AASERT SUMMER RESEARCH TEAM

Dr. Linda Hayden, Principal Investigator

Parallel Processing Team

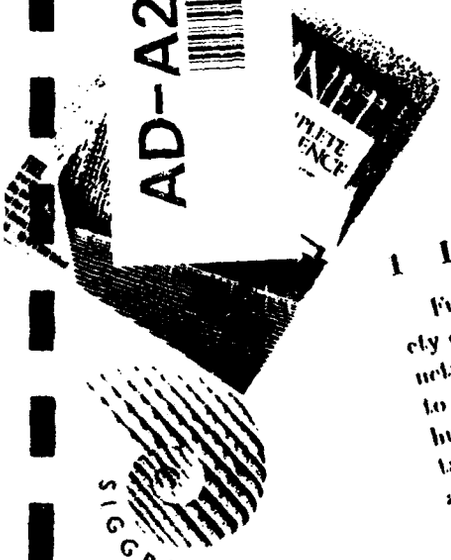
Dr. Johnny Houston, Instructor
Michelle Brown, Assistant Instructor
Ervin Howard, ECSU
Derrek Burrus, High School Researcher
Kuchumbi Hayden, High School Researcher
Connie Sawyer III, High School Researcher

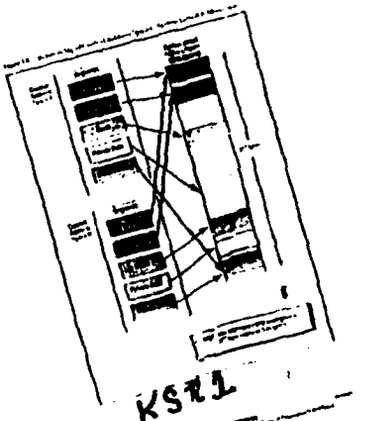
Computer Visualization

Dr. Jingyuan Zhang, Instructor
Stephanie Vaughan, Assistant Instructor
Sharon Saunders, ECSU
Kelvin Trotman, ECSU
Denisa Edwards, ECSU
Jackie Hall, High School Researcher
LaVonna Felton, High School Researcher

Elizabeth City State University

ELIZABETH CITY, NORTH CAROLINA 27909





1 Interactive parallel programs

Future parallel applications will be run on a variety of parallel machines linked with high performance networks. The primary purpose of our research is to exploit these modern systems' capabilities to offer human-interactive interfaces that can execute simultaneously with a parallel application's computation and storage tasks. Specifically, we are exploring the potential for increases in performance and ... ality gained by the on-line interaction with their supercomputer ... on networked parallel ... the scientific ... larger ...

Souvenir of Tour of the United States Capitol
Washington, D.C.
CONDUCTED BY THE CAPITOL GUIDE SERVICE

7-14-94

AVS Network News

ADMI ANNUAL WORKSHOP
July 21-24,1994
SPELMAN COLLEGE
ATLANTA, GEORGIA

Georgia Tech

SEQUENT Accounts & Files

TECHNICAL REPORT OF THE

ECSU HOME - INSTITUTION SUPPORT PROGRAM

1994 FOURTH QUARTER

SUBMITTED TO

THE OFFICE OF NAVAL RESEARCH

BY

ELIZABETH CITY STATE UNIVERSITY

Dr. Linda Hayden, Principal Investigator
Box 672 ECSU
Elizabeth City, NC 27909
(919) 335-3617 FAX: 919-335-7408
email: LHAYDEN@UMFORT.ECSU.EDU

PAST FUNDING INFORMATION:
FUNDING PERIOD: 1/1/91 TO 9/30/93
GRANT # N00014-91-J-1308
R&T CODE: 4331800---02
S. O. CODE: 1133
DISBURSING CODE: N00179
AGO CODE: N66005
CAGE CODE: OJLKO

ECSU-AASERT Summer Research Project in Parallel Processing and Computer Visualization

Dr. Linda Hayden, Principal Investigator

Summer 1994

This ECSU-AASERT research project will annually support minority undergraduates who are not now a part of our program. It will also allow us to included female and minority precollege students in our summer research training. All students hired under this research project will be actively involved in **computer science** research projects. The subareas of their research investigations are **parallel processing** and **computer graphics/visualization. Each will also be assigned a computer networking problem to investigate.**

Undergraduate Computer Science majors must be full time ECSU students with a minimum 2.8 overall GPA, 3.0 GPA in their major courses and must be recommended by two of their major professors. The undergraduates will work in the laboratory for 5 hours each day, 5 days each week for 6 weeks.

Precollege students selected must have completed a minimum of three credits of mathematics including geometry and algebra II. Grades of B or better in these courses plus recommendation of two science/mathematics teachers will be required. The precollege students will work in the laboratory for five weeks, 5 hours each day, 5 days each week. All students, both precollege and undergraduate must be citizens of the United States.

The **Instructor** for each team will be a member of the ECSU faculty/staff who is knowledgeable in the subdiscipline. Instructors will work with the students for approximately 3 hours each day, 4 days each week.

Consultants will be available to team members daily via email and may make one visit to the ECSU site.

Assistant Instructors will be graduate students pursing a Masters or Ph.D in Computer Science. Assistant Instructors will work with the students 5 hours each day.

Planned Activities

Trip to the Library of Congress (Washington, DC July 14-17 tenative date)
Trip to the NC Supercomputing Center (Research Triangle Park, NC 2-3 days)
SIGGRAPH Conference (Orlando, Fla. July 25-27, 1994 for CV team)
ADMI Conference & Georgia Tech. Atlanta, GA (July 21-23 for PP team)
Lectures by visiting consultants(Sharon Ramsey of Alcoa Aluminum)
Final Research Project Reports

Schedule

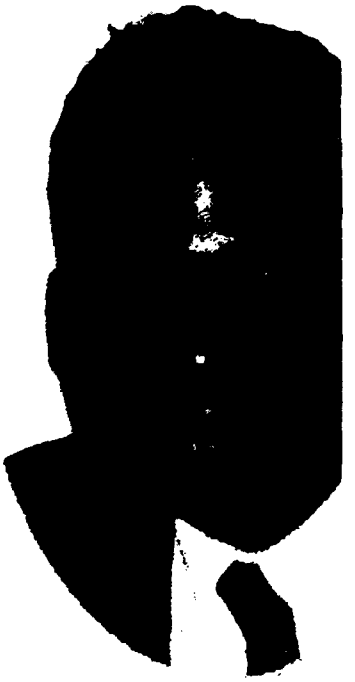
Week 1 ECSU students work with Instructors
Weeks 2-6 High School Students join the teams

Connie Sawyer, Junior
Camdem County High (PP)

Denisa Edwards, Senior
ECSU (CV)

Derrek Burrus, Senior
Camden County High
(PP)

Kevin Trotman, Senior
ECSU (CV)



Ervin Howard, Senior
ECSU (PP)

Highlights from the '94 AASERT Research Training Project in Parallel Processing and Computer Visualizaton. Sponsored by the Office of Naval Research and Elizabeth City State Univeristy.

Highlights from the '94 AASERT Research Training Program in Parallel Processing and Computer Visualization. Sponsored by the Office of Naval Research and Elizabeth City State University

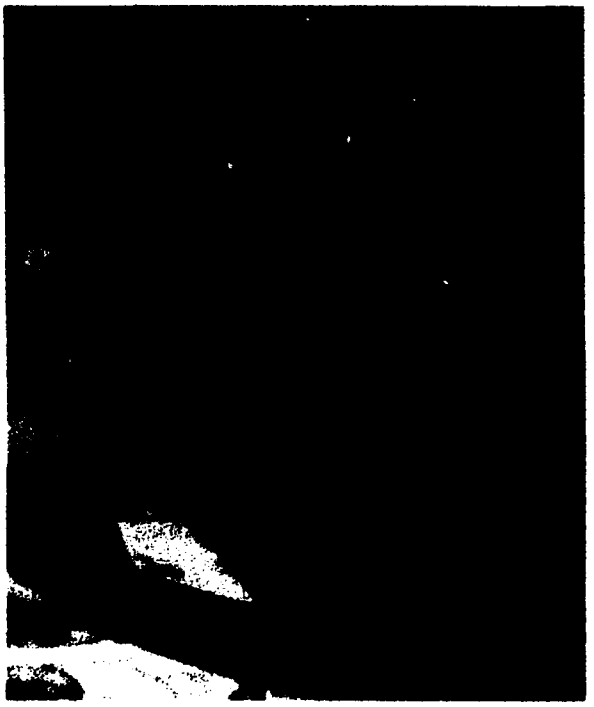
Sharon Saunders, Senior
ECSU (CV)

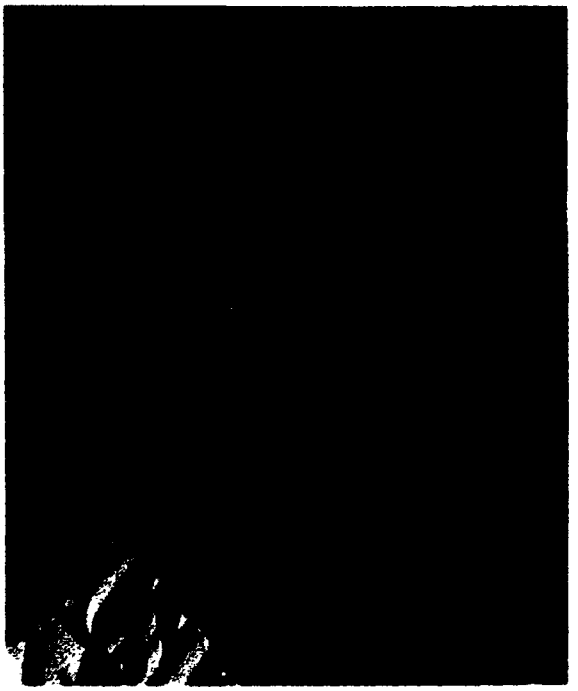
LaVonna Felton, Recent Graduate
I.C. Norcom High School (CV)

Kuchumbi Hayden, Senior
I.C. Norcom High School (PP)

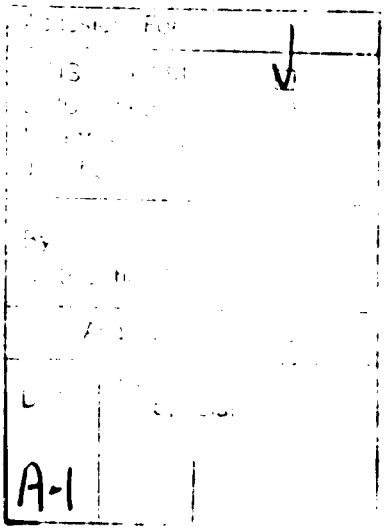
Jackie Hall, Recent Graduate NC
School of Math and Science (CV)

Highlights from the '94 AASERT Research Training Project in PP and CV sponsored by the Office of Naval Research and Elizabeth City State University.





Michelle Brown Emmanual(left) and Stephanie Vaughan (right) Talk with PP and CV team members in the Blue Room during lunch. Both are graudate students in Computer Science.



Highlights from the '94 AASERT Research Training Project in PP and CV Sponsored by the Office of Navi Research and Elizabeth City State University

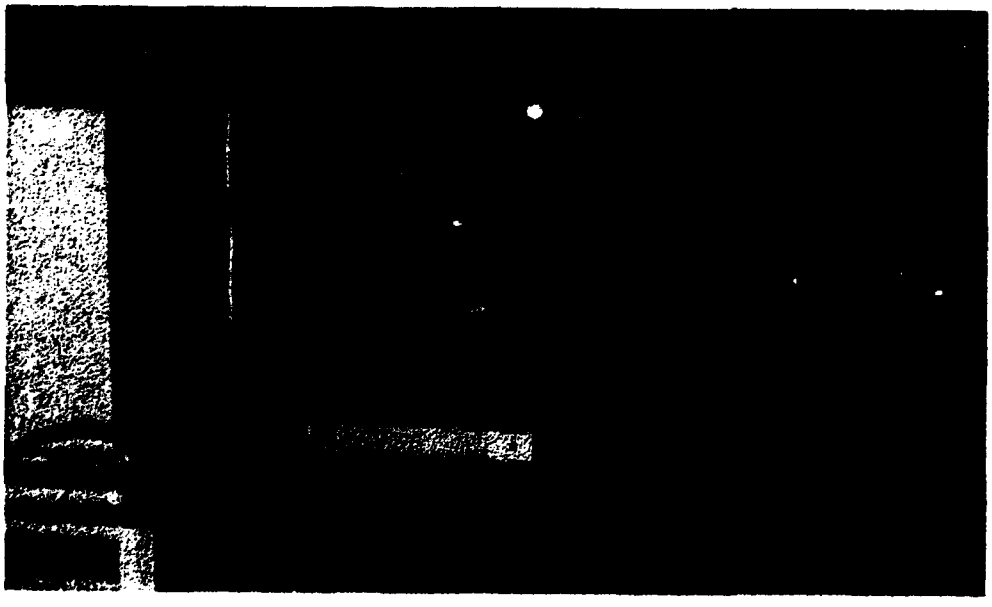
Dr. Linda Hayden and
Michelle Brown-Emmanual

Kevin trotman, Dr. Jingyuan Zhang, Denisa Edwards and Sharon Saunders attend the AVS Training Workshop held at the North Carolina Supercomputing Center (NCSC)

Sharon Saunders and Mr
Coleman also attended the

NORTH CAROLINA SUPERCOMPUTING CENTER (NCSC) TRAINING ROOM.

NCSC TRAINING STAFF



DR. HAYDEN AT A NCSC TRAINING ROOM WORKSTATION

Parallel Processing team

Derrek & Connie on BeBop with Coo on the background

Dr. Johnny Houston, Instructor (center) with Michelle Brown Emmanual, HU Graduate Student (back left)

Team Members: Ervin Howard (left), Connie Sawyer (missing from photo), Kuchumbi Hayden (right front) and Derrek Burrus (right rear)

Parallel Processing Research Project Description

Reference : Laboratories for Parallel Computing by Christopher H. Nevison, Jones and Bartlett Publishers, 1994 ISBN 0-86720-470-2

The sieve of Eratosthenes has long been a standard benchmark program for integer operations on a sequential computer. We will develop a parallel prime number sieve to demonstrate several concepts fundamental to parallel computing. This example also illustrates a process of parallel program development which can be usefully applied to many problems.

After defining the problem and a sequential solution, we will begin the development of a parallel algorithm by analyzing the actions which must be taken and the order constraints on those actions. This makes it possible to define a maximally parallel, although impractical, algorithm. We will then develop a practical algorithm which can be mapped to a network of message passing processors, a pipeline.

The mapping of the algorithm to the array of processors brings up the issue of load balancing. We will develop an algorithm for static load-balancing, allocating the work so that each processor will have about the same amount to do.

We will investigate the performance of the algorithms by measuring the speedup and efficiency. Amdahl's Law gives theoretical limits on the speedup which can be obtained from parallel computers. This will lead to a final refinement of the algorithm based on using an efficient sequential algorithm within processors while maintaining the pipeline between processors.

The final aspect of algorithm development will be an analysis of communication issues: 1) Buffering between the processors reduces processor idle time spent waiting for communication with a neighboring processor; and 2) The effect of packing the integer messages into larger messages between processors so as to increase overlapping of communication with computation.

Parallel Processing Team

Instructor:	Dr. Johnny Houston
Assistant Instructor:	Michelle Brown-Emmanual
Consultant:	Dr. Nan Schuller/Christopher Nevison
ECSU Student 1:	Ervin Howard
High School Student 1:	Connie Sawyer
High School Student 2:	Kuchumbi Hayden
High School Student 3:	Derrek Burrus

KENDALL SQUARE RESEARCH, TECHNICAL SUMMARY, 170 Tracer Lane Waltham Massachusetts 02154, 1992.

The KSR-1 is different than other systems before it, and it also has improved on certain qualities of those systems. It has scalabilty and low cost of highly parallel processing, along with high performance and easy use.

There is a demand for higher computer performance in society today, as well as a need for society to keep up with the vast data generated by electronic devices. There have been many approaches to improving computers, including the 4GL's, mainframes, supercomputers, and MPP's. All of these tried to improve on the progress of the preceding system, but these all only dealt with one area of work, such as science and engineering, and database intensive processing applications.

Modern computing has four servers to go through to produce what it is purposed for: compute, data/storage, access/communication, and visualization. The KSR-1 can function as any one, combination, or all of these and does these simultaneously. This is basically because of ALLCACHE, which enables different processors to function at the same time. The KSR-1 also provides production-level parallel computing power that is the contemporary industry standard.

The KSR-1 is ideal for numerically intensive processing because of its software environment's ability to address the requirements of multiprocessor parallel processing. The numerous enhancements by other people have also improved production by the software.

There are many different languages in parallel processing, with the most popular being Fortran. Fortran lets programmers develop and port code from other systems quickly and easily. It has many convenient features, such as strength reduction and reduction scheduling, as well as extensions like Additional Data Types (INTEGER*1 and LOGICAL*1). Other languages include C and C++.

This article was very interesting to me. It made me understand a little more about what we would be doing this summer. I can say that now I do have an idea about what parallel processing is all about.

Reviewed by: Connie Sawyer

From Molecules to Merchandise: THE EVOLUTION OF PARALLEL PROCESSING

BY: DR. GEORGE O. GARDNER III
ECSU Parallel Processing Team

A computing technology initially embraced by the scientific world is beginning to climb the corporate ladder. Prudential Securities in New York; Allstate in Northbrook, Ill.; Hallmark Cards in Kansas City, Mo.; and Kmart in Troy, Mich. All have the technology's potential to revolutionize commercial processing.

At Prudential Securities in New York, the parallel processors can do in 30 seconds what takes five to eleven minutes to process data on a 32-bit serial computer.

At Allstate, an IBM parallel processor will improve system availability and customer service.

In Kansas City, Mo. the Hallmark Cards Co. now have point-of-sale cash registers in 250 card-specialty stores, plus mass-channel locations gathering data, and this provides them a representative sample of there sales.

Kmart uses parallel processing for similar reasons as Hallmark Cards.

There are four basic ways to configure processors in a parallel system. The four ways in which the CPUs can be configured are on a bus, in a mesh, as hypercubes or in switched networks.

One is a bus, which is comparable to a straight line with several processors attached to it. This configuration is not scalable and can become congested at more processors are added, but bus-based systems typically have the lowest price of the four configurations.

In a mesh arrangement, a processor resides at each "intersection". Also called a node. This interconnection arrangement is suitable for solving certain kinds of technical problems, but is expensive to implement as the number of nodes grows.

The three-dimensional hypercube array uses processors that are numbered successively according to the binary system, with each processor connected to its neighbor. The advantage is in the small number of connections needed to communicate from one node to another.

In switched networks, processors are connected through complex sets of switches. Switches are expensive to implement, however, and for many applications they are slow to make connections compare to fixed interconnection networks with routing.

Companies reaping the most from parallel computers have strong internal capabilities and good vendor ties.

This article was well written and easy to read. It covered the fundamentals of parallel processing to a "t". So that a beginner could comprehend the authors thoughts. I would recommend this article to others to read. I congratulate the author on a well written story.

Reviewed by: Kuchumbi Hayden; Parallel Processing
ECSU Parallel Processing Team

A. Carle, K. Kenennedy, C. Koelbel, J. Millor-Crummey, S. Warren
Overwiew of the Fortran Parallel Programming Tools Project
Rice University

Summary:

The Fortran Parallel Programming Tools Project is a three year program. The program is assigned to design and construct a suite of research prototype software tools. The tools known as the D system, will assist in Fortran D, which is an abstract language for expressing parallel programs. The tools will help support development of these parallel programs. All of this is a part a major system, not stated in the article, known as the Fortran language which is simply an example of a high-level language. Its name is an acronym for FORmula TRANslation.

The development of a program written in the Fortran D language can be very difficult and challenging, but its success will assist scientist in the development of high-performance, machine-independent programs for parallel machines. The research group of the D system hope to serve as a model for future efforts by vendors.

The D research will focus on analyzing programs that will support and give detailed information on how data can exploited by compiler implementation.

There are three tools the D tools efforts will focus on, which are an intelligent editor, a source-level debugger, and a performance analyzer. The editor is responsible for the union of the wide range of program analysis technology. The D debugger will support the radically reconstructed debugging code. The performance analyzer will collect the information and present it in the original single-threaded program source.

This article gives a very explicit explanation of the overall project however by not giving a solid definition of what Fortran is itself it causes the reader to resort to other resources to get the whole picture.

This article was reviewed by Derrek Burrus.

KSR1 Training At Research Triangle (Supercomputer Center)

By Ervin M. Howard

Our trip to the Research Triangle located in Durham, NC involved attending a KSR1 training program that helped us in preparation for the AASERT Summer Research Program of Elizabeth City State University for high school students. Dr. Johnny Houston, Senior Research Professor and Instructor for the AASERT Parallel Processing Team, Kevin Trotman, a Computer Science major, Dr. Jinyguan Zheng, a Professor of Computer Science, and Michelle Brown-Emmanuel, a graduate assistant to the Parallel processing team were also participants to this training program. We resided in the Red Roof Inn on the date of June 26, 1994. We left the hotel for our first training class at 1:00 p.m. where Mr. Lee Bartolotti, an employee of the MCNC Supercomputer Center taught the first phase of the program which was the Introduction to Parallel Algorithms and Programming on June 27,1994. The next two training classes were taught by Mr. Eric Sills who is also an employee of the facility. These classes are KSR1 Architecture and Programming Model I: (Compliers, KAP,PRESTO-) After these classes, we and other participants from other universities learned several commands using UNIX which enabling all of us to access the KSR1 system to use these parallel processing techniques.

Our next class after lunch involved the Programming Model II.(Regions, sections, tiles, spc-) was also taught by Mr. Eric Sills. The last training class was Programming Model III: (Pthreads) was taught by Mr. Greg Byrd. After this class, we all practiced writing certain commands that will compile written programs in Fortran using the KSR1 system to run serial and parallel processing. After this last class, we pressed onward to ECSU to assist the students in program from our training we received.

I have really enjoyed my trip and experience of learning this new system to compile these programs using serial and parallel processing. I hope I can have another opportunity to participate in another training program that will increase my awareness of parallel processing.

July 1 Report: Parallel Processing Activities

Parallel Processing Team Participants consists of :

1. Dr. Johnny L. Houston: Team Instructor/Advisor
2. Michelle Brown-Emmanuel : Graduate Assistant
3. Ervin M. Howard: ECSU student
4. Kuchumbi Hayden: High school student
5. Derrek Burrus: High school student
6. Connie Sawyer: High school student

- The students were instructed to understand certain details on Parallaxis which will enable them to program in parallel programming. Here detailed information regarding Parallaxis.

What is Parallaxis ?

Parallaxis is a procedural programming language for implementing massively parallel algorithms. This paper concentrates on showing how to translate a massively parallel algorithm into a Parallaxis program.

The two paramount ideas in Parallaxis are:

1. Let the programmer handle parallelism explicitly.
2. For each parallel algorithm, have the hardware configuration specified as well.

What we would like to express with these statements is, that each parallel algorithm depends to a very high degree on the parallel computer structure that is bound to execute it. That is, a parallel algorithm for fluid dynamics simulation might require a two-dimensional grid of processors with a four-neighbor connection. A different parallel algorithm, let's say for generating fractal curves, may need a binary tree topology to perform well. From our point of view, each program should not only define the algorithm (the software part) in some syntax, but also specify the processor configuration and the connections (the hardware part).

With Parallaxis, we only consider SIMD (single instruction, multiple data) machines, which more or less correspond to massive parallelism, since the simpler structure, as compared to the more general MIMD (multiple instruction, multiple data) class of computers, allows a higher degree of system integration, so more processors cooperate in one machine. Prominent examplels are the Connection Machine CM-2 [1] and the MasPar MP-1 [2]. In SIMD systems, all processor elements (PEs) are alike, consisting of an ALU (arithmetic logic unit) and local memory, so the definition of the hardware structure reduces to the number of processors requires and the (regular) connection pattern needed for data exchange. All PEs are controlled by a single global sequencer that executes the instruction cycle, so at every point in time, all PEs execute the same instruction or remain idle. This severe SIMD-restriction is not always a disadvantage, for there is a large number of applications exhibiting "data parallelism" (natural parallelism) that translates nicely into SIMD. Futhermore, SIMD programming is much simpler, for there is only a single thread of control and no problems with synchronization of processes occur. The language Parallaxis is based on the sequential procedural language Modula-2[3]. However, additional language constructs to express parallelism have been included;

Concerning the network topology

- number of processors needed
  This is similar to an array declaration in Pascal.
- number and direction of links between processors
  Each link is defined by a functional expression.

Concerning the algorithm

- distinguishing variable declarations for scalars and vectors
- a parallel block encloses statements dealing with vector data
- explicit processor selection for a statement sequence (all others remain inactive)
- vector to scalar reduction with any binary function

The new approach taken in Parallaxis is the declaration of the PE structure and its interconnection network. The physical structure of the parallel computer system (number of PEs and network topology) is hidden by a layer of abstraction; the application programmer uses an unlimited resource of virtual processors and virtual connections between them. The

mapping between the arbitary number of virtual processors with their virtual connections on one side, and the limited number of physically available processors with their communications links on the other side, is being performed either by the operating system (as in case of the Connection Machine, which provides virtual PEs) or by a complier (as in case of the MasPar, which does not offer virtual PEs). This procedure is transparent to the Parallaxis programmer and may be recognized only by considering execution time. Therefore, it is possible to do all testing and debugging of Parallaxis programs on an inexpensivel single processor system, before starting the real application on a parallel computer system.

- Dr. Linda Hayden taught the program logic of how the PeopleWave program operates through pictoral examples.

- The students were taught by Dr. Johnny Houston on the concept of parallel processing using Fortran . (The students were introduced to one of the programming language, Fortran which was used in parallel processing.

- The students were taught basic commands in UNIX to manipulate the VI text editor and email messages to accounts and from other sources.

- The students were to complete a written and oral report on their research material use word processor packages such as Microsoft Works, StudentPageMaker and WordPerfect.

Parallel Processing Visiting Lecture Report

Parallel Processing Team Participants consists of :

1. Dr. Johnny L. Houston: Team Instructor/Advisor
2. Michelle Brown-Emmanuel : Graduate Assistant
3. Ervin M. Howard: ECSU student
4. Kuchumbi Hayden: High school student
5. Derrek Burrus: High school student
6. Connie Sawyer: High school student

Parallel Procesing: Lecture Series

- Dr. K.C Wong from Fayetteville State University presented a lecture in Parallel Processing using the KRS1 system. Dr. Wong was scheduled to spend two days in Elizabeth City. His lecture included the following topics on July 6, 1994 was the first phase of Parallel Processing: KSR1 system

  - Using parallel programming techniques for real-world problems.

  - Three KRS1 Parallel Programming Models
    Automatic Semiautomatic Parallelization
    PRESTO Interface
    Pthread Model

- Four types of objects needed in programming language.
  pthreads
  mutex
  conditional variable
  barrier

  - Using Fortran as the ideal programming language to demonstrate parallelism.

  - Dr. Wong explained how the Basic program: The Sieve

of Erastothenes is performed.

- Dr. Wong continued his lecture in Parallel Processing on July 7, 1994 which consist of :
    - How to create pthread objects
    - How to create a mutex
    - How to create a control variable
    - How to create a barrier
- Dr. Wong also develop an algorithm to write the Sieve of Erastothenes in Fortran using pthreads.
- The Sieve of Eratosthenes method of finding prime numbers is written in the Fortran language.

DR. WONG KR31 PARALLEL PROCESSING LECTURE SERIES

Parallel Programming Models (KRS1)
1. Automatic Paralelization (learners use it)
    * KAP in Fortran only
2. PRESTO Interface
    * In Fortran or C
3. Pthread Model
    * In Fortran or C

(all these models differ in degrees of involvements of programmer)

Objects Needed In Programing Language

* Pthreads-a single sequential flow of controls within a process.
* Mutex (Mutual Exclusion)-enabling multiple pthreads in task synchronize their access the signals of codes.
* Condition Variable-enabling a pthread that is the owner of a mutex to unlock and block the mutex.
* Barrier-to coordinate the work of a set of threads (synchronize pthreads)

How To Create Pthread Objects In General Steps are need
1. Define a variable to store the address of the object. (pointer variable in c, integer variable in fortran)
2. Invoke approximate routines

Parallel Programming Models (KRS1)

- Automic Parallelization
- PRESTO Interface
- Pthread Model

Object Needed In Programming Language

- Pthreads
- Mutex
- Condition Variable
- Barrier

How To Create Pthread Objects In General

- Define A Variable
- Invoke Approximate Routines

Operations We Can Perform

- On Pthreads
- On Mutex
- On Condition Variable
- On Barries

Operations We Can Perform Upon These Objects

1. On Pthreads
    - Call PTHREAD_CREATE
    - Call PTHREAD_CREATE
    - Call PTHREAD_CREATE
2. On Mutex
    - Call PTHREAD_MUTEX_INIT
    - Call PTHREAD_MUTEX_LOCK(MUTI,STATUS)
    - Call PTHREAD_MUTEX_UNLOCK(MUTI,STATUS)
3. On Conditional Variable
    - Call PTHREAD_COND_INIT
    - Call PTHREAD_COND_WAIT(COND1,MUTI,STATUS)
    - Call PTHREAD_COND_SIGNAL(COND1,STATUS)
4. On Barriers
    - Call PTHREAD_BARRIER_INIT
    - Call PTHREAD_BARRIER_CHECKIN(IBOR1,ISEQUENCE_NO,ISTATUS)
    - Call PTHREAD_BARRIER_CHECKOUT(IBOR1,ISEQUENCE_NO,ISTATUS)

```
      PROGRAM ERATOSTHENES
C **** PROGRAMMERS: ERVIN M. HOWARD & MICHELLE BROWN-EMMANUEL
C **** INSTRUCTOR: DR. JOHNNY L. HOUSTON
C **** PARALLEL PROCESSING TEAM (AASERT SUMMER RESEARCH PROGRAM
C **** THIS PROGRAM IS DEVELOP FOR THE SOLE PURPOSE OF TESTING
C **** IN THE FORTRAN LANGUAGE TO COMPLETE THE LAST INTIALIZED PHASE
     **** IN USING THIS PROGRAM FOR KRSI PARALLEL PROCESSING


C **** THIS PROGRAM WILL FIND A SERIES OF PRIMES NUMBERS FROM
C **** 1000 USING THE SIEVE OF ERATOSTHENES


C ********** LIST OF VARIABLES AND THEIR DUTIES *********
C  K: THIS VARIABLE WILL HELP PRINT THE RESULTS BY SERVING AS A COUNTER
C  I: LOOP CONTROL VARIABLE THAT CONTROL THE LOOP FOR PN(I) TABLE
C  P: LOOP CONTROL VARIABLE THAT CONTROL THE LOOP FOR PN(P) WHILE
C     PLACING THE PRIME NUMBER IN PN(P)
C  C: LOOP CONTROL VARIABLE THAT WILL PRINT THE PRIME NUMBERS FROM THE TABLE
C  N(): N TABLE WILL REPRESENT THE SIEVE OF ERATOSTHENES PROCESS OF
C  ELIMINATING THE MULTIPLES.
C  ************************************************************************


      INTEGER N(1000), PN(200)
      INTEGER K,I,P,C

      OPEN ( UNIT = 3, FILE = 'A:E.DAT')

C     BEGIN: INTIALIZING PROCESS
      DO 15, I = 2, 1000, 1
      N(I) = 0
   15 CONTINUE
      K = 0

C ********** THIS PART WILL TEST TO SEE IF THERE IS A EMPTY STORAGE
C ********** PLACE IN THE TABLE AND IF THERE IS THEN THE PRIME NUMBER
C ********** WILL PLACE IN THERE.


C     BEGIN: CAPTURING THE PRIME NUMBERS

      DO 25, P = 2, 1000
      IF (N(P) .LT. 0) GOTO 25
        K = K+1
        PN(K) = P
      IF (P .GT. SQRT(1000)) GOTO 25

C     BEGIN: CROSSING OUT FACTORS USING THE SIEVE OF ERASTOTHENES
C        ,   BY PLACING -1 IN THE TABLE

      DO 18, I = P, 1000, P
       N(I) = -1
   18 CONTINUE
   25 CONTINUE

C     BEGIN: PRINTING THE PRIME NUMBERS FROM THE TABLE

      C = 1
      WRITE(3,69)
      DO 29, I = 1,K

      WRITE(3,70) PN(I)
      C = C+1
      IF (C .LE. 7) GOTO 29
      WRITE(3,71)
      C = 1
   29 CONTINUE

C     END: ENDING CAPTURING PRIMES
C     END: ENDING PRINTING PRIMES

   69 FORMAT(1X, 'SIEVE OF ERATOSTHENES PRIME NUMBERS')
   70 FORMAT(5X,I4)
   71 FORMAT(1X)
      STOP
      END
```

```
SIEVE OF
       2
       3
       5
       7
      11
      13
      17
      19
      23
      29
      31
      37
      41
      43
      47
      53
      59
      61
      67
      71
      73
      79
      83
      89
      97
     101
     103
     107
     109
     113
     127
     131
     137
     139
     149
     151
     157
     163
     167
     173
     179
     181
     191
     193
     197
     199
     211
     223
     227
     229
     233
     239
     241
     251
     257
     263
     269
     271
     277
     281
     283
     293
     307
     311
     313
     317
     331
     337
     347
     349
     353
     359
     367
     373
     379
     383
     389
     397
     401
     409
     419
     421
     431
     433
     439
     443
     449
     457
     461
     463
     467
     479
     487
     491
     499
     503
     509
     521
     523
     541
     547
     557
     563
     569
     571
     577
     587
     593
     599
     601
     607
     613
     617
     619
     631
     641
     643
     647
     653
     659
     661
     673
     677
     683
     691
     701
     709
     719
     727
     733
     739
     743
     751
     757
     761
     769
     773
     787
     797
     809
     811
     821
     823
     827
     829
     839
     853
     857
     859
     863
     877
     881
     883
     887
     907
     911
     919
     929
     937
     941
```

```
System PeopleWave;

CONST SIZE                    = 4;
    NUMBEROFNEIGHBORS = 8;
TYPE          Matrix =   ARRAY[1..SIZE],[1..SIZE] OF INTEGER;

CONFIGURATION   grid[1..SIZE],[1..SIZE];


CONNECTION
          dir(0):grid[i,j]   <->    grid[i-1,j].dir(4);
          dir(1):grid[i,j]   <->    grid[i-1, j+1].dir(5);
          dir(2):grid[i,j]   <->    grid[i,j+1].dir(6);
          dir(3):grid[i,j]   <->    grid[i+1,j+1].dir(7);

SCALAR   ScalarWave: Matrix;
        i,k: INTEGER;
VECTOR  WaveElement:INTEGER; Averager:INTEGER;
         AllNeighbors:INTEGER;OneNeighbor:INTEGER;


PROCEDURE  ShowWave(SCALAR Wave: Matrix);
SCALAR
        i,j:INTEGER;

BEGIN
      FOR i:=1 TO SIZE DO
            FOR j:=1 TO SIZE DO
                writeInt(Wave[i,j],1);
            END;
       WriteLn;
      END;
        WriteLn;
        WriteLn;
        WriteLn;
END ShowWave;

BEGIN (*PeopleWave*)

     PARALLEL

            (* Intitialize the wave *)
          IF dim2= SIZE THEN
```



```
                WaveElement:= 1
        ELSE
                WaveElement:= 0
        END;
    (* Divisor for determining new WaveElement is normally 3
                but only 2 on edges *)

        IF (dim1= 1)  OR (dim1= SIZE; OR
                (dim2= 1) OR (dim2= SIZE) THEN
            Averager:= 2
        ELSE
                Averager:= 3
        END;
ENDPARALLEL;

 STORE (WaveElement, ScalarWave);


  ShowWave(ScalarWave);
    FOR i:= 1 TO SIZE DO
     PARALLEL
       (*retrieve and average (weighted) info about
                neighbors*)
      AllNeighbors:= WaveElement;
   FOR k:= 0 TO NUMBEROFNEIGHBORS-1 DO

     Oneneighbor:=0;

     RECEIVE grid.dir(k) (OneNeighbor)
     FROM   grid.dir((k+4) MOD 8) (WaveElement);

    IF k < 5 THEN
      AllNeighbors:= AllNeighbors + OneNeighbor;
   END;

   END;

     WaveElement:= AllNeighbors DIV Averager;

    IF  WaveElement >=1 THEN
     WaveElement:=1
   ELSE
     WaveElement:=0
```



```
END.
ENDPARALLEL;

 STORE (WaveElement, ScalarWave
 ShowWave(ScalarWave)
 ENDI'FOR*)
END PeopleWave
```

```
(*Michelle Brown-Emmanuel*)
(*AASERT Summer Research Project*)
(*Summer 1994*)
(*This is the output for the PeopleWave with the 1's ini
tialized in the first row and the last column*)
```

```
1111111111  1001100000  1011000000  1010000000
0000100000  1001100000  1011000000  1010000000
0000100000  0001100000  0011000000  0010000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
0000100000  0001100000  0011000000  0110000000
```

```
1010000000  1010000000  1000000000
1010000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
1100000000  1000000000  1000000000
```

```
1000000000
1000000000  1000000000
1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
1000000000  1000000000  1000000000  1000000000
```



```
(*Michelle Brown Emmanuel*)
(*AASERT Summer Research Project*)
(*Summer 1994*)
(*This is the output for the PeopleWave with the 1's ini
tialized in the last two columns*)
```

```
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
(0000000011  0000000110  0000001100  0000011000
```

```
(0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
0001100000  0001100000  0001100000
```

```
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
01100000000  11000000000  10000000000  10000000000
```

Connie Sawyer
Articles

1. BEYOND COMPUTING

2. Parallel & Distributed Technology Journal
NEW JOURNAL

3. Parallel Computing
v15(1990) -- present

4. Parallel Computing News
v3 n3(Mar 1990) -- present

5. Banerjee, U., *Dependence Analysis for Supercomputing*, Kluwer Academic Publishers, Boston, Mass., 1988b.

6. Callahan, D., Cocke, J., Kennedy, K., "Compiling Programs for Distributed-Memory Multiprocessors," *Journal of Supercomputers*, October 1988

7. Lewis, T.G., and H. El-Rewini. 1992. *Introduction to Parallel Computing*. Prentice-Hall, Engelwood Cliffs, NJ.

8. Osterhaug, A. 1986. *Guide to Parallel Programming on Sequent Computer Systems*. Sequent Computer Systems, Beaverton, OR.

9. Quinn, M.J. 1985. A note on two parallel algorithms to solve the stable marriage problem. *BIT*, vol. 25, pp. 473-476.

10. Reed, D.A., and M.L. Patrick. 1985. Parallel, iterative solution of sparse linear systems: Models and architectures. *Parallel Computing*, vol. 2, pp. 45-67.

11. Wong, C.K., and S.-K. Chang. 1974. Parallel generation of binary search trees. *IEEE Transactions on Computers*, vol. C-23, no. 3, Mar., pp. 268-271.

12. [Engelhardt 91] Stefan Engelhardt
Automatische Übersetzung einer massiv parallelen Programmiersprache fur sequenielle und parallele Rechnerarchitekturen Diplomarbeit Nr. 791, Universitat Stittgart, June 1991

13. [Barth Braunl Sembach 90] Barth Braunl Sembacn
Parallaxis User Manual
Computer Science Report, no. 3/90,
Universitat Stuttgart, March 1990

14. [Braunl 91a] Thomas Braunl
Parallaxis Modula massiv parallel
c't MAgazin fur Computertechnik, Heft 6,
June 1991, pp. 34-35 (5)

3

Parallel Programming Research Team Weekly Report
Progress Report #2

by Michelle Brown Emmanuel

July 4 - July 8, 1994

This week the precollege students have been very busy working on several tasks. They have been working on their group activities which are due today. They have learned how to save files and formatting diskettes on the Suns.

The precollege students have also been busy working on the Parallelaxis Code for Peops. Their Assignment. They have typed in the code and compiled it as it was given to them. Now they are working on modifying this code. The were all given a different size matrix to work with. K. Hayden was given a 6 X 6, C. Sawyer was given a 7 X 7, and D. Burrus was given a 5 X 5. I have completed modifying the code for a 10 X 10. I have completed the initialization for the ones to first appear on the diagonal, in the first column and in the last two columns. In the report I have included a copy of my program with the 1's initialized in the fifth column & first row and with the 1's initialized in the last two columns. They students have not really have a lot of time to work on their problem because we have been busy working with Dr. Wong.

Dr. Wong was our visiting lecturer from Fayetteville State University on Wednesday and Thursday. We really enjoyed the lecture that was given by Dr. Wong. He was very informative with his lecture.

On day 1 of Dr. Wong's lecture he discussed the three different programming models using the KSR1. These three models where KAP, PRESTO, and Pthreads. These three models differ in degrees of involvement with the programmer. KAP is a preprocessor that is only available in FORTRAN. With KAP the programmer is not responsible for the correctness. The processor makes the necessary corrections. This type of preprocessor is not recommended for REAL programmers. This type of processor would be good for engineers. Then he briefly talked about PRESTO. PRESTO is available in FORTRAN and C. With PRESTO the programmer is responsible for some of the correctness. Dr. Wong did not really talk a lot about the first two models. His focus was mainly on Pthreads. Pthread processors are available in FORTRAN and C. With the Pthread model the programmer is responsible for all correctness. Dr. Wong recommends this model for programmers. The Pthread model involves four objects. Pthread is the first object. A Pthread is a single sequential flow of controls within a process. The next object is Mutex. Mutexes(Mutual Exclusion) enables multiple Pthreads in task to serialize their access to segments of code. The third object that was discussed was Condition Variable. The Condition Variable enables a Pthread that is the owner of a mutex to unlock the mutex and lock it. The last object that was discussed was a Barrier. A barrier is used to coordinate work of a set of Pthreads. Barrier synchronize Pthreads at designated points. This concluded our discussion with Dr. Wong for Wednesday, July 6, 1994.

Dr. Wong began his discussion on Thursday, July 7, 1994 talking about how to create Pthread objects. He stated that there are two major steps involved in creating these objects. These two steps are (1) Define a variable to store the address of the object(ptr variable in c, integer variable in FORTRAN) and (2) Invoke appropriate subroutine. After discussing the two steps involved in creating an object Dr. Wong showed us how to create these objects.

To create a Pthread

Computer Graphics/Visualization Team

" Is the Network Up?..... Can I get to Sun1? "

Dr. Jingyuan Zhang, Instructor

Stephanie Vaughan, Graduate Student

Team Members: Denisa Edwards
Sharon Saunders
Kevin Trotman
LaVonna Felton
Jackie Hall

Computer Graphics/Visualization Project Description
3-D Modeling and Viewing

The computer visualization project, which students funded under AASERT will investigate, shall consist of three stages. In the first stage, the student researchers will be given lectures concerning solid modeling and visualization. For the solid modeling, they will learn how to represent a solid object using a edge-based boundary model. They will also be taught how to obtain a new object from an existing one or from scratch using Euler operators. As to visualization, they will be given the concepts of 3D viewing, shading and texturing.

In the second stage, the students will define data structures for a solid object using the edge-based boundary model and define a subroutine for each Euler operator. Then we will use the Euler operators to build a set of primitives such as cube, sphere, cylinder, cone and torus. We will also use the Euler operators to build high level operators like sweeping (including both translational sweep and rotational sweep), gluing and assembling. After this stage, students have a simplified solid modeling system based on Euler operators.

In the third stage, students will develop the software for the Gouraud shading and Phong shading as well as for the solid texturing. Finally, they will design objects using the solid modeling system build in the second stage and visualize these objects using the software developed in this stage.

Visualization/Graphics Team

Instructor:	Dr. Jingyuan Zhang
Assistant Instructor:	Stephanie Vaughan
Consultant:	Dr. Scott Owens
ECSU Student 1:	Sharon Saunders
ECSU Student 2:	Kevin Trotman
ECSU Student 3:	Denisa Edwards
High School Student 1:	LaVonna Felton
High School Student 2:	Jackie Hall

RENDERMAN

RenderMan uses various geometric shapes, such as spheres, cylinders, and paraboloids in order to obtain various objects. These shapes have basic parameters needed. Radius which is the distance from the center of the figure, height is the length of the figure, thetamax which is the distance around the figure, zmax and zmin covers the height of the figure from the top and bottom. (Upstill pp.12-13)

In order to move an object created in the rib.file, in renderman, the transformation function is used in the coordinate system. Transformation consist of three parts: translate, rotate, and scale. Translate concatenates a translation onto the current transformation. Translate changes the location of an object. Rotate turns the object around an axis. And scale changes the size of an object in the x, y, and z directions. (Upstill pp. 111-113)

The member of the computer visualization team demonstrated several steps in understanding the quadric surfaces and changing the transformations of a rib.file in RenderMan. The steps are on the next page.

WHAT IS RENDERMAN

RenderMan is first of all a scene description methodology: a comprehensive way to describe objects, scenes, lights and cameras so that a computer can create images from them. Renderman bring various surfaces to synthetic imagery. The key is flexibility in shading the surfaces in the scene. The key idea behind RenderMan is the separation between the modeling and the rendering domains.(Upstill pp.12-15)

Rendering is the process of generating a synthetic image of a scene given a precise description of the geometry and other characteristics of the scene.(Upstill pp.137)

There are five coordinate systems used in RenderMan: object, world, camera, screen, and raster. Object is the coordinate system in which the current geometric primitive is defined. The modeling transformation converts from object coordinates to world coordinates. World is the standard reference coordinate system. The camera transformation converts from world coordinates to camera coordinates. The camera has the vantage point at the origin and the direction of view along the positive z-axis. The projection and screen transformation convert from camera coordinates to screen coordinates which is the 2D normalized coordinate system corresponding to the image plane. The raster transformation converts to raster coordinates. The raster or pixel coordinate system is an area of 1 in this coordinate system corresponds to the area of a single pixel. This coordinate system is either inherited from the display or set by selecting the resolution of the image desired. These coordinate systems are the importance of a rib file structure.(Upstill pp.51-53)

In order to move an object created in the rib file the transformation system is used. Transformation has three parts translate, rotate, and scale. Translate concatenates a translation onto the current transformation. Rotate moves the object on an axis of the transformation. And the scale moves the object up and down on the x, y, and z scale independently.(Upstill pp.111-113)

Rib files also have light source that should are predefined by RenderMan. The ambient light source distributes light uniformly throughout space in all directions. Distant light flows uniformly in space in one direction. Point light source distributes light through space from a single point. And also spotlight source which simulates a cone of light emitted from one point toward another point. This light source covers both position and direction. The positional and directional coordinates of light sources are treated like those of any geometric object. The position and direction are transformed by the current transformation at defined when the light source is declared.(Upstill pp.218-221)

RenderMan also has a RGB color system. RGB stands for red, green and blue. The coordinates for color is color[0 0 0], as numbers fall on the range from 0 to 1. White is [0 0 0] and dark grey is [1 1 1]. The idea of multi-channel color values is essential to color graphics.(Upstill pp.42-43)

This written report is just a run through of some important parts of the rib file structure. Coordinate systems, the transformation system, the light source, and the RGB color system are some necessities in creating and viewing an object.

AASERT Summer Research
Computer Visualization Team
Renderman
Demonstration1

Stage 1: Understand the quadric surfaces.

1) Copy the file "quads.rib" under class directory to your directory.

2) Render it.

3) Look at "quads.rib" to understand the syntax of each of six quadric surfaces.

4) Change parameters of each quadric surface (e.g. radius of the sphere, height of the cylinder), and render it. Figure out if the image is expected. (Answer: The image that appeared after the changes was much longer and wider.)

Stage 2: Demonstrate transformations.

1) Copy the file "quads.rib" under class directory to your directory.

2) Figure out the position of each quadric surface. (Answer: Each quadric surface is at a negative ninety degree angle facing upright.)

3) Delete all transformations after WorldBegin.

4) Using "TransformBegin" and "TransformEnd", add a translation to place each quadric surface in the position you figured out in number two.

5) Render it, and see if each quadric surface is at the right place. (Answer: Yes, each quadric surface is at the negative ninety degree angle as described in number two.

Computer Visualization Research Group
Weekly Report

LaVonna Felton
July 1, 1994

During the week of June 27 - July 1, 1994 , I worked with the UNIX along with the vi commands and individual networking assignments.

UNIX commands:

1. cd - changes the directory
2. ls - lists the file
3. chmod - changes a file
4. more - displays the files
5. cp - copies the file
6. mkdir - creates a new directory
7. cat - displays the file
8. rmdir - removes the directory
9. pwd - displays the current working directory

Dr. Zhang then discussed the RenderMan System with the CV Team.

1. Renderman has a five coordinate system.
   a. object - coordinate system with current geometric primitive is defined
   b. world - standard reference coordinate system
   c. camera - coordinate system with the vantage point at the origin and the direction of view along the positive z-axis
   d. screen - 2-D normalized coordinate system corresponding to the image plane
   e. raster - (pixel coordinate system) an area of 1 in this coordinate system corresponds to the area of a single pixel. This coordinate system is either inherited from the display or set by selecting the resolution of the image desired.
2. The RIB File Structure
3. The RGB System (red, green, blue)
4. Light Sources (ambientlight, distantlight, pointlight, spotlight)
5. Three transformations (translate, rotate, scale)
6. Surfaces (planar, quadric)

INSTRUCTIONS FOR USING RENDERMAN
(Temporary)

1). Log into a workstation using "cast" and start "openwin".

NOTE: Please do not use "sun1", for it is reserved by "class".

2). Open a shell window and telnet to "sun1" using your own username.

3). Create a rib file (ending with .rib).

NOTE: you may first copy a rib file from other sources
and then modify it. You have two resources to use.

a). From user "class".

```
cd ~class
ls *.rib
cp xxxx.rib ~your_username
     (Note: xxxx should be replaced by an actual name)
cd
```

b). From user "zhang"

```
cd ~zhang
cd tutorial
ls
cd chxx     (xx should be replaced by a number)
ls *.rib
cp xxxx.rib ~your_username
     (Note: xxxx should be replaced by an actual name)
cd
```

4). Copy the rib file created to user "class", under the directory with your last name.

```
cp xxxx.rib ~class/your_last_name
```

5). Go to "sun1" and make sure that "class" has logged on "sun1".

And type in

```
cd
cd your_last_name
ls *.rib
```

6). Type in

```
Render xxxx.rib
```

NOTE: you can repeat steps 3) to 6) as many as needed.

NOTE: if you do not want to keep the rib file under your home directory, Steps 2) to 4) can be simplified into the following.

a). Open a shell window and telnet to "sun1" using class.
b).

```
cd ~class
ls *.rib
cp xxxx.rib your_last_name
cd your_last_name
ls *.rib
```

c). Modify xxxx.rib

sun1%

How to Make Textures from Aldus Digital Darkroom (ECSU)

STAGE 1: Copy TIFF files to your diskette.

1). Go to the Mac located in 114 (i.e. vending machine room).

2). Select folder "aldus digital darkroom".

3). Selcet the folder which contains the photo you want and seclect the photo

4). Choose "Save As" in the "File" pull down menu.

5). Select your diskette as the "drive".
Type in the file name, say, "myphoto.tif".
Select "TIFF" (Tag Image File Format).
Click "Save".

6). Choose "8 bits per pixel" in the "TIFF Save Opitions"

STAGE 2: Copy TIFF files to the Mac with IP address "198.85.48.82".

1). Find the Mac with IP address "198.85.48.82".

2). Copy "myphoto.tif" in your diskette to the folder "NET".

STAGE 3: Transfer TIFF files to "sun1" (198.85.48.1).

1). Select "NCSA/BYU Telnet2.5" under "NET".

2). Choose "Open Connection" in the "FILE" pull down menu.

3). Type in "198.85.48.1" as the "Session Name".
Check "FTP Session".
Click "OK".

4). Type in "user".
Type in your user name on "sun1" at the "Username:" prompt, and then your password.

5). Type in the command "put myphoto.tif"

NOTE: you may put other files to sun1. if you want to get file from sun1, type in "get filename".

6). Type in "bye".

STAGE 4: Make textures.

1). log into a SUN worksation using your user name.

2). Type in "txmake myphoto.tif myphoto.txt"

NOTE: you can display a TIFF file by typing "tiffdspy myphoto.tif", and dispaly a texture by typing "txdspy myphoto.txt"
NOTE: For how to incorporate texture files into RIB files, refer to Computer Visualization Summer Research (Worksheet #2), especially Stage 4.

Session Name: 198.85.18.1 1

Computer Visualization Summer Research

Worksheet #1

Stage 1: Understand the quadric surfaces.

1). Copy the file "quads.rib" under ~class directory to your directory.

2). Render it.

3). Look at "quads.rib" to understand the syntax of each of six quadric surfaces.
4). Change parameters of each quadric surface (e.g. radius of the sphere, height of the cylinder), and render it.
Find out if the image is what you expected. If not, figure out why.

Stage 2: Practice transformations

1). Copy the file "quads.rib" under ~class directory to your directory.
2). Figure out the position of each quadric surface.
3). Delete all transformations after WorldBegin.
4). Using "TransformBegin" and "TransformEnd", and a translation to place each quadric surface in the position you figured out in 2).
5). Render it, and see if each quadric surface is at the right place.
If not, try again.

Stage 3: Define the camera.

1). The current camera is positioned at (0 0 -6) and pointed to +z.
2). Try to place camera at different positions and point to different directions and see their effects.
(Note: use the command "trans", and replace the old world to camera transformation by the RIB requests produced by "trans").

Stage 4: Understanding the projection.

1). Change the "fov" parameter, i.e. the number after "fov", render it, and find out the relationship between that parameter and the zoom of the physical camera.
2). Put the second "Projection" line as comment (prefixed by ##).
3). Insert "ScreenWindow" line after the first "Projection" line.
4). Change the size of the screen window, render it, and find out the relationship between the screen window size and the zoom of the physical camera.

Stage 5: The output image.

1). Change the size of the output image to 160 by 100, and see what happens.
2). Add

"origin" [20 20]

at the end of the "Display" line, and see what happens.

RenderMan CV Weekly Report

by Jingyuan Zhang and Stephanie Vaughan

July 5, - July 8 1991

This week, the precollege students along with ECSU students viewed technical, art and design and education slide sets from Siggraph 91. These slides consisted of figure, objects and designs.

The CV student researchers clarified his/her project dealing with designing a scene. Each CV research student will be responsible for designing his/her own scene. This week, the precollege students were asked to read the paper "Modeling and Analysis of Empirical Data in Collaborative Environments."
The students were asked to write the main ideas of the paper and make a reference list for the topics in the library of the Congress.

The ECSU students have began working on the user interface to define a complex object in a RIB file. This project will be broken into three parts. Kevin will work on creating a primitive. Sharon will be responsible for the transformation, union, intersection and difference. Denise responsibility include printing the rib file, deleting a name, copy old new, rename old new, help and the quit sections of the interface.

```
##RenderMan RIB
version 3.03
Display "bigroom.tif" "file" "rgba"
##Display "ROOM" "framebuffer" "rgb"
ShadingRate 1
##Format 256 192 -1
Format 1024 768 -1
CropWindow 0 1 0 1
##Projection "perspective"
Projection "orthographic"
##LightSource "distantlight" 1 "from" [100 100 100] "to" [0 0 0]
##LightSource "ambientlight" 2 "intensity" 0.2
##ScreenWindow -0.5 0.5 -0.375 0.375
ScreenWindow -120 120 -90 90
Identity
Rotate 120.000000 0.000000 0.000000 1.000000
Rotate -35.264390 1.000000 0.000000 0.000000
Rotate 135.000000 0.000000 1.000000 0.000000
Translate -200.000000 -200.000000 -200.000000
Clipping 1e-10 1e+38
WorldBegin
LightSource "distantlight" 1 "from" [200 200 200] "to" [0 0 0] "intensity" 1
LightSource "ambientlight" 2 "intensity" 0.2
LightSource "spotlight" 3 "intensity" [64]"coneangle" [0.20944]"conedeltaangle"
[0.0174533]"from" [150 150 100]"to" [0 0 0]
##TransformBegin
##Translate 25 130 50
##Scale 25 25 25
##LightSource "spotlight" 3 "intensity" 1 "from" [0 0 3] "to" [0 0 0] "coneangle" 60
"conedeltaangle" 10 "beamdistribution" 2
##TransformEnd
##wall    begin    ----------
AttributeBegin
SolidBegin "primitive"
Translate -0.5 75 50
Scale 1 150 100
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
## +x
```



```
AttributeBegin
Surface "brick"
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
AttributeEnd
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
SolidBegin "primitive"
Translate 75 -0.5 50
Scale 150 1 100
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
AttributeBegin
Surface "brick"
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
AttributeEnd
## +x
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
##Color [0 0 1]
SolidBegin "primitive"
Translate 75 75 -0.5
Scale 150 150 1
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
## +x
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
## +z
AttributeBegin
Surface "wood" "ringscale" 100
```

```
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
AttributeEnd
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
AttributeEnd
##wall end

##desk begin ----------
AttributeBegin
Translate 40 100 50
Rotate 90 0 0 1
Rotate 90 1 0 0
Scale 50 50 50
Surface "plastic"
SolidBegin "union"
SolidBegin "primitive"
Color [0.2 0.4 0.8]
Translate 0 0 0
Scale 1 1 1
Polygon "P" [-1 -0.05 0.5 -1 -0.05 -0.5 1 -0.05 -0.5 1 -0.05 0.5]
Polygon "P" [-1 -0.05 0.5 -1 0.05 0.5 -1 0.05 -0.5 -1 -0.05 -0.5]
Polygon "P" [1 0.05 -0.5 -1 0.05 -0.5 -1 0.05 0.5 1 0.05 0.5]
Polygon "P" [1 0.05 -0.5 1 0.05 0.5 1 -0.05 0.5 1 -0.05 -0.5]
Polygon "P" [1 -0.05 0.5 1 0.05 0.5 -1 0.05 0.5 -1 -0.05 0.5]
Polygon "P" [-1 0.05 -0.5 1 0.05 -0.5 1 -0.05 -0.5 -1 -0.05 -0.5]
SolidEnd
SolidBegin "primitive"
Color [1 1 1]
Translate 0.5 -0.6 0
Scale 1 1 1
Polygon "P" [-0.05 -.5 0.5 -0.05 -.5 -0.5 0.05 -.5 -0.5 0.05 -.5 0.5]
Polygon "P" [-0.05 -.5 0.5 -0.05 -.5 0.5 -0.05 .5 -0.5 -0.05 -.5 -0.5]
Polygon "P" [0.05 .5 -0.5 -0.05 .5 -0.5 -0.05 .5 0.5 0.05 .5 0.5]
Polygon "P" [0.05 .5 -0.5 0.05 .5 0.5 0.05 -.5 0.5 0.05 -.5 -0.5]
Polygon "P" [0.05 -.5 0.5 0.05 .5 0.5 -0.05 .5 0.5 -0.05 -.5 0.5]
Polygon "P" [-0.05 .5 -0.5 0.05 .5 -0.5 0.05 -.5 -0.5 -0.05 -.5 -0.5]
SolidEnd
SolidBegin "primitive"
Translate -0.5 -0.6 0
Color [1 1 1]
Scale 1 1 1
Polygon "P" [-0.05 -.5 0.5 -0.05 -.5 -0.5 0.05 -.5 -0.5 0.05 -.5 0.5]
```



```
Polygon "P" [-0.05 -.5 0.5 -0.05 -.5 0.5 -0.05 .5 -0.5 -0.05 -.5 -0.5]
Polygon "P" [0.05 .5 -0.5 -0.05 .5 -0.5 -0.05 .5 0.5 0.05 .5 0.5]
Polygon "P" [0.05 .5 -0.5 0.05 .5 0.5 0.05 -.5 0.5 0.05 -.5 0.5]
Polygon "P" [0.05 -.5 0.5 0.05 .5 0.5 -0.05 .5 0.5 -0.05 -.5 0.5]
Polygon "P" [-0.05 .5 -0.5 0.05 .5 -0.5 0.05 -.5 -0.5 -0.05 -.5 -0.5]
SolidEnd
SolidEnd
AttributeEnd
##desk end

##chair begin ----------
AttributeBegin
Translate 90 100 30
Rotate 180 0 0 1
Rotate 90 1 0 0
Scale 50 50 50
Surface "plastic"
Color [1 1 1]
SolidBegin "union"
SolidBegin "primitive"
Color [.2 .4 .8]
Translate 0 0 0
Scale 1 1 1
Polygon "P" [-0.4 -0.025 0.4 -0.4 -0.025 -0.4 0.4 -0.025 -0.4 0.4 -0.025 0.4]
Polygon "P" [-0.4 -0.025 0.4 -0.4 0.025 0.4 -0.4 0.025 -0.4 -0.4 -0.025 -0.4]
Polygon "P" [0.4 0.025 -0.4 -0.4 0.025 -0.4 -0.4 0.025 0.4 0.4 0.025 0.4]
Polygon "P" [0.4 0.025 -0.4 0.4 0.025 0.4 0.4 -0.025 0.4 0.4 -0.025 -0.4]
Polygon "P" [0.4 -0.025 0.4 0.4 0.025 0.4 -0.4 0.025 0.4 -0.4 -0.025 0.4]
Polygon "P" [-0.4 0.025 -0.4 0.4 0.025 -0.4 0.4 -0.025 -0.4 -0.4 -0.025 -0.4]
SolidEnd
SolidBegin "primitive"
color [ 1 1 1]
Translate -0.33 0.33 0
Scale 1 1 1
Polygon "P" [-0.025 -0.4 0.4 -0.025 -0.4 -0.4 0.025 -0.4 -0.4 0.025 -0.4 0.4]
Polygon "P" [-0.025 -0.4 0.4 -0.025 0.4 0.4 -0.025 0.4 -0.4 -0.025 -0.4 -0.4]
Polygon "P" [0.025 0.4 -0.4 -0.025 0.4 -0.4 -0.025 0.4 0.4 0.025 0.4 0.4]
Polygon "P" [0.025 0.4 -0.4 0.025 0.4 0.4 0.025 -0.4 0.4 0.025 -0.4 -0.4]
Polygon "P" [0.025 -0.4 0.4 0.025 0.4 0.4 -0.025 0.4 0.4 -0.025 -0.4 0.4]
Polygon "P" [-0.025 0.4 -0.4 0.025 0.4 -0.4 0.025 -0.4 -0.4 -0.025 -0.4 -0.4]
SolidEnd
SolidBegin "primitive"
Color [1 1 1]
```

```
Translate .33 -.65 .33
Scale 1 1 1
Polygon "P" [-0.05 -0.65 0.05 -0.05 -0.65 -0.05 0.05 -0.65 -0.05 0.05 -0.65 0.05]
Polygon "P" [-0.05 -0.65 0.05 -0.05 0.65 0.05 -0.05 0.65 -0.05 -0.05 -0.65 -0.05]
Polygon "P" [0.05 0.65 -0.05 -0.05 0.65 -0.05 -0.05 0.65 0.05 0.05 0.65 0.05]
Polygon "P" [0.05 0.65 -0.05 0.05 0.65 0.05 0.05 -0.65 0.05 0.05 -0.65 -0.05]
Polygon "P" [0.05 -0.65 0.05 0.05 0.65 0.05 -0.05 0.65 0.05 -0.05 -0.65 0.05]
Polygon "P" [-0.05 0.65 -0.05 0.05 0.65 -0.05 0.05 -0.65 -0.05 -0.05 -0.65 -0.05]
SolidEnd
SolidBegin "primitive"
Translate -.33 -.65 .33
Scale 1 1 1
Polygon "P" [-0.05 -0.65 0.05 -0.05 -0.65 -0.05 0.05 -0.65 -0.05 0.05 -0.65 0.05]
Polygon "P" [-0.05 -0.65 0.05 -0.05 0.65 0.05 -0.05 0.65 -0.05 -0.05 -0.65 -0.05]
Polygon "P" [0.05 0.65 -0.05 -0.05 0.65 -0.05 -0.05 0.65 0.05 0.05 0.65 0.05]
Polygon "P" [0.05 0.65 -0.05 0.05 0.65 0.05 0.05 -0.65 0.05 0.05 -0.65 -0.05]
Polygon "P" [0.05 -0.65 0.05 0.05 0.65 0.05 -0.05 0.65 0.05 -0.05 -0.65 0.05]
Polygon "P" [-0.05 0.65 -0.05 0.05 0.65 -0.05 0.05 -0.65 -0.05 -0.05 -0.65 -0.05]
SolidEnd
SolidBegin "primitive"
Translate .33 -.65 -.33
Scale 1 1 1
Polygon "P" [-0.05 -0.65 0.05 -0.05 -0.65 -0.05 0.05 -0.65 -0.05 0.05 -0.65 0.05]
Polygon "P" [-0.05 -0.65 0.05 -0.05 0.65 0.05 -0.05 0.65 -0.05 -0.05 -0.65 -0.05]
Polygon "P" [0.05 0.65 -0.05 -0.05 0.65 -0.05 -0.05 0.65 0.05 0.05 0.65 0.05]
Polygon "P" [0.05 0.65 -0.05 0.05 0.65 0.05 0.05 -0.65 0.05 0.05 -0.65 -0.05]
Polygon "P" [0.05 -0.65 0.05 0.05 0.65 0.05 -0.05 0.65 0.05 -0.05 -0.65 0.05]
Polygon "P" [-0.05 0.65 -0.05 0.05 0.65 -0.05 0.05 -0.65 -0.05 -0.05 -0.65 -0.05]
SolidEnd
SolidBegin "primitive"
Color [1 1 1]
Translate -.33 -.65 -.33
Scale 1 1 1
Polygon "P" [-0.05 -0.65 0.05 -0.05 -0.65 -0.05 0.05 -0.65 -0.05 0.05 -0.65 0.05]
Polygon "P" [-0.05 -0.65 0.05 -0.05 0.65 0.05 -0.05 0.65 -0.05 -0.05 -0.65 -0.05]
Polygon "P" [0.05 0.65 -0.05 -0.05 0.65 -0.05 -0.05 0.65 0.05 0.05 0.65 0.05]
Polygon "P" [0.05 0.65 -0.05 0.05 0.65 0.05 0.05 -0.65 0.05 0.05 -0.65 -0.05]
Polygon "P" [0.05 -0.65 0.05 0.05 0.65 0.05 -0.05 0.65 0.05 -0.05 -0.65 0.05]
Polygon "P" [-0.05 0.65 -0.05 0.05 0.65 -0.05 0.05 -0.65 -0.05 -0.05 -0.65 -0.05]
SolidEnd
SolidEnd
AttributeEnd
#chair end
```



```
##floor lamp begin
-----------------------------------------------------------------
AttributeBegin
##Translate 25 125 50
Translate 130 25 50
##Rotate 180 0 0 1
##Rotate 90 1 0 0
Scale 25 25 25
Surface "matte"
Color [0.5 0.2 1]
Cylinder 0.1 -2 2 360
    TransformBegin
    Translate 0 0 -2
    Rotate 180 0 -2 0
   Cone -1 0.5 360
    TransformEnd
Color [0 1 1]
## The following line has been added.
Displacement "ripple" "amplitude" 1 "frequency" 16
Hyperboloid 0.5 -1 2 0.2 0 3 360
AttributeEnd
##floor lamp end

##desk lamp begin
-----------------------------------------------------------------
AttributeBegin
Surface "metal"
Translate 40 140 60
Rotate -60 0 0 1
Scale 7 7 7
##base
SolidBegin "primitive"
Color [.1 .1 .1]
Translate 0 0 0
Scale 2 0.4 2
##  -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
##  -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
##  +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
##  +x
```

```
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
## stand
SolidBegin "primitive"
Color [.9 .9 0.9]
Cylinder 0.4 0.2 4.8 360
Disk 0.2 0.4 360
Disk 4.8 0.4 -360
SolidEnd
##shade
Color [.5 1 .2]
Translate 0.5 0 5
Rotate -60 0 1 0
Rotate 90 1 0 0
Cylinder 1 -2 2 180
Disk -2 1 180
Disk 2 1 180
AttributeEnd
##desk lamp end

##garbage can begin
-----------------------------------------------------------------
AttributeBegin
Color [0.5 0.5 0.5]
Surface "plastic"
Translate 80 30 29
##Rotate -60 0 0 1
Scale 2 2 2
Torus 6 0.5 0 180 360
Hyperboloid 6 0 0 4 0 -14 360
Disk -14 4 360
AttributeEnd
##garbage can end

##frame begin
-----------------------------------------------------------------
AttributeBegin
Translate 50 0.5 75
Rotate 90 0 0 1
```



```
Scale 10 10 10
##Center
SolidBegin "primitive"
Translate 0.25 0 0
Scale 0.5 4 3
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
## +x
AttributeBegin
TextureCoordinates 0 0.125 1 0.125 0 0.875 1 0.875
Surface "mytexture" "tmap" "hayden2.txt"
Patch "bilinear" "P" [0.5 0.5 0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 0.5 -0.5 -0.5]
AttributeEnd
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
## the following line has been added
Scale 1 1.333333 0.75
##CV Frame
Surface "plastic"
Color [0.5 0.4 0.2]
SolidBegin "union"
##Top Frame
SolidBegin "primitive"
Translate 0 2.1 2.6
Rotate -90 0 0 1
Rotate -90 1 0 0
Scale 1 1 1
## Bottom
Patch "bilinear" "P" [0 0 0 4.2 0 0 0 0.6 0 4.2 0.6 0]
## Top
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0.6 0.5 4.2 0.6 0.5]
## Left Side
Patch "bilinear" "P" [0 0 0.8 0 0.6 0.5 0 0 0 0 0.6 0]
## Right Side
Patch "bilinear" "P" [4.2 0 0.8 4.2 0.6 0.5 4.2 0 0 4.2 0.6 0]
## Rear Side
```

```
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0 0 4.2 0 0]
## Front
Patch "bilinear" "P" [0 0.6 0.5 4.2 0.6 0.5 0 0.6 0 4.2 0.6 0]
SolidEnd
##Bottom Frame
SolidBegin "primitive"
Translate 0 -2.1 -2.6
Rotate 90 0 0 1
Rotate 90 1 0 0
Scale 1 1 1
## Bottom
Patch "bilinear" "P" [0 0 0 4.2 0 0 0 0.6 0 4.2 0.6 0]
## Top
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0.6 0.5 4.2 0.6 0.5]
## Left Side
Patch "bilinear" "P" [0 0 0.8 0 0.6 0.5 0 0 0 0 0.6 0]
## Right Side
Patch "bilinear" "P" [4.2 0 0.8 4.2 0.6 0.5 4.2 0 0 4.2 0.6 0]
## Rear Side
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0 0 4.2 0 0]
## Front
Patch "bilinear" "P" [0 0.6 0.5 4.2 0.6 0.5 0 0.6 0 4.2 0.6 0]
SolidEnd
## Left Frame
SolidBegin "primitive"
Translate 0 2.1 -2.6
Rotate 90 0 1 0
Rotate 180 0 0 1
Scale 1.24 1 1
## Bottom
Patch "bilinear" "P" [0 0 0 4.2 0 0 0 0.6 0 4.2 0.6 0]
## Top
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0.6 0.5 4.2 0.6 0.5]
##Left Side
Patch "bilinear" "P" [0 0 0.8 0 0.6 0.5 0 0 0 0 0.6 0]
## Right Side
Patch "bilinear" "P" [4.2 0 0.8 4.2 0.6 0.5 4.2 0 0 4.2 0.6 0]
## Rear Side
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0 0 4.2 0 0]
## Front
Patch "bilinear" "P" [0 0.6 0.5 4.2 0.6 0.5 0 0.6 0 4.2 0.6 0]
SolidEnd
##Right Frame
```



```
SolidBegin "primitive"
Translate 0 -2.1 2.6
Rotate 90 0 1 0
Scale 1.24 1 1
## Bottom
Patch "bilinear" "P" [0 0 0 4.2 0 0 0 0.6 0 4.2 0.6 0]
## Top
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0.6 0.5 4.2 0.6 0.5]
##Left Side
Patch "bilinear" "P" [0 0 0.8 0 0.6 0.5 0 0 0 0 0.6 0]
## Right Side
Patch "bilinear" "P" [4.2 0 0.8 4.2 0.6 0.5 4.2 0 0 4.2 0.6 0]
## Rear Side
Patch "bilinear" "P" [0 0 0.8 4.2 0 0.8 0 0 0 4.2 0 0]
## Front
Patch "bilinear" "P" [0 0.6 0.5 4.2 0.6 0.5 0 0.6 0 4.2 0.6 0]
SolidEnd
SolidEnd
AttributeEnd
##frame end

##clock begin ----------
AttributeBegin
Translate 0.5 35 75
Rotate 90 0 1 0
Rotate -90 0 0 1
Scale 2 2 2
Surface "plastic"

Color [1 1 1]
Disk 0 8 360

Color [0.2 0.4 0.8]
Torus 9 1 0 180 360
##Cylinder 10 -1 0 360
Color [1 0 0]
TransformBegin
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 30 0 0 1
Translate 7 0 0
```

```
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 60 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 90 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 120 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 150 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 180 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 210 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 240 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 270 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
```



```
Rotate 300 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd
TransformBegin
Rotate 330 0 0 1
Translate 7 0 0
Disk 0.1 0.5 360
TransformEnd

## Center
Color [1 0 0]
Disk 0.3 0.75 360

Color [0 0 0]
## Minute hand
TransformBegin
##Rotate -30 0 0 1
Polygon "P" [0 -6 0.1 1 0 0.1 1 0.75 0.1 -1 0.75 0.1 -1 0 0.1]
Translate 0 0.75 0
Disk 0.1 1 180
TransformEnd
## hour hand
TransformBegin
Rotate -90 0 0 1
Polygon "P" [0 -4 0.2 1 0 0.2 1 0.75 0.2 -1 0.75 0.2 -1 0 0.2]
Translate 0 0.75 0
Disk 0.2 1 180
TransformEnd
AttributeEnd
##clock end


##Monitor begin
------------------------------------------------------------
AttributeBegin
Translate 50 100 50
Scale 2 2 2

Translate 0 0 15.5
AttributeBegin
##Color [1 0 0]
##TextureCoordinates 0 0.125 1 0.125 0 0.875 1 0.875
```

```
Surface "mytexture" "tmap" "room.txt"
Translate -46 0 0
Rotate -10 0 0 1
Cylinder 50 -6 6 15
AttributeEnd

AttributeBegin
Color [0 0 1]
Translate 7.6 6 -10
##Rotate -10 0 0 1
Scale 1 2 1.5
Patch "bilinear" "P" [0 -0.5 0.5 0 0.5 0.5 0 -0.5 -0.5 0 0.5 -0.5]
AttributeEnd
Surface "plastic"
Color [1 1 1]
SolidBegin "difference"
SolidBegin "union"
SolidBegin "intersection"
SolidBegin "union"
##Monitor back, cube
SolidBegin "primitive"
Translate -5 0 0
Scale 10 10 10
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
## +x
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
##Monitor Center,
SolidBegin "primitive"
Translate 0 0 0
Scale 1 1 1
## -y
Patch "bilinear" "P" [30 -25 25 0 -5 5 30 -25 -25 0 -5 -5]
## -x
```



```
Patch "bilinear" "P" [0 -5 5 0 5 5 0 -5 -5 0 5 -5]
## +x
Patch "bilinear" "P" [30 25 25 30 -25 25 30 25 -25 30 -25 -25]
## +y
Patch "bilinear" "P" [0 5 5 30 25 25 0 5 -5 30 25 -25]
## +z
Patch "bilinear" "P" [0 -5 5 30 -25 25 0 5 5 30 25 25]
## -z
Patch "bilinear" "P" [30 25 -25 30 -25 -25 0 5 -5 0 -5 -5]
SolidEnd
SolidEnd
SolidBegin "primitive"
Translate -45 0 0
Cylinder 50 -50 50 360
Disk 50 50 360
Disk -50 50 -360
SolidEnd
SolidEnd
SolidBegin "primitive"
Translate -2.5 0 -2.5
Sphere 5 -5 0 360
Disk 0 5 360
SolidEnd
SolidBegin "primitive"
Translate -2.5 0 -10
Cone 5 5 360
Disk 0 5 360
SolidEnd
SolidBegin "primitive"
Translate -2.5 0 0
Cylinder 7 -10.5 -10 360
Disk -10 7 360
Disk -10.5 7 -360
SolidEnd
SolidEnd
##Monitor Center,
SolidBegin "primitive"
Translate 2 0 0
Scale 1 1 1
## -y
Patch "bilinear" "P" [30 -25 25 0 -5 5 30 -25 -25 0 -5 -5]
## -x
Patch "bilinear" "P" [0 -5 5 0 5 5 0 -5 -5 0 5 -5]
```

```
## +x
Patch "bilinear" "P" [30 25 25 30 -25 25 30 25 -25 30 -25 -25]
## +y
Patch "bilinear" "P" [0 5 5 30 25 25 0 5 -5 30 25 -25]
## +z
Patch "bilinear" "P" [0 -5 5 30 -25 25 0 5 5 30 25 25]
## -z
Patch "bilinear" "P" [30 25 -25 30 -25 -25 0 5 -5 0 -5 -5]
SolidEnd
SolidEnd
##System Box
Color [ 1 1 1]
SolidBegin "primitive"
Translate -2.5 0 -13
Scale 15 15 5
## -y
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 -0.5]
## -x
Patch "bilinear" "P" [-0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5 -0.5 -0.5 -0.5 0.5 -0.5]
## +y
Patch "bilinear" "P" [-0.5 0.5 0.5 0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
## +x
Patch "bilinear" "P" [0.5 -0.5 0.5 0.5 0.5 0.5 0.5 -0.5 -0.5 0.5 0.5 -0.5]
## +z
Patch "bilinear" "P" [-0.5 -0.5 0.5 0.5 -0.5 0.5 -0.5 0.5 0.5 0.5 0.5 0.5]
## -z
Patch "bilinear" "P" [-0.5 -0.5 -0.5 0.5 -0.5 -0.5 -0.5 0.5 -0.5 0.5 0.5 -0.5]
SolidEnd
AttributeEnd
##Monitor end

WorldEnd
```


Computer Visualization Summer Research (Worksheet #2)

Stage 1: CSG (Constructive Solid Geometry).

1). Copy the file "glass.rib" under ~class directory to your directory.

2). Render it.

3). Look at "glass.rib" to understand the syntax of CSG, and
Draw the CSG tree for that RIB file.

Stage 2: Color and Opacity

1). Change the color of the blue sphere in the glass to whatever color you want, and render it.
2). Change the opacity of the glass and render it.
The syntax is
Opacity [R G B] .
Opacity [1 1 1] is completely opaque, and
Opacity [0 0 0] is completely transparent.

Stage 2: Manipulating light sources.

In the "glass.rib", there are 4 kinds of lights. Try the following.
1). Change the parameters of the ambient light.
2). Change the parameters of the distant light.
3). Change the parameters of the point light.
4). Change the parameters of the spot light.

Stage 3: Standard surfaces.

1). Copy the file "torus.rib" under ~class directory to your directory.

2). Render it.

3). Try to change the surface type "plastic" to another surface type like "constant", "matte" or "metal".

Stage 4: Texture mapping.

1). Copy the file "quads_b.rib" under ~class directory to your directory.

2). Copy all the files ending with .sl (shading language file) and all the files ending with .txt (texture file) under ~class directory to your directory.

3). Compile the shading language files using
shader *.sl

4). Examine "quads_b.rib" to figure out how to incorporate shading language files into RIB files. Exchange surface types (e.g., exchange "wood" with "blue_marble").

5). Figure out how to incorporate texture files into RIB files. Replace texture file "coated.txt" by another texture file.

A Winning Formula

by C. J. Houtchens

The women in this story are in a minority group so small that in some years when national statistics are gathered, there is no one in the category at all. These women have paid a high price to be counted among their small number and have beaten the long odds placed against them simply because of their gender and race.

Martha Brown '89, Linda Hayden '89, Joan Sterling Langdon '89, and Elaine Smith '88 hold Ph.D.s in mathematics from American University. Their colleague Ann Taylor '88 earned her AU doctorate in education administration with an emphasis on teaching college-level mathematics. Their achievement defies the notion that females should abandon math in junior high school and leave the difficult calculations to men. It rejects the idea that an advanced degree is not an attainable goal for African-Americans, who historically have not funneled into the graduate school system at a rate anywhere near comparable to that of white Americans.

In the seven years 1985 through 1991, according to a National Research Council survey contained in the June 1992 Commission on Professions in Science and Technology report *Professional Women and Minorities*, 1,887 white men, 470 white women, 27 African-American men, and a mere 12 African-American women in the entire United States reported having earned a doctorate in mathematics. It doesn't take more than a knowledge of simple arithmetic to figure out that AU's four female African-American math Ph.D.s represent a full *one-third* of a very elite club. And with four more African-American women currently working on math doctorates at AU, it's clear the success of those seven years wasn't just a fluke.

We must be doing something right.

If you ask Nina Roscher, chair of AU's chemistry department and a faculty member since 1974, what it is that makes the difference, her cheeks pinken and she says quietly, "Dr. [Mary] Gray and I work hard for women." Gray, a former chair of AU's mathematics and statistics department and a professor there since 1968, is the person most often credited with focus-

It doesn't take more than a knowledge of simple arithmetic to figure out that AU's female African-American math Ph.D.s represent a full one-third of a very elite club.

PHOTO BY MELISSA LAITSCH

Elaine Smith points the way in a classroom at the District of Columbia's Wilson High School, where she teaches and also directs a new after-school tutoring project, the Math Center. Smith, who taught at the college level for many years, decided that she could steer more kids into math by reaching them in high school, before they begin to opt out of hard courses. "Our kids are slipping through the cracks," she says. "We have to meet [them] where we find them and move them from here."

ng AU's commitment to women and minorities in math. Roscher has had similar success boosting AU's stats for women and minorities with Ph.D.s in her own field.

"I was one of 9 women out of 450 chemistry graduate students" working toward a 1964 Ph.D. at Purdue University, Roscher recalls, with a slight tightening of her lips. "Very unpleasant. . . . It certainly has a lot to do with why I am committed."

Gray claims to have had an easier time in grad school at the University of Kansas. Except, she says, "I had a real jerk for my first class, who said, 'What are you doing here? Why don't you stay home and take care of kids?' But that just made me work harder." Gray has long been committed to civil, women's, and human rights, and was elected chair of Amnesty International USA last fall (see page 7).

Both Gray and Roscher have been in the trenches, and neither is the type to pull punches. Graduate school in math is an arduous and *at least* three-year-long haul in a woman's prime childbearing time . . . with no guarantees of good jobs, promotions, or tenure at the other end, says Gray. "[There] is just point-blank prejudice on the part of people in the mathematical community that women can't do math. . . . When I started out thirty years ago they would say it. Now they don't say it, but it still affects their hiring decisions, their promotion and tenure decisions," Gray says. Adds Roscher, "[And] we can't appreciate fully the problems that black women face."

But being realistic about all those problems is the first thing AU does right. For starters, through networking with professional associations, other academic institutions, and alumni of the mathematics Ph.D. program, the university makes a conscious effort to attract women to the program who may have been out of school for a while. Candidates for the mathematics education doctorate must have already earned a master's degree in either math or education. Both Gray and Roscher say that older students tend to be more committed to completing the Ph.D. But they also often have more demands to juggle—like spouses and children—and are likely to be giving up fairly comfortable earnings in order to take on the life of a student again.

Linda Hayden had earned a master's of teaching in mathematics from the University of Cincinnati in 1972 and was teaching math and computer science at Elizabeth City University, North Carolina, when she decided that she really needed a Ph.D. She tried a computer science program at another university first but encountered "a lot of frustration," she says. "[The program] was brand new. . . . Courses weren't in place, and teachers couldn't give you a curriculum and say, 'Choose from these courses. When you do this amount of work then you will take your comps [comprehensive examinations], and so on. . . . It was all men, and there was just no understanding at all, no role models there. . . . I said, this is just not

PHOTO BY MELISSA LAITSCH

Joan Sterling Langdon takes a rare pause at her desk at Bowie State University, Maryland, where she is an associate professor of computer science. For the past four years, Langdon has also served as director of an undergraduate and graduate student training program at NASA's Goddard Space Flight Center, cosponsored by the center and Bowie State. "The 'Dr.' in front of your name really makes a difference to NASA," she says.

able at this time at this institution." Hayden settled for a master's degree computer science instead, as did graduate school colleague, Joan rling Langdon. But Langdon kept king. When she heard about the AU gram she says, "I called Linda up in morning, somewhere like 6 o'clock. She listened. She said, 'Okay.' Then hung up the telephone."

'I rolled over and went back to ep," laughs Hayden.

'She called me back about an hour r, and she said, 'Joan, I really wish hadn't called me,'" Langdon jokes. ien, of course, everything started veloping."

'Joan and I were both very motivat- . . . and were looking for a place ere [earning the Ph.D.] was do-able hin a finite amount of time," Hayden plains. "We got the impression that if were dedicated and worked hard, would get it. We didn't mind that. . . . e I tell my students, 'Put your butt in chair and your head in the book.' . . . sometimes you can get into situa- ns where there are a lot of politics olved. Also, being black women, in ces where there is a lot of racism, re are a lot of other undercurrents t sort of deter you from your goal. didn't feel that [when we visited]."

"You don't want to go to a school ere you don't think you are going to out of there," says Ann Taylor. At thune-Cookman College in Daytona ach, Florida, where she has taught ce 1968 and is now vice president of idemic affairs and dean of the facul- Taylor knew two faculty members o had gotten doctorates from AU. nerican had a track record, as far as as concerned, with African-Ameri- is being able to complete their rk."

That's the kind of track record that

PHOTO BY MELISSA LATSCH

Martha Brown demonstrates her teacher training workshop leadership style. Brown, supervisor of mathematics for Prince George's County, Maryland, schools, oversees curriculum and staff development for the county's 175 kindergarten through twelfth-grade schools, a job that keeps her on the road in Prince George's and at conferences around the country. Nevertheless, she is working part-time on another degree . . . in divinity, at Wesley Seminary.

historically black schools, colleges, and universities—like Bethune-Cookman—have enjoyed for many years. For one thing, they offer strong role models. For another, says Elaine Smith, they have the reputation of being "far more nurturing and supportive of their students." Both are factors AU College of Arts and Sciences dean Betty Bennett points to in assessing AU's success. "The College of Arts and Sciences has always had a large number of women faculty members," she says. Out of twenty-five women on the math and stat faculty, six, or 24 percent, are women—a high percentage considering that nationally women constitute only 5 percent of math faculties at Ph.D.-granting institutions. Two of AU's female math faculty are tenured full professors. One of those, Nancy Flournoy, is the current department chair. "These women, who have accomplished so much themselves, have served as role models to the students," Bennett says. "And the commitment of the faculty to increasing the numbers of women and minorities has resulted in attracting those students and seeing them through. I think they realize that they are coming to a supportive environment."

But *supportive*, as Joan Langdon points out, doesn't necessarily mean *coddling*. Langdon grew up in South Carolina during the days of segregation and attended an all-black high school. "[The teachers] were not just peachy keen and everything that you did was wonderful. They let you know when you were messing up and they let you know when you were doing the right thing," she says. "That is the way it is with Mary Gray. . . . Primarily, she made sure that we stayed on track and did our work and got out of The American University."

Langdon remembers a time when she was working on her dissertation and was "really busy" but wanted to go home at Christmas to see her dad. "Dr. Gray said to me, 'Go . . . when you finish your degree.'" Langdon says her father laughed and agreed, "Well, you know, you really should work before you play."

"Mary has been a wonderful friend," says Ann Taylor. "I am not going to say that I was happy with Mary all the way through, because sometimes she gave me a fit. But it was all worth it. She was that way because she wanted me to do well and she wanted me to do my best."

Elaine Smith and Gray both acknowl-

ge that they butted eads through most of he ten years that Smith as working on her egree while also teach- ig as an adjunct faculty ember. "There were a t of things I admire bout Mary and still o. . . . [But] we are both, ry strong women, very rong women. . . . You now she didn't take uch. I didn't take uch. I have a mouth. he has a mouth. You et the picture," Smith ys. For Smith, support so came in the form of lassmates—a network ray explicitly encour- ges.

PHOTO BY TONY POWELL

Linda Hayden '89 works out at the blackboard at Elizabeth City University, North Carolina, where she is a professor in the department of mathematics and computer science. Hayden recalls graphing functions at home for hours as a high school student "because they were beautiful to me." Today, in addition to regular teaching duties, she pursues grants to provide math conference trips and special project experience for a select group of undergraduate math majors she calls "Hayden's Scholars."

"I would never have ade it without them," nith says. "Never. I bugged people to eath. Child, there were spouses who topped talking to me because I was lling up their husbands so much at ght" looking for help with complicat- d concepts. "I didn't even care because had to get it, and they were willing to elp me get it. That is how I survived."

Says Martha Brown, "The big thing r me was, [would] AU permit part- me participation in the program? I had job. I needed a job. And if I [was going o get a Ph.D.] I had to do it on a part-time basis. That was not . . . a problem with AU."

But the other women were taking time out of their already established careers and looked to the university for solid financial help. That's where part-time teaching positions and Nina Roscher came in.

Roscher has spent more than half her career in university administration, particularly in graduate and academic affairs, and since 1986, has also held a part-time position as program director of science education in the National Science Foundation. She knows the world of grants cold and since 1981 has used that expertise to bring hundreds of thousands of dollars to AU for more than thirty women and minority graduate students in math, chemistry, and law, through the federal government's Patricia Roberts Harris Fellowship Program. Named for the first African-American woman to attain a cabinet position (secretary of housing and urban development and of health education and welfare under Pres. Jimmy Carter) and to head a U.S. law school (at Howard University) the program provides selected graduate schools with stipends for students whose race or gender or both have been, in federal jargon, *underrepresented* in certain fields. For mid-career students like Hayden, whose husband and nine-year-old son packed up and moved to Washington with her, and Langdon, who, when she came to AU was a single mother with six-year-old twin girls and a thirteen-year-old daughter, the stipends made all the difference.

Make no mistake, for every one of these women, getting that Ph.D. was a sacrifice. But they have more than gender and race in common. These women also possess incredible energy—just

But supportive doesn't necessarily mean coddling.

try pinning one of them down on the telephone after six o'clock in the morning or before ten o'clock at night—and another quality, one that perhaps can best be called tenacity.

Says Elaine Smith, "It was a struggle, honey. It was a long and hard struggle from beginning to end. . . . I am fond to this day of telling my students that I got my Ph.D. in Perseverance. At some point I just refused to walk away not having that piece of paper to show for all the time and energy that I knew I had already put into it."

"It was tough for me," says Ann Taylor, who "burned up I-95 and the airlines" between Washington, where she lived while working on her degree, and Daytona Beach, where her husband is on the faculty at Bethune-Cookman. "I shed a lot of tears. I can remember many an evening leaving the campus and crying all the way through Rock Creek Park as I drove home. But I don't know anybody who gets through a doctorate without shedding a tear."

And was it worth it?

"I can honestly say yes. We were up a lot of hours, we were poor, we went through a lot of changes . . . but it was all worth it. It was a good move," Langdon says.

"Yes," says Smith. "To see the looks on my students' faces as I tell them the stories of how I went after that degree and to hear them ask me why is it that they call me 'Doctor,' giving them the opportunity to share some of these things, that alone has been worth it."

PHOTO BY HILARY SCHWAB

NINA ROSCHER

MARY GRAY

What with the promotions and the job offers, the conferences and the grants, and the recognition that all of these women have received since earning that degree, it would be easy to say that Smith is just mouthing platitudes. But that's clearly not her style. What she is talking about is indeed the real payoff of AU's success . . . because the women in this story believe that every African-American woman who achieves that Ph.D. is a new role model for an elementary school child, or a high school student, or a college student who could be in graduate school someday and on the way to making sure that this small group gets bigger faster.

Says Linda Hayden, who has returned to teaching at historically black Elizabeth City University, where this past year 60 percent of the seniors from a group she mentored decided to go on to graduate school, "That's what gets me up in the morning."

PHOTO BY HILARY SCHWAB

Ann Taylor, vice president of academic affairs and dean of the faculty of Bethune-Cookman College, *left*, savors a sweet moment with her former Ph.D. advisor and mentor, AU professor of mathematics and statistics Mary Gray, following the April 1992 ceremony in which Taylor received one of the university's first Lodestar awards, recognizing her achievements.

Linda Bailey Hayden, PH.D.
Mathematics and CS Dept., ECSU
Box 672 1704 Weeksville Rd
Elizabeth City, NC 27909
(919) 335-3617 fax: (919)335-3487

1318 Roosevelt Blvd.
Portsmouth, VA 23701
(804) 485-0979
email: lhayden@uncecs.edu

EDUCATION

NSF-UFE, COMPUTATIONAL SCIENCE 1993-94, NC Supercomputer Center, Triangle Park,NC
NSF-UFE, PARALLEL PROCESSING 1992-93, Colgate University, Hamilton, NY.
NSF-UFE, COMPUTER GRAPHICS 1990 & 1993, Georgia State University, Atlanta, GA.
NSF-UFE, SOFTWARE ENGINEERING 1990, Georgia State University, Atlanta, GA.
PH.D. MATHEMATICS 1988, American Univ., Washington, DC.
M.S.. COMPUTER SCIENCE 1983 Old Dominion University, Norfolk, VA.
M.A. MATHEMATICS/EDUCATION 1972 University of Cincinnati, Cincinnati, Ohio.
B.S. MATHEMATICS/PHYSICS 1970 Virginia State University, Petersburg, VA.

TEACHING EXPERIENCE

PROFESSOR OF COMPUTER SCIENCE, 1989-present, Elizabeth City State Univ.
ASSOCIATE PROFESSOR OF COMPUTER SCIENCE, 1988-89, Univ. of the District of Columbia.
ASSISTANT PROFESSOR OF COMPUTER SCIENCE, 1985-88, American Univ., Washington, DC.
ASSISTANT PROFESSOR OF COMPUTER SCIENCE, 1980-85, Elizabeth City State Univ.
ASSISTANT PROFESSOR OF MATHEMATICS, 1979-80, Norfolk State University, Norfolk, VA.
VISITING PROFESSOR OF MATHEMATICS, 1976-78, University of Kentucky, Lexington, KY.
INSTRUCTOR OF MATHEMATICS, 1972-76, Kentucky State University, Frankfort, KY.

PUBLICATIONS, HONORS AND AWARDS

Computer Visualization Session Chair, ACM/SIGCSE'94, Pheonix, AZ.
Consultant, Advance Placement Computer Science Reading, ETS, 1994.
Dissertation and Post-Doctoral Proposal Reviewer, AAUW Washington, DC 1993,94.
Proposal Reviewer, Department of Education, Javit Program, 1998.
Referee, Papers submitted to the 1993 ACM/SIGCSE conference
Presenter, 9th International Conference on Technology and Education Paris, France, 1992
Poster Session Chair, ACM-SIGCSE, 1991 Conference, San Antonio, TX.
Proposal Reviewer, NSF-Instrumentation and Laboratory Improvement Program, 1991.
Proposal Reviewer, NSF-Undergraduate Curriculum and Course Development Program, 1991.
Textbook Reviewer, Pascal: Understanding Programming and Problem Solving, Douglas Nance author, West Publishers, 3rd Ed. 1993, ISBN 0-314-93304-2.
Textbook Reviewer, Pascal Laboratory Manual, Carol W. Wilson author, West Publishers, 2nd Ed. 1993, ISBN 0-314-86529-2.
Author (Dr. Mary Gray, second author) of A Successful Intervention Program for High Ability Minority Students, School Science and Mathematics Journal, April 1990.

RESEARCH ACTIVITIES--FUNDED

Dept, of Transportation 1993-4, $80,000
Office of Naval Research 1994-99, $1,216,000
Office of Naval Research 1990-94, $123,400
Sonia Kovalevsky High School Math Day'94 $5000
Egyptian CBE Seed Grant , 1994-95 $47,000
ACM/SIGGRAPH 1991, $5,000
NSA, 1991-93, $74,800
NSA, 1992-94, $165,900
ACM/SIGGRAPH 1994, $5,000
DOE "Successful Partnering with HBCU" Infrastructure, 1994-96, $1,000,000.00(pending)

1994 AASERT Summer Research Program
in
Parallel Processing and Computer Visualization

Signature Pages