Targets of 200 to 400 meters should be distinguishable. Such data would allow meteorologists to observe the turret cloud phenomena and refine his severe weather warnings both in time and location.

However, pin-pointing the occurrence of severe weather more accurately still cannot prevent the disaster. Once it has occurred, other remote-sensing technology can be used for damage assessment. Side-looking aircraft radar (SLAR) is an example. The geostationary area scanner might be used for this purpose. With another set of optics having higher resolution, smaller areas might be viewed for damage assessment after the clouds have cleared. In an area measuring approximately 80 x 100 km, those features approximating 80 to 100 meters in size could be imaged as a first gross assessment of the destruction. Damage assessment data can be used for many purposes: emergency routing of traffic (especially rescue vehicles), evacuation steps, estimates of emergency housing needs, and location of emergency food stations. The application of sensor technology to the disaster problem is a matter of timeliness in acquiring the observational data in usable form.

1.1.1.8 Global Oceanography (O)

The needs of the community of ocean researchers for synoptic environmental data are naturally divided into two categories: (1) the coastal environment and, (2) the global oceans. The former was discussed in Section 1.1.1.2. Some of the most important applications of remote sensing to global oceanography are:

01. Study of biological processes
02. Sea-ice surveillance
03. Study of current patterns

Assessment of features of biological significance in the more productive waters of the global oceans (approximately 10 percent of the total global area) require resolution on the order of 1 km and remotely acquired signatures of a quality that will permit the determination of chlorophyll to within a factor of two for concentrations of 0.2 mg/m^3 or greater. Observations of chlorophyll in the open oceans for concentrations as low

as 0.02 mg/m^3 are important for global ecosystem analysis. Pollution detection and environmental impact are particularly important in monitoring the natural and cultural stress induced in the coastal region.

Space techniques have demonstrated an immediate sensor application for observation of the polar environment. Resolution as large as 10 km may be used to delineate major boundaries and ice movements. However, the most pressing problem is to define, monitor, and forecast the amount and location of open water in polar regions, in particular, the Arctic. Resolution on the order of 30-100 meters is needed, with a preference for infrared over optical observations because of the nonavailability of solar illumination during the winter months.

Currents on a global scale can be mapped with relatively low resolution. Spatial dimensions of 10 km in extent would be meaningful for ocean applications. However, higher resolution may be desired to permit sampling between cumulus clouds. In general, the use of ocean color to monitor currents and biological and ecological features requires high sun elevation angles and a scan which looks away from the sunside of the spacecraft.

REFERENCES

1. Advanced Scanners and Imaging Systems for Earth Observations, NASA Technical Report SP-335, 1973.
2. Vachon, R. I., Gonzalez, R. C., et al, ERISTAR-Earth Resources Information Storage, Transformation, Analysis, and Retrival, NASA Report CR-61392, 1972.
3. Colwell, R. N., Monitoring Earth Resources from Aircraft and Spacecraft, NASA Report SP-275, 1971.
4. Park, A. B., "User Needs in Agriculture and Forestry," Proc. of the Conf. on Aerospace Methods for Revealing and Evaluating Earth's Resources, Princeton Univ., Princeton, N. J., Sept., 1969.
5. Remote Sensing of Earth Resources, A compilation of papers prepared for the 13th meeting of the Panel of Science and Technology, Committee on Science and Astronautics, U. S. House of Representatives, January, 1972.
6. Remote Sensing of Earth Resources: A Literature Survey with Indexes, NASA Report SP-7036, 1972.
7. Katz, A. M., "Useful Applications of Earth-Oriented Satellites," National Academy of Sciences, Vol. 6, 1969.

1.1.2 Data Requirements Survey

The following sections examine the data requirements of the applica-

tion areas discussed in Section 1.1.1. All discussions are keyed to the identifying labels introduced in that section. For example, general agricultural applications will be referred to as A, while specific areas within agriculture will be identified by A followed by a number.

1.1.2.1 Number of Spectral Bands

Figure 1.1.2(1) summarizes the spectral band requirements of the eight general areas discussed in Section 1.1.1. This figure vividly illustrates the differences in needs of these areas. Only in the thermal IR region (0.8 to 1.4 microns) does there seem to be a real consistency of needs.

1.1.2.2 Repeat Coverage

Typical minimum rates of coverage for various resolution requirements are shown in Figure 1.1.2(2). The implications of these rates on the volumes of data generated are discussed in Section 1.1.2.4.

1.1.2.3 Spatial Resolution and Field of Coverage

The unit of resolution used in this study is EIFOV (Effective Instantaneous Field of View) which is defined as "the minimum linear dimension on the surface (at nadir) at which user specified characteristics can be discovered." Field of coverage is defined as the swath width for a nadir pointing sensor. Typical resolution and field of coverage requirements for the application areas discussed in Section 1.1.1 are summarized in columns two and three of Table 1.1.2(I), Section 1.1.2.4.

1.1.2.4 Data Rates

The data rate D_R (in bits/sec) for each of the application areas discussed in Section 1.1.1 may be calculated using the following relation:

$$D_R = \frac{S_W}{R_L} \times \frac{V}{R_P} \times N_C \times N_{bp}$$

where

S_W = swath width (m)
R_L = resolution along a scan line (m)
V = satellite ground track velocity (m/sec)
R_P = resolution along scan path (m)
N_C = number of channels
N_{bp} = number of bits per pixel (picture element) (bits)

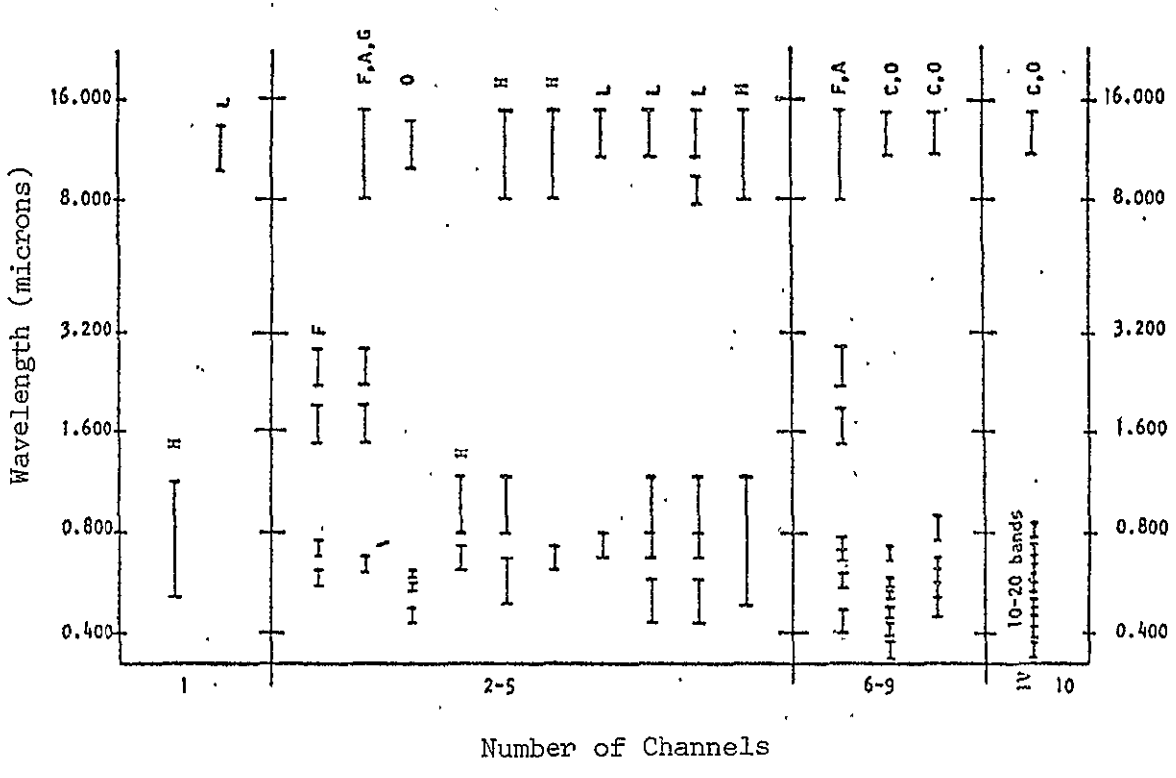


Figure 1.1.2(1) Spectral Requirements Summary

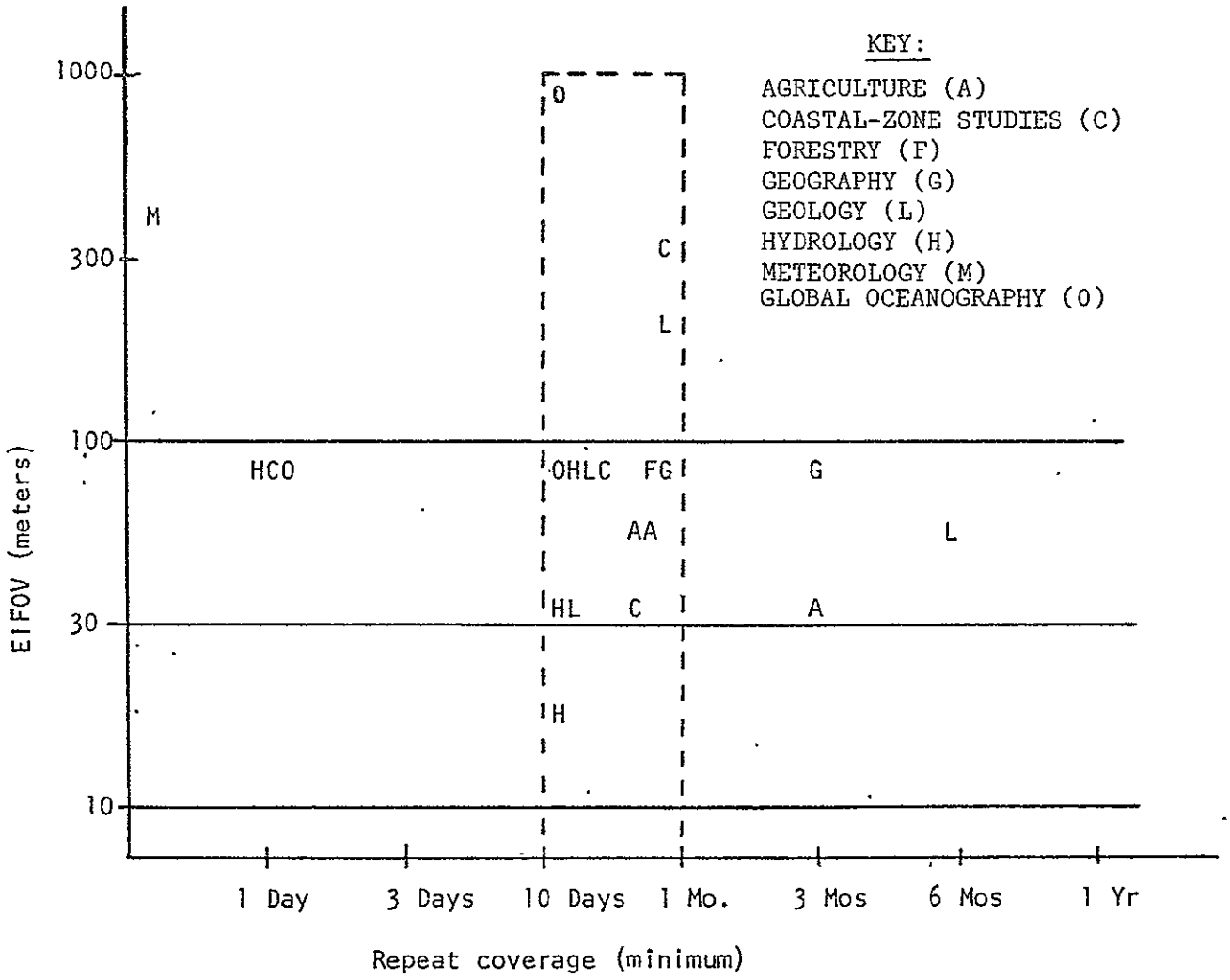


Figure 1.1.2(2). Typical Rates of Coverage

Table 1.1.2(I) shows typical data rate ranges for the application areas discussed in Section 1.1.1. All calculations are based on the following figures:

S_W: selected from Figures 1.1.2(1) and 1.1.2(2).

$R_L = R_P$ - resolution given in Figures 1.1.2(1)-1.1.2(2).

V = 6500 meters/sec.

N_C: selected from Figure 1.1.2(1).

N_{bp} = 6, which gives a range of 64 gray levels for each picture element.

Table 1.1.2(II) indicates the number of classes for each application area.

Table 1.1.2(I) Typical Data Rate Ranges

Application	Resolution (m) min-max	Field of Coverage (km) min-max	No. of Channels min-max	Data Rates (M bits/sec) min-max
A1	30-50	185	4-7	11.5- 56.1
A2	30-50	185	4-7	11.5- 56.1
A3	30-50	185	4-7	11.5- 56.1
A4	10-30	50	4-7	8.7- 137.0
A5	30-50	185	4-7	11.5- 56.1
C1	30-50	200	6-20	18.7- 173.0
C2	30-50	200	6-20	18.7- 173.0
C3	30-50	200	6-20	18.7- 173.0
C4	50-100	200	6-20	4.7- 62.4
C5	50-100	200	6-20	4.7- 62.4
C6	3-10	40	6-20	93.6-3470.0
C7	30-300	200	6-20	.5- 173.0
F1	50-100	185	4-7	2.9- 20.2
F2	5-10	15-30	4-7	23.4- 328.0
F3	10-30	185	4-7	32.1- 505.0
F4	30-50	185	4-7	11.5- 56.1
G1	30-50	185	4	11.5- 32.1
G2	30-50	185	4	11.5- 32.1
G3	50-100	185	4	2.9- 11.5
G4	5-10	15-30	4	23.4- 187.0
G5	5-10	15-30	4	23.4- 187.0
L1	50-80	185	1-5	1.1- 14.7
L2	50-80	185	1-5	1.1- 14.4
L3	50-80	185	1-5	1.1- 14.4
L4	50-80	185	1-5	1.1- 14.4
L5	10-30	15	1-5	.7- 29.3
L6	100-200	185	1-5	.2- 3.6
H1	40-60	200	1-3	2.2- 14.6
H2	30-50	200	1-3	3.1- 26.0
H3	10-30	50	1-3	2.2- 58.5
H4	30-70	200	1-3	1.6- 26.0
H5	50-80	200	1-3	1.2- 9.4
M1	200-400	800	2	.4- 1.6
M2	200-400	800	2	.4- 1.6
O1	1-10km	400	4-20	0.0* 0.3
O2	30-100	200	4-20	3.1- 173.0
O3	1-10km	200	4-20	0.0** 0.2

* 624 bits/sec **312 bits/sec

Table 1.1.2(II) List of Applications

A1	Agricultural Census
A2	Plant Species Identification
A3	Plant Stress (Due to Insects, Drought, or Moisture)
A4	Soil Conservation Practices
A5	Crop Yield Estimates
C1	Mapping of Shorelines
C2	Mapping of Shoals
C3	Wetlands Inventory
C4	Bathymetry Determination
C5	Bottom Topography Studies
C6	Mean High/Low Water Line Determination
C7	Pollution Detection
F1	Forest-Nonforest Delineation
F2	Forest Typing
F3	Detection of Forest Fires
F4	Plant Stress Detection
G1	Land-Use Change
G2	Earth Resources Location
G3	Delineation of Urban/Rural Areas
G4	Detailed Urban Structure
G5	Traditional Map Preparation
L1	Structural Geology (Faults, Folds, Lineaments)
L2	Geomorphology (Landform Classification)
L3	Lithologic Mapping
L4	Geologic Hazards
L5	Landslides
L6	Volcano Studies
H1	Delineation of Land-Water Boundaries
H2	Delineation of Hydrologically-Related Terrain Hectares
H3	Hydrodynamics, Including Floods, Reservoirs, and Estuaries
H4	Water Quality Evaluation
H5	Snow Cover and Run-Off Evaluation
M1	Cloud Cover Survey
M2	Prediction and Assessment of Natural Disasters
O1	Study of Biological Processes
O2	Sea-Ice Surveillance
O3	Study of Current Patterns.

REFERENCES

1. Advanced Scanners and Imaging Systems for Earth Observations, NASA Technical Report SP-335, 1973.
2. Colwell, R. N., Monitoring Earth Resources for Aircraft and Spacecraft, NASA Report SP-275, 1971.
3. Katz, A. M., "Useful Applications of Earth-Oriented Satellites", National Academy of Sciences, vol. 6, 1969.
4. Remote Sensing of Earth Resources: A Literature Survey with Indexes, NASA Report SP-7036, 1972.
5. Park, A. B., "User Needs in Agriculture and Forestry", Proc. of the Conf. on Aerospace Methods for Revealing and Evaluating Earth Resources, Princeton Univ., Princeton, N. J., Sept., 1969.

1.1.3 Algorithms Survey

The response of a multispectral scanner at any sampling time may be arranged in the form of a measurement or pattern column vector

$$\underline{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_k \\ \vdots \\ x_n \end{pmatrix} \qquad (1.1.3\text{-}1)$$

where x_k is the amplitude response of the kth channel in the system and n is the number of channels.

Given M categories or classes of desired classification (such as M crop types, for example), one of the principal approaches in pattern recognition system design is to determine from representative (training) data M decision functions $d_1(\underline{x})$, $d_2(\underline{x})$, ..., $d_M(\underline{x})$ with the property that if an observation $\underline{x}$ belongs to the ith category, then

$$d_i(\underline{x}) > d_j(\underline{x}) \quad \text{for all } j \neq i \qquad (1.1.3\text{-}2)$$

Once the decision functions have been estimated, they are used as the basis for automatic data classification [1]. Thus, given a sample $\underline{x}$ of unknown origin, the sample is assigned to the ith category if $d_i(\underline{x})$ yields the largest value upon substitution of the observation into all functions. Ties are resolved arbitrarily.

The boundary between the ith and jth classes is given by values of $\underline{x}$ for which $d_i(\underline{x}) = d_j(\underline{x})$. In other words, the equation of the boundary separating these two classes is given by

$$d_i(\underline{x}) - d_j(\underline{x}) = 0 \qquad (1.1.3\text{-}3)$$

For M classes, there are M(M - 1)/2 such boundaries. The first two algorithms discussed below are based on the decision-function concept.

1.1.3.1 Maximum-Likelihood Algorithm

A maximum-likelihood decision rule is one in which the decision functions are of the form

$$d_k(\underline{x}) = p(\underline{x}/\omega_k)\, p(\omega_k) \qquad k = 1, 2, \ldots, M \qquad (1.1.3\text{-}4)$$

where ω_k denotes the kth category, $p(\underline{x}/\omega_k)$ is the multivariate probability density function of the samples belonging to category ω_k, and $p(\omega_k)$ is the probability of occurrence of ω_k.

The form of $d_k(\underline{x})$ in Eq. (1.1.3-4) is determined by the nature of $p(\underline{x}/\omega_k)$ and $p(\omega_k)$. When $p(\underline{x}/\omega_k)$ is the normal density, it can be shown [1] that Eq. (1.1.3-4) may be expressed in the form

$$d_k(\underline{x}) = \ell n\ p\ (\omega_k) - \frac{1}{2}\ \ell n|\underline{C}_k| - \frac{1}{2}\ [(\underline{x} - \underline{m}_k)'\underline{C}_k^{-1}(\underline{x} - \underline{m}_k)] \qquad (1.1.3\text{-}5)$$

where ℓn is the natural logarithm, $\underline{C}_k$ and $\underline{m}_k$ are the covariance matrix and mean vector of the samples of category ω_k, and $|\underline{C}_k|$ is the determinant of $\underline{C}_k$. The prime (′) in Eq. (1.1.3-5) indicates transposition.

The parameters of the decision function shown in Eq. (1.1.3-5) are $p(\omega_k)$, $\underline{C}_k$, and $\underline{m}_k$. Once estimated, these components completely specify the decision function of each category. Studies with multivariate remotely-sensed data indicate that the normal density assumption is valid for numerous classification tasks [2].

1.1.3.2 Perceptron Algorithm

Decision functions based on the perceptron approach are of the form

$$\begin{aligned} d_k(\underline{x}) &= w_{k1}\phi_1(\underline{x}) + w_{k2}\phi_2(\underline{x}) + \ldots + w_{kN}\phi_N(\underline{x}) + w_{k,N+1} \\ &= \underline{w}_k'\underline{\phi}(\underline{x}) \end{aligned} \qquad (1.1.3\text{-}6)$$

where $\underline{w}_k = (w_{k1}, w_{k2}, \ldots, w_{k,N+1})'$ is the parameter vector, and $\underline{\phi}(\underline{x}) = (\phi_1(\underline{x}), \phi_2(\underline{x}), \ldots, \phi_N(\underline{x}), 1)'$. The functions $\phi_i(\underline{x})$ are real, single-valued functions of the patterns $\underline{x}$. Note that Eq. (1.1.3-6) can represent any nonlinear function of finite degree, depending on the choice of $\underline{\phi}(\underline{x})$. For example, a linear decision function may be constructed by letting N = n (see Eq. (1.1.3-1)), and $\phi_i(\underline{x}) = x_i$. In this case, Eq. (1.1.3-6) becomes

$$d_k(\underline{x}) = w_{k1}x_1 + w_{k2}x_2 + \ldots + w_{kn}x_n + w_{k,n+1} \qquad (1.1.3\text{-}7)$$

From Eq. (1.1.3-3), the equation of the boundary between the ith and jth pattern classes is given by

$$d_i(\underline{x}) - d_j(\underline{x}) = (w_{i1} - w_{j2})\, x_1 + \ldots + (w_{in} - w_{jn})\, x_n + (w_{i,n+1} - w_{j,n+1}) = 0 \qquad (1.1.3\text{-}8)$$

which is the equation of a hyperplane in n-dimensional space. More complex functions may be constructed by varying the degree of nonlinearity of the functions $\{\phi_i(\underline{x})\}$.

In applying Eq. (1.1.3-6) to a classification task, the functions $\{\phi_i(\underline{x})\}$ are specified. The problem then becomes the estimation of the coefficients for each class. The procedure normally followed is to use sample patterns from each class to determine these coefficients. One of the simplest ways to accomplish this task is by means of the following basic perceptron algorithm.

Given M pattern classes, $\omega_1, \omega_2, \ldots, \omega_M$, assign arbitrary initial values to the coefficient vectors $\underline{w}_1, \underline{w}_2, \ldots, \underline{w}_M$. Then, at the kth iterative step in the algorithm, if a pattern $\underline{x}(k)$ belongs to class ω_i and

$$d_i[\underline{x}(k)] > d_j[\underline{x}(k)] \quad \text{for all } j \neq i \qquad (1.1.3\text{-}9)$$

where $d_i[\underline{x}(k)] = \underline{w}_i'(k)\underline{\phi}[\underline{x}(k)]$, then

$$\underline{w}_j(k+1) = \underline{w}_j(k) \qquad \text{for } k = 1, 2, \ldots, M \qquad (1.1.3\text{-}10)$$

On the other hand, if $\underline{x}(k)$ belongs to class ω_i and for some ℓ

$$d_i[\underline{x}(k)] < d_\ell[\underline{x}(k)] \qquad (1.1.3\text{-}11)$$

then the following adjustments are made on the ith and ℓth coefficient vectors

$$\underline{w}_i(k+1) = \underline{w}_i(k) + c\,\underline{x}(k)$$
$$\underline{w}_\ell(k+1) = \underline{w}_\ell(k) - c\,\underline{x}(k) \qquad (1.1.3\text{-}12)$$

The other coefficient vectors remain unchanged. The factor c is a positive correction increment.

Basically, what this algorithm does is to change the parameter vectors only when an error in classification occurs. The procedure is said to have converged when an entire iteration through all sample patterns produces no errors. Several illustrations of this algorithm may be found in reference [1].

1.1.3.3 Table Look-Up

The table look-up approach is based on prestoring in fast, random-access core memory the desired answer (e.g., crop-type) for all combinations of multispectral scanner outputs from selected channels [3-6]. Specifically, each set of measurements from a given point on the ground is interpreted as that address in core memory where the answer can be retrived. Substituting the simple retrieved operation for the lengthy computations required by the conventional approach offers two advantages:

(1) The processing time is reduced by more than an order of magnitude.

(2) The multispectral scanner data can be processed by computers having minimal sophistication, complexity, and cost.

These two advantages may make it possible to use an on-board computer to perform the classification function in flight. A general approach to table look-up is given below.

Two Dimensional Case

All computer based systems for classifying MSS data operate by partitioning the multidimensional measurement space into non-overlapping regions associated with each known class. Measurements which are spectrally dissimilar to all of the known classes are regarded as belonging to the so-called threshold class. From Figure 1.1.3(1) it is clear that a pixel* with the particular measurement vector $\underline{\hat{x}} = (\hat{x}_1, \hat{x}_2)'$ should be assigned to Class 1 provided the following equations are satisfied

$$L_1(1) \leq x_1 \leq H_1(1) \qquad (1.1.3\text{-}13)$$

$$L_2(1,\hat{x}_1) \leq \hat{x}_2 \leq H_2(1,\hat{x}_1) \qquad (1.1.3\text{-}14)$$

$L_1(1)$ and $H_1(1)$ are the minimum and maximum values $\hat{x}_1$ can have to be associated with Class 1. Similarly, the quantities $L_2(1,\hat{x}_1)$ and $H_2(1,\hat{x}_1)$ are the minimum and maximum values $\hat{x}_2$ can have for the specific case $x_1 = \hat{x}_1$ to be associated with Class 1.

From this example, it is clear that the procedure defined by Figure 1.1.3(2) can be used to decide whether or not a pixel having measurement vector $x = (x_1, x_2)'$ should be assigned to class C. The values of the

* The term pixel is used often in image processing to denote an image or picture element.

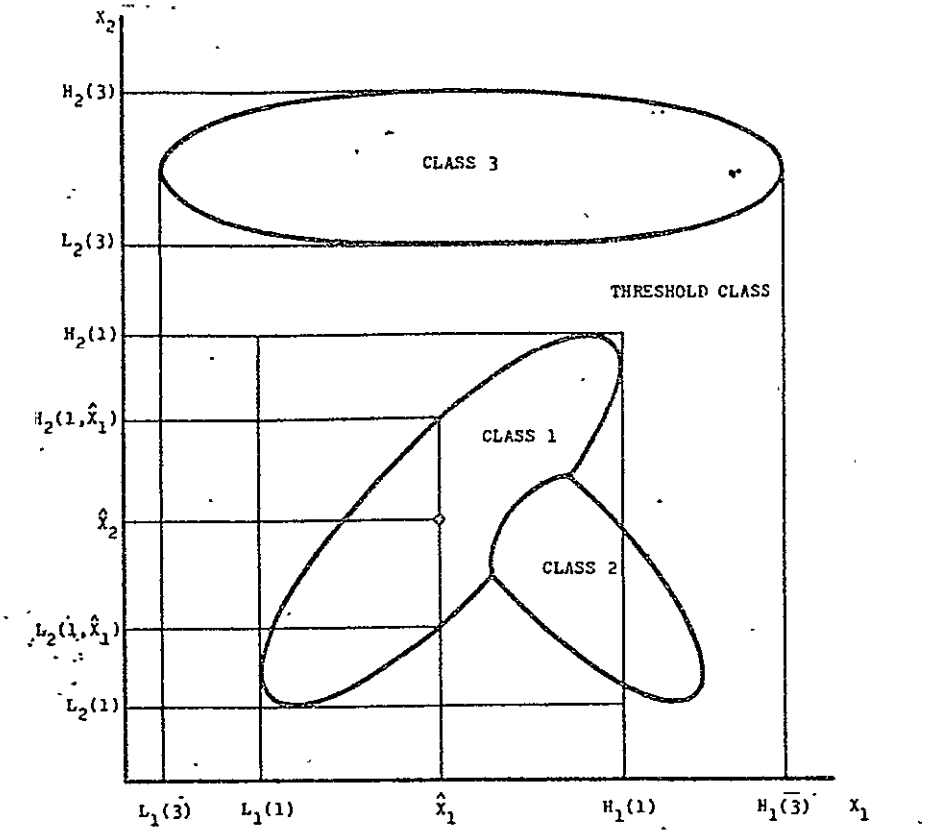


Fig. 1.1.3(1) Explanation of Classification Algorithm for Two Dimensions

constants $L_1(C)$, $H_1(C)$ and $L_2(C,x_1)$ for all $L_1(C) \leq x_1 \leq H_1(C)$ have to be prestored in the random-access core memory.

The values $L_1(C)$ and $H_1(C)$ can be regarded as the contents of two-dimensional arrays. Each array would require N_C 8-bit (assuming $0 \leq x_i \leq 255$) bytes of the core storage, where N_C is the number of classes. Similarly, it is possible to regard $L_2(C,x_1)$ and $H_2(C, x_1)$ as two two-dimensional arrays, each requiring $N_C N_{1max}$ bytes of core storage where N_{1max} is given by Eq. (1.1.3-16) below.

From Figure 1.1.3(1) it can be seen that $N_1(C)$ is the length which results from projecting that measurement space region associated with class C

$$N_1(C) = H_1(C) - L_1(C) + 1 \qquad (1.1.3\text{-}15)$$

$$N_{1max} = \text{Max}\,[N_1(C)] \qquad 1 \leq C \leq N_C \qquad (1.1.3\text{-}16)$$

onto the x_1 axis. In Figure 1.1.3(1), $N_C = 3$ and $N_{1max} = N_1(3) > N_1(1) > N_1(2)$.

The core requirements can be minimized by storing only the number of values that x_1 can actually assume for a given case rather than the fixed number N_{1max} dictated by the worst case. This dynamic core assignment is accomplished by storing and retrieving $L_2(C,x_1)$ and $H_2(C,x_1)$ using P_1 as a pointer in the one-dimensional arrays $L_2(P_1)$ and $H_2(P_1)$. The value of P_1 is computed using Eq. (1.1.3-17)

$$P_1 = O_1(C) + x_1 \tag{1.1.3-17}$$

where $O_1(C)$ is a class-dependent offset given by Eq. (1.1.3-18).

$$O_1(C) = \begin{cases} 1 - L_1(1) & \text{for } C = 1 \\ 1 - L_1(C) + \sum_{i=1}^{C-1} N_1(i) & \text{for } 2 \leq C \leq N_C \end{cases} \tag{1.1.3-18}$$

Figure 1.1.3(3) shows the core arrangement which results from applying Eqs. (1.1.3-17) and (1.1.3-18) for the case shown in Figure 1.1.3(1).

Once the boundary information is prestored in the core memory, the classification proceeds. A hypothesis C is formed concerning which class gave rise to the measurement vector $\underline{x} = (x_1,x_2)'$. The initial hypothesis is that class assigned to the preceding pixel. If this hypothesis fails, classes are tested in descending order of a priori probability. The class

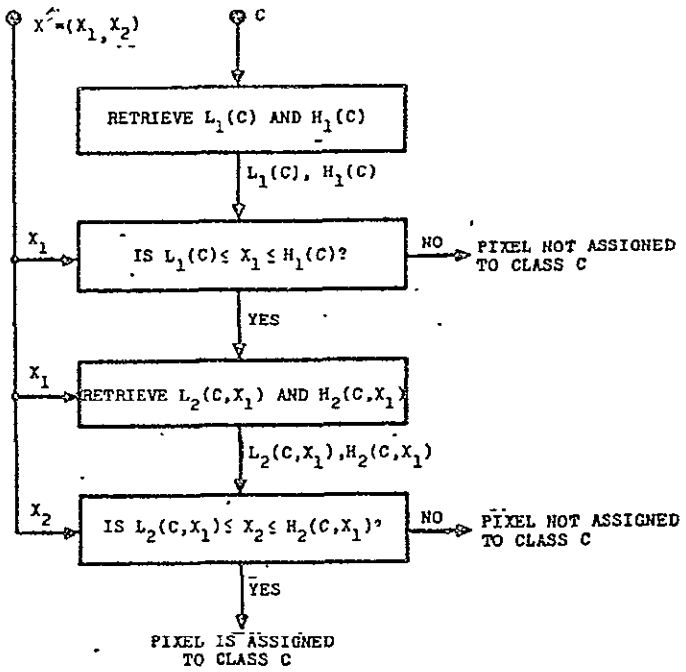


Fig. 1.1.3(2) Method for Determining Whether Pixel is from Class C

C	$O_1(C)$	X_1	P_1	$L_2(P_1)$	$H_2(P_1)$
1	$1-L_1(1)$	$L_1(1)$	1	$L_2[1,L_1(1)]$	$H_2[1,L_1(1)]$
		.	.	.	.
		.	.	.	.
		.	.	.	.
		$H_1(1)$	$N_1(1)$	$L_2[1,H_1(1)]$	$H_2[1,H_1(1)]$
2	$1-L_1(2)+N_1(1)$	$L_1(2)$	$N_1(1)+1$	$L_2[2,L_1(2)]$	$H_2[2,L_1(2)]$
		.	.	.	.
		.	.	.	.
		.	.	.	.
		$H_1(2)$	$N_1(1)+N_1(2)$	$L_2[2,H_1(2)]$	$H_2[2,H_1(2)]$
1	$1-L_1(3)+N_1(1)+N_1(2)$	$L_1(3)$	$N_1(1)+N_1(2)+1$	$L_2[3,L_1(3)]$	$H_2[3,L_1(3)]$
		.	.	.	.
		.	.	.	.
		.	.	.	.
		$H_1(3)$	$N_1(1)+N_1(2)$ $+N_1(3)$	$L_2[3,H_1(3)]$	$H_2[3,H_1(3)]$

Figure 1.1.3(3) Organization of the Prestored Tables of Boundary Information for the Case Shown in Figure 1.1.3(1)

hypothesis C points to $L_1(C)$ and $H_1(C)$, the minimum and maximum values x_1 can have and still be inside the measurement space region associated with class C. If $L_1(C) \leq x_1 \leq H_1(C)$, the hypothesis is tested further; otherwise a new hypothesis is formed as explained previously. The class hypothesis C also points to $O_1(C)$, a class-dependent offset which is added to x_1. The resulting value P_1 points to $L_2(C,x_1)$ and $H_2(C,x_1)$, the minimum and maximum values x_2 can have, for the given value of x_1, and still be inside the measurement space region associated with class C. If $L_2(C,x_1) \leq x_2 \leq H_2(C,x_1)$, the pixel is assigned to class C, otherwise a new hypothesis is formed. If the measurement vector $\underline{x} = (x_1,x_2)'$ does not lie within the prestored boundaries of any of the known classes, it is assigned to the threshold class.

Four-Dimensional Case

The measurement vector for each pixel consists of $n \geq 4$ measurements, and the objective is to assign the pixel to one of the N_C known classes or else the threshold class. The first step is to form a class hypothesis C.

The initial hypothesis is that the pixel is from the same class as the previous pixel. If this hypothesis fails, the classes are searched in order of decreasing a priori probability. The class hypothesis C is used to retrieve the number of those four channels which are most effective in discriminating class C from all of the other classes.

The class hypothesis points to $L_1(C)$, $H_1(C)$ and $O_1(C)$. A test is made whether or not $L_1(C) \leq x_1 \leq H_1(C)$. If not, a new hypothesis is formed. If so, the value x_1 is added to $O_1(C)$ to derive the pointer P_1 which points to the core memory address where $L_2(C,x_1)$, $H_2(C,x_1)$, and $O_2(C,x_1)$ are stored. The process is continued in the same way to determine whether $L_2(C,x_1) \leq x_2 \leq H_2(C,x_1)$ and $L_3(C,x_1,x_2) \leq x_3 \leq H_3(C,x_1,x_2)$ and $L_4(C,x_1,x_2,x_3) \leq x_4 \leq H_4(C,x_1,x_2,x_3)$. If all these conditions are met the pixel is assigned to class C. The first time one of these conditions is not satisfied, a new class hypothesis is formed. If none of the N_C class hypotheses satisfy all four conditions, the pixel is assigned to the threshold class.

1.1.3.4 Clustering

Clustering is a data analysis technique by which one attempts to determine the "natural" or "inherent" relationships in a set of observations or data points. It is sometimes referred to as unsupervised classification because the end product is generally a classification of each observation into a "class" which has been established by the analysis procedure, based on the data, rather than by the person interested in the analysis.

Presently, the typical multispectral classification experiment is conducted as follows [7-16]. The data, collected in a single region under favorable conditions by airborne or spaceborne sensors, are examined in their entirety by the experimenter, who decides which areas are most representative of the region as a whole. The samples from these areas are assembled to form a training set, which is characterized by ground truth information delineating the terrain types of interest. A statistical categorizer, or decision box, is constructed on the basis of the statistical parameters extracted from the training set. The classification performance is then evaluated on another portion of the data (the test set) for which the location and extent of the different types of ground cover are also known to the experimenter.

The details of the various experiments differ with respect to the sources of data, the method of labeling the training and test sets, the terrain types to be identified, the number of spectral bands, the degree of the statistical sophistication of the categorizer, and the methods of evaluating the results, but the general scheme of classifying samples of the test set according to their similarity to a preselected training set is the same.

Extrapolation of the performance levels evaluated in this manner to an operational satellite data gathering system is doubtful. In most experiments both the training sets and the test sets are selected on the basis of visual inspection of the available data. Terrain classes without large uniform representation are frequently deleted from consideration, as are regions of abnormally high variability. The data are divided into training and test sets in such a way as to enhance the probability of successful classification. Even in as well conceived an experiment to extend the spatial recognition range as that described in [12], an intruding cloud required complete reassignment of the intended training region. In view of the amount of information to be collected by the satellite systems, it seems unlikely that a considerable fraction of this data can be visually screened in time to allow modification of the required decision parameters. If, on the other hand, interactive systems are developed to a sufficient degree to allow human analysis of much of the imagery, then the whole concept of automatic terrain classification becomes superfluous.

The point of departure from standard statistical classification techniques can be in the application of an unsupervised learning approach, by means of clustering algorithms [17], to circumvent the difficulties of collecting representative training sets.

To get an intuitive idea of what is meant by natural or inherent relationships in a set of data consider the example shown in Figure 1.1.3(4). If the spectral reflectance of vegetation in a visible wave band were plotted against reflectance in an infrared wave band, dry vegetation and green vegetation could be expected to form discernible clusters.

If the data of interest never involved more than two attributes (measurements or dimensions), cluster analysis might always be performed by visual evaluation of two-dimensional plots such as that in Figure 1.1.3(4). But beyond two or possibly three dimensions, visual analysis is impossible.

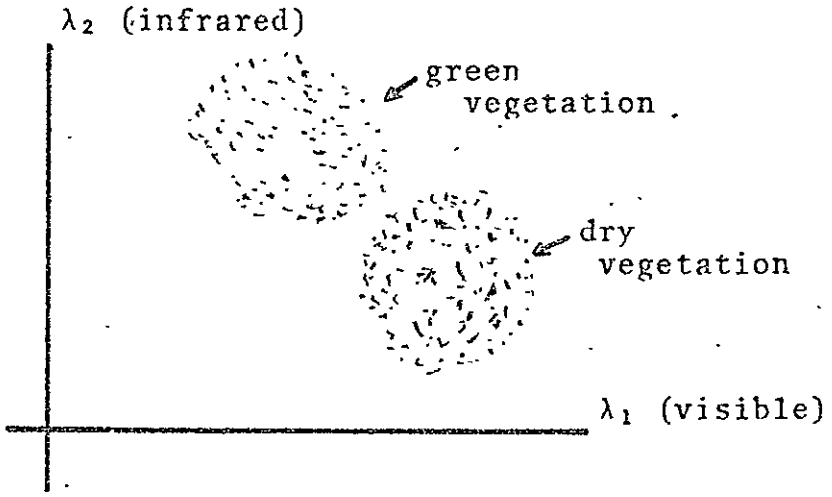


Figure 1.1.3(4) Example of data clusters

For such cases, as in multispectral data, it is desirable to develop computer algorithms to perform cluster analysis. Haralick and Kelly [18] have presented two such algorithms: the first partitions the image sequence and the second partitions the measurement space. In both, the partition is constructed by finding appropriate center sets and chaining to them all similar points. The resulting clusters are simply connected.

The reader interested in the many possible ways of defining clustering in quantitative terms may consult references [19] and [20]. Essentially, the definition of a clustering algorithm depends on the specification of two distance measures: a measure of distance between data points or individual observations; and a measure of distance between groups of observations (clusters). Figure 1.1.3(5) is a block diagram for a typical clustering algorithm [21]. The point-to-point distance measure is used in the step labeled "Assign each vector to nearest cluster center". The distance between groups of points (clusters) is calculated in the step "compute separability information".

REFERENCES

1. Tou, J. T., and Gonzalez, R. C., Pattern Recognition Principles, Addison-Wesley Publishing Co., Reading, Mass., 1974.

2. Crave, R. B., Malila, W. A., and Richardson, W., "Stability of the Normal Density Assumption for Processing Multispectral Scanner Data," IEEE Trans. Geos. Elec., GE-10, pp. 158-165, 1972.

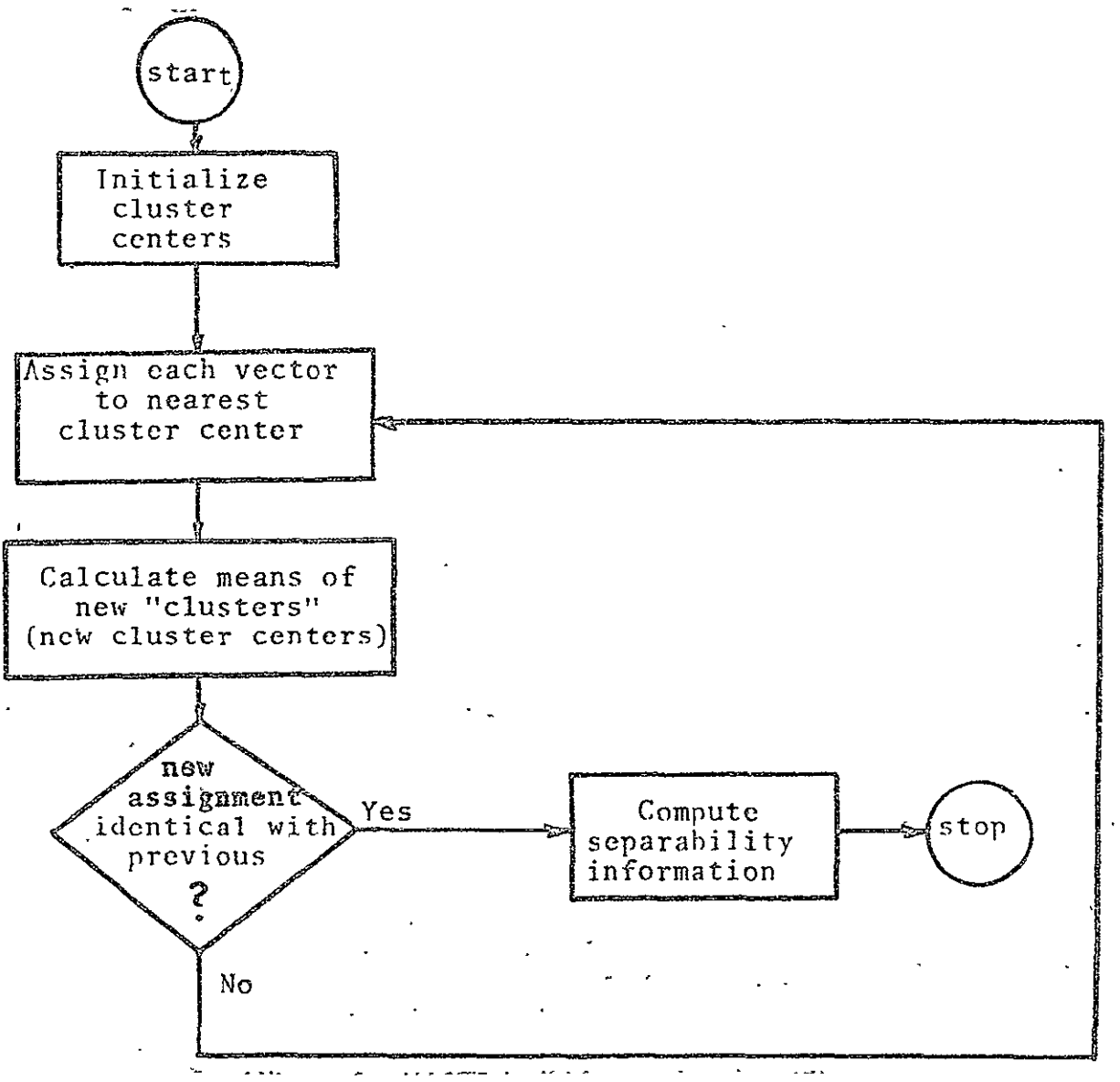


Figure 1.1.3(5) Clustering Algorithm

3. Eppler, W. G., Helmke, C. A., and Evans, R. H., "Table Look-Up Approach to Pattern Recognition," Proc. of the 7th International Symposium on Remote Sensing of the Environment, University of Michigan, May, 1971.

4. Odell, P. L., Duran, B. S., and Coberly, W. A., "On the Table Look-Up in Discriminant Analysis," J. Stat. Comput. Simul., vol. 2, pp. 171-188, 1973.

5. Odell, P. L., and Duran, B. J., "Comparison of Some Classification Techniques," IEEE Trans. on Computer, vol. C-23, no. 6, June 1974.

6. Eppler, W. G., "An Improved Version of the Table Look-Up Algorithm for Pattern Recognition," Proc. of the 9th International Symposium on Remote Sensing of the Environment, University of Michigan, April, 1974.

7. Hoffer, R. M., Johannsen, C. J., and Baumgardner, M. F., "Agricultural Applications of Multispectral Sensing," Proceedings of the Indiana Academy of Science 76, pp. 386, 1966.

8. Steiner, D., "A Methodology for the Automated Photo-Identification of Rural Land Use Types" in Automatic Interpretation and Classification of Images, edited by A. Grasselli, Academic Press, Inc., New York, p. 235, 1969.

9. Holmes, R. A., "An Agricultural Remote Sensing Information System," Proc. of the IEEE Electronic and Aerospace Systems Convention, Washington, D. C. EASCON '68 Record, p. 142, September, 1968.

10. Richard, A. J., Torline, R. J., and Allen, W. A., "Computer Identification of Ground Patterns from Aerial Photographs," Proc. of the Seventh Symposium on Remote Sensing of the Environment, The Univeristy of Michigan, Ann Arbor, Michigan, p. 1357, 1971.

11. Fu, K. S., Landgrebe, D. A., and Phillips, T. L., "Information Processing of Remotely Sensed Agricultural Data," Proc. IEEE 57, p. 639, 1969.

12. Hoffer, R. M. and Goodrick, F. E., "Variables in Automatic Classification Over Extended Remote Sensing Test Sites," Proceedings of the Seventh Symposium on Remote Sensing of the Environment, The University of Michigan, Ann Arbor, Michigan, p. 1967, 1971.

13. Anuta, P. E., Kristof, S. J., Levandowski, D. W., MacDonald, R. B., and Phillips, T., "Crop, Soil, and Geological Mapping from Digitized Multispectral Satellite Photography," Proc. of the Seventh Symposium on Remote Sensing of the Environment, The University of Michigan, Ann Arbor, Michigan, p. 1983, 1971.

14. Marshall, R. E., Thompson, N. S., and Kriegler, F., "Use of Multispectral Recognition Techniques for Conducting Wide Area Wheat Surveys," Proc. of the Sixth Symposium on Remote Sensing of the Environment, The University of Michigan, Ann Arbor, Michigan, p. 2, 1969.

15. Hoffer, R. M., Anuta, P. E., and Phillips, T. L., "Application of ADP Techniques to Multiband and Multiemulsion Digitized Photography," Proc. Am. Soc. Photogrammetry Convention, San Francisco, September, 1971.

16. Anuta, P. E., MacDonald, R. B., "Crop Surveys from Multiband Satellite Photography Using Digital Techniques," J. Remote Sensing Environment 2(1), 53, 1971.

17. Ball, G. H., "Classification Analysis," Stanford Research Institute Technical Note, Menlo Park, California, November, 1970.

18. Haralick, R. M. and Kelly, G. L., "Pattern Recognition Measurement Space and Spatial Clustering for Multi-Images," Proc. IEEE 57, p. 654, 1969.

19. Ball, G. H., "Data Analysis in the Social Sciences: What About the Details," Proc. Fall Joint Computer Conference, December, 1965.

20. Wacker, A. G., and Landgrebe, D. A., "Minimum Distance Classification in Remote Sensing," LARS Information Note 030772, Laboratory for Applications of Remote Sensing, Purdue University, W. Lafayette, Indiana, February, 1972.

21. Swain, P. H., "Pattern Recognition: A Basis for Remote Sensing Data Analysis," LARS Information Note 111572, Laboratory for Applications of Remote Sensing, Purdue University, W. Lafayette, Indiana, November 1972.

22. Nagy, G., and Tolaba, J., "Nonsupervised Crop Classification Through Airborne Multispectral Observations," IBM J. RES. Develop., pp. 138-153, March, 1972.

23. Gupta, J. N., and Wintz, P. A., "Multi-Image Modeling," TR-EE 74-24, School of Electrical Engineering, Purdue University, West Lafayette, Indiana, September, 1974.

1.2 MSS SENSOR TECHNOLOGY

1.2.1 Electromechanical Scanners

Electromechanical scanners are capable of producing geometrically-registered, multispectral images in spectral channels extending from the visible to the thermal infrared. In addition, accurate radiometric measurements can be made. Image formation is now accomplished by mechanical scanning. Typically, mechanical motion causes the scene to be sampled in the cross-track direction by a detector or array of detectors while satellite motion provides the orthogonal scan component.

The information extracted from scanner data is in the spatial, spectral, and temporal distribution of radiation from a scene. For the most part, sensor advancement means improving the spatial resolution for a given operating distance. More recently, attention has been given to spectral distribution and automatic classification based on the spectral information from the scene. Since the spectra of the vegetation features vary with their growth cycle or season, the spectral classification task becomes easier when temporal variations are included. Thus, the advanced images may be viewed as a high resolution multispectral scanner with the ability to observe a scene periodically.

When photomultiplier detectors are employed, the internal gain is assumed to be sufficiently high that amplifier noise can be neglected in comparison with photoelectron (shot) noise. In this case, the limiting noise can be expressed as the square root of the number of photoelectrons released during an integration period from a resolved elemental area of background while the signal is the difference between the corresponding numbers of photoelectrons associated with target and background.

Photomultiplier tubes are currently used in multispectral scanner systems for the short wavelength channels. For high resolution systems, PMT's are used for wavelengths shorter than about 800 nm; in lower resolution systems, PMT's may not be used at all or only at wavelengths shorter than 500 nm.

Silicon planar p-n junction photodetectors represent the most advanced semiconductor technology available at this time and provide good resolution and parameter control. These units show good high-frequency response, high quantum efficiency, low noise, and require no special cooling for most applications. Because the photodiode has no internal signal gain, high performance channels are usually limited by the noise of the first amplifier stage. Silicon photodiodes are normally used in multispectral scanning systems for wavelengths between about 0.4 μm and 1.1 μm. Other semiconductor materials are used in the fabrication of p-n junction photodetectors for wavelengths longer than 1.1 μm. In general, infrared photodiodes must be operated at low temperature to achieve optimum performance, with the longer wavelength detectors requiring the lower operating temperature.

Photoconductive detectors are most useful at infrared wavelengths longer than 3 μm. These devices are essentially variable resistors in which the conductivity of the bulk material increases monotonically with the magnitude of optical power absorbed in the active volume of the units.

1.2.1.1 Detector Cooling Systems

Future infrared imagers and scanners will utilize cooled quantum- or photodetectors. These detectors require cooling to cryogenic temperatures, i.e., below approximately 120°K to achieve background-limited performance.

In general, the long wavelength cutoff and detectivity, as well as other detector parameters, are determined by the operating temperature. Photodetectors operating in the 8-13 μm atmospheric window require lower operating temperatures than those operating in the 3-5 μm region. For remote sensing applications, future imagers and scanners, particularly those operating in the 8-13 μm region, will utilize intrinsic photodetectors cooled to 100°K or below to achieve background-limited performance.

Within the next decade it is anticipated that detector cooling requirement for airborne and space-borne infrared systems will generally lie in the 50-120°K region with perhaps a few applications requiring temperatures

as low as 20°K. The cooling capacity at these temperatures will range from a few milliwatts for a single photovoltaic detector to perhaps a watt for large arrays of photoconductive detectors.

The basic design parameters for a cryogenic detector cooling system are: the required operating temperature and temperature stability, heat load at the operating temperature, the alignment requirements of the cooled detectors relative to the optics, and the reliability and operating life of the system.

For spaceborne systems, where weight and power are usually limited, the selection of a cooling system involves a tradeoff between the detector and cooling system parameters for a given mission duration.

Three basic types of cooling systems are considered for spaceborne systems.

(a) Passive radiators for spaceborne system which cool detectors by direct radiation to the low-temperature sink of deep space.

(b) Open-cycle systems which use fluid or solid cryogens stored in a dewar, or stored high pressure gas which provides refrigeration by the Joule-Thompson effect. Solid cryogens are only applicable to spaceborne systems.

(c) Closed-cycle systems employing a mechanical refrigerator using helium gas as the working fluid or closed-cycle Joule-Thompson systems.

For airborne application, open cycle systems using liquid cryogens stored in a dewar, or closed-cycle refrigerators are the only logical system choices. Open-cycle systems using liquid nitrogen or liquid helium have been used to cover the temperature range from 42°to 70°K. These systems are relatively simple, low in cost, have good temperature stability and do not introduce mechanical vibrations or microphonics to the focal plane. They are capable of providing continuous refrigeration for the duration of a single aircraft flight without resupply.

For spaceborne applications, passive radiators are applicable for small heat loads down to their lowest temperature limit. Solid-cryogen coolers are also applicable for low heat loads, and can be used to reach temperatures below those achievable with radiators. For a specific spaceborne instrument, the particular constraint of the instrument should be considered in detail before selecting the cryogenic cooler.

Many of the cooling requirements for spaceborne applications can probably be met with passive radiators. Current passive radiator systems are not large enough to handle the increased heat loads anticipated larger detector arrays. Large passive radiators having a cooling capacity of approximately 1 watt at 70-80 ^{O}K temperature level are quite sufficient for many spaceborne applications.

Open-cycle cooling systems may not be applicable for the one-to-two year missions with focal plane heat loads which are expected to exceed 100 milliwatts.

1.2.1.2 Optical Systems

Multispectral electromechanical scanning systems can be classified broadly as image-space and object-space scanners. The former requires an imaging system that covers the full field of view with the required resolution. Object-plane scanners take the load off the optical system and place it on the scanning system sequencing a narrow optical field of view.

Narrow-field systems usually include simple spheres, parabolas, Cassegrain, Newtonian, Ritchey-Chretien, and Dall-Kirkham systems. Using the most expensive optical system (the Ritchey-Chretien), one can obtain resolution of 0.02 mr for about 1 degree full field (17 mr).

Wide-field systems (10 degrees or more) are much more difficult to design and build. They have scanning mechanisms that are simpler to implement, however, and are therefore of considerable interest. The two main candidates for such wide-angle optical systems are the Schmidt and the Bonwers-Maksutov system. Both provide resolutions on the order of 0.1 mr out to angles of about 15 degrees.

1.2.1.3 Hadamard System

A newer optical technique that makes use of the multiplex advantages in spectroscopy is an image-space scanner called the Hadamard system. Where the normal spectrometer uses one exit slit and rotates a prism for multiple imaging, the Hadamard system uses a mask that provides multiple exit slits. A system of this type requires that the entire system be imaged. Any variations in the radiation from the opaque elements causes variations in the apparent scene radiation. The nonuniformities cause scene noise and uniform radiation can increase photon noise. This system

is of great advantage for sampling a scene with a few variations, compared to a single detector initially. Scanning noise, however, is a serious problem.

A comparison can be made with a system employing a linear array of detectors. For an array of n detectors to cover the n resolution elements of the scan line, the gain in signal-to-noise ratio is $n^{1/2}$. The detectors probably cost more, having a varying responsivity from element to element and require a bias supply, cold shielding, preamplifiers, etc. The Hadamard scanner uses one detector which reduces cost. The number of preamps and bias supplies, weight, and cooling requirements are lower, but the system requires a scanning mask, sampling system, and data processor for inverting the matrices.

1.2.2 Solid-State Scanners

Solid state scanners are in the early stages of development at the present time, but they offer important advantages for future earth-orbit satellite missions. Advantages include: no mechanical scanning, built-in geometric accuracy, the high quantum efficiency of silicon for the visible range, high resolution, good stability, low voltage operation, and improved signal-to-noise ratio performance.

The fundamental performance criteria for integrated, self-scanned solid-state arrays closely parallel those for alternative sensors. Among the important criteria are:

(1) Geometric resolution which is determined by element-size spacing and number.

(2) Quantum efficiency and its variation with length, as determined by reflection and absorption losses in layers overlying the silicon.

(3) Scan rates using integration and, together with element-size, quantum efficiency, and available light, determine available signal.

These factors set limits on dynamic range and boundary conditions on signal-to-noise ratio of the sensor itself.

Of importance in practical systems is device stability, particularly in short term and, although of lesser consequence, power supply requirements. Scan rate, taken with the number of elements, imposes bandwidth requirements per monolithic chip and for the total system. Dynamic range requirements and signal compression and encoding requirements are dictated by

system constraints on the one hand; on the other, they are limited by available transmission bandwidth and by precompression and precoding noise associated with the signal.

As with other sensors, solid state self-scanned arrays require geometric, radiometric, and electrical calibration. Intrinsically, geometric calibration is straightforward and, once established, is subject only to mechanical distortions of the sensor structure. Element-to-element variations in photosensitivity or dark current must, as with other sensors, be calibrated out. Depending on the particular sensor design, additional calibration must be provided when additional per-element variations exist; for example, phototransistors require gain or linearity calibration, usually at two or more points, plus additional calibration for temperature variations. In general, determination of photosensor geometry by the photomasking process during fabrication and the chemical stability of the silicon/silicon dioxide system make geometric, radiometric and electrical characteristics of solid-state self-scanned arrays very stable.

Silicon-detector arrays represent an imaging technique that makes use of linear arrays of solid-state detectors operating in what is termed a "pushbroom scan" mode. In such a system, a detector array is used to image the scene in the cross-track direction and spacecraft motion is used to provide the orthogonal scan component. The primary advantages resulting from the use of these arrays are:

(1) With the array oriented in a cross-track configuration, continuous coverage of a wide swath width of terrain can be obtained. Mechanical scanning is eliminated, since the satellite subpoint motion along the ground track provides the single-axis scanning motion that is required, and the detector elements are interrogated electronically.

(2) Large arrays can be formed containing several thousand detector elements. The use of high-density arrays offers high resolution capability.

(3) The precise geometric alignment of the detector elements resulting from the use of micro circuit fabrication techniques, in addition to the precision of alignment within the optical system, offers an advantage in accuracy of ground reconstruction of images.

(4) The detectors are operated in an integration mode, thus providing high scan efficiency. The exposure time of each detector element is limited only by the permissible image motion at the optical focal plane.

The high scanning efficiency results in a long exposure time, increasing the signal-to-noise ratio and resolution.

There are three candidate sensors: Photodiodes, phototransistors and charge-coupled devices (CCD). All three are similar in mechanism, limitations, and potential of the sensing portions, but differ significantly in the arrangements for signal amplification and scanning. In photodiode and phototransistor arrays, scanning is accomplished by switching, while, in charge-coupled devices, scanning is accomplished by movement of potential wells.

In general, there are two self-scanning modes: digital multiplexing and analog charge-transfer. The photodiodes and phototransistors ordinarily use digital multiplexing. The CCD sensors have analog charge-transfer readout. Since all of these sensors are silicon, their operation is restricted to the visible and near infrared (1.1 μm). The readout circuitry is generally on the same silicon chip as the array of detector elements (usually 100 or more sensors per chip).

At present, silicon photodetector elements can be assembled into line arrays of several-thousand elements with a center-to-center element spacing of 15 μm for a high-resolution, push-broom mode of scanning.

Charge-coupled devices are in the early developmental stage, but CCD scanners possess better performance capabilities for multispectral earth imaging from orbit. Using the basic CCD principle, it is possible to transport a photon-generated signal charge over long distances within a chip and to sense the charge with a preamplifier having potentially an extremely small input capacitance. This in turn leads to amplification with very low input noise levels. Using the buried-channel CCD concept, it is possible to achieve very high transfer efficiencies over wide dynamic ranges, which minimizes image distortion. Using transparent gates over the photosensing regions, it is possible to achieve high net quantum efficiencies.

CCD is a new class of semiconductor structures normally operating in (thermal) non-equilibrium and utilizing, as the signal carriers, minority charge transported by moving potential wells. In essence, therefore, a CCD is a nearly ideal semiconductor analog shift register. The CCD concept permits the design of highly complex functional devices at potentially low cost. In addition CCD has the attributes of:

(1) silicon fabrication simplicity

(2) high packing density

(3) high reliability

(4) low power requirements

(5) potential low-noise analog signal processing

Self-scanned arrays also appear worth exploring for infrared applications, although the advantages in the infrared range appear not to be as significant as in the visible spectrum. The limitations result from the fact that the devices are still in the early developmental stage and infrared detector development has not reached the technological maturity of silicon detectors and arrays. Major differences are contained in the following: (a) present HgCdTe detectors operate generally in the photoconductive mode; (b) small scale (approximately 100-element) IR arrays have been manufactured only through the physical assembling of individual detectors; and (c) 1/f noise in the infrared detectors is considerable higher than in silicon detectors. The implications of these differences for the IR systems are as follows: (a) present systems require a drop in the incoming radiation; (b) detector bias (1 mW per detector) is required; and (c) the low impedance of the detectors (25-250Ω) requires high-power preamplifier stages.

It is currently possible to make high-density arrays of IR detectors (photo conductive, photovoltaic) out of narrow band gap semiconductors for use in mechanical scanners. However, it is not possible yet to integrate these detectors with their associate electronics onto one chip; this is because the technology for fabricating transistors in materials such as InAs, InSb, HgCdTe, PbSnTe, etc., has not been established. It is possible to consider hybrid devices consisting of IR detector arrays cemented to a silicon chip that contains the necessary readout circuitry.

REFERENCES

1. Am. Met. Soc., Proc. of the Conference on Atmospheric Radiation, August, 1972.

2. ASP., Proc. of the American Society of Photogrammetry Seminar on Operational Remote Sensing, Houston, Texas, February, 1972.

3. Billingsley, F. C., "Applications of Digital Image Processing" in Applied Optics, vol. 9, no. 2, pp. 289-299, February 1970.

4. Billingsley, F. C., "Considerations for Digital Image Processing" in Optical Telescope Technology, NASA SP-233, pp. 673-692, 1970.

5. Billingsley, F. C., Goetz, A. F. H. and Lindsley, J. N., "Color Differentiation by Computer Image Processing" in Photo. Sci. and Engrg., vol. 14, no. 1, pp. 28-35, 1970.

6. Duntley, S. Q., Boileau, A. R. and Presidendorfer, R. W., "Image Transmission by the Troposphere 1" in J. Opt. Soc. Am., vol. 47, no. 6, pp. 499-506, 1957.

7. Efron, E., "Image Processing by Digital Systems" in Photogrammetric Engrg., vol. 34, p. 1058, 1968.

8. Elterman, L., Vertical Attenuation Model with Eight Surface Meteorological Ranges 2 to 13 km, Report AFCRL-70-0200, 1970.

9. Gaven, J. V., Jr., Tavitian, J. and Harabedian, A., "The Informative Value of Sampled Images as a Function of the Number of Grey Levels Used in Encoding the Images", in Photo, Sci. and Engrg., vol. 14, no. 1, pp. 16-20, 1970.

10. Huang, T. S., "Combined Use of Digital Computers and Coherent Optics in Image Processing," SPIE Computerized Imaging Techniques Symposium, Washington, D. C., June 1967.

11. Manned Spacecraft Center, Earth Observation Aircraft Program, Fiscal Year 1970.

12. McClatchey, R. A., Fenn, R. W., Selby, J. E. A., Volz, F. E. and Garing, J. S., Optical Properties of the Atmosphere, Report AFCRL-71-0279, May 1971.

13. NASA, Data Users Handbook, Earth Resources Technology Satellite, NASA Goddard Space Flight Center Document No. 71SD4249, Revised 1972.

14. NASA. Earth Resources Research, Volume 1, NASA MSC-02576, July 1971.

15. NASA. Remote Sensing of Earth Resources, A Literature Survey with Indexes, NASA SP-7036, September 1970.

16. NASA. Spacecraft Earth Horizon Sensors NASA Space Vehicle Design Criteria (Guidance and Control), NASA SP-8033, December 1969.

17. Norwood, V. T., Optimization of a Multispectral Scanner for ERTS, in Michigan, pp. 227-235, 1969.

18. Taylor, Ralph E., "Tracking and Data Acquisition for NASA Spacecraft Systems," NASA Goddard Space Flight Center, Preprint, August 1972.

19. TRW. PFCS/PADS, A Collection of Papers on Precision Attitude Determination and Control, papers presented at AIAA Guidance, Control and Flight Mechanics Conference, Hofstra University, Hemstead, New York, August, 1971.

20. Annable, R. V., "Radiant Cooling," Applied Optics, 1970.

21. Gabron, F., McCullough, J. E., Merriam, R. L., "Spaceborne Passive Radiator for Detector Cooling," Proc. of 20th National Infrared Information Symposium (IRIS), 1972.

22. Maddocks, F. E., "Application of Turbomachinery to Small-Capacity, Closed-Cycle, Cryogenic Systems," Advances in Cryogenic Engineering, vol. 13, 1968.

23. Prast, G., The Vuilleumier (VM) Cycle. Cryogenics and Infrared Detection, Boston Technical Publishers, Inc., Cambridge, Ma., 1970.

24. Wolfe, W. L., (Editor), Handbook of Military Infrared Technology, U.S. Government Printing Office, Washington, D. C. 1965.

25. Amelio, G. F., Bertram, W. J., Jr., and Tomsett, M. F., "Charge-Coupled Imaging Devices: Design Consideration," IEEE Trans. on Electron Devices, ED-18, no. 11, pp. 986-992, 1971.

26. Strain, R. J., "Properties of an Idealized Traveling-Wave Charge-Coupled Device," IEEE Trans. on Electron Devices, ED-19, no. 10, 1972.

27. Weimer, P. K., Pike, W. S., Kovac, M. G., and Shallcross, F. V., "The Design and Operation of Charge-Coupled Image Sensors," Paper Presented to the IEEE Solid State Circuits Conference, Philadelphia, Feb. 15, 1973.

1.3 PREPROCESSING ALGORITHMS

1.3.1 Sensor Corrections

In order to illustrate the kind of MSS sensor corrections that will be required on-board a satellite we briefly review the ERTS system which is typical of most multispectral scanners.

The purpose of the MSS system is to accurately produce images of the earth from a low altitude satellite. The energy received from the earth's surface, both reflected and emitted, is sensed and digitized by the system. The digitized data are transmitted to the ground where they are recorded for later processing. The recorded data are either analyzed directly by a computer or transferred first to a photographic transparency.

The recorded data on the magnetic tapes includes timing and radiometric calibration information that can be converted by computer programs into spectral signatures for direct analysis. In photographic transparencies, the density of the film is proportional to the scene energy levels. There are twenty-four channels in the system and six channels are involved in making a one band picture. To produce a perfect picture of the scene imaged, the channel gain and the offset of each of the six channels in a band must be known exactly. This is necessary so that the correct film density can be assigned to the digital words generated by the multiplexer. If this condition is met, individual lines in the picture at almost all density levels will not be discernible. However, if the gains and offsets are in error in one or more channels, the individual adjacent lines which should have nearly the same scene information will have different average levels of density in each channel and, consequently, will be displayed as

stripes in the picture. Also, for signature analysis the channel gain and offset are equally important to maintain relative response between the four bands. For these reasons, a calibration system is required in the MSS system to report periodically on the status of the channel gain and offset.

1.3.1.1 Radiometric Response

The integrating sphere is used as a light source to establish absolute gain/sensitivity for the scanner. The integrating sphere is calibrated using a primary standard source. Table 1.3.1(I) gives a listing of the radiance required for each channel to produce full scale signals at the output of the multiplexer. The integrating sphere used to determine the radiance required was calibrated at GSFC. The details are given in the Acceptance Test Report (HS 324-5196).

Table 1.3.1(I) Radiance Necessary to Produce Full Scale at the Multiplexer Output ($mw/cm^2 \cdot ster$)

	BAND			
CHANNEL	1	2	3	4
A	2.45	1.86	1.84	4.81
B	2.43	1.95	1.68	4.80
C	2.34	1.94	1.73	4.76
D	2.33	1.87	1.80	4.84
E	2.39	1.92	1.74	4.70
F	2.35	1.96	1.78	4.70

1.3.1.2 Determination of Gain and Offset

Over a wide range of shades of gray, the eye responds to ratios, instead of absolute levels. The eye can detect a sharp-edged junction between two fields that differ by little more than 2 percent in level. As a result the gain determination procedure should be designed to compute the gains of all channels in a band to within an uncertainty of 2 percent peak-to-peak. Furthermore, based on visual impressions of a test transparency, it has been determined that an acceptable offset spread in a band for strip-free pictures is 30 mv. The gain and offset are defined in Figure 1.3.1(1).

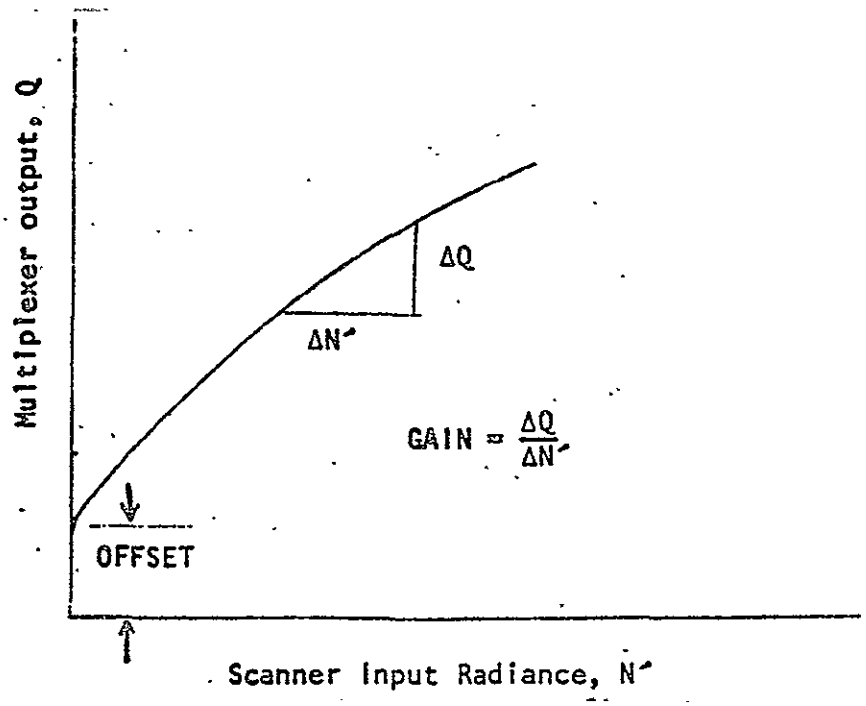


Figure 1.3.1(1) Determination of Gain and Offset

The gain and offset of each channel in a band are determined by processing the gray wedge in conjunction with the sun calibration pulse. The sun pulse is used to modify the gain and offset computed from the gray wedge.

The reference (initial) gain and wedge have to be known a priori because the calibration gray wedge (different for each scanner channel) can only provide information to compute the change in gain and offset from a previously recorded reference gray wedge. The integrating sphere is all that is needed to obtain the reference calibration data. Two different radiance outputs are used to establish the initial offset. One of these two settings, or an average of several settings, is used to establish the gain.

1.3.2 Data Bulk Reduction

The extremely large volume of data generated by a MSS imposes a severe computational burden on the on-board processor. The application of appropriate data compression and/or feature selection techniques can sometimes reduce the severity of this problem. The algorithms used should not destroy more than the maximum acceptable amount of information and should be capable of efficient compression of different kinds of data (i.e.,

vegetation, desert, mountain, etc.) that a multispectral sensor might encounter over changing terrain.

1.3.2.1 Coding

Data compression algorithms for multispectral scanner data, (e.g., ERTS data), have been developed, analyzed and tested to the point where these algorithms and their performances are reasonably well understood. From the results of these studies and experiments there appear to be two basic types of data compression algorithms that are applicable: (1) transform coding; (2) coding by BLOB.

1.3.2.1.1. Transform Coding

Transform coders perform a sequence of two operations. The first operation is a linear transformation that transforms the set of statistical dependent data elements into a set of more independent coefficients. The second operation is to individually quantize and code each of the coefficients. A variety of linear transformations is available to implement transform coding or MSS data, but the eigenvector transformation is the optimum linear transformation in two senses: The mean square error between the original and reconstructed data is less than for any other linear transformation; also, it eliminates all correlations in the data.

In this method an N-vector of data samples

$$\underline{x} = (x_1, x_2, \ldots, x_N)' \qquad (1.3.2\text{-}1)$$

is transformed into an n-vector of coefficients

$$\underline{y} = (y_1, y_2, \ldots, y_n)' \qquad (1.3.2\text{-}2)$$

by the transformation

$$\underline{y} = \underline{T}\,(\underline{x} - \underline{m}) \qquad (1.3.2\text{-}3)$$

where $\underline{m}$ is the mean vector of $\underline{x}$,

$$\underline{m} = E\,[\underline{x}] \qquad (1.3.2\text{-}4)$$

and $\underline{T}$ is a n x N matrix whose rows are the eigenvectors of the covariance matrix of $\underline{x}$: That is, if

$$\underline{C} = E\,[(\underline{x} - \underline{m})\,(\underline{x} - \underline{m})]' \qquad (1.3.2\text{-}5)$$

Then the rows $\underline{t}$ of $\underline{T}$ are the n solutions to the equation

$$\underline{C}\,\underline{t} = \lambda \underline{t} \qquad (1.3.2\text{-}6)$$

corresponding to the n largest eigenvalues $\lambda = \lambda_1 > \lambda_2 > \cdots > \lambda_n$. A replica $\hat{\underline{x}}$ of the data is reconstructed from the coefficient $\underline{y}$ by transformation

$$\hat{\underline{x}} = \underline{T}' (\underline{y} + \underline{m}) \qquad (1.3.2\text{-}7)$$

For n = N the reconstructed data $\hat{\underline{x}}$ is identical to the original data $\underline{x}$; i.e., $\hat{\underline{x}} = \underline{x}$. For n < N some error $(\hat{\underline{x}} - \underline{x})$ is incurred. However, the eigenvector transformation results in the least mean square error

$$\varepsilon^2 = \{E \, || \, \hat{\underline{x}} - \underline{x} \, ||^2\} \qquad (1.3.2\text{-}8)$$

of all linear transformations. The elements $y_1, y_2, \ldots, y_n$ of $\underline{y}$ are uncorrelated and have variances given by the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. It can also be shown that the mean square error Eq. (1.3.2-8) is given by

$$\varepsilon^2 = \sum_{i=1}^{N} \lambda_i - \sum_{i=1}^{n} \lambda_i = \sum_{i=n+1}^{N} \lambda_i \qquad (1.3.2\text{-}9)$$

so that retaining only the first n of the N coefficients results in a mean square error given by the sum of the variances (eigenvalues) of the discarded coefficients.

In any particular application the number of coefficients n that must be retained depends on how fast the eigenvalues $\lambda_1 > \lambda_2 > \lambda_3 > \ldots > \lambda_N$ decrease. If the data samples are not correlated, all N eigenvalues have significant values and we must choose n = N for negligible distortion. On the other hand if the data samples are highly correlated then the eigenvalues decrease rapidly so that only the first few have significant values and all but the first few can be discarded. In this case n << N and significant sample compression ratios

$$R_s(\varepsilon^2) = \frac{N}{n} \qquad (1.3.2\text{-}10)$$

can be obtained with negligible distortion. Hence the compression ratio that can be achieved depends on the amount of redundancy in the data and the amount of error that can be tolerated.

For correlated data the n retained coefficients have different rms values $\sqrt{\lambda_1} > \sqrt{\lambda_2} > \ldots > \sqrt{\lambda_n}$ so that a different number of bits should be used to code each coefficient. The mean square quantization error is minimized by choosing $m_i \sim \log \lambda_i$. This is called block quantization. If the original N data samples have m bits each, then the bit compression ratio achieved by the eigenvector transformation and block quantization is

$$R_b(\varepsilon^2) = \frac{mN}{m_1 + m_2 + \ldots + m_n} \qquad (1.3.2\text{-}11)$$

It can be shown that $R_b(\varepsilon^2)$ can be made less than $R_s(\varepsilon^2)$ (for some ε^2).

The eigenvector transformation and the block quantization method have been used to encode waveforms [2,3] and pictures [4,5]. Monochromatic pictures were encoded and reconstructed with no noticeable distortion at 2.5 bits per picture element. Ready and Wintz [6] implemented the same technique for MSS data. They considered two additional transforms (Fourier and Hadamard) also applicable to the MSS data. Both of these transformations are non-source-dependent in that the set of orthonormal basis vectors is fixed regardless of the source characteristics. The rows of the Fourier transformation matrix are the sampled, harmonically-related sine and cosine functions. The rows of the Hadamard matrix are the discrete version of orthogonal Walsh functions.

1.3.2.1.2 Coding by BLOB

BLOB first subdivides the multi-imagery into 2 x 2 subsets (pixel groups) of picture elements (pixels). A hypothesis testing algorithm computes the mean (gray level) and variance (texture) of each 2 x 2 array and compares the first and second order statistics of adjacent subsets. Adjacent subsets having similar first and second order statistics are merged into blobs. In this manner the entire imagery is partitioned into blobs such that the picture elements within each blob have similar gray levels and/or variances. By varying either of two parameters, the amount of consideration given to gray level and texture can be adjusted [8].

The boundaries may be weak in some spectral channels and strong in others. To take this consideration into account, the variance (F-test) and mean (student t-test) tests are implemented separately in each channel. This has been called a "multiple-univariate approach."

Finally, BLOB transforms the boundaries into a sequence of binary digits with the help of contour tracing algorithm discussed in [3]. The output of the contour tracer is in the form of directionals which are coded.

1.3.2.2 Feature Selection

Use of coding procedures reduces the data volume that must be processed by the data analysis algorithm. This in turn reduces both the time required to read the data into the computer and the time required to do the mathematical operations on the data. For example, classification algorithms are usually designed to operate on fewer number of spectral

channels because including more channels requires excessive computer time. In that case it is desirable to know the relative importance of the individual features from the classification viewpoint. This suggests the study of feature selection; that is, the selection of subsets of feature measurements from the complete set.

The eigenvector transformation discussed in Section 1.3.2.1.1 is also one of the most popular techniques for feature selection. The procedure is usually referred to as the principal components method [1] and consists of the following steps.

(1) Compute the covariance matrix $\underline{C}$ of the given data. This matrix is n x n.

(2) Obtain the n eigenvectors and associated n eigenvalues of $\underline{C}$. The eigenvectors are n-dimensional.

(3) Choose the m eigenvectors associated with the m largest eigenvalues of $\underline{C}$, where $m < n$.

(4) Form an m x n transformation matrix A' whose rows are the m eigenvectors selected in Step (3).

(5) Reduce all original vectors $\underline{x}$ into a set of $\underline{y}$ vectors by means of the transformation

$$\underline{y} = A'\underline{x}$$

The resulting vectors are of lower dimensionality but, according to Eq. 1.3.2-8, this reduction results in the least mean square error.

REFERENCES

1. Hotelling, H., "Analysis of a Complex of Statistical Variables into Principal Components," J. Ed. Psychol., vol. 25, 1933.
2. Wintz, P. A., and Kurtenbach, A. J., "Waveform Error Control in PCM Telemetry," IEEE Trans. on Information Theory, vol. IT-14, Sept. 1968.
3. Wintz, P. A., and Kurtenbach, A. J., "Analysis and Minimization of Message Error in PCM Telemetry Systems," Tech. Rpt. TR-EE 67-19, School of Electrical Engineering, Purdue University, Lafayette, Indiana, December 1967.
4. Wintz, P. A. and Habibi, A., "Linear Transformations for Encoding 2-Dimensional Sources," Tech. Rpt. TR-EE 70-2, School of Electrical Engineering, Purdue University, Lafayette, Indiana, March 1970.
5. Habibi, A., and Wintz, P. A., "Image Coding by Linear Transformation and Block Quantization," IEEE Trans. on Communication Technology, vol. 19, no. 1, February 1971.
6. Ready, P. J. and Wintz, P. A., "Multispectral Data Compression through

Transform Coding and Block Quantization," LARS Inf. Note 050572, Purdue University, West Lafayette, Indiana, May 1972.

7. Habibi, A., "Hybrid Coding of Pictorial Data," IEEE Trans. on Communications, vol. COM-22, no. 5, May 1974.

8. Gupta, J. N. and Wintz, P. A., "Multi-Image Modeling," TR-EE 74-24, School of Electrical Engineering, Purdue University, West Lafayette, Indiana, September 1974.

9. Nilsson, N. J., Learning Machines, New York: McGraw-Hill, 1965.

10. Grettenberg, T. L., "Signal Selection in Communication and Radar Systems," IEEE Trans. Information Theory, vol. IT-9, pp. 265-275, October 1963.

11. Fu, K. S., and Chen, C. H., "Sequential Decisions, Pattern Recognition and Machine Learning," School of Electrical Engineering, Purdue University, Lafayette, Indiana, Tech. Rpt. TR-EE 65-6, April 1965.

12. Min, P. J., Landgrebe, D. A., and Fu, K. S., "On Feature Selection in Multiclass Pattern Recognition," Proc. 2nd Ann. Princeton Conf. on Information Sciences and Systems, pp. 453-457, 1968.

13. Min, P. J., "On Feature Selection in Multiclass Pattern Recognition," Ph.D. dissertation, January 1969, LARS Information Note 080568, August 1968, Tech. Rpt. TR-EE 68-17, Purdue University, Lafayette, Indiana.

14. Anderson, T. W., and Bahadur, R. R., "Classification into Two Multivariate Normal Distributions with Different Covariance Matrices," Am. Math. Stat., vol. 33, no. 2, pp. 420-431, 1962.

15. Fu, K. S., Landgrebe, D. A. and Phillips, T. A., "Information Processing of Remotely Sensed Agricultural Data," Proc. IEEE, vol. 57, no. 4, April 1969.

16. Fu, K. S., Min, P. J. and Li, T. J., "On Feature Selection in Pattern Recognition," Proc. 1968 Allerton Conf. on Circuit and System Theory, October 1968.

1.4 CANDIDATE FOR DETAILED STUDY

In this section a detailed study of a particular application area is carried out. The area which has been selected for this study is the remote detection and classification of agricultural crops. This choice is based on the importance and relevance of this problem in terms of human as well as economic considerations.

There are two principal items of information relative to agriculture remote sensing systems - (1) total crop acreage and (2) total expected yield. On a world-wide scale, the first item requires classification of ground crop information into one of approximately twelve different crops (pattern classes). Once the total acreage of each crop has been determined,

the yield can be calculated by assessing the state of health of each crop. It has been suggested that a minimum of three states of health ranging from normal to abnormal should be adequate for a close approximation of the expected yield from a particular crop [1]. Therefore, the crop classification problem may be divided into two stages. The first stage is used for classifying ground information into one of the crop categories. The second stage then classifies each into one of the health states. Assuming twelve crops and three states of health this is equivalent to a thirty-six-class pattern recognition problem.

1.4.1 Baseline Data Format

From Table 1.1.2(I), entry A5, the average resolution for crop classification is 40m, while the scanner swath width (field of coverage) is 185 km. From Figure 1.1.2(1), the desired number of bands (channels) for this application is seven. These bands consist of the four ERTS bands, plus two bands in the mid-infrared range and a band in the thermal region.

Using the data rate formula given in Section 1.1.2.4 with

S_w = 185 km

$R_L = R_p$ = 40 m

V = 6500 m/sec

N_C = 7

N_{bp} = 6

we find that the data rate for this application is

$D_R = 3.16 \times 10^7$ bits/sec

This data format will be referred to as the "baseline data format" in subsequent sections.

1.4.2 Preprocessing Algorithms

1.4.2.1 Coding

From the results of data compression algorithms studies, there appear to be two candidate algorithms applicable to on board data compression. They are: (1) transform coding (Section 1.3.2.1.1); and (2) Coding by BLOB (Section 1.3.2.1.2). Both of these algorithms are non-information preserving in the sense that an exact replica of the original data cannot be constructed from the coded data.

The studies further indicate that coding by BLOB achieves a higher

compression ratio than that obtained by transform coders, and this suggests that this approach is a better candidate algorithm for on board processing purposes, though it involves more computation.

To determine the performance of this method of data compression experimentally two sets of data were considered [2]. One was aircraft imagery (Laboratory for Application of Remote Sensing (LARS) Run #71053900) and the other was ERTS-1 imagery (LARS Run #73041801).

BLOBS are larger and fewer at high significance levels. The contour tracer output contains fewer initial point locations and gray levels that must be coded resulting in a higher compression ratio. Figure 1.4.2(1) illustrates the effect of various significance levels on the compression ratio.

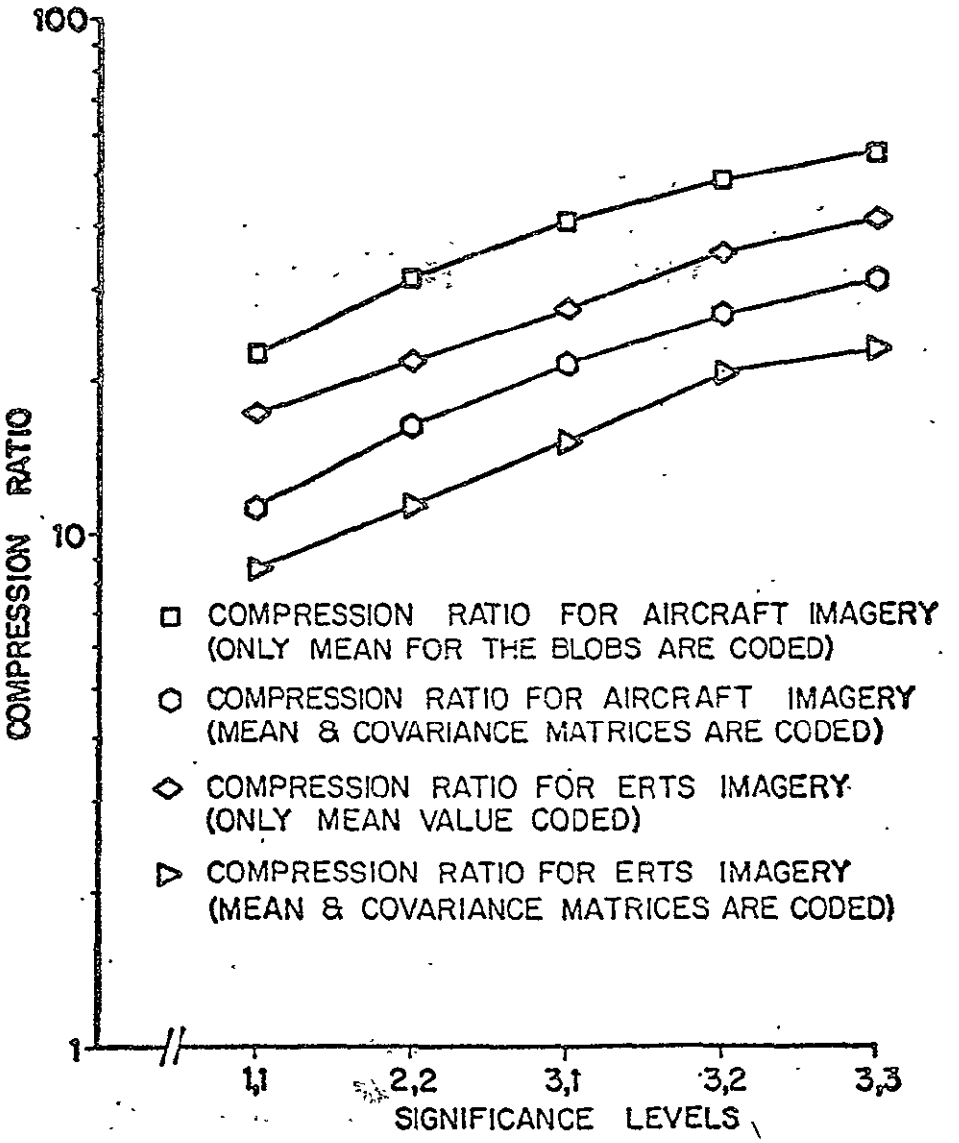


Figure 1.4.2(1) Data Compression Ratio for BLOB-Contour Encoding Scheme.

To determine the effect of data compression on classification accuracy in both these images, the blobs of the decoded data were classified using a minimum distance classification algorithm [2]. The classifier was implemented to distinguish corn, forage, soybean, forest and water in aircraft imagery, and wheat, pasture and other in ERTS-1 imagery. The results are shown in Figure 1.4.2(2). Presently, machine classification of multispectral data images is most commonly done on a point-by-point (spectral signature) basis. Both data sets were also classified according to their spectral signature analysis. It is evident from Figure 1.4.2(2) that compression ratios of the order of 12 and 32 for ERTS-1 and aircraft imageries, respectively, can be obtained without degrading the classification accuracy.

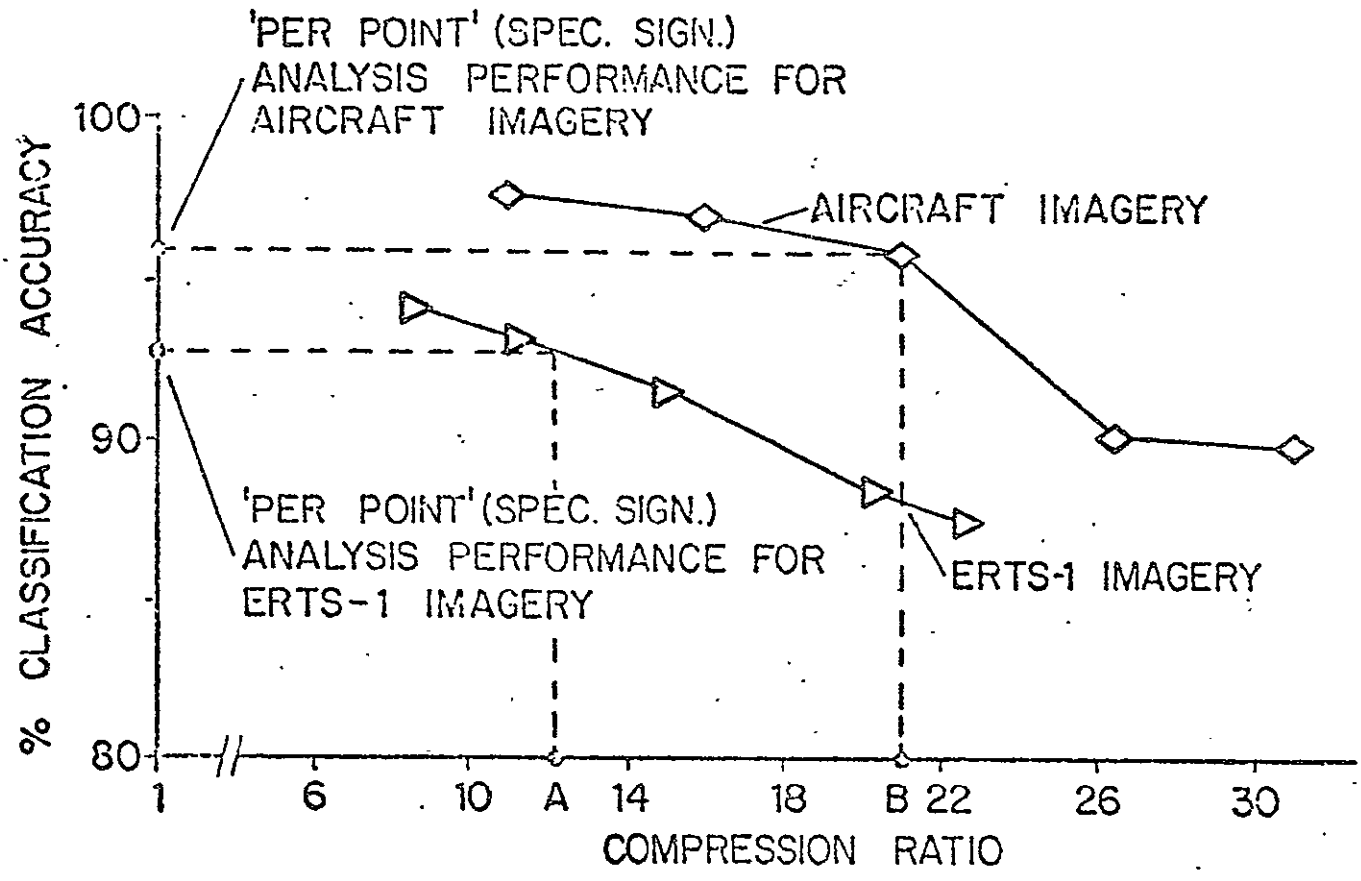


Figure 1.4.2(2) Compression Ratio vs Percent Classification Accuracy

1.4.2.2 Feature Selection

The candidate algorithm for feature selection is proposed to be based on eigenvector transformation discussed in greater detail in Section 1.3.2.2.

To illustrate the effect of this transformation on ERTS-1 data, an estimate of the 4 x 4 covariance matrix of the data was determined. The data was taken from an agricultural scene in Ogle, Lee and DeKalb Counties in the midwestern United States (LARS Run #72032806). The eigenvalues, matrix of eigenvectors and the variance in the individual channels is given in Table 1.4.2(I).

Table 1.4.2(I) Variance Eigenvalues and Eigenvectors of ERTS-1 Data

Variance	Eigenvalues	Matrix of Eigenvectors			
Channel 1-14.3	124.39	0.0491	-0.0327	0.8255	0.5612
Channel 2-27.6	41.97	0.5602	0.8024	0.1143	-0.1705
Channel 3-85.9	1.97	0.1927	-0.1147	-0.5526	0.8027
Channel 4-41.7	1.19	0.8040	-0.5846	0.0023	0.1079

Clearly the first two principal components of the transformed data contained 98.14% of the total variance contained in all four spectral bands of the original data. A set of training fields was selected to train the classifier on the data from the first two principal components to classify the data into three classes (corn, soybean and other). A scatter plot of the first principal component versus the second principal component is given in Figure 1.4.2(3). The decision boundaries shown yield a classification accuracy of 98% determined on the training fields.

This indicates that the eigenvector transformation of the ERTS-1 data reduces the data bulk by two orders of magnitude without any loss of information. In addition, since the most information is contained only in two dimensions, handling and processing of the transformed data becomes an easier task.

1.4.2.3 Conclusion

Although preprocessing plays a central role in data transmission, we have concluded that its usefulness is limited for on-board processing. The increase in hardware and software required to carry out the coding operations is a necessity if data is to be transmitted from the host system to some processing station. In most applications, especially in agriculture, the actual classification can usually be best carried out directly on the input data. For this reason, attention will be focused on the classifica-

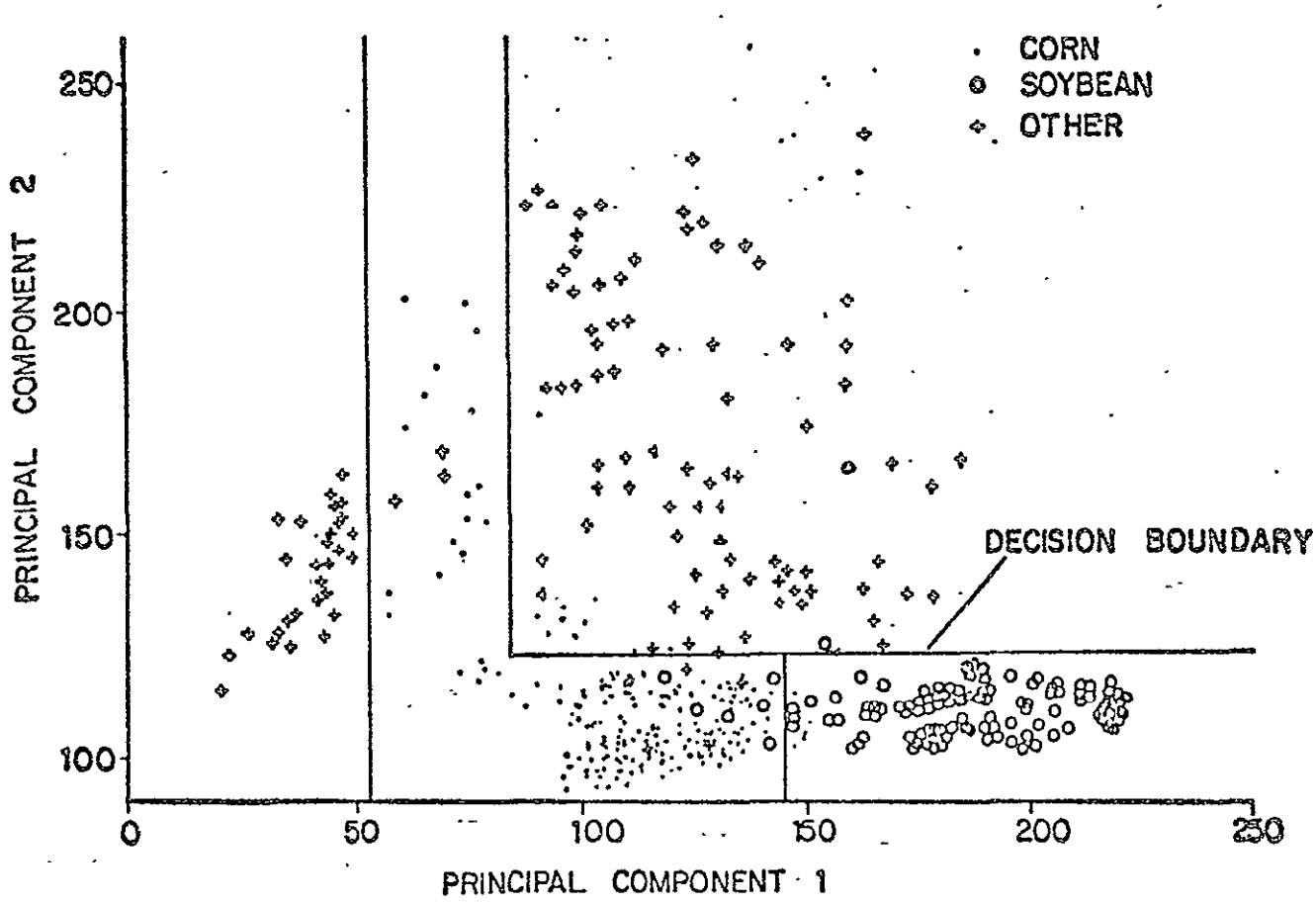


Figure 1.4.2(3) A Scatter Plot of First Principal Component vs the Second Principal Component of ERTS-1 Data

tion aspects of an on-board system for agricultural applications.

1.4.3 Analysis Algorithms

The analysis algorithms which are most suitable for on-board agricultural data processing are the maximum-likelihood and table look-up approaches discussed in Section 1.1.3.

Attention will be focused in Section 2 on the hardware and software requirements of these two algorithms; as will be seen, the two approaches pose quite distinct implementation problems.

REFERENCES

1. Private Communication with LARS personnel, Purdue University, West Lafayette, Indiana.
2. Gupta, J. N. and Wintz, P. A., "Multi-Image Modeling," TR-EE 74-24, School of Electrical Engineering, Purdue University, West Lafayette, Indiana, 1974.

3. Wilkins, L. C., and Wintz, P. A., "Studies on Data Compression, Part 1, Picture Coding by Contour," TR-EE 70-17, School of Electrical Engineering, Purdue University, West Lafayette, Indiana, 1970.

4. Hotelling, H., "Analysis of a Complex of Statistical Variables into Principal Components," J. Ed. Psychol., vol. 25, 1933.

5. Tou, J. T., and Gonzalez, R. C., Pattern Recognition Principles, Addison-Wesley Publishing Co., Reading, Massachusetts, 1974.

2 ON-BOARD PROCESSOR REQUIREMENTS

This section contains the computational requirements of the data analysis algorithms, a review of some computer architectures and organizations, a design of some computer architectures capable of handling the algorithm computational requirements, and a discussion of the on-board processor environmental effects.

In Section 2.1 we survey some earth resources user requirements and the data analysis algorithms for implementing them. In addition, we analyze in detail the basic computational requirements; i.e., the number of multiplications, additions, etc., required to implement these algorithms for the baseline data format.

In Section 2.2 we present some computer architectures and organizations with particular emphasis on pipeline, array processors and multiprocessors, since it is apparent that some sort of parallel processor will be required to keep up with the high data rates required by the users. Memory and software requirements are also discussed. A number of on-board processors were then designed to efficiently implement the maximum likelihood and table look-up algorithms at the required rates.

Finally, the environmental effects on the on-board processor for both the earth-synchronous and the sun-synchronous orbits are discussed in Section 2.3.

2.1 ALGORITHM COMPUTATIONAL REQUIREMENTS

This section deals with the basic computational requirements of preprocessing and analysis algorithms. It will be assumed throughout the following discussion that the estimation of the parameters for the eigenvector transformation of Section 2.1.1.2 and the training of the maximum-likelihood and perceptron algorithms of Section 2.1.2.1 are carried out on the ground.

The following notation is used throughout this section:

- n: number of features (channels)
- $\underline{x} = (x_1, x_2, \ldots, x_n)'$: pattern vector
- M: number of pattern classes
- N_P: total number of picture elements (pixels) in a data frame
- N_A: total number of additions
- N_M: total number of multiplications

N_C: total number of comparisons

N_S: total number of square roots

2.1.1 Preprocessing Algorithms

The preprocessing algorithms considered in this section are the data bulk reduction procedures discussed in Section 1.3.2. The two principal approaches discussed in that section are based either on an eigenvector transformation or BLOB coding. The computational requirements for these two appraoches are discussed below.

2.1.2 Basic Computational Requirements

(1) Eigenvector Transformation

The reduction of a pattern vector with q components to n components by means of an eigenvector transformation requires nq additions and nq multiplications. For a total of N_P pattern vectors the following totals apply:

$$N_A = nqN_P \qquad (2.1.2\text{-}1)$$

$$N_M = nqN_P \qquad (2.1.2\text{-}2)$$

where N_A and N_M are the total number of additions and multiplications, respectively.

(2) BLOB Coding

The BLOB coding algorithm operates on two rows of an image at a time. Letting W represent the number of columns per row we have the following processing requirements:

(a) Additions in calculating the mean - 4nW

(b) Additions in calculating F-test = 0

(c) Additions in calculating t-test = 5nW

(d) Multiplications in calculating mean and variance - n(n+1)W

(e) Multiplications in calculating F-test - 2nW

(f) Multiplications in calculating t-test - 6nW

(g) Additions in storing mean and variance - $nW + \frac{n(n+1)W}{2}$

Then, the total number of additions required to process two rows of an image is given by

$$\text{additions} = \left[10n + \frac{n(n+1)}{x}\right] W$$

and the number of multiplications is given by

$$\text{multiplications} = [8n + n(n+1)]\, W.$$

If we let V equal the number of rows in an image then we have the following total figures:

$$N_A = \left[10n + \frac{n(N+1)}{2}\right] \frac{V}{2} W \qquad (2.1.2\text{-}3)$$

$$N_M = [8n + n(n+1)] \frac{V}{2} W \qquad (2.1.2\text{-}4)$$

However, $VW = N_p$, the total number of elements in the digital image, so that

$$N_A = \frac{1}{2}\left[10n + \frac{n(n+1)}{2}\right] N_P \qquad (2.1.2\text{-}5)$$

$$N_M = \frac{1}{2}[8n + n(n+1)] N_P \qquad (2.1.2\text{-}6)$$

2.1.3 Storage Considerations

The storage requirements of the eigenvector transformation are simply those associated with the q x n transformation matrix. The transformation program itself consists of nothing more than the instructions required to multiply the transformation matrix by each input pattern vector. Therefore, the total storage requirements are qn words for the matrix plus that required to store the short multiplication program.

Experiments conducted at LARS with the BLOB algorithm indicate that approximately 35,000 32-bit words, including data storage, are required for coding with this approach.

2.1.4 Analysis Algorithms

2.1.4.1 Basic Computational Requirements

The operations required to implement the maximum-likelihood, perceptron, clustering, BLOB, and table look-up algorithms are examined in this section. Expressions for the number of arithmetic and other required operations such as comparisons and square roots are derived. The results are then compared in Section 2.1.2.2 using typical parameter values.

(1) Maximum-Likelihood Approach - Classification of a pattern vector $\underline{x} = (x_1, x_2, \ldots, x_n)'$ into one of M classes by the maximum-likelihood approach requires implementation of Eq. (1.1.3-5)

$$d_k(\underline{x}) = \ln p(\omega_k) - \frac{1}{2} \ln |\underline{C}_k| - \frac{1}{2}[(\underline{x}-\underline{m}_k)' \underline{C}_k^{-1}(\underline{x}-\underline{m}_k)] \qquad (2.1.2\text{-}7)$$

$$k = 1, 2, \ldots, M$$

The terms $\ln p(\omega_k)$, $\frac{1}{2} \ln |\underline{C}_k|$, and $\underline{C}_k^{-1}$ are estimated during training and, consequently, need not be calculated during the classification process.

Therefore, the principal computation for each $\underline{x}$ involves multiplications and additions in the last term of Eq. (2.1.2-7) above.

For n channels, $\underline{x}$ and $\underline{m}$ have n components and $\underline{C}_k^{-1}$ is an n x n matrix. Thus, the number of arithmetic operations is determined as follows:

(a) formation of $(\underline{x}-\underline{m}_k)$: n additions

(b) multiplication of $\underline{C}_k^{-1}$ by $(\underline{x}-\underline{m}_k)$: $n(n-1)$ additions; n^2 multiplications

(c) multiplication of $(\underline{x}-\underline{m}_k)$ by the results of (b): (n-1) additions; n multiplications

(d) multiplication of 0.5 by the results of (c): 1 multiplication

(e) addition of all three terms in Eq. (2.1.2-7): 3 additions

The results are, therefore,

number of additions = $n^2 + n + 2$

number of multiplications = $n^2 + n + 1$

These results are for one class and one pattern. Letting M equal the number of classes and N_P the total number of patterns to be classified yields:

$$N_A = (n^2 + n + 2)\, M\, N_P \qquad (2.1.2\text{-}8)$$

$$N_M = (n^2 + n + 1)\, M\, N_P \qquad (2.1.2\text{-}9)$$

In order to classify each observation $\underline{x}$ into one of M classes it is required that the maximum of M decision functions be determined, as was indicated in Section 1.1.3.1. Therefore, in addition to the total number of operations given in Eqs. (2.1.2-8) and (2.1.2-9), it is also necessary to perform N_C comparisons, where

$$N_C = N_P(M-1) \qquad (2.1.2\text{-}10)$$

This relationship follows from comparing M decision function values for N_P patterns.

(2) Perceptron Approach - Classification of a pattern vector into one of M classes by the perceptron approach requires implementation of Eq. (1.1.3-6)

$$d_k(\underline{x}) = w_{k1}\phi_1(\underline{x}) + w_{k2}\phi_2(\underline{x}) + \ldots + w_{kN}\phi_N(\underline{x}) + w_{k,N+1}$$

$$= \underline{w}_k' \underline{\phi}(\underline{x}) \qquad k = 1, 2, \ldots, M \qquad (2.1.2\text{-}11)$$

where the coefficients $\{w_{ki}\}$ and functions $\{\phi_i(\underline{x})\}$ are determined during training.

In order to establish the number of arithmetic operations required to implement the perceptron approach it is necessary to choose a particular set of functions $\{\phi_i(\underline{x})\}$. One of the most common approaches is to express the above decision functions in the form of an rth degree polynomial [1] by means of the following recursive formula:

$$d^r(\underline{x}) = \left(\sum_{P_1=1}^{n} \sum_{P_2=P_1}^{n} \cdots \sum_{P_r=P_{r-1}}^{n} w_{P_1 P_2 \cdots P_r} x_{P_1} x_{P_2} \cdots x_{P_r} \right) + d^{r-1}(\underline{x}) \tag{2.1.2-12}$$

where the first term $d^0(\underline{x})$ is given by $d^0(\underline{x}) = w_{n+1}$. In this equation the superscript indicates the degree of the polynomial. The class subscript has been dropped from the equation to simplify the notation. It is understood that there are M decision functions, one for each class.

The number of additions in Eq. (2.1.2-12) is given by:

$$\text{additions} = \frac{(n+r)!}{n!\, r!} - 1$$

and the number of multiplications by:

$$\text{multiplications} = \sum_{q=1}^{r} (q+1) \left\{ \frac{(n+q)!}{n!\, q!} - \frac{(n+q-1)!}{n!(q-1)!} \right\} - 2$$

These results are for one pattern and one class. Letting M equal the number of classes and N_P the total number of patterns to be classified results in the following total number of arithmetic operations:

$$N_A = \left[\frac{(n+r)!}{n!\, r!} - 1 \right] MN_P \tag{2.1.2-13}$$

$$N_M = \left[\sum_{q=1}^{r} (q+1) \left\{ \frac{(n+q)!}{n!q!} - \frac{(n+q-1)!}{n!(q-1)!} \right\} - 2 \right] MN_P \tag{2.1.2-14}$$

As in the case of the maximum-likelihood approach, it is also necessary to perform N_C comparisons, where

$$N_C = N_P(M-1) \tag{2.1.2-15}$$

(3) Clustering Approach - The clustering approach discussed in Section 1.1.3.4 operates directly on the complete set of data to be classified. This is in contrast with the two approaches discussed above which classify one pattern vector at a time. The number of operations required to implement the clustering approach is determined as follows:

(a) sample mean for each class M, $\mu_j = \frac{1}{L} \sum_{i=1}^{L} \underline{x}_i$; $j = 1, 2, \ldots, n$

additions: n L M

multiplications: n M

where $L = N_p/M$. It is noted that M, the class or mode of a cluster, is not specified as in the case of the two approaches discussed above, but is instead determined by the algorithm based on the properties of the data being analyzed.

(b) sample variance $\sigma_j^2 = \frac{1}{L-1} \sum_{i=1}^{L} (x_{ij} - \mu_j)^2$; $j = 1, 2, \ldots, n$

additions: 2 n L M; multiplications: n(L+1) M

(c) mode (class center $m_{kj} = \mu_j + \sigma\left[\frac{2(k-1)}{M-1}\right] - 1$; $j = 1, 2, \ldots, n$

additions: (3M+1) n; multiplications: (3M+L) n: $k = 1, 2, \ldots, M$

(d) Euclidean distance $D = \sum_{i=1}^{n} (x_i - m_{ki})^2$ $k = 1, 2, \ldots, M$

additions: $n N_p M$; multiplications: $n N_p M$

Let I be the number of iterations allowed for convergence of the algorithm. Then, the total number of computations involved in assigning all the data vectors to one of M classes is:

$$\text{additions} = (n L M + 2n L M + (3M + 1)n + n N_p M) I$$
$$= (n N_p + 2n N_p + 3n M + n + n N_p M) I$$
$$= n I[M + 3)N_p + 3M + 1]$$

$$\text{multiplications} = (n M + n N_p + n M + 3Mn + n + n N_p M) I$$
$$= n I[5 M + (M + 1) N_p + 1]$$

to determine the distinctness of classes we have the following additional computations:

(e) Swain-Fu distance: $\nabla = \frac{\sqrt{A_1 A_2}}{\sqrt{A_1 + A_2}}$

where $A_k = \text{Tr}\left[\underline{C}_k^{-1}(\underline{m}_1 - \underline{m}_2)(\underline{m} - \underline{m}_2)'\right]$. Calculating $\underline{C}_k^{-1}$ involves n^3 additions and n^3 multiplications. Then, the calculation of A_k involves $2n^3 + 1$ additions and $2n^3 + n^2$ multiplications. Finally, the calculation of ∇ for two classes involves:

additions: $4n^3 + 3$; multiplications: $4n^3 + 2n^2 + 2$;

square roots: 3

Considering next every pair of classes yields:

additions: $\frac{M(M-1)}{2}[4n^3 + 3]$; multiplications: $\frac{M(M-1)}{2}[4n^3 + 2n^2 + 2]$;

square roots: $\frac{3M(M-1)}{2}$.

The total number of computations required for clustering is then:

$$N_A = n\,I\,[(M+3)N_P + 3M + 1 + \frac{M(M-1)}{2}\,[4n^3 + 3] \qquad (2.1.2\text{-}16)$$

$$N_M = n\,I\,[5M + (M+1)N_P + 1] + \frac{M(M-1)}{2}\,[4n^3 + 2n^2 + 2] \qquad (2.1.2\text{-}17)$$

$$N_S = \frac{3M(M-1)}{2} \qquad (2.1.2\text{-}18)$$

where N_S is the number of square roots.

In addition it is also necessary to perform

$$N_C = N_P(M - 1) \qquad (2.1.2\text{-}19)$$

comparisons.

(4) Blob Classification - After the data have been compressed by the BLOB preprocessing algorithm, it may be desired to classify each blob into one of M classes. Two principal approaches are used in the BLOB classification algorithm: (1) minimum-distance classification, and (2) maximum-likelihood classification. Letting N_B represent the number of blobs, we have the following figures for the minimum-distance approach:

$$N_A = (2n^3 + n^2 + n + 2)\,MN_B \qquad (2.1.2\text{-}20)$$

$$N_M = (2n^3 + n^2 + n + 5)\,MN_B \qquad (2.1.2\text{-}21)$$

$$N_C = N_B(M - 1) \qquad (2.1.2\text{-}22)$$

In addition, there are $N_B M$ log operations.

The maximum-likelihood figures are obtained from Eqs. (2.1.2-8) through (2.1.2-10) with $N_P = N_B$.

$$N_A = (n^2 + n + 2)\,MN_B \qquad (2.1.2\text{-}23)$$

$$N_M = (n^2 + n + 1)\,MN_B \qquad (2.1.2\text{-}24)$$

$$N_C = N_B(M - 1) \qquad (2.1.2\text{-}25)$$

(5) Table Look-Up - The principal operation in the table look-up approach discussed in Section 1.1.3.3 consists simply of

$$N_A = (11 + M)\,MN_P \qquad (2.1.2\text{-}26)$$

additions. Although this technique is attractive from a computational point of view, it must be kept in mind that the actual program and storage requirements are considerably more complex than the approaches discussed above.

2.1.4.2 A Numerical Comparison

The results obtained in the previous section are best appreciated by

comparing the number of required operations on an absolute basis. Table 2.1.1(I) presents such a comparison for the following typical parameter values:

$n = 4$.

N_P = (3300 rows) x (2500 columns) = 8.25×10^6 pixels

$M = 5$

$I = 15$

Number of points/blob = 10. Then $N_B = N_P/10 = 8.25 \times 10^5$

These values are typical of those used in connection with ERTS-1 MSS data.

Table 2.1.1(I) A comparison of Analysis Algorithms for One Data Frame

Approach	Equations Used (Section 2.1.2)	Number of Additions	Number of Multiplications	Number of Other Operations
Maximum-Likelihood	(8),(9),(10)	9.075×10^8	8.663×10^8	N_C: 3.30×10^7
Perceptron	(13),(14),(8), [r=2], (15)	5.775×10^8	1.485×10^9	N_C: 3.30×10^7
Clustering	(16,(17), [I=15],(18), (19)	3.960×10^9	2.970×10^9	N_S: 30 N_C: 3.30×10^7
BLOB [Min. Dist.]	(5),(6),(20), (21),(22)	8.251×10^8	8.044×10^8	Log: 4.13×10^6 N_C: 3.30×10^6
BLOB [Max-Likelihood]	(5),(6),(23), (24),(25)	1.114×10^9	1.040×10^9	N_C: 3.30×10^6
Table Look-Up	(26)	6.600×10^8	--	--

The results obtained in Table 2.1.1(I) for the perceptron approach assume decision boundaries of the second degree (r=2). This assumption is made to arrive at a meaningful comparison between the perceptron and maximum-likelihood approaches, since the latter implements second degree boundaries when the data are assumed to have Gaussian properties. This is by far the most common assumption followed in processing remotely-sensed data by the maximum-likelihood approach.

The results shown for the BLOB approach take into account Eqs. (2.1.2-5) and (2.1.2-6). Although these equations are associated with preprocessing functions, they are also required in classification. In other words, preprocessing is a necessary step prior to classification by the blob approach.

2.1.4.3 Storage Considerations

Implementation of the maximum-likelihood algorithm for M classes requires M storage locations for $\ln p(\omega_k)$, M storage locations for $\frac{1}{2} \ln |\underline{C}_k|$, Mn storage locations for the mean vectors $\underline{m}_k$ and Mn^2 storage locations for the matrices $\underline{C}_k^{-1}$. The actual classification algorithm can be implemented in less than 2500 storage words. In addition, it is necessary to have a buffer for the input data. The size of the buffer depends on the processor speed.

The perceptron algorithm requires $M(n+r)!/n!r!$ storage locations for the coefficients of the decision functions. The algorithm itself can be stored in less than 2000 words. A buffer is also required for this approach.

Experiments conducted at LARS with the clustering algorithm indicate that 57,000 32-bit words are required to process 40,000 data vectors (with n=4). This includes program and data storage. In addition, a buffer whose size depends on processor speed is required.

The storage requirements for the classification aspect of the BLOB algorithm depends on the approach taken. For the minimum-distance approach it is necessary to store the mode centers along with the covariance matrix of each blob. Since the number of blobs varies with the application the storage requirements must be determined by actual experimentation. The classification algorithm itself can be stored in less than 2000 words and a buffer is also required.

The maximum-likelihood approach to blob classification has the same requirements as the general maximum-likelihood algorithm discussed above.

Experiments with the table look-up approach indicate that this approach requires in the order of 31,000 words of storage [5,6]. This figure applies to a maximum of 24 classes with n=4. A buffer whose size depends on processor speed is also required.

The above comparisons are intended only as rough guidelines of algorithm storage requirements. The actual figures for each algorithm depend on a number of factors such as processor instruction set and speed, program optimization, and the application area.

2.1.4.4 Determination of the Algorithm Constants

The constants in the preprocessing and analysis algorithms depend on

the particular application. For example, the $\underline{m}_k$, $|\underline{C}_k|$, etc., in the maximum likelihood algorithm decision functions given by equation (2.1.1-7) depend on the statistics of the data classes to be classified. The on-board processor must implement this equation, but the processor design, speed of operation, etc., are invariant to the numerical values of the constants.

There are two possibilities for determining the numerical values for these constants: (1) the spectral signature bank approach; and (2) training.

The signature bank approach assumes that the required numbers for each application are known and stored so that for any particular application the required set of numbers can be looked up in the bank. This approach is not feasible at this time because the constants for any particular application depend on many different parameters in a complicated way that is not well understood at this time. For example, the spectral signature of corn depends on lighting conditions, soil characteristics, the amount of moisture in the leaves, the point in the growing cycle, etc. When, if ever, the dependence of the constants on these parameters becomes well enough understood to allow application of this approach is uncertain.

The second approach is called training and requires that a sample of each type of data to be classified be obtained and the required numerical values measured from this data sample. For example, the $\underline{m}_k$ in the maximum likelihood algorithm are the mean intensity values in each spectral band. The idea is to choose a typical data sample (called a training sample), make the measurements from this sample, and use them in the decision function.

If the training approach is used there are two possibilities for training the classifier; i.e., obtaining the numerical values of the constants. The first approach is to collect the data sample and compute these parameters on board the satellite. The second approach is to transmit the raw data sample to a ground computer, compute the parameters on the ground, and transmit these back to the satellite. The latter approach appears to be the most feasible at this time. The entire problem of how to go about collecting training samples, verifying ground truth, etc., is not well understood at this time. NASA is presently expending considerable resources in an attempt to solve these problems. The results of these studies will complement the results of our study.

REFERENCES

1. Tou, J. T., and Gonzalez, R. C., Pattern Recognition Principles, Addison-Wesley Publishing Co., Reading Mass., 1974.
2. Crave, R. B., Malila, W. A., and Richardson, W., "Stability of the Normal Density Assumption for Processing Multispectral Scanner Data," IEEE Trans. Geos. Ele., GE-10, pp. 158-165, 1972.
3. Gupta, J. N., and Wintz, P. A., "Multi-Image Modeling," TR-EE 74-24, School of Electrical Engineering, Purdue University, West Lafayette, Indiana, September, 1974.
4. Hotelling, H., "Analysis of a Complex of Statistical Variables into Principal Components," J. Ed. Psychol., vol. 25, 1933.
5. Eppler, W. G., Helmke, C. A., and Evans, R. H., "Table Look-Up Approach to Pattern Recognition," Proc. of the International Symposium on Remote Sensing of the Environment," University of Michigan, May 1971.
6. Eppler, W. G., "An Improved Version of the Table Look-Up Algorithm for Pattern Recognition," Proc. of the 9th International Symposium on Remote Sensing of the Environment, University of Michigan, April 1974.

2.2 COMPUTER ARCHITECTURE AND ORGANIZATION

2.2.1 Historical Data

Historically, computer organization has developed along the classical lines of the Von Neuman machine [1]. When the arithmetic logic unit (ALU) is designed to operate on n-bit operands in a time-sequential mode, the operation of the computer is termed serial mode. Traditionally, a computer whose ALU is serial has been classified as a serial computer. If the ALU execution on an n-bit operand is performed separately and simultaneously, the mode of operation is parallel. Approximately n times as much hardware is required in a parallel organization as compared to a serial organization. This is somewhat offset by the fact that the serial control unit is more complicated than that of a parallel organization. However, the parallel structure executes operations nearly n times faster [2,3].

First generation computers were serial. The performance/cost ratio was relatively small. To increase the speed of operation, second generation machines were designed with a parallel structure. Transistor and better random access memory technologies significantly increased the performance/cost ratio. The user base broadened with the development of higher level programming languages which generated a demand for more computing power.

Third generation computers use integrated circuit technology and improved peripheral and memory technologies to continue the improvement in the performance/cost ratio. The organization of these computer systems still uses a parallel organization. System resources are utilized more efficiently in a few of these systems by adding a second processing unit. In this organization, independent processors, each with identical physical organization, performs relatively independent functions. The memory unit is a resource shared by each processor. Although the processor cost has increased two-fold, the performance has increased by a lesser factor. Even in third generation machines the Von Neuman influence is evident in most computer organizations. Flynn [4] and Higbie [6] have presented new concepts which attempt to avoid the ambiguous term "parallelism" in categorizing computer organizations.

Four major classifications for computer processors have been defined:

(1) Single Instruction Stream-Single Data Stream (SISD) processors;

(2) Multiple Instruction Stream-Single Data Stream (MISD) processors;

(3) Single Instruction Stream-Multiple Data Stream (SIMD) processors;

(4) Multiple Instruction Stream-Multiple Data Stream (MIMD) processors.

The Instruction Stream is the sequence of instructions as executed by the processor while the Data Stream is the sequence of data called for by the instruction stream. Computer organizations are described by the multiplicity of hardware subsystems utilized to service the Instruction and Data Streams. No longer is a system structure identified by the physical construction of its ALU, but rather by the number of instruction streams, the number of data streams, and their interactions.

2.2.2 Parallel Processor

SISD Computer. This system has a single processor. Several computer systems using this basic organization achieve computing power by overlapping the execution of sequential instructions. That is, the execution of the next instruction begins as soon as the first phase of the current instruction is completed. When conditional instructions are encountered, "dummy" instructions are executed until the decision or conditional process is complete. The CDC 6600 series [5] and the IBM 360/90 series [7] are examples of computers falling into this class.

2.2.3 Pipeline Processor

MISD Computer. This structure has become known as a pipeline computer [8,9]. The whole philosophy of pipelining is to implement a series of independent processors, or processor subsystems, in the form of a long pipeline. As a computation flows through the collection of processors, each processor performs a specific sub-operation. Several computations may be in process in distinct processors at the same time. The overlapping of computations is made possible by locating output registers with each segment of the pipeline. The CDC STAR [10] and the TI-ASC [11] are the most significant examples of computers in this class.

2.2.4 Array Processor

SIMD Computer. Several computers (SPAC, [12] SOLOMON, [13] and ILLIAC IV [14]) have been proposed to process multiple data streams. ILLIAC IV is the only one of consequence to have achieved an operational status. Computers falling in this class of structure are described alternately as array processors, associative array processors, parallel processors, vector processors, or othogonal processors depending upon their organizational differences. Communication between processors is predetermined and instruction stream execution operates simultaneously on all data streams. No overlapping of instruction execution is presently used in this organization.

2.2.5 Multiprocessor

MIMD Computer. This is an example of a multiprocessor system with either shared or multiple memories in which more than one program is executing on its own data (i.e., multiprogramming). The Carnegie-Mellon system [15] is an example of such an organization and employs a number of independent minicomputers with a shared memory.

Looking at these organizations with respect to the data acquisition from a multispectral scanner, an obvious observation is that data acquired from the multispectral scanner represents a multiple data stream. A SIMD organization is one possible consideration. However, to achieve the data rate-constraints, a MIMD structure represents an alternate approach.

2.2.6 Memory

In a small general-purpose digital computer the memory is the most expensive of the various computer subsystems. The memory size must be in

excess of the requirements set forth by the average job that is anticipated, must be fast to prevent a limiting of the computer performance, and must be low power to reduce power supply costs. Small computers today are turning, almost exclusively, to semiconductor memories which are out-performing core in both speed and cost for small to medium-sized memories [16,17].

The architecture of contemporary machines has included a single memory which is used for both program storage and temporary storage. With the advent of the microprocessor and the semiconductory memory, the memory has been partitioned into two separate memories, program memory and working memory. Because of the volatility of semiconductor memories, the program memory is a read-only memory (ROM) and the working memory is a random-access memory (RAM).

2.2.6.1 Program Memory

In examining the algorithms for earth resources applications the programs are not lengthy and complex since speed is the predominating factor; therefore, the program memory can be relatively small. Projections indicate that the on-board computer will be of the multi-instruction multiple-data-input type in the 1980-1990 time span. To implement this type of architecture, one method is to interconnect a number of microprocessors; one or more would be in each stage of the pipeline, and to accomplish the throughput requirements, pipelines could be parallelled. Each microprocessor or group of microprocessors could be assigned a small program memory. At present the access time for a semiconductor memory is much faster than the cycletime of the microprocessor so that something like five microprocessors could share a memory. Since both microprocessor speed and memory speed will increase, this ratio may still stand in the 1980-1990 time frame.

The program memory must be non-volatile and there must be a convenient way to change the program. Today, this is accomplished in programmable read-only memories (PROM). Electrically-alterable read-only memories (EAROM) will be available soon. The common type of PROM today requires an ultra-violet light source to erase the memroy. By 1980 the bits of the EAROM can be selected randomly for writing into either state.

2.2.6.2 Working Memory

The working memory criteria can be satisfied by a random-access

volatile memory (RAM). Presently these are either bipolar or metal-oxide semiconductor (MOS). The bipolar RAMs enjoy on the average a speed advantage and the MOS RAMs are superior in their efficient use of power. Currently the bipolar chips contain 4000 bits, require 6 mil^2/bit and an access time of 50-100 ns. This performance is expected to improve at a substantial rate through the 1980's.

2.2.6.3 Bulk Memory

The on-board earth resources computer will not need an on-board bulk memory to store the MSS signals for later processing. An on-board computer is being designed to operate in real time and to immediately process all the data as it is produced by the sensors.

The output of the computer contains processed data in the form of classification data for each pixel. For the candidate format discussed in Section 1.4 there are four spectral bands with 6 bits of gray-level quantization for each pixel. There are 1.3 μs per pixel so that the input data rate is 31.2 MHz.

The output data is classified into 12 classifications or four bits per pixel. This gives an output data rate of 5.2 megabits per second. At this point the on-board processor has compressed the data by a factor of six to one. If these output data are to be preserved, an on-board magnetic tape recorder would be required or a wide band data link would be necessary between the spacecraft and the ground. Another approach is to sort the classification data into twelve registers aboard the spacecraft. The registers could be read out at the end of each scan line indicating the combined type and quality of crop along the scan line. This greatly reduces the output data rate but sacrifices the knowledge of the exact location and quality of a particular crop.

2.2.6.4 New Memory Technologies

A number of new technologies [17] have been developed in the memory area. One of the most promising is the silicon-on-sapphire complementing MOS memory element. These circuits offer one very important advantage. The insulation resistance between the components is extremely high and the decrease in substrate parasitic capacitance is so great that it appears that MOS circuits built on sapphire could be as fast as or faster than bipolar integrated circuits.

The metal-alumina-silicon field-effect transistor is especially well-adapted for use as an electronically-alterable read-only memory. This transistor is the same as a conventional MOS except that the alumina is used in place of oxide as the gate dielectric material. The characteristics of the alumina dielectric material can be changed by applying a voltage to the gate. There is a critical gate voltage level which, when reached, alters the transistor's threshold voltage. Shifting the threshold voltage from one level to another corresponds to the storage of one bit of information. Fortunately, the change in threshold voltage is non-volatile allowing the device to be especially suited as a program memory.

Charge-coupled devices appear to have their greatest potential in replacing disk and drum storage since the device works an excellent shift register. Presently there appears to be no need for high-capacity shift registers for on-board earth-Resources computers.

For severe environment applications the Metal-Nitride Oxide Semiconductor (MNOS) has shown great promise coupled with the fact that it is a non-volatile memory element. The devices have been applied in a large memory which replaces disk or drum [19]. An MNOS memory could serve either as a program memory or a working memory or both.

2.2.7 Software

The requirements on the software for the on-board processing of earth-resources data are basically different from the requirements of ground-based processing software. This stems from the fact that the ground computers have traditionally been large-scale general-purpose digital computers. These computers are used for many tasks and may even be assigned to completely different roles at the conclusion of the earth-resources mission. The on-board processor is dedicated to a specific task and more of the system implemented in hardware than in software. This is complemented by the speed advantage of hardware over software.

A little over a decade ago the software costs were less than the costs of the hardware. With the introduction of Large Scale Integration (LSI), the cost of the computer has declined considerably; however, the labor costs for software design has continued to rise. Some estimates put the cost of a single instruction in a complicated program in the neighborhood of ten dollars; another figure is that one hour of labor is required for each two instructions in a program [20]. The trend, then, is to put more

software functions into hardware. For the earth-resources on-board computer this trend is emphasized even more strongly. Since speed is the predominant factor, as many of the computer functions as possible will have to be done in hardware. In early computers binary addition was the only function implemented in hardware. To increase speed, more and more functions have been implemented in hardware.

The on-board earth-resources computer does not need all the features of a large general-purpose machine, but it does need speed. It seems practical, then, to do the time consuming jobs by hardware and use software where speed is not essential so that flexibility can be maintained. As an example, the maximum-likelihood algorithm involves primarily additions and multiplications. It was very evident from the start that multiplication could not be done in software and that either hardware multiplication or a table look-up multiplication method would be employed. The addition function, of course, is readily available in hardware form in all CPU chips or arithmetic logic units.

Since high speed is a prime requirement of the on-board computer, the programs must be short, simple and efficient. Efficiency dictates the use of an assembly language for the programming. In the maximum-likelihood algorithm the total number of operations required is small; further, the pipelining technique breaks down the computation into individual tasks. The microprocessors are assigned to each task, and each task has its own program. These programs are very short since the total operation has been broken down into parts (tasks). The individual programs would reside in an electrically-alterable read-only memory (EAROM). By ground command to the spacecraft, coefficients could be changed in the program to match ground-truth measurements.

An advantage of utilizing identical microprocessors in the on-board computer is that programming becomes fairly standard. A cross-assembler would be used to develop each program for each task for loading into the EAROMs. The cross-assembler program would be run on any available computer of adequate capacity. A higher-level language would be of little use since it is not envisioned that a large amount of program will be needed; also, a higher-level language is not as efficient.

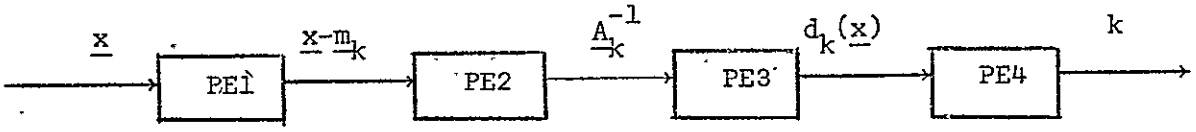
2.2.8 Computer Architecture Examples

Attention will be focused on implementations of the maximum-likelihood

and table look-up algorithms. Section 2.2.8.1 deals with a microprocessor implementation of the maximum-likelihood algorithm. This is followed in Section 2.2.8.2 with a microprocessor implementation of the table look-up approach. Section 2.2.8.3 deals with a microprocessor implementation of an expanded form of the maximum-likelihood algorithm. This is followed in Section 2.2.8.4 by a hardware implementation of this approach. Finally, Section 2.2.8.5 presents a hardware implementation of the table look-up algorithm. All examples are based on the baseline data format defined in Section 1.4.

2.2.8.1 Microprocessor Organization for the Maximum-Likelihood Algorithm

Four processing elements are pipelined to execute the maximum likelihood function for one class as shown in Figure 2.2.8(1). Each processing element (PE) is organized about a microprocessor; the performance measure is the execution time. A PE utilizing the INTEL 8080 is shown in Figure 2.2.8(2).



$$\underline{A}_k^{-1} = (\underline{x}-\underline{m}_k)' \underline{C}_k^{-1}$$

$$d_k(\underline{x}) = C_k - 1/2\ \underline{A}_k^{-1}\ (\underline{x}-\underline{m}_k)$$

Figure 2.2.8(1) Pipelined Organization for Executing the Maximum Likelihood Algorithm

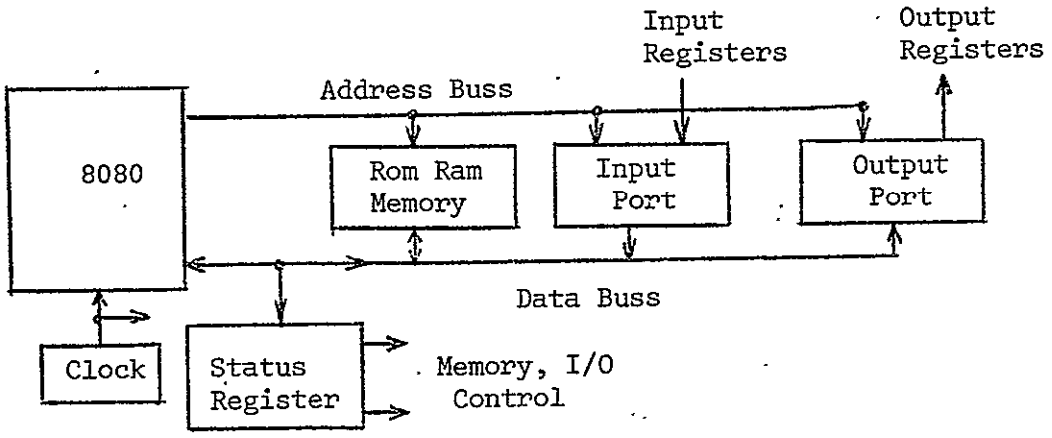


Figure 2.2.8(2) Microprocessor System Organized About an INTEL 8080 Microprocessor.

The vector $\underline{x}$ is input to PE1. PE1 forms $\underline{x}-\underline{m}_k$ and outputs the result to PE2 which generates $(\underline{x}-\underline{m}_k)\ \underline{C}_k^{-1}$ and outputs it to PE3 where $d_k(\underline{x})$ is computed. PE4 determines the classification category, k.

Processing Element PE1.

The n input registers and nM output registers are required for n spectral bands and M classification categories. In order to determine the time required to compute $\underline{x}-\underline{m}_k$ the required code was written using the instruction set of the 8080 as illustrated in Table 2.2.8(I). This computation must be repeated for each of the m categories, k = 1, 2, ..., M. The total execution time is

$$TE = (13.5M + 7.5)n\ \mu s.$$

Table 2.2.8(I) Program for Generating $\underline{x}-\underline{m}_k$ in PE1

MNEMONIC	OPERAND	BYTES	COMMENT
LXI	H, ARRAY	3	initialize $\underline{m}_{k1}$ array
IN	X1	2	input x_1 component of $\underline{x}$
MOV	D,A	1	
SUB	M11	1	generate $x_1 - m_{11}$
OUT	X11	2	output $x_1 - m_{11}$ component
INX	H	1	increment HL
MOV	A,D	1	
SUB	M21	1	generate $x_1 - m_{21}$
OUT	X21	2	output $x_1 - m_{21}$ component
INX	H	1	increment HL
⋮		5(M-3)	generate and output $x_1 - m_{i1}$ component i = 3, . . , M - 1
MOV	A,D	1	
SUB	MM1	1	generate $x_1 - m_{M1}$
OUT	XM1	2	output $x_1 - m_{M1}$ component

This segment of program is written for the x_1 component of $\underline{x}$, but must be executed for each of the remaining components $x_2, \ldots, x_n$.

Processing Element PE2.

PE2 computes the product

$$\underline{A}_k^{-1} = (\underline{x}-\underline{m}_k)'\ \underline{C}_k^{-1}.$$

nM input registers and 2nm output registers are required. Table 2.2.8(II) shows the 8080 code required to use table look-up to form the products. x_i-m_{1i} and the coefficients of $\underline{C}_1^{-1}$ are used to address a multiplication ROM from which the product is read. This code must be executed for each

Table 2.2.8(II) Program for Generating $\underline{A}_1^{-1}$ in $PE2_1$

MNEMONIC	OPERAND	BYTES	COMMENT
LXI	H, TEMP	3	initialize location of n temporary storage locations
LXI	B, CROW1	3	initialize row 1 in $\underline{C}_1^{-1}$ matrix
IN	X11	2	input $x_1 - m_{11}$ component
OUT	X11	2	output $\bar{x}_1 - \bar{m}_{11}$ to PE3
MOV	D,A	1	
LDAX	B	1	
MOV	E,A	1	
LDAX	D	1	generate product $(x_1 - m_{11}).C_{11}^1$
MOV	T1, A	1	T1 ← (A)
INX	B	1	increment BC
INX	H	1	increment HL
⋮		6(n-2)	generates products $(x_1-m_{11}).C_{1i}^1$ i=2, ..., n-1
LDAX	B	1	
MOV	E, A	1	
LDAX	D	1	generates product $(x_1-m_{11}).C_{1n}^1$
MOV	Tn, A	1	Tn ← (A)
LXI	H, TEMP	3	
LXI	B, CROWi	3	initialize row i in $\underline{C}_1^{-1}$ matrix
IN	X1i	2	input x_i-m_{1i} component
OUT	X1i	2	output $\bar{x}_i-\bar{m}_{1i}$ to PE3
MOV	D, A	1	
LDAX	B	1	
MOV	E, A	1	
LDAX	D	1	generates product $(x_i-m_{1i}).C_{i1}^1$
ADD	T1	1	accumulate sum in T1
MOV	T1, A	1	
INX	B	1	
INX	H	1	
⋮		7(n-2)	product $(x_i-m_{1i}).C_{ij}^1$ j = 2, ..., n-2
LDAX	B	1	
MOV	E, A	1	
LDAX	D	1	product $(x_i-m_{1i}).C_{in}^1$
ADD	1	1	
MOV	1, A	1	
LXI	, TEMP	3	
LXI	, CROWn	3	initialize row n in $\underline{C}_1^{-1}$ matrix
IN	X1n	2	input x_n-m_{1n} component
OUT	X1n	2	output $\bar{x}_n-\bar{m}_{in}$ to PE3
MOV	D, A	1	
LDAX	B	1	
MOV	E, A	1	
LDAX	D	1	product $(x_n-m_{1n}).C_{n1}^1$
ADD	T1	1	
OUT	A11	2	output A_{11}^{-1} of $\underline{A}_1^{-1}$
INX	B	1	
INX	H	1	

$k = 1, 2, \ldots, M$. The program execution time in PE2 is

$$TE = (21.5n + 15.5)nM\ \mu s.$$

Processing Element PE3.

PE3 accepts the 2nm outputs of PE2 and generates

$$d_k(\underline{x}) = C_k - 1/2\ \underline{A}_k^{-1}(\underline{x}-\underline{m}_k)$$

with 2nm input registers and m output registers. A program is presented in Table 2.2.8(III). The execution time is

$$TE = (23.0n + 15.5)M\ \mu s.$$

Table 2.2.8(III) Program For Generating $d_1(\underline{x})$ in PE3

MNEMONIC	OPERAND	BYTES	COMMENT
LXI	H, CONS	3	initialize location of constant C_1
IN	X11	2	input $x_1 - m_{11}$ component
MOV	D, A	1	
IN	A11	2	input A_{11}^{-1} component of $\underline{A}_1^{-1}$
MOV	E, A	1	E ← (A)
ODAX	D	1	product $A_{11}^{-1}.(x_1-m_{11})$
MOV	B, A	1	B ← (A)
IN	X12	2	
MOV	D, A	1	
IN	A12	2	
MOV	E, A	1	
LDAX	D	1	product $A_{12}^{-1}.(x_2-m_{12})$
ADD	B	1	
MOV	B, A	1	accumulate sum in B
⋮		9(n-3)	$A_{1i}^{-1}(x_i-m_{1i})$ $i=3, \ldots, n-1$
IN	X1n	2	
MOV	D, A	1	
IN	A1n	2	
MOV	E, A	1	
LDAX	D	1	product $A_{1n}^{-1}(x_n-m_{1n})$
ADD	B	1	
RAR		1	divide by 2
			$1/2(\underline{x}-\underline{m}_1)'\underline{C}_1^{-1}(\underline{x}-\underline{m}_1)$
MOV	B, A	1	
MOV	A, C1	1	
SUB	B	1	generates $d_1(\underline{x})$
OUT	D1	2	outputs $d_1(\underline{x})$

Processing Element PE4.

PE4 compares the M values of $d_k(\underline{x})$ to determine the classification.

This requires m input registers and a single output register. Table 2.2.8 (IV) describes the program for PE4. The time required is

$$TE = (19.5M + 2.5)\mu s$$

Table 2.2.8(IV) Program for Determining Classification k in PE4

	MNEMONIC	OPERAND	BYTES	COMMENT
	MVI	C, 1	2	C ← 1
	MVI	E, 2	2	E ← 2
	IN	D1	2	input d_1 ($\underline{x}$)
	MOV	B, A	1	B contains largest
	IN	D2	2	input d_2 ($\underline{x}$)
	CMP	B	1	compare d_1 ($\underline{x}$), d_2 ($\underline{x}$)
	JC	ALPHA	3	
	MOV	B, A	1	replace d_1 ($\underline{x}$), 1
	MOV	C, E	1	with d_2 ($\underline{x}$), 2
ALPHA:	INR	E	1	
	IN	D3	2	
	⋮			
LAST:	MOV	A,C	1	
	OUT	k	2	Output classification

Table 2.2.8(V) presents the comparison of current and projected program execution times for each processing section for n and M equal to 4 and 12 respectively. The 1974 column and 1983 column result from data taken from the median curve of Figure 3.2.2(1) for projected microprocessor add times.

Table 2.2.8(V) Comparison of Current and Projected Program Execution Times for the Processing System of Figure 2.2.8(1)

	Number of Microprocessors	8080 Microprocessor 3.5 μs ADD	1974 Microprocessor 1.0 μs ADD	1983 Microprocessor 100 ns ADD
PE1	1	678.0 μs	193.7 μs	19.37 μs
PE2	1	4872.0	1392.0	139.20
PE3	1	1290.0	368.6	36.86
PE4	1	231.5	66.1	6.61

The bottleneck in the pipeline is PE2. PE2 can be split into M parallel processing elements as in Figure 2.2.8(3). Each $\underline{x}-\underline{m}_k$ (k = 1, ..., M) is input to a separate $PE2_k$ and the computation for each category proceeds in

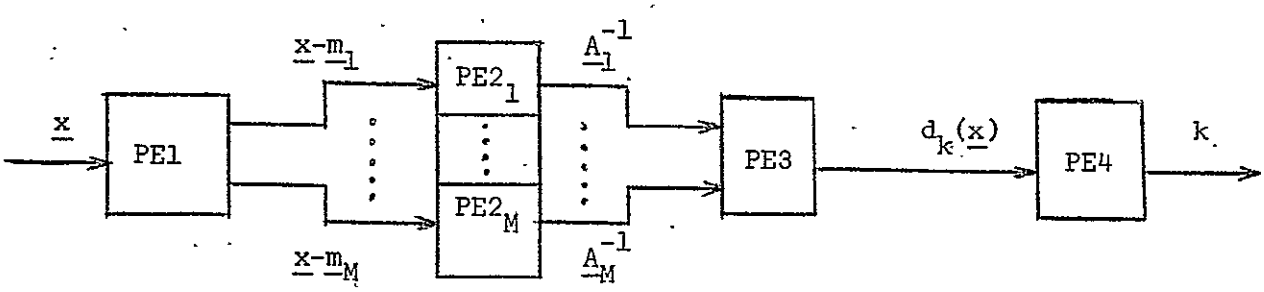


Figure 2.2.8(3) Organization of a Processing System for Executing the Maximum Likelihood Algorithm.

parallel. Table 2.2.8(VI) gives the execution times for M=12.

Table 2.2.8(VI) Comparison of Current and Projected Program Execution Times for the Processing System of Figure 2.2.8(3)

	Number of Microprocessors	8080 Microprocessor 3.5 μs ADD	1974 Microprocessor 1.0 μs ADD	1983 Microprocessor 100 ns ADD
PE1	1	678.0 μs	193.7 μs	19.37 μs
PE2	12	406.0	116.0	11.60
PE3	1	1290.0	368.6	36.86
PE4	1	231.5	66.1	6.61

The bottleneck is now PE3. Suppose that PE3 is organized as 2 processing elements in parallel. Each processing element handles only 6 of the 12 categories. The program execution time in PE3 is halved, requiring 645.0 μs in the INTEL 8080 microprocessor. Now PE1 is the constraining section of the pipeline. Proceeding in this manner we arrive at the processor shown in Figure 2.2.8(4). The resulting number of processors and their execution times for each of the processing elements are summarized in Table 2.2.8(VII).

The maximum likelihood processor of Figure 2.2.8(4) requires 5,814 Intel 8080 microprocessors for implementation with a projection of 171 microprocessors in 1983. It also requires the sixteen 16K multiplication ROMs required in PE2 and PE3. Another method for implementing the multiplication is with a peripheral, hardware multiplier as illustrated in Figure 2.2.8(5). The microprocessor outputs the multiplicand and the multiplier to output registers MCAND and MLIER, respectively, and the multiplier generates the product (PROD) which is input to the microprocessor. The resulting program execution times were determined and found to be

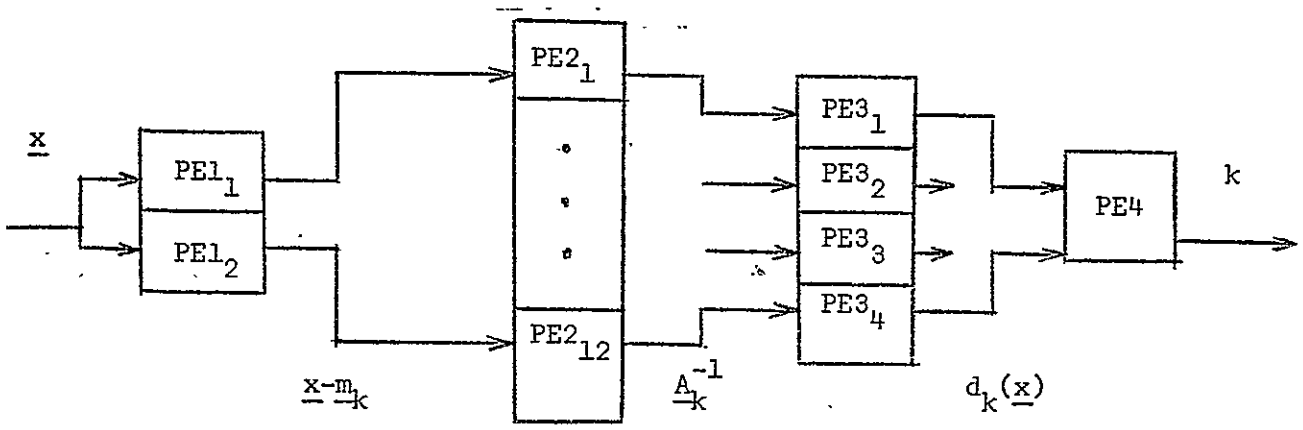


Figure 2.2.8(4) A More Efficient Organization for Executing the Maximum Likelihood Algorithm.

Table 2.2.8(VII) Comparison of Current and Projected Program Execution Times for the Processing System of Figure 2.2.8(4)

	Number of Microprocessors	8080 Microprocessor 3.5 μs ADD	1974 Microprocessor 1.0 μs ADD	1983 Microprocessor 100 ns ADD
PE1	2	354.0 μs	101.1 μs	10.11 μs
PE2	12	406.0	116.0	11.60
PE3	4	322.5	92.1	9.21
PE4	1	231.5	66.1	6.61

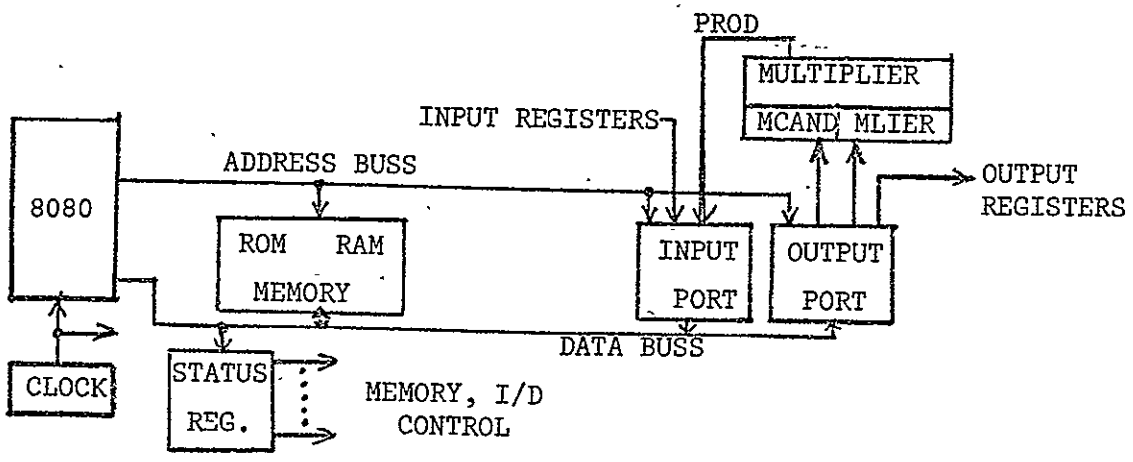


Figure 2.2.8(5) Microprocessor System Organized About an INTEL 8080 Microprocessor and a Hardware Multiplier.

$$PE2: \quad T = 22.5n^2 + 16n + 2 \ \mu s$$

$$PE3: \quad T = (29.5n + 10.5)\frac{M}{4} \ \mu s.$$

The program execution times for PE2 and PE3 when organized with a ROM and a hardware multiplier are listed below. These differences are primarily due to the faster execution times of the instructions for addressing memory as compared to that of the INPUT/OUTPUT instructions.

PROCESSING ELEMENT	ROM MULTIPLIER	HARDWARE MULTIPLIER
PE2	406.0	426.0
PE3	322.5	385.5

Use of the hardware multiplier affects the previous results very little. However, since PE2 was the critical section in the pipeline organization, the increased execution time in PE2 from 406 μs to 426 μs will cause the number of microprocessors to increase. This is reflected in the following comparison.

	ROM MULTIPLIER		HARDWARE MULTIPLIER	
	no. multiplexed systems	total no. microprocessors	no. multiplexed systems	total no. microprocessors
INTEL 8080	306	5,814	321	6,099
1974 Microprocessor	88	1,672	92	1,748
1983 Microprocessor	9	171	10	190

As long as the multiplication time is much less than the execution time of the INPUT instruction, these results are valid.

2.2.8.2 Microprocessor Organization Using Table Look-Up (TLU)

Most of the processing in the TLU algorithm involves calculating the new address to use as a pointer to recover the boundaries for a particular dimension. Little processing is required to compare the data with the boundaries. If no microprogramming feature is available, the single microprocessor approach is inviting. A typical search loop for the TLU implementation with the Intel 8080 is:

Instructions	Execution Cycles	
INB		
CMP M	2	
JP OUT	3	/Jump out of Loop
INXH	1	

CMPM	2	
JM OUT	3	/Jump out of Loop
INXH	1	
ADD M	2	
MOV E,A	1	
INX H	1	
ADC M	2	
MOV L,E	1	
MOV H, A	1	
JMP START	3	
	23	

In the worst case 90% of the processing time is spent in this comparison loop. Since the processor spends 46 μs in this loop and, in the worst case must execute the loop 48 times (n=4, M=12) for each pixel, the maximum pixel rate is about 500 pixel/second. To achieve the baseline format data rate of 1.3×10^6 pixels/second, the processing load must be distributed over 2600 identical systems.

If no microprogramming feature is available, the pointer calculation must be done using the last eight steps in the programming. The overhead due to this processing amounts to 12 machine cycles or about 25 microseconds per pointer. For the n=4, M=12 machine a data throughput of 10,000 pixels/second is achieved using 36 microprocessors and 300 K bytes of memory. The same data throughput may be achieved by paralleling about 20 single microprocessor units with about the same total memory requirements. The fully parallel system is faster but requires more memory to handle the worst case boundary size.

2.2.8.2.1 On-Board Computer Organization for the Table Look-Up Pattern Classification

In this section, we present an on-board computer organization for the 4-dimensional table look-up algorithm described in Section 1.1.3.3. These include the general specifications, memory planning for the table entries, and the multi-microprocessor system design.

We consider n=4 channels per scan line so that each pixel is a 4-dimensional vector $\underline{x} = (x_1, x_2, x_3, x_4)'$, each element requiring 6 bits. If we include the sign bit and a parity-check bit, we allow one 8-bit byte per measurement $\underline{x}_i$. Thus, the computer can be either an 8 bit or a 16 bit machine. We consider M=12 classes and a data rate of 1 MHz. There are six multispectral scan lines. We present the computer architecture per scan line; the processors for all scan lines are identical.

Eppler's dynamic memory assignment technique discussed in Section 1.1.3.3 requires a memory of roughly 32K 16 bit memory words.

Figure 1.1.3(1) suggests that the region in measurement space associated with a particular class can be one of two types. Class 3 is completely separate from all other classes and shares a boundary with only the Threshold Class; this type of class will be referred to as disjoint. The regions associated with Classes 1 and 2 share a common boundary. If Class 1 were not present, the boundary for Class 2 would be extended into the region now assigned to Class 1 and vice versa. Classes 1 and 2 will be referred to as overlapping.

Boundary information is stored in main memory for only those measurement space points actually assigned to the class. For this reason overlapping classes require less storage than disjoint classes having the same statistics. Therefore the upper-limit on the memory requirement can be derived by considering each class separately (i.e., the disjoint case). It is shown in [25] that $N_1(C)$, the number of values x_1 can have and still be inside the measurement space region assigned to Class C, is given by

$$N_1(C) = 2\, Q^{1/2}\, |K_1(C)|^{1/2} \qquad (2.2.8\text{-}1)$$

In this equation Q is a user-specified threshold parameter indicating the maximum Mahalanobis distance from the mean that a measurement vector can have and still be assigned to that class. For a four-dimensional classifier using Q=12.0 excludes fewer than 2 percent of samples taken from a Normal distribution and $K_1(C)$ is the one-dimensional covariance matrix (involving terms for only the first channel) for Class C. The value $N_1(C)$ represents the length of the line which results from projecting Region C onto the x_1-axis as in Figure 1.1.3(1). Similarly $N_2(C)$, the number of (x_1,x_2)-combinations inside the region assigned to Class C, is given by

$$N_2(C) = \pi Q |K_2(C)|^{1/2} \qquad (2.2.8\text{-}2)$$

In this equation $K_2(C)$ is that part of the covariance matrix for Class C which involves x_1 and x_2 terms. The value $N_2(C)$ represents the area which results from projecting Region C onto the (x_1,x_2)-plane as in Figure 1.1.3 (1). Similarly $N_3(C)$, the number of (x_1,x_2,x_3)-combinations inside the region assigned to Class C, is given by

$$N_3(C) = \frac{4\pi}{3}\, Q^{3/2}\, |K_3(C)|^{1/2} \qquad (2.2.8\text{-}3)$$

In this equation $K_3(C)$ is that part of the covariance matrix for Class C which involves the (x_1,x_2,x_3)-subspace. Finally $N_4(C)$, the number of (x_1,x_2,x_3,x_4)-combinations inside the region assigned to Class C, is given by

$$N_4(C) = \frac{\pi^2}{2} Q^2 \left|K_4(C)\right|^{1/2} \qquad (2.2.8\text{-}4)$$

In this equation $K_4(C)$ is the full four-dimensional covariance matrix and $N_4(C)$ represents the volume inside the four-dimensional ellipsoid for a disjoint class.

Equations (2.2.8-1) through (2.2.8-4) give the n-dimensional volume for each class separately. It is useful to define the total and the average n-dimensional volume for an application according to Eqs. (2.2.8-5) and (2.2.8-6), respectively. (Notations were defined in Section 1.1.3.3).

$$\tilde{N}_n = \sum_{c=1}^{N_c} N_n(C) \qquad (2.2.8\text{-}5)$$

$$\overline{N}_n = \frac{\tilde{N}_n}{Nc} \qquad (2.2.8\text{-}6)$$

The quantity $\tilde{N}_n = N_c\overline{N}_n$ is very important in that it is the number of different values pointer P_n can have; see Figure 1.1.3(3). For each possible value of P_1 main memory must be provided to store L_2, H_2, and O_2. The quantities L_2 and H_2 can each be stored in one 8-bit byte, but O_2 requires 2 bytes because it can exceed 255. As shown in Figure 1.1.3(3), the total memory required to store all L_2, H_2, and O_2 values for a complete application is $4\tilde{N}_1 = 4N_c\overline{N}_1$ bytes. Similarly $4\tilde{N}_2 = 4N_c\overline{N}_2$ bytes are required to store all values of L_3, H_3, and O_3. The memory required to store all values of L_4 and H_4 is $2\tilde{N}_3 = 2N_c\overline{N}_3$. It can be seen from Figure 1.1.3(5) that $4N_c$ bytes of main memory must be provided to store all values of L_1, H_1, and O_1. The total memory required to store all of the boundary information for a complete four-dimensional Table Look-Up classification is given by

$$N_B = 4N_c + 4\tilde{N}_1 + 4\tilde{N}_2 + 2\tilde{N}_3 = 4N_c(1 + N_1 + \overline{N}_2 + 0.5\overline{N}_3) \qquad (2.2.8\text{-}7)$$

The main memory requirement was calculated using statistics from a study involving a wide variety of sensors, platforms, and applications. Flight Line C1 [28] is a 6.4 km by 1.6 km agricultural test site near Purdue University; data was collected on June 28, 1966, using the Michigan multispectral scanner at an altitude of 800 meters.

An upper-limit for main memory requirement (i.e., assuming disjoint classes) was calculated using Eqs. (2.2.8-1) through (2.2.8-7) and the covariance matrix for each of the classes. Table 2.2.8(VIII) gives the

Table 2.2.8(VIII) Computed Upper-Limit Estimates for Main Storage and Table-Making Calculations for the Case Q=12

	Study Class	Optimum Search Sequence	N_1	N_2	N_3	N_4	N_B Storage Required (Bytes)	Number of Points Tested to Computer Tables
LARS Flight Line C1	Soybean	1,6,9,11	18	230	2,530	46,300	6,530	46,300
	Corn	6,9,1.12	17	193	2,090	32,800	5,020	32,800
	Oats	1,6,9,11	21	242	4,600	90,500	10,252	90,500
	Wheat I	6,8,9,12	20	284	2,160	25,100	5,560	25,100
	Red Clover	1,9,6,12	15	158	1,740	43,200	4,172	43,200
	Alfalfa	1,6,9,11	13	126	1,300	37,900	3,156	37,900
	Rye	1,6,9,12	16	163	1,670	15,700	4,056	15,700
	Bare Soil	9,12,1,11	11	93	760	8,150	1,936	8,150
	Wheat II	1,6,9,11	19	614	11,950	232,000	26,432	232,000
	Total: $\tilde{N}$	---	156	2,103	28,800	531,650	66,636	531,850
	Average: $\bar{N}$	---	17	233	3,200	59,072	7,404	59,094

search sequence which minimizes the memory requirement and N_B, the number of 8-bit bytes of main memory required to store the necessary tables. This table shows that to store the complete representation of the classification regions of the measurement space requires at most 66,636 bytes in the case of Flight Line C1. For this case 531,850 different (x_1,x_2,x_3,x_4)-combinations must be tested to compute the prestored tables.

In addition to the disjoint case, in which each class is treated separately, computations were performed to determine the actual memory requirement taking into account the overlap between classes. This was accomplished by finding the various volumes using the maximum likelihood classifier for assumed normal statistics as in [28]. Table 2.2.8(IX) shows that the actual core requirement is 92 percent of the worst-case requirement (see Table 2.2.8(VIII)) for the case of Flight Line C1.

The results presented in Table 2.2.8(IX) support the following important conclusions:

(1) The main memory requirements for a four-channel Table Look-Up classifier are easily satisfied. Flight Line C1 can be run on a minicomputer with 32K 16-bit words or 64K 8-bit words.

Table 2.2.8(IX) Actual Requirements for Main Storage and Table-Making Calculations for the Case Q=12

Study	Class	Optimum Search Sequence	N_1	N_2	N_3	N_4	N_B Storage Required (Bytes)	Number of Points Tested to Computer Tables
LARS Flight Line C1	Soybean	1,6,9,11	18	236	2,246	40,096	5,506	46,300
	Corn	6,9,1,12	17	178	1,895	28,569	4,570	32,800
	Oats	1,6,9,11	21	251	4,172	78,290	9,432	90,500
	Wheat I	6,8,9,12	26	302	2,136	24,074	5,584	25,100
	Red Clover	1,9,6,12	15	154	1,670	37,934	4,012	43,200
	Alfalfa	1,6,9,11	13	129	1,153	32,243	2,874	37,900
	Rye	1,6,9,12	16	161	1,509	13,805	3,718	15,700
	Bare Soil	9,12,1,11	11	90	762	8,150	1,924	8,150
	Wheat II	1,6,9,11	19	587	10,720	202,556	23,860	232,000
	Total: $\hat{N}$	---	156	2,088	26,263	465,717	61,480	531,850
	Average: $\bar{N}$	---	17	232	2,918	51,746	6,831	59,094

(2) Use of the Table Look-Up algorithm in five dimensional measurement space is probably not practical. For five dimensions the memory requirements are 1,045,440 bytes for Flight Line C1. If more than four channels are required, it is more practical to compute the four best linear combinations of channels and use the result in the four-dimensional Table Look-Up algorithm.

(3) Using Eqs. (2.2.8-1) through (2.2.8-7) gives moderately good estimates of the main RAM memory requirements without having to actually carry out the maximum likelihood computations.

2.2.8.2.2 Multi-Microprocessor Computer System for On-Board Table Look-Up Pattern Classification

For n = 4 channels and M = 12 pattern classes, we need a multiprocessor which allows parallel retrieving and comparing boundary information as well as concurrent address pointer calculations. A modified version of the table look-up algorithm emphasizing the inherent parallelism was developed; the required multiprocessor/multi-memory module computer system per each class is presented in Figure 2.2.8(7). The lower boundary information $L_j(P_i)$, the upper boundary information $H_j(P_i)$, and the pointers $O_j(P_i)$,

are stored in memory module M_1, M_2, and M_3, respectively. At each memory cycle, these are simultaneously fetched. The processors are assigned special missions; i.e., P_1 performs the comparison $L_j(P_i) \leq x_i$; P_2 performs the comparison $x_j \leq H_j(P_i)$ and P_3 is responsible for pointer updating $P_j = O_j(P_i) + x_j$. Each measurement x_i in the pixel $\underline{x}$ may be input to all three processors at the same time so that concurrent executions are enhanced.

Since each processor is required to perform only simple computations and logical decisions, we use microprocessor approach with microprogrammed control. Table 2.2.8(X) shows the system characteristics in the Intel 8080 microprocessors.

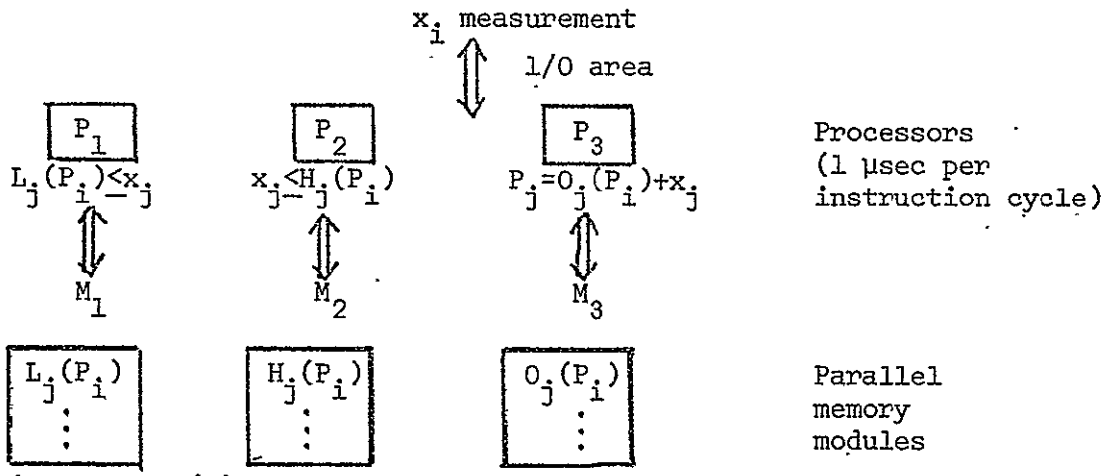


Figure 2.2.8(7) A 3-Processor and 3-Memory Module Subsystem for Table Look-Up Classification (Per Class).

Table 2.2.8(X) Major System Characteristics of the Multiprocessor System for 4-Channel Table Look-Up

# of μ-processors used	36 (Intel 8080 n-MOS)
# of 4Kx1 RAM chips required	$\frac{150K \times 8}{4K \times 1} = 300$
memory cycle	450 n sec (n-MOS RAM's)
processing speed per each pixel	4 μsecs or 250 KC data rate (1 μsec per addition or comparison)
# of registers or buffers required	36 16-bit address buffers; 72 8-bit data buffers; (need 2 8-bit buffers, 1 16-bit buffer per μ-processor)
Power required	36 watts + 100 watts = 136 watts (1 watt/processor and 0.5 watts/4Kx1 RAM)
Total # of IC chips	36 + 300 + 108 = 450

2.2.8.3 Microprocessor Organization for the Expanded Maximum Likelihood Algorithm

By expanding Eq. (1.1.3-5) and merging coefficients we generate the following expression for the maximum likelihood decision function:

$$d_k(\underline{x}) = A_{k1}x_1^2 + A_{k2}x_2^2 + A_{k3}x_3^2 + A_{k4}x_4^2 + A_{k5}x_1x_2 + A_{k6}x_1x_3 + A_{k7}x_1x_4 + A_{k8}x_2x_3 + A_{k9}x_2x_4 + A_{k,10}x_3x_4 + A_{k,11}x_1 + A_{k,12}x_2 + A_{k,13}x_3 + A_{k,14}x_4 + A_{k,15} \quad (2.2.8\text{-}8)$$

Figure 2.2.8(8) depicts a system architecture for implementing Eq. (2.2.8-8) in which each PE is organized about a microprocessor. PE3 is the same as PE4 in the previous microprocessor system organizations. PE1 and PE2 are organized for efficient execution of Eq. (2.2.8-8) rather than the matrix format presented in Eq. (1.1.3-5). $PE1_k$ of PE1 generates the output P_k which consists of the partial products formed from the pattern vector input $\underline{x}$, and a partial sum.

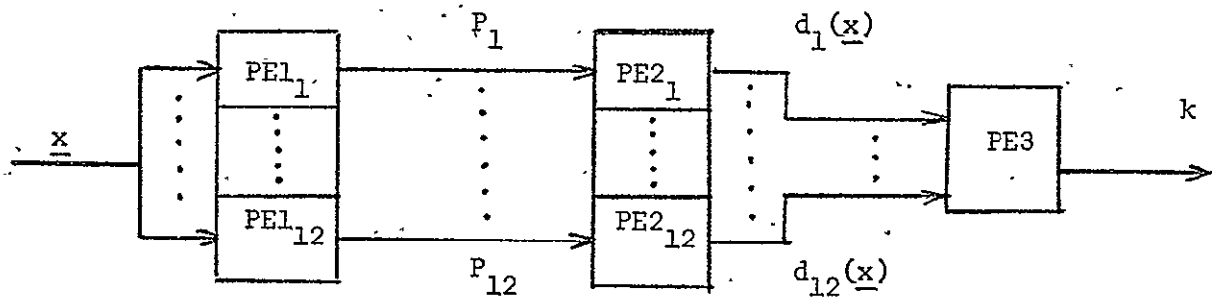


Figure 2.2.8(8) A Microprocessor System Organized Similarly to the Optimal Pipelined Arithmetic Unit

$$P_k = x_1^2, x_2^2, x_3^2, x_4^2, x_1 \cdot x_2, x_1 \cdot x_3, x_1 \cdot x_4, x_2 \cdot x_3, x_2 \cdot x_4, x_3 \cdot x_4, \text{PARTIAL}_k$$

where

$$\text{PARTIAL}_k = A_{k,11}x_1 + A_{k,12}x_2 + A_{k,13}x_3 + A_{k,14}x_4 + A_{k,15}$$

for k = 1, 2, ..., 12.

P_k is input to $PE2_k$ of PE2 which forms $d_k(\underline{x})$. $d_k(\underline{x})$ is input to PE3 (k = 1, ..., 12) for classification.

Table 2.2.8(XI) presents the program for PE2. The program for PE1 is listed in Progress Report No. 8. Each PE in PE1 and PE2 has a peripheral

Table 2.2.8(XI) Program for Generating $d_k(\underline{x})$ in PE2

MNEMONIC	OPERAND	BYTES	COMMENT
LXI	H,COEFFA	3	initialize coefficient location
IN	PARTIAL	2	
MOV	TEMP,A	1	initialize TEMP with partial sum
IN	X1SQ	2	input x_1^2
OUT	MCAND	2	
MOV	A,COEFFA	1	input $A_{k,1}$
OUT	MLIER	2	
IN	PROD	2	input product $A_{k,1} \cdot x_1^2$
ADD	TEMP	1	accumulate sum
MOV	TEMP,A	1	
INX	H	1	
⋮		96	multiply each $A_{k,j}$ by appropriate partial product, x_2^2, x_3^2, x_4^2, $x_1 \cdot x_2$, $x_1 \cdot x_3$, $x_1 \cdot x_4$, $x_2 \cdot x_3$, $x_2 \cdot x_4$, for $j = 2, \ldots, 9$
IN	X3X4	2	input product $x_3 \cdot x_4$
OUT	MCAND	2	
MOV	A,COEFFA	1	input $A_{k,10}$
OUT	MLIER	2	
IN	PROD	2	input product $A_{k,10} \cdot x_3 \cdot x_4$
ADD	TEMP	1	accumulate final sum
OUT	Dk	2	output $d_k(\underline{x})$

hardware multiplier. Table 2.2.8(XII) gives the comparison of current and projected execution times for each processing section in Figure 2.2.8(8).

Table 2.2.8(XII) Comparison of Current and Projected Execution Times for the Processing System of Figure 2.2.8(8)

	Number of Microprocessors	8080 Microprocessor 3.5μs ADD	1974 Microprocessor 1.0μs ADD	1983 Microproce... 100 ns ADD
PE1	12	338.0	96.57	9.66
PE2	12	342.5	97.86	9.79
PE3	1	231.5	66.1	6.61

In Table 2.2.8(XIII) the number of multiplexed, parallel systems and the total number of microprocessors required to process data at the acquisition rate of 1.33 μs per pixel for the baseline data format defined in Section 1.4 is summarized.

Table 2.2.8(XIII) Comparison of the Number of Microprocessors Required for the Processing System of Figure 2.2.8(8)

	Number of Multiplexed Parallel Systems	Total Number of Microprocessors
INTEL 8080	258	6,450 (258 x 25)
1974 Microprocessor	74	1,850
1983 Microprocessor	8	200

Each $PE1_k$ duplicates the generation of the partial products formed from the input vector. These products are not functions of the classification categories and need be generated only once. Figure 2.2.8(9) presents a much more efficient organization. PE1A generates the partial products P

$$P = x_1^2, x_2^2, \ldots, x_2 \cdot x_4, x_3 \cdot x_4.$$

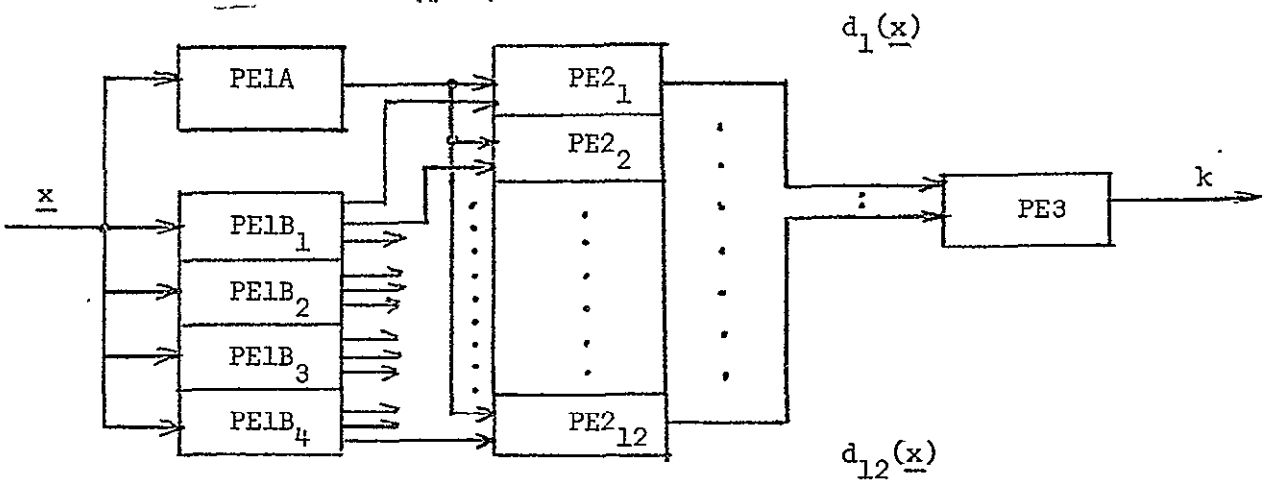


Figure 2.2.8(9) A More Efficient Microprocessor Organization For Implementing Eq.(2.2.8-8)

in parallel, PE1B generates the partial sum, $PARTIAL_k$, k = 1, ..., 12. PE1B consists of four (4) parallel microprocessor processing elements. Each processing element generates three (3) partial sums. For instance, PE1B generates

$$PARTIAL_1 = A_{1,11}x_1 + \ldots\ldots\ldots + A_{1,15}$$

$$PARTIAL_2 = A_{2,11}x_1 + \ldots\ldots\ldots + A_{2,15}$$

$$PARTIAL_3 = A_{3,11}x_1 + \ldots\ldots\ldots + A_{3,15}$$

PE2 and PE3 are the same as in the Figure 2.2.8(8) organization.

The program can be split between PE1A and PE1B. However, each processing element in PE1B must execute the segment of code related to the partial sums three times, one for each of the classification categories. Table 2.2.8(XIV) and Table 2.2.8(XV) summarize the results for this system architecture. Notice that this organization requires the fewest number of microprocessors, but not a spectacular decrease. Eq. (2.2.8-8) requires twenty-four (24) multiplications while the matrix form

$$(\underline{x}-\underline{m}_k)'\underline{C}_k^{-1}(\underline{x}-\underline{m}_k)$$

requires only twenty (20) for n = 4. The efficient use of PE1A provided the slight improvement.

Table 2.2.8(XIV) Comparison of Current and Projected Execution Times for the Processing System of Figure 2.2.8(9)

	Number of Microprocessors	8080 Microprocessor 3.5μs ADD	1974 Microprocessor 1.0μs ADD	1983 Microprocessor 100 ns ADD
PE1A	1	200	57.1	5.71
PE1B	4	384	109.7	10.97
PE2	12	342	97.7	9.77
PE3	1	231.5	66.1	6.61

Table 2.2.8(XV) Comparison of the Number of Microprocessors Required For the Processing System of Figure 2.2.8(9)

	Number of Multiplexed Parallel System	Total Number of Microprocessors
INTEL 8080	289	5,202
1974 Microprocessor	83	1,494
1983 Microprocessor	8	162

A summary of all the microprocessor organizations is provided in Table 2.2.8(XVI). The effort to identify the hardware requirements for these various architectures culminated in Tables 2.2.8(XVII) through 2.2.8(XXI). Several considerations were taken into account and are:

1. In calculating the required RAM memory each processing element in a particular processing section has the same amount of RAM. However, the program store was assumed to be in only one of the

Table 2.2.8(XVI) Comparison of Number of Microprocessors for Different Function and Multiplication Implementations

	ROM Multiplier System of Figure 2.2.8(5)		Hardware Multiplier System of Figure 2.2.8(5) Using PE in Fig. 2.2.8(6)	
	no. multiplexed systems	total no. microprocessors	no. multiplexed systems	total no. microprocessors
INTEL 8080	306	5,814	321	6,099
1974-Microprocessor	88	1,672	92	1,748
1983-Microprocessor	9	171	10	190

(a) Eq. (1.1.3-5) Implementation

	Hardware Multiplier System of Figure 2.2.8(8)		Hardware Multiplier System of Figure 2.2.8(9)	
	no. multiplexed systems	total no. microprocessors	no. multiplexed systems	total no. microprocessors
INTEL 8080	258	6,450	289	5,202
1974-Microprocessor	74	1,850	83	1,494
1983-Microprocessor	8	200	9	162

(b) Eq. (2.2.8-8) Implementation

processing elements. Since each processing element is working in synchronization in a processing section, the instruction to be executed is output to all by the master element.

2. Different types of multiplication schemes were utilized to access the impact on system program execution times. ROM multipliers, PLA multipliers, and hardware multipliers were considered.
3. For calculating power, a power standard was developed by identifying the major components in the system and associating a typical power dissipation with each. In future projections, the power dissipation was assumed to remain constant. For instance, although the projection gave fewer and assumed more powerful microprocessors, the power dissipation per μP chip remained 1.0 watt. The same was true for RAMs, ROMs, PLAs hardware multipliers, and I/O registers. A power standard is provided in the bottom row of Table 2.2.8(XVII).

Table 2.2.8(XVII) System Hardware Requirements for the Microprocessor Processing System Using ROM Multipliers

	Total Number Micro-processors	Required Memory (Bytes)	ROM Multiplier Memory (K Bytes)	Number of I/O Registers	*Power (Kwatts)	Weight (Kgrams)	Volume (Cubic Meters)	Cost (K$)
INTEL 8080	5,814	78,697	78,336	97,002	16,286	978	1.96	6,300
1974 Micro-processor	1,672	22,889	22,528	27,896	4.684	281	0.57	1,870
1983 Micro-processor	171	2,665	2,304	2,853	0.479	29	0.06	186
Power Standard	INTEL 8080 1.0 watt	INTEL 8102 RAM 1024 words x 1 bit 150 mwatt	INTEL 8316 ROM 2048 words x 8 bits 175 mwatt	2-TI SN74L95 38 mwatt				

* based on Low-power TTL MSI and Silicon Gate n-channel MOS technology. Power Standard is Provided as bottom line of table.

Table 2.2.8(XVIII) Same as Table 2.2.8(XVII) Except the ROM Multipliers Are Replaced by PLA Multipliers

	Total Number Micro-processors	Required Memory (Bytes)	Number of PLA Multipliers**	Number of I/O Registers	*Power (Kwatts)	Weight (Kgrams)	Volume (Cubic Meters)	Cost (K$)
INTEL 8080	5,814	78,697	4,896	97,002	12.285	845	1.69	5,854
1974 Micro-Processor	1,672	22,889	1,408	27,896	3.533	243	0.49	1,684
1983 Micro-Processor	171	2,665	144	2,853	0.361	25	0.05	172
			DM 7575 550 mwatts					

* Same power standard as Table 2.2.8(XVII)

** National Semiconductor DM 7575/DM 8575 programmable logic array (PLA). A 16K x 8-bit memory would be required to provide equivalent function. Power Dissipation is 550 mwatts.

4. Realistic estimates were made concerning weight, volume, and cost. In each table the number of chips required was calculated. Weight was generated on the basis of 250 integrated circuits/kg and 125 μP circuits/kg. Integrated circuit and circuit board volume was calculated on the basis of 8 cm^3 for integrated circuit chips and

Table 2.2.8(XIX) System Hardware Requirements for the Microprocessor Processing System Using Hardware Multipliers

	Total Number Micro-processors	Required Memory (Bytes)	Number** of Multipliers	Number of I/O Registers	*Power (Kwatts)	Weight (Kgrams)	Volume (Cubic Meters)	Cost (K$)
INTEL 8080	6,099	82,584	5,136	101,757	14.983	1070	2.14	6,911
1974 Micro-processor	1,748	23,960	1,472	29,164	4.204	307	0.62	1,967
1983 Micro-processor	190	2,968	160	3,170	0.467	33	0.07	215

* Power standard same as Table 2.2.8(XVII) except for multipliers

** Iterative array multiplier using 8-2 bit x 4 bit binary parallel multipliers such as TI SN74LS261, F 93S43, or AM 2505. Includes multiplicand and multiplier registers. Power dissipation 958 mwatts.

Table 2.2.8(XX) System Hardware Requirements for the Microprocessor Processing System of Figure 2.2.8(8) Using Hardware Multipliers and Eq. (2.2.8-8) Implementation

	Total Number Micro-processors	Required Memory (Bytes)	Number of Multipliers	Number of I/O Registers	*Power (Kwatts)	Weight (Kgrams)	Volume (Cubic Meters)	Cost (K$)
INTEL 8080	6,450	52,991	6,192	18,834	13.160	451	0.90	3,936
1974 Micro-processor	1,850	15,455	1,776	5,402	3.774	129	0.26	1,129
1983 Micro-processor	200	1,991	192	584	0.408	14	0.03	122

* Power standard same as Table 2.2.8(XVII), Table 2.2.8(XIX).

Table 2.2.8(XXI) System Hardware Requirements for the Microprocessor Processing System of Figure 2.2.8(9) Using Hardware Multipliers and Eq. (2.2.8-8) Implementation

	Total Number Micro-Processors	Required Memory (Bytes)	Number of Multipliers	Number of I/O Registers	*Power (Kwatts)	Weight (Kgrams)	Volume (Cubic Meters)	Cost (K$)
INTEL 8080	5,202	57,098	4,913	22,253	8.817	418	0.84	3,441
1974 Micro-processor	1,494	16,722	1,411	6,391	2.533	120	0.24	987
1983 Micro-processor	162	2,218	153	693	0.275	13	0.03	107

* Power standard same as Table 2.2.8(XVII), Table 2.2.8(XIX).

16 cm^3 for μP chips. Quantity costs were considered to be \$50/μP, \$10/RAM, \$5/ROM, \$4/REGISTER, etc.

2.2.8.4 Hardware Organization for the Expanded Maximum Likelihood Algorithm

In this section we design some processors for computing the maximum likelihoods given by Eq. (1.1.3-5) and the expanded version given by Eq. (2.2.8-8) that are based on hardware implementations as opposed to software implementations using microprocessors. This amounts to using hardware multipliers and adders in an efficient arrangement for performing the required computations. The first two organizations are based on Eq. (1.1.3-5) and the third organization is based on Eq. (2.2.8-8).

2.2.8.4.1 Serial Organization

In a serial computer system, we simply compute each of the 16 terms $g_k(ij)$ of Eq. (1.1.3-5 for $1 \leq 1$, $j \leq 4$ one at a time. The following serial pipelined arithmetic unit (SPAU) is designed to serve the purpose. Two adders and two multipliers are required in this SPAU as shown in Figure 2.2.8(10).

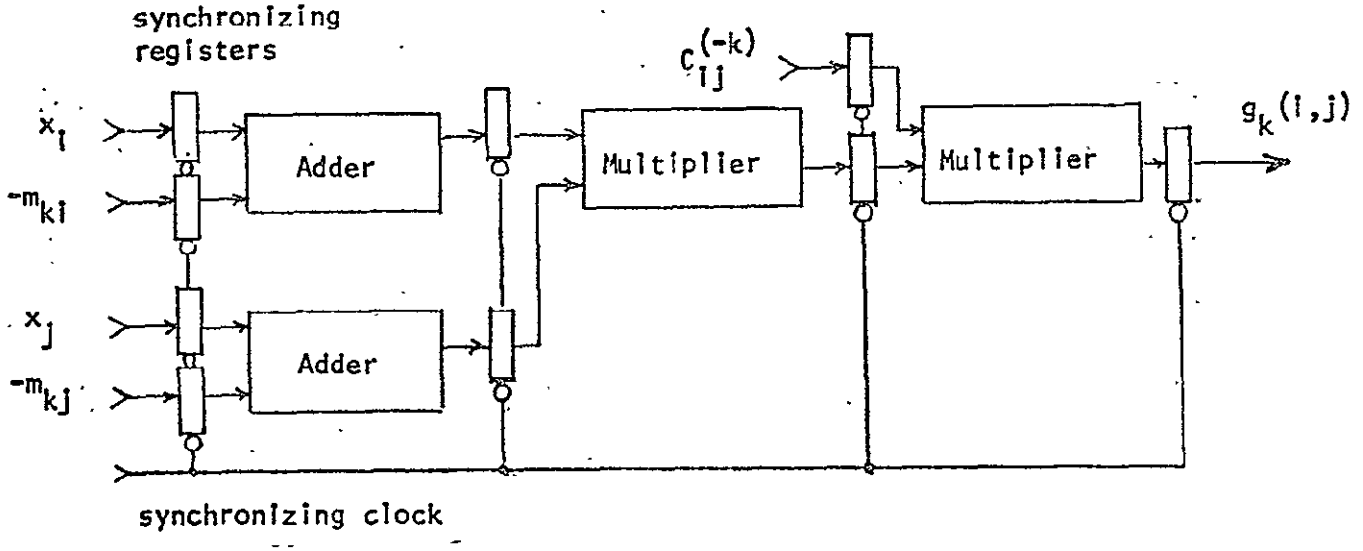


Figure 2.2.8(10) The Serial Pipelined Arithmetic Unit (SPAU)

The whole serial computer, consisting of three processing elements pipelined in cascade is shown in Figure 2.2.8(11). RAM stands for random-access memory and EAPROM means electrically-alterable programmable read-

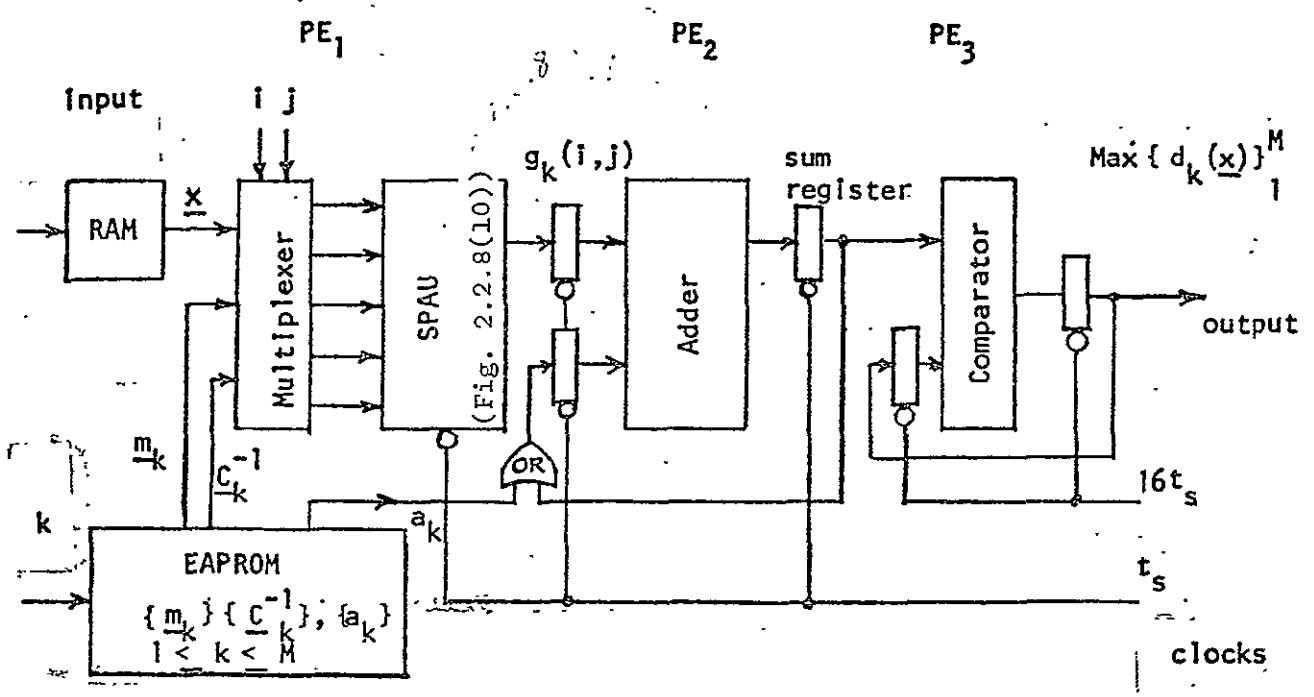


Figure 2.2.8(11) The Serial Computer Organization

Table 2.2.8(XXII) Time and Hardware Requirements for the Serial Computer Organization

No. of Arithmetic Units or μ-processors used	No. of Registers	Memory for Data & Coefficients	Total Processing Time Required
2 adders, 2 multipliers & 1 multiplexer for PE_1; 1 adder of PE_2 and 1 comparator for PE_3	9 for PE_1 2 for PE_2 3 for PE_3	4 8-bit RAM words for $\underline{x}$ 252 8-bit EAPROM words for $\{\underline{m}_k\}_1^{12}$ $\{\underline{C}_k^{-1}\}_1^{12}$ and $\{a_k\}_1^{12}$	Let t_s be the pipeline period. We need 16 t_s per class and $16 \cdot t_s \times 12 = 192\ t_s$ for 12 classes
Total 7 devices	Total 14 registers		

only memory.

The hardware and time requirements are listed above in Table 2.2.8 (XXII).

In Table 2.2.8(XXIII) t_r = time delay of register;
t_a = time delay of adder;
t_m = time delay of multiplier
t_c = time delay of comparator.
The system speed $t_s = t_r + \max(t_a, t_m, t_c)$.

2.2.8.4.2 Parallel Organization

All of the 192 terms in Eq. (1.1.3-5) for $g_k(i,j)$ for $1 \leq i, j \leq 4$ and $1 \leq k \leq 12$ can be calculated in parallel. Also, the summing stage in PE_2 and the comparing stage in PE_3 of Figure 2.2.8(13) can be done in $\log_2(n^2+1) = \log_2 17 = 5$ and $\log_2 M = \log_2 12 = 4$ steps instead of 15 or 11 steps, respectively. A parallel system is shown in Figure 2.2.8(14). Each PE_{1k} for $1 \leq k \leq 12$ in Figure 2.2.8(14) contains $n^2 = 16$ copies of the SPAU shown in Figure 2.2.8(10). All of the PE_{2k} for $1 \leq k = 12$ are identical as shown in Figure 2.2.8(13a). The detailed diagram for PE_3 in Figure 2.2.8(12) is shown in Figure 2.2.8(13b). The memory requirement is the same as in the serial system. The parallel system is about 192 times faster than the serial system unit requires $\frac{971 + 2477}{7 + 14} = 164$ times more hardware cost.

Table 2.2.8(XXIII) Time and Hardware Requirements for the Parallel Computer Organization

No. of computing devices used	No. of registers	Total processing time required
2 x 16 x 12 = 384 adders and 384 multipliers in computing stage 16 x 12 = 192 adders used in summing stage 11 comparators used in comparing stage	9x16x12 = 1728 in computing stage 33x12 = 726 in summing stage 23 in comparing stage	One $t_s = t_r + \text{Max}(t_a, t_m, t_c)$ for 12 classes
Total 971 devices	Total 2477	

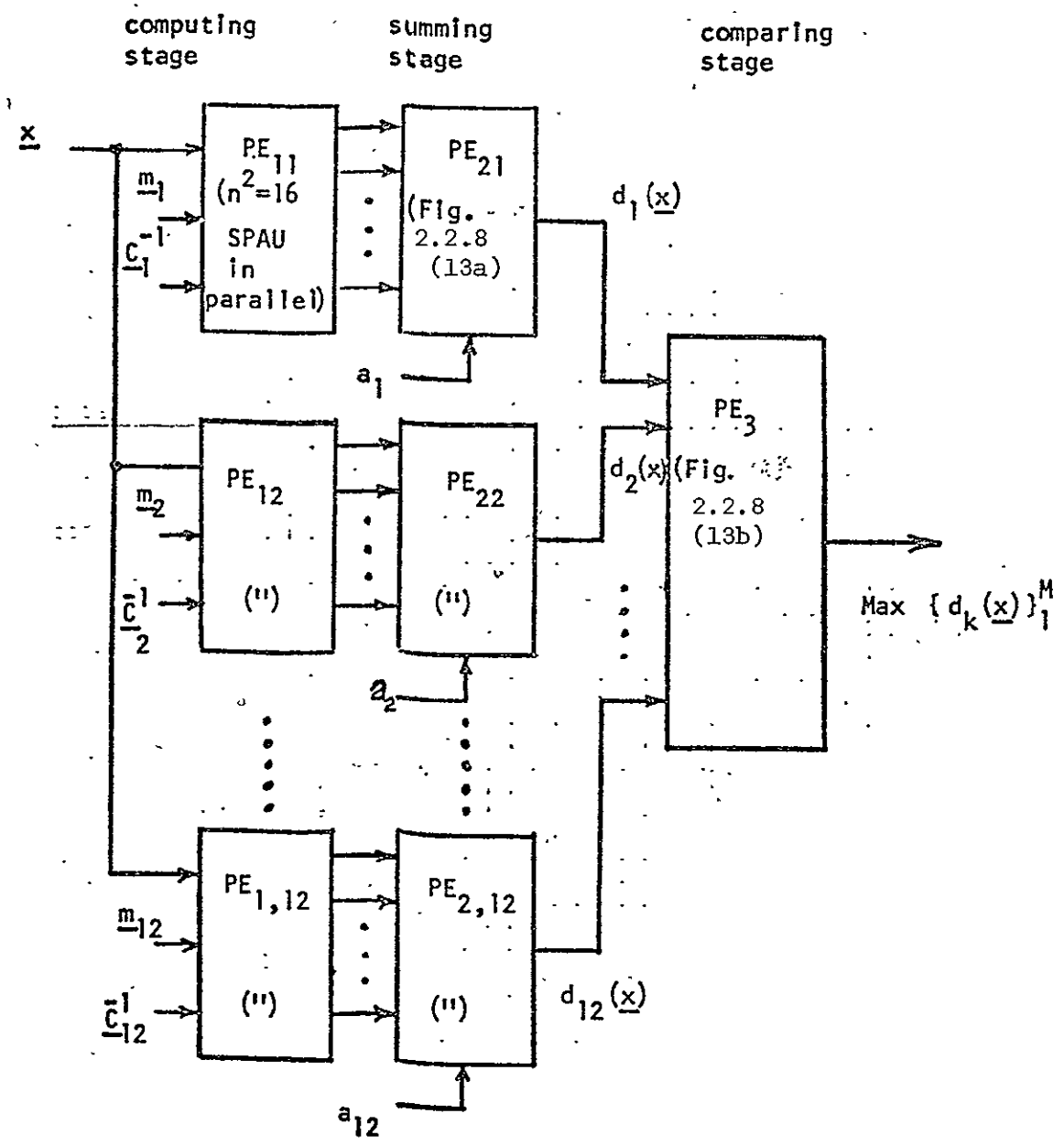


Figure 2.2.8(12) The Parallel Computer Organization

2.2.8.4.3 The Optimal Organization

All of the coefficients A_{ki}'s in Eq. (2.2.8-8) can be computed in advance. This implies a gain in speed of the on-board processor and a reduction in hardware.

In Figure 2.2.8(14) the first pipeline stage PE_1 contains m=12 identical optimal pipelined arithmetic units (OPAU) as in Figure 2.2.8(15). The comparing stage PE_2 is identical with that shown in Figure 2.2.8(13b). In

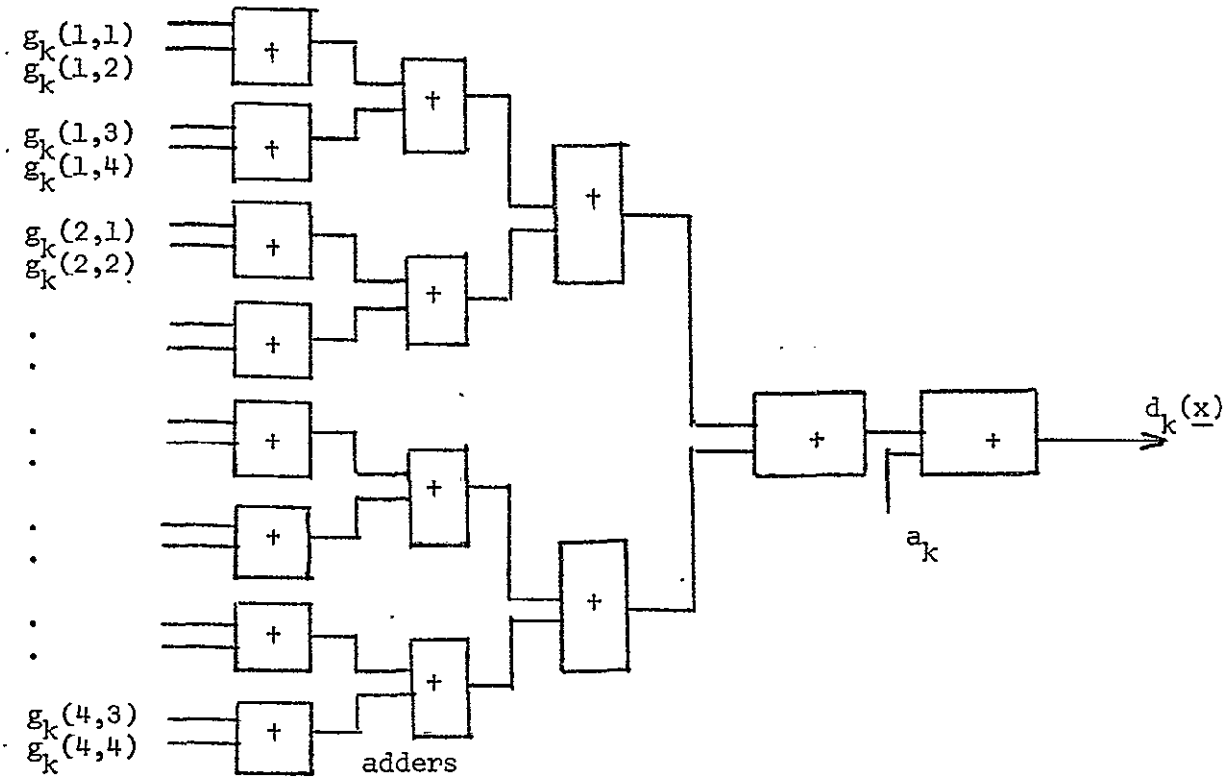


Figure 2.2.8(13a) The Parallel Summing Stage PE_{2k} for $1 \leq k \leq 12$

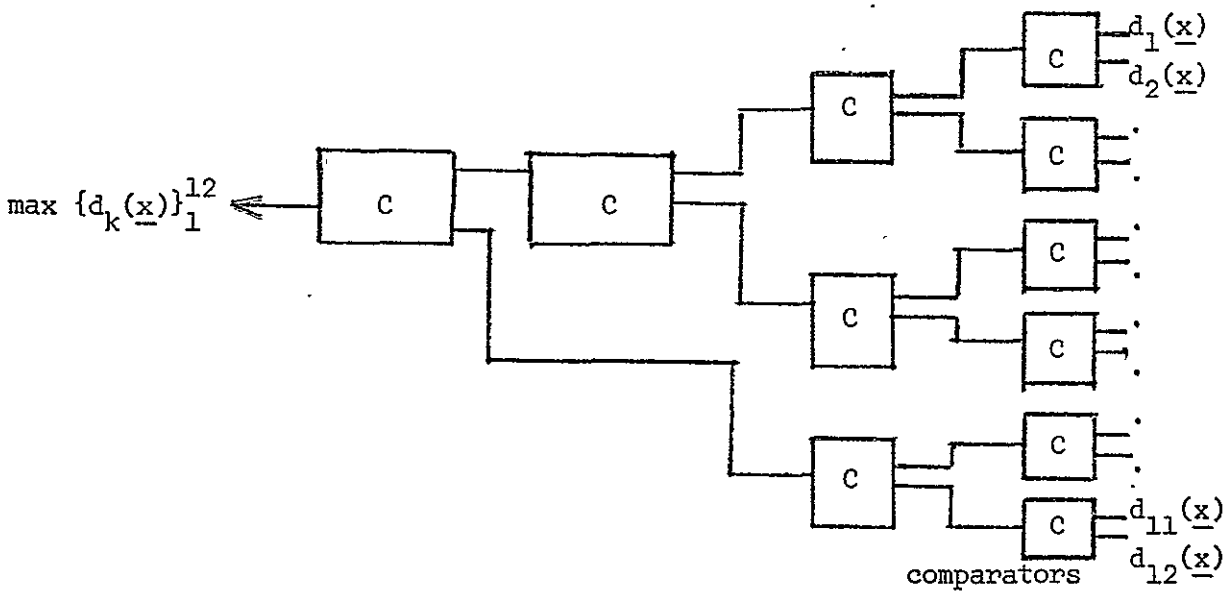


Figure 2.2.8(13b) The Parallel Comparing Stage PE_3

Table 2.2.8(XXIV), we summarize the complexity associated with such an optimal system.

Table 2.2.8(XXIV) Time and Hardware Requirements for the Optimal Computer Organization

No. of Computing devices	No. of Registers used	Memory Requirement for coeff's and data	Total Processing Time req'd.
24x12=288 multipliers 14x12=168 adders 11 comparators Total 467 devices	59 x 12 + 23 = 731 latches	4 8-bit RAM words for $\underline{x}$ 15x12+12=192 8-bit EAPROM words for the coefficients	One unit $t_s = t_{register} + Max(t_{adder}, t_{multiplier}, t_{comparator})$.

Table 2.2.8(XXV) Comparison of the Three Computer Organizations

Organization	Total Computing Devices (α)	Total Processing Time* (β)	Total Memory Requirement	Performance Efficiency Measure ($\alpha \cdot \beta$)
Serial Computer (cheap but slow)	7	192 t_s	256 8-bit words	7 x 192 = 1074
Optimal Computer (medium cost and fast)	467	1 t_s	196 8-bit words	467 x 1 = 467
Parallel Computer (expensive but fast)	971	1 t_s	256 8-bit words	971 x 1 = 971

* $t_s = t_{latches} + Max.\ (t_{adder}, t_{multiplier}, t_{comparator})$

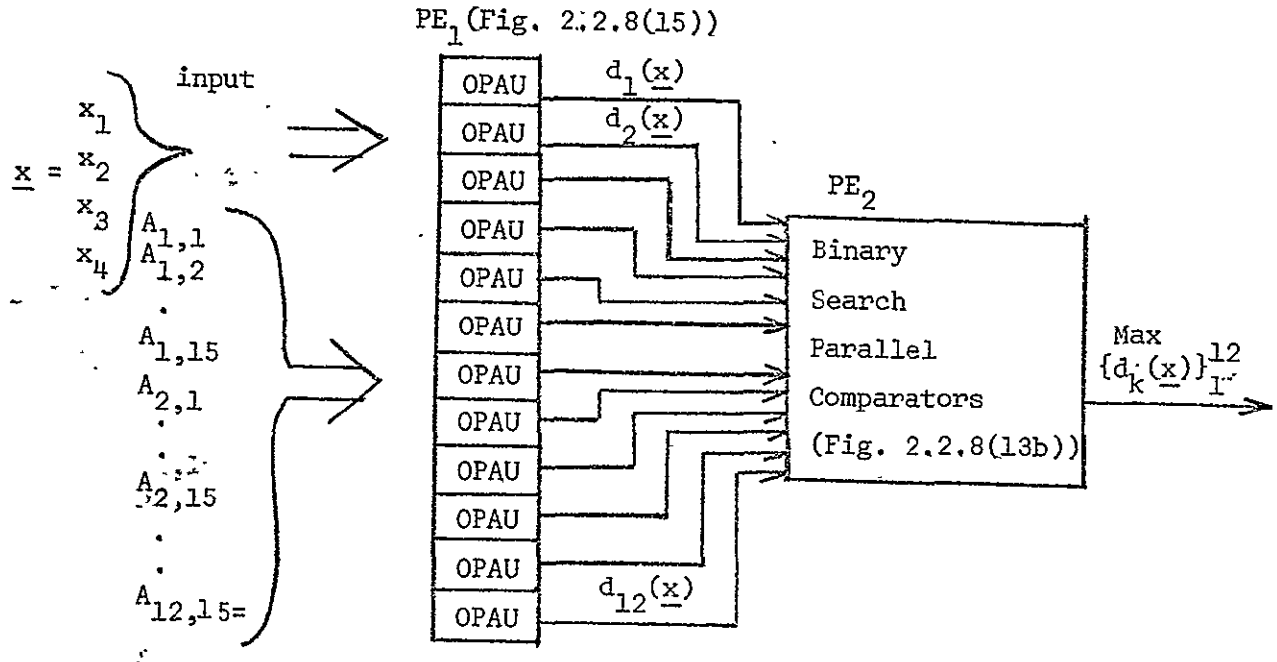


Figure 2.2.8(14) The Optimal Computer Organization

2.2.8.4.4 Comparison of the Three Computer Organizations

In Table 2.2.8(XXV), we summarize the processing time and hardware requirements for the three computer organizations. A performance efficiency measure is defined as the product of the number of time units and the number of hardware devices required according to the current level of technology. From Table 2.2.8(XXV), we conclude with the following remarks:

(1) The serial organization would cost the least in hardware at the expense of intolerable slowness.

(2) The parallel system is very costly because of excessive hardware.

(3) The memory requirement for storing input data and constant coefficients is relatively the same for all the three organizations. The synchronizing registers (latches) required in each organization is roughly proportional to the required device numbers.

(4) The optimal organization offers an improvement over the other two organizations. Its performance measure is roughly 2 times better than the strictly parallel system and 2.3 times better than the serial system.

2.2.8.5 A Hardware Table Look-Up Processor

A hardware implementation of the TLU algorithm using currently available TTL and MOS technologies is shown in Figure 2.2.8(16). The timing

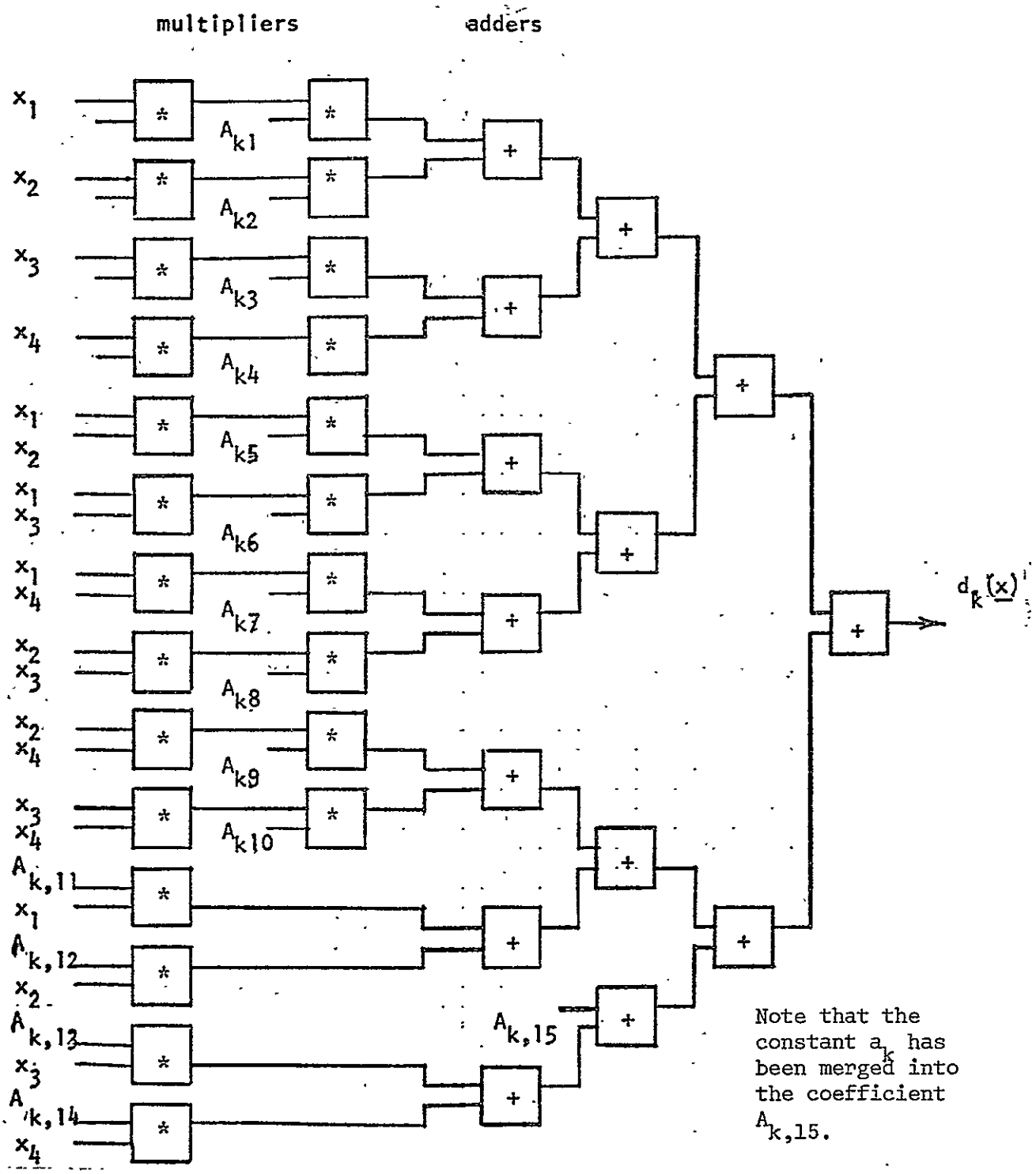


Figure 2.2.8(15) The Optimal Pipelined Arithmetic Unit (OPAU).

generator is shared between parallel units (one for each pattern class) and contributes insignificantly to the overall IC count and power dissipation. Two TTL IC's are required for each of the digital comparators; the total

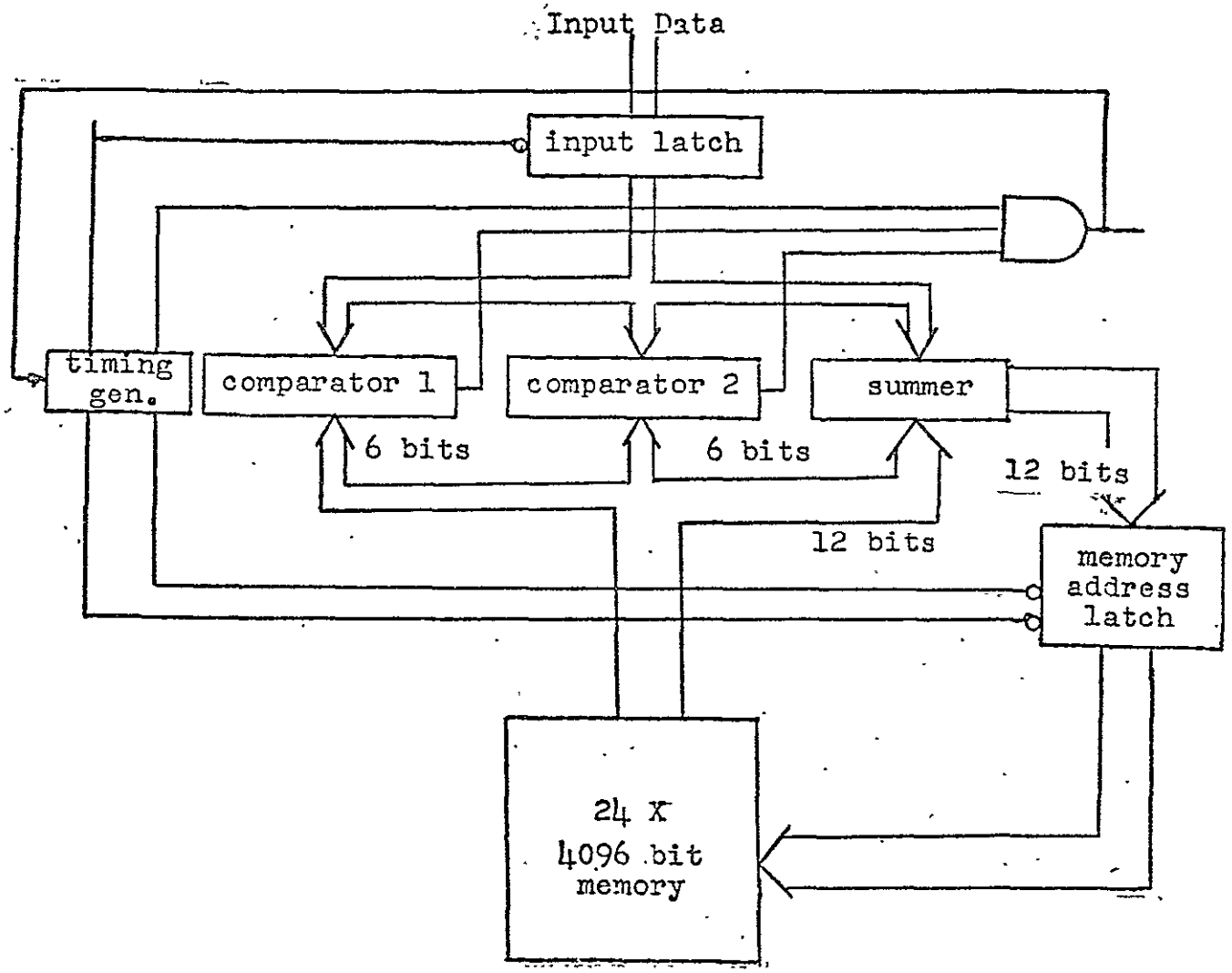


Figure 2.2.8(16) A Hardware Table Lookup Processor

dissipation in each of the processors would be approximately one watt. The processing delay encountered in the comparator stage amounts to no more than six gate delays. A twelve-bit, full carry, look-ahead adder may be implemented in three TTL IC's requiring 1.6 watts and performing a sixteen bit addition in eight gate delays or approximately 60 nanoseconds. The memory address and data input latches may comprise three TTL IC's which dissipate a total of one watt and introduce an additional 15 nanoseconds processing delay.

The time required to generate and latch the new address is 75 nanoseconds; the time required to process each feature dimension is this time plus the memory access time. For currently available NMOS RAM's, the memory access time is in the neighborhood of 500 nanoseconds, so the real

processing delay in this type of processor is caused by memory access delays. An estimated 4096 words of length 24 bits are required for each of the classes the processor must recognize. Using current memory density and dissipation figures, the number of memory IC's required is about 24 per processor and these IC's dissipate about 12 watts.

The total IC count for a twelve-class, six-bit-resolution computer is approximately 400, and the total dissipated power is roughly 200 watts. The volume of the unit is approximately 1/2 cubic foot.

REFERENCES

1. Taub, A. H. (ed.), John Von Neuman - Collected Works, vol. 5., Pergamon Press, Oxford, 1963.
2. Lewin, D., Theory and Design of Digital Computers, Thomas Nelson and Son Ltd., Canada, 1972.
3. Hellerman, H., Digital Computer System Principles, 2nd Ed., McGraw-Hill Book Company, New York, 1973.
4. Flynn, M. J., "Very High-Speed Computing Systems", Proc. IEEE, vol. 54, no. 12, December, 1966, pp. 1901-1909.
5. Thornton, J. E., "Parallel Operation in the Control Data 6600", Proc. AFIPS 1964 FJCC, pt. 11, pp. 33-40.
6. Higbie, L. C., "Supercomputer Architecture", Computer, December, 1973, pp. 48-58.
7. Anderson, D. W., Sparcio, F. J., and Tomasulo, R. M., "The Model 91; Machine Philosophy and Instruction Handling", IBM J. Res. and Dev., November 1966.
8. Graham, W. R., "The Parallel and the Pipeline Computers", Datamation, April 1970, pp. 68-71.
9. Ramamoorthy, R. V., and Kim, K. H., "Pipelining--The Generalized Concept and Sequencing Strategies", Proc. National Comp. Conf., 1974, pp. 289-297.
10. Holland, S. A., and Purcell, C. J., "The CDC STAR-100: A Large Scale Network Oriented Computer System", Proc. IEEE Computer Conf., September, 1971, pp. 55-56.
11. Watson, W. J., "The TI-ASC - A Highly Modular and Flexible Super Computer Architecture," AFIPS, FJCC, 1972, pp. 221-230.
12. Unger, S. H., "A Computer Oriented Toward Spatial Problems," Proc. IRE, vol. 46, October, 1958, pp. 1744-1750.
13. Slotnick, D. L., Borch, W. C., and McReynolds, R. C., "The Solomon Computer", AFIPS PROC. FJCC, vol. 22, 1962, pp. 97-107.
14. Barnes, G. H., et al., "The ILLIAC IV Computer," IEEE Trans. on Computers, vol. C17, August 1968, pp. 746-757.
15. Bell, C. G., and Wulf, W. A., "C.mmp - A Multimini processor", AFIPS Proc. FJCC, vol. 41, Part II, 1972, pp. 765-777.

16. Feth, G. C., "Memories are bigger, faster--and cheaper," IEEE Spectrum, pp. 28-35, November 1973.

17. Koehler, H., "Advances in Memory Technology," Computer Design, pp. 71-7, June 1974.

18. Eimbinder, J., Semi-Conductor Memories, Wiley-Interscience, New York, pp. 8-10, 1971.

19. Brewer, J. and Hadden, D. R., Jr., "Block-Oriented Random Access MNOS Memory," 1974 National Computer Conference, Washington, D. C., pp. 837-40, September 1974.

20. Falk, H., "Microcomputer Software Makes Its Debut," IEEE Spectrum, vol. 11, no. 10, pp. 78-84, October, 1974.

21. Pazaris, S. D., "A 40ns 17-Bit by 17-Bit Array Multiplier," IEEE Trans. on Comp., vol. C-20, April 1971, pp. 442-447.

22. Ghest, R. C., "A 2's Complement Digital Multiplier--the AM 2505," Application Note, Advanced Micro Systems, November 1971.

23. Stefanelli, R., "A Suggestion for a High-Speed Parallel Binary Divider," IEEE Trans. on Comp., vol. C-21, January 1972, pp. 42-55.

24. Cappa, M., and Hamacher, V. C., "An Augmented Iterature Array for High-Speed Binary Division," IEEE Trans. on Comp., vol. C-22, February 1973, pp. 172-175.

25. Eppler, W. G., "An Improved Version of the Table Look-Up Algorithm for Pattern Recognition," Proc. of the 9th Intn'l Symp. on Remote Sensing of Environment, University of Michigan, April 1974, pp. 793-812.

26. Jones, C., "Implementation of an Advanced Table Look-Up Classifier for Large Area Land-Use Classification," Proc. of the 9th Intn'l Symp. on Remote Sensing of Environment, University of Michigan, April 1974, pp. 813-824.

27. Odell, P. L., and Duran, B. J., "Comparison of Some Classification Techniques," IEEE Trans. on Comp., vol. C-23, no. 6, January 1974.

28. Fu, K. S., Landgrebe, D. A. and Philips, T. L., "Information Processing of Remotely Sensed Agricultural Data," Proc. IEEE, vol. 57, no. 4, April 1969, pp. 639-653.

29. Enslow, F. H., (Ed.), Multiprocessors and Parallel Processing, Wiley-Interscience, New York, 1974.

2.3 ON-BOARD PROCESSOR ENVIRONMENTAL EFFECTS

The components of an on-board processor can be divided into several classes. Each class will contain a particular type of component, and each class will be affected differently by the space environment. The majority of the components will be made up of four classes: (1) the electronically-alterable programmable read-only memory (EAPROM); (2) the random-access memory (RAM); (3) the microprocessor chip (μP); and (4) the hardwired logic chip. The environmental effects on each class of components is heavily

dependent upon the spacecraft orbit and the spacecraft environment.

2.3.1 Orbit

For earth-resources missions there are two types of orbits. One is the earth-synchronous orbit in which the spacecraft continuously views a given area of the earth's surface, and the other is the sun-synchronous orbit in which the spacecraft crosses the equator at the same time of day.

2.3.1.1 Earth-Synchronous Orbit

In order that the spacecraft remain at a fixed point with respect to the earth's surface, the spacecraft orbit must be equatorial, circular, and at an altitude of 35,870 km. Such an orbit would provide opportunities for viewing of short-lived events while they occur, or at the first cloud-free opportunity, and/or monitoring time-variable environmental phenomena.

The major potentially-damaging effect at synchronous altitude is space radiation due to the trapped electrons in the Van Allen belts. The electron flux can be enhanced by a magnetic storm which raises the quiet-day flux more than an order of magnitude. The quiet-day flux is around 10^6 electrons per square cm per sec. [1]. Of more importance is the dose received by the spacecraft during the transfer orbit when the radiation belts are penetrated. The major spacecraft damage is to the solar cells since they are on the outer surface. The spacecraft shell provides some shielding for the electronics. Metal-oxide silicon field-effect transistors (MOSFET) are more susceptible to the effects of radiation than are bi-polar transistors. An evaluation of the effects of radiation received during a penetration through the Van Allen belts can be made by observing the radiation data taken on MOSFETs with various thicknesses of shielding. In previous NASA tests, ten orbits of penetrating the Van Allen belts each orbit (four days) produced a 30-mV shift in the gate threshold for one-quarter gram per square cm of aluminum shielding (one thickness of spacecraft skin). One-gram and two-grams-per-square-cm shielded MOSFETs showed no deterioration. For the types of components projected at present, there appears to be no problem with space radiation at synchronous altitude when normal shielding precautions are taken.

Another area of concern in spacecraft environmental system design is the temperature control. At synchronous orbit altitudes the spacecraft is almost always in the direct sunlight except for brief periods when the sun

is eclipsed by the earth. The temperature can be controlled fairly accurately and maintained at a value which provides good safety margins. Based on the operating temperature range of the existing components in the four classes above, no temperature problems are anticipated at synchronous orbit.

At the time of writing this report, ATS-6 had experienced a year's exposure [2] to the synchronous orbit environment. The spacecraft carries two small general-purpose digital computers for use in the altitude-control system. These computers show no effect to this environment. The temperature of the Service Module has varied from 22°C to 26°C which is well within the 5°C to 35°C temperature specification.

2.3.1.2 Sun-Synchronous Orbit

The typical earth-resources mission has been flown at 900-km altitude. The spacecraft crosses the equator at the same local time each revolution; LANDSAT-1 crossed the equator at approximately 9:40 a.m. each orbit. The relationship of the ERTS-orbital plane to the centers of the earth and the sun remains constant while the earth rotates beneath the observatory [2]. At this altitude the earth is completely covered in 18 days.

The 900-km earth-resources orbit is just below the proton and the electron trapped-radiation belts. The flux from magnetic storms does not penetrate to this altitude. Since considerable experience has already been gained for operation of field-effect transistors at this altitude, the present indication is that normal chassis covers provide sufficient shielding against space radiation for MOSFET devices.

Currently, NASA is experiencing problems with CMOS devices at higher altitudes. While early CMOS devices were sufficiently immune to radiation effects, the process line was changed and later devices show a greater susceptibility to radiation. This is a temporary problem, and based on past performance, it is safe to predict that during the 1980's the CMOS devices will once again be sufficiently immune to radiation effects.

2.3.2 Spacecraft Environment

During the 1980's the method of launch of an earth-resources spacecraft will be by the space shuttle. The shuttle can place a 6000-kg payload in a 620-km sun-synchronous orbit; however, it can only place a 600-kg payload into a 900-km sun-synchronous orbit. A small decrease in orbital altitude results in a large increase in orbited payload weight.

To be cost-effective, the shuttle requires a large number of users and a large weight-carrying capability. This can be achieved at the lower altitudes. Utilization of a higher weight payload usually results in a lower cost for the payload. Redundancy can be incorporated plus additional shielding so that lower cost components can be used. The additional weight can also be used to control the payload environment so that environmental testing can be reduced. Based on present-day technology and the successes in flying payloads in both earth-synchronous and sun-synchronous orbits, the on-board processor designer should have no trouble in meeting the spacecraft environmental specifications for either sun-synchronous or earth-synchronous orbits.

REFERENCES

1. Rickets, L. W., Fundamentals of Nuclear Hardening of Electronic Equipment, Wiley-Interscience, New York, pp. 76-79, 1972.
2. Applications Technology Satellite ATS-6, "Experiment Check-Out and Continuing Evaluation Report," NASA--Goddard Space Flight Center, Greenbelt, Md., X-460-74-340, December, 1974.

3 TECHNOLOGY FORECAST AND ASSESSMENT

The ability of the on-board processors designed in Section 2 to implement the algorithms described in Section 1 in real time for the required throughput data rates depends on the components that will be available at the time of system design. The lead-time required for design, procurement, fabrication, checkout, and launch is about 5 years, so that 1980-1990 launches will utilize 1975-1985 technology. Consequently, we require accurate component and system technology forecasts for the next 10 years.

Section 3.1 deals with performance measurement criteria.

Section 3.2 contains a survey of the electronic component technology available in 1975. Future improvements in component technology from 1975 to 1985 are projected.

In Section 3.3 we review the computer system technology available in 1975 and we forecast future system technology using both manufacturers' estimates and a technology forecasting model.

Section 3.4 contains a survey of existing satellite on-board computers and a discussion of future on-board processor technology.

In Section 3.5 we develop a forecast feedback system that allows the incorporation of the projected component and system technologies into the on-board processor architectures. The results are then used to obtain a better estimate of projected performances.

Finally, Section 3.6 contains a discussion of a component and system technology that is in its infancy, but which, with sufficient stimulation, could make a significant impact on on-board processors toward the end of the 1980-1990 decade.

3.1 PERFORMANCE MEASUREMENT CRITERIA

Many measurement performance criteria have been proposed to evaluate hardware structures, algorithmic processes, and computer systems. See, e.g., bibliographies on performance evaluation by Buchholz [2] and Miller [3]. An interesting trend can be discerned from the historical literature on performance measures.

Skalansky [4] suggested a measure to evaluate the hardware performance characteristics of several binary adders. The evaluation criteria was defined in terms of the adder's efficiency $\eta = R/I$. The value of R was taken as the number of bits in each operand. Several formulas for generating

values of I were proposed using the gate - normalized addition time and the number of two-input AND gates and OR gates.

In an effort to compare the efficiency of algorithms in processing matrices and linear equations, Householder [5] indicates that the number of numerical operations plus the number of recordings of intermediate results generate an overall measure of the efficiency of the computational process. If a benchmark such as time were used as the measure for evaluating each system's performance, the affect of the system's instruction set and architectural hierarchy would be integrated by the execution of the computational task and provide a significant indication of system performance.

A method for evaluating the computing power of various systems known as throughput [6-8] is defined in terms of the specific task to be performed. The task might be the compilation of a FORTRAN statement or the execution of an I/O statement. In this context, throughput is measured in jobs per unit time. An interesting aspect of this measure is that the operating system, compiler, and data management affect the measure as well as the hardware architecture and memory hierarchy. This method of evaluation has been proposed and used for large, complex digital systems in multiprocessing, multiprogramming, and real-time environments.

Other meaningful measures for performance measurement and evaluation of complex computing systems are based upon workload characteristics [9], utility functions [10], hardware monitoring [11], and data base of performance [12,13].

Anacker and Wang [14] presented a method for performance evaluation of computing systems with memory hierarchies in which the degradation of performance due to system architecture and data flow paths in the CPU and memory were taken into consideration by calculating the sum of instructions plus data words processed or manipulated per unit time by the system in carrying-out a given task.

Leis [15] introduced a parameter called "setup" and explored its relationship to processing power and instruction set in evaluating hardware design. A computer architecture was identified as the set of constraints the system imposes on the user in executing his computational task; setup is a measure of the amount of data manipulation performed prior to the system's execution of an instruction identified as a step in the algorithm being executed. For instance, if a system architecture is organized

around a single address instruction format, setup is required to access one of the operands prior to the execution of an arithmetic operation. A system organized around a two- or three-address instruction format does not require the same amount of setup. Changes in setup are measured in terms of overhead. The more setup required in a given application points to a lower performance.

Trends are toward the development of performance measures which integrate all the operating characteristics of the computing system. This study incorporates these considerations by taking into account both the hardware organization (system architecture) and the software overhead (number of microprocessor cycle times) required to optimize and implement the algorithms discussed in Section 2.

3.2 COMPONENT TECHNOLOGY

3.2.1 1975 Component Technology

Some 1975 micro-electronic technology families are listed in Table 3.2.1(I), and a qualitative comparison of some MOS circuit techniques are listed in Table 3.2.1(II).

Table 3.2.1(I) Some 1975 Component Technologies

FAMILY	DENSITY GATES/mm^2	LSI ON-CHIP POWER-DELAY PRODUCT, pJ 15V	5V	1V	SMALLEST DELAY, ns
SCHOTTKY BIPOLAR	30-40	-	5	-	2
CMOS	30-40	50	5	-	10
STATIC NMOS	80-120	50	5	-	20
CMOS/SOS	80-120	25	3	-	3
I^2L BIPOLAR	100-120	-	5	1	10

The reliability of these components is illustrated by the statistics listed in Table 3.2.1(III). The mean time between failure (MTBF) of 10^3 - 10^5 hours for a 1000 component system is marginal for an on-board processor.

Present LSI technology allows up to several thousand gates per chip.

The maximum power dissipation per chip ranges from 0.5-2.0 W; very few dissipate more than 1 W.

Two present day memory technologies are illustrated in Table 3.2.1(IV). Both are 8,000 bits. The charge coupled device (CCD) has a faster transfer

Table 3.2.1(II) Qualitative Comparison of MOS Circuit Techniques

Circuit Technique	Speed	Power	Density	Advantages	Disadvantages
Static Logic	0-2 MH_z	5 mW Per Gate	200-300 Gates Per Chip Readily Obtained	Runs at any frequency d.c. to max. No external clocks required. Easiest of PMOS types to interface with bipolar. Simplest to use for system design.	Slowest. Highest Power. Higher cost than 2ø or 4ø (larger chip area per function).
2 ø Dynamic	10 Kc-5 MH_z	Lower Power than Static Logic (depending upon Frequency and Pulse Width.	Higher Density than Static Logic.	Lower Power. Higher Speed. Better Noise Immunity than Static Logic.	Minimum Frequency of Operation. External clocks must be supplied, therefore more system noise and complication.
4 ø Dynamic Logic	10 Kc-10 MH_z	Only Reactive Power, so total power is less than static or 2ø Dynamic Logic. Microwatt d.c. power dissipation	Higher Density than 2ø Dynamic Logic	Still Lower Power. Higher speed than 2 ø. Better Noise Immunity than Static Logic r 2ø	Minimum Frequency of operation. External Clocks required, therefore more system noise and complication.
Complementary MOS Logic	0-25 MH_z	Microwatt d.c. power dissipation	Lower Density than Static Logic. Approx. 300 Gates Per Chip.	Lowest System Power. Highest Speed. Good Noise Immunity. Single Power Supply. Low Output Impedance in both directions. Wide Power Supply Variations.	Largest MOS Chip Area. More processing steps required. Highest MOS die cost.

Table 3.2.1(III) 1975 Component Reliability

Chips passing die sort electrical test	100
Chips passing visual inspection of lead bonds	98
Components passing all tests after packaging	80
Components operable at first system test	78
Components operable after 18 hour burn at maximum temperature.	76
Subsequent failure rate of surviving components in normal service	.001-.01 per 1,000 hours
Lab or computer room severe environment	.01-0.1% per 1,000 hours
MTBF for 1,000 component system	10^3-10^5 Hours

Table 3.2.1(IV) Comparison of RAM and CCD for Bulk Storage

	8K RAM (IBM, 1973)	8KCCD (Bell-Northern, 1974)
Die Size (MILS)	145 x 201 (29K MIL^2)	168 x 178 (30K MIL^2)
Access	Random to 64 b Blocks	Random to 256 b Blocks
Bit Transfer Rate	1.8×10^6 b/s	10^6 b/s
Active Power (MW/8K)	22.5	15
Standby Power(MW/8K)	2.5	1
Technology	Std. P-SI Gate	2-Level N-SI Gate

rate but slower access time because it is a dynamic shift register device so that, like a disc memory, readout is delayed until the data comes past the readout electronics.

3.2.2 1975-1985 Component Technology

The equivalent number of gates per IC package (Chip) increased two orders of magnitude from 1960 to 1970 (from about 5 to about 500), and another order of magnitude between 1970 and 1975 (about 5,000 by the end of 1975). Three more orders of magnitude remain before the optical diffraction limit is reached. Figure 3.2.2(1) illustrates this evolution from SSI (small scale integration) through MSI (medium scale integration) to LSI (large scale integration) [20]. All indications are that the number of gates/chip will increase by another two orders of magnitude between 1975 and 1985.

For some IC's the number of pins on the package must increase in order to effectively utilize the increased number of gates. The pin count per package has increased exponentially over the past 15 years. A less than exponential increase is predicted for the next 10 years because of problems in manufacturing, assembly, seal, and testing. The insertion yield and repairability will also tend to slow the growth. A fast 16-bit CPU would require 100-150 pins. This appears feasible as illustrated in Figure 3.2.2(2).

Future memory technology is projected in Table 3.2.2(I). For the 7 year span that is shown it is expected that cell area, power/bit, and MTBF/bit will improve by about one order of magnitude. Storage capacity and cost increases approach one-and-a-half orders of magnitude. Some experts are predicting a 128K bit chip by 1980. All indications suggest between one and two orders of magnitude improvement between 1975 and 1985. Power requirements are directly proportional to cycle time, but only to the

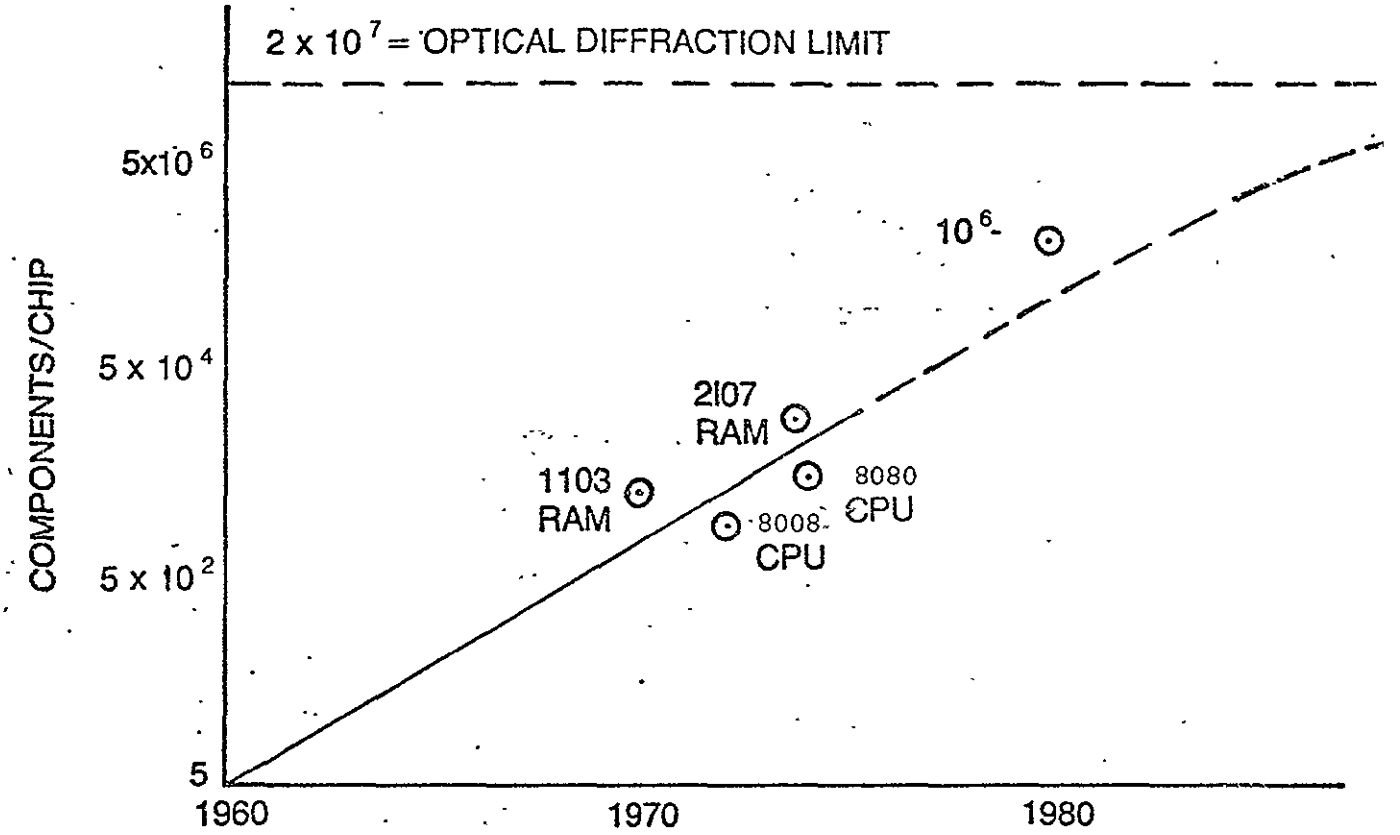


Figure 3.2.2(1) Maximum Components/Chip vs Time (Projection)

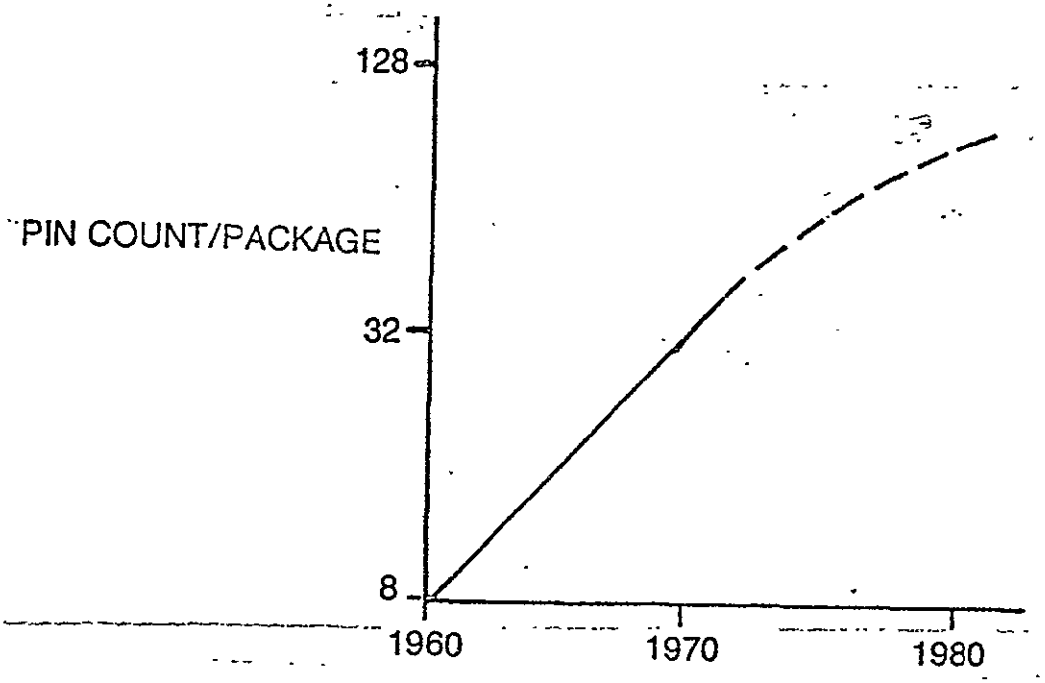


Figure 3.2.2(2) LSI Package Pin Count Limitations

Table 3.2.2(I) Possible Trends in Dynamic MOS RAM ∿ 500 ns Cycle Time

Storage Capacity	1K	4K	16K	64K
Date Introduced	1972	1974	1976	1979
Cell Area, Mil^2	6	2	1	0.5
Chip Area, In^2	.02	.03	.04	.06
Power/Bit, μW				
Active	300	100	20	4
Standby	4	1	0.5	0.2
MTBF/Bit, Hr.	10^{10}			10^{11}
Component Mfg. Cost/Bit, 2 yr. After Intro. -¢	0.3	0.1	.03	.01
Memory System Price/Bit to OEM -¢	1.2	0.4	.12	.04

square root of storage capacity (Power ∿ $\sqrt{\#\ bits}$ · access time) and are not expected to pose any problems.

Present prices of microprocessor components reflect an attempt by suppliers to recover some of their development costs. The factory cost of a microprocessor (or any manufacturable LSI component) is under ten dollars. Rapid decreases in the price of microprocessor components are likely over the next few years. Memory, input-output devices, and power supplies may well be more important factors in the total cost of a system.

The engineering cost of a microprocessor chip is substantial, perhaps on the order of 0.5 to 1 million dollars. This is a significant barrier to the development of custom microprocessors for single users.

The performance, cost, and reliability of many microcomputer systems is determined by memory. Continued rapid decrease in the cost of memory is likely. Charge-coupled device memories and bubble memories are not likely to challenge the dominance of conventional semiconductor memory techniques. Eventually read-write memory and mask-programmed read-only memory will approach the same cost, while electrically programmable read-only memory will continue to be more costly.

It should be possible to build highly reliable microcomputer systems through use of well-planned but relatively cheap testing, screening, and burn-in procedures.

Some 1980 LSI costs projections are presented in Table 3.2.2(II) alo with some of the parameters governing costs.

3.3 SYSTEM TECHNOLOGY

System technology is progressing at a rate comparable to component technology. Some 1973 and projected 1978 minicomputer and microcomputer system characteristics are illustrated in Table 3.3(I).

Table 3.2.2(II) Projected 1980 LSI Component Prices

	MOS	BIPOLAR
Chip size for 30% yield	1.5 cm^2	0.6 cm^2
Cost/good Chip	$9.00	$3.60
Number of devices/chip	10^6	10^5
Number of gates/chip	10^5	10^4
Factory cost of package	$28.00	$17.00
Factory cost/gate	.03¢	.17¢
Selling price/gate	.06¢	.35¢

Table 3.3(I) Comparison of Minicomputer/Microcomputer Characteristics

	1973		1978	
	Minicomputer	Microcomputer	Minicomputer	Microcomputer
Execution time (μs)	0.5-2.0	2.0-25.0	0.1-2.0	0.1-10.0
Word Length (bits)	8-32	4-8	8-32	4-16
Number of Instructions	100-200	20-60	150-250	100-200
Technology	Core	MOS	Bipolar/MOS	Bipolar/MOS

3.3.1 Technology Forecasting Model

Under the Apollo program [16] NASA developed a system to forecast space vehicle weight and performance based upon trend forecasting. Because of the success of this technique and the excellent documentation available, it was adapted to the forecasting of computer performance.

As illustrated in the preceeding sections, component and system parameters are progressing at an exponential rate. Consequently, our model is based on the logarithm of the NASA linear maximum likelihood model.

The computation and plotting routines of the linear maximum-likelihood model were modified to accept the logarithm of the input data. To make a projection that would be good to the year 1990, input data over the range of 1960 to 1975 was needed. The periodical, Computers and Automation, publishes a monthly survey of computers along with their first date of installation. Every few years, beginning in 1960, the periodical has published a digital computer specification survey. This survey includes cycle time, add time, multiply time, etc. Using the monthly survey and the specification survey, data versus time can be extracted for use in the prediction program.

Figure 3.3(1) resulted from using input data taken from minicomputer surveys for the years 1970 to 1975 to predict computer cycle time. A

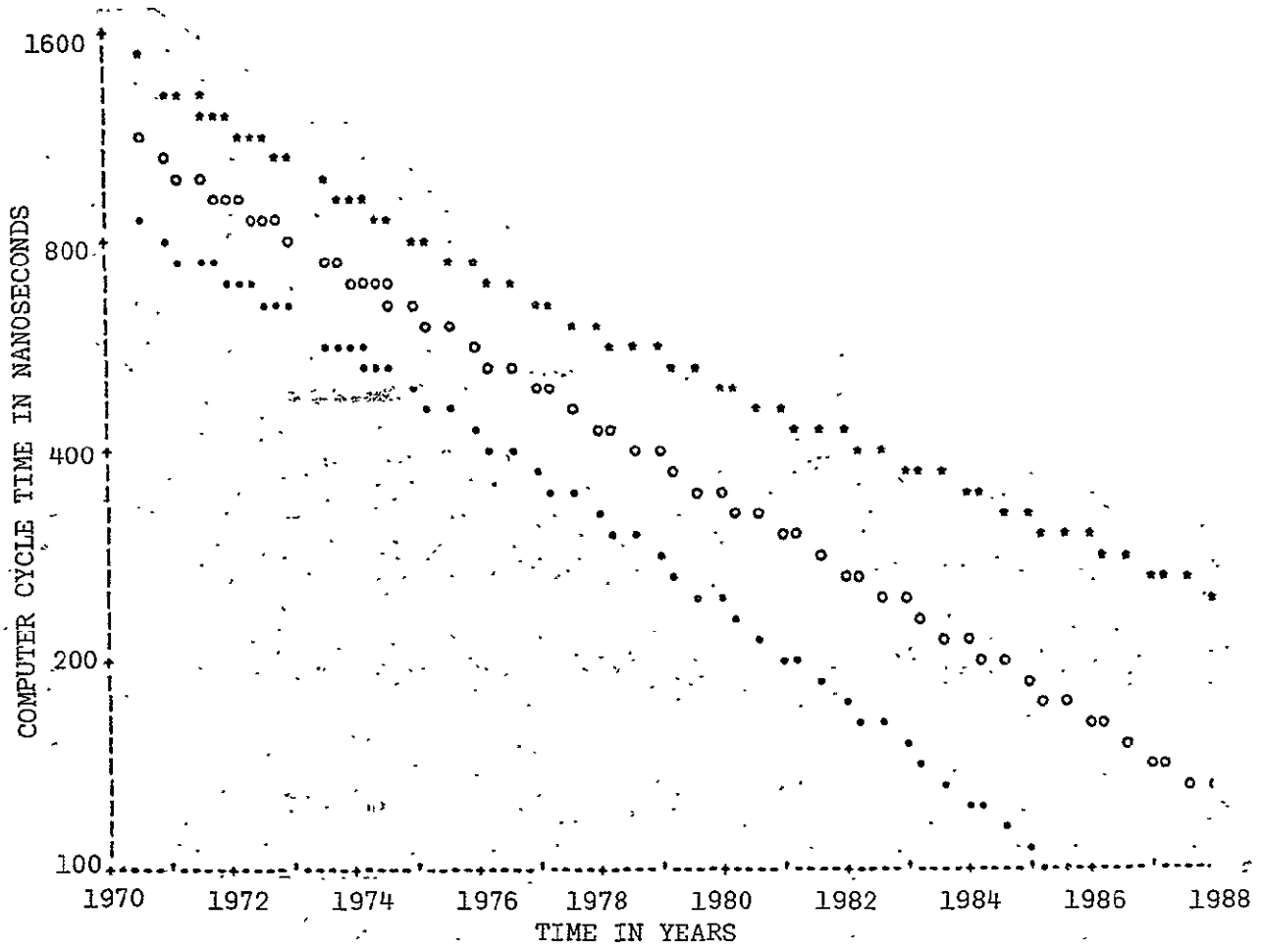


Figure 3.3.1(1) Computer Cycle Time Forecast

more meaningful result was obtained by taking minicomputer add times for the years 1970 to 1975 and combining it with the add times of computers from 1960 to 1970. The resulting prediction is shown in Figure 3.3(2). The expected value for the computer add time in 1983 will be 100 ns. The small circles and asterisks represent the 80% confidence limits. This curve is based upon expected values. Even today there are machines with add times of 100 ns, but the average or expected value today is 800 ns. These computer generated projections are in close agreement to the predictions made by experts in the electronics industries discussed in Section 3.2 and 3.3.

3.4 ON-BOARD PROCESSOR TECHNOLOGY

NASA is at present in a program for standardizing its on-board computers. The Goddard Space Flight Center has developed the Advanced On-Board Processor (AOP) [21] which had its first flight on OAO-3 21 August 1972. There is an on-going program to reduce size and power consumption

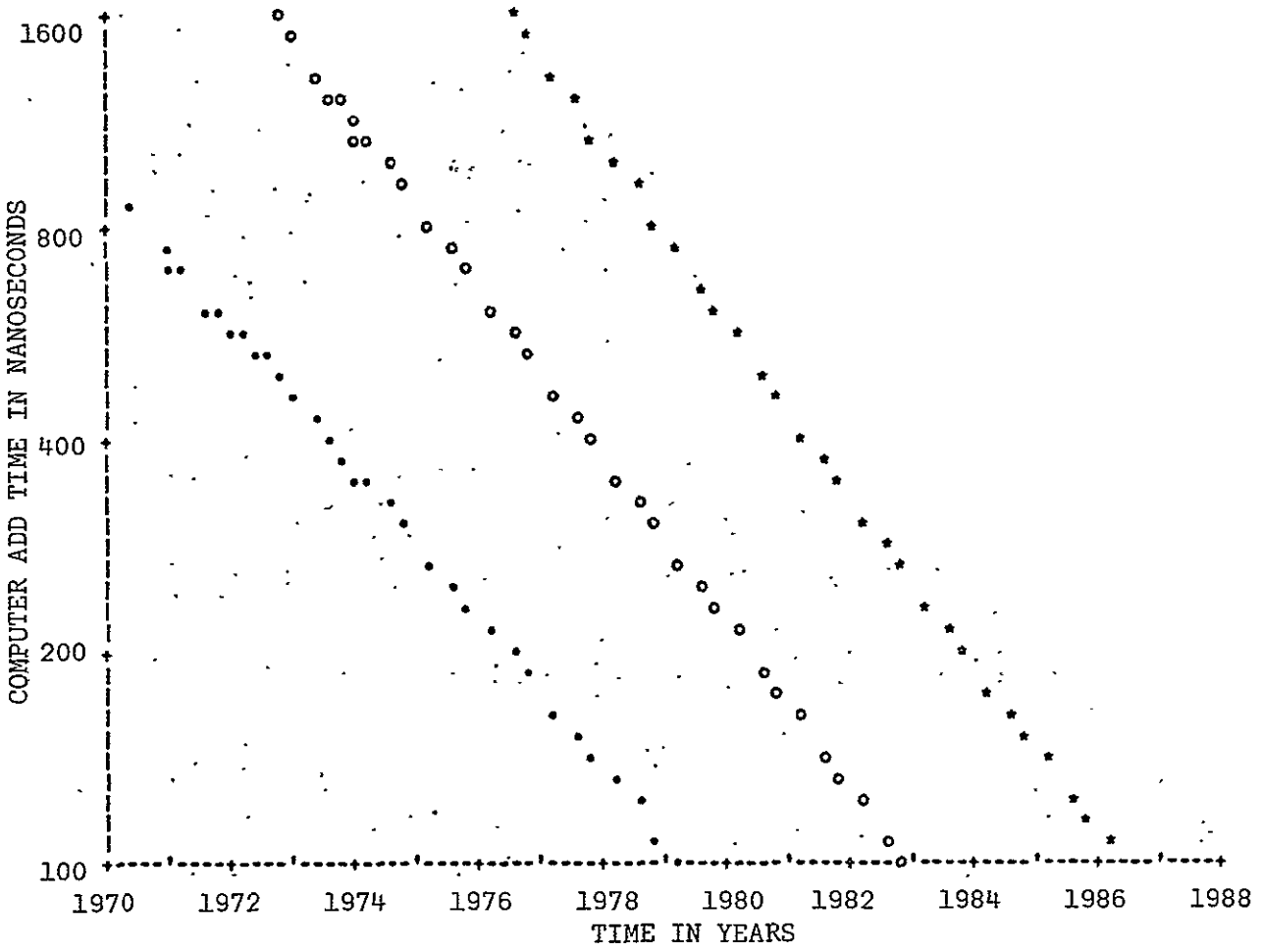


Figure 3.3.1(2) Computer Add Time Forecast

of this computer so that it can be used on smaller spacecraft. The machine has an 18-bit word length and a 4-microsecond add time.

The Marshall Space Flight Center is sponsoring several versious of on-board computers: The Hybrid Technology Computer (HTC) by IBM, the Space Ultra-reliable Modular Computer (SUMC) by RCA, and the Hughes Aircraft Long-Life Fault-Tolerant System. The RCA SUMC is being made of Silicon-on-Sapphire CMOS.

Technology requirements for the on-board processor differ from computer system technology because of space, power, weight, reliability, environment and cost constraints. Many of these parameters, e.g., space qualification and check out time and costs, increase as the square of the number of components. Experience has shown that, while the number of transistors or gates in a logic system has increased by orders of magnitude, the number of integrated circuit packages has remained essentially constant. To support

this theory, the advice of an expert in digital logic design, D. R. Lokerson of NASA/GSFC furnished the data shown in Figure 3.4(1). This figure shows the number of IC packages and the number of transistors or gates for the telemetry encoding system for the Interplanetary Monitoring Platform (IMP) spacecraft over the ten year period 1965-1975. The IMP telemetry encoder increased in computational power over the past decade. It used the newest technological developments; it was the first spacecraft to use integrated circuits and the first spacecraft to use MOS transistors. Note that the number of IC's remained essentially constant while the equivalent number of gates increased by two orders of magnitude over the 10 year period. It is the opinion of Lokerson that systems with more than 1,000 packages become increasingly difficult to test. While the number of gates in the packages may continue to rise, testing, checkout, and reliability of pin connections will dictate a maximum system size of approximately a 1,000 packages per system for the next ten years.

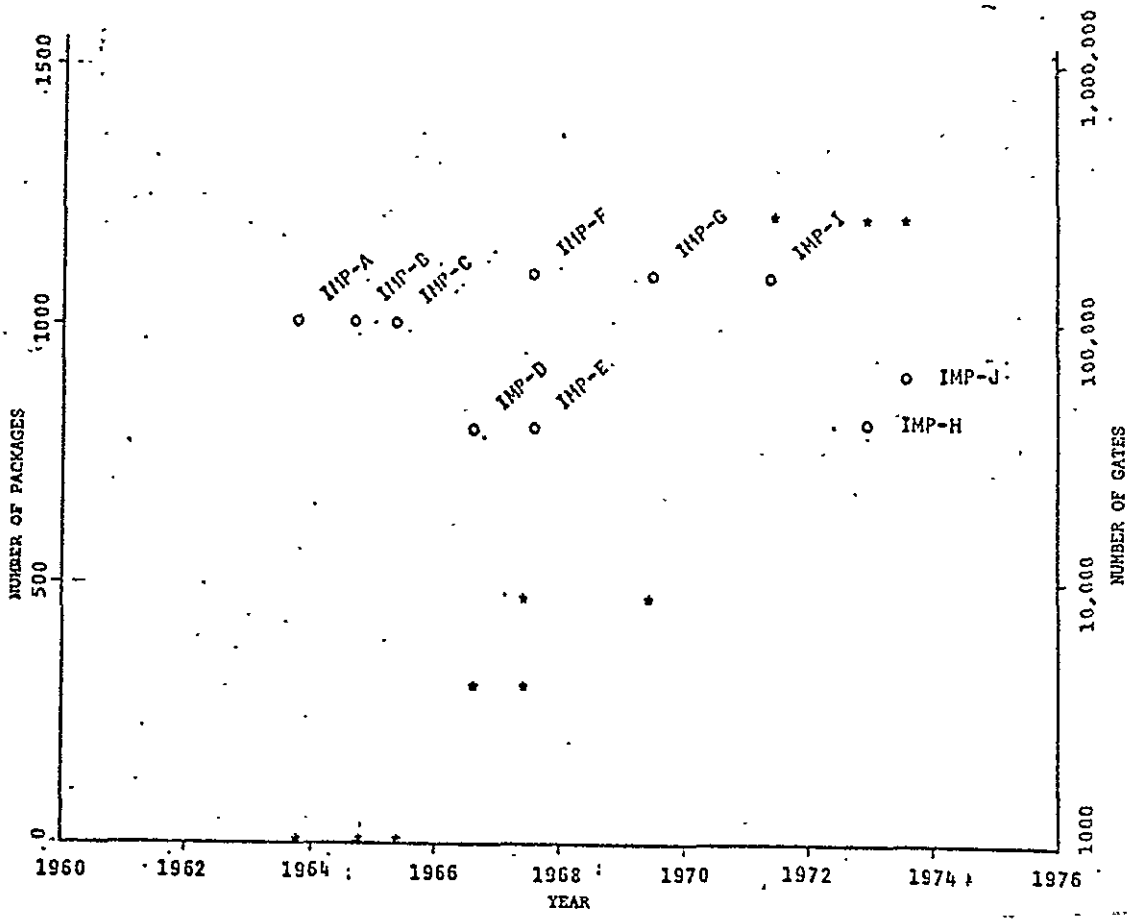


Figure 3.4(1) Numbers of IC Packages and Gates for the IMP Spacecraft

The constraint of 1,000 chips is based mainly on cost. The technology is available for building larger processors; however, the cost to design, test, and check out a processor goes up with the number of packages much faster than linearly. We will, of course, investigate the sensitivity of our results to variations in this data point.

In summary, we conclude the following for the on-board processor for 1975-1985:

(1) The maximum allowable number of IC packages will stay constant at 1,000.

(2) The add time will increase by one order of magnitude.

(3) The number of gates per IC package will increase by one-to-two orders of magnitude.

(4) The data throughput capability will increase by two-to-three orders of magnitude. (This follows from items 1, 2, and 3.)

3.5 FORECAST FEEDBACK SYSTEM

Because of the complex interrelations between on-board processor requirements, architectures, etc., a systematic approach based on a feedback model was developed. The object of the forecast feedback system is to help give the best estimate of on-board processor architecture organization and performance. The forecast feedback system block diagram is shown in Figure 3.5(1).

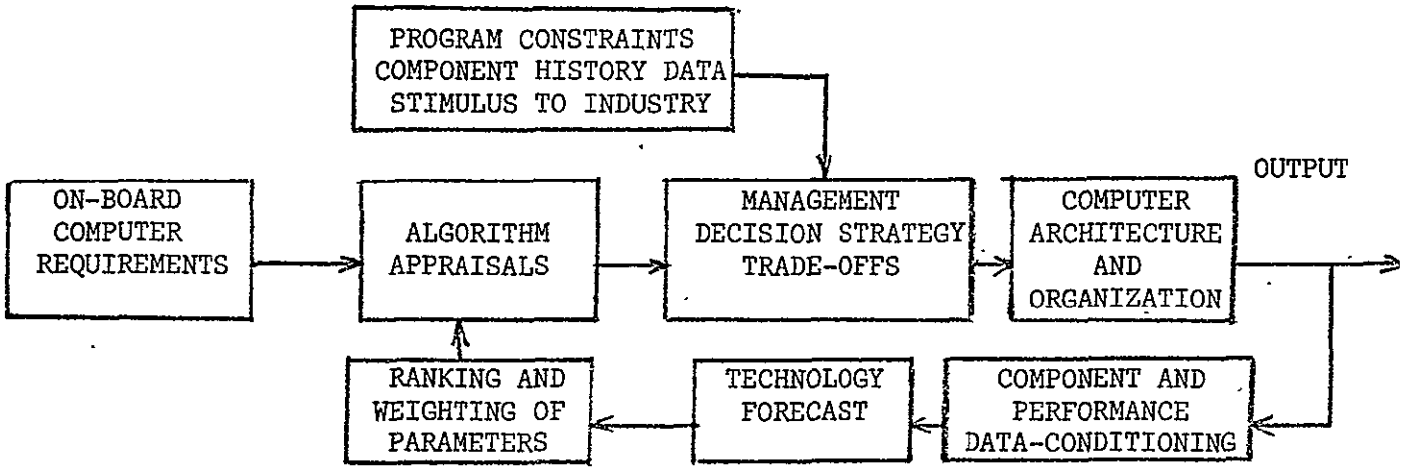


Figure 3.5(1) Forecast Feedback System Model

3.5.1 Input Data

Data is fed into the loop in two inputs. The primary input is the on-board computer requirements data. These are the data requirements generated in Section 2.1 of this study. To generate the first set of data, typical applications were selected in the fields of agriculture, geography, meteorology, and oceanography. On-board processor requirements were generated to satisfy the spatial and spectral requirements. They are fed into the loop by means of the block in Figure 3.5(1), labelled Algorithm Appraisals. Here the appropriate algorithm is selected or indicated. In the first iteration of the loop there is no feedback to perturb the algorithm appraisal. The algorithms are then passed on to the block labelled Management Decision Strategy Trade-Offs. The second set of inputs occur in this block. These inputs are the program constraints, the component history data and any stimulus to industry which may affect the rate with which technological advances are made. NASA management, along with the members of the study team, made the decisions and trade-offs required to select the appropriate algorithms consistent with the program constraints. The algorithms are then passed on to the computer architecture designer (box labelled Computer Architecture and Organization).

3.5.2 Output Data

The output from the feedback system is taken from the box of Figure 3.5(1) labelled Computer Architecture and Organization. The output is two computer designs, one a general-purpose computer and the other a hardwired special-purpose computer. Each is based upon using present-day technology without benefit of a technology forecast.

3.5.3 Feedback Loop

The component parameters required for the implementation of the two designs are fed into the technology forecasting program for a projection of the expected performance in the 1980's. The parameters were ranked and weighted; processor add time was deemed the most meaningful and useful parameter. This information was passed to the Algorithm Appraisal box where the selection between the maximum-likelihood and the table look-up algorithm occurred. Two new designs resulted, one based upon using a set of general-purpose computers and the other on the use of hardwired logic.

A second and third iteration around the loop produced the final processor architectures described in Section 2.

3.6 OTHER TECHNOLOGIES

There are some new approaches that show promise for drastically increasing the computational power of an on-board processor, e.g., the tse computer and the Josephson tunneling circuits. Both would require considerable stimulation to develop them to their full potential.

3.6.1 The tse Computer

Image information is two-dimensional. Two-dimensional data processing is usually executed as a sequence of serial computations by a word-oriented machine. Several parallel-processing architectures [22-24] have been proposed in which the processor organization is an array of processing elements capable of operating on a number of words simultaneously. Most of these systems have not reached an operational status because of the prohibitive cost involved in their construction.

Some forms of optical computing operate at speeds exceeding those of digital techniques. Optical computing has generally been limited to optical analog devices based upon the mathematical concepts of coherent or Fourier optics and holography. Optical analog computing derives power from the fact that the computing systems are relatively simple and that parallel processing is a natural property of the lens [25]. The area of optical digital devices is not as well-developed. Strong[26,27] has explored the feasibility of performing logical operations on binary images using coherent and non-coherent optical techniques. This research indicates that a two-dimensional computer architecture using coherent and non-coherent optical devices has attractive image processing capabilities.

A program at the Goddard Space Flight Center has removed the ground rules with which conventional computers are designed and a new design philosophy and concept has emerged. Schaefer and Strong [28,29], Earth Observation Systems Division, NASA/GSFC, have proposed a family of two-dimensional logic devices capable of performing simple, parallel logical operations simultaneously on one or two binary images. The following summarizes a small part of their research.

Consider an image composed of a 512 x 512 rectangular array of picture elements in which the gray level of each element is quantized to six bits. One way to visualize the digitized image is as six binary image planes, each plane containing 512 x 512 bits. The binary image plane or bit plane is a two-dimensional binary data array called a "tse" which comes from the word for the Chinese writing characters. Just as one tse represents many English letters, one binary tse represents many binary bits.

A family of tse logic devices which utilize electro-optical technology to perform simple, parallel logical operations simultaneously on one or two tses has been proposed. Figure 3.6.1(1) illustrates a tse gate capable of ANDing two binary image planes. The interleaver is a passive element which combines corresponding positional elements in the image A and the Image B inputs to the

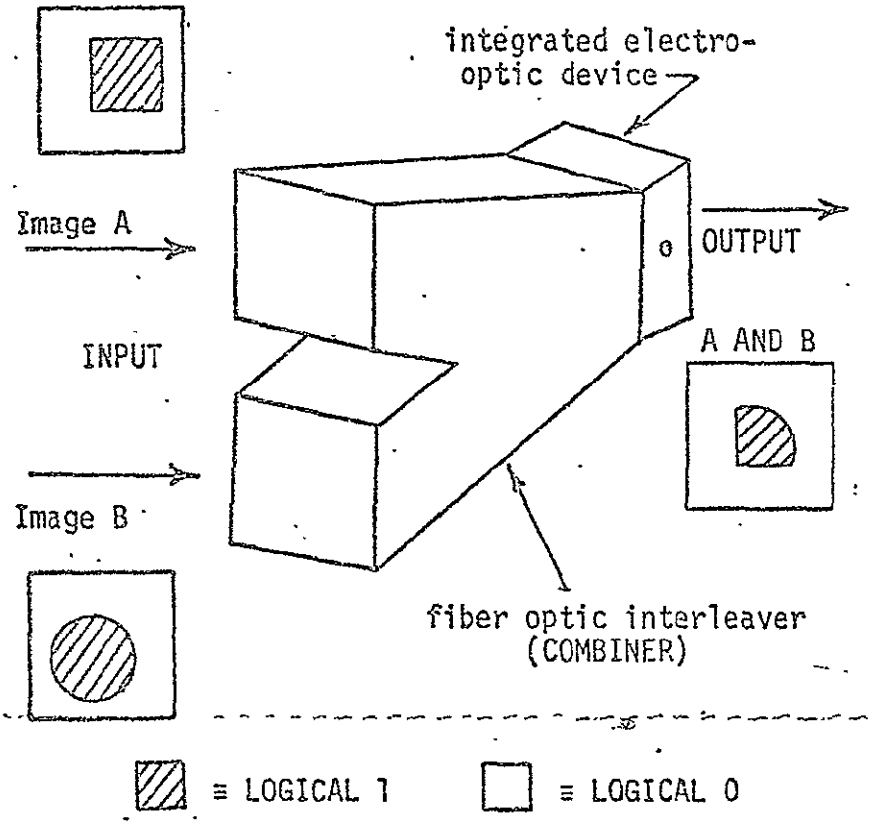


Figure 3.6.1(1) A Two tse Input, Digital AND Gate

same elemental position at the interface of the electro-optical device. The electro-optical device is an active integrated circuit which converts the optical inputs into electrical signals which are logically ANDed in a conventional manner. The output electrical signals are converted to an optical output by an electro-luminescense process. Since the fan-out of the active circuit is one, an interleaver must be used in reverse to duplicate the output image and to increase the effective fan-out. Each output from the duplicator must interface to a REFORMATOR which is an active tse buffer device to restore the proper optic signal levels. Figure 3.6.1(2) demonstrates how the effective fan-out from the AND gate can be increased to four.

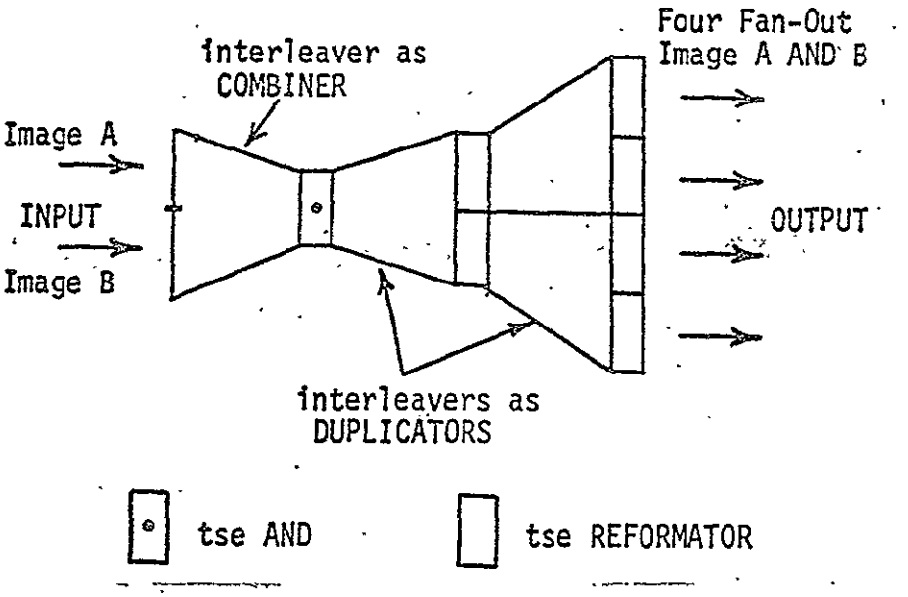


Figure 3.6.1(2) Use of DUPLICATORS to Increase Effective Fan-Out Of a tse Device to Four

Other elementary tse operations are OR, Exclusive-OR, NEGATE, and SLIDE operations. These primitive operations and the results of performing them on typical images are depicted in Figure 3.6.1(3). All of these operations, with the exception of the SLIDE operation, represent conventional logical operations. The SLIDE operation performs an image translation either up, down, right, left, or combinations of these directions. Con-

Consider an image composed of a 512 x 512 rectangular array of picture elements in which the gray level of each element is quantized to six bits. One way to visualize the digitized image is as six binary image planes, each plane containing 512 x 512 bits. The binary image plane or bit plane is a two-dimensional binary data array called a "tse" which comes from the word for the Chinese writing characters. Just as one tse represents many English letters, one binary tse represents many binary bits.

A family of tse logic devices which utilize electro-optical technology to perform simple, parallel logical operations simultaneously on one or two tses has been proposed. Figure 3.6.1(1) illustrates a tse gate capable of ANDing two binary image planes. The interleaver is a passive element which combines corresponding positional elements in the image A and the Image B inputs to the

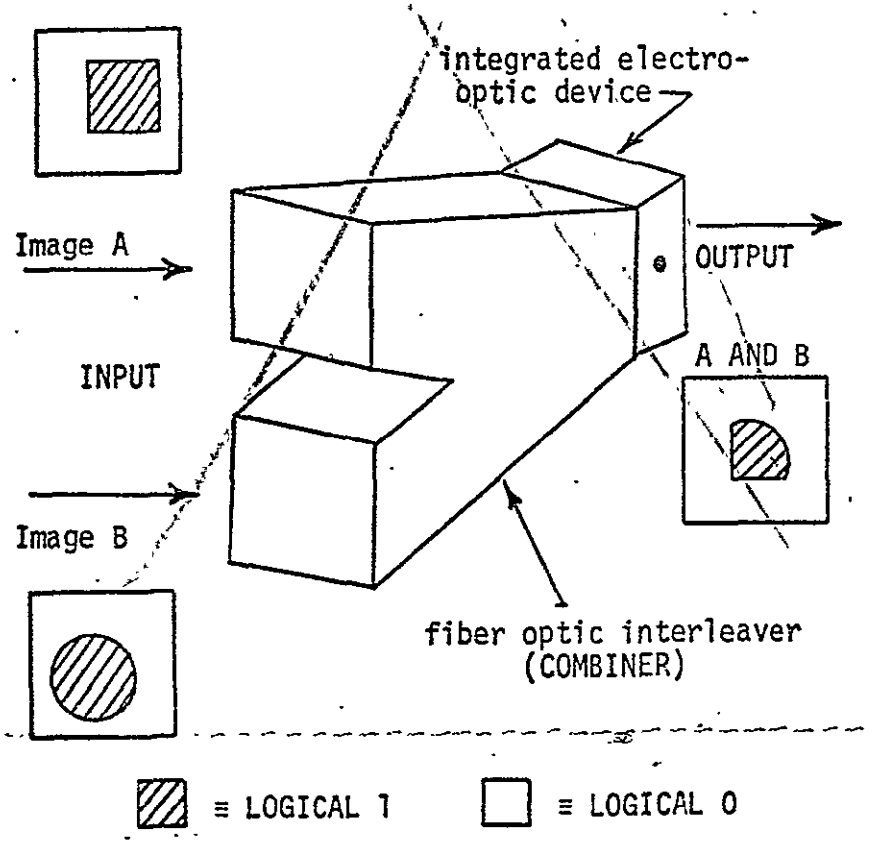


Figure 3.6.1(1) A Two tse Input, Digital AND Gate

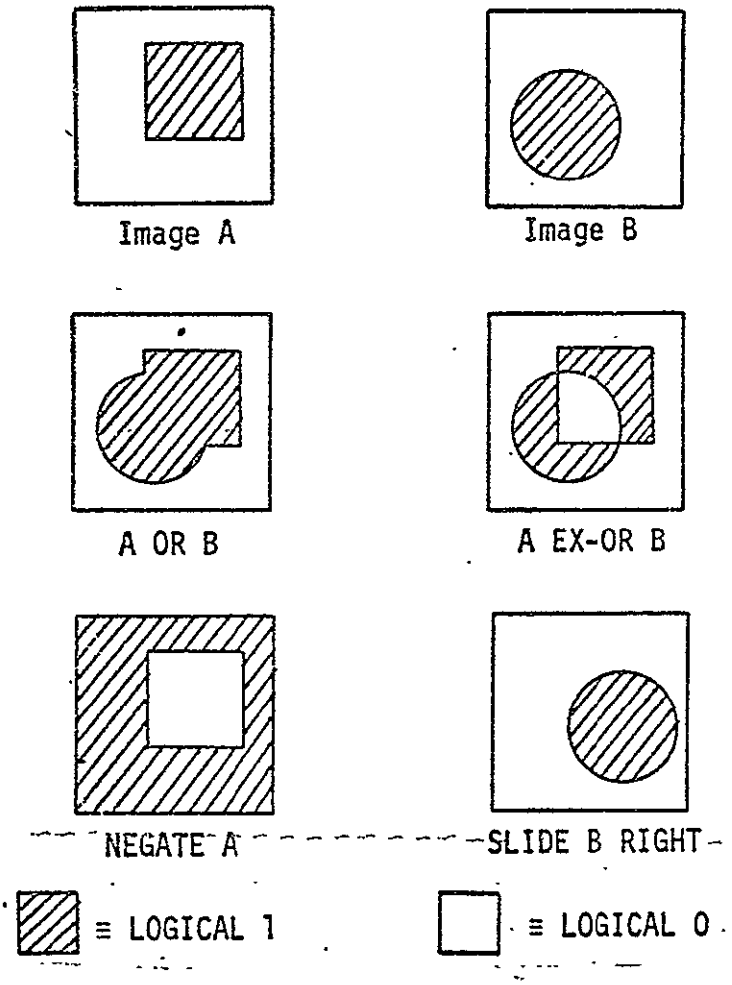


Figure 3.6.1(3) An Example of Elementary tse Operations on Typical Images

ceptually, this operation is generated by interfacing two fiber optic bundles with a physical offset between the bundles. Therefore, when a tse is transmitted along the optic data path of one bundle, the tse at the output of the second bundle is transmitted with the same physical offset. Positions in the output path which are not interfaced to an input because of the offset are masked in such a way that they contribute logical zero in the output image.

A tse "contractor" device is a control device to indicate the presence of a 1-element in any position of the tse. If there are no 1-elements in any position of the binary image, the output of the device is logic-0; otherwise, the output is logic-1. This device is different in that the input is a data

plane, but the output is a single logic signal. A device of this type is very necessary for implementing conditional image operations.

Tse devices can be interconnected in much the same manner as conventional digital logic gates to implement special purpose architectures for executing specific algorithms. As they are interconnected to organize more complicated structures, methods for controlling the devices are essential. All active tse devices are assumed to have a one-bit control line for turning them on or off. In the off state the output tse is a zero-tse (all elements in the array are logic-0).

An example of a system organization with tse components is the LOGICAL OPERATIONS UNIT of Figure 3.6.1(4). This unit is a subsystem in a larger tse computer organization [30] for executing Strong's parallel counting algorithm [28].

Although tse components are still in an early developmental stage, the parallelism produced by these devices readily projects a system structure which allows for the parallel, concurrent, or simultaneous execution of processing tasks. Future advances in integrated circuit and optical technologies in the next 5-10 years are expected to provide effective computer architectures for processing two-dimensional data.

3.6.2 Josephson Tunneling Devices

For over twenty years, attempts have been made to utilize the principles of superconductivity in the building of computers. If the extremely low losses of the superconductive state could be utilized in a logic element, then a very low-power computer could be built. In 1956, D. A. Buck [31] demonstrated the "cryotron," which was capable of performing both logic and memory functions. This device was not competitive with semiconductor computing elements in speed; therefore, it has not made an impact in the computing field. In 1962, B. D. Josephson predicted [32] that supercurrent could flow through an insulator sandwiched between two superconductors. The prediction was later confirmed, and his work spawned a whole host of investigations into devices utilizing this principle. The promise of extremely fast switching speeds at very low powers is now being realized with the implementation of the Josephson Tunneling Logic (JTL).

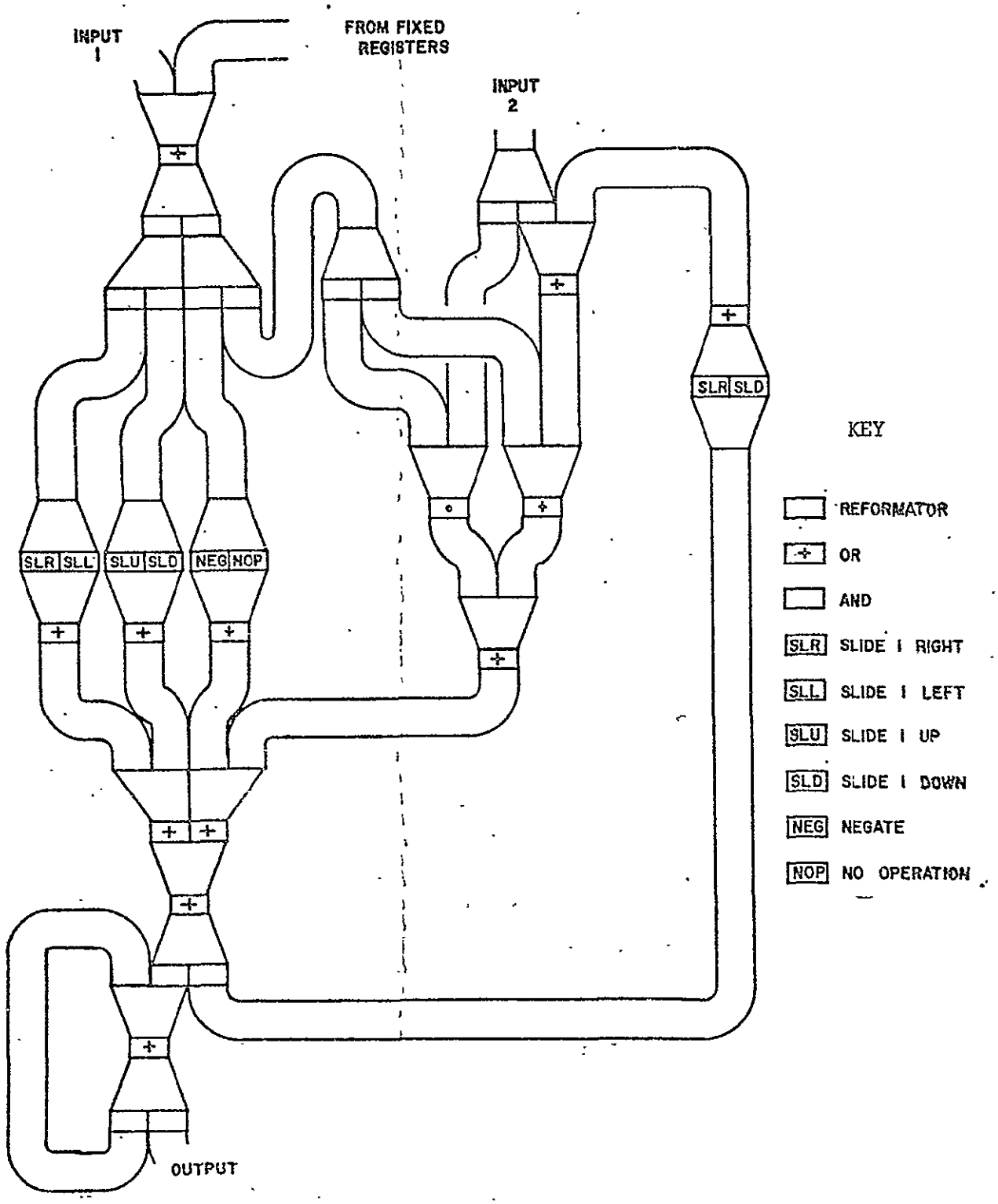


Figure 3.6.1(4) Organization of the tse Logical Operations Unit

The first measurements [33] of the speed-power product of this device showed an improvement of three orders of magnitude over that of the fastest semiconductor logic. This represents a speed capability of a single

processor which, if it were built as an on-board processor, could do the multispectral scanner processing job serially in real time.

As reported by W. Anacker [34], the basic gate is made up of two superconductors, a and b, with an oxide, c, between them as shown in Figure 3.3.2 (1). Control lines, d, e, and f, insulated from the gate, supply the input

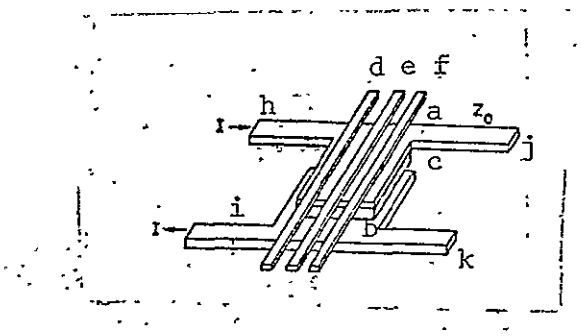


Figure 3.6.2(1) Basic Josephson Tunneling Gate

control signals. Each gate is supplied with a gate current, Ig, in and out through strip lines, h and i, respectively. For the JTL to be operative, this current must be present. With Ig present, and with no control-line currents present, maximum current flows through the gate, and the gate voltage is zero. With the application of a control current in one of the control lines and in the direction of the gate current, the gate current will switch to a low value and the voltage will rise to a quantum level. This sends a pulse down the output strip lines, j and k, which are terminated in their characteristic impedance with a resistance $2R_o$. The current in one of the output strip lines is used to control the next Josephson tunneling gate. Depending upon the ratio of the gate line width to the control line width, the gain, VIg/VIc can be made greater than unity. This allows the direct interconnection of units with no need for amplification of signals. The OR operation is formed by a current on any of the three control lines, d, e, and f. The AND operation is accomplished by using the opposite-sense currents in the control lines. The opposite-sense currents must be present in all three control lines for the devices to switch. Since the gate output strip lines are the control for the next gate, the use of one of the lines will set up an OR operation, and the use of the other will set up the AND operation. A NOT operation is also available, but it must be performed in a time sequence. Once a gate has switched, it can only be brought back to

the initial state by removing the gate current. Memory cells have also been built using three gates per cell. The direction of the current in the storage loop is a measure of the stored bit value. The memory is non-destructive readout. A switching time of 600 ps has been achieved [35], which indicates access rates of 1.5 GHz. The write cycle approached 1 ns.

The performance [33] of the JTL device is depicted in Figure 3.6.2(2). The measured speed-power product of several gates in series showed an average

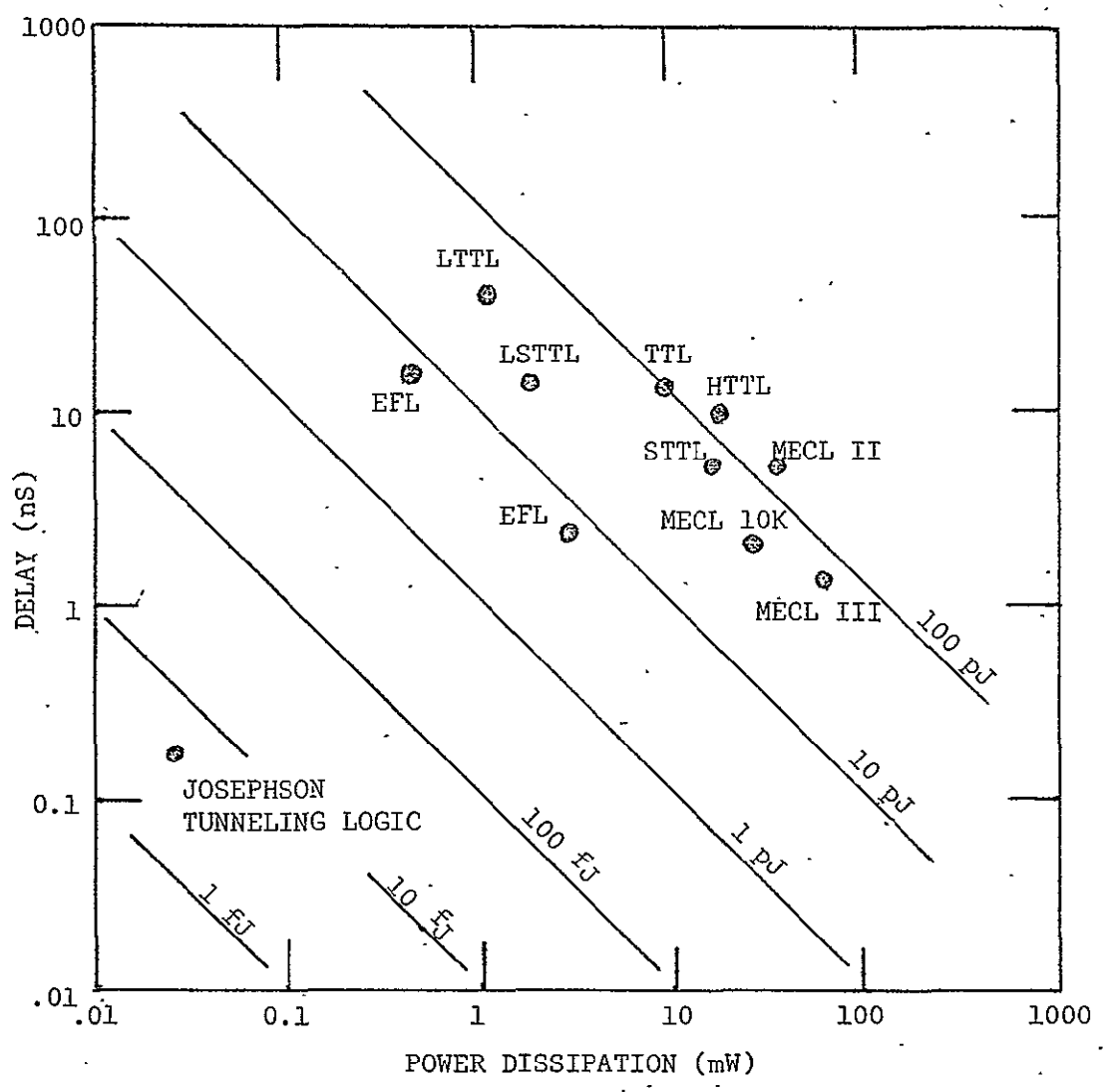


Figure 3.6.2(2) Performance Comparison for Logic Gates [33]

speed-power product of 5 femtojoules. This is three order of magnitude improvement over conventional semiconductor logic circuits.

The main disadvantage of the Josephson Tunneling devices is the fact that they have to be supercooled. It can be adapted, however, to space

flight use, since the space vacuum can be used to prevent heat losses into the supercooled system. Likewise, the space shuttle provides the capability of insuring that the cryogenic systems can be orbited at low cost.

A rule-of-thumb by J. K. Hulm [36] states that the improvement using cryogenics, to be worthwhile, must generally be a factor of at least 1,000. Using this rule, there will be no sudden rush to the use of Josephson Tunneling devices; however, the results of Figure 3.6.2(2) were obtained using non-optimum devices. If improvements in JTL technology are made faster than in semiconductor technology, then this sytem may have a practical application for space flight use. Because of the handicap of the low operating temperature, the devices will probably never enjoy mass production. Therefore, there will be substantial economic reasons to continue to use semiconductor logic circuits.

REFERENCES

1. Bright, J. R., Editor, Technological Forecasting for Industry and Government, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, pp. 146-7, 1968.
2. Buchholz, W., "A Selected Bibliography on Computer System Performance Evaluation," Computer Group News, vol. 2, no. 8, March 1969, pp. 21-22.
3. Miller, E. F., Jr., "Bibliography on Techniques of Computer Performance Analysis," COMPUTER, September/October 1972, pp. 39-47.
4. Skalansky, J., "An Evaluation of Several Two-Summand Binary Adders," IRE Trans. Electronic Computers, vol. EC-9, June 1960, pp. 213-226.
5. Householder, A. S., Principles of Numerical Analysis. New York: McGraw-Hill, 1953, p. 81.
6. Arbuckle, R. A., "Computer Analysis and Thruput Evaluations," Computers and Automation, vol. 15, no. 1, January 1966, pp. 12-15, 19.
7. Calingaert, P., "System Performance Evaluation: Survey and Appraisal," Comm. ACM, vol. 10, no. 1, January 1967, pp. 12-18.
8. Buchholz, W., "A Synthetic Job for Measuring System Performance," IBM System Journal, vol. 8, no. 4, 1969, pp. 309-318.
9. Ferrari, D., "Workload Characterization and Selection in Computer Performance Measurement," COMPUTER, July/August 1972, pp. 18-24.
10. Grochow, J. M., "Utility Functions for Time-Sharing System Performance Evaluation," COMPUTER, September/October 1972, pp. 16-19.
11. Arndt, F. R., and Oliver, G. M., "Hardware Monitoring of Real-Time Computer System Performance," COMPUTER, July/August 1972, pp. 25-29.
12. Winder, R. O., "A Data Base for Computer Performance Evaluation," COMPUTER, March 1973, pp. 25-29.

13. Cooperman, J. A., Lynch, H. W., and Tetzlaff, W. H., "SPG: An Effective Use of Performance and Usage Data," COMPUTER, September/October 1972, pp. 20-23.

14. Anacker, W., and Wang, C. P., "Performance Evaluation of Computing Systems with Memory Hierarchies," IEEE Trans. Computers, vol. EC-16, no. 6, December 1967, pp. 764-773.

15. Leis, C. T., "Minicomputer Hardware Architecture," PROC. IEEE, vol. 61, no. 11, November, 1973, pp. 1535-1538.

16. NASA SP-6009, "Forecasts and Appraisals for Management Evaluation," National Aeronautics and Space Administration, Washington, D. C., vols. 1 and 2.

17. Smith, J., York APL, Ryerson Polytechnical Institute, Toronto, 1972.

18. Hobbs and McLaughlin, "Minicomputer Survey," Datamation, pp. 50-61, July, 1974.

19. Lanford, H. W., Technological Forecasting Methodologies, American Management Association, 1969, 1972.

20. Hodges, D. A., and Allison, D. R., "Tutorial on LSI, Microcomputer Hardware, Software, and Systems," 10th IEEE Computer Society International Conference, San Francisco, CA, p. 12, Feb. 25-27, 1975.

21. Hartenstein, R. G., et al, "The Advanced On-Board Processor," NASA, Goddard Space Flight Center, Greenbelt, MD, X-715-71-451, October, 1971.

22. Unger, S. M., "A Computer Oriented Toward Spatial Problems," PROC. IRE, vol. 46, October 1958, pp. 1744-1750.

23. Slotnick, D. L., Borch, W. C., and McReynolds, R. C., "The Solomon Computer," AFIPS PROC. FJCC, vol. 22.

24. Barnes, G. H., et al, "The ILLIAC IV Computer," IEEE Trans. on Computers, vol. C17, August 1968, pp. 746-757.

25. Stroke, G. W., "Optical Computing," IEEE SPECTRUM, vol. 9, December 1972, pp. 24-41.

26. Strong, J. P., III, "Binary Image Operations Using Non-Coherent Optical Techniques," Pattern Recognition, Pergamon Press, vol. 5, 1973, pp. 51-60.

27. Strong, J. P., III, "Two-Dimensional Image Computation for Earth Resource Observation Program," Optical Computing Systems Workshop, Carnegie-Mellon University, September 1972, pp. 27-35.

28. Schaefer, D. H., and Strong, J. P. III, "tse Computer," X-943-75-14, NASA, GSFC, January 1975.

29. Schaefer, D. H., and Strong, J. P. III, "Two-Dimensional Radiant Energy Array Computers and Computing Devices," Patent Application Serial No. 468614, May 8, 1974.

30. Metcalf, A. G. and Bodenheimer, R. E., "Organization of Array Logic (tse) Devices for Executing a Parallel Counting Algorithm," Proc. 1975 Southeastern Symposium on System Theory, Auburn, Ala., March, 1975.

31. Buck, D. A., "The Cryotron - A Superconductive Computer Component," Proceedings of the IRE, April, 1956.

32. Josephson, B. D., "Possible New Effects in Superconductor Tunneling," Physics Letters, vol. 1, p. 251, 1962.

33. Herrell, D. J., "Femtojoule Josephson Tunneling Logic Gates," IEEE Journal of Solid-State Circuits, vol. SC-9, no. 5, October 1974.

34. Anacker, W., "Josephson Tunneling Devices - A New Technology With Potential for High-Performance Computers," 1972 Fall Joint Computer Conf., AFIPS conf. Proc., vol. 41, Montuale, N.J. AFIPS Press, 1972, pp. 1269-1278.

35. Zappe, H. H., "A Subnanosecond Josephson Tunneling Memory Cell with Nondestructive Readout," IEEE Journal of Solid-State Circuits, vol. SC-10, no. 1, February 1975.

36. Hulm, J. K., Cohen, M. H., "Superconductors in Technology," in Superconductivity in Science and Technology, University of Chicago Press, Chicago & London, 1968.

4 FEASIBILITY, TRADE-OFF, AND SENSITIVITY ANALYSIS

This section identifies the pertinent performance parameters, isolates the independent and necessary parameters, and relates these parameters to the system requirements for each of the user requirements discussed in the preceding sections. This will allow us to determine the feasibility of on-board processing for each user type in the 1980-1990 time frame and to perform a tradeoff analysis to determine the sensitivity of our results to each of the important system parameters.

The significant parameters related to the performance of the on-board processor are tabulated in Table 4(I).

Table 4(I) On-Board Processor Performance Parameters

R	data bit rate (bits/sec)
n	number of spectral bands (channels)
b	number of bits per resolution element per spectral band (bits)
S_w	swath width (meters)
R_L	resolution along a scan line (meters)
v	satellite ground track velocity (meters/sec)
R_p	resolution along scan path (meters)
M	number of classes
c	system cost (dollars)
p	system power (watts)
w	system weight (kilograms)
V	system volume (meters^3)
T	time (years)
n_c	number of components
N_A	number of additions
N_M	number of multiplications
r	pixel rate (resolution elements/sec)
λ_i	system constants (i=0,1)
k_i	system constants (i=0,2,3,4,5)
P_i	System Complexity function (i=1,2,3,4)

From Section 1.1.2.4, the incoming data bit rate is:

$$R = S_w/R_L \times v/R_p \times n \times b = r \times n \times b \qquad (4\text{-}1)$$

where

$$r = S_w/R_L \times v/R_p \qquad (4\text{-}2)$$

The pixel rate r, the number of ground resolution elements per second observed by the MSS, has been determined for each user requirement. One could choose any three independent parameters from the relation in equation (4-1). We chose the set R, n, and b to specify system performance. The number of classes M is also an independent parameter.

Since each of the processors investigated achieve their speed by distributing the processing load between many similar sub-processors, the cost, power, weight, and volume are all essentially proportional to the number of units required to satisfy the design. For example, a doubling of the data rate R requires doubling the number of parallel processors which corresponds to a doubling of the power, weight, etc. We now define a system complexity function P_i for each of the four processors i = 1,2,3,4 that relates the effects of the various parameter requirements. By proper choice of the system scale constants k_i the functions P_i are synonymous with "cost", "power", "weight", "volume", or "number of components".

4.1 COMPLEXITY FUNCTION DEPENDENCE ON M, n, r, b, and R.

In this section we define a complexity function for each of the four processor implementations described in Section 2 in terms of the parameters defined in Table 4(I). The dependence on the time T is discussed in Section 4.2.

4.1.1 The Microprocessor Maximum Likelihood Method (μPML)

The circuit complexity required for both the matrix equation (Section 2.2.8.1) and the reduced form equation (Section 2.2.8.3) is determined by the number of multiplication operations. For the classification of a single resolution element the minimum number of multiplications and additions in terms of the number of channels n and the number of classes M is given in Table 4.1.1(I).

Table 4.1.1(I) Mathematical Operations for Classification

# Operations	Reduced Form	Matrix Form
N_A	$M(3n + n^2)/2$	$M(n^2 + n + 2)$
N_M	$(n^2/2 + n/2)(M+1) + Mn$	$M(n^2 + n + 1)$

Regardless of how these operations are distributed throughout the processing system, for a given data rate the system complexity is proportional

to the processing time per pixel. Table 4.1.1(I) shows that for either formula the processing time per pixel is proportional to $M(n^2 + n)$ since multiplication time greatly exceeds addition time. A higher data rate can be achieved by increasing the number of processing elements over which the processing is distributed so that system complexity increases proportionally to the pixel rate.

The sensitivity to the number of bits per pixel depends on the multiplication method used in the processor. The example architectures in Section 2.2 use the Intel 8080 microprocessor unit but other manufacturers produce bitwise expandable units. For these units the complexity of each processing element is essentially proportional to the number of bits processed in parallel by the microprocessor array. If the multiplications are done by ROM table-look-up the amount of ROM required varies drastically with the number of bits in the words to be multiplied, e.g., to generate a (b+2)-bit product from two b-bit inputs requires

$$\text{ROM bits} = (b+2)\, 2^{2b} \qquad (4.1.1\text{-}1)$$

For $b > 6$ the processor complexity is dominated by the ROM requirements, and system complexity increases exponentially with the number of bits. On the other hand, if a PLA or hardware multiplier is used, the circuit complexity is proportional to the number of bits (at least for $b \leq 16$); we conclude that the system complexity is proportional to b.

The resulting system complexity function for the microprocessor-based maximum-likelihood classifier (μPML) is

$$P_1 = k_1\, M(n^2 + n) \cdot r \cdot b = k_1\, M(n + 1)\, R \qquad (4.1.1\text{-}2)$$

where k_1 is a system normalizing constant to be determined.

4.1.2 Hardware Maximum Likelihood Method (HML)

The primary difference between the hardware and microprocessor methods is that the multiplications and additions that were handled by software in the microprocessor method are here replaced one-for-one by hardware units. The relationship of total system complexity to the parameters M, n, b, and r remain essentially the same. However, the optimum pipeline processing unit has a data rate which exceeds that required by many users and system complexity can be reduced by time-sharing some of the processor arrays. Time-sharing requires additional circuitry to multiplex the shared processors and this increases the complexity function as the inverse of the product of the pixel rate r and the number of channels n. The additional hardware complexity for time sharing is proportional to the word length b.

The system complexity function for this method is, therefore,

$$P_2 = [k_2\ M(n+1)R + k_5\ b^2/R] \qquad (4.1.2\text{-}1)$$

For all systems of interest in this study $k_2 M(n+1)\ R >> k_5\ b^2/R$, i.e., for any realistic data rate, the reduction in complexity by time-sharing processing subsystems swamps the additional multiplexing complexity. Consequently, the complexity function for the hardware maximum likelihood processors is given by

$$P_2' = k_2\ M(n+1)\ R \qquad (4.1.2\text{-}2)$$

4.1.3 Microprocessor Table Look-Up (μPTLU)

Relative to the previous methods, the table look-up algorithm trades arithmetic hardware for memory storage. The system complexity depends almost entirely upon the memory required to store the feature space decision boundaries which depends largely upon the training sample statistics and the classification algorithm used. For most situations, the bulk of the memory is used to make comparisons in the last dimension searched. Boundaries must be stored for each point in the projection of the decision volume on to the space spanned by the previously searched dimensions. Examination of equations (1), (2), (3), and (4) of Section 2.2.8.2.1 indicates, at least for uncorrelated spectral measurements, that the memory requirement increases exponentially with the number of dimensions (channels). The empirical results of Table 2.2.8(XIII) for a typical agricultural classification problem may be used in conjunction with Eq. (2.2.8-7) to tabulate the storage requirements versus number of dimensions used, as illustrated in Table 4.1.3(I). F(n) is a simple mathematical function that is a reasonable model of the number of memory bytes required as a function of n.

Table 4.1.3(I) Table Look-Up Memory Requirements

No. of Spectral Bands (n)	1	2	3	4	5
Memory Requirements (Bytes)	18	342	4824	61,000	1,050,000
$F(n) = 1.38\ (15)^n$	21	311	4666	70,000	1,050,000

An increase in the number of channels n also causes a proportional increase in execution time since the additional dimensions must also be searched. To maintain the same pixel rate the system complexity is increased to handle the additional processing load.

If the number of bits per channel is increased while keeping the system dynamic range constant, the number of pointers which must be stored

is increased. The new memory requirements may be expressed in terms of the old memory requirements by the relation

$$N_{B\ new} = 4\ N_c[1+2^{\nabla b}\overline{N}_1 + 2^{(n-1)\ \nabla b}\overline{N}_{n-2} + (0.5)2^{n\nabla b}\overline{N}_{n-1}] \qquad (4.1.3\text{-}1)$$

Equation (4.1.3-1) is a modified version of Eq. (2.2.8-7), $\overline{N}_1, \ldots, \overline{N}_n$ are for the original system, $N_{B\ new}$ is for the bit-increased system, n is the number of channels, and ∇b is the word-size change in bits. In any practical system most of the memory is used to store the last dimension boundaries, so that the last term in equation (4.1.3-1) predominates.

Equation (4.1.3-1) reflects only the increase in memory words (bytes) required to store the pointers and boundaries. The word length in fact must also change, perhaps even past the byte level. Pointer length must be changed to handle the added memory; increasing the word length by ∇b will handle both the boundary and pointer requirements.

Increases in the pixel rate are handled by multiplexing identical systems so that the total system complexity is directly proportional to the pixel rate r. Changing the number of classes M changes the total execution time proportionately regardless of the distribution of the processing load. The number of parallel systems must therefore be increased proportionately to maintain the same pixel throughput rate.

Taking all of the above factors into account the system complexity function for microprocessor table look-up (μPTL) is given by

$$P_3 = k_3\ M\ r\ b\ n\ \lambda_1^{nb} = k_3\ M\ R\ \lambda_1^{nb} \qquad (4.1.3\text{-}2)$$

where k_3 and λ_1 are system constants.

4.1.4 Hardware Table Look-Up (HTLU)

The system complexity for the microprocessor system is equally valid for the hardware approach except for the normalizing constant, i.e., the complexity function for hardware table look-up (HTLU) is given by

$$P_4 = k_4\ M\ R\ \lambda_1^{nb} \qquad (4.1.4\text{-}1)$$

4.2 COMPLEXITY FUNCTION DEPENDENCE ON TIME

It was shown in Section 3 that computer cycle times, memory access times, and logic propagation delays in newly available equipment decrease exponentially with time. To project the complexity functions into the future, therefore, each complexity function requires a multiplicative

factor of the form

$$f(t) = \lambda_o^{-T} \qquad (4.2\text{-}1)$$

where λ_o is the normalized yearly decrease in processing delays and T is the number of years into the future. It is implicitly assumed that λ_o is approach-independent, i.e., the rate of these decreases will occur equally for each of the four processor architectures. This assumption is reasonable since each of the four processor architectures utilize the same semiconductor technology.

We must also take into account the fact that the components increase in complexity and computational power with time. The system designer deals with these components on a macroscopic level and the design complexity which he must face is essentially independent of the microscopic complexity; it depends only on the total number of components used in the design and not on whether they are SSI, MSI, or LSI chips. Figure 3.4(1) illustrates the evolution of the data processing systems for the IMP series of spacecraft over the 10-year period 1964-1974. This is a spacecraft data handling and telemetry system, but the design techniques are comparable to our on-board processor designs and, indeed, the later systems utilized a computer-like processor. All designs were near the feasibility limits for a reliable spacecraft system at the time of design. Note that the number of components stayed nearly constant between 750 and 1000, while the gate count increased exponentially (two orders of magnitude in less than 10 years). Figure 3.2.2 (1) gives a semiconductor manufacturer's projection of the number of components on an LSI chip for 1975-1985. The projection indicates a tenfold increase over the next ten years.

If the per IC power dissipation, cost, weight, design effort, etc. remain fairly constant over this period as they have over the last 10 years, then the complexity functions P should reflect the savings due to increases in the computational power of the components, i.e., the increase in the equivalent number of gates per IC. This requires an additional exponential term in each of the system complexity functions. Alternatively, the value of λ_o in Eq. (4.2-1) can be modified to accommodate this additional factor. The resulting complexity functions are summarized in Table 4.2(I).

4.3 EVALUATION OF SYSTEM CONSTANTS

In Sections 4.1 and 4.2 we determined the manner in which the complexity functions for each of the four processor architectures depend on the system

Table 4.2(I) On-Board Processor Complexity Functions

Processor	Complexity Function
Microprocessor Maximum Likelihood (μPML)	$P_1 = k_1\ M(n+1)\ R\ \lambda_o^{-T}$
Hardware Maximum Likelihood (HML)	$P_2 = k_2\ M(n+1)\ R\ \lambda_o^{-T}$
Microprocessor Table Look-Up (μPTLU)	$P_3 = k_3\ M\ R\ \lambda_1^{nb}\ \lambda_o^{-T}$
Hardware Table Look-Up (HTLU)	$P_4 = k_4\ M\ R\ \lambda_1^{nb}\ \lambda_o^{-T}$

parameters M, n, b, r, R, and T. The remaining problem is to determine the system constants λ_o, λ_1, k_1, k_2, k_3, and k_4.

4.3.1 Evaluation of λ_o

In Section 4.2 we described two factors that contribute to the exponential term λ_o^{-T}; (1) the exponential increase in processor speed due to decreases in computer cycle times, memory access times, and propagation delays; and (2) the exponential increase in the number of gates per IC.

From the technology forecast results of Section 3 we conclude that computer cycle times, memory access times, and propagation delays will decrease by one order of magnitude from 1974 to 1984. We further conclude that the equivalent number of gates per IC chip will increase between one and two orders of magnitude during the same period; we choose to use the more conservative number. Consequently, we choose λ_o to allow two order of magnitude improvement in system complexity (one order of magnitude for speed and one for the number of gates per IC, i.e., we set

$$\lambda_o^{-10} = .01 \qquad (4.3.1\text{-}1)$$

so that

$$\lambda_o = 1.58 \qquad (4.3.1\text{-}2)$$

4.3.2 Evaluation of λ_1

λ_1 can be accurately evaluated from the empirical results of Table 2.2.8(XIII) and Table 4.1.3(I). That is, for table look-up crop classification for agricultural data with b = 6 bit data words we can equate the λ_1^{nb} term in Eqs. (4.3.3-2) and (4.1.4-1) with the F(n) function of Table 4.1.3 (I) to obtain

$$\lambda_1^{nb} = (15)^n \qquad (4.3.2\text{-}1)$$

The scale factor 1.38 in F(n) can be absorbed into the system constants k_3 and k_4. Setting b = 6 in Eq. (4.3.2-1) we find

$$\lambda_1 = 1.6 \qquad (4.3.2\text{-}2)$$

Strictly speaking, this value is valid for b = 6 and is an average value for agricultural data. However, almost all users surveyed in the user requirement survey summarized in Section 1 indicated b = 6 bit data words and the few exceptional cases did not deviate significantly from b = 6. (The maximum mentioned was b = 7.) Empirical studies of the table look-up algorithm for other than agricultural data are not available. However, the decision boundaries for other data have the same general characteristics as for agricultural data, i.e., they can be modeled reasonably well by second order polynomials as attested to by the fact that the Gaussian maximum likelihood classifier works well for other than agricultural data. Consequently, the memory requirements for other than agricultural data are not expected to vary significantly from the average for agricultural data. We conclude that $\lambda_1 = 1.6$ is a reasonable value for the purposes of this study.

4.3.3 Evaluation of k_1, k_2, k_3, k_4

The complexity functions P_i are now defined except for the normalizing constants k_i. These normalizing constants can be obtained from the data points resulting from the detailed processor designs described in Section 2.2.8. In that section we determined the number of IC packages, the power requirements, the weight, the volume, and the cost of each of the four processor architectures for the base line system using 1975 technology. These results along with the definition of the baseline system are summarized in Table 4.3.3(I)

Table 4.3.3(I) Complexity Functions for the Baseline System Described in Section 1.4; i.e., n=4 Spectral Bands M=12 Classes, b=6 Bits, R=31.2M Bits/Sec and T=0 (1975 Technology)

Processor	Complexity Function	#IC's (K)	Power (Kw)	Volume (m^3)	Weight (Kg)	Cost (K$)
μPML	P_1	30.0	3.5	.50	250	1700
HML	P_2	.6	.3	.03	15	15
μPTLU	P_3	90.0	45.0	.75	340	2250
HTLU	P_4	1.2	.6	.10	36	32

By setting $\lambda_o = 1.58$, $\lambda_1 = 1.6$, n = 4, M = 12, b = 6, R = 31.2×10^6 and T = 0 in each of the equations listed in Table 4.2(I) and equating to the values of P_i in Table 4.3.3(I) we determine the scale factors k_1, k_2, k_3, and k_4 for each of the performance measure listed in Table 4.3.3(I). The resulting scale factors for each performance measure are presented in Table 4.3.3(II).

Table 4.3.3(II) Scale Factors for the Complexity Functions of Table 4.2(I) for Each Performance Measure.

Scale Factor	#IC's	Power(w)	Volume(m^3)	Weight(kg)	Cost
k_1	1.6×10^{-5}	1.8×10^{-6}	2.67×10^{-10}	1.34×10^{-7}	9.08×10^{-4}
k_2	3.21×10^{-7}	1.6×10^{-7}	1.6×10^{-11}	8.01×10^{-9}	8.01×10^{-6}
k_3	3.03×10^{-9}	1.52×10^{-9}	2.53×10^{-14}	1.15×10^{-11}	7.59×10^{-8}
k_4	4.05×10^{-11}	2.02×10^{-11}	3.37×10^{-15}	1.21×10^{-12}	1.08×10^{-9}

4.4 SENSITIVITY ANALYSIS

The sensitivity of the complexity functions to the various sytem parameters is illustrated by the parametric curves in Figures 4.4(1), 4.4(2), and 4.4(3). These curves were obtained by setting all system parameters to their baseline values n = 4 spectral bands, M = 12 classes, b = 6 bits, R = 31.2×10^6 megabits, and T = 0 except for the parameters listed in the figures which were then varied from their baseline values. Consequently, these curves illustrate the deviations in the complexity functions as the system parameters vary from their baseline values.

All of complexity functions involve the terms $R \lambda_o^{-T}$, i.e., all four complexity functions increase linearly with the bit rate R and decrease exponentially with the time T. Figure 4.4(1) illustrates the behavior of the complexity functions as a function of T as R is varied in multiples of two from 7.5 megabits/sec to 120 megabits/sec. The normalized complexity factor, valid for all four processor architectures, is plotted on a linear scale.

The ML and TLU processors differ in their dependence on the number of spectral bands n and the data word length b. Recall that $R = r \cdot n \cdot b$ so that the ML processor complexity functions increase as bn(n+1). The normalized ML complexity function is presented in a linear scale in Figure 4.4(2) as a function of n between 1 and 20 for b = 3, 4, ..., 8. Doubling

b doubles the complexity function whereas the dependence on n is essentially as n^2.

The normalized TLU complexity function is plotted on a logarithmic scale in Figure 4.4(3) as a function of b and n. Except for very small

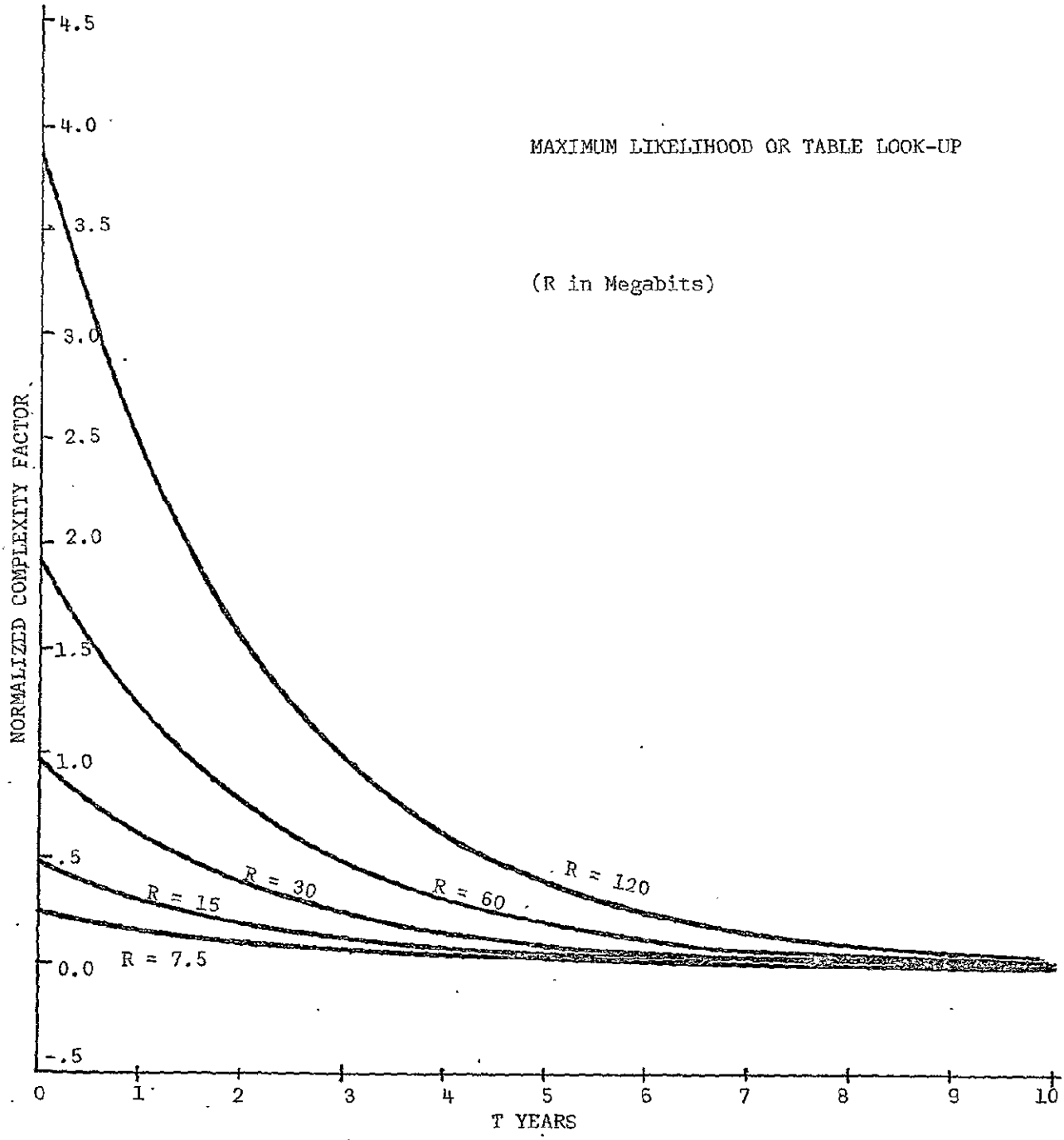


Figure 4.4(1) Complexity Function Sensitivity to R and T (all Processors)

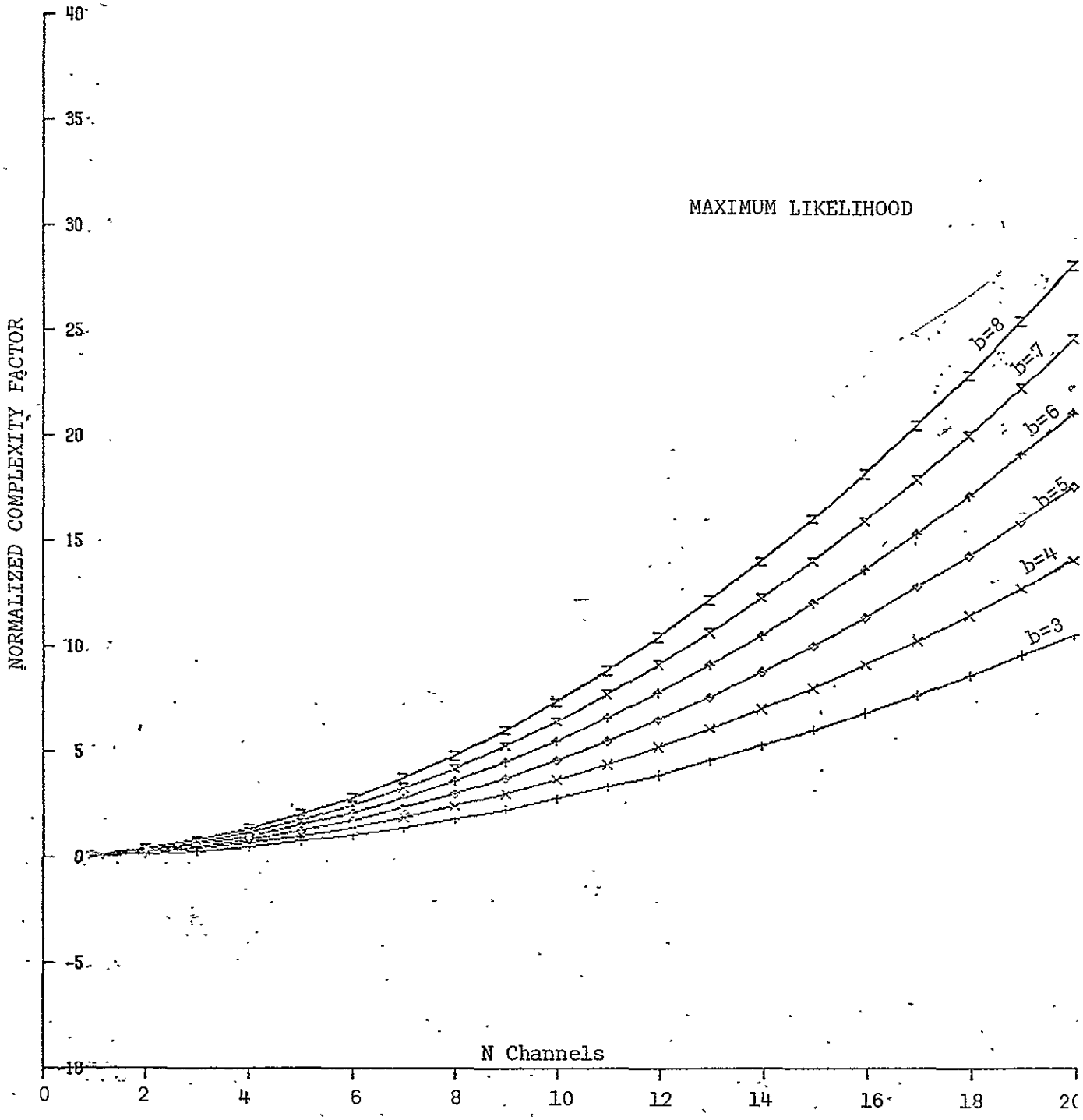


Figure 4.4(2) Complexity Function Sensitivity to b and n (ML Processors)

values of b and n the dependence is essentially exponential in both b and n.

FEASIBILITY ANALYSIS

The feasibility of on-board processing for the applications areas defined in Section 1 depends on the particular set of system parameters (user

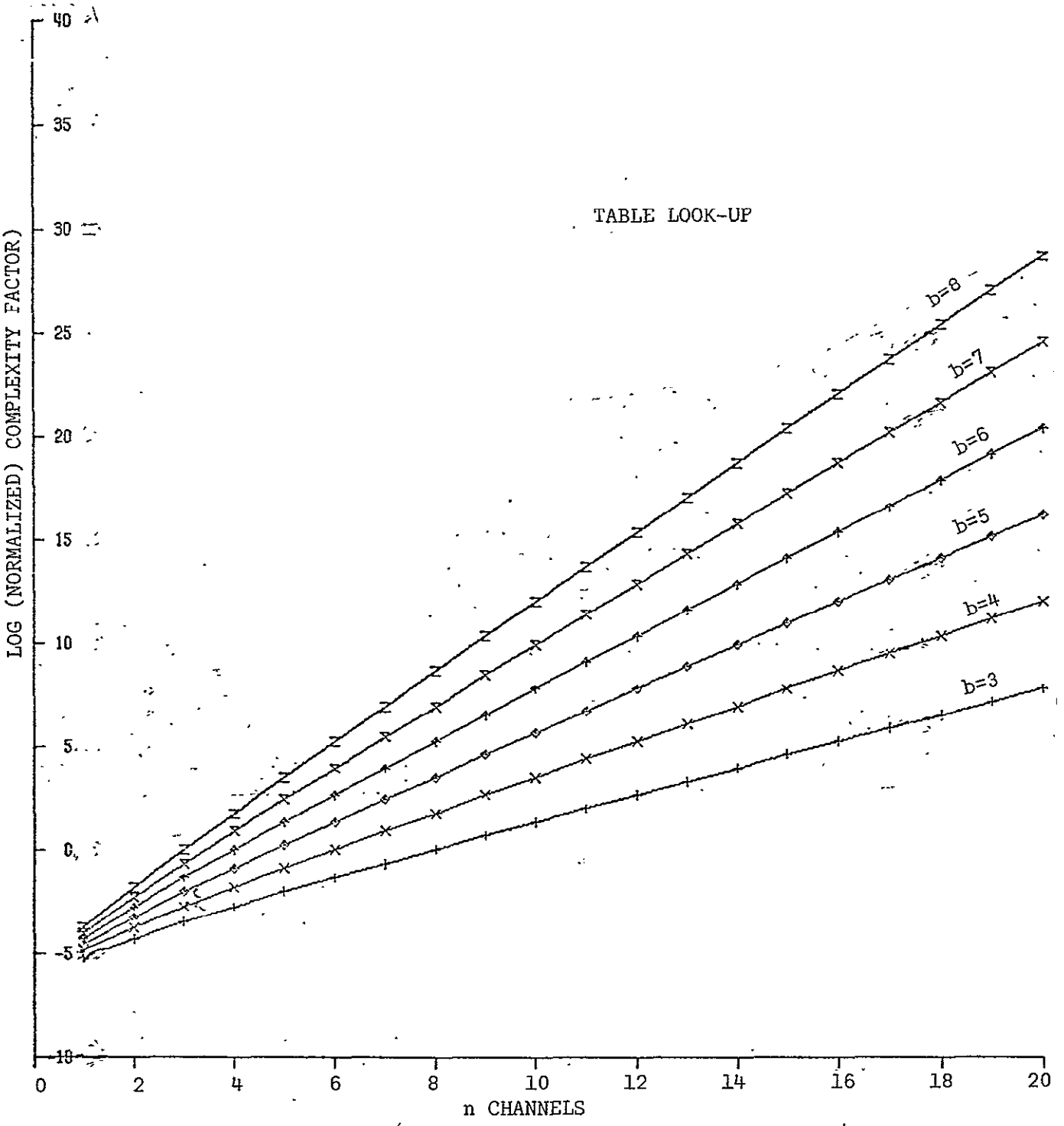


Figure 4.4(3) Complexity Function Sensitivity to b and n (TLU Processor)

requirements) n, M, b, R, the processor, the time T, and the definition of "feasible".

4.4.1 What is Feasible?

For a particular processor to be "feasible" at a particular point in time requires that it meet certain constraints on performance, complexity,

volume, weight, power, cost, reliability, environment, etc. Each of the four system architectures meets the performance constraint since each was designed to accomplish the required task. All four processors use standard integrated circuit technology and meet the data throughput rates by adding more components (IC's) in parallel. The volume, weight, and power dissipation of integrated circuits can be kept within limits by simply keeping the number of integrated circuits within limits. The radiation, temperature, and other environmental constraints can be met by each processor as discussed in Section 2. The limiting factors are cost and reliability which can also be kept within bounds by imposing a constraint on the number of components. Consequently, we conclude that on-board processing using a particular processor is feasible provided we constrain the number of IC's in the processor to a reasonable number.

4.4.2 Feasibility Curves

In this section we plot the number of IC's as a function of the time T for each of the four processors for each user application described in Section 1.1. Each figure contains two graphs. The first graph is for the maximum data-rate requirements listed in Table 1.1.2(I) and the second graph is for the minimum requirements listed in the same Table. Each curve is labeled with the user application symbol described in Section 1.1 and Table 1.1.2(II). The time scale shows launch date and allows for a five year lag between conception and launch. Other time lags can be handled simply by adding or subtracting the appropriate number of years from the time axis.

The parts cost of the on-board processor increases linearly as the number of IC's. The costs associated with check out increase as the square of the number of IC's. Limiting the number of IC's in the on-board processor to about 1000 appears to satisfy all constraints, i.e., cost is reasonable relative to the total system cost (launch, sensors, telemetry, etc.), reliability is pushing the limits of present day technology as discussed in Section 3, while volume, weight, and power dissipation do not appear to present serious difficulties. This is discussed in greater detail in Section 3.4. A horizontal line is drawn on each graph at 500 IC's, 1,000 IC's (which corresponds to our suggestion for a reasonable (feasible) number of IC's), and 2,000 IC's.

Some applications are feasible in 1980, some are not feasible by 1990, and some become feasible at some time between 1980 and 1990, depending on

the type of processor. For example, some of the applications requiring many spectral bands (e.g., oceanography) are never feasible using the TLU processor since the complexity (number of IC's) goes up exponentially with the number of spectral bands. The TLU processors can handle 4 or 5 spectral bands at the most.

A few of the curves indicate a fraction of an IC. The reason is that an IC is considered equivalent to a certain number of gates that increases an order of magnitude over the ten year period. A fraction of an IC (log # IC's $\leq$ 0) means that the on-board processing task can be accomplished with fewer gates than available in the "average" IC of that year.

If we accept the 1000 IC definition of feasibility, then we can list the year in which each user application first becomes feasible. Table 4.4.3 (I) indicates the first year that each processor can accomplish each of the applications listed in Table 1.1.2(I) for the worst case (maximum requirements) situation. The symbol N indicates not on or before 1990. All but one of the maximum requirements can be met on or before 1986, and about half can be met in 1980 by the HML processor.

Table 4.4.3(II) is similar to Table 4.4.3(I) except that the minimum requirements for each application were used. The HML processor can meet all but one requirement in 1980 the μPML processor can meet all but one requirement by 1990. The HTLU processor can meet most requirements in 1980, but a few cannot be met by 1990 due to the number of spectral bands.

Tables 4.4.3(III) and 4.4.3(IV) are similar to Tables 4.4.3(I) and 4.4.3(II) except that 500 IC's was used as the definition of feasible. Similarly, Tables 4.4.3(V) and 4.4.3(VI) are for the case of 2000 IC's. A factor of 1.58 in the number of IC's corresponds to one year in the date the processor becomes feasible. Multiplying the number of IC's by 1.58 makes the processor feasible one year earlier. Or, stated the other way, waiting one year means the processor can be designed with 1/1.58 = .67 as many IC's.

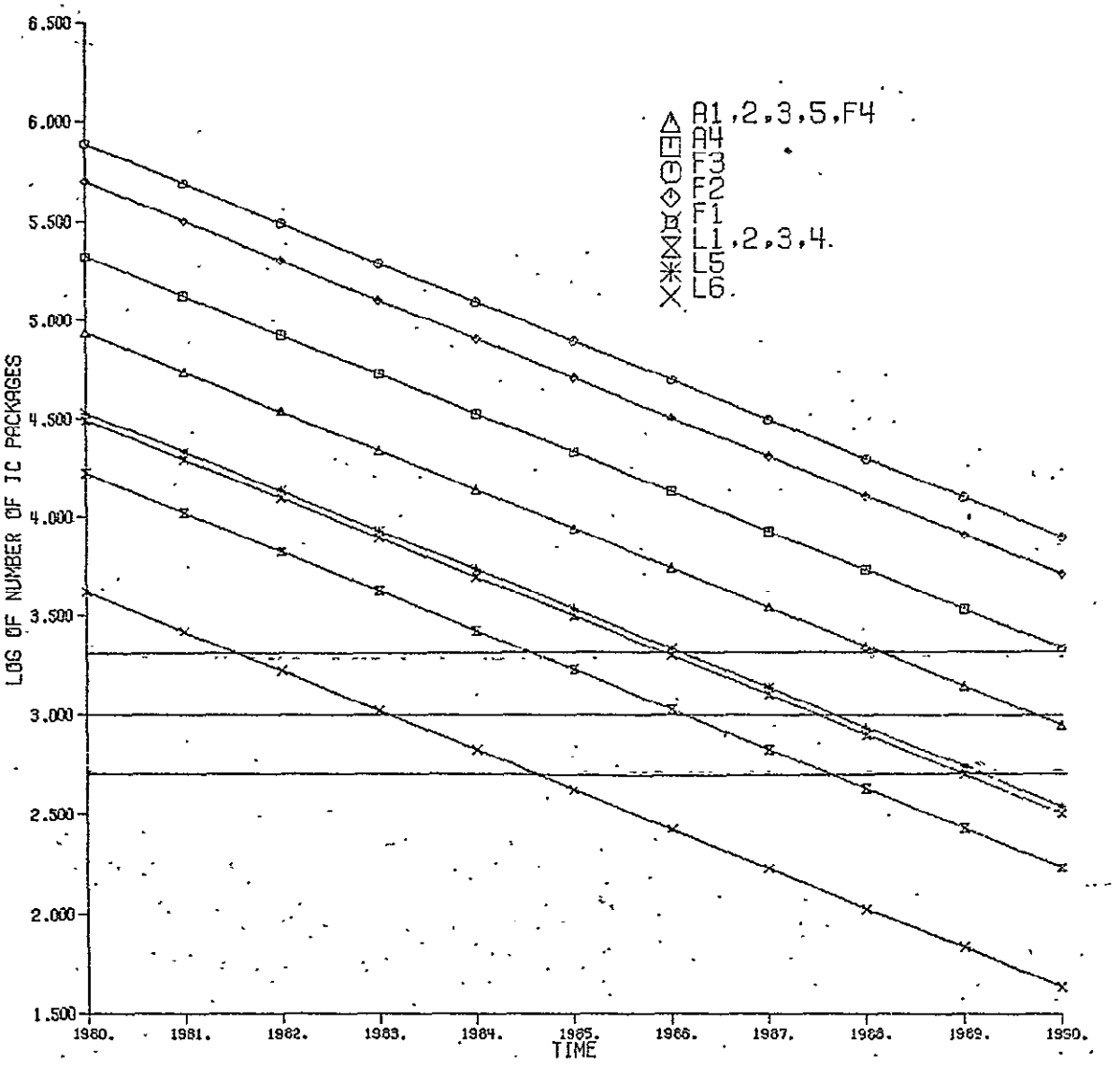


Figure 4.4.2(1)A Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

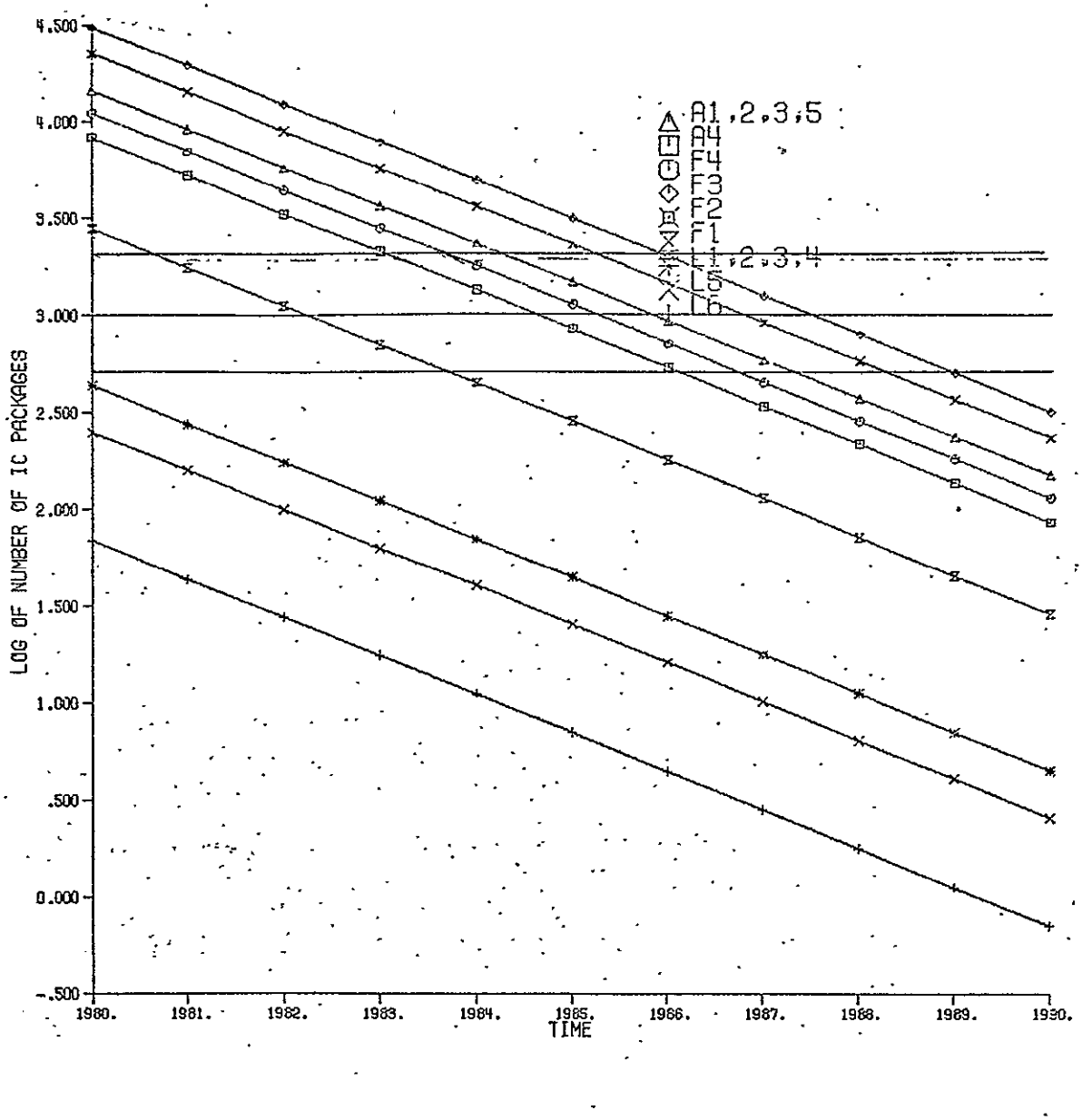


Figure 4.4.2(1)B Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

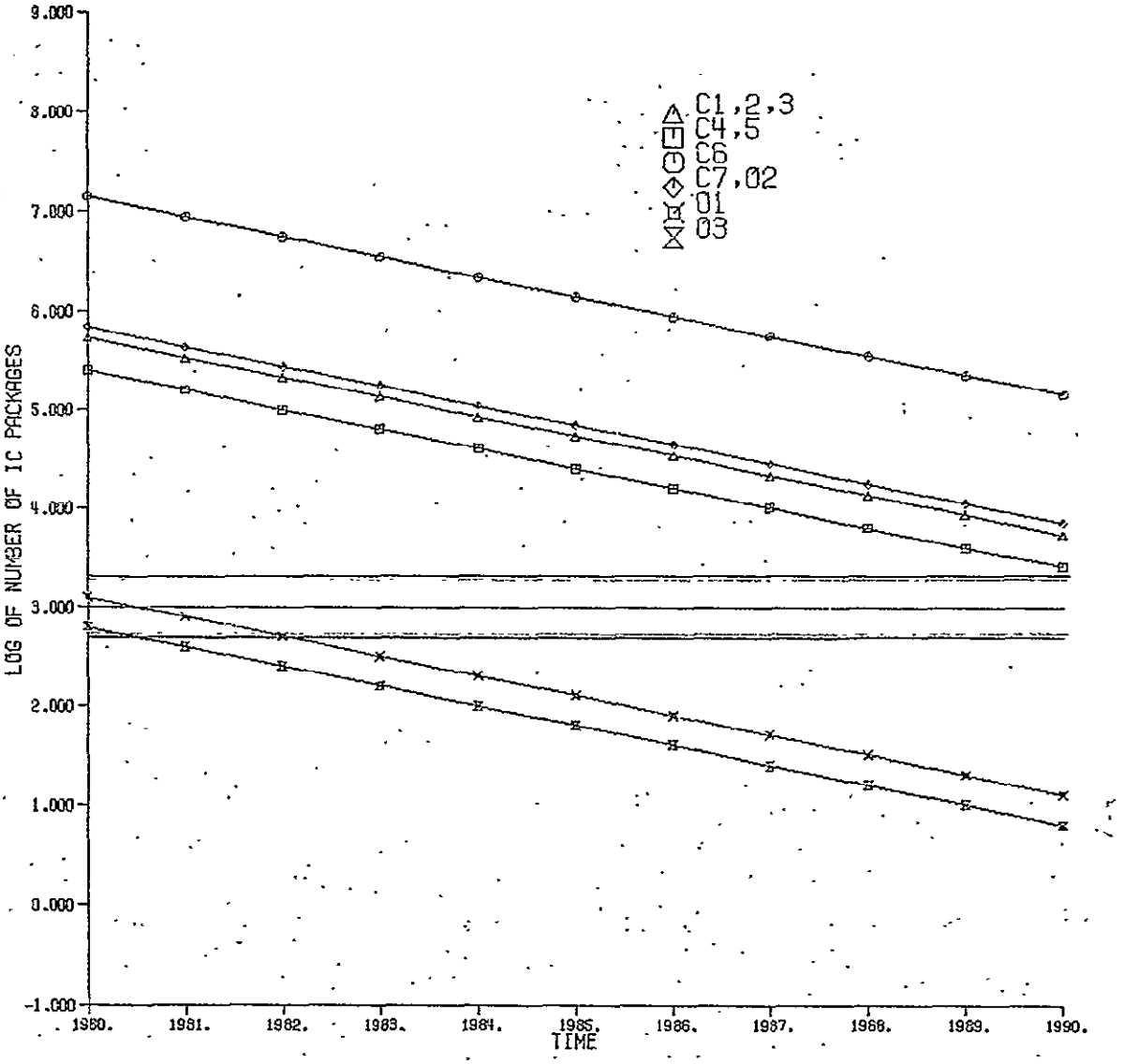


Figure 4.4.2(2)A Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I): Applications: Coastal-Zone Studies and Global Oceanography.

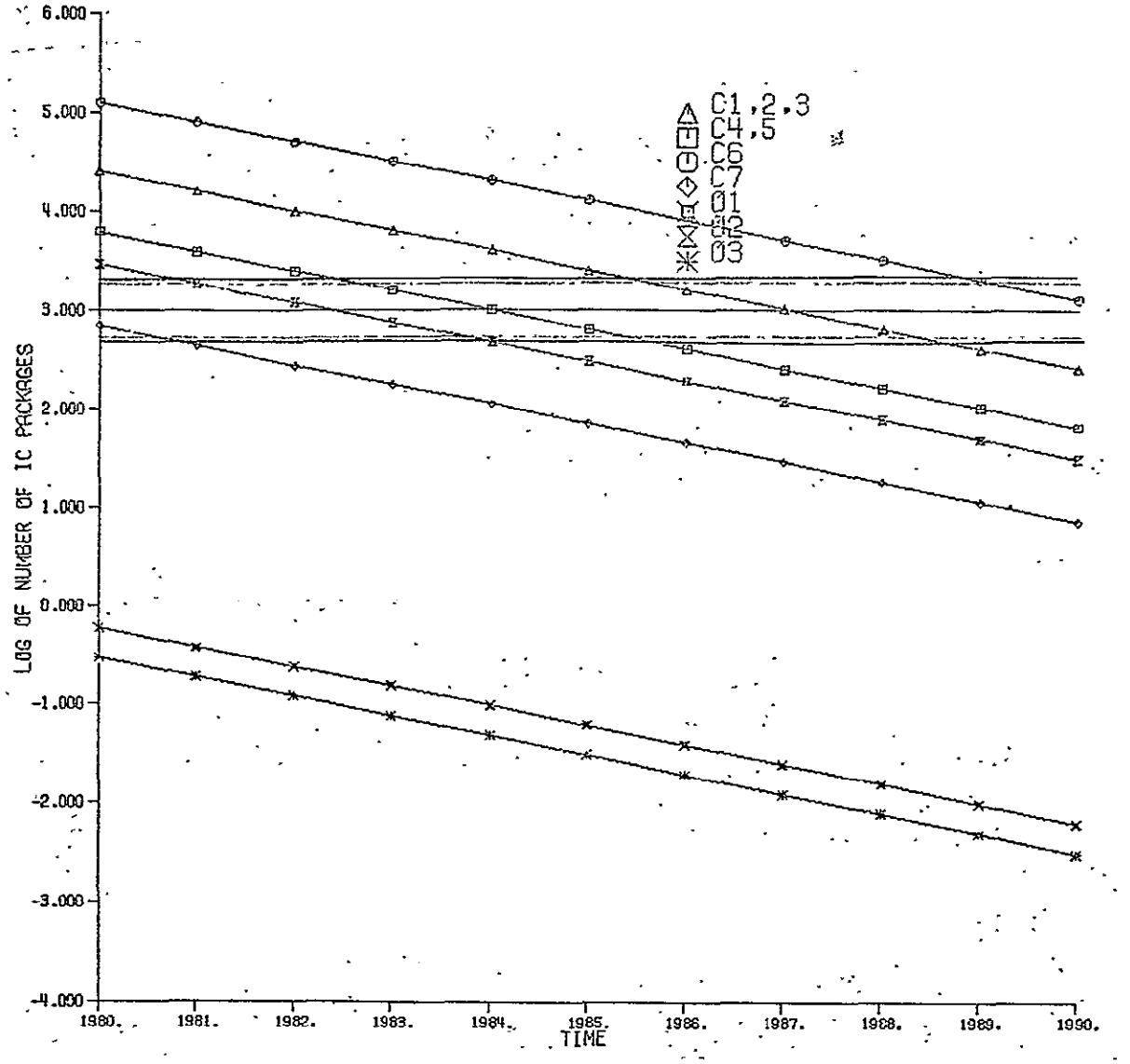


Figure 4.4.2(2)B Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Coastal-Zone Studies and Global Oceanography.

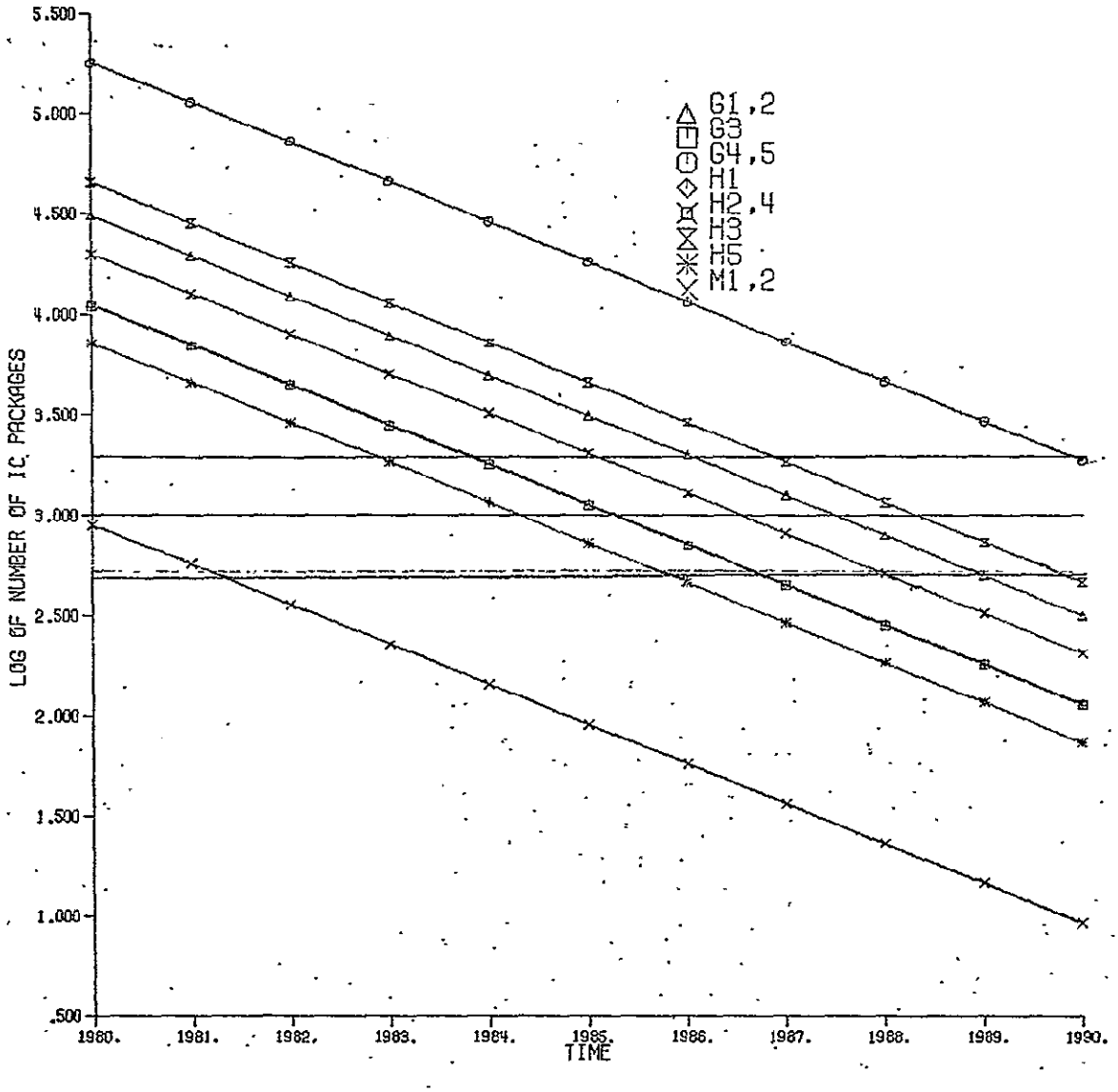


Figure 4.4.2(3)A Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(1): Applications: Geography, Hydrology and Meteorology.

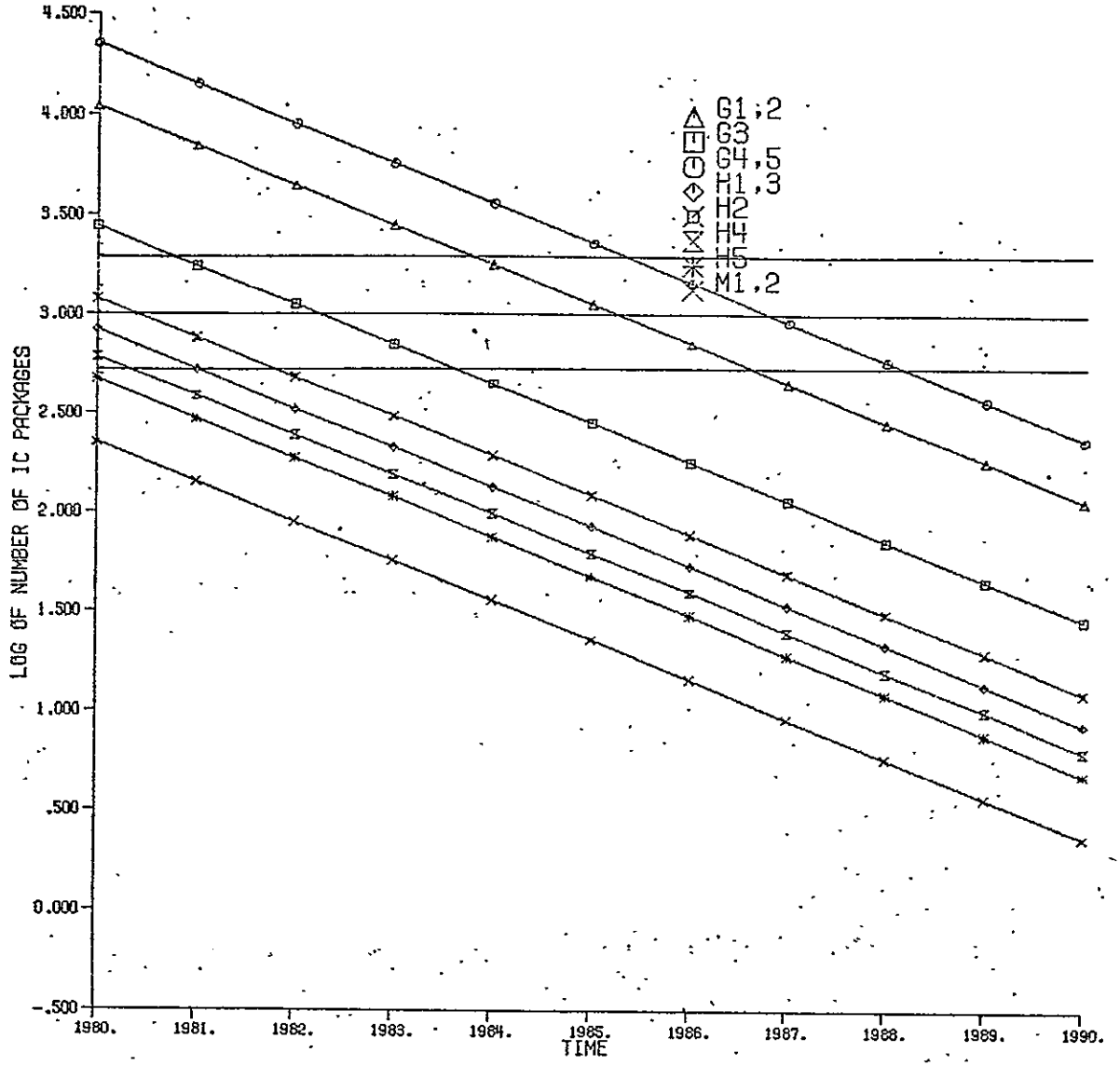


Figure 4.4.2(3)B Number of IC's vs. Launch Date to Implement the Microprocessor Maximum Likelihood (μPML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

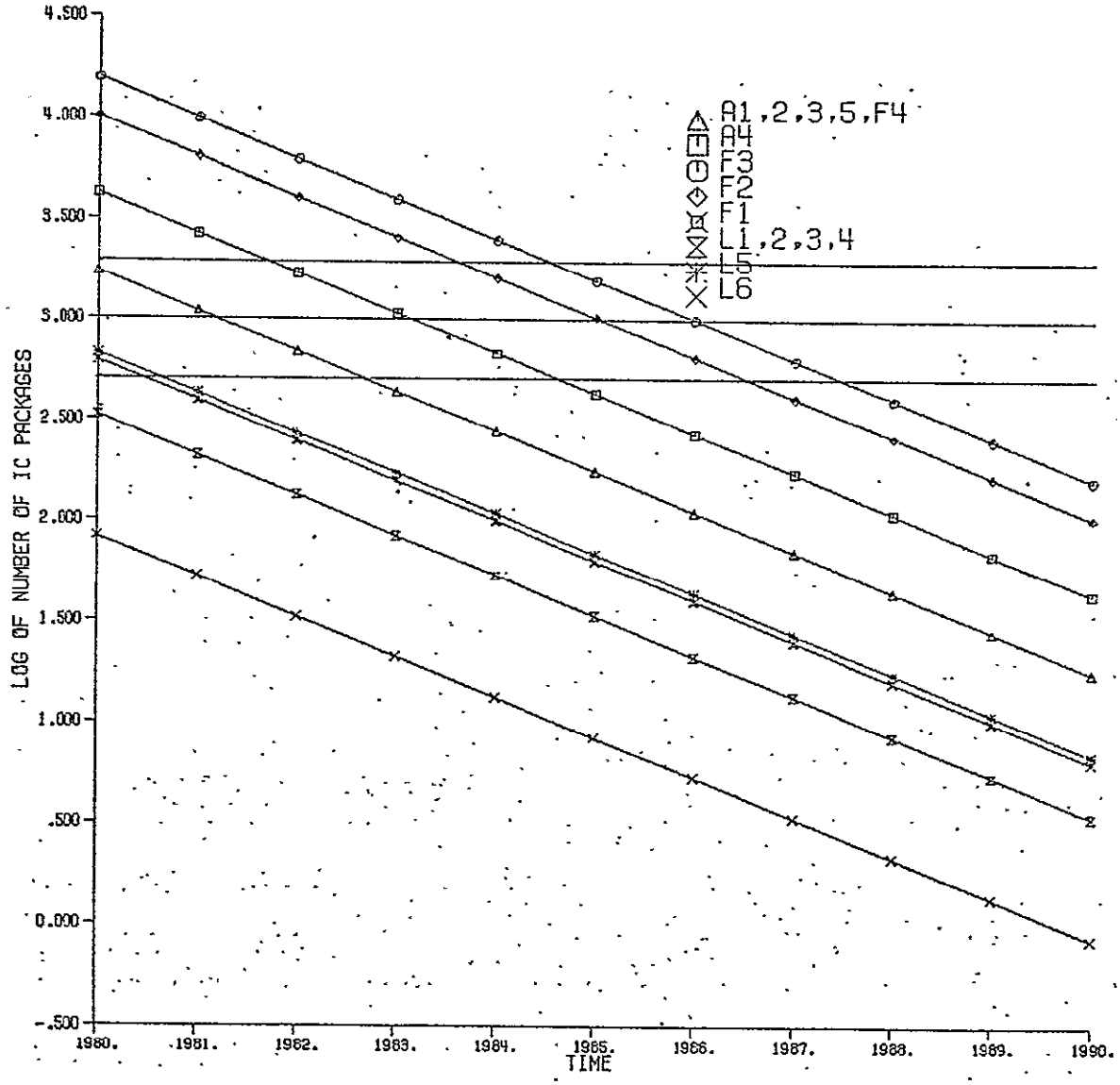


Figure 4.4.2(4)A Number of IC's vs. Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

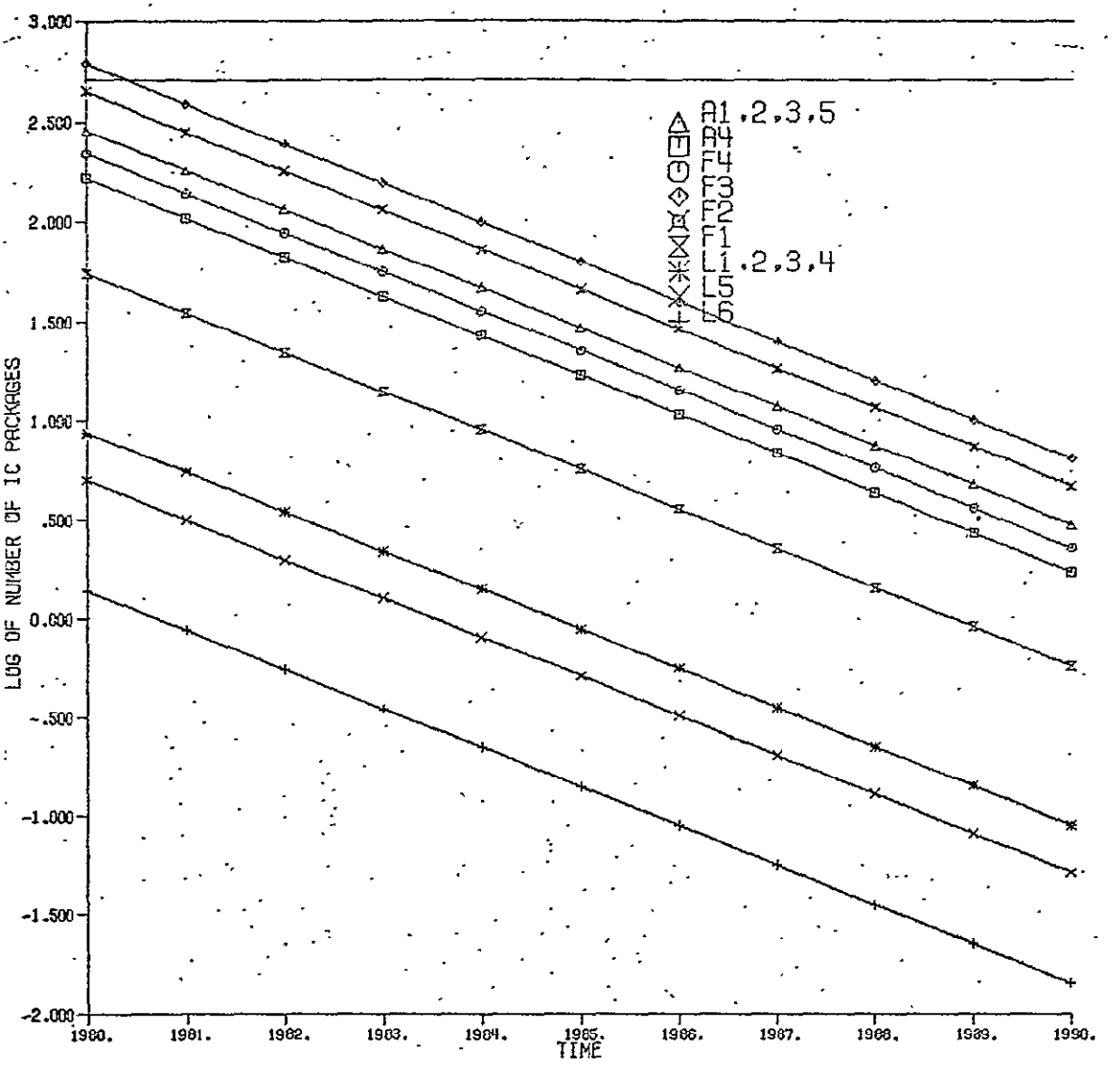


Figure 4.4.2(4)B Number of IC's vs. Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

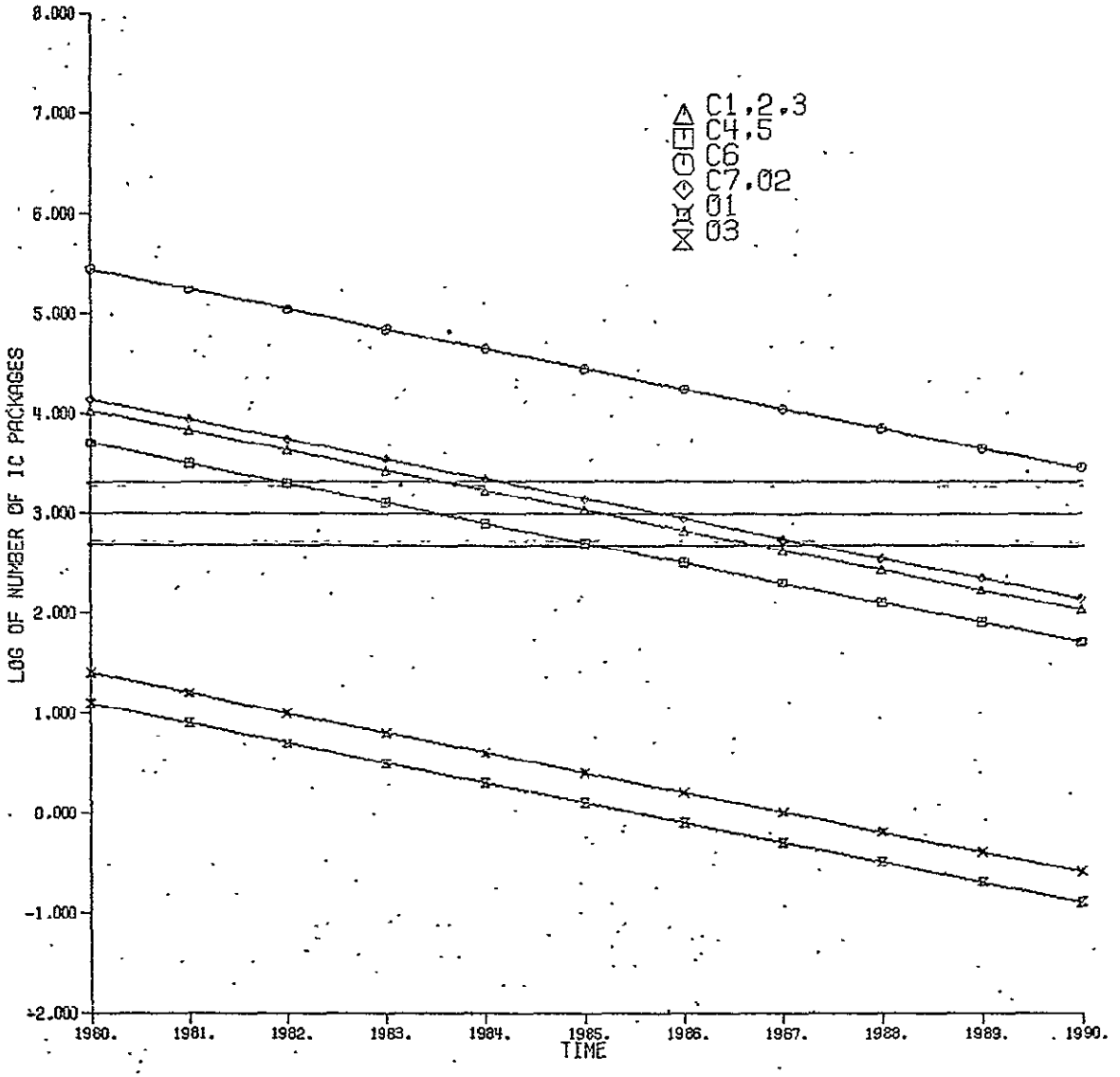


Figure 4.4.2(5)A Number of IC's vs. Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Coastal-Zone Studies and Global Oceanography.

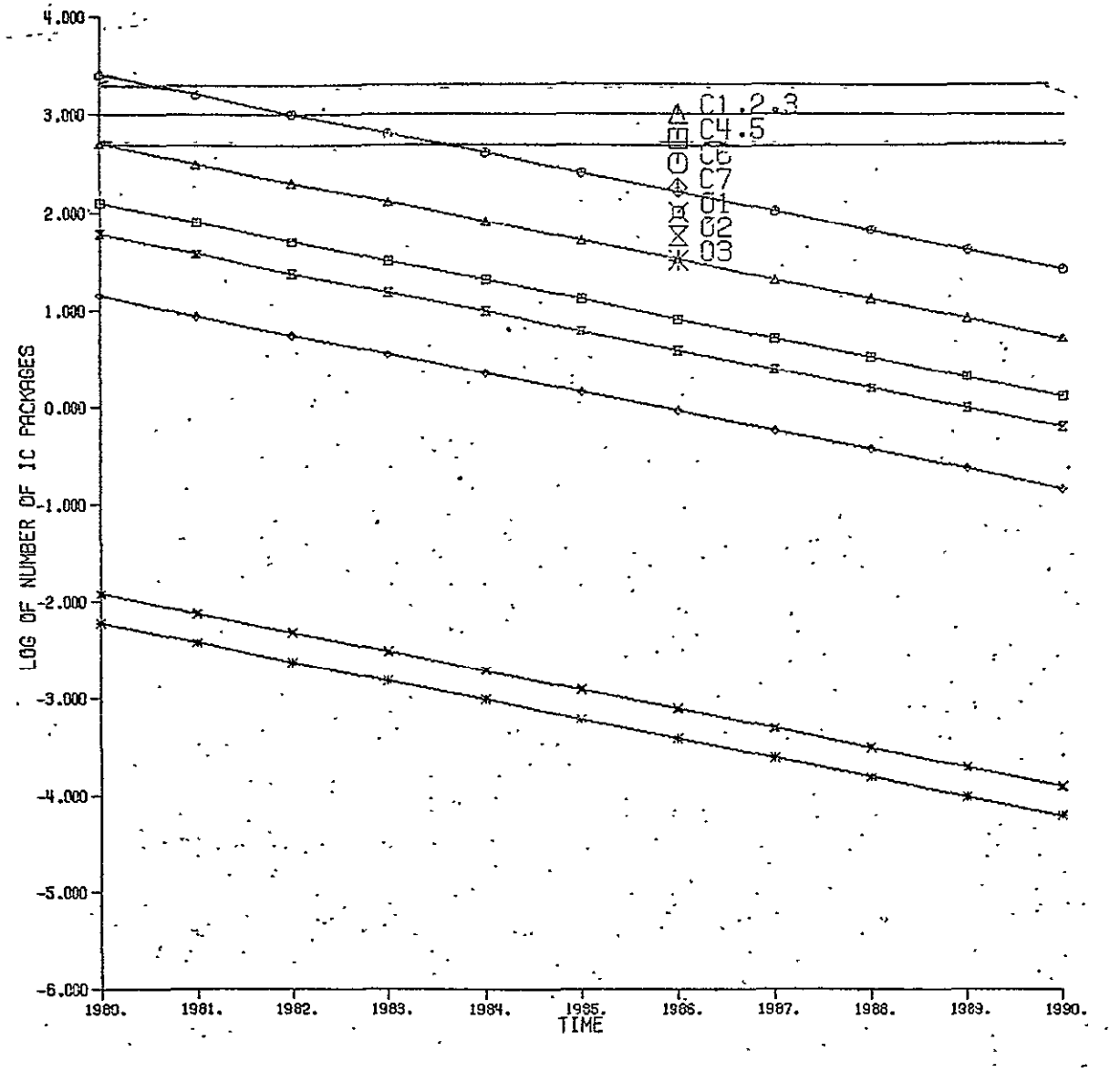


Figure 4.4.2(5)B Number of IC's vs. Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I): Applications: Coastal-Zone Studies and Global Oceanography.

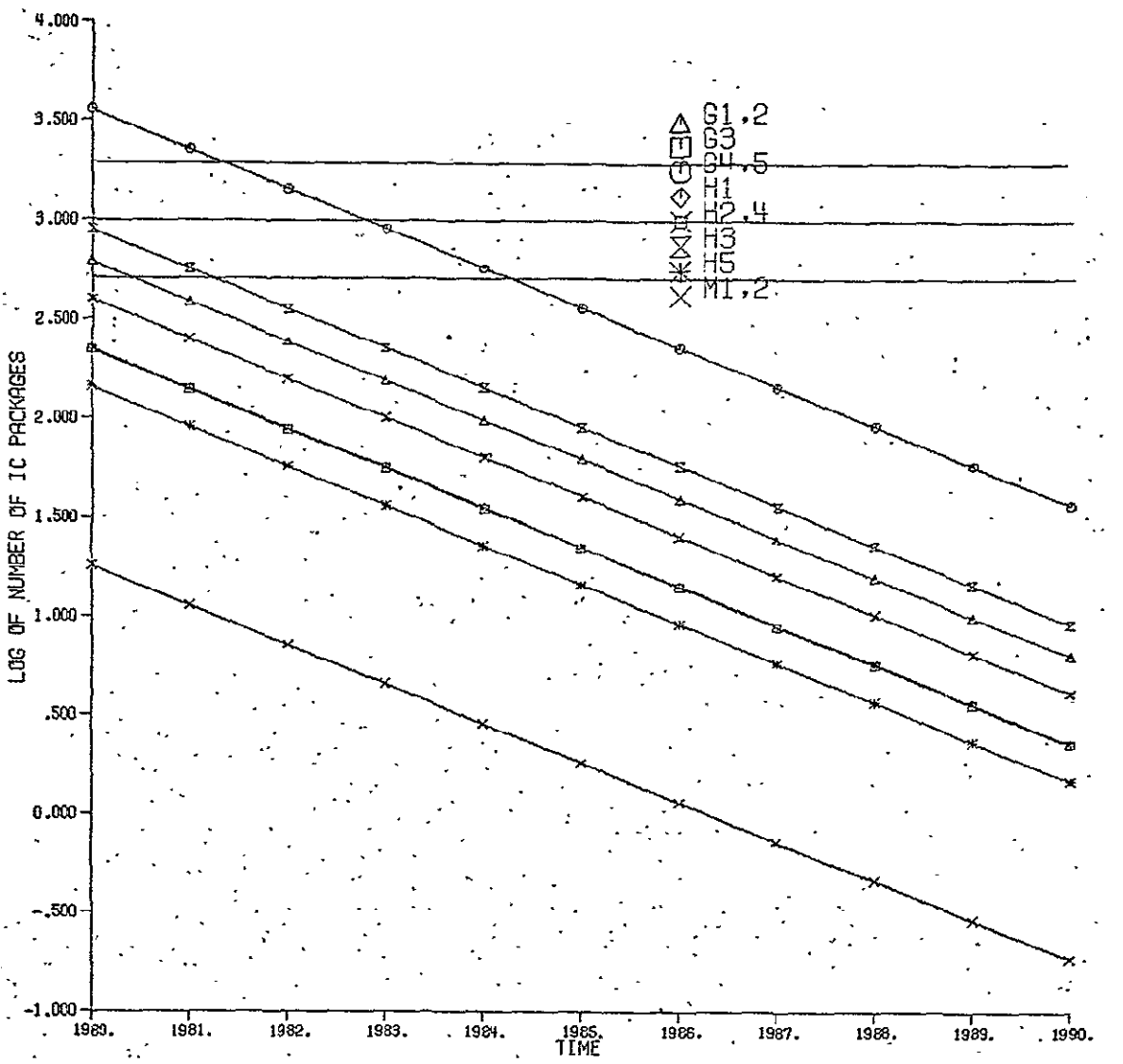


Figure 4.4.2(6)A Number of IC's vs: Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I): Applications: Geography, Hydrology, and Meteorology.

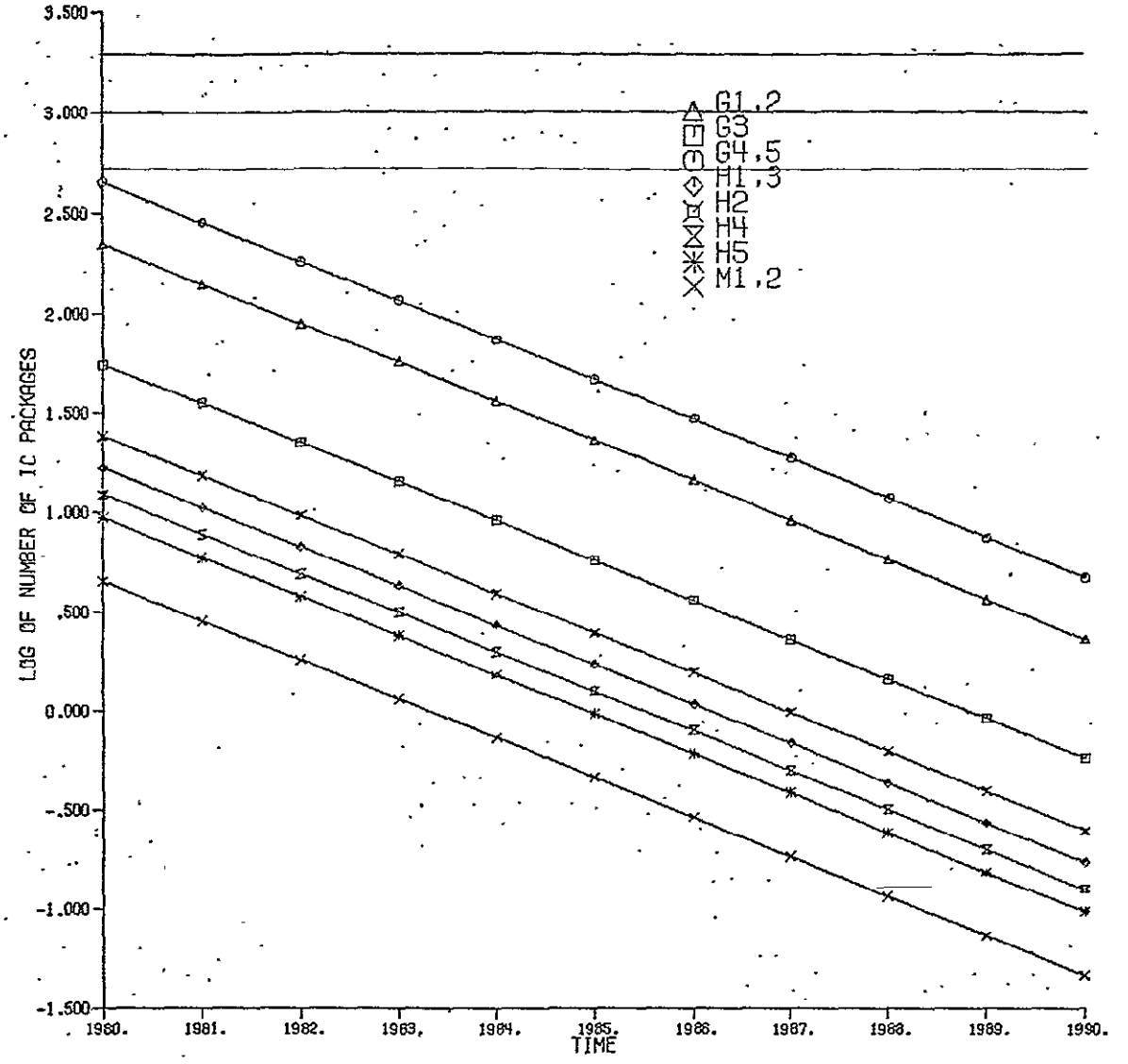


Figure 4.4.2(6)B. Number of IC's vs. Launch Date to Implement the Hardware Maximum Likelihood (HML) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

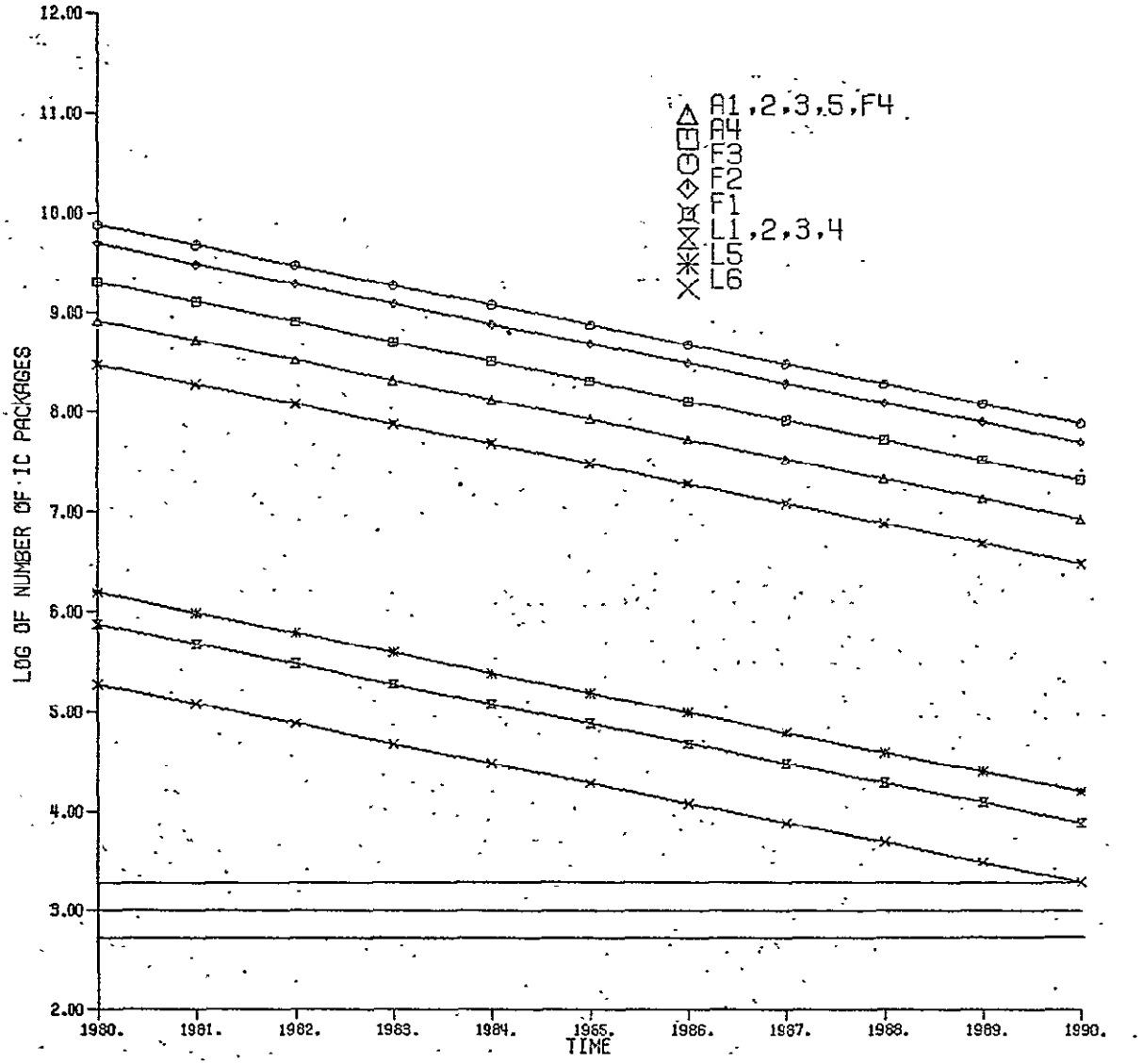


Figure 4.4.2(7)A Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I), Applications: Agriculture, Forestry and Geology.

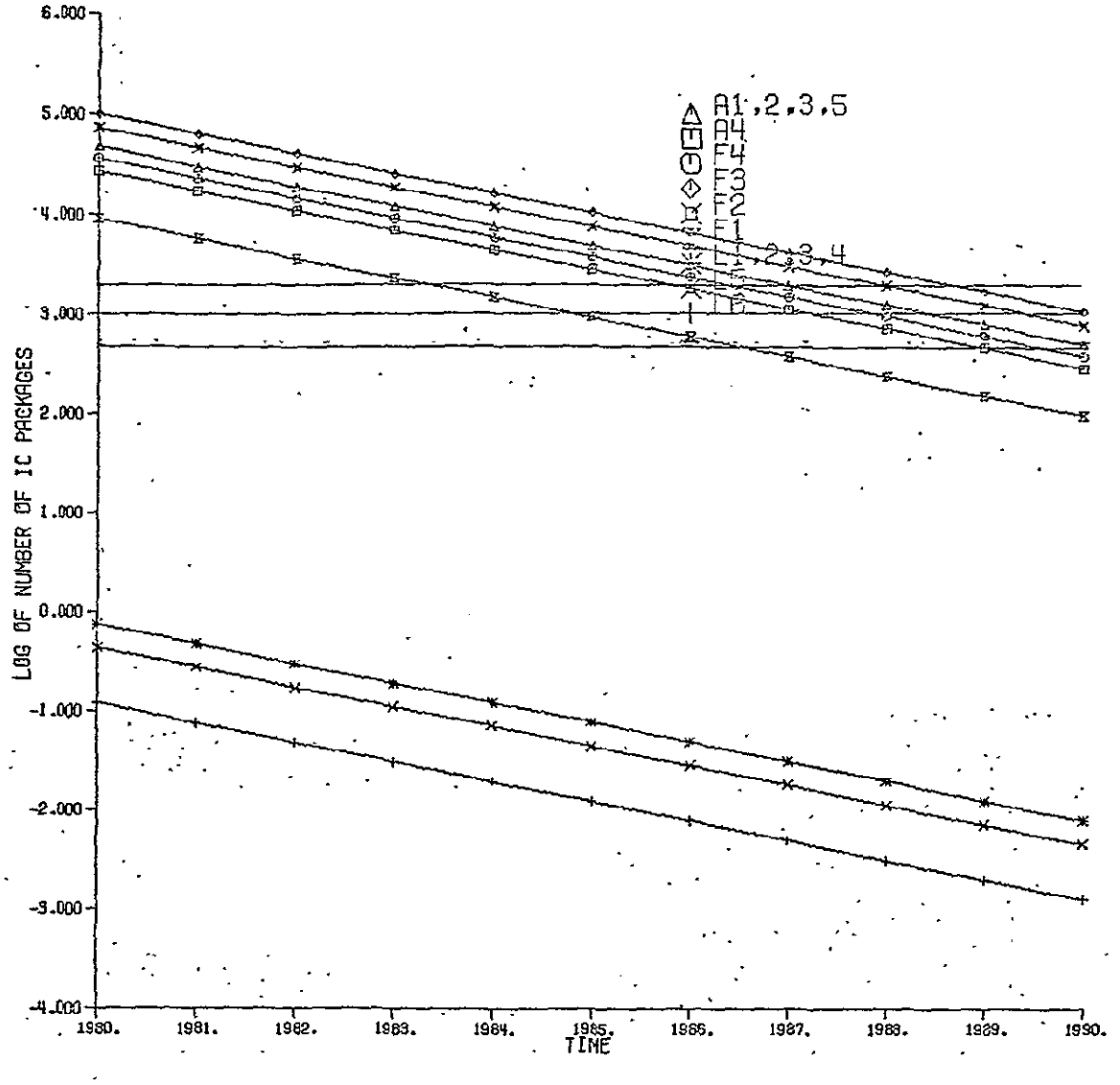


Figure 4.4.2(7)B Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

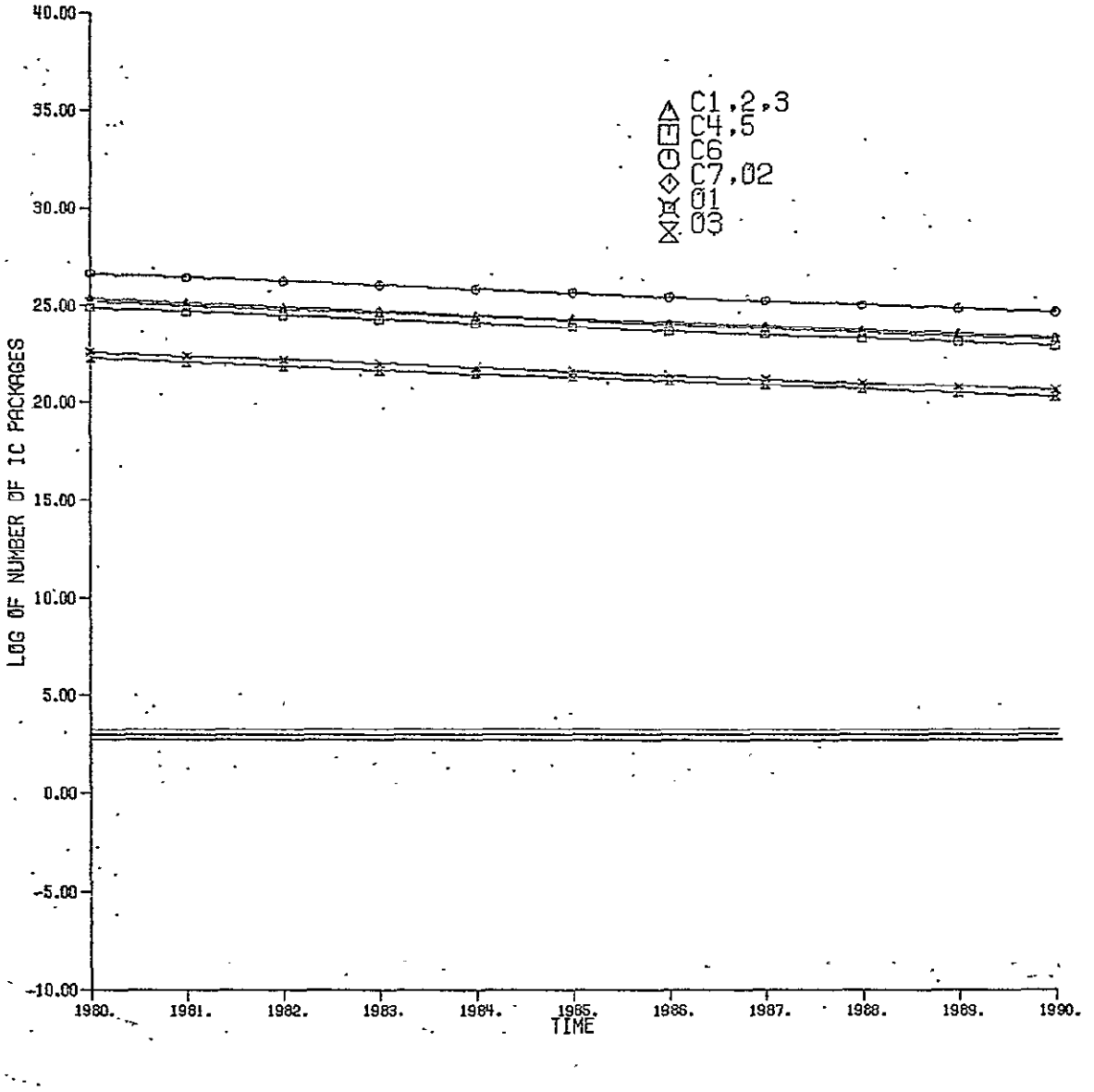


Figure 4.4.2(8)A Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I): Applications: Coastal-Zone Studies and Global Oceanography.

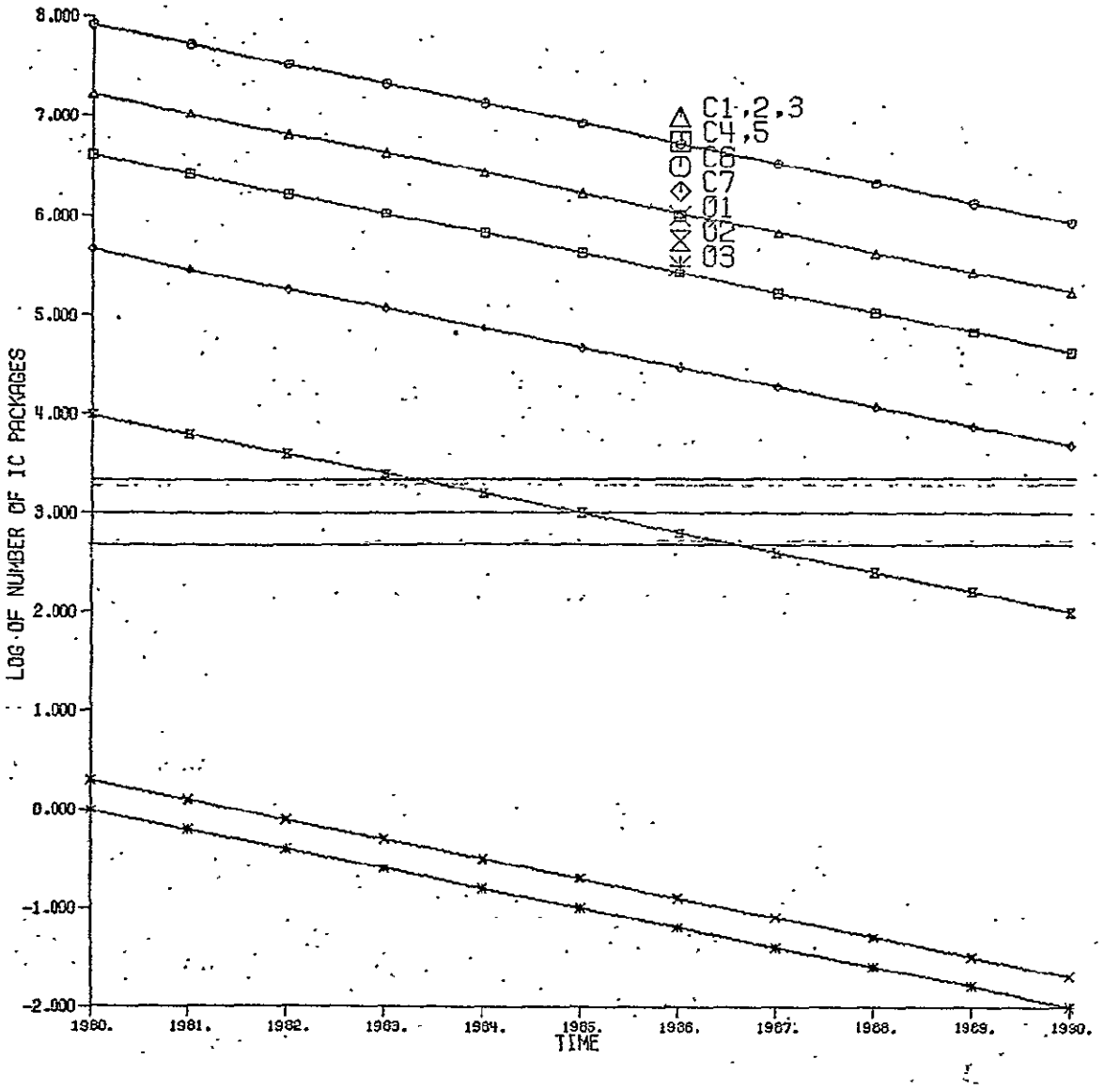


Figure 4.4.2(8)B Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Coastal-Zone Studies and Global Oceanography.

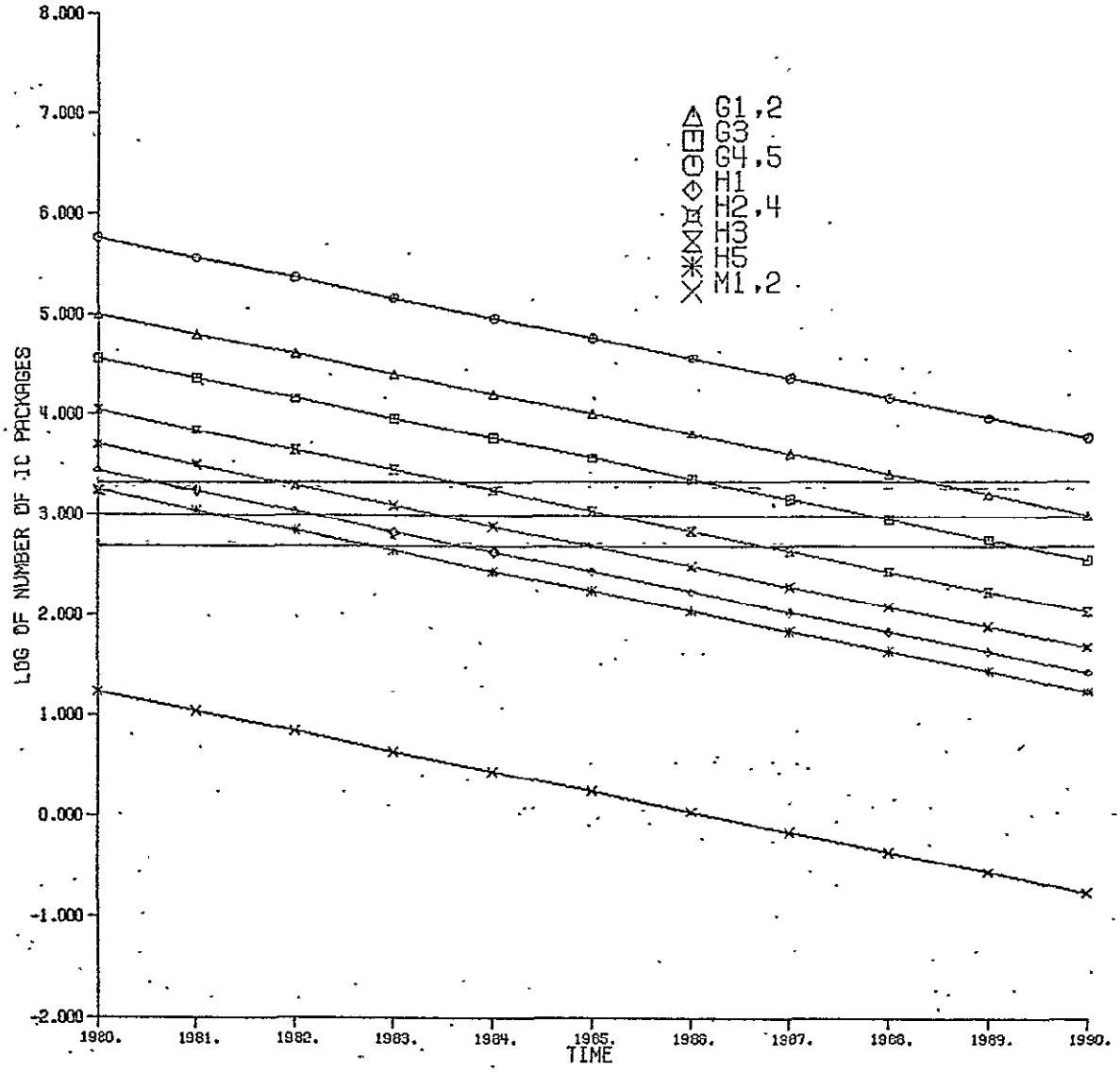


Figure 4.4.2(9)A Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

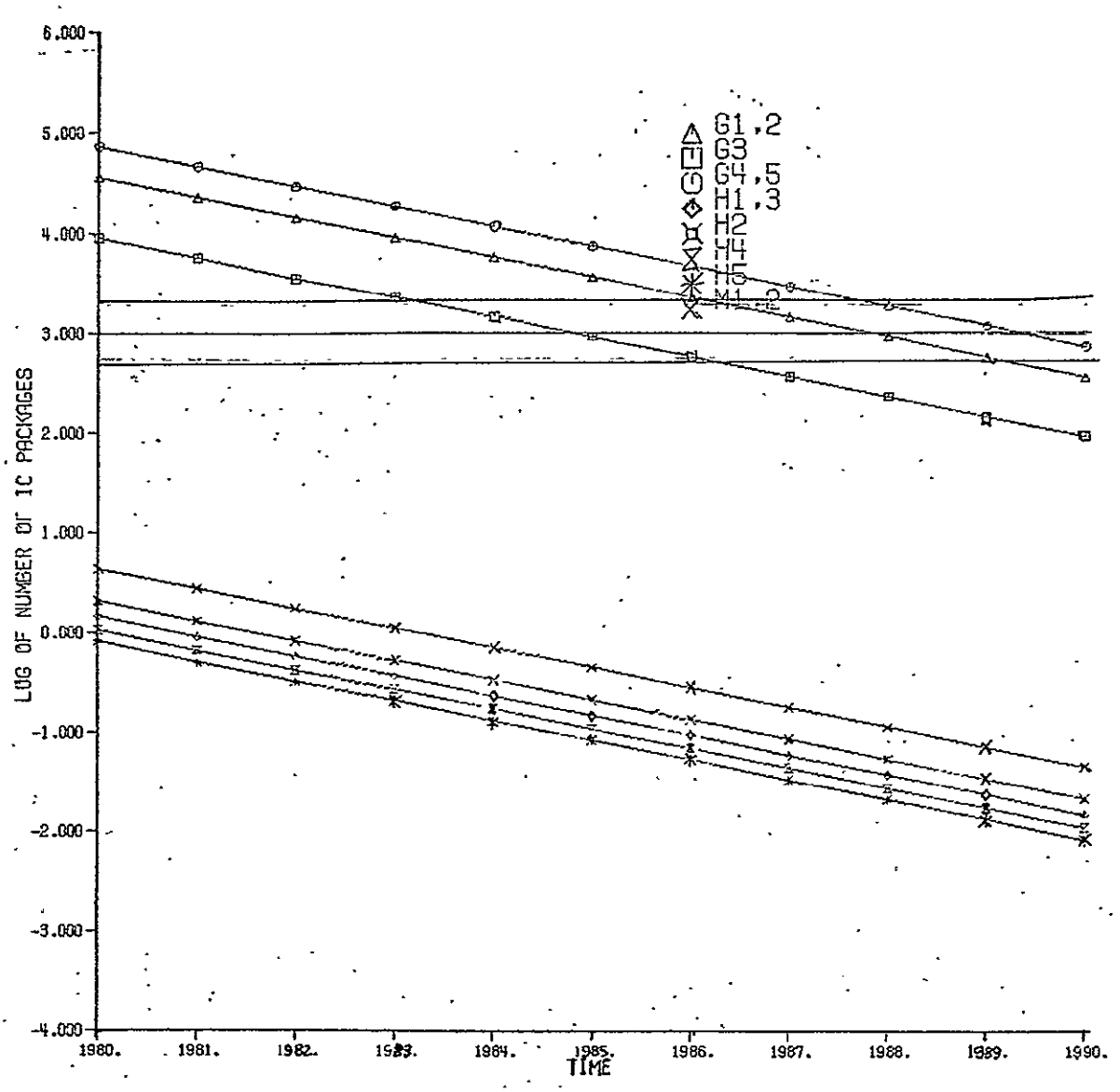


Figure 4.4.2(9)B Number of IC's vs. Launch Date to Implement the Microprocessor Table Look-Up (μPTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

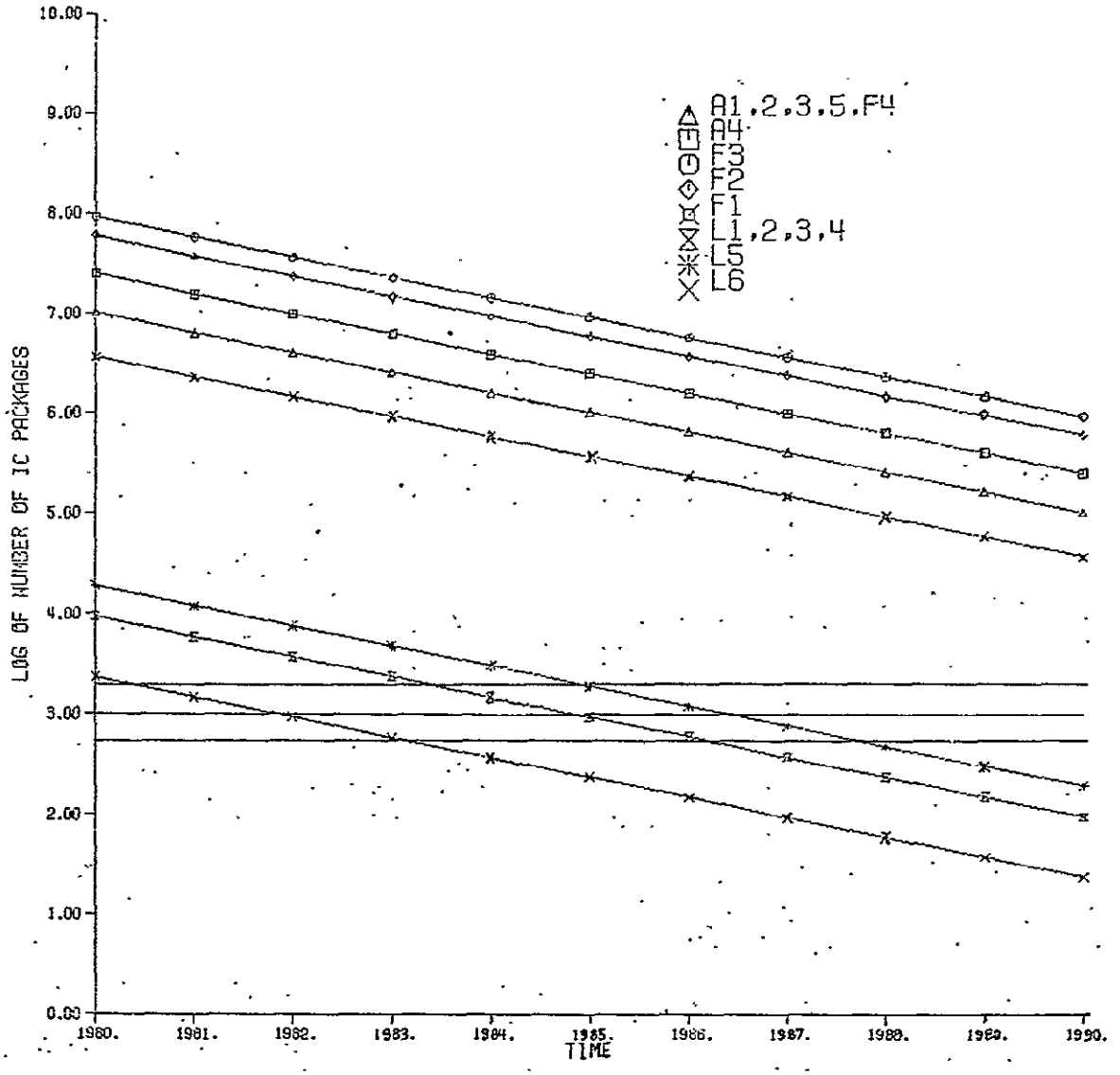


Figure 4.4.2(10)A Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up(HTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

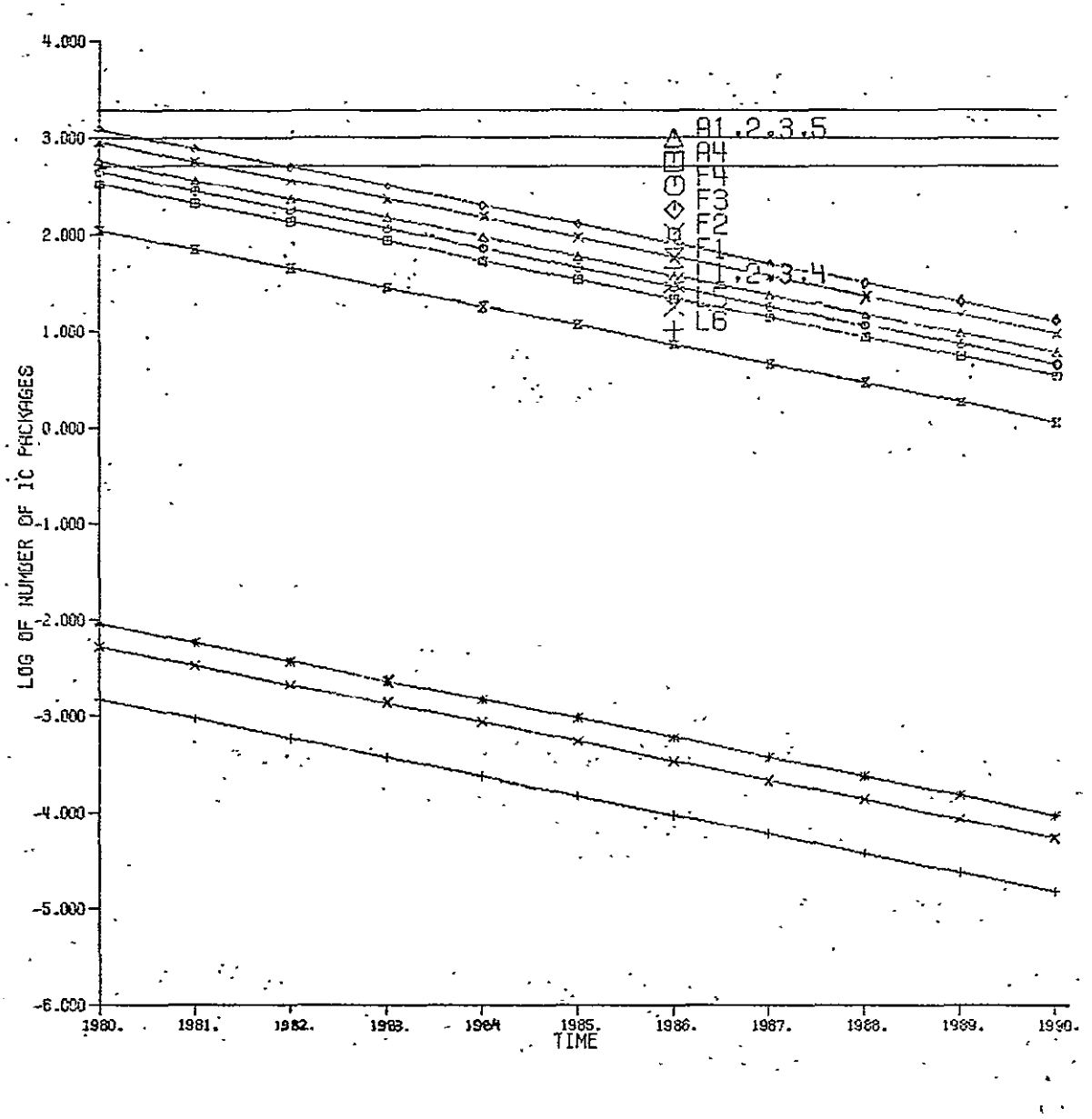


Figure 4.4.2(10)B Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up (HTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Agriculture, Forestry and Geology.

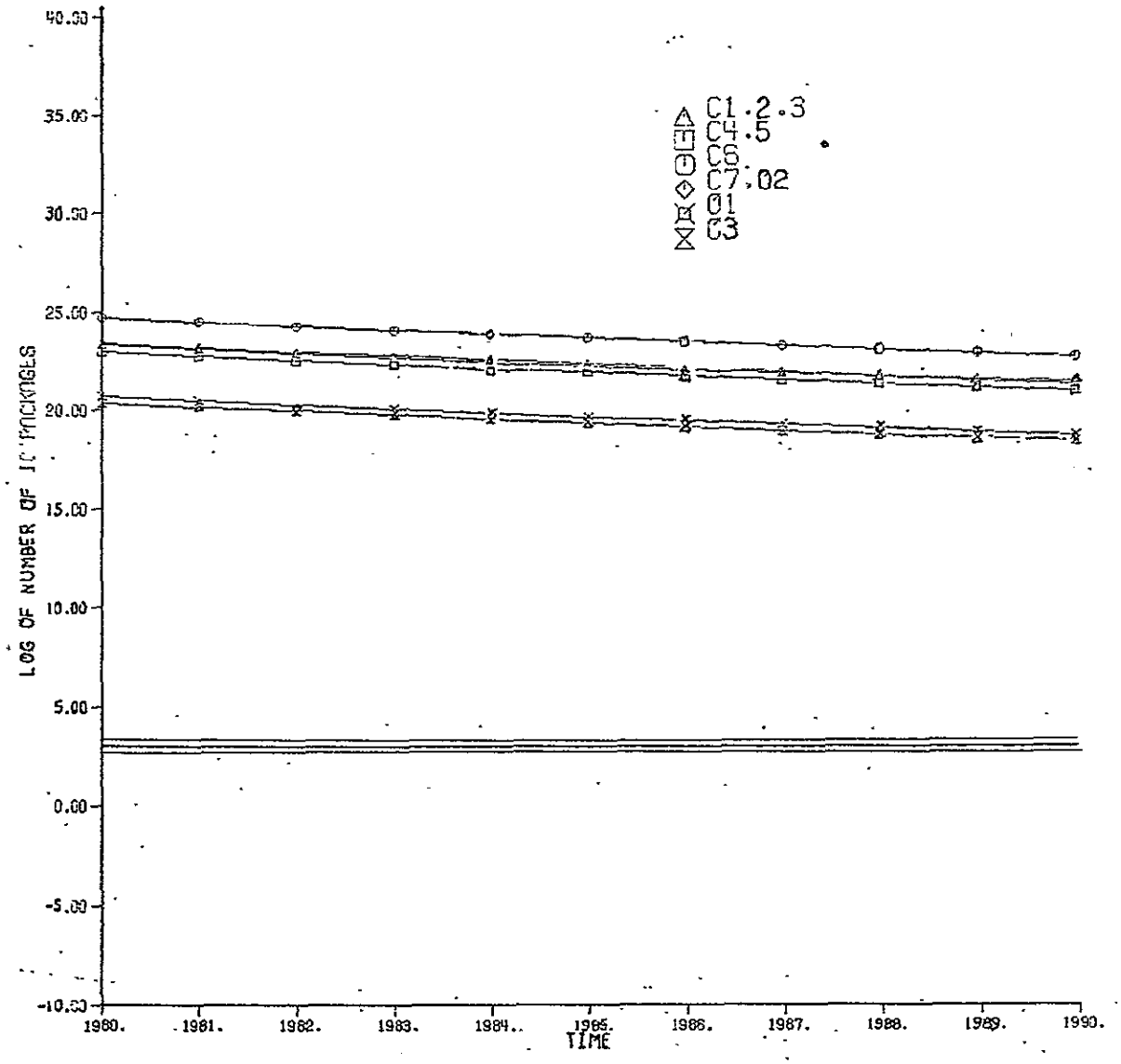


Figure 4.4.2(11)A Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up (HTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Coastal-Zone Studies and Global Oceanography.

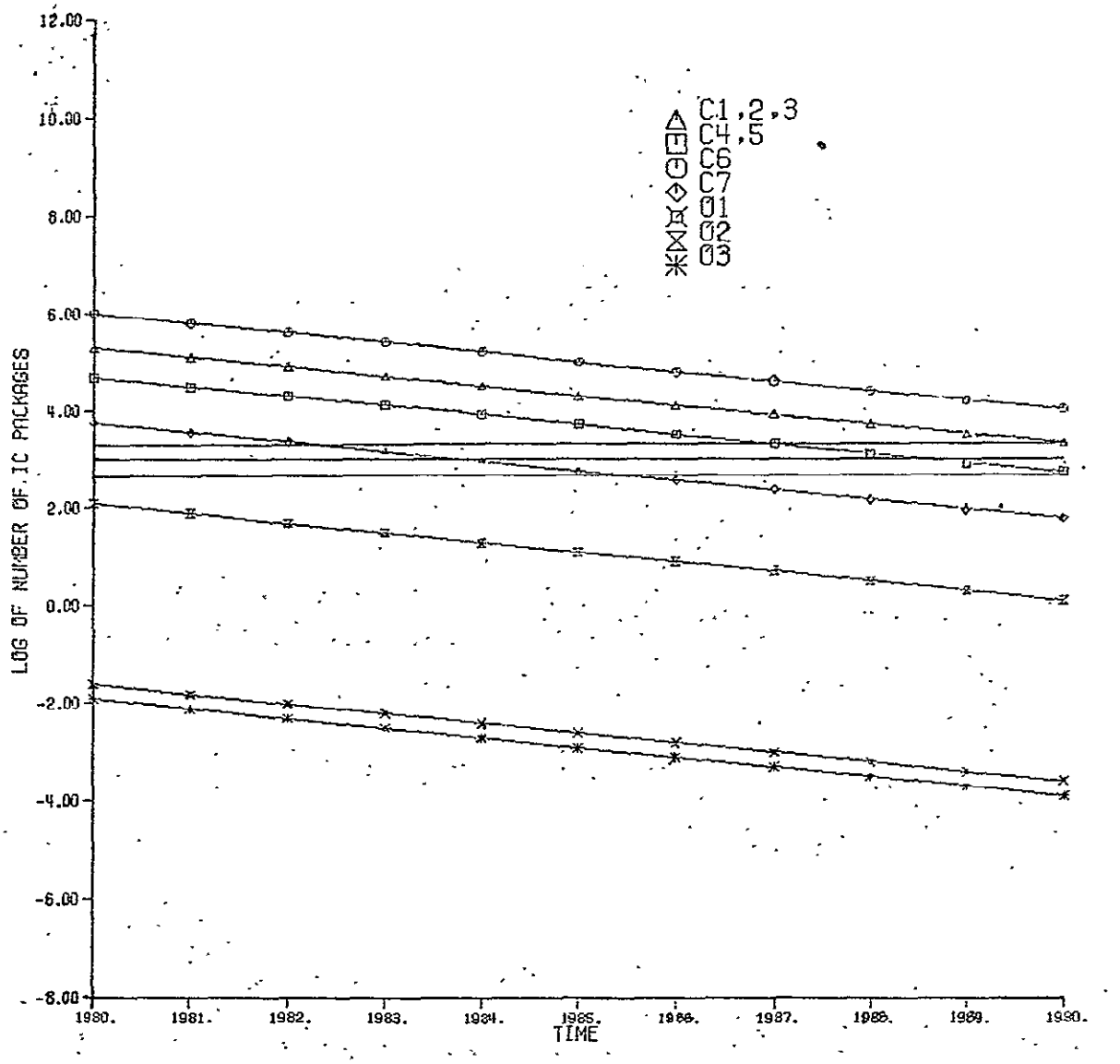


Figure 4.4.2(11)B Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up (HTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Coastal-Zone Studies and Global Oceanography.

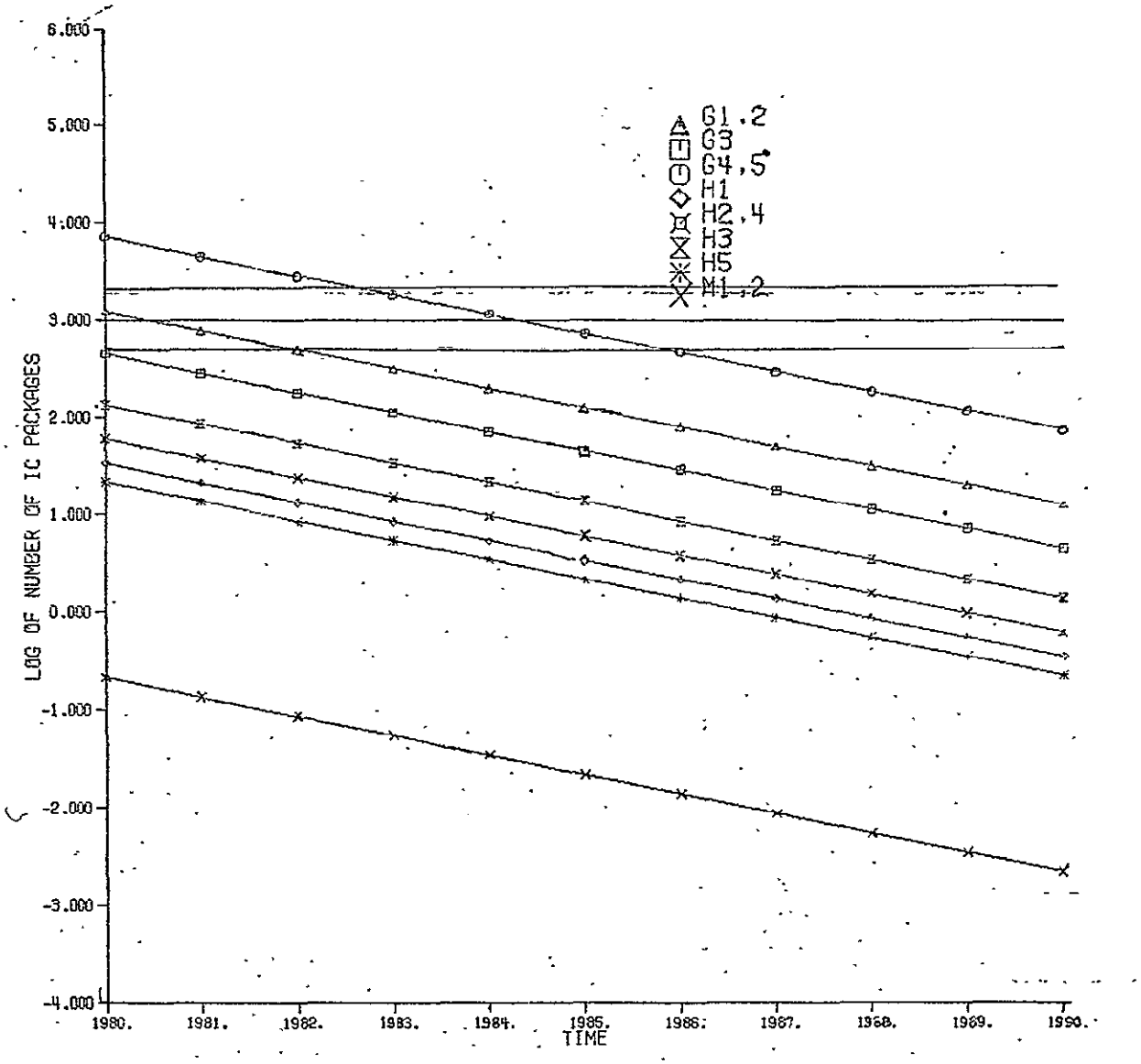


Figure 4.4.2(12)A Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up (HTLU) Processor for the Maximum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

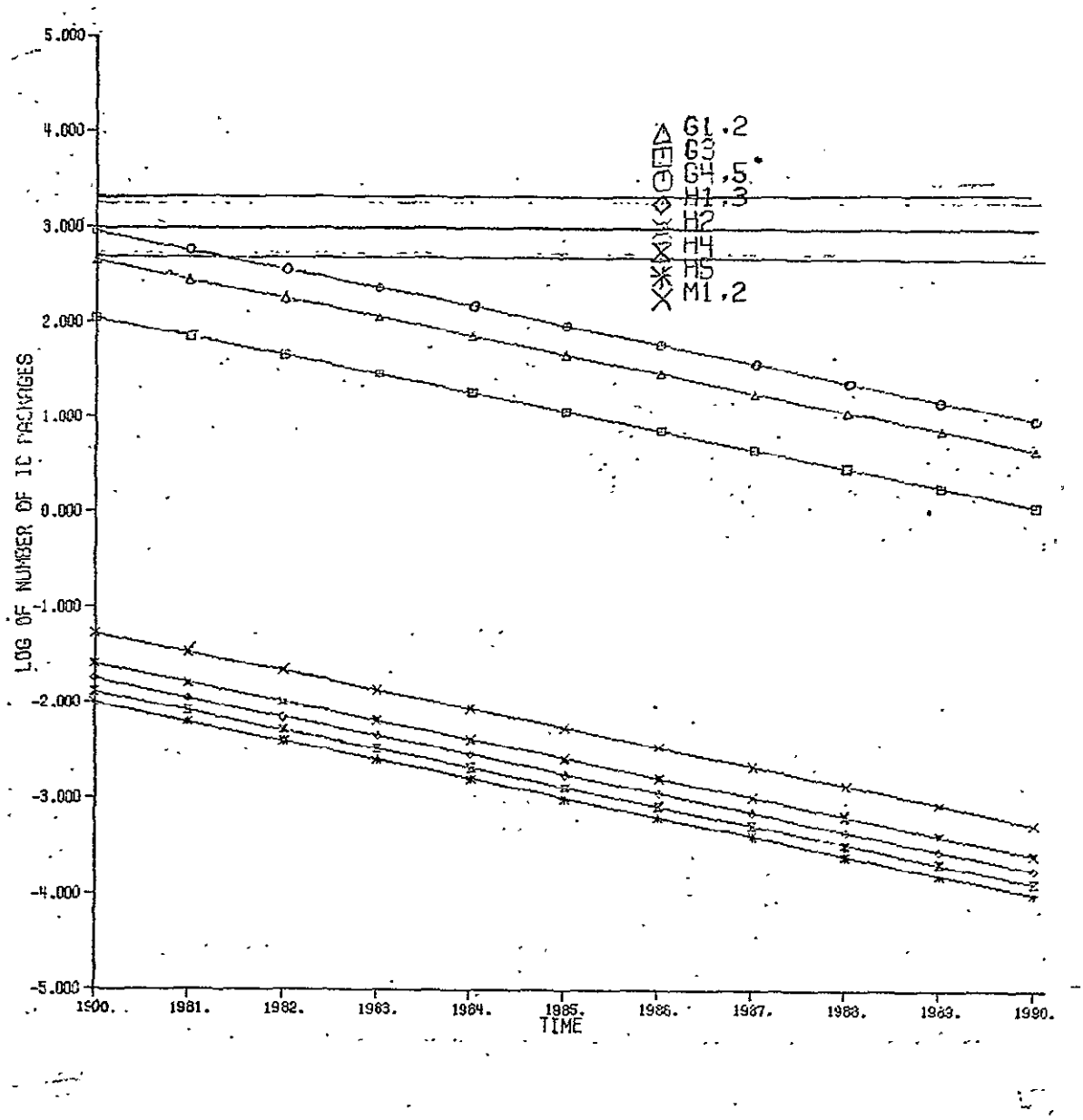


Figure 4.4.2(12)B Number of IC's vs. Launch Date to Implement the Hardware Table Look-Up (HTLU) Processor for the Minimum Data Rate Requirement in Table 1.1.2(I). Applications: Geography, Hydrology and Meteorology.

Table 4.4.2(V) For Each Application Listed in the First Column, the Succeeding Columns List the Year that the Processor Becomes Feasible (2000 IC's) for the Maximum Requirements Listed in Table 1.1.2(I): N means Not Feasible by 1990.

Application	Microprocessor Maximum Likelihood (μPML)	Hardware Maximum Likelihood (HML)	Microprocessor Table Look-Up (μPTLU)	Hardware Table Look-Up (HTLU)
A1	1988	1980	N	N
A2	1988	1980	N	N
A3	1988	1980	N	N
A4	N	1981	N	N
A5	1988	1980	N	N
C1	N	1983	N	N
C2	N	1983	N	N
C3	N	1983	N	N
C4	N	1981	N	N
C5	N	1981	N	N
C6	N	N	N	N
C7	N	1984	N	1984
F1	1985	1980	N	N
F2	N	1983	N	N
F3	N	1984	N	N
F4	1988	1980	N	N
G1	1986	1980	1988	1980
G2	1986	1980	1988	1980
G3	1983	1980	1986	1980
G4	1989	1981	N	1982
G5	1989	1981	N	1982
L1	1984	1980	N	1983
L2	1984	1980	N	1983
L3	1984	1980	N	1983
L4	1984	1980	N	1983
L5	1986	1980	N	1984
L6	1981	1980	1990	1980
H1	1983	1980	1980	1980
H2	1985	1980	1981	1980
H3	1986	1980	1983	1980
H4	1985	1980	1981	1980
H5	1982	1980	1980	1980
M1	1980	1980	1980	1980
M2	1980	1980	1980	1980
O1	1980	1980	N	N
O2	N	1984	N	N
O3	1980	1980	N	N

Table 4.4.2(VI) For Each Application Listed in the First Column, the Succeeding Columns List the Year that the Processor Becomes Feasible (2000 IC's) for the Minimum Requirements Listed in Table 1.1.2(I): N means Not Feasible by 1990.

Application	Microprocessor Maximum Likelihood (μPML)	Hardware Maximum Likelihood (HML)	Microprocessor Table Look-Up (μPTLU)	Hardware Table Look-Up (HTLU)
A1	1984	1980	1987	1980
A2	1984	1980	1987	1980
A3	1984	1980	1987	1980
A4	1983	1980	1985	1980
A5	1984	1980	1987	1980
C1	1985	1980	N	1990
C2	1985	1980	N	1990
C3	1985	1980	N	1990
C4	1982	1980	N	1986
C5	1982	1980	N	1986
C6	1988	1980	N	N
C7	1980	1980	N	1982
F1	1980	1980	1983	1980
F2	1985	1980	1988	1980
F3	1986	1980	1988	1980
F4	1983	1980	1986	1980
G1	1983	1980	1986	1980
G2	1983	1980	1986	1980
G3	1980	1980	1983	1980
G4	1985	1980	1987	1980
G5	1985	1980	1987	1980
L1	1980	1980	1980	1980
L2	1980	1980	1980	1980
L3	1980	1980	1980	1980
L4	1980	1980	1980	1980
L5	1980	1980	1980	1980
L6	1980	1980	1980	1980
H1	1980	1980	1980	1980
H2	1980	1980	1980	1980
H3	1980	1980	1980	1980
H4	1980	1980	1980	1980
H5	1980	1980	1980	1980
M1	1980	1980	1980	1980
M2	1980	1980	1980	1980
O1	1980	1980	1980	1980
O2	1980	1980	1983	1980
O3	1980	1980	1980	1980

4.5 POSSIBLE EFFECTS OF NASA STIMULUS TO INDUSTRY

Landsat-1 and 2 remote-sensing satellites have effectively demonstrated the value of monitoring of the earth's processes and resources. Earth has only a limited ability to support life, and we must continuously determine our effect on the environment. To be effective, this monitoring must be done on a long term basis and, while the rewards will be great, the costs will not be cheap. On-board processing would be very important to reduce the flow of data to the ground stations and consequently reduce the costs. Under practical cost and power considerations, the technology is not here today to build and fly a practical system, although one could be flown to satisfy a few users.

It is reasonable to assume that the technological development of certain critical components would hasten the application date of a cost-effective system. NASA funds could be set aside for the development of critical components of the on-board processor. It is the recommendation of this study for NASA not to stimulate industry through increased funding in those areas which are receiving a natural stimulation. As an example, we feel that it would be unwise for NASA to undertake the development of faster memory, logic modules and microprocessor chips for on-board processors. There is already considerable economic pressure on industry to develop these items and improvements in their performance will occur without NASA's aid to industry. This philosophy would allow NASA to spend its resources in areas which receive little stimulus from outside sources and yet are vital to the continuation of NASA programs. The special technologies discussed in Section 3.6 are example of the type of technology which would benefit the greatest through NASA stimulation and support.

5 SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

This section contains an overview of the entire study reported in detail in the preceding sections. Significant results and conclusions are discussed, and recommendations for future actions by NASA are presented.

Section 5.1 defines the study objectives, gives a description of the tasks, and a description of the significant results of each of the tasks.

Section 5.2 describes the overall conclusions resulting from the total study.

Section 5.3 contains a number of recommendations to NASA as a result of this study.

5.1 SUMMARY

In this section we give a brief description of the problem of on-board earth resources data processing. The problem can be subdivided into a number of tasks, some of which are interdependent. The procedure for carrying out each of the tasks and the significant results are discussed.

5.1.1 Study Objective

Most of the past effort in the field of earth resources data processing has been research oriented. Earth resources imagery has been provided by NASA to a number of researchers who have processed the data in various ways in order to determine what, if any, useful information could be extracted from it. These experiments have demonstrated that useful information can indeed be extracted from aircraft and satellite multispectral scanner imagery of the earth's surface. Economic studies have indicated potential cost effective systems based on these techniques. Consequently, it is anticipated that during the 1980-1990 decade earth resources satellites will be designed and flown for specific purposes, i.e., to monitor severe weather systems, to monitor water pollution, to survey and monitor world food production, etc. In these applications it may be more cost effective to process the data on-board the satellite and transmit the data products directly to the users rather than transmit the raw data to a ground processing station for generating the data products and then distributing the data products to the users via another satellite system.

The purpose of this study was to investigate the feasibility of an on-board earth resources data processor launched during the 1980-1990 time frame. Since about five years are required to design, build, check out, and

launch such a system, a 1980 system would be based on 1975 technology, and a 1990 system would be based on 1985 technology.

In order to determine the feasibility of on-board processing we must first define the on-board processor. This requires that we define both the technology available for use in the design and the computational requirements required of the processor. The computational requirements depend on the algorithms that the processor must implement which in turn, depend on the data products that must be extracted from the data to satisfy the users. Consequently, in order to determine the feasibility of on-board data processors we must start with a study of projected user applications to define the data format (data throughput rate, number of spectral bands, etc.) and the information extraction algorithms the processor must implement. Based on these constraints and the constraints imposed by the available technology we can design some on-board processors and evaluate their feasibility. The study plan is summarized in Figure 5.1.1(1).

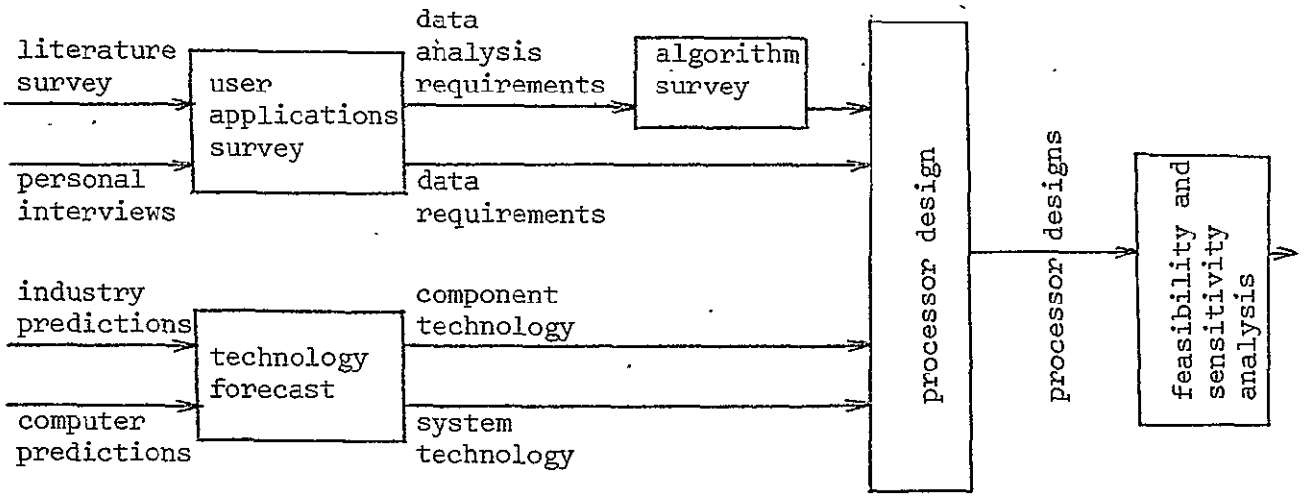


Figure 5.1.1(1) Study Plan

5.1.2 User Applications Survey

A number of studies to define earth resources applications areas, their data requirements, priorities, etc., have recently been completed. The major source and reference documents include:

OA - Earth Resources Programs Summary

Results and Projected Applications-ERTS-1 Applications and Investigations

TERSSE volumes 1 to 8

Advanced Scanners and Imaging Systems for Earth Resources

EOS - Paylord Discussion Group Report

EOS - Mission Review Group Report

Interplan Cost Benefit Study

Dynatrend Cost Benefit Study

EARTHSAT Cost Benefit Core Studies

From these references and from a number of personal interviews with researchers at a number of government and university laboratories, we determined eight potential classes of earth resources data users: Agriculture, Coastal-Zone Studies, Forestry, Geography, Geology, Hydrology, Meterology, and Global Oceanography, and a number of subclasses within each of these classes. For each potential user we determined the minimum and the maximum resolution, the minimum and maximum field of coverage, the minimum and maximum number of spectral bands, and finally, based on these, the minimum and maximum data rate out of the multispectral scanner. These results are tabulated in Table 1.1.2(I). The resolutions range from a minimum of 3 m to a maximum of 10 km. The fields of coverage range from 15-800 km. The number of spectral bands ranges from 1-20 and the resulting data rates range from a minimum of 312 bits/sec to 3470 megabits/sec.

Since the data requirements for the various users cover such a wide range we selected a single candidate data format for the initial effort. This candidate data format has a swath width of 185 km, a resolution of 40 m, a satellite ground track velocity of 6500 m/sec, 7 spectral bands, and 6 bits per data word. The resulting data rate from the MSS is 32 megabits/sec. This data rate satisfies all but two of the minimum data rates and about half of the maximum data rates suggested by the users.

5.1.3 Data Analysis Algorithm Survey

Almost all of the data users surveyed indicated that their objectives could be satisfied using spectral signature analysis. Consequently, a detailed survey was made of algorithms for classifying n-dimensional vectors into one of M categories or classes, where n is the number of spectral bands. As a result of this survey it was determined that four algorithms warranted detailed analysis. These are clustering, maximum likelihood, perceptron and table look-up.

Clustering is an unsupervised data analysis technique used to determine th natural or inherent data classes in a set of observations. Many such algorithms

have been studied. All essentially make a scatter plot of a subset of the data to determine the different groupings within the data. Each group is assigned a label, and all of the data with this label are compared to ground truth to associate each label with one of the classes defined by the data user. After this training is completed, each data point is classified by measuring the distance between it and each of the cluster centers and classifying it according to the nearest cluster.

The maximum likelihood algorithm is a statistical proceedure based on the probability density function of the data. For the case of Gaussian data, which is a valid approximation for multispectral imagery of the earth's surface, only first and second order statistics are required. Training is accomplished by measuring these statistics for some data samples from known classes and then assuming that all data from the same class has these same statistics. Subsequently, data are classified by comparing their statistics to the statistics of each of the classes and deciding in favor of the class they most closely resemble.

The perceptron algorithm is based on a set of decision functions which are adjusted by an iterative technique to fit training data of known class and then used to classify subsequent data.

The table look-up algorithm essentially stores in a large table (computer memory) all possible outcomes of the data and associates with each possible outcome one of the classes. Training is required to associate one of the classes with each of the possible values of the input data. Subsequent data are then classified by using the data point to address the memory to look up the classification.

The clustering, maximum likelihood and perceptron algorithms require a significant amount of computation, mainly additions, multiplications and comparisons. The table look-up algorithm requires a much smaller amount of computation, but significantly more memory.

5.1.4 Preprocessing Algorithms

The extremely large volume of data generated by the MSS imposes a severe computational burden on the on-board processor. The possibility of using a preprocessor between the sensor and the processor to reduce the bulk of data by using data compression, feature selection, etc., techniques was studied.

Transform coding allows a data bulk reduction by a factor of 2 to 4 for most multispectral data without degrading the data quality.

The BLOB algorithm developed at Purdue University achieves data bulk reduction by a factor of 10 to 30, but requires more computation and more memory than transform coding.

5.1.5 Algorithm Computational Requirements

Each of the data analysis algorithms (clustering, maximum likelihood, perceptron, and table look-up) and the preprocessing algorithms (transform coding, and BLOB coding) were analyzed in detail relative to their computational requirements, i.e., the number of additions, multiplications, comparisons, etc., required to implement these algorithms along with the requirements imposed by the sequence of operations (some operations can be done in parallel while others follow a sequence where one operation must be completed before the next can begin). These algorithm computational requirements were tabulated for each of the data analysis and preprocessing algorithms.

Using a preprocessor to reduce the load on the processor is not a lucrative alternative. Even though the preprocessor can reduce the data load by a factor from 2 to 30 and thus reduce the complexity of the data processor by this amount, the total system complexity is not reduced because the savings in processor complexity are more than overwhelmed by the increase in the preprocessor complexity.

It was further determined that the perceptron and clustering algorithms require a more complex processor than the maximum likelihood and table look-up algorithms for all user requirements. Consequently, we concluded that only the maximum likelihood and table look-up algorithms are worthy of further consideration.

5.1.6 Technology Forecast and Assessment

A detailed survey of 1975 component technologies was completed. A number of 1975 microelectronics technology families are listed in Table 3.2.1(I). The speeds, power, size, cost, reliability, etc., of each are tabulated.

Component technology was also projected from 1975 to 1985 using estimates obtained from component manufacturers and other experts in the field. The major conclusions are that some parameters associated with microelectronic component technology are changing at rates between 1 or 2 orders of magnitude every 10 years, with the result that overall component performance is changing by several orders of magnitude every 10 years. In particular, the number of components (gates, transistors, etc.) per chip increased by a factor of 10 between 1965 and 1975 and is expected to increase by another factor of 10 between 1975 and 1985. In addition, propagation delays decreased by one order

of magnitude between 1975 and 1985, and are expected to decrease by another order of magnitude between 1985 and 1995. With the equivalent number of gates in an IC chip increasing by a factor of 10 and the processing speed increasing by a factor of 10, the total number of computations per unit time (computational power) increases by a factor of 100.

Projections for computer system technology resulted in similar estimates, i.e., microcomputer cycle times, add times, etc., are projected to decrease by one order of magnitude during the next 10 years as they have for the past 10 years. The number of bits of memory contained in a given area on an IC chip are likewise projected to increase by an order of magnitude over the next 10 years as they have over the past 10 years. Meanwhile, the size and power dissipation, per IC chip is expected to stay constant while the number of pins per package which increased by a factor of four between 1965 and 1975 is expected to increase by only a factor of two between 1975 and 1985.

We also developed a computer model that uses input data from past years to predict future values of these parameters. These computer generated projections are in close agreement to the predictions made by experts from the microelectronics industry.

5.1.7 On-Board Processor Designs

A number of on-board processors capable of implementing the maximum likelihood and table look-up algorithms for the candidate input data format were designed. In order to operate in real time at the 32 megabit/sec data rate the designs are based on multiprocessor concepts using pipeline and parallel arrays of subprocessors. Sufficient subsystems were added in parallel to obtain the 32 megabit/sec throughput.

Two different design approaches were investigated in detail. One is a hardware approach consisting of logic circuits designed to efficiently implement the mathematical operations required by the algorithms. One special purpose hardware design was done to implement the maximum likelihood algorithm and another special purpose hardware design was designed to implement the table look-up algorithm.

The second design approach uses microprocessors which allows a number of different computations to be done with the same hardware under software control. Programs for implementing all of the computations were written in order to determine the number of instruction cycles required to implement the algorithm. This established the throughput data rate and, consequently, the

number of parallel subsystems required to handle the 32 megabit/sec rate.

Applying both of these design approaches to both algorithms resulted in four system designs. Hardware maximum likelihood (HML), hardware table look-up (HTLU), microprocessor maximum likelihood (μPML) and microprocessor table look-up (μPTLU). For each of these designs the number of IC's, power, volume, weight, and cost were determined based on 1975 technology.

Because microprocessors are significantly slower than TTL circuits the hardware approaches require fewer IC's, less power and volume, and cost less than the microprocessor designs.

5.1.8 Feasibility Trade-Off and Sensitivity Analysis

Each of the processor designs handle the 32 megabit input rate by distributing the processing load between many similar subprocessors. Consequently, the number of IC's, power, weight, volume, and cost are all essentially proportional to the number of subprocessors. Therefore, we defined a system complexity function for each of the four processors and determined its dependence on the following parameters using 1975 technology:

- R data bit rate (bits/sec)
- n number of spectral bands (channels)
- b number of bits per resolution element per spectral band (bits)
- M number of classes
- r pixel rate (resolution elements/sec)

From the results of the component and system technology forecasts we further took into account the complexity function dependence on time for 10 years into the future. The resulting complexity functions are listed in Table 5.1.8(I)

Table 5.1.8(I) Processor Complexity Functions

Processor	Complexity Function
microprocessor maximum likelihood (μPML)	$P_1 = k_1\, M(n+1)\, R(1.5)^{-T}$
hardware maximum likelihood (HML)	$P_2 = k_2\, M(n+1)\, R(1.5)^{-T}$
microprocessor table look-up (μPTLU)	$P_3 = k_3\, M\, R\, (1.6)^{nb}\, (1.5)^{-T}$
hardware table look-up (HTLU)	$P_4 = k_4\, M\, R\, (1.6)^{nb}\, (1.5)^{-T}$

The scale factors k_1, k_2, k_3, and k_4 were determined for each performance measure (number of IC's, power, volume, weight and cost) and are listed in Table 4.3.3(II).

These models for the four design approaches were then used to determine the sensitivity of the complexity to the various system parameters. This was accomplished setting all system parameters to their baseline values n = 4, M = 12, b = 6, R = 32 x 10^6, and T = 0 (the candidate baseline format) except for one or more parameters which were then varied from their baseline values. Figures 4.4(1), 4.4(2) and 4.4(3) show the sensitivity of the designs to variations in the data throughput rate R, the time T, the number of bits per data word b and the number of spectral bands n.

Any feasibility analysis depends on the definition of feasible. For a particular processor to be "feasible" at a particular point in time requires that it meet certain constraints on performance, complexity, volume, weight, power, cost, reliability, environment, etc. Each of the four system architectures meets the performance constraint since each was designed to accomplish the required task. All four processors use standard integrated circuit technology and meet the data throughput rates by adding more components (IC's) in parallel. The volume, weight, and power dissipation of integrated circuits can be kept within limits by simply keeping the number of integrated circuits within limits. The radiation, temperature, and other environmental constraints can be met by each processor as discussed in Section 2. The limiting factors are cost and reliability which can also be kept within bounds by imposing a constraint on the number of components. Consequently, we conclude that on-board processing using a particular processor is feasible provided we constrain the number of IC's in the processor to a reasonable number.

The parts cost of the on-board processor increases linearly as the number of IC's. The costs associated with check out increase as the square of the number of IC's. Limiting the number of IC's in the on-board processor to about 1000 appears to satisfy all constraints, i.e., cost is reasonable relative to the total system cost (launch, sensors, telemetry, etc.), reliability is pushing the limits of present day technology as discussed in Section 3, while volume, weight, and power dissipation do not appear to present serious difficulties. Figures 4.4.3(1) through 4.4.3(12) show the number of IC packages required to implement each of the four hardware designs as a function of time from 1980-1990.

If we accept the 1000 IC definition of feasibility, then we can list the year in which each user application first becomes feasible for each design approach. These results asre summarized in Table 4.4.3(1) and 4.4.3(2). Summary tables corresponding to other definitions of feasibility could easily be

generated from Figures 4.4.3(1) through 4.4.3(12). These results are summarized in more compact form in Figures 5.1.8(1) through 5.1.8(3), which show the percentages of user applications that can be implemented by each of the four design approaches for both the minimum and the maximum user requirements.

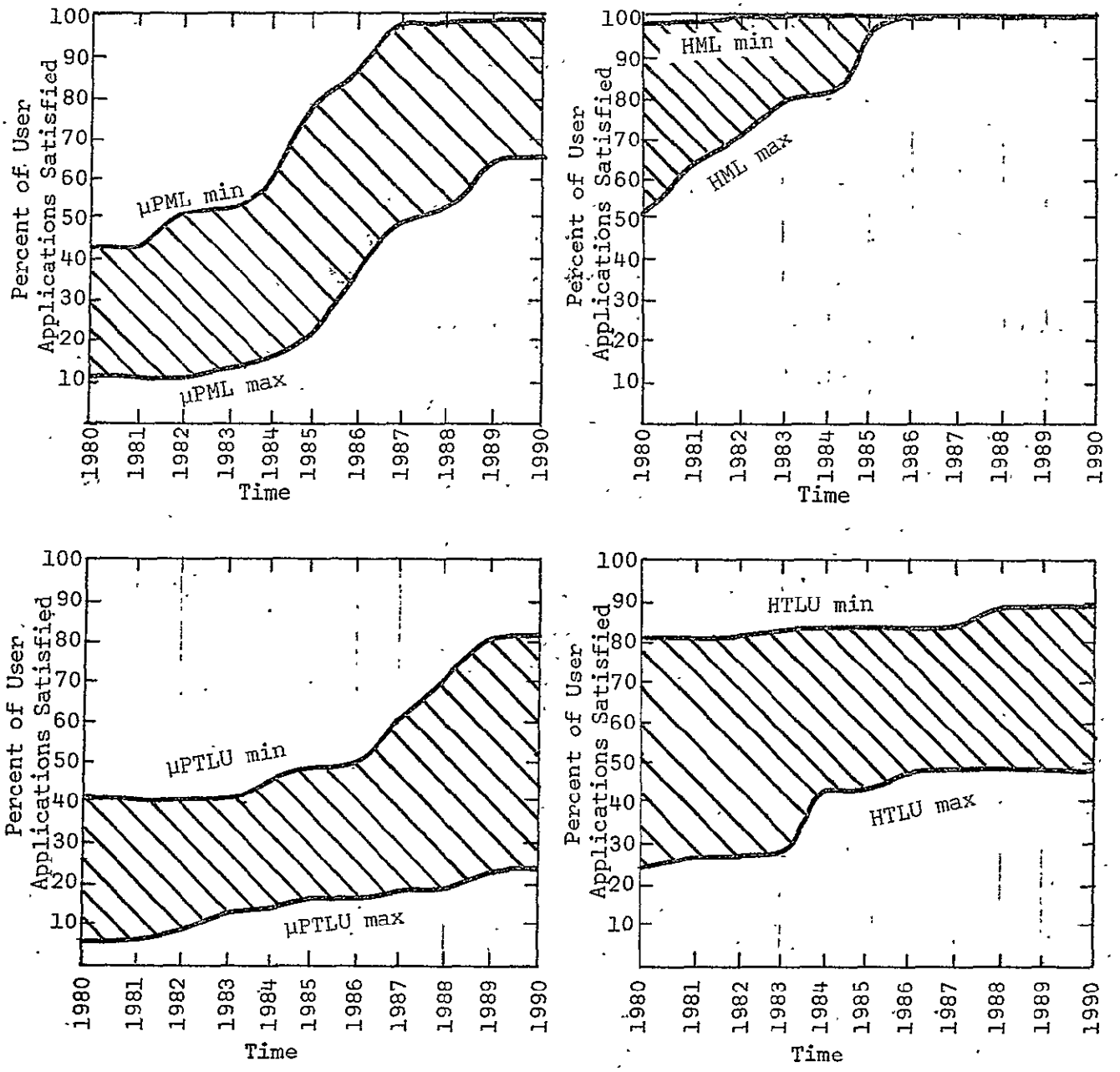


Figure 5.1.8(1) Percent of User Applications Satisfied as a Function of Time (1,000 IC's).

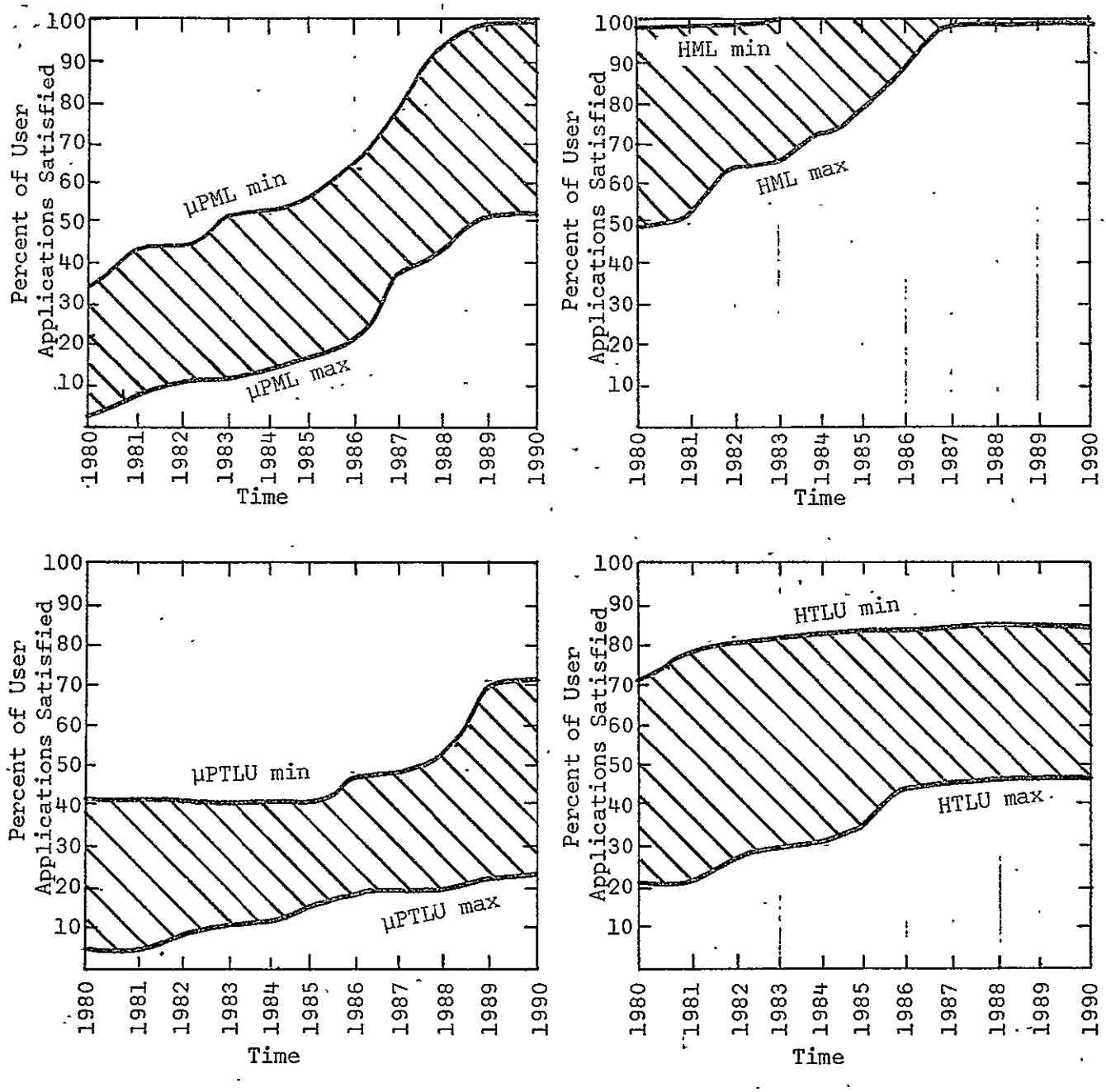


Figure 5.1.8(2) Percent of User Applications Satisfied as a Function of Time (500 IC's).

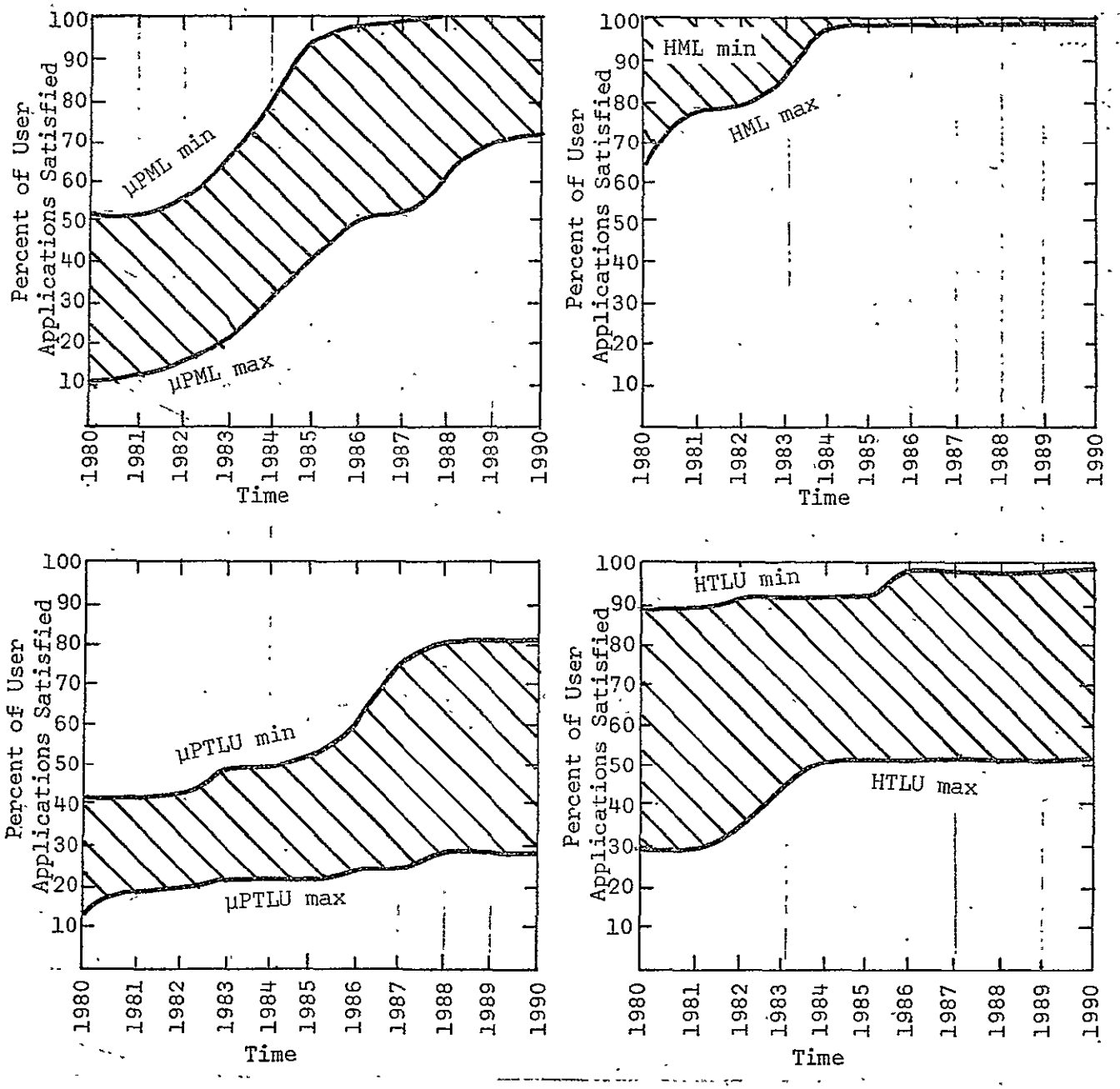


Figure 5.1.8(3) Percent of User Applications Satisfied as a Function of Time (2,000 IC's).

5.2 CONCLUSIONS

Conclusions of this study are:

(1) From results of the user applications survey we conclude that potential users will require a wide range of resolutions, a wide range of field of coverage, a wide range of number of channels, and these in turn result in a wide range of data throughput rates.

(2) From the results of the survey of data analysis algorithms we conclude that the maximum likelihood and table look-up algorithms are superior to other known algorithms for all user requirements. The table look-up algorithm is superior to the maximum likelihood algorithm, except for situations requiring more than five spectral bands.

(3) From the results of the investigation of the possibility of using a preprocessor to reduce the data load on the processor, we conclude that the total on-board system complexity is minimized with no preprocessor.

(4) From the results of the component and computer system technology forecasts and assessment, we conclude that the on-board processor capability (the amount of throughput it will be able to handle) will increase by two orders of magnitude between 1975 and 1985.

(5) From the on-board processor designs and evaluations we conclude that implementations utilizing specially designed hardware require less hardware, power, volume, weight, and cost less than microprocessor (software) based systems.

(6) From the feasibility and sensitivity analysis, we conclude that most but not all user applications could be satisfied by an on-board processor sometime between 1980 and 1990.

5.3 RECOMMENDATIONS

5.3.1 Handling the On-Board Processor Output Data Products

While this study was directed towards determining the feasibility of on-board processors for the 1980-1990 time frame, the question remains as to how the output of an on-board processor could be treated. Now that this study has established the feasibility of on-board processing, the problem of

compressing and distributing the on-board computer output needs to be addressed. It is recommended that a study be made to investigate the uses of on-board processor output with particular attention paid to data rates and formats.

5.3.2 Dates for Cost-Effective Launches

Our study concludes that some users could be satisfied with a processor designed today and flown in 1980. Other users cannot be satisfied until 1990 and beyond. These conclusions are based on technical feasibility and do not address the question of economics. It is recommended, therefore, that a study be made to establish a projected time frame for the launch of cost-effective earth resources missions.

5.3.3 Stimulation of Industry

Finally, it is recommended that no stimulus be given to industry to develop large-scale integration (LSI) technology for on-board earth resources processors. It is recommended instead that the resources be used to encourage the solution of problems peculiar to NASA which do not have a parallel in industry, such as improvements in multi-spectral scanners and special techniques such as parallel processing. While this philosophy may be contrary to NASA's "spin-off" philosophy, stilulii from other sources are already present in the LSI field and further stimulus by NASA would have little effect and would be wasteful of NASA resources.