QUARTERLY TECHNICAL PROGRESS REPORT

October, November, December 1971

MASTER

RECEIVED BY TIC MAY 12 1972

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

QUARTERLY TECHNICAL PROGRESS REPORT

October, November, December 1971

UTUCDCS-QPR-71-4

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

fy

TABLE OF CONTENTS

	Page
1. CIRCUIT RESEARCH.	1
1.1 Fail-safe Bundle Repeater/Restorer	2
1.1.1 Progress in general.	2
1.1.2 Input circuits	2
1.1.3 SABUMA (Safe Bundle Machine.	4
1.2 APE (Project No. 25)	7
1.2.1 The Remotely Tunable Data Receiver	7
1.3 PENTECOST.	11
1.3.1 Penetron	11
1.3.2 Plumbicon.	11
1.4 Ergodic (Project No. 39)	11
1.4.1 Summary.	11
1.4.2 Status	12
1.4.3 Future Work.	12
1.5 Telemaze (Project No. 41).	14
1.5.1 Project Summary.	14
1.5.2 Square Input Recognition	14
1.5.3 Path Generator	14
1.5.4 Input Square Processing.	16
1.5.5 Video Generation	17
2. HARDWARE SYSTEMS RESEARCH	20
2.1 LASCOT (Project No. 09)	
2.2 OLFT (Project No. 12)	
2.2.1 Operation of Intermediate Cooled Light Valve.	21
2.2.2 Optical Configuration for Intermediate Cooled System	21
2.3 BLAST (Project No. 19)	23
2.3.1 Final Status	23
2.4 SEMANTRIX (Project No. 24)	24
2.5 LINDA.	25
2.5.1 Summary.	25
2.5.2 Project Status	25
2.6 RASER.	26
2.6.1 Sync Generator for CRT Monitor	26
2.7 STEREOMATRIX (Project No. 30).	28
2.7.1 Transformer.	28
2.7.2 Coefficient Generator.	28
2.8 SCANTRIX (Project No. 35).	28
2.8.1 Introduction	28
2.8.2 DAC.	30
2.8.3 Buffer	30
2.8.4 Human Eye Response to Brightness	30

	Page
2.9 FROG (Project No. 36)	30
2.9.1 Project status	30
2.9.2 The Scheme	34
2.9.3 The Stimuli.	34
2.9.4 The World Model: Structure of the Memory	35
3. SOFTWARE SYSTEMS RESEARCH	38
3.1 Numerical Processes.	38
3.1.1 Ordinary Differential Equations. . .	38
3.1.2 Sparse Matrix Inversion.	40
3.1.3 The Steady-State Package	43
3.1.4 Plot Package	43
3.1.5 Elimination.	44
3.1.6 Matrix Inversion Package	48
3.2 Non-Numerical Packages	49
3.2.1 Item Analysis.	49
3.3 Graphical Remote Access Support System (Grass).	51
3.3.1 Disk Monitor System.	51
3.3.2 Display Terminals.	52
3.3.2 Information Retrieval.	53
3.3.4 Monitors	54
3.3.5 Remote Data Structure Utilities. . .	55
3.4 Computer Maintenance and Construction. . .	59
3.4.1 Graphics-8 Hardware.	59
3.4.2 Equipment Maintenance Log Summary. .	60
3.4.3 Stereomatrix Interface	61
3.5 Mini-Machine Modeling System	62
3.5.1 General Description.	62
3.5.2 The Data Structure	63
3.5.3 Interface with the Numerical and Output Packages.	65
4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III.	66
4.1 Interactive Picture Processing	68
4.1.1 Show-and-Tell.	68
4.1.2 Atlas Extension.	68
4.1.3 Terminal Communication	69
4.1.4 Scan/Display Devices	69
4.2 Parallel Processing.	70
4.2.1 Texture Analysis	70
4.2.2 Interval Coverings	70
4.2.3 Cartesian Coverings.	70
4.2.4 Vari-valued Logic.	71
4.2.5 PAX Language Support	71

	Page
4.3 Structural Inference	72
4.3.1 Scene Segmentation	72
4.3.2 Inference of 3-Dimensional Model Orientation.	73
4.3.3 SOL Language Support	74
4.4 Applications	75
4.4.1 Cervical Smears.	75
4.4.2 Brain Mapping.	75
4.4.3 Cytospectrometer	75
4.4.4 Remote Manipulation.	76
4.5 Computer Systems	77
4.5.1 IBAL Assembler	77
4.5.2 Operating System	77
4.5.3 TM II.	77
4.5.4 TP, AU, and IOP.	78
4.6 Bibliography	80
4.6.1 External Documents Issued.	80
4.6.2 Logic Drawings Issued.	81
4.6.3 Engineering Drafting Report.	81
4.7 Administration	83
4.7.1 Personnel Report	83
4.7.2 Computer Usage Log Summaries	83
5. ILLIAC IV	84
5.1 ILLIAC IV Assembler (ASK).	85
5.2 ILLIAC IV Simulator (SSK).	85
5.3 Glypnir.	85
5.4 Parallil Assembler (Pandora)	85
5.5 Documentation.	86
5.6 B6500 Service.	86
5.7 Financial Report	87
6. CENTER FOR ADVANCED COMPUTATION	89
6.1 Research Program and Plan.	89
6.2 Major Accomplishments.	89
6.3 Problems Encountered	91
6.4 Fiscal Status.	91
6.5 Future Plans	91
7. SWITCHING THEORY AND LOGICAL DESIGN	92
8. OFFICE OF THE SOUPAC CONSULTANTS.	96
9. MACHINE AND SOFTWARE ORGANIZATION STUDIES	98
9.1 FORTRAN Parallelism Detection.	98
9.1.1 ASSIGN	98
9.1.2 Scanner.	99
9.1.3 IF Tree Processing	99
9.1.4 Statistical Package.	100

	Page
9.2 Weighted Operator, Expression Parsing and Scheduling	100
9.3 Simulation Processor	101
9.4 Text Searching	102
9.4.1 D-Machine Microprogram and Simulator	102
9.4.2 I/O Software	102
9.4.3 D-Machine Assembler.	103
10. COMPUTER SYSTEMS ANALYSIS	105
10.1 Computer Network Modeling.	105
10.2 Center Throughput Analysis :	105
11. THEORY OF DIGITAL COMPUTER ARITHMETIC	107
11.1 Continued Fraction Arithmetic.	107
12. Binary Search Trees of Bounded Balance.	112
13. ADAPTIVE ALGORITHMS FOR ELLIPTICAL DIFFERENCE EQUATIONS	113
14. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS.	115
15. NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES.	117
15.1 Educational Timesharing System	117
15.2 DOS Batch.	119
15.3 GIZMO.	119
15.4 PDP-11 Hardware.	120
15.5 PDP-11 Hardware Test Program Library Monitor	121
16. GENERAL DEPARTMENT INFORMATION.	125
16.1 Personnel.	125
16.2 Bibliography	126
16.3 Colloquia.	127
16.4 Drafting	128
16.5 Shop's Production.	128

1. CIRCUIT RESEARCH

(Supported in part by the Office of Naval Research under Contract
N000 14-67-A-0305-0007, W. J. Poppelbaum, Principal Investigator.)

Summary

Bernard Tse describes details of the bundle generator for the fail-safe Bundle Repeater/Restorer. Trevor Mudge is putting the finishing touches on SABUMA. The APE report of Yiu Wo's details the remotely tunable data receiver that will be installed in each APE (autonomous processing element) machine. PENTECOST now has its dual voltage receiver tube and special pickup whose properties are described by Panigrahi. Jim Cutler discusses a preliminary method of generating a bundle of signals that are both spatially and temporally random. Finally, Ed Pott describes the local control for Telemaze - the remote control system possessing a long (and variable) feedback delay.

M. Faiman - editor

1.1 Fail-safe Bundle Repeater/Restorer (Project No. 21)

1.1.1 Progress in general

The bundle repeater/restorer 3-level switches, which make up the main body of the system, are being fabricated by the electronics shop. The artwork of the printed circuit has been laid out and a prototype should be ready shortly. Each printed circuit card can accommodate the switching of 12 wires, so that all six repeater/restorer stages can be put into 36 cards.

1.1.2 Input circuits

A scheme has been devised which utilizes an analog signal (taken off a potentiometer) as input to drive the bundle of 64 wires selectively. A diagram of the circuit is shown in Figure 1. The number of wires driven by each of the 7 outputs of the counter is shown in Table 1.

Two SN74193's are connected as a 7-bit counter. On the 2^6 count, all 7 bits of the counter are loaded, and in this case all 64 wires of the bundle will be switched on. The outputs of the counter drive the wires of the bundle, and they are also summed at the op-amp to be compared with the input signal.

By using this scheme it is possible to generate a bundle of 64 wires --- an even number of wires, which is necessary since the wires are grouped in pairs.

Bernard Tse

1.1.3 SABUMA (Safe Bundle Machine)

The construction of the project is complete. Two things remain to be done before its performance can be assessed.

First, some suitable power supplies must be found. This is not as straightforward as might be expected, since the three valued logic maps into +2.5V, 0V and -2.5V dc. For testing purposes bench supplies are being used.

Secondly, and more important, a set of testing circuits must be built to facilitate the debugging of the 400 similar multiplier circuits which form the bulk of the machine's arithmetic unit.

The testing procedure can be understood by considering Figure 2. The test circuits T_i are comparators that cause the lamps to light when their two inputs don't agree. Incorporating this subsystem into the project eases debugging greatly and also allows regular checking, so that system performance can be more readily evaluated.

The schematic of a test circuit is shown in Figure 3. When $x = c$ T_1 is off, as is T_2 . Hence point A $\approx$ 0.8V, i.e. T_3 is on, causing lamp L to light. If $x > c$, T_1 turns on, point A goes nearly to ground, turning off T_3 and lamp L. If $c > x$, T_2 turns on and again point A goes nearly to ground; turning off T_3 and lamp L. Thus a faulty multiplier is detected by its corresponding lamp not lighting in any one of the 9 test states.

Trevor Mudge

Figure 2. SABUMA Multiplier Tester.

Figure 3. Lamp-Driving Comparator.

1.2 APE (Project No. 25)

1.2.1 The Remotely Tunable Data Receiver

Although receiver design is thoroughly understood, it remains a very time consuming process to design and develop a good one for a specific application. During the past quarter, much of the effort was devoted to the design and construction of a remotely tunable data receiver for the APEs. The basic requirements of this receiver are as follows: (1) micro-power consumption, with a maximum drain of 2.5 mA from a 6 volt source; (2) remotely tunable for receiving pulse width modulated RF from any of the eleven channels within a frequency band of 5.3 MHz; (3) having a bandwidth of at least 10 kHz to resolve a 100 μ s input pulse; (4) having a sensitivity of 500 μ v.

Various types of receiving principles and techniques have been investigated for the design. It was found that the heterodyne receiver is the most suitable type because of its high sensitivity, high selectivity, and simplicity in tuning. However, it is also well known that image frequency response and spurious response are the inherent shortcomings of this type of receiver, as indicated in Figure 1. Although more sophisticated receiver designs, such as the superheterodyne or double conversion heterodyne types, could overcome these shortcomings, they would easily exceed the limit on the power consumption. Happily enough, proper reception with a heterodyne receiver is still possible if channel frequencies are assigned carefully, so that no channel frequency exists near a spurious response frequency. For the APE machine, there are eleven channels spaced 530 kHz apart between 15.0 MHz and 20.3 MHz. With a 10.9 MHz intermediate frequency, the closest that a channel frequency approaches a spurious response frequency occurs when the receiver is tuned to 15.0 MHz. In this case, the closest significant

Figure 1. Image Frequency and Spurious Response of a Heterodyne Receiver.

spurious response frequency is given by:

$$f_S = f_{IN} + \frac{f_{IF}}{2} = 15.0 + \frac{10.9}{2}$$
$$= 20.45 \text{ MHz}$$

where f_S = spurious response frequency

f_{IN} = input carrier frequency

f_{IF} = intermediate frequency

Because the highest channel frequency for the APE machine is 20.3 MHz, as mentioned above, this spurious response f_S is still 150 kHz away from the closest channel frequency. If the pass band of the receiver is less than twice this amount, the spurious response f_S will not have serious effect. Furthermore, 10.9 MHz is within the tuning range of the commercially available 10.7 MHz intermediate frequency transformer for FM receivers. Therefore, 10.9 MHz is chosen to be the intermediate frequency for the data input receiver of the APE, and the readily available 10.7 MHz IF transformers are employed in the design. After lots of design and development work, a prototype receiver with the schematic shown in Figure 2 has been successfully built. The tuning voltage comes from a digitally controlled voltage source that is set by the contents of the program instruction. Also, this voltage source is designed to have a negative temperature coefficient so that it will compensate for the positive temperature coefficient of the tuning coils and diodes in order to provide a better overall temperature stability.

Figure 2. Remotely Tunable Heterodyne Receiver.

The prototype receiver has passed various tests and mass production of the data receiver for the APEs is now underway.

Yiu Wo

1.3 PENTECOST (Project No. 31)

1.3.1 Penetron

The Penetron tube has been installed in the monitor. It was possible to change the color from red to white by varying the anode voltage from 11 kV to 16 kV and simultaneously changing the focus voltage from 1500V to 2400V. A voltage switch that can change the focus voltage in less than 200 μ s has been assembled and tested. A similar switch is being assembled for the anode voltage.

1.3.2 Plumbicon

The red-extended Plumbicon tube (Amperex 16xQ-RIG) has been installed in the GE-26 camera. The resolution of the picture was inferior to that of the vidicon tube (7735B) that came with the camera. A filter wheel made of two sections of red and green wratten filters sandwiched between two plastic plates was made. The filter wheel is fixed on the spindle of a hysteresis synchronous motor that turns at 1800 rpm. By placing the moving filter wheel in front of the camera, it will be possible to register alternate red and green fields on the Plumbicon tube.

G. Panigrahi

1.4 Ergodic (Project No. 39)

1.4.1 Summary

This quarter was used in studying methods of generating an ergodic process of the kind described in the last report, namely, a bundle of wires whose signals are both spatially and temporally random. It is to this

project's advantage to keep this ergodic generator as simple as possible in terms of circuitry. The basic block of a generator is a circular shift register which has the properties of a ergodic generator with the exception of randomness. With this fact in mind an investigation is being made of adding randomness to a circular shift register by three methods:

- 1) Wiring the outputs of the circular shift register in a random fashion.
- 2) Inputting the circular shift register in a random fashion.
- 3) A combination of 1) and 2).

1.4.2 Status

The experimental generator has a 16-bit circular shift register, a 16-bit latch, switches as inputs to the shift register, and timing circuitry. Basically, after 15 cycles the timing circuit can be programmed to shift the bits in the shift register from 1 to 15 places in the 16th cycle. The block diagram and timing diagram for this circuit are shown in Figure 1.

1.4.3 Future Work

Experiments are planned for the above generator in which the randomness of the process will be considered. A generator using the minimum amount of circuitry will be the result.

..... Jim Cutler

Figure 1. Experimental Ergodic Generator and Timing Diagram.

1.5 Telemaze (Project No. 41)

1.5.1 Project Summary

The object of TELEMAZE is to develop a simple system with an adjustable feedback delay, which could control the trajectory of a distant vehicle. The condition which is characteristic of this system is that the feedback delay time is much greater than the vehicle's movement time. In this system, control of the vehicle is obtained by steering an "image" of the vehicle through a "model maze" at the local station.

1.5.2 Square Input Recognition

Several types of data tablet techniques have been investigated which would energize one of the 16 by 16 matrix squares. The currently proposed method is to use a linear array of light emitting diodes (LED) and photodetectors, mounted on the faceplate of the display cathode ray tube (CRT). It has been suggested that infrared LEDs should be used instead of the visible type. This could reduce interference from ambient light, and visible light from the CRT, by filtering the input to the photodetectors.

1.5.3 Path Generator

The function of the path generator is to sense the input requests "steering" the vehicle through the "model maze". The block diagram is shown in Figure 1. Under static conditions the infrared light will traverse the CRT face and fall on the photodetectors. A request for the vehicle to move from one square to a new square (i.e., path generation) is initiated by placing the index finger on the CRT square desired. The presence of the finger is detected by breaking one of the array of light beams. To establish a unique square, it is necessary to consider the model of the hand. For a right-handed person, the index finger will serve as a pen to write upon the CRT face. The X array will scan from left to right and the Y array

Figure 1. Path Generator.

from top to bottom. To prevent the arrays from sensing the rest of the hand (which may be allowed to rest upon the faceplate), the completed array will be disabled for remaining cycle. This technique will establish a unique square found beneath the fingertip. Once either coordinate has been detected, it is retained in the respective latch. After both coordinates are found the scan is completed, and then the coincidence detector will generate a pulse, allowing them to be processed.

1.5.4 Input Square Processing

The function of the input square processor is to examine the coordinates of the square entered and, if it is a valid request, store it in the buffer memory. To insure a continuous trajectory this square is examined against the last square stored in the buffer memory. If square A is considered to be the square in which a remote vehicle rests, the condition imposed upon the controller is that the vehicle cannot jump randomly, i.e., cannot skip squares. The vehicle is allowed to move one square in the X or Y direction but not simultaneously. An example to show this is shown in Figure 2a. It is requested to move from square A to B. To realize the shortest path from square A to B there are only two choices allowed.

To implement this design, the new square is allowed to pass into the new square register as in Figure 2b. The last valid square is retained in the last square register. These coordinates are compared for equality and to see if either coordinate represents a displacement of +1, while the other has remained constant. Then the square is stored into the buffer memory, and transmitted to the remote vehicle.

1.5.5 Video Generation

The basic function of the video generator is to display the "model maze" (1 bit) stored in the maze memory and the trajectory information (1 bit) stored in the local buffer memory. The basic display considered consists of a black and white television monitor. The video square generator functions to sequence through both the "path" and "maze" memories and generates the necessary enable signals for the video logic. To display these two bits of information on a black and white CRT, 4 different levels are needed. It is proposed that the black and white level and 2 intermediate shades of gray be used, which will be generated within the "video level" logic block shown in Figure 3.

Edward J. Pott

Figure 2a. Possible Legal Paths from A to B.

Figure 2b. Square Processor (X Component).

Figure 3. (Local Station) Video Block Diagram.

2. HARDWARE SYSTEMS RESEARCH

(Supported in part by the Atomic Energy Commission under Contract US AEC AT(11-1) 1469, W. J. Poppelbaum, Principal Investigator.)

Doug Sand's OLFT report deals with cooling the crystal through its Curie temperature while writing, and describes modifications to the design of the optical subsystem. The BLAST report is probably the last of the series to appear, with complete details of the project to be found in Larry Wallman's thesis. Trevor Mudge updates the picture of Sematrix, much of whose design is complete, but still lacks in a few mechanicals. Dick Blandford now has a 52-fiber array working on LINDA and he also discusses other phases of the project that are under investigation. RASER - Random-to-Serial Converter - the device which accepts random coordinates for display and outputs them in raster mode, now has a sync generator for its CRT, courtesy of Tak Katoh. Shiv Verma describes clean up work on Stereomatrix, which should be ready shortly. The Scantrix report by Sik Yuen discusses the video line-at-a-time buffer, a D/A for gray levels and a test for contrast sensitivity of the human eye. Finally, Dev Bose describes the nature and implementation of the stimuli, as well as the world model for FROG.

M. Faiman - editor

2.1 LASCOT (Project No. 09)

Improvements have been made to the scanner and modulator, and a laser has been ordered.

M. Cooper

2.2 OLFT (Project No. 12)

2.2.1 Operation of Intermediate Cooled Light Valve

The intermediate cooled light valve has been operated for several hours, without observable difficulty, in the range of -45 to -55°C , which includes the Curie temperature T_C (about -52°C). As T_C is passed during cooling, ferroelectric domains form which are in general of random shape and field orientation; however, if a surface charge pattern has been written at a temperature slightly above T_C , it is possible to form a single domain covering the whole crystal when T_C is reached. Near T_C the charge storage times are much greater than one hour (the maximum observation time) and pattern modulation characteristics appear uniform over the crystal area, except at domain boundaries which are sharply defined in the optical image. Unfortunately, the maximum resolution yet achieved is about 250 line-pairs. Further system refinement and operational familiarity is required to determine the resolution limit, which is probably constrained by the electron-beam spot size. In addition, various electronic circuits are being modified to improve operation.

2.2.2 Optical Configuration for Intermediate Cooled System

Figure 1 is a diagram of the optical configuration now being constructed for the intermediate COLFTAR system, using the transmission-mode light valve. The general placement of light valve, Fourier-transform lens, and vidicon is similar to that of the reflex light-valve system. The single beamsplitter B generates the reference beam and object beam from the

Distance $\overline{XM_1L} = \overline{LM_2V}$

B = Beamsplitter

P = Polarizer

X = Electro - Optic Crystal

L = Fourier - Transform Lens

Figure 1. COLFTAR Optical System.

collimated laser beam and further serves to combine the reference beam and Fourier transform prior to detection at the vidicon faceplate. The polarizers P_{11} and $P_{\perp}$ on either side of the light-valve crystal are used to extract the amplitude-modulated component of the light-valve output, as well as to block the unwanted path $\overline{M_2BX}$ through the beamsplitter. Polarizer P_{θ} , oriented at angle θ to P_{11} , say, is used to adjust the relative amplitudes of the reference beam and Fourier transform to obtain optimal "beam-balance". For clarity, several additional components are not included in the diagram.

This configuration is designed for the transmission light valve, but can be modified for the reflex light valve by including another beam-splitter. Thus the majority of the system can be built and tested concurrent with the construction of the reflex light valve.

Doug Sand

2.3 BLAST (Project No. 19)

2.3.1 Final Status

During the past quarter the work on BLAST was completed. The system as it stands now is a non-feedback configuration, where minor adjustments are required periodically. The system itself is satisfactory in that it does show this type of stereo display to be acceptable. Considerably more work would be required to develop a system compatible with or nearly compatible with broadcast TV.

The final report (thesis) on BLAST will be issued in the near future.

Larry Wallman

2.4 SEMANTRIX (Project No. 24)

This I/O device for locating and positioning colored blocks on a 2-dimensional surface comprises three subsystems.

1) Block location

All that remains to be completed is the construction of a suitable power supply for the near field generator - the induction mechanism that supplies power to the blocks. The blocks themselves have also to be built in large numbers. At present there are only a few used for test purposes.

2) Electro-mechanical hand-arm system

Most of the mechanical parts for this have been fabricated. The torque motors have been installed; however, some suitable flexible drive couplings have to be manufactured. The grip on the end of the hand has to be designed and built.

3) Logical interface with computer and teletype

Design is complete and the PC boards are being generated by the layout department. These are being debugged as they arrive. Control switches and cabinets for the racks have still to be decided upon.

Trevor Mudge

2.5 LINDA (Project No. 28)

2.5.1 Summary

LINDA (LINE Drawing Analyzer) is a pattern recognition project which analyzes simple line drawings by looking at their component parts. A given line drawing is first separated into its simple parts (triangle, square, etc.). Each part is then identified by sweeping it across a linear array of photocells on a CRT. The line drawings are displayed by means of a flying spot scanner. Identification of the component parts together with information concerning their relative position permits the recognition of a small vocabulary of drawings. Details of the project including a discussion of the photocell array and the identification procedure have been given in previous reports.

2.5.2 Project Status

The photocell array consisting of 52 fiber optic light pipes arranged in a line to transmit light to the photocells is essentially complete and its operation is satisfactory. Some difficulty has been encountered from ambient light 'noise' resulting from the fiber optics mechanical arrangement, although this is not serious. The present array is in working order but a new array with a different mechanical arrangement will be built for improved operation.

Work is underway for the development of the logic which will allow a line drawing to be scanned, separated into its parts and identified. This work is divided into three areas:

1. Separation of the line drawing parts.
2. Shading mechanism (parts of a line drawing must be shaded to be identified - this is discussed in the quarterly report

for July, August, and September, 1971).

3. Identification of the complete figure.

Some progress is being made in area 1. The shading mechanism should be ready for building in the next month. Work in area 3 has only been 'chalked out'.

Dick Blandford

2.6 RASER (Project No. 29)

2.6.1 Sync Generator for CRT Monitor

A 31.5 kHz crystal controlled master oscillator is divided by 2 to generate horizontal sync pulses and by 525 to generate vertical sync pulses. This method then generates $525/2 = 262 \frac{1}{2}$ horizontal pulses for each vertical sync pulse and provides a stable fixed interlace.

The divide by 525 is broken up into subdivisions. Typical subdivisions are 3, 5, 7 which may be cascaded ($3 \cdot 5 \cdot 5 \cdot 7 = 525$) to yield a frequency division of 525.

In Figure 1 C-2 and C-3 are decade counters (SN7490), and each of them is internally interconnected to provide a divide-by-two counter and a divide-by-five counter. C-1 and C-4 are divide-by-twelve counters (SN7492), and each of them is internally interconnected to provide a divide-by-two counter and a divide-by-six counter.

The master oscillator is fed to inputs A and BC of C-1. Then output QA (31.5 kHz divided by 2) is fed to a one-shot OS-3 to produce a 4.76 μ s horizontal sync pulse. One-shot OS-6 provides horizontal blanking. However, OS-3 is controlled by the output of one-shot OS-1, which delays the pulse 1.5 μ s for the "front porch".

ALL TRIM POTS ARE .6W
ALL RESISTORS ARE 1/4W
ALL DIODES ARE IN4151
FUSE IS 1A

- 1 CD4001E COSMOS
 - 1 SN7400
 - 1 SN7404
 - 1 SN7408
 - 2 SN7490 DECADE COUNTER
 - 2 SN7492 DIVIDE - BY - TWELVE COUNTER
 - 4 SN74123
- TOTAL 12 IC'S

Figure 1. Sync Generator for RASER Monitor.

The output Q_C of C-1 is 31.5 kHz divided by 3 and is fed to input BC of C-2. Q_D of C-2 is 31.5 kHz divided by 15. The output of G-1 is 31.5 kHz divided by 525, and this clears the counters C-2, C-3 and C-4.

Six horizontal equalizing pulses are obtained during the interval when Q_A of C-2 is 0, since input A is 31.5 kHz divided by 6. C-3 similarly allows six vertical sync pulses to be obtained. Both C-2 and C-3 are cleared at the end of interlace.

Tak Katoh

2.7 STEREOMATRIX (Project No. 30)

2.7.1 Transformer

The coordinate transformer has been operating satisfactorily. The cursor multiplexing part has also been tested. The driver-receiver circuits have been tested. The system control is being wired. The automatic scaling circuits are tested.

2.7.2 Coefficient Generator

The circle generator of the cursor part of this unit has been redesigned and tested. The new circuit is shown in Figure 1.

The rack wiring, documentation and card drawings have been completed. It has been decided to redesign some parts of the system to accomplish the main features of this unit.

Shiv Verma

2.8 SCANTRIX (Project No. 35)

2.8.1 Introduction

During the past quarter work was done on the output stages of Scantrix. A Digital-to-Analog converter (DAC) was built, and the buffer

Figure 1. Variable Size Circle Generator.

cards were designed. Human eye response to linear and exponential level changes of brightness were studied. All these will be discussed below.

2.8.2 DAC

Refer to Figure 1 for the circuit configuration. The DAC simply consists of four NAND gates corresponding to four bits of gray levels on the output. The 741 and the resistors combined together function as an adder which forms the weighted sum of the input signals as the analog output.

2.8.3 Buffer

The system comprises two 128 x 4-bit buffers. The buffers are built with SN7495s operated in the right shift, serial-to-parallel mode. Refer to Figure 2 for the circuit.

2.8.4 Human Eye Response to Brightness

It is known that human eyes are more sensitive to brightness changes when the objects being viewed are relatively dim than when they are very bright. To test this an amplifier giving exponential output was designed. An LED was directly fed by the DAC while a second LED was connected to the DAC through the exponential amplifier. By comparison, it was observed that the LED with the exponential input gave a better contrast in brightness levels especially at high intensities. Refer to Figure 3 for the circuit.

Sik Yuen

2.9 FROG (Project No. 36)

2.9.1 Project status

The design of the World Model is still in progress. Advances made in this direction and thoughts given to the system as a whole are summarized in the following sections. In the next quarter, a simulation of the scheme is planned.

- 1 N5741V SIGNETICS OP AMP
- 2 MC889P MOTOROLA HEX INVERTER
- 3 SN7400N QUAD 2 INPUT POS. NAND GATES

Figure 1. 4-bit DAC.

Figure 2. Buffer Card Block Diagram.

Figure 3. DAC with Linear and Exponential Amplifier Outputs.

2.9.2 The Scheme

FROG is a movable block which is capable of going from one point to another in a bounded flat region. The entire area is divided into a number of regions characterized by different textures. FROG moves back and forth between several fixed locations under on-line control of a computer. The block is radio controlled. One of the locations in one of the regions is a safe resting place for FROG. Whenever it is alarmed or it encounters a predator it takes refuge at this location. At other locations any one of a fixed number of bugs (food with varying taste) and predators is presented to FROG through random selection. In order to feed on the desirable stimuli and to avoid the dangerous ones FROG has to go back and forth between the locations.

The system may roughly be divided into the following parts:

- 1) The environment or the world of FROG.
- 2) The movable block or FROG.
- 3) The control unit - to act as an interface between the computer and FROG.
- 4) The computer programs, such as
 - a) Motion control program.
 - b) Stimuli generating program.

2.9.3 The Stimuli

The cohabitants of FROG are some bugs (edible and inedible) and predators. The machine learns to discriminate between them by means of their visual characteristics. According to Lettvin, Maturana, et. al. in "What the Frog's Eye Tells the Frog's Brain",¹ the retina of the frog contains

¹J. Y. Lettvin, H. R. Maturana, W. S. McCulloch and W. H. Pitts

Proceedings of the IRE, November 1959, pp. 1940-1951.

four groups of fibers which break down every image into four main characteristics which are then transmitted to the brain. The characteristics are:

1. Sustained contrast: the presence of a sharp boundary, whether it is moving or still, with much or little contrast.
2. Net convexity: whether or not an object has a curved boundary, whether its movement is smooth or jerky.
3. Moving Edge: whether the boundary detected by the sustained contrast detectors is moving or still.
4. Net dimming: how large the object is, how much dimming occurs in the field.

These are the characteristics from which FROG infers the good or bad taste of a bug or the pain it receives from a predator.

Each bug has a value n ($0 < n < 1$) for its good taste and its bad taste is put as $1-n$. A good tasting bug is one whose good taste is greater than its bad taste. A bad tasting bug is one whose bad taste is greater than its good taste. A bug is assumed not to inflict any pain. By contrast, the predators it encounters inflict pain to a varying degree.

2.9.4 The World Model: Structure of the Memory

The World Model is essentially an adaptive fitter, such that when the input sensory data and the somatic signals are presented to it a course of action for FROG is decided. The world model consists of a 'memory' where the experiences with different stimuli are stored, followed by a 'memory integration unit' (MIU), which is to handle the various memory outputs properly on presentation of a stimulus and to make them available to a 'decision unit' (DU), which decides on the mode of activity.

Structurally the memory of FROG is divided into several parts (memory planes) equal in number to the number of regions in the environment. Experiences in the various regions are stored in separate memory planes. The memory plane for each region is again divided into three parts. The first is the good taste memory (GTM) which associates the distinguishing visual characteristic of each bug with its taste. The second is the bad taste memory (BTM) which associates the visual characteristics of bad tasting bugs with their tastes. The third is a predator memory (PM) which associates the visual characteristics of a predator with the amount of pain it inflicts.

Dev Bose

PUBLICATIONS

Edward E. Carr, "A Synchronization System for a Television Bandwidth Compression Scheme", Department of Computer Science Report No. 483, University of Illinois, Urbana-Champaign, October 1971, (Master's thesis)

Richard T. Cheng, "Coefficient Generator and Cursor for the Stereomatrix 3-D Display System", Department of Computer Science Report No. 484, University of Illinois, Urbana-Champaign, October 1971, (Ph.D. thesis)

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

C00-1469-0199

2. TITLE

Quarterly Report - October, November, December

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

W. J. Poppelbaum
Principal Investigator

M. Faiman, Editor
Associate Professor of Computer Science

Organization

University of Illinois
Department of Computer Science
Urbana, Illinois

Signature

W. J. Poppelbaum

Date

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

3. SOFTWARE SYSTEMS RESEARCH

Most work on the 360 system this quarter went toward finishing existing projects, while the PDP-8 group has concentrated on adding features to their basic system to make it more useable. The delivery of a new disk for the PDP-8 has been delayed until March due to a misunderstanding in the features required of it.

3.1 Numerical Processes

3.1.1 Ordinary Differential Equations (R. L. Brown)

In adopting the subroutine DIFSUB for use on the PDP-11/20 with DOS, several problems were encountered. First, DIFSUB, as compiled and assembled by DEC software, is much too large to fit in core (20K, 16 bit words). Second, both single and double precision FORTRAN floating point software are at present unreliable, and more importantly for the development of the project, time consuming at execution.

Several solutions for the first problem have been considered. First, only one of the two methods for solving differential equations in DIFSUB will eventually be used. The Adams method is currently being used since it is simpler, doing no matrix inversion. This reduces the program's size somewhat, but further reduction is necessary. DIFSUB has, therefore, been reduced to an essential control routine linked with the MAIN program (which calls DIFSUB), the CONTROL section, and the elements of the Floating Point Package Library and by Extended Arithmetic Library that are necessary for execution and which are kept in high core above the program (see Figure 1).

These subroutines are kept on disk as contiguous overlay files which can be called into a Subroutine Overlay Buffer so that they are properly positioned relative to the program elements they were originally linked to. To accomplish this transfer a special subroutine was written which appears to the CONTROL element to be its only subroutine. Calls to all eight subroutines are of the form

CALL CONECT(#, ARG1, ARG2, ...)

where # is an integer specifying which subroutine overlay is actually needed, and ARG1, ARG2, ... is a variable length argument list for that particular overlay. CONECT then brings the overlay into the buffer and transfers control to it. CONECT is essentially invisible to the program logic and occupies very little core.

The second problem has two solutions. First, DEC plans to update its FORTRAN software, the present version being somewhat primitive and much of the program will probably be reduced to assembly language at some time anyway. The execution time problem remains, and since single precision is more than half as accurate as double precision with the PDP-11/20 software, the program is presently in single precision form.

A version of DIFSUB in this form has been assembled and is being debugged at present.

Figure 1. Core Map of DIFSUB on the PDP-11/20

3.1.2

Sparse Matrix Inversion (J. S. Deogun)

CORRECTION: The following diagram which appeared in connection with the eigenvalue problem on page 57 in the 3rd QPR 1971 is incorrect.

Figure 2

It should be as follows.

* R_B indicates Rank of B

The previous organization of the eigenvalue problem has been dropped as all the functional routines in the SPARSE package have been changed. The present changes are the result of considerable improvement in the SPARSE package. The changes are of an organizational and programming nature, the basic techniques remaining the same.

It is of importance to mention some of the features incorporated in the new version of SPARSE which make it easier to handle the eigenvalue problem. The present version of SPARSE was developed with the eigenvalue problem in mind. Various routines are designed in a way to allow handling a matrix or its transpose without any changes to the routines, by changing values of some parameters. This feature helped considerably in simplifying the eigenvalue problem. Previously, in the eigen-problem

program, a different version of some routines (SPARSE, PIVOT SELECTION, etc.) was needed to work on the transpose of a matrix.

The present version of SPARSE is much more flexible and easier to modify to individual requirements. SPARSE has been broken into several modularized subroutines whose functions are essentially self-contained, and common to both the eigenvalue problem and the standard differential equation solver. Only the main subroutine, SPARSE, requires modification.

This quarter, time was spent on studying the new version of SPARSE. All the necessary changes in SPARSE routines, required by the eigen problem have been completed. A new main program was written which uses the new version of SPARSE. Currently, the program is being run with small test examples. We expect it to be completed very soon.

Following is a block flow diagram of the relationship between the new routines.

ROUTINES

3.1.3 The Steady-State Package (B. van Melle)

Many problems arise in which the numerical formulation does not in itself prevent variables from assuming illegal values. This can cause difficulties, e.g. trying to extract the square root of a negative number. Work has proceeded this quarter on providing for boundary conditions in numerical problems. The user may specify an upper bound and/or a lower bound on certain variables. These are coded into the vectors LIST and P2LM to be used later by the new subroutine RANGER. This routine simply examines the given variables to see if they are within the proper boundaries; if not, they are adjusted so that they lie a specified distance inside the bounds. The algebraic-equation solver DIFMF3 has been altered to make use of this routine. At each step, RANGER is called. A fairly large value is given to the parameter specifying the distance from the boundary to move a variable that has strayed out of bounds; thus, the errant variable is hopefully put on the right track. If too small a distance is specified, the variable sometimes has a tendency to repeatedly go out of bounds. The tests run thus far indicate that this is a reasonable strategy. Once the restricted variables are moved out of the dangerous range, the solution is obtained with much the same speed as before on problems with no such restrictions.

3.1.4 Plot Package (W. Chung)

During this quarter the plot package has been rewritten and tested together with a small linkage routine LINK. The calling sequences for the plot package and LINK will be

```
SUBR(N,NL,M2,T,G,Y,YL,I1,IPNT,TEND,ITEXT)
```

```
LINK(N,NL,M2,T,G,Y,YL,I1,IPNT,TEND,IDRAW)
```

LINK will call SUBR dynamically in such a way that the load module consisting of SUBR and CALLER is loaded at the moment of initial plot request from the user and called for every time step in DIFSUB until it is deleted by a no-plot request. DIFSUB passes the values of Y, YL, T, and TEND to SUBR which saves those it determines are needed, and produces graphical output for one time interval. At first, the number of plotting

points is specified by a user. But the actual number of points plotted is generally less than that given by the user, since the time step of integration is controlled independently by DIFSUB. This should result in the saving of computation time and the accuracy of the results as DIFSUB is not reinitialized at each time interval. The plotting may be stopped at any time by the user, the coordinate of the output can be shifted down or up easily, and the values of Y, YL at any time T can be computed and presented to the user by an interpolation during the interactive plotting phase. Different variables for different time limits (TEND) at different numbers of points (IPNT) may be plotted by returning control to P2 (DIFSUB). For this case, the plot package is deleted and reloaded. These features provide flexibility, modularity, and economic use of memory (the plot package resides in core during only the plot phase) required by the graphics system.

Further considerations will be given to graphical input and output which will provide a way of analyzing the numerical computation and allow more user control over the drawing.

3.1.5 Elimination (J. Frazao)

Beginning development in October, a new program, PEE (Partial Evaluator for Equations) was added to the Graphics Simulation and Modeling System.

Designed to interface between the existing WEED and COMP programs, PEE travels recursively down each of the equations passed by WEED and compiles a list of all terminal nodes encountered. At each node the lists associated with its various operands are combined into a single list, so that upon returning to the top of an equation, a single list remains, containing the necessary information about each of the variables in the equation. Each list entry is then used to create a single element in a sparse matrix array, called the MX-array. As each equation is processed, a corresponding row is created in MX. When all equations are processed, MX and its associated pointers are converted into FORTRAN-compatible data, and control is passed to COMP.

Each variable-list entry contains four fields as shown in Figure 2--one for the variable name; a type-field (indicating whether the entry is a constant or a S1, S2, L or M-type variable); a derivative-type field (indicating whether the partial derivative of that variable with respect to the given equation is a constant⁽¹⁾, S1-dependent, S2-dependent or M-dependent⁽²⁾; and a constant-pointer field (which points to a constant in a table whenever the type-field or derivative-type-field is constant).

Whenever a +, -, *, / or unary- node is encountered while compiling the variable-list, the left and right⁽³⁾ lists are scanned, and new types, derivative-types and constant-pointers are evaluated for entries in both lists. For all other operations and for the right list of a / node (denominator), all entries have their derivative-type set to four (non-linear). Since a derivative-type can only be increased (or at most remain the same) as lists are processed, all further processing for a given entry is bypassed as soon as its derivative-type becomes four. After the node is processed, the resultant list becomes simply the updated left list concatenated with the updated right list.⁽⁴⁾

In building the MX-array, only variables whose types are three or four (L and M variables) are used from the variable-list, all other entries being ignored. Each row in MX corresponds to an equation, and each column to a variable. The variable names are rearranged so as to have all the M-variables first. Associated with the MX array are a PW array (a table of single-precision constants with one entry for every element in MX), two row vectors (IRN and IRP), and two column vectors (ICN and ICP), each of which contains one entry per row (or column) in MX. See Figure 3.

(1) If the partial of a variable is a constant, it is necessarily non-zero.

(2) The partial of any variable cannot depend on L-variables only (from the definition of linearity).

(3) Left list only, in the case of unary-.

(4) Whenever multiple entries for a single variable are present, the first entry for that variable is treated as the significant one.

Figure 2. Variable-List Format

4/5 variable #

var. #

Figure 3. MX-Array

IRN and ICN give the number of elements in each row (column) of MX, and IRP and ICP contain pointers to the first element of each row (column). As each new MX-element is created, its corresponding IRN and ICN counts are increased by one, and if it is the first element in a row and/or column, the associated IRP and/or ICP pointers are set.

Each MX-element contains six halfword fields as shown in the blow-up of Figure 3. A zero in RNEXT (or CNEXT) indicates the last element in that row (or column). Constants are placed in the PW-array when their corresponding MX elements have derivative-type 0; no entry is made in PW otherwise, even though space is allocated.

Finally, all pointers are converted to indices, as used in a FORTRAN dimensioned array, and PEE performs an XCTL to COMP.

3.1.6 Matrix Inversion Package (A. Whaley)

The rewrite of SPARSE, mentioned last quarter, appears to have been completed successfully. A subroutine (CHECK) was written to monitor the matrix inversion as it proceeds, and at present it makes no complaints during the inversion of a 6 x 6, 16 x 16, 17 x 17, and 32 x 33 matrix. These are the only problems that have been tested so far.

SPARSE has been modularized into packages for pivot selection, pivot row division, and forward elimination. SPARSE itself handles righthand side operations and back substitution. Afterward, it compiles code to keep all unpivoted variables at zero, corresponding logically to the addition of a sufficient number of equations to make the matrix square.

COMPIL is invoked by the inversion package with the code for a machine instruction to be generated, and the subroutine number in which it is placed. Also supplied are the array and index for the operand and a location to be used in executing the instruction immediately. Execution allows CHECK to locate programming errors by continually verifying that $Ax = b$ as A becomes an identity matrix, and b becomes x.

Conditional operations are provided that COMPIL may bypass generation of certain code if stated operands are already contained in floating point registers. A scheme is also used for optimizing the use of general registers, although this will only be useful for matrices of large rank. Efficiency of core utilization is also provided for. Storage is provided in 1024 byte increments to each subroutine on a demand basis.

The inversion package has been integrated into the system, together with a symbolic version of SETUP called the Partial Evaluator of Equations, and all appears to work perfectly. Changes have been made to all routines in the numerical integration program to use the new inversion subroutines, and success in this area appears to be quite close.

3.2 Non-Numerical Packages

3.2.1 Item Analysis (J. Koch)

Item analysis continued to operate without errors this quarter. The equation parsing routine, PARSE, was modified to generate an extra equation and variable for each occurrence of the LOG and SQRT functions. Both the natural logarithm and square root functions have points of singularity and complex results. When searching for an initial steady state, the "random" initial values used could cause these functions to cross these points of singularity and abort the analysis. But with the addition of extra equations and variables, the arguments of LOG and SQRT can be forced to be initially positive. Item analysis now modifies user's equations as follows:

User's Equation

$$R = \text{SQRT}(S) + T$$

$$W = X * \text{LOG}(Y)$$

$$X = \text{LOG}(\text{SQRT}(Y)) / Z$$

Output Equations

$$\begin{aligned} R &= \text{KOTCH001} + T \\ \text{KOTCH001} * \text{KOTCH001} &= S \end{aligned}$$

$$\begin{aligned} W &= X * \text{KOTCH002} \\ \text{EXP}(\text{KOTCH002}) &= Y \end{aligned}$$

$$\begin{aligned} X &= \text{KOTCH003} / Z \\ \text{EXP}(\text{KOTCH003}) &= \text{KOTCH004} \\ \text{KOTCH004} * \text{KOTCH004} &= Y \end{aligned}$$

Because extra equations are generated, Item Analysis now provides an extra area where they are placed by PARSE. Two extra parameters, PTR and END, point to this area and two more possible bits were added, MOREON and EXTRAS. The complete parameter list for PARSE now reads:

PARSPARM	DS	OF	
	DS	X	
PADDR	DS	AL3	ADDR OF WHAT I AM TO SCAN.
PEND	DS	A	END OF WHAT TO SCAN.
VLOC	DS	A	LOC OF LOCAL VARIABLES.
VGLB	DS	A	GLOBAL.
VPAR	DS	A	PARAMETERS.
VEI	DS	A	E/I NAMES.
VLH	DS	A	LH SIDE VARIABLES.
VCON	DS	A	CONSTANTS.
FPES	DS	A	WHERE TO GET FREE SPACE.
LASTEON	DS	A	ZERO WHEN STARTING NEW EQN TREES.
COUNT	DS	A	EQN NUMBER IN BLOCK.
S&WORK	DS	A	ADDR OF WORK AREA FOR MESS. TO PIP8.
ERRLIST	DS	A	ADDR OF ERRLIST DCB FOR PARSE ERRORS.
*			POSSIBLE FLAG BITS:
VIHBIT	EQJ	X'80'	ONLY 1 VARIABLE ON LEFT SIDE OF =
HALTBIT	EQJ	X'40'	ON IF PARSE HALTED.
ERRBIT	EQJ	X'20'	ON IF ERRORS DETECTED BY PARSE.
MOREON	EQJ	X'08'	ON IF MORE EQNS GENERATED BY PARSE.
EXTRAS	EQJ	X'04'	ON IF PARSING THOSE EXTRA EQNS.
PFLG	DS	X	WHERE THE ABOVE FLAGS ARE TURNED ON.
TRANS	DS	AL3	ADDR OF EBCDIC TO ASCII TRANS. TABLE.
FILE	DS	A	ADDR USED BY XAPPEND IN PARSE
PTR	DS	A	ADDR OF 1ST AVAIL POS FOR EXTRA EQNS
END	DS	A	ADDR OF END OF ABOVE AREA

3.3 Graphical Remote Access Support System (GRASS)

Several steps were taken this quarter to improve the ease of use and versatility of the system. Suggestions from users are being implemented as they are received, with the result of a considerable improvement in the case of performing several of the basic drawing functions (see 3.3.2, 3.3.4).

Program development on the local system has been severely impeded by equipment unreliability, especially with regard to the disk, which has never performed reliably since it was returned from Data Disk in late September. Debugging has been confined to the times when the disk is operational, which ranges from several hours to several days at a time.

Because of the disk problem, a new disk has been ordered (see 3.4), but until its arrival, continued difficulties in program development are anticipated.

3.3.1 Disk Monitor System (R. Haskin)

The DMS has remained unchanged this quarter. It has proved as reliable as the disk all quarter.

Acquisition of the new disk poses several problems with regard to the DMS and, consequently, program development activities on the local system. These are caused by the fact that the Systems Industries disk controller employs 128 word sectors for addressing the disk, and 129 words (128 data + 1 link) are required by the DMS. Three possible solutions are seen.

1. Use 129 words out of each two contiguous sectors. The advantage is that it is fast, but there is the disadvantage that only half the disk space is used. The capacity of the disk will be decreased to $\frac{4}{5}$ of its current value; there will be space for 1024 rather than 1280 blocks.

2. Have the 128 data words reside in each sector, one sector per block, and map the link word into one of eight sectors containing

only link words. When a block's link word is needed, the associated link word block is read into core and the proper word fetched or stored. If no optimization were required, this would degrade I/O speed by a factor of 2 on read and 3 on write (read link block, update link word, write data block, write link block). If the integrity of a link block which is in core could be assumed (often a tenuous assumption), the degradation could be considerably reduced. In any case, the new software necessary to implement this method is considerable, making it a doubtful choice.

3. Implementation of the new DEC PS/8 monitor to replace the DMS. The good points are that the system programs are improved, the assembler is compatible with PALD, and the system uses 128 word blocks. The bad point is that considerable effort would be involved in conversion between the DMS and PS/8. All files on all DECTapes would have to be reformatted, and several system programs, particularly LD8I and WLOD, would have to be rewritten. Consequently, this is seen as a long-range rather than as a short-range solution.

3.3.2 Display Terminals (R. Haskin)

ACID has survived unmodified all quarter. A notable event has been that since the machine has been moved to its new location in room 31J, no "garbaged" pictures have occurred.

Work is in progress on the replacement of the current joystick and switch arrangement with a more durable one, along with the addition of several "built-in" functions which will be invoked via the pushbuttons on the joystick box. These functions are

<u>Button(s)</u>	<u>Function</u>
1	Hit, same as pressing any button in the current configuration.
2 (3)	Y (X) Constrained hit--caused a joystick hit using the current value of the constrained coordinate (as last set by a 'set constraint' hit) in place of the appropriate cursor coordinate. The other coordinate remains unaltered.
4	REORG--causes the program's "current location" X & Y to be set to the joystick cursor location. Same as current "REORG" function in GNDRW, which it replaces.
5 (6)	Y (X) Continue-Mode Hit--same as regular joystick hit (button 1), but uses the Y (X) value of the <u>previous</u> joystick hit in place of the cursor value. This function is useful for drawing figures containing right angles.
7 (8)	Y (X) Set constraint value. This hit sets the value to be used in future Y or X constrained joystick hits (buttons 2 & 3). The value used is the appropriate coordinate of the cursor location. The value remains in effect until reset.
10	PANIC--Forces a "panic area" joystick hit, having the same effect as a normal joystick hit in the panic area.
11	RETURN--Forces a "return area" joystick hit, having the same effect as a normal joystick hit in the return area.

3.3.3 Information Retrieval (R. Haskin)

The current version of IR has been in production use with GRASS all quarter. Modifications to GODISK (device driver subroutine) and to the block sorting routine, which will be necessary for use with the new disk, have been coded and are awaiting the arrival of the disk for testing.

3.3.4 Monitors (R. Haskin, J. Nickolls)

GRASP (360 Remote Monitor)

Production use has been reliable all quarter.

2701 Data Link

The interface between the PDP-8 and the 2701 PDA has been reliable all quarter.

GLASP (PDP-8 Local Monitor)

Several small changes have been made to GLASP this quarter. Most of these involve the implementation of the new joystick pushbutton functions (see 3.3.2). PICSEG, the joystick hit filtering routine, has been completely rewritten to allow constrained joystick hits and "return" and "panic" area hits. Also, several previously unused fields in the PCB have been appropriated for such uses as containing the X and Y constraint values, passing return codes from program segments and others.

Program Segments

GNDRW: An "edit mnemonic" function has been added to allow easy modification of instance definitions. This function, invoked from the keyboard with an E<mnemonic name> command, decomposes the mnemonic storage block into its constituent line, text, and terminal blocks; and replaces the current picture with these. The definition can now be modified with the standard drawing functions, and the parameters and terminal type names can be changed. The mnemonic is saved again in IR with the usual "C&" and "S<mnemonic name>" commands.

GLASP Status

The system is now in a state of flux as input from users is being taken. Requests for more convenience of operation, mostly in minor, detail areas, are being implemented steadily. With the addition

of a reliable disk, the system will reach a level which would enable it to be used in a true production environment.

Applications Support (J. Stynes)

More models have been developed this quarter to test the effectiveness and applicability of the Simulation and Modeling System. A highway system has been designed and sent to the remote system for analysis. A turborator system is also being developed and will shortly be analyzed. During the design of these models, possible changes were suggested that would facilitate the user's task of model description. Some suggestions which are currently being implemented are the ability to display parameters, the ability to assign equations directly from the drawing mode, and the use of horizontal and vertical constraints in the line drawing mode.

Next quarter, in order to develop the system's capability for handling a broad class of problems, a number of users from different department will be employing the system.

3.3.5 Remote Data Structure Utilities (J. Nickolls)

COMMUNE

As part of the applications support development, COMMUNE has been extensively expanded and improved this quarter. See DCS Report No. 466 for an introduction.

The data structure handling capability has been expanded from just line and text blocks to any block of a picture or the whole picture. A large number of pictures may be handled at one time. The current routines have been updated to implement the changes, and new routines have been written to fetch and save pictures and blocks from the Remote Filing System, XFILE (see DCS File No. 867).

Each picture being processed is assigned a number from 1 to 100 by the FORTRAN user. Each picture has a Picture Control Block (PCB) which maintains information about the location, length, and type

of each data structure block making up the picture. Since COMMUNE is re-entrant, the user's internal work area contains a pointer to the linked chain of PCB's. This work area has been shortened considerably, since each PCB now contains a work area for its picture.

All existing programs using COMMUNE will run with the new version without modification. The new routines and calling sequences follows.

The routines MASK, REPLY, MESSAGE, and ERASE are unchanged.

LINE, LINED

LINE and LINED have two new optional parameters IPIC and IBLK. The other parameters have not changed from their description in DCS Report No. 466. The FORTRAN calling sequence is

```
CALL { LINE } * (X, Y, INTEN[, IPIC[, IBLK]][, &STMT]**)
```

IPIC is the user assigned picture number from 1 to 100. Its default is 1. IBLK is the data structure block number from 1 to 10 into which the line entry is to be placed. This currently must be 1 and is defaulted to 1. IPIC must be present in order to have IBLK present. &STMT is optional independently of all other parameters.

The data structure blocks represented by IBLK are below.

IBLK:	1	Line Block
	2	Text Block
	3	Mnemonic Instance Block
	4	Spare
	5	Terminal and Connection Block
	6	Declaration Block
	7	Equation and Parameter Block
	8	Spare
	9	Spare
	10	Spare

* Braces indicate that a choice must be made of one of the enclosed possibilities.

** Brackets indicate that parameters enclosed are optional and may be omitted.

TEXT

TEXT has had the same modifications as LINE and LINED.

```
CALL TEXT(X,Y,ITEXT,LEN[,IPIC[,IBLK]][,&STMT])
```

IPIC is defaulted to 1, IBLK to 2. IBLK currently may be 2, 6, 7, 8, 9, or 10. Block 6 is the Declaration Block containing entries for Definitions, Globals, Locals, and Special Ops, respectively. Block 7 is the Equations and Parameters Block with entries for equations and parameters in that order.

SEND has the new optional parameter IPIC, and IBLK may now assume values from 1 to 10 to send any single block of a picture. A value of -1 sends the entire picture. The FORTRAN calling sequence is

```
CALL SEND(IBLK,ICODE[,IPIC])
```

SENDP is a new routine which sends a picture to a PDP-8 terminal. Its effect is the same as a SEND with IBLK = -1. The header block is sent followed by all initialized data structure blocks.

SAVE

SAVE is a new routine which saves a block of a picture in the Remote Filing System, XFILE. The FORTRAN calling sequence is

```
CALL SAVE(IBLK,NAME,IKEEP[,IPIC])
```

where

IBLK--denotes the block to be saved (1-10)
-1 means save the entire picture.

NAME--an INTEGER*4 vector array with 8 elements. Contains a text string representing the file name of the picture in XFILE. The syntax for the file name is the same as for LSD--see DCS Report No. 466. If the user name and library name are omitted, the logon name of the terminal user and the library name "PICLIB" are used.

Examples: a. PICTUR/WHALEY.PICLIB
b. PICTUR (Logon user name used)

IKEEP--1 = Keep the block in the data structure.
0 = Delete the block after saving it.

IPIC--Optional picture number of picture to be saved.
Defaults to 1 when not specified.

File overlay is automatic.

SAVED

SAVEP is a new routine which saves an entire picture in XFILE. The FORTRAN calling sequence is similar to that of SAVE:

```
CALL SAVEP(IPIC,NAME,IKEEP)
```

FETCH

FETCH is a new routine to fetch a block of a picture from XFILE.

```
CALL FETCH(IBLK,NAME[,IPIC][,&STMT])
```

IBLK, NAME, and IPIC behave as in SAVE. &STMT is an optional parameter which indicates the statement number to which control is to be passed (via a RETURN 1 statement) if the block or picture is not found.

FETCHP

FETCHP is similar to FETCH, except that the entire named picture is fetched.

```
CALL FETCHP(IPIC,NAME[,&STMT])
```

CLEAN, CLEANP

CLEAN and CLEANP are new routines which facilitate keeping COMMUNE's dynamic storage requirements at a minimum. Whenever a FORTRAN program is done with a picture or block of a picture, it should call CLEANP or CLEAN to delete the data structures from memory. This is not necessary before a STOP statement, as the storage is then automatically freed.

Debugging of these routines will continue next quarter, with more facilities planned such as routines to make changes in existing data structures like translation, rotation, scaling, insertion, and deletion. Some of these would be akin to the PDP-8 terminal drawing functions.

3.4 Computer Maintenance and Construction

3.4.1 Graphics-8 Hardware (C. E. Carter)

DATA DISK

As was mentioned in last quarter's report, our old Data Disk was returned to California for updateing and overhauling. Since its return, many hours have been spent in keeping it alive long enough to get work done. Sixteen new amplifier cards have been constructed for use in the attempt to maintain 48 operating tracks on the disk.

NEW DISK

The purchase order was let to System Industries of California for the new disk and controller. Delivery date was to be December 22, 1971. A specification problem has necessitated a delay in shipment. We are hoping to get it next quarter.

In the meantime rack space is being prepared for the new disk installation and preliminary drawings for its connection have been made.

COMPUTEK

It has become clear that the remoting of one of our Computek terminals is desirable if for no other purpose than for demonstrations in larger areas such as conference rooms. In this light work has started and parts have been ordered to drive a Computek in room 115 DCL or room 234 DCL.

Likewise, more use and the possibility of more trouble has caused us to look at our card production again. One new set of cards is being planned for in the coming year.

A drawing and wiring list is complete for an expanded pushbutton box for the Computek. Parts are on order and completion and check out will be next quarter.

PDP-7/630

A transmitter and receiver tester have been built and checked out for use in diagnosing card faults. Its primary function is limited to transmitter and receiver cards wherein a full character buffering is facilitated.

3.4.2 Equipment Maintenance Log Summary (H. Lopeman, R. Miller)

PDP-8

1. Acc bit 3 driver burned out. (Replaced transistor)
2. I.F. bit 1 switch bad. (Replaced)
3. IPL switch sticky. (Replaced)

338

1. Ext stop flag errors. Could not duplicate failures with any existing software.
2. Bit 0 on pushbutton box not working. (Replaced broken wire in cable)
3. Light pen insensitive. (Adjusted sensitivity control on light pen)
4. Blower burned out in 338 cabinet. (Replaced blower)

Disk

1. Many problems with disk tracks. Since the data disk was returned the last time, it is extremely sensitive to temperature and voltage variations. The problem is in the disk cabinet proper.

Nothing has been found in the amplifiers to indicate problems there. Data Disk has admitted to a new style platter and surface coating.

Teletype

1. Problem with PDP-8 TTY clock causing errors. (Adjusted clocks for temperature compensation. This cures the problem within about a 5° temperature range.

PDP-7

1. Repaired punch, #2 hole intermittent, needed adjustments on magnet spring tension.
2. Data request would come on whenever tests were read in, found R121 board bad in Loc. 3J29.
3. Same problem as above, scope checked and found R-202 bad in Loc. 3E9.
4. Pie would not turn on; problem found in cable cards; wiggled cable cards and pie would turn on; checked for bad solder joints--all were all right; cleaned contacts on cable cards and appeared all right then.
5. Problems in multiplexor; data was all garbage; shorted transistor in board R141-3028.

3.4.3 Stereomatrix Interface (I. Cunningham)

Due to delays in the completion of the stereomatrix display, no attempt has been made to complete the final connections of the interface.

Work is continuing with the development of the software but progress has been hampered by disk difficulties on the PDP-8.

It is hoped that sufficient subsections of the display will be operable this quarter in order that the interface and some software may be checked.

3.5 Mini-Machine Modeling System (W. Tam)

3.5.1 General Description

This is the outline of a software package currently being implemented on the PDP-11. The package will provide the interface between the user and the numerical package of the ILLISM-E interactive modeling system. The input format is simple enough that a graphics package can be easily inserted in front to incorporate graphic input. The basic structure is a block and equation oriented simulation system. Each element is defined in terms of terminals and equations; networks are made up of connections of terminals. Some features of the system include:

1. The system is designed to solve E-type equations only. I-type algebraic identities of the form $I_1 + I_2 + I_3 + I_4 = 0$ obtained by connecting more than one I-type variable are not allowed. Under certain circumstances, a slightly modified version of the problem can be solved by using E-type equations, but in general the system uses exclusively E-type connections.

2. The equations for each element are converted directly into relocatable machine code during initial definition, and are stored within the data structure defining the element. The actual addresses are filled in at run time. This cuts down considerably the processing time required to connect individual elements to form a network. The process is reduced to that of slapping together modules of relocatable code from the various elements and readjusting the relative addresses. Some degree of optimization is built in by deleting extra equations and gives fairly efficient code.

3. Since the networks can be formed by simply combining relocatable codes, networks can be stored in almost the same form as elements. Each network is compiled only once into another element. The storage requirement is hence almost minimum (only a symbol table plus the relocatable code need to be stored; the connection table is discarded after compilation).

4. Further optimization is performed during the generation of network codes. Equations that need to be evaluated once only are separated, so that during execution, these codes are executed only once, saving a considerable amount of time.

3.5.2 The Data Structure

The data structure for storing elements and networks consists of a directory and the element/network definition tables. This is usually stored on disk or loaded from tape to disk when the user comes on to the system. When the user requests a certain element, ILLISM-E will search for the name in the directory, and copies the element/network definition table into core (Figure 4).

Figure 4

Each element/network definition table contains two parts: the symbol table and the relocatable code. The symbol table contains entries for terminals, parameters, and variables, with links to all the operand fields in the relocatable codes where the variable is referenced. The actual storage of these variables (operands) are obtained when storage allocation takes place just before execution. A linking routine will then follow these links and fill in the actual addresses, hence making the codes readily executable. Also, these addresses are entered into the symbol table, so that they can be referenced by the output/graphics package.

DATA STRUCTURE FOR ELEMENT/NETWORK DEFINITION TABLE

3.5.3 Interface with the Numerical and Output Packages

The output of this package is two column vectors $\underline{x}'$, $\underline{x}$, and the parameters N, T, H, where

N = dimension of the vectors

T = time

H = interval

plus the code that computes $\underline{x}'$ as a function of $\underline{x}$ and other parameters (see Figure 5).

After the user has input the parameters and the initial conditions, ILLISM-E will be swapped out to disk leaving in core the items shown in Figure 5. At the end of the computation of $\underline{x}'$, the control section will put on top of the stack the address of $\underline{x}$, N, T, and H. Then the integration package will be called in from disk to perform the integration, giving the next value of $\underline{x}$.

Again, the code will be swapped to compute the next value of $\underline{x}'$ and so on. At intervals or certain conditions specified by the user, the control section will call in the output/graphics package, and using the symbol table, output the values of specified variables in either numerical or graphics form.

Figure 5

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

C00-1469-0199

2. TITLE

4th Quarterly Progress Report
(October, November, December 1971)

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:

Title of conference _____

Date of conference _____

Exact location of conference _____

Sponsoring organization _____

- ☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)-

C. W. Gear, Professor
and Principal Investigator

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois 61801

Signature

Charles W. Gear

Date

January 31, 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

4. IMAGE PROCESSING AND PATTERN RECOGNITION RESEARCH: ILLIAC III
(Supported in part by Contract AT(11-1)-2118 with the U.S. Atomic
Energy Commission.)

A program of experiments with an image processor is being carried out. Our goal is to design a sight sensory system predicated upon parallel strategies for image processing. This development must of course evolve from our experience with the Illiac III system. The key property we seek is the ability of the system to learn rapidly from instructional interaction with professional personnel (not necessarily trained in computer technology). The validity of this orientation has been, we believe, vindicated by the rapid growth in significant image processing applications we can now attack.

Highlights of the program this quarter include:

1. Show-and-Tell has bedded down to being a workable, documented package for interactive picture processing.
2. The Covering Theory concept - extending to pattern recognition methods originating in switching theory, continues to be a very fertile ground for investigation. Interval Coverings have previously been described here. Now Cartesian Covers, the latest extension of the covering theory methodology, are introduced here - accompanied by an operational program for their construction.
3. Vari-valued Logic, an outgrowth of the covering theory work which extrapolates from two-valued variables of switching theory to finitely quantized variables, has been introduced to provide a decision calculus for dealing with the pattern recognition problems.

4. The ATLAS software package, an eidetic-based system for the processing of visual data, is emerging from the cocoon of its initial conception. Attempted applications to brain mapping and to aerial photointerpretation have stimulated much examination and extension of these concepts. (A summary of these concepts will be issued next quarter.)
5. Shape Recognition, a necessary ingredient of the ATLAS system, has been successfully attacked in the work of Maruyama for simple two-dimensional shapes called "angularly simple". Extension to more involved forms is proceedings apace.
6. Considerable advances in completing the Illiac III: the Pattern Articulation Unit (PAU), Arithmetic Unit (AU) and Taxicrinic Processor (TP) are reported.

4.1 INTERACTIVE PICTURE PROCESSING

4.1.1 Show-and-Tell

Work was begun this quarter in three areas: (1) Documentation of the existing Show-and-Tell package, in which image processing is accomplished primarily on the IBM 360/75; (2) Conversion of Show-and-Tell from Disk Monitor System [DMS] to the new PS-8 Operating System; and (3) Addition of a PAU Simulator to Show-and-Tell so that some image-processing can be done without requiring the 360.

We have been testing the PS-8 Operating System with a DEC-tape system device by using the Graphics Group's PDP-8. A program was written to convert DMS-format ASCII files to PS-8 format, and the Show-and-Tell symbolic files were processed. When the Maintenance Processor Interface is complete, we will begin using PS-8 on our machine, treating the Illiac III core as a super-high-speed disk. Programs and data files will normally reside on LINC-tape and be swapped into core at the start of a session, and dumped back into LINC-tape at the end.

Routines to simulate GATEIA-level PAU operations have been written by the hardware staff and are operational. Work has been started on writing a driver to do some of the higher level PAU instructions under Show-and-Tell. When the hardware PAU is complete, the low-level simulator modules will be replaced by calls to the PAU, with no changes required in the driver or the Show-and-Tell translator.

4.1.2 Atlas Extension

This past quarter was spent on the definition of a new procedure-oriented language, GODOL, which will be interpretively implemented on our PDP8e. GODOL was designed to supersede Show-and-Tell, which was found

inadequate for Atlas use; also, it will considerably improve our local image and graphical processing capabilities, though with some sacrifice of turnaround time.

4.1.3 Terminal Communication

The specifications for the LINC tape controller have been modified to include hardware checksum generation and checking and automatic block searching. The design effort is continuing.

4.1.4 Scan/Display Devices

This quarter was spent in eliminating discrepancies between existing scanner wiring and wire lists. This involved: (1) Addition of eight new wire lists; (2) Correction of two wire lists; and (3) Location and clarification of numerous wire lists. The total number of wires affected were between 900 and 1000.

No new optical or mechanical work was done on the current scanners due to other commitments. However, the requirements for converting a 46 mm. transport into a 35 mm. full frame transport were set out rather fully. Such a transport can be constructed with little investment in new parts.

4.2 PARALLEL PROCESSING

4.2.1 Texture Analysis

Using signal detection theory, a technique has been successfully demonstrated here which detects and isolates different textural regions in scenes of practical importance. To this purpose, a template is selected which defines a universe of patterns whose occurrences in the scene of analysis are to be studied for the purposes of discrimination between various textures. To quantify a judicious choice of size and shape of the template given a textural scene, we have resorted to the Time Series Analysis. Considering the textural scene as a two-way time series, traditional analysis can be carried out to determine the order of statistical dependence among neighboring picture cells. It is anticipated that a reasonable interpretation, using the Signal Detection Theory techniques employed earlier, can be obtained in terms of the Time Series Analysis.

(For further information see References [1], [2], and [3].)

4.2.2 Interval Covers

An application of interval covering theory to the synthesis of optimal filters for local feature extraction creates a need for an algorithm which can synthesize an interval cover of a set against its complement, when the complement is a very large set (e.g. has approximately 10^{12} elements) and can not be handled directly. Such an algorithm has been developed (Michalski) and work is being done on its computer implementation (Pitt).

4.2.3 Cartesian Covers

The concept of a 'cartesian cover' is an extension (developed by McCormick and Michalski [4]) of an interval cover, and is very useful for

storing and handling n-ary relations. An algorithm synthesizing a cartesian cover of a set F^1 against a set F^0 , where $F^1 \subseteq E$, $F^0 \subseteq E$, and $F^1 \cap F^0 = \emptyset$ (E is a universe of elements), has been developed and a PL/1 program implementing it is already in operation.

4.2.4 Varivalued Logic

A new logistic system called 'varivalued logic' (or 'variable-valued logic') has been proposed as a response to a need for a more adequate system for solving problems in pattern recognition and artificial intelligence. A program for synthesis of quasi-minimal varivalued logic expressions (VL-expressions) is already in operation. This new concept was presented at a seminar at the Coordinated Science Laboratory at the University of Illinois, Urbana, Illinois, on December 8 and at Purdue University on December 14.

4.2.5 PAX Language Support

A new number of the PAX User's Newsletter (PLANE TALK) was published and distributed to over 35 individuals and installations. One new installation received a copy of the PAX system tape this quarter.

4.3 Structural Inference

4.3.1 Scene Segmentation

An approach is being developed (1) to partition scenes into closed sets called regions and (2) to form composites of regions. Partitioning scenes into regions is performed in two steps:

(1) We assume that pictures are given as a set of picture points in the plane such that each point is associated with values of local attributes, e.g. its coordinates or a gray value. For N local attributes, the picture can be represented in a N -dimensional property space. In the property space, each picture point is represented by a N -dimensional vector whose n -th component carries the value of the n -th local attribute at this point.

(2) The property space is tessellated into N -dimensional unit cells. A region is supposed to be a set of neighboring picture points so that the density distribution of points in those unit cells containing the property space representation of the picture points is as uniform as possible. The uniformity of a properly normalized density distribution of points can be conveniently measured in terms of the entropy considered in information theory. We have stated algorithms that:

- (a) infer a metric in the property space for a given training set of pictures already partitioned into regions.
- (b) partition a picture into regions, using a previously established metric in the property space. The essential tool of this algorithm is a clustering technique using mode-searching and clustering in a minimal spanning tree.

The regions forming a partition of a scene can be represented as

nodes of a graph whose edges are labelled by attributes or relations and weighted by their values. A very efficient representation of such a graph is a vari-valued logic (VL) expression (see Section 4.2.4). Composites of regions can be formed by determining cartesian complexes synthesizing the vari-valued logic expression representing that graph. The algorithm A-8 that synthesizes VL-expressions allows control of the formation of cartesian complexes by imposing heuristic rules, permitting extensive experimentation on inferring composite objects. Furthermore, the representation of labelled, weighted graphs as VL-expressions is very efficient for applying graph transformation rules that have been developed earlier (see QPR April-June 1971, Section 4.3.1).

A description of these developments will be given in a thesis* to be issued in January 1972.

4.3.2 Inference of 3-Dimensional Model Orientation

The general goal is to identify a two-dimensional projection of a three-dimensional object as representing a specific model. As a first step toward this end, work has begun on a method of determining whether or not a given shadow or silhouette can occur as a result of any type of rotation or translation of the model. Certain partial inferences of internal lines can be made directly from the shadow. Using these inferences and information from the shadow's outline, a graph representation can be formed and compared with a similar representation of the model itself. By rotation, weighting, and adjustment of the partial inferences, it is possible to fit

* Raulefs, P., "Methodological Aspects of Scene Segmentation", Department of Computer Science, University of Illinois, Urbana, Illinois, M.S. thesis, (in preparation).

partitions of the model to the shadow. When one or more partitions are fitted, the model can be rotated to check for consistency with the entire figure.

4.3.3 SOL Language Support

Implementation of the SOL Language has begun. The first version of the language will be embedded in PL/1. We are considering a few areas of application for SOL, and the first version will be used as a tool for experimenting with the language. This hopefully will provide us with sufficient information about the vital subset of PL/1 and other necessary operations for the language. Later we will define a complete self-contained language with proper implementation.

To resolve an ambiguity in declaration of subgraphs, we have had to add level specifications to our syntax, which gives us sufficient information about the scope of a subgraph. Another main feature of the language is that it allows intersecting subgraphs. This feature enables us to use repeated names. To accomplish this we had to introduce the concept of disjoint subgraphs (i.e. graph is a disjoint subgraph at level 1).

Run time storage allocation for elements of the language use PL/1 list processing facilities to simulate these structures. These are, namely, based variables which are completely under the user's control. In this first version, we do not have a block structure of our own, because the based variables are used for definition of our elements. By defining Free Operation, however, we can effectively have a block structure which gives the user the ability of using repeated names for main structures (pointer, set, graph) by first freeing the former ones.

As our language deals with dynamic structures which are modified frequently at execution time, the best time to discover errors will be at execution time.

4.4 Applications

4.4.1 Cervical Smears

The initial phases of this project were brought to a close this quarter. Some experimental results which used the interval covering technique (Section 4.2.2 this report) were presented in a talk given at the IEEE-SPIE-Pattern Recognition Society Conference on Two-Dimensional Signal Processing in Columbia, Missouri. A paper on the same subject was written and submitted to the IEEE Transactions on Computers. In addition, a thesis titled "Parallel Image-Processing for Automated Cytology" (by John Read, to be issued as a Department of Computer Science Report) was brought to a state of near completion.

4.4.2 Brain Mapping

Little was completed concerning brain mapping this quarter due to commitments to projects of a higher priority.

4.4.3 Cytospectrometer

This quarter saw the complete re-design and construction of a new main frame for cytospectrometer experiments. The unit now features modular vertical cylinders with quarter inch thick sandwiched carrier plates which may be extensively machined without disturbing the main frame. All segments and carrier plates are readily demountable for changes.

A novel method of pumping liquid samples at the extremely low flow rates needed (ca. $5 \times 10^5 \text{ } \mu\text{m}^3/\text{sec}$ or slower) has been devised. It operates by controlled thermal expansion of the liquid volume and is inherently friction and vibration free. Tests have shown the method to be simple and very controllable.

The long working distance microscope optics and camera were mounted and adjusted. Sample pictures were made during the thermal pump test.

Since the required pump rate is now available, the high voltage electrostatic tests for Taylor cone formation may now be made. Assembly proceeds apace.

4.4.4 Remote Manipulation

Report UIUCDCS-R-71-490 titled "A Problem in Form Perception: Odd Shape Detection", has been written, and the following is the abstract.

A procedure to determine an odd shape or form in a given set of shapes, each set consisting of more than two shapes, is described. The "oddness" of a shape in a set of shapes on the basis of a specified feature is defined and then the "most odd" shape on the basis of many detected features is defined. To make the system response like an average human response, modification of an assumed weight vector by means of an "attention mechanism" is considered.

Some of the results attained by the system simulation on various criteria are as follows. The gestalt measure, which is the perimeter divided by the square root of the area, and which has been considered an important invariance of the form by many psychologists, becomes less important here. Linear Separation and Euclidean Distance Separation methods are not significantly different. The attention mechanism described gives good system responses, and the shape representation, called the pattern sequence, $S_r(x_0)$, has many advantages in the analysis of geometric shapes.

4.5 COMPUTER SYSTEMS

4.5.1 IBAL Assembler

This quarter work centered around semantic implementation of the IBAL language. As reported previously, we have been planning a two-pass translator for IBAL. Contrary to our expectations, the first pass now carries the major part of the translation and takes care of all high level features in the language. The intermediate code generated by the first pass has an almost one-to-one correspondence with machine instructions, which makes the second pass very straightforward. A major contribution of the second pass, however, is a good Pointer Register (PR) Allocation Algorithm, which is one of the key factors for an efficient code.

All structural definitions and implementation strategies for the first pass are now being laid out. Some of them are the following:

- (1) Memory allocation for procedures and displays.
- (2) Procedure initialization and exit.
- (3) Calling sequence (convention).
- (4) Actual formal parameter correspondence.
- (5) Relation between the active display and data area.
- (6) Run-time storage for structures (and arrays).

4.5.2 Operating Systems

There have been no new developments in this area because of time spent in other areas having higher priority.

4.5.3 TM II

The two types of printed circuit cards necessary for communication with the IA went through layout and prototypes were etched. Checkout of the cards was deferred.

National DM8599 memory chips underwent initial testing to determine whether they could be used in place of the SN7489N previously under consideration.

4.5.4 TP, AU, and IOP

The production and checking of all (66) logical drawings for the control of Arithmetic Unit 0 (AUO) have been completed. Also all (62) printed circuit boards (PCB) derived from these drawings have been laid out, wire listed and wired. The testing, layout in the mainframe, back panel wire listing, and back panel wiring for these PCB remains to be done before operational checkout can be started for the Arithmetic Unit.

The basic machine of the Taxicrinic Processor 0 (TPO) is completed to the following extent - logical design, 100%; layout and wire listing of printed circuit boards for control, 70%; printed circuit board wiring, 55%; and printed circuit board testing, 30%. All processing hardware for TPO is completed. The final machine of the TPO (hardware needed in addition to the basic machine) is complete as follows: logical design, 20%; and layout and wire listing of PCB, 10%. The control point flow charts, associated with the PAU instruction sequence have been completed.

REFERENCES

- [1] Hannan, Multiple Time Series, John Wiley & Sons, Inc., New York, 1970.
- [2] Box and Jenkins, Time Series Analysis - Forecasting and Control, Holden-Day, Inc., New York, 1970.
- [3] Bhattacharya, P.K., "Order of Dependence in a Normally Distributed Two-Way Series", University of Arizona.
- [4] McCormick, B.H. and Michalski, R.S., "Interval Generalization of Switching Theory", Department of Computer Science, University of Illinois, Urbana, Illinois, Report 442, May 1970.

4.6 BIBLIOGRAPHY

4.6.1 External Documents Issued

Outside Lectures:

B. H. McCormick, Washington University
Seattle, "Seeing and Believing", September
10, 1971.

John Read, Two Dimensional Digital Pro-
cessing Conference, University of Missouri,
Columbia, "Problems and Programs for Computer
Processing of Microscope Images", October
6-8, 1971.

B. H. McCormick, Annual Review of Electronics
Conference, University of Illinois, Urbana,
"Digital Systems for Automated Cytology",
October 18, 1971.

B. H. McCormick, Distinguished Visitors
Program of the IEEE Computer Society,
University of Missouri, Columbia, Missouri,
"Image Processing in Automated Cytology",
November 11, 1971.

Seminars: "Pattern Recognition in Biological and Technical Systems"

C. W. Eriksen, "Selective Attention in
Visual Perception", October 12, 1971.

John Read, "Problems and Programs for Computer
Processing of Microscope Images", October
21, 1971.

Kiyoshi Maruyama, "Some Problems of Modeling
of Human Shape Perception", October 28, 1971.

Heinz Von Foerster, "Three Problems of Compu-
tation in the Central Nervous System",
November 3, 1971.

Charles Lewis, "Perception of Depth in Pictures",
November 10, 1971.

L. E. Jones, "Multiple Discriminant and
Regression Models for the Analysis and Repre-
sentation of Multidimensional Psychophysical
Relationships", November 17, 1971.

M. E. Babcock, "A-Synchronous Sampling of
Speech", December 1, 1971.

Emanuel Donchin, "Computer Aided Studies of Event-Related Brain Potentials", December 9, 1971.

Peter Shaw, "Pattern Recognition By Humans of Letters and Words", December 15, 1971.

Department of Computer Science Reports:

UIUCDCS-R-71-489	Maruyama, K., "An Approximation Method for Solving the SOFA Problem", December 1971.
UIUCDCS-R-71-490	Maruyama, K., "A Problem in Form Perception: Odd Shape Detection", December 1971.

Papers Accepted for Outside Publication:

Maruyama, K., "An Approximation Method for Solving the SOFA Problem", will appear in An International Journal of Artificial Intelligence.

Maruyama, K., "A Procedure to Determine Intersections Between Polyhedral Objects", will appear in Issue #3 of the International Journal of Computers and Information Sciences.

Maruyama, K., "On the Parallel Evaluation of Polynomials", accepted for publication in IEEE Transactions on Computers.

4.6.2 Logic Drawings Issued

The following new logic drawings have been drawn and issued during the past quarter:

AU Control Logic	46 drawings
TP Control logic	8 drawings

4.6.3 Engineering Drafting Report

At present time 90% of all AU control logic drawings are completed. 33 TP control logic drawings were updated and are ready to be issued. 20 of the previously issued Power Turn-on logic drawings have been updated and respectively crossreferenced. During the past quarter a total of 197 drawings,

including new logic drawings, drawing changes, layouts, thesis and report drawings were processed by the 2118 drafting section.

4.7 ADMINISTRATION

4.7.1 Personnel Report

Senior Staff

Professor Bruce H. McCormick - Principal Investigator

Assistant Professor R. S. Michalski

Professional Staff

Robert C. Amendola
Richard T. Borovec
John S. Read

Research Engineering Assistant

S. Paul Krabbe

Electronic Engineering Assistant

Joseph V. Wenta

Digital Computer Technician II

George T. Lewis

Drafting

Stanislavs Zundo

Research Assistants

Walter Donovan
Steve Fierce
Lakshmi Goyal
S. N. Jayaramamurthy
Ahmad E. Masumi
Kiyoshi Maruyama
Frank Murakawa
Dan Pitt
Peter Raulefs
Kuo Wen

Secretarial

Mrs. Judy Arter

4.7.2 Computer Usage Log Summaries

1. The PDP8/e was used approximately 242 hours this quarter.

2. The Scanner-Monitor-Video System was used as follows:

Production and Demonstration	72 hrs	50 mins
Preventive Maintenance	46 hrs	40 mins
Corrective Maintenance	21 hrs	05 mins

3. IBM 360/75 utilization totalled \$3,534.60.

U. S. ATOMIC ENERGY COMMISSION
UNIVERSITY-TYPE CONTRACTOR'S RECOMMENDATION FOR
DISPOSITION OF SCIENTIFIC AND TECHNICAL DOCUMENT

(See Instructions on Reverse Side)

1. AEC REPORT NO.

COO-2118-0030

2. TITLE

ILLIAC III QUARTERLY PROGRESS REPORT--OCTOBER-DECEMBER 1971

3. TYPE OF DOCUMENT (Check one):

- ☒ a. Scientific and technical report
☐ b. Conference paper not to be published in a journal:
 Title of conference _____
 Date of conference _____
 Exact location of conference _____
 Sponsoring organization _____
☐ c. Other (Specify) _____

4. RECOMMENDED ANNOUNCEMENT AND DISTRIBUTION (Check one):

- ☒ a. AEC's normal announcement and distribution procedures may be followed.
☐ b. Make available only within AEC and to AEC contractors and other U.S. Government agencies and their contractors.
☐ c. Make no announcement or distribution.

5. REASON FOR RECOMMENDED RESTRICTIONS:

6. SUBMITTED BY: NAME AND POSITION (Please print or type)

Professor Bruce H. McCormick
Principal Investigator
Illiac III Project

Organization

Department of Computer Science
University of Illinois
Urbana, Illinois

Signature

Bruce H. McCormick

Date

February 24, 1972

FOR AEC USE ONLY

7. AEC CONTRACT ADMINISTRATOR'S COMMENTS, IF ANY, ON ABOVE ANNOUNCEMENT AND DISTRIBUTION RECOMMENDATION:

8. PATENT CLEARANCE:

- ☐ a. AEC patent clearance has been granted by responsible AEC patent group.
☐ b. Report has been sent to responsible AEC patent group for clearance.
☐ c. Patent clearance not required.

5. ILLIAC IV

(This work was supported in part by the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, and in part by the Advanced Research Projects Agency as administered by the Rome Air Development Center, under Contract No. USAF 30(602)4144.)

REPORT SUMMARY

Main achievements during the quarter include:

- 1) the consolidation of Glypnir, considerable progress in the provision of separately compiled subroutines and a new faster scanner which will provide an advanced macro facility,
- 2) the building of a prototype parallel assembler which will assemble a limited subset of ASK at a rate of about a quarter of a million cards a minute,
- 3) the addition of an ILLIAC IV timing simulator to the ILLIAC IV functional simulator,
- 4) continued improvement of the reliability and availability of the B6500 at the University

Project expenditures through December 1971:

Burroughs Corporation	\$28,448,264.00
University of Illinois	8,459,400.00

SOFTWARE

5.1 ILLIAC IV Assembler (ASK)

Early in the quarter, work on the ILLIAC IV Assembler (ASK) ceased except for general Assembler maintenance and the writing of a maintenance manual. The manual is now essentially complete and several bugs which have appeared in the Assembler itself have been put to rights.

5.2 ILLIAC IV Simulator (SSK)

During the quarter, a timing simulator was built into SSK, thus allowing ASK and Glypnir programmers to get an estimate of ILLIAC IV elapsed time and other statistics at the same time their programs are being simulated, with no significant increase in simulation time. The new simulation has been used constantly during the last half of this quarter and is operating satisfactorily.

5.3 Glypnir

Most of the first month of the quarter was spent adjusting the Glypnir Compiler to Burroughs Mark II.0 Software. Apart from taking advantage of the more reliable second generation of B6500 software, the compiler was found to have increased its speed by about 10-15%. The remainder of the quarter was spent in designing and coding facilities to allow separately compiled subroutines to be bound to a main Glypnir program, in providing a new, faster scanner with advanced macro facilities, and various small improvements that will improve the execution speed and storage efficiency of Glypnir programs. Additions to the Glypnir programming manual are in hand.

5.4 Parallel Assembler (Pandora)

The Pandora project is an investigation into the feasibility of providing an assembler as similar as possible to ASK, but one which executes on ILLIAC IV at a target speed of a million cards a minute. During the quarter, a prototype assembler, accepting a subset of ASK

but coded in Glypnir, has been written. It is presently assembling code at about a quarter of a million cards a minute, and has produced either correctly executing code or the appropriate syntactic or semantic error messages for a small number of sample programs. The prototype is at present being reviewed and extended.

5.5 Documentation

During the quarter, the project documentation service printed and distributed manuals, introductory material and other ILLIAC IV documentation to users and other engineers. The service also undertook the editing and supervision of eight ILLIAC IV related documents which have been subsequently printed or are in the process of being published.

5.6 B6500 Service

Since the introduction of Burroughs level II software, the reliability and availability of the B6500 being used by the ILLIAC IV project and the Center for Advanced Computation at the University has improved. However, system reliability and software facilities do not yet allow reliable accounting procedures to be introduced with any certainty. An average of 12,000 jobs were processed each week, and the machine was unavailable for an average of about five hours a week. Although use of the machine was inhibited to some extent by disk errors toward the end of the quarter, this problem has been corrected.

ADMINISTRATION

5.7 Financial Report

Actual expenditures through September 1971:

Burroughs \$28,063,847.00

University 8,211,386.00

Expenditures and obligations for October, November,
December, 1971:

	<u>October</u>	<u>November</u>	<u>December</u> <u>Estimate</u>
Burroughs	\$222,335.00	\$162,082.00	NA
University	75,920.00	73,483.00	\$98,500.00

Estimated expenditures and obligations through December 1971:

Burroughs \$28,448,264.00

University 8,459,400.00

DOCUMENTS

- Bernhard, Winfried. "ILLIAC IV Codes for Jacobi and Jacobi-like Algorithms." Center for Advanced Computation Document No. 19, Center for Advanced Computation, University of Illinois at Urbana-Champaign (November 5, 1971).
- Lermit, J. and Randal, J.M. "Augmented Significance Routines for ILLIAC IV." ILLIAC IV Document No. 255, ILLIAC IV Project, University of Illinois at Urbana-Champaign (October 8, 1971).
- McDaniel, Lawrence. "A Jacobi Algorithm for ILLIAC IV." Center for Advanced Computation Document No. 21, Center for Advanced Computation, University of Illinois at Urbana-Champaign (November 8, 1971).
- Randal, J. M. "A New ILLIAC IV Timing Simulator." ILLIAC IV Document No. 256, ILLIAC IV Project, University of Illinois at Urbana-Champaign (December 1971).
- Stevens, James. "A Fast Fourier Transform Subroutine for ILLIAC IV." Center for Advanced Computation Document No. 17, Center for Advanced Computation, University of Illinois at Urbana-Champaign (October 13, 1971).
- Systems Control, Inc. "Study of the Radar Signal Data Processing Tasks on ILLIAC IV." ILLIAC IV Document No. 254, ILLIAC IV Project, University of Illinois (November, 19, 1971).

6. CENTER FOR ADVANCED COMPUTATION

6.1 Research Program and Plan

The principal objective of this program is the development and testing of numerical techniques and software systems for use of ILLIAC IV over the ARPA network. This is being accomplished through activities in the following areas:

1. Development of numerical techniques suitable for parallel processing in the following areas:
 - a. Numerical solution of linear systems of equations
 - b. Numerical solution of algebraic eigenvalue problems
 - c. Ordinary and partial differential equations
 - d. Time series analysis
 - e. Linear programming
 - f. Non-linear equations
2. Investigation of the capabilities of ILLIAC IV to perform data management and statistical analysis functions
3. Development of ILLIAC IV/B6500 graphics programs
4. Development of an ARPA network terminal system and the B6500 network control program
5. Development of programs in large scale calculations dealing with input-output economic modeling, molecular structure calculations, quadratic assignment algorithms for various classes of spatial allocation problems, and atmospheric dynamics.

In addition, education of segments of the ILLIAC IV user community is being accomplished through seminars, classes, and the development and dissemination of tutorial materials.

6.2 Major Accomplishments

During this quarter the first level user language for the ARPANET Terminal System (ANTS) on the DEC PDP-11 was completed and debugged. Remote terminal access was completed. The remaining PDP-11 peripherals were delivered and installed.

Two ILLIAC IV graphics display algorithms were completed: a surface presentation of $f(x,y)$ for an $m \times n$ core contained array and a 64-bit contour display algorithm which was checked and modified to conform to ILLIAC IV subroutine systems. Further development in this area has been suspended until the hardware configuration of ILLIAC IV is stabilized.

A B6500 system graphics package was completed this quarter. This integrated system allows users interactive and programmatic access to Computec displays, cal-comp plotter and raster scan graphics on the Gould printer. A complete simulated usage of ILLIAC IV graphics was performed with graphics results from ILLIAC IV SSK graphics simulations displayed on a Computec and recorded on the Gould printer. Work next quarter will include documentation of these graphic systems routines.

The hardware interface between the B6500 and the IMP was designed and two units constructed at Paoli. One unit was debugged and the second unit partially debugged. Software design of the B6500 NCP was completed and the programs were 90% decoded. Efforts next quarter will involve completion of program debugging of the NCP. Discussions were initiated with the University of California at Santa Barbara regarding future use of their B6700 over the ARPANET. The Center is providing advice to UCSB regarding the creation of a B6700 NCP.

A parallel algorithm for finding the real roots of the polynomials with real coefficients using Sturm sequences and an alternative parallel Jacobi code (for finding the eigenvalues and eigenvectors of a symmetric matrix) that is more efficient on the ILLIAC IV were completed. A parallel algorithm for the eigenvector solution of large size matrix Riccati equations that arise in many engineering applications (filtering, linear regulator with quadratic performance index, etc.) was completed. Documents for these three algorithms will be forthcoming.

We are examining the local intermediate scale computer requirements for this ARPA contract as well as other Center contracts. We are considering retaining the Burroughs B6500 or obtaining a DEC KI-10.

The prototype for a large scale integrated econometric and inter-industry model was completed this quarter and is on-line on the B6500. This system can be used for forecasting future national manpower

requirements based upon changing national economic priorities. CAC documents were published this quarter describing the system and associated research uses.

A grant to the Center was received from the National Science Foundation to support a study of the efficiency of ILLIAC IV in atmospheric dynamics calculations. This one year grant will share personnel costs with this ARPA contract while funds under this ARPA contract will provide the computer time on ILLIAC IV and the computer time for ILLIAC IV simulations on the B6500.

6.3 Problems Encountered

B6500 performance during this quarter has improved. However, the level of performance is still not that which is desired. Hardware and software problems are slowly being resolved.

6.4 Fiscal Status

Actual expenditures through 31 September 1971: \$199,429.00.

Estimated expenditures and obligations through 31 December 1971:

<u>October</u>	<u>November</u>	<u>December Estimate</u>
\$82,981.00	\$85,170.00	\$57,000.00

Total estimated through 31 December 1971: \$424,580.00

6.5 Future Plans

Work will continue in the research program described in the first section of this report. We anticipate completion of the second phase of the ANTS system during next quarter to include completion of remote job entry to the UCLA 360/91 and assimilation of recently delivered DEC peripheral hardware into the ANTS system. We expect to be able to submit an initial report next quarter on the capabilities of ILLIAC IV to perform data management functions. In addition, we will begin a survey of the user community requiring an ILLIAC IV statistical package.

7. SWITCHING THEORY AND LOGICAL DESIGN

(Supported in part by the National Science Foundation under Grant Number U.S. NSF-GJ-503.)

Several transformations for the simplification of any given network with NOR gates were developed. Also their effectiveness was tested on examples by IBM 360/75. Examining the computer outputs, these transformations were further improved.

Logical design of optimal networks with MOS was continued. T. Shinozaki implemented a computer program for this design problem based on Liu's work. (This is his master degree thesis.) Since the algorithm is not iterative, the program yields results very quickly. Since the algorithm may make some MOS cells complicated, Liu developed an algorithm to make the complexity of MOS cells as uniform as possible.

S. MUROGA

Transformation procedures for the simplification of NOR networks were modified and improved. We also added several new procedures. These transformations use the concept of "permissible functions". We have the following procedures to calculate permissible functions.

- (1) A procedure to obtain maximum sets of permissible functions.
- (2) Procedures to obtain compatible sets of permissible functions.

The major transformation procedures we have are as follows.

- I. Pruning procedures using the concept of permissible functions.
- II. A network transformation procedure using pruning procedures.
- III. A procedure for replacing a gate.
A procedure for merging two gates.
- IV. An extension of the procedure in II.

V. An extension of the procedures in III.

VI. Network transformation procedures using the concepts of permissible and error functions.

We applied these procedures to intermediate solutions generated by the branch-and-bound NOR network synthesis programs in order to test their effectiveness. For a network consisting of 12 gates and 34 connections (the first solution by the branch-and-bound program for a certain four variable function) we applied the above procedures and we were able to obtain the minimum network for this function (7 gates and 19 connections).

We are trying further improvements to these procedures.

(Y. Kambayashi)

We continued the study of the network transformations, mentioned in the previous Quarterly Report, whereby nonoptimal networks of gates are transformed to more nearly optimal networks while the original output functions are preserved.

The emphasis was on the refinement of the original procedures, Procedure II and Procedure IV, and the development of several smaller, faster procedures for more specific network transformation applications and for use in programs where time is an essential consideration.

(J. N. Culliney)

The main effort was to improve the redundancy check procedures for NOR networks, discussed in the previous Quarterly Report (July-August-September, 1971). A new version of the redundancy check program, aimed at

obtaining optimal networks from a "universal network", was prepared and tested. Out of 77 non-trivial 3-variable P-equivalence functions, this new version obtained the 3-level minimum networks for 74 functions. This result shows this new version is quite powerful, although further improvements are desired.

(H. C. Lai)

The manuscript of my Ph.D. thesis was being prepared during this quarter. In the mean time, an algorithm was developed for the synthesis of optimal MOS networks in which the complexities of all cells were made as uniform as possible. An algorithm to synthesize optimal feed-next MOS network was also developed.

(T. K. Liu)

A users manual for an integer linear program which was developed by our research group for solving network synthesis problems using the all-inter-connection formulation of a feed-forward network is now being prepared. William H. Kautz wrote a paper in the February, 1970 issue of IEEE Transactions on Computers which gave an example of a six gate combinational circuit with a loop. The program mentioned above is used to derive a loop-free network which is equivalent to Kautz's network and an optimal feed-forward network with seven gates and thirteen interconnections was obtained.

(K. Hohulin)

Publications

Muroga, S. Threshold Logic and Its Applications. John Wiley, November 1971, 478 pages.

Report No. UIUCDCS-R-71-488

Nakagawa, Tomayasu and Hung-Chi Lai, "Reference Manual of FORTRAN Program ILLOD-(NOR-B) for Optimal NOR Networks", Department of Computer Science, University of Illinois, December 1971, 104 pages. (This describes how to use the FORTRAN program ILLOD-(NOR-B) for the logical design of optimal networks with NOR gates. The report includes the program listing.)

8. OFFICE OF THE SOUPAC CONSULTANTS

The fifth edition of the Soupac Manual was prepared by December 1971 and arrangements are being made to have this edition available in the bookstore. Some refinements and additions have been made. Greater usability and availability of the manual are needed to accompany wider distribution of the Soupac system of programs). A primary goal is to make the manual understandable with no need of interpretation by the Soupac Consultants.

Additions to the manual include a rewritten User's Guide, a Glossary of Terms on Data Representation, Densities of Some Probability Distributions (to use in recoding data and as definitions for the Random Number Generator), and a brief description of some stand-alone programs maintained by the Soupac Group. Two introductions to manual sections appeared in this edition. There will be one for each section eventually.

Several more Soupac programs were allocated (meaning programmed to run in varying region sizes depending on the size of input and required work areas). Optimizing a program by allocating and using level H Fortran is not usually trivial. For example some parameters or data have to be checked early (usually in a special subroutine) in order to determine the size of the problem or H will require rewriting of G compiled code. This opportunity might also be taken to "clean up" code.

The "PARM=DYNAM" feature (whereby more than one version in different sizes of a program exists) has been dropped. Non-allocated programs now need a fixed region, while allocated ones need a region determined by the problem. A handout on appropriate region estimates is available from the Soupac consultants. Also a "PARM=NOGEN" Option is now available to suppress printing of code generated by the #REPEAT Option.

A new version of the Linear Programming routine is nearing completion and a Soupac/Calcomp Plotting routine has been started. Also some new non-parametric procedures may be available in the near future.

(K. W. Dickman)

9. MACHINE AND SOFTWARE ORGANIZATION STUDIES

The following is a collection of related work aimed at improved designs for computer and software systems. We are interested in parallel and pipeline processors, small primary memories, effective use of rotating memories and some questions concerning user languages for problems including typical FORTRAN type calculations, simulation languages and a variety of file processing problems.

(D. Kuck)

9.1 FORTRAN Parallelism Detection - (S. C. Chen)

During this quarter, the new version of DO-loop subprogram DOHANDLE has been incorporated by the assignment subprogram ASSIGN with its new backsubstitution and recurrence features. The whole analyzer program is now being tested with real FORTRAN programs, and some preliminary results have been obtained successfully.

Some refinements about the strategies to handle IF statements within DO blocks, and a better way to get higher efficiency measurement within each block are under consideration which will be the major objectives for the next quarter.

9.1.1 ASSIGN - (P. Kraska)

- 1) Transcendental functions have been replaced by a tree of height ≥ 3 , depending on the complexity of the argument expression.
- 2) Block-assignment operators have been included in the tree-height calculation and the total number of operations.
- 3) RECUR(sion) procedure was re-written to permit more than ten iterations of a DO-loop. This procedure will be further modified next quarter so that non-subscripted left-hand-side expressions are not accumulated.

4) Several other procedures and declarations were modified to make program execution more efficient.

5) During the next quarter several modifications to ASSIGN are expected to be made which will either correct minor errors or improve program flexibility so that a larger class of FTN programs may be analyzed.

9.1.2 Scanner - (R. Towle)

The past quarter has been spent stabilizing the collection of routines. An overlay structure was developed so that over 600 K of programs could fit in 450 K. The code in ASSIGN and HAN was cleaned up. Several static four-dimensional arrays have been converted to dynamic linked structures.

Most of the quarter has been spent working on the scanner. Ed Davis' IF-expander has been integrated so that IF statements can be expanded as the FORTRAN program is being scanned. Several crucial entries in PROGRAM that have been incorrect have been fixed. The ability to preassign values to variables was rewritten and debugged. Several undetected infinite loops have been eliminated. The complete collection of FORTRAN built-in functions is being recognized. Operator counts are now kept for each assignment statement. This count does not include those for subscript arithmetic.

9.1.3 IF Tree Processing - (D. Wills)

During the last quarter a program was written and debugged that detected IF trees in FORTRAN programs. IF trees are made up of nodes (IF statements) and branches (logical flow of the program between nodes). The program will test several FORTRAN programs to see if IF trees can be handled well by a multi-processor machine. Loops within the IF tree are recognized

and marked as such. All the assignment statements within single IF trees are collected and passed to ASSIGN, which handles assignment statements in a multi-processor machine.

9.1.4 Statistical Package - (R. Strebendt)

During this quarter a statistics processing program was written to present the results of the analysis program in readable format and to accumulate various statistical data for the FORTRAN program studied. This program includes the following routines:

STSEXEC - Statistics Processing Executive. Based on a set of control options supplied by the user, a list of requested and required tasks is constructed. Based on the list of available tasks for each input record type and the list of requested and required options, a sequence of subroutine calls is initiated to do all of the desired processing for each individual statistical record and for the composite data from the records for the complete analysis of the FORTRAN program.

PTRACE - Path Trace. All possible control flow paths in the FORTRAN program are explored and statistical data is accumulated for each path.

HISTOGRAM - Histogram Printout. The data generated by the path tracing program is printed graphically as a set of histograms.

In order to obtain the statistical data several of the analysis program routines were modified to collect and to output this data into the file used as input to the statistical processing program.

9.2 Weighted Operator, Expression Parsing and Scheduling - (P. Kraska)

As predicted in the previous quarter, an algorithm was developed which optimally parses an expression of the form

$$e = (\prod_{i=1}^n f_i) / (\prod_{i=1}^m f'_i)$$

such that tree-height is minimized by appropriately selecting denominator factors to be divided into the numerator until a balance of numerator and denominator is achieved. This algorithm was included in the PL/1 program which calculates the expression parse such that tree-height is near-optimally low.

During the next quarter I will complete the analysis of the weighted-node scheduling algorithm developed earlier for a parallel system of processors working on a weighted-node, directed graph. Other processor configurations will also be investigated.

9.3 Simulation Processor - (E. W. Davis)

Evaluation of the performance of the simulation machine will be carried out on a software system that is nearing completion. A program that analyzes GPSS simulation programs for concurrently executable blocks (routines) has been completed. This program prepares test GPSS code for input to the system and gathers some statistics on probable performance. Two routines have been completed which gather run time parameters from actual runs of test programs then input this data during evaluation runs.

The system will be complete when a simulator of the machine is completed. One element of the simulator is a table of execution times for the GPSS blocks on a multiprocessor machine. To get this table GPSS blocks are being converted to FORTRAN for analysis on the parallelism extraction system discussed in other sections of this report.

Several test simulation programs have been acquired. Experiments with these programs on the existing parts of the system are encouraging.

9.4 Text Searching - (W. H. Stellhorn)

Coding is complete for the first version of the interactive language for text searching applications, and the program is currently being debugged by means of the B5500 simulation for the D-machine, modified to produce s-level simulations. Simulation will continue during the next quarter, and it is hoped that hardware will become available to allow extensive testing of the program on the D-machine itself.

9.4.1 D-Machine Microprogram and Simulator - (J. R. Rinewalt)

The microprogram for word and string instructions was rewritten for optimization. Apparently, much of the inefficiency in the original code was due to errors in the APL version of the simulator. Three modifications to the basic operation of the instructions were made and two new word instructions were implemented to increase the flexibility of the S-level language. There are now approximately 400 words of microstore available for I/O instructions and growth.

The D-machine simulator was also modified to provide S-level simulation capability. For S-level simulation, the output of the simulator now consists of the S-level memory locations changed by the instruction, and the number of D-machine cycles required to execute the instruction.

9.4.2 I/O Software - (R. Goldberg)

The bootstrap and the D-machine absolute loader (DIABLO) are written, and now that the hardware modifications have been completed, debugging will start. The I/O instruction decoder is presently being coded, and should be operational within a month.

9.4.3 D-Machine Assembler - (E. J. Polley, Jr.)

The assembler for the D-machine is nearly complete. All the instructions except those concerned with I/O are implemented and working. This includes the special string processing instructions which will be useful in file manipulation work.

The only object cards being punched currently are for the B-5500 simulator but cards compatible with the D-machine loader will be capable of being produced by early February. As soon as the format for the I/O instructions are settled they will be implemented.

Two long programs have been assembled thus far, a 570 statement program in 2.82 seconds and a 1469 statement one in 5.87 seconds. The time for the longer program means an assembly rate of 15000 cards a minute. This speed is achieved of course by keeping the intermediate file in core.

Publications

Cartegini, C., "Scanner for the Analysis of Parallelism in Fortran Programs and IF-Tree Detection," (M.S. Thesis) Department of Computer Science, University of Illinois at Urbana-Champaign, 1971.

Han, J., "Tree Height Reduction for Parallel Processing of Blocks of Fortran Assignment Statements," (M.S. Thesis) Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-72-493, 1972.

Swanson, L., "Simulation of a Tree Processor," (M.S. Thesis) Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-72-503, 1972.

Budnik, P., and D. Kuck, "The Organization and Use of Parallel Memories," IEEE Trans. Comput., vol. C-20, Dec. 1971, pp. 1566-1569.

ABSTRACT: As computer CPUs get faster, primary memories tend to be organized in parallel banks. The fastest machines now being developed can fetch of the order of 100 words in parallel. Unless memory and compiler designers are careful, serious memory conflicts and resulting performance degradation may result. Some of the important questions of design and use of such memories are discussed.

Kuck, D., and A. Sameh, "Parallel Computation of Eigenvalues of Real Matrices,"
IFIP Congress 71, Ljubljana, August, 1971, Booklet TA-1, pp. 24-28.

ABSTRACT: To realize the full speed potential of forthcoming array computers, one must properly organize both an algorithm for the arithmetic unit and the data in a highly interleaved memory. We consider matrix eigenvalue calculations by the Jacobi, Householder and QR methods. In each case we present storage schemes for the matrices in a parallel memory which allows simultaneous access to proper elements. In terms of these storage schemes we discuss parallel implementation of the above mentioned algorithms and compute over-all efficiencies of machine use.

10. COMPUTER SYSTEMS ANALYSIS

(Supported in part by the National Science Foundation under Grant No. NSF GJ 28289.)

The goal of this research is the development of analytical tools for system modeling and analysis of real time computer networks. A queueing theory model for a geographically distributed computer network is being developed. Priority assignment and job dispatching rules for the network are being investigated.

10.1 Computer Network Modeling (W. McKinney and L. Mills)

We have simplified the notation and made some minor improvements in our formal model (developed last quarter). We are currently investigating nonpriority finite length queues with multiple servers having different service rates. We have solved the two server problem and work is in process for the three and four server problems. We hope to be able to formulate a general solution to the n server problem based on this limited data.

Under the assumptions of Poisson input and exponentially distributed service times, we are using a GPSS model to simulate the effects of maximum queue length and the number of servers on waiting times for nonpreemptive priority queues. Additionally, we are investigating network efficiency as a function of the number of servers. We plan to simulate these effects under the assumption of a hyperexponential input distribution.

10.2 Center Throughput Analysis (S. Mamrak and W. Barr)

We have developed an algorithm for scheduling n jobs having arbitrary loss functions which can reduce by 2^{n-4} the number of calculations required in the dynamic programming solution. Given a set of n jobs $\{J_1, J_2, \dots, J_n\}$,

each of which has a known processing time and an arbitrary loss function (that specifies the loss associated with completing the job at any given time), we define an ordering of jobs, which when executed nonpreemptively on a single machine minimizes the total loss. Our approach is to use a modified dynamic programming algorithm in which we carefully monitor the partial schedules generated to enable us to eliminate unnecessary calculations.

We have determined that our measure of goodness, γ , is very indicative of the level of activity in computing centers. So far, we have examined the throughput in the Illinet IBM System 360/75 and have found a high degree of correlation between γ and the turnaround time for jobs in different priority classes. We are continuing to investigate this correlation using current 360/75 throughput data. We have also developed an algorithm for job dispatching between centers based on γ . We plan to use the GPSS model of the network to evaluate the efficiency of the algorithm.

(E. K. Bowdon)

11. THEORY OF DIGITAL COMPUTER ARITHMETIC

11.1 Continued Fraction Arithmetic

The generality of the continued fraction method for finding a solution of a quadratic equation was extended during the quarter. The quadratic is written in the form:

$$x^2 + b_k x - c_k = (x-u)(x+v) = 0 \quad (1)$$

The problem is, given the coefficients b_k and c_k , to find the root u as a continued fraction. Since $c_k = uv$ and $b_k = v-u$, the value of the magnitude v of the negative root is easily determined when u is known.

At this point, it is convenient to delineate four areas in the c_k, b_k plane and relate each area to properties of the root magnitudes u and v .

- 1) $c_k < -\frac{b_k^2}{4}$. Both roots are imaginary.
- 2) $-\frac{b_k^2}{4} \leq c_k < 0$. Both roots are real and of the same sign.
- 3) $c_k \geq 0, b_k < 0$ (second quadrant). The roots are real and of opposite sign, with $u > v$.
- 4) $c_k \geq 0, b_k \geq 0$ (first quadrant). The roots are real and of opposite sign, with $v \geq u$.

It is first shown that any point in the first quadrant of the c_k, b_k plane may be scaled to lie within a triangular wedge such that $\frac{1}{2} \leq u \leq 1$. Since $v = u + b_k$ and $c_k = uv$, it follows that $c_k = u b_k + u^2$, and the range $\frac{1}{2} \leq u \leq 1$ is equivalent to

$$\frac{1}{2} b_k + \frac{1}{4} \leq c_k \leq b_k + 1. \quad (2)$$

Multiplying equations (1) and (2) by 2^{2j} (j an integer) yields

$$(2^j x)^2 + (2^j b_k)(2^j x) - 2^{2j} c_k = 0 \quad (3)$$

$$2^{j-1}(2^j b_k) + 2^{2(j-1)} \leq 2^{2j} c_k \leq 2^j(2^j b_k) + 2^{2j} \quad (4)$$

Let $2^j x = x'$, $2^j b_k = b'_k$, and $2^{2j} c_k = c'_k$. Then

$$(x')^2 + b'_k x' - c'_k = 0 \quad (5)$$

$$2^{j-1} b'_k + 2^{2(j-1)} \leq c'_k \leq 2^j b'_k + 2^{2j} \quad (6)$$

Given c'_k and b'_k , the scaling procedure is then:

- a) Determine the value of j , such that equation 6 is satisfied.
- b) Multiply c'_k and b'_k by 2^{-2j} and 2^{-j} , respectively, to obtain c_k and b_k , which satisfy equation 2.
- c) When the root u is determined, find the positive root u' of equation 5, by scaling u in accordance with $u' = 2^j u$.

Note that the scaling procedure reduces to that normally employed for square roots in floating point computers, when $b_k = 0$.

For any point c'_k, b'_k in the first quadrant, an integer value of j can be found such that equation 6 is satisfied. It is therefore sufficient, for the first quadrant, to solve equation 1 subject to the constraints of equation 2, with $b_k \geq 0$.

For the second quadrant, with $b_k < 0$ it is sufficient to replace $b_k = v-u$ by $b'_k = -b_k = u-v$. Equation 1 becomes

$$x^2 + b'_k x - c_k = (x-v)(x+u) = 0. \quad (7)$$

Solution of (7) yields the magnitude v of the negative root. The value of u is then $u = b_k'' + v$.

The solution for the case of two imaginary roots has not been considered. Attempts to find a method of solution for two real roots of the same sign have thus far been unsuccessful. The preceding observations, however, indicate that a continued fraction solution of the quadratic can be found if $c_k \geq 0$; i.e., if the two roots are real and are of opposite sign.

James E. Robertson

11.1.1.1

Let $x^2 + b_k x - c_k = (x-u)(x+v) = 0$ be a quadratic equation with $b_k \geq 0$ and $c_k > 0$. A solution u of the form

$$u = \frac{1}{q_1} + \frac{1}{q_2} + \dots + \frac{1}{q_n} + \dots$$

where $q_i \in \{1, 1/2, 1/4\}$, has previously been achieved, including recursion relations, for b_i, c_i , selection rules for q_i , and proof of convergence.* The generalization of the above method was obtained by using a solution of the form

$$u = \frac{p_1}{q_1} + \frac{p_2}{q_2} + \dots + \frac{p_n}{q_n} + \dots$$

where $p_i, q_i \in \{1, 1/2\}$. Recursion relations for b_i, c_i and selection rules for p_i, q_i were developed. The proof of convergence is similar to the previous case.

Amnon Bracha

*Quarterly Technical Progress Report, Department of Computer Science, University of Illinois, July-September 1971.

The algorithm for square rooting using continued fractions, as mentioned in an earlier report,^{*} has been successfully extended to a limited class of quadratics given by,

$$x^2 + b_k x - c_k \equiv 0 = (x-u)(x+v)$$

for $\frac{1}{2} \leq u \leq 1^{\dagger}$ and $b_k \geq 0$. In terms of c_k and b_k , these restrictions are, $(c_k - \frac{1}{2} b_k) \geq \frac{1}{4}$, $(c_k - b_k) \leq 1$ and $b_k \geq 0$.

Following is the algorithm QD.

QD-0: [Check] If $b_k < 0$ or if $(c_k - \frac{1}{2} b_k) < \frac{1}{4}$ or if $(c_k - b_k) > 1$

then algorithm terminates without a solution; otherwise,

QD-1: [Range] Set $J_1 \leftarrow J_2 \leftarrow 0$; if $(c_k - \frac{5}{8} b_k) < \frac{25}{64}$ then set

$q_1 \leftarrow 1$, and go to step QD-2; otherwise set $q_1 \leftarrow \frac{1}{2}$; now

if $(c_k - \frac{3}{4} b_k) < \frac{9}{16}$ then set $J_1 \leftarrow 1$, and go to step QD-2;

otherwise set $J_2 \leftarrow 1$,

QD-2: [Initialize] Set $P_0 \leftarrow 0$, $Q_0 \leftarrow P_1 \leftarrow 1$, $Q_1 \leftarrow q_1$,

$b_{k-1} \leftarrow q_1 c_k$, $c_{k-1} \leftarrow 1 + q_1 (b_k - b_{k-1})$, $i \leftarrow 2$;

QD-3: [Select] If $(c_{k-i+1}) > (b_{k-i+1} + \frac{1}{2} + \frac{J_1}{8} + \frac{J_2}{4})$ then set

$q_i \leftarrow \frac{1}{4}$ and go to step QD-5; otherwise if c_{k-i+1}

$\leq (\frac{1}{2} b_{k-i+1} + \frac{5}{16} + \frac{J_1}{16} + \frac{3J_2}{16})$ then set $q_i \leftarrow 1$ and go

to step QD-5; otherwise set $q_i \leftarrow \frac{1}{2}$;

^{*}Quarterly Technical Progress Report, April-June 1971, PP 142-143.

[†]Here we have assumed a conditional scaling of b_k and c_k has been done by Robertson's method (given on P.106 of this report), so that this condition is satisfied.

QD-4: [Advance] Set

$$b_{k-i} \leftarrow q_i c_{k-i+1} - q_{i-1} c_{k-i+2} + b_{k-i+2},$$

$$c_{k-i} \leftarrow q_i (b_{k-i+1} - b_{k-i}) + c_{k-i+2},$$

$$P_i \leftarrow q_i P_{i-1} + P_{i-2},$$

$$Q_i \leftarrow q_i Q_{i-1} + Q_{i-2},$$

$$i \leftarrow i + 1;$$

QD-5: [Loop Test] If $i \leq i_{\max}$ then go to step QD-4; otherwise,

QD-6: [Final] $u (= \text{root}) \leftarrow P_i / Q_i$.

The value of $i_{\max}$ will be decided by machine precision, in case, this algorithm is implemented in hardware. If this algorithm is implemented in software, however, the value of $i_{\max}$ will be dictated by the allowable error.

It should be noted that the algorithm requires only shifts and adds in each iterative step. A division is necessary only in the final step.

A complete proof of convergence of this algorithm is now available and will soon be published in a Master's thesis.

Kishor Trivedi

12. Binary Search Trees of Bounded Balance

Jurg Nievergelt
Edward M. Reingold

Binary search trees are an important technique for organizing large files, because they are efficient for both random and sequential access of records. We are interested in the case that the set of names is dynamic, i.e. it changes in time through insertions and deletions. To improve the search time over that of trees which have grown at random, one looks for trees which satisfy the following conflicting requirements: they must be close to being balanced, so that the search time is short; one must be able to restructure them with little effort when they have grown too far away from being balanced; and restructuring should be required only rarely. Adel'son-Vel'skii and Landis described a class of trees, called AVL trees, which strike an elegant compromise between these conflicting requirements.

We have introduced a new class of binary search trees, called trees of bounded balance, or BB trees for short. BB trees share with the AVL trees the property that they are easy to maintain in their form despite insertions and deletions of nodes, and that search time is only moderately longer than in balanced trees. They differ from AVL trees in two important respects:

- they contain a parameter which can be varied so the compromise between short search time and frequency of restructuring can be chosen arbitrarily, and
- the insertion and deletion algorithms do not require a pushdown store.

13. ADAPTIVE ALGORITHMS FOR ELLIPTICAL DIFFERENCE EQUATIONS

During this quarter, Martin A. Diamond completed his doctoral thesis, "An Economical Algorithm for the Solution of Elliptic Difference Equations Independent of User-Supplied Parameters."* Diamond's algorithm is of interest for three reasons:

- 1) It is superior to other methods for certain difficult problems, according to experimental evidence;
- 2) It requires no a priori parameter estimates; and
- 3) It may be applied to variational difference equations as well as finite difference equations.

During this quarter, I began work with M. Heidari of the Illinois State Geological Survey, on the numerical solution of the diffusion convection equation that describes the flow of liquid waste injected in porous substrata under high pressure. Galerkin's Method or the Finite Element Method seems essential for obtaining an accurate solution. We are adapting Diamond's algorithm to apply to this problem.

(Paul E. Saylor)

* Diamond, Martin A., "An Economical Algorithm for the Solution of Elliptic Difference Equations Independent of User-Supplied Parameters," Department of Computer Science, Report No. UI-UCDCS-R-71-492, December 5, 1971.

Publication

Diamond, Martin A., "An Economical Algorithm for the Solution of Elliptic Difference Equations Independent of User-Supplied Parameters", Department of Computer Science, Report No. UIUCDCS-R-71-492, December 5, 1971.

60

14. COMPUTER LANGUAGES FOR MATHEMATICS AND NUMERICAL ANALYSIS

We present abstracts of Digital Computer Laboratory Reports which are related to the work of this quarter. This research has been supported by the National Science Foundation under NSF Grant GJ-328.

Phillips, J. Richard. "The Structure and Design Philosophy of OL/2 - An Array Language - Part II: Algorithms for Dynamic Partitioning." Report No. UIUCDCS-R-71-420.

ABSTRACT: This report contains a formal description of dynamic partitioning as it is defined for the OL/2 language. Various modes of partitioning are defined for different types of arrays in such a way that the subarrays which result from partitioning are allowed to vary over the original array in a nearly arbitrary manner. The information structure which is used to implement dynamic partitioning is described in detail. The last part of the report is concerned with the algorithms which build and maintain the information structure. These algorithms are described in a language independent form and are valuable to anyone who is concerned with the design of array languages.

Sanford, Wayne C. "An Interactive Syntax Analyzer." Report No. UIUCDCS-R-72-504.

ABSTRACT: This report is concerned with the design of an interactive syntax analyzer for the OL/2 language. The main purpose of the syntax analyzer is to allow a user to construct statements on-line and have the analyzer identify any syntactic errors before proceeding to the next statement. This allows a program to be written which is free of

all syntatic errors. This is accomplished without actually compiling any code and implemented using a compiler-compiler. It is clear that this approach to error detection allows a syntatically correct source program to be passed to a production compiler. Other applications and extensions of interactive syntax analyzers are also suggested in this report.

REFERENCES

- Phillips, J. Richard. "The Structure and Design Philosophy of OL/2 - An Array Language - Part II: Algorithms for Dynamic Partitioning," Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-71-420, September, 1971.
- Sanford, Wayne C. "An Interactive Syntax Analyzer," Department of Computer Science, University of Illinois at Urbana-Champaign, Report No. UIUCDCS-R-72-504, January, 1972.

15. NUMERICAL METHODS, COMPUTER ARITHMETIC AND ARTIFICIAL LANGUAGES

(Supported in part by the National Science Foundation under Grant No. US NSF GJ 812.)

15.1 Educational Timesharing System

The Educational Timesharing System (ETS) became operational in December 1971. We are now in the process of acquiring some on-line usage in order to find any bugs we may have missed. Students who have used the system are enthusiastic about it and are interested in continuing work on it. In addition to testing and tuning the system, we are spending much effort in bringing the documentation up to date. The System User's Guide which provides instructions for users who write programs which will run without the benefit of an interpreter is ready for the final draft. Documentation for student users is being assembled from separate user's guides for each of the system routines. This manual is also in the typing stage. Documentation on the monitor proper is about 50% complete.

Even though we have a working system, there are parts which we want to upgrade or improve. We are also making plans for the addition of more core to the system. This basically involves increasing the user area; however, the current assembler is not dynamically relocatable. Until our new assembler is done, we must have the system always swap the assembler back to the same place. We intend to create an "anchor mode" of execution in which the system will force a job to return to its last position when it is swapped in.

DECTape drivers have been written and are being integrated into the system. The DECTape filing system is also expected to be complete within a week.

We are also looking for interested students who would like to work on the system. We are trying to create a library of demonstration

routines for use with non-programmers. These will consist of special lessons for GIZMO and some game playing programs.

Most of the design work is complete for a new assembler; however, work on it has been delayed until the DECTape system is complete.

Work on a new card reader driver for Educational Timesharing System on the PDP-11 was begun and is complete. Work was delayed for a time when the card reader's speed was increased from 200 CPM to 300 CPM and is currently delayed by hardware problems in both card reader and PDP-11 CPU.

The authors were studying ETS to familiarize themselves with the many aspects of this complex program. Working with the card reader was helpful in learning about the PDP-11 interrupt programming of peripherals. Writing future programs will be helpful in learning other aspects of ETS.

Since disk space on the PDP-11 is small, PAL (assembler) source decks must be compressed so that more students can store their source decks on disk. Research was conducted in the field of program compaction to find feasible algorithms for compressing assembler source decks. A simple algorithm now in use compresses blanks by substituting a string of them by the number of blanks. Since all ASCII characters are positive, this negative number embedded in the source code is easily recognizable.

(Bob Feretich, Barry Finkel, Jim Hart, Don Oxley)

15.2 DOS Batch

Extensive modifications and additions to Digital Equipment Corporation's Disk Operating System for the PDP-11 (referred to as DOS) have resulted in a batch system for the PDP-11 which supports over 500 jobs/day (average job = 300 cards and 300 lines). The system is largely protected from student errors (all programs are assembly language) by a run-time interpretive debugging program with many diagnostic aids.

The system software includes an assembler and linker provided by DEC and modified here. Original software includes the spooling facilities and interpreter program. Current programming is oriented toward improvement of system performance and reliability.

DOS batch was in effect for the last half of the fall 1971 semester supporting a section of CS 201. Initial reliability was good, and improved quickly. The students operated the system facilities with little assistance outside of teaching staff.

(Russ Atkinson, Ian Stocks)

15.3 GIZMO

An early version of GIZMO, a CAI system implemented on the PDP-11 was completed in June 1971. During fall semester 1971-72, twenty CS 201 students, under the direction of myself, completed the structure and plans for an extended GIZMO with the following facilities:

1. An interactive teacher-mode which makes it possible for a teacher, without previous experience with a computer, to construct a complete lesson easily.

2. A lesson compiler capable of accepting the result of the teacher-mode and outputting a lesson in perfect format for GIZMO student-mode.
3. An expanded student-mode capable of understanding, among other things, student's arithmetic expressions and more importantly, capable of analyzing the BNF-like structure of the teacher's lesson.

User's manuals are in the process of being written. The program itself is now almost ready for student and teacher use in the timesharing environment supplied by ETS on the PDP-11

(Al Davis)

15.4 PDP-11 Hardware

Few changes have been made in the PDP-11 hardware over this quarter. The GDI card reader was upgraded from 200 pm to 300 pm. The programmer for the Intel read only memory chips was completed but the ROM PC board for the PDP-11 has been delayed due to heavy work loads in the PC shop. Sound insulating cases have been installed on all available teletypes. The cases have significantly dampened the noise in room 123. An off-line tester was built for the ill-fated TTY circuit cards. Sufficient interfaces and spares are now available to handle 12 TTY's. The long awaited additional 12K of core memory is on the way and will be installed in the next quarter.

(Jim Miller)

15.5 PDP-11 Hardware Test Program Library Monitor

The Hardware Test Program Library Monitor was written to provide an easy and convenient method to store, update and use DEC's main-dec maintenance programs. These programs are useful in diagnosing machine or peripheral errors and are supplied as object paper tapes. There are more than fifty such programs, all with cryptic identifying names. These paper tapes are awkward to use and store. It was, therefore, decided to store the equivalent programs on magnetic DECTapes. DECTapes are DEC's unique, extremely reliable magnetic tapes. They are 3/4 inch wide, but on a small (approximately 4") reel. In order to keep track of these programs, once stored on DECTape, a monitor system had to be developed.

The monitor is essentially a program library and associated maintaining routines. Object-paper tapes can be copied from the high speed reader to DECTape. A directory is maintained and can be listed. Programs are stored sequentially on DECTapes and are called for by a 5 character abbreviation. The directory includes all abbreviations and a 20. character description for each program on the tape. Additions and deletions are easily made. Any program in the directory can be loaded and executed from the DECTape.

Three devices must be available to use the monitor: the processor, the DECTape drives (at least one) and the teletype. The monitor is self-explanatory with operational instructions (except those dealing with loading the monitor and mounting tapes) printed on the teletype.

The monitor performs the following functions:

1. Printing of operational instructions and explanation of other functions.

2. Initialization of new tapes for use with the monitor.
3. Listing directory of tape (both abbreviations and descriptions).
4. Purging (clearing) of entire directory.
5. Addition of programs to the DECTape from paper object tapes. Abbreviations and descriptions to correspond with the program are entered on the teletype keyboard. A maximum of 101. entries is allowed per DECTape and no more than approximately 540. blocks of the tape may be used.
6. Deletion of programs from the directory. The space used by the deleted program is not reclaimed until the COPY option is used on the tape.
7. Recopying of the directory and tape, thus, eliminating space wasted by deleted programs (see above). Optionally, a back-up copy (before space reclamation) may be produced.
8. Loading and executing a program from the library tape. Control is not returned to the monitor, which must be reloaded via the bootstrap (see note 4 below).
9. If a program is not executed, the monitor may be restarted and all options are again available. Many user errors also allow the restart option if desired.

Notes:

1. The monitor is non-interrupt driven and utilizes no operating system. Consequently, the disk is never needed.
2. In the case of duplicate entries in the directory, the program available for use will be the first one entered in the directory that has not subsequently been deleted.

Thus, if two or more programs with the same abbreviation are present in the directory, the first program would be used. If the first entry with the abbreviation is then deleted, the second program would be used.

3. Any number of programs may be stored as each DECTape is self-contained and any number of DECTapes may compose a library. Each DECTape has both a copy of the monitor and its own directory after it is initialized.
4. The monitor is invoked by entering a short bootstrap program either: A) by hand, via the switch register; B) from a paper tape if an absolute loader is in core at the time; or C) from read-only memory. The bootstrap loads the monitor from a (pre-initialized) library tape. The teletype will then print instructions for use of the monitor.
5. The only time more than one tape unit is needed is for recopying, and then only if a back-up copy is desired.
6. The monitor asks the user to specify which tape units are to be used for library tapes and which for back-up tapes. This is useful if one or more of the DECTape units are malfunctioning. The selected units must be set for remote operation and "write UNLOCKED" manually.
7. Programs stored on the library will not be loaded higher than a fixed core location (currently 40000₈). Thus, the monitor and system bootstrap, absolute loader, etc. will remain intact unless diagnostic programs cause overwriting of these programs.

8. Although the monitor was written as a "Hardware Test" monitor, many other programs may be stored on the library tapes and retrieved when needed, without the use of any operating system (ETS, DOS, etc.).

(Jim Block)

16. GENERAL DEPARTMENT INFORMATION

16.1 Personnel

The number of people associated with the Department in various capacities is given in the following table:

	<u>Full-time</u>	<u>Part-time</u>	<u>Full-time Equivalent</u>
Faculty	22	4	23.97
Visiting Faculty	3	0	3.00
Research Assoc. and Instructors	2	0	2.00
Graduate Research Assistants	1	65	33.00
Graduate Teaching Assistants	0	28	13.625
Professional Personnel	10	1	10.50
Administration and Clerical	19	0	19.00
Nonacademic Personnel (Monthly)	18	1	18.50
Nonacademic Personnel (Hourly)	0	54	19.82
TOTAL	73	153	143.415

*This report does not include personnel employed on the ILLIAC IV Project.

The Department Advisory Committee consists of Professor J. N. Snyder, Head of the Department, Professors E. K. Bowdon, D. F. Cudia, K. W. Dickman, M. F. Faiman, H. G. Friedman, C. W. Gear, D. B. Gillies, W. J. Kubitz, D. J. Kuck, B. H. McCormick, R. G. Montanelli, S. Muroga, T. A. Murrell, J. Nievergelt, J. R. Phillips, W. J. Poppelbaum, S. R. Ray, E. M. Reingold, J. E. Robertson, P. E. Saylor, D. L. Slotnick, J. E. Vander Mey, D. S. Watanabe, and T. Wilcox.

16.2 Bibliography

During the fourth quarter, the following publications were issued by the laboratory:

Report Numbers

- No. 17 Stevens, James, "A Fast Fourier Transform Subroutine for ILLIAC IV.", October 13, 1971.
- No. 19 Bernhard, Winifried, "ILLIAC IV Codes for Jacobi and Jacobi-like Algorithms.", November 5, 1971.
- No. 21 McDaniel, Lawrence, "A Jacobi Algorithm for ILLIAC IV.", November 8, 1971.
- No. 254 Systems Control, Inc., "Study of the Radar Signal Data Processing Tasks on ILLIAC IV.", November 19, 1971.
- No. 255 Lermitt, J. and Randal, J. M., "Augmented Significance Routines for ILLIAC IV.", October 8, 1971.
- No. 256 Randal, J. M., "A New ILLIAC IV Timing Simulator", December, 1971.
- No. 487 Kuck, David J. and Sameh, Ahmed, "PARALLEL COMPUTATION OF EIGENVALUES OF REAL MATRICES", November 1, 1971.
- No. 488 Nakagawa, Tomoyasu and Lai, Hung-Chi, "REFERENCE MANUAL OF FORTRAN PROGRAM ILLOD-(NOR-B) FOR OPTIMAL NOR NETWORKS", December 1971.
- No. 489 Maruyama, Kiyoshi, "AN APPROXIMATION METHOD FOR SOLVING THE SOFA PROBLEM", October 1971.
- No. 490 Maruyama, Kiyoshi, "A PROBLEM IN FORM PERCEPTION: ODD DETECTION", December 1971.

Theses

- No. 483 Carr, Edward Ellis, "A SYNCHRONIZATION SYSTEM FOR A TELEVISION BANDWIDTH COMPRESSION SCHEME", October 1971, (M.S.).
- No. 484 Cheng, Richard Tien-Ren, "COEFFICIENT GENERATOR AND CURSOR FOR THE STEREOMATRIX β -D DISPLAY SYSTEM", October 1971, (PhD.).
- No. 492 Diamond, Martin A., "AN ECONOMICAL ALGORITHM FOR THE SOLUTION OF FINITE DIFFERENCE EQUATIONS", December 1971, (PhD.).
- No. 493 Han, J., "TREE HEIGHT REDUCTION FOR PARALLEL PROCESSING OF BLOCKS OF FORTRAN ASSIGNMENT STATEMENTS", 1972, (M.S.).

16.3 Colloquia

"Analysis and Synthesis of Sorting Algorithms", by Professor C. L. Liu, Department of Electrical Engineering and Project MAC, Massachusetts Institute of Technology, October 4, 1971.

"The MERIT Computer Network", by Professor Bertram Herzog, Department of Computer Science, 1037 North University Building, University of Michigan, Ann Arbor, Michigan 48104, October 11, 1971.

"Who Really Won the Pennant? Ranking Methods for Paired Comparison Experiments", by Professor John S. de Cani, Department of Statistics and Operations Research, University of Pennsylvania, Philadelphia, Pennsylvania, October 18, 1971.

"Interactive Modeling Using a RAND Tablet", by Dr. Edward C. DeLand, Physical Sciences Department, The Rand Corporation, 1700 Main Street, Santa Monica, California 90406, October 25, 1971.

"System Concepts and Performance Characteristics of the G. E. Light Valve Color T.V. Projector", by Dr. William E. Good, General Electric Company, Electronics Park, Syracuse, New York 13201, November 1, 1971.

"Requiem for the Lambda Calculus", by Professor Clement L. McGowan, Division of Applied Mathematics, Brown University, Providence, Rhode Island 02912, November 8, 1971.

"Survey of the Current State of the Art in Matrix Computation and Some Recent Results in Generalized Eigenvalue Problems", by Professor Cleve Moler, University of Michigan, November 18, 1971.

"The Implementation of SPITBOL", by Professor Robert B. K. Dewar, Department of Computer Science, Illinois Institute of Technology, Chicago, Illinois 60616, November 12, 1971.

"Tandem Queueing Systems with Constant Service", by Professor Donald L. Epley, Department of Electrical Engineering, University of Iowa 52240, November 15, 1971.

"Computer Graphics in Molecular Biology", by Professor Robert Langridge, Department of Biochemical Sciences, Princeton University, Princeton, New Jersey 08540, November 22, 1971.

"On Classes of Program Schemes", by Professor David Gries, Department of Computer Science, Cornell University, Ithaca, New York, December 13, 1971.

16.4 Drafting

During the fourth quarter, a total of 263 drawings were processed by the general departmental drafting section:

Large Drawings	85
Medium Drawings	44
Small Drawings	83
Layouts	1
Report Drawings	15
Change Order Drawings	31
Miscellaneous	<u>4</u>
Total	263

(M. Goebel)

16.5 Shop's Production

Job orders processed and completed during the fourth quarter of 1971 are as follows:

	<u>AEC 2118</u>	<u>AEC 1469</u>	<u>Other</u>
Machine Shop	5	7	3
Electronic Shop	1	51	8
Chemical Shop		33	5
Layout Shop		23	4

(F. P. Serio)

DEPARTMENT OF COMPUTER SCIENCE
GRADUATE COLLEGE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

REPORT REQUEST FORM

Report Number

Title

Fold and Staple as shown

NAME:

ADDRESS:

Fill out Mailing Label

NAME:

ADDRESS:

CUT ALONG THIS LINE

FOLD

Stamp

Mail Room
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

FOLD

STAPLE

INSTRUCTIONS FOR COMPLETING FORM NTIS-35 (10-70) (Bibliographic Data Sheet based on COSATI

Guidelines to Format Standards for Scientific and Technical Reports Prepared by or for the Federal Government, PB-180 600).

1. **Report Number.** Each report shall carry a unique alphanumeric designation. Select one of the following types: (a) alphanumeric designation provided by the sponsoring agency, e.g., **FAA-RD-68-09**; or, if none has been assigned, (b) alphanumeric designation established by the performing organization e.g., **FASEB-NS-87**; or, if none has been established, (c) alphanumeric designation derived from contract or grant number, e.g., **PH-43-64-932-4**.
2. Leave blank.
3. **Recipient's Accession Number.** Reserved for use by each report recipient.
4. **Title and Subtitle.** Title should indicate clearly and briefly the subject coverage of the report, and be displayed prominently. Set subtitle, if used, in smaller type or otherwise subordinate it to main title. When a report is prepared in more than one volume, repeat the primary title, add volume number and include subtitle for the specific volume.
5. **Report Date.** Each report shall carry a date indicating at least month and year. Indicate the basis on which it was selected (e.g., date of issue, date of approval, date of preparation).
6. **Performing Organization Code.** Leave blank.
7. **Author(s).** Give name(s) in conventional order (e.g., John R. Doe, or J. Robert Doe). List author's affiliation if it differs from the performing organization.
8. **Performing Organization Report Number.** Insert if performing organization wishes to assign this number.
9. **Performing Organization Name and Address.** Give name, street, city, state, and zip code. List no more than two levels of an organizational hierarchy. Display the name of the organization exactly as it should appear in Government indexes such as **USGRDR-I**.
10. **Project/Task/Work Unit Number.** Use the project, task and work unit numbers under which the report was prepared.
11. **Contract/Grant Number.** Insert contract or grant number under which report was prepared.
12. **Sponsoring Agency Name and Address.** Include zip code.
13. **Type of Report and Period Covered.** Indicate interim, final, etc., and, if applicable, dates covered.
14. **Sponsoring Agency Code.** Leave blank.
15. **Supplementary Notes.** Enter information not included elsewhere but useful, such as: Prepared in cooperation with . . . Translation of . . . Presented at conference of . . . To be published in . . . Supersedes . . . Supplements . . .
16. **Abstract.** Include a brief (200 words or less) factual summary of the most significant information contained in the report. If the report contains a significant bibliography or literature survey, mention it here.
17. **Key Words and Document Analysis.** (a). **Descriptors.** Select from the Thesaurus of Engineering and Scientific Terms the proper authorized terms that identify the major concept of the research and are sufficiently specific and precise to be used as index entries for cataloging.
(b). **Identifiers and Open-Ended Terms.** Use identifiers for project names, code names, equipment designators, etc. Use open-ended terms written in descriptor form for those subjects for which no descriptor exists.
(c). **COSATI Field/Group.** Field and Group assignments are to be taken from the 1965 COSATI Subject Category List. Since the majority of documents are multidisciplinary in nature, the primary Field/Group assignment(s) will be the specific discipline, area of human endeavor, or type of physical object. The application(s) will be cross-referenced with secondary Field/Group assignments that will follow the primary posting(s).
18. **Distribution Statement.** Denote releasability to the public or limitation for reasons other than security for example "Release unlimited". Cite any availability to the public, with address and price.
- 19 & 20. **Security Classification.** Do not submit classified reports to the National Technical Information Service.
21. **Number of Pages.** Insert the total number of pages, including this one and unnumbered pages, but excluding distribution list, if any.
22. **Price.** Insert the price set by the National Technical Information Service or the Government Printing Office, if known.

BIBLIOGRAPHIC DATA SHEET		1. Report No. UIUCDCS-QPR-71-4	2.	3. Recipient's Accession No.	
4. Title and Subtitle Quarterly Progress Report				5. Report Date	
				6.	
7. Author(s)				8. Performing Organization Rept. No.	
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801				10. Project/Task/Work Unit No.	
				11. Contract/Grant No.	
12. Sponsoring Organization Name and Address Department of Computer Science University of Illinois Urbana, Illinois 61801				13. Type of Report & Period Covered Quarterly Progress Report - Oct.-Dec. '71	
				14.	
15. Supplementary Notes					
16. Abstracts An abstract is not applicable.					
17. Key Words and Document Analysis. 17a. Descriptors Not Applicable					
17b. Identifiers/Open-Ended Terms Not Applicable					
17c. COSATI Field/Group					
18. Availability Statement RELEASE UNLIMITED				19. Security Class (This Report) UNCLASSIFIED	
				20. Security Class (This Page) UNCLASSIFIED	
				21. No. of Pages 128	
				22. Price	