REP

Form Approved
OMB No. 0704-0188

2

Public reporting burden for thi: ncluding the time for reviewing instructions, searching existing data sources, gathering and maintaining the n Send comments regarding this burden estimate or any other aspect of this collection of information, inclu Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arli perwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
March 1992

REPORT TYPE AND DATES COVERED
Technical June 1991 - March 92

4. TITLE AND SUBTITLE
The Application of Eulerian and Lagrangian Hydrodynamic Computer Codes to Desk-Top Personal Computers

5. FUNDING NUMBERS
C

6. AUTHOR(S)
Steven M. Buc

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
System Planning Corporation
1500 Wilson Blvd
Arlington, Virginia 22209

8. PERFORMING ORGANIZATION REPORT NUMBER
SPC 92-282-3

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Advanced Research Projects Agency (ARPA)
3701 N. Fairfax Drive
Arlington, Virginia 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Contract No. MDA972-90-C-0044

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Recent advancements in personal computer hardware indicate that sufficient memory, disk storage, and processing speed exist to make the operation of both Eulerian and Lagrangian hydrocodes practical, and considerably less expensive. A limited investigation was conducted to evaluate the ease with which several public domain codes could be adapted to run on 386/486 machines, and whether they would operate with sufficient accuracy and speed. Three codes were selected: 1) HELP, a multi-material Eulerian program for compressible fluid and elastic-plastic flows; 2) NIKE, an implicit finite-deformation, finite-element code; and 3) DYNA, an explicit finite element and finite difference code. All of these codes readily adapted to the personal computer hardware. Issues and code modification approaches are discussed. Example runs are presented.

14. SUBJECT TERMS
Eulerian, Lagrangian, personal computers, shaped charge, penetrators, hydrocodes, FORTRAN, examples

15. NUMBER OF PAGES
13

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
Unlimited

SPC 92-282-3

THE APPLICATION OF EULERIAN AND LAGRANGIAN HYDRODYNAMIC COMPUTER CODES TO DESK-TOP PERSONAL COMPUTERS

March 1992

Prepared for

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington, Virginia 22203-1714

Prepared by

Steven M. Buc

System Planning Corporation
1500 Wilson Blvd
Arlington, Virginia 22209

CONTENTS

1. BACKGROUND[1]

Typical Eulerian and Lagrangian hydrodynamic computer models are computationally intensive and require considerable computer memory and data storage requirements. In the past, the use of these models has been restricted to very fast and powerful computers such as the CRAY line of machines. Recent developments in the personal computer field, however, indicate that sufficient memory, disk storage, and processing speed exist to make the operation of these hydrocodes practical, and considerably less expensive. As a result, a limited investigation was conducted to evaluate the ease with which several public domain codes could be adapted to run on 386/486 machines, and whether they would operate with sufficient accuracy and speed. For this study, three codes were selected: 1) HELP, a multi-material Eulerian program for compressible fluid and elastic-plastic flows in space and time; 2) NIKE, an implicit, finite-deformation, finite-element code for analyzing the static and dynamic response of solids; and 3) DYNA, an explicit finite element and finite difference code. For all of these examples, only the two dimensional versions were converted, since it was felt that the added complexity of converting the three dimensional versions would simply be an issue of more computer memory and processing time.

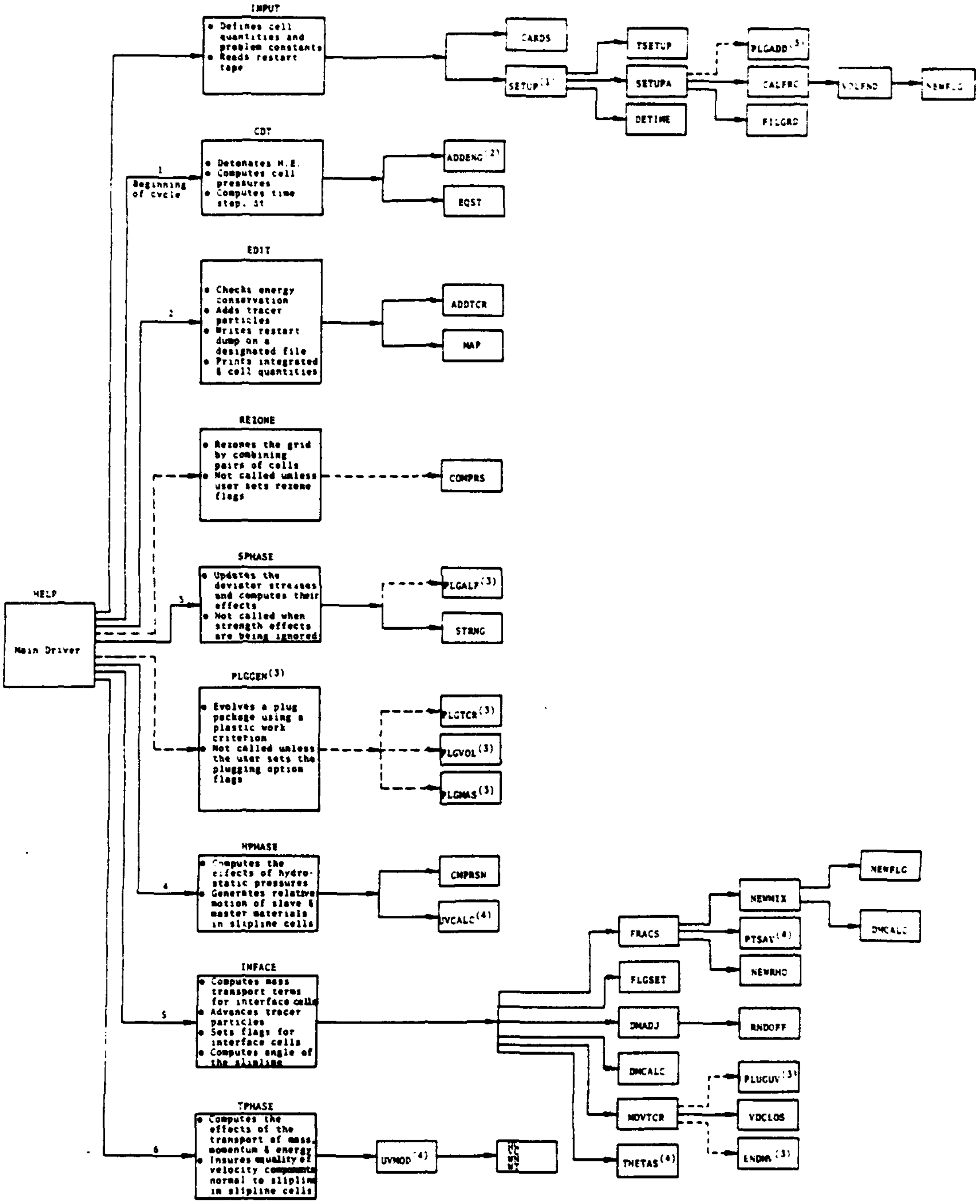
[1] The author wishes to acknowledge the invaluable assistance and participation of Lawrence Livermore National Laboratory and the Systems, Science and Software Corporation; in particular, Richard Herrmann, Robert Sedgwick, and George Hiler.

2. ADAPTING THE HELP CODE TO THE 386/486 PC

The HELP code was originally developed and improved by the Systems, Science and Software Corporation (S-Cubed) in the early 1970's under contract to the U.S. Army Ballistics Research Laboratory and the U.S. Air Force Armament Laboratory. S-Cubed graciously provided the FORTRAN listing of the main processor and those graphics routines which they did not consider to be proprietary. The flow chart for the main routine is presented in Figure 2.1, and describes some of the features of this code. The basic code is only six hundred thousand bytes of FORTRAN. However, the many multi-dimensioned arrays, required to track several parameters within each Eulerian cell was expected to push the size of the executable code into the megabyte region. For this reason, an extended memory FORTRAN compiler was used so that the basic 640K limitation of these PC machines could be exceeded.

Several weeks were spent upgrading the original code, primarily with respect to format statements, and writing a suitable pre-processor to make data entry more convenient. The original code was also written with every variable declared to double precision. This presented a problem, when the converted code was first compiled. Its memory requirement exceeded the four megabytes of extended memory available on the 386 machine. Naturally, for this computer code to have utility, it should calculate with adequate precision, and allow a useful size Eulerian grid to be employed. The arrays were currently dimensioned to permit a grid of 80 by 300, or 24000 cells, which was considered a minimum requirement for usefulness. As a result, the double precision requirement was removed and an example run made to compare any inaccuracies with the example output provided by S-Cubed. Fortunately, even after 400 cycles of an example shaped charge calculation, numerical differences were only occurring within the 6th to 8th decimal place, depending on the variable in question, and they were not growing. The inherent single precision of the PC processor seemed adequate. After discussions with S-Cubed, it was concluded that should double precision be required for some future calculation, to maintain code efficiency, the single best variable to declare in double precision would be the cell pressure terms, since these drive all the calculations.[2]

Figure 2.1
General Flow Diagram of the HELP Code


INPUT
• Defines cell quantities and problem constants
• Reads restart tape
CARDS
SETUP(1)
TSETUP
SETUPA
DETIME
PLGADD(3)
CALFRC
FILGRD
VOLFND
NEWFLG
CDT
1
Beginning of cycle
• Detonates H.E.
• Computes cell pressures
• Computes time step. Δt
ADDENG(2)
EQST
EDIT
2
• Checks energy conservation
• Adds tracer particles
• Writes restart dump on a designated file
• Prints integrated & cell quantities
ADDTCR
MAP
REZONE
• Rezones the grid by combining pairs of cells
• Not called unless user sets rezone flags
COMPRS
HELP
Main Driver
SPHASE
3
• Updates the deviator stresses and computes their effects
• Not called when strength effects are being ignored
PLGALF(3)
STRNG
PLGGEN(3)
• Evolves a plug package using a plastic work criterion
• Not called unless the user sets the plugging option flags
PLGTCR(3)
PLGVOL(3)
PLGMAS(3)
HPHASE
4
• Computes the effects of hydro-static pressures
• Generates relative motion of slave & master materials in slipline cells
CMPRSN
UVCALC(4)
INFACE
5
• Computes mass transport terms for interface cells
• Advances tracer particles
• Sets flags for interface cells
• Computes angle of the slipline
FRACS
FLGSET
DMADJ
DMCALC
MOVTCR
THETAS(4)
NEWMIX
PTSAV(4)
NEWRHO
NEWFLG
DMCALC
RNDOFF
PLUGUV(3)
VDCLOS
ENDMA(3)
TPHASE
6
• Computes the effects of the transport of mass, momentum & energy
• Insures equality of velocity components normal to slipline in slipline cells
UVMOD(4)
HC
VC
WN
WT

1) Called only when generating problem on cycle 0.
2) Called only when detonating a high explosive.
3) Called only when using the plugging model.
4) Called only when using sliplines.

[2] Richard Herrmann

To display the HELP calculation outputs graphically, a unique post processor routine was written, since the graphics commands provided by S-Cubed relied on commercial software, which SPC did not have. Figures 2.2 through 2.6 show the configuration output for an example shaped charge calculation.

Figure 2.2
Example Model with Eulerian Grid Overlay

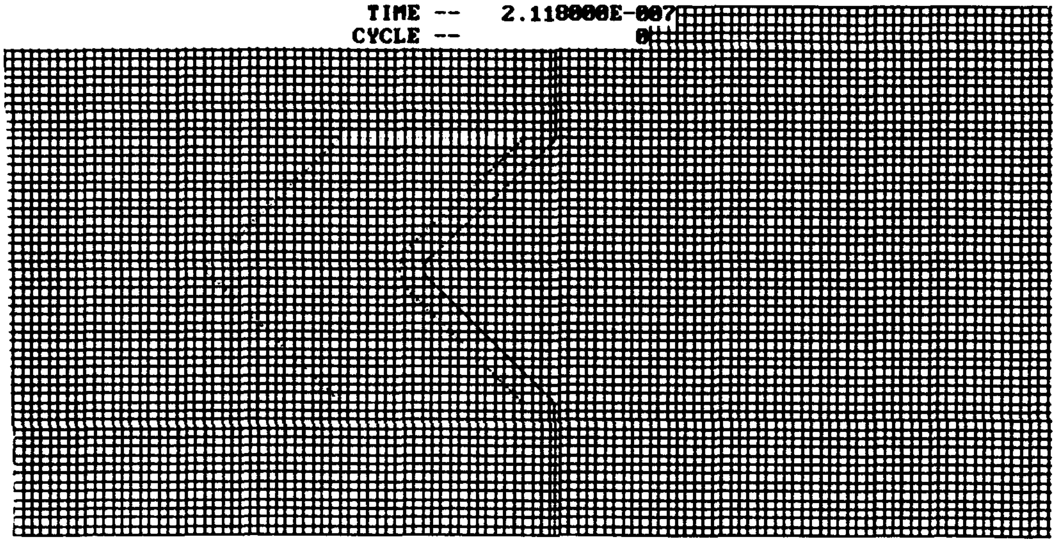


Figure 2.3
Example Shaped Charge Model at Cycle Zero

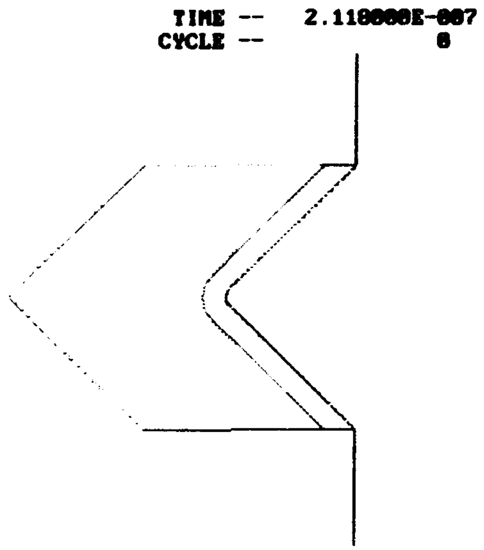


Figure 2.4
Example Shaped Charge Model at Cycle 50

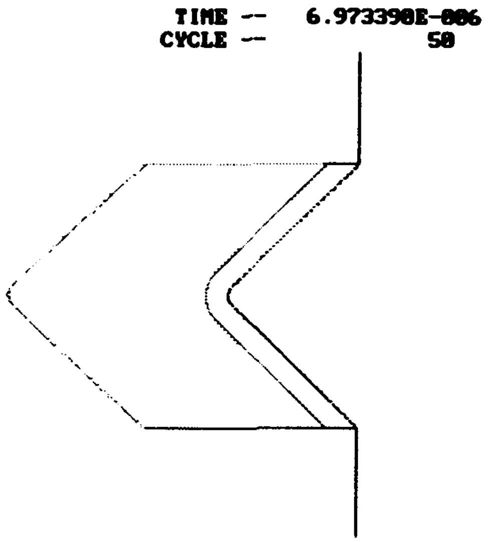


Figure 2.5
Example Shaped Charge Model at Cycle 200

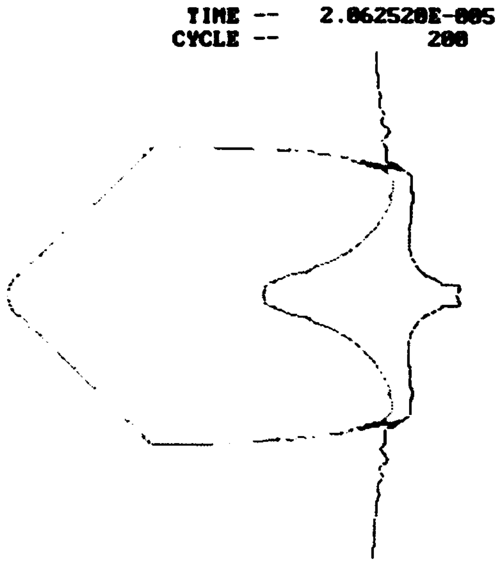
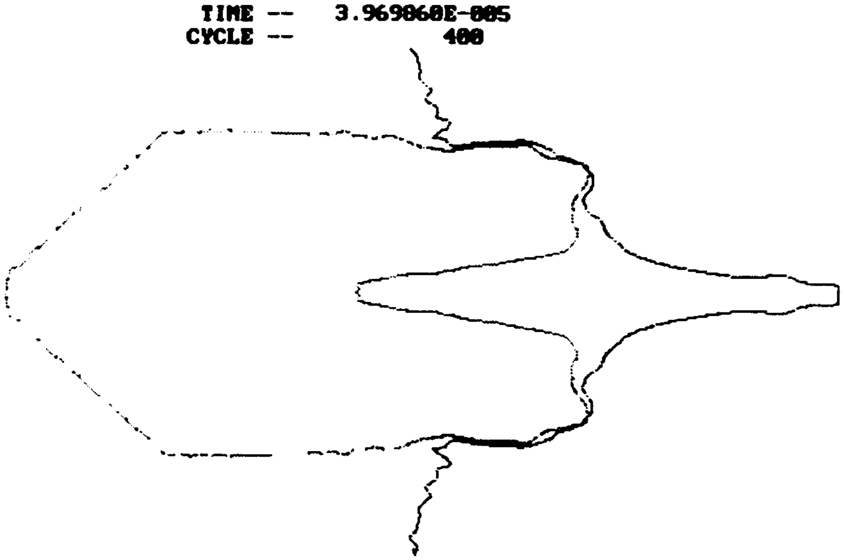


Figure 2.6
Example Shaped Charge Model at Cycle 400



This example shaped charge calculation runs in about one hour on the 386 SX-16 computer it was originally written on. However, it was benchmarked on a 486 DX-50 and ran in ten minutes. Currently, a fully equipped 486 DX2-66 is not a very expensive machine to buy, under $2000, and would decrease run time that much more. In addition, most machines can be expanded to accommodate up to 16 or 32 megabytes of RAM and can mount hard drives up to a gigabyte. This offers the possibility of going to full double precision with the code, and increasing the Eulerian grid considerably.

3. ADAPTING NIKE AND DYNA TO THE 486 PC

Following this early success with the HELP code, it was decided to test the PC hardware and code modification requirements of the NIKE and DYNA software. Lawrence Livermore National Laboratory (LLNL) graciously provided a VAX FORTRAN version on magnetic tape of the main routines, the pre and post processors, MAZE and ORION, respectively, and several example calculations. After transferring the FORTRAN to floppy disks, it was observed that the FORTRAN listings for both main routines were in the one to three megabyte range, so the 486 DX-50 with 8 megabytes of extended memory RAM was immediately selected as the baseline hardware.

In summary, VAX FORTRAN is very friendly to the DOS environment. For each of the four software packages -- NIKE, DYNA, MAZE, and ORION -- perhaps ten to twenty lines of code had to be modified in each to get them configured for the PC. These lines involved format statements and input and output file declarations. Within a week they were all compiled on the 486.

Getting ORION fully operable, however, took a little more time. This post processor is fully graphics oriented. Fortunately, LLNL wrote the graphics routines as simple primitives, as referenced in their device independent graphics library. In other words, all pen strokes are called out in the configuration and contour plotting and graphing. All that was needed was to incorporate the basic graphics routines allowed by the FORTRAN compiler, of which there are four -- initialize graphics/text mode, select pen color, move pen, draw line. All complicated graphics are created with these simple commands.

However, the device independent graphics library does not reduce lettering to its basic primitives. In other words, the code still calls a graphics driver to write text and numbers on the screen. To get around this, the basic pen strokes were normalized for all twenty-six letters, ten numbers, and several mathematical symbols, to allow lettering to be performed with the basic graphics commands in the FORTRAN compiler. A scale factor converts the normalized pen strokes into screen coordinates. If one does not want to develop text pen stroke commands, one should obtain graphics drivers compatible with the extended memory compiler in use, of which there are many. Figures 3.1 through 3.4 show an example NIKE calculation of a Taylor Anvil test. Figures 3.5 and 3.6 shows the early stages of an EFP warhead detonation as modeled by DYNA. In the EFP model the explosive is detonated in the bottom left corner, and one sees the bottom liner beginning to expand slightly.

Figure 3.1

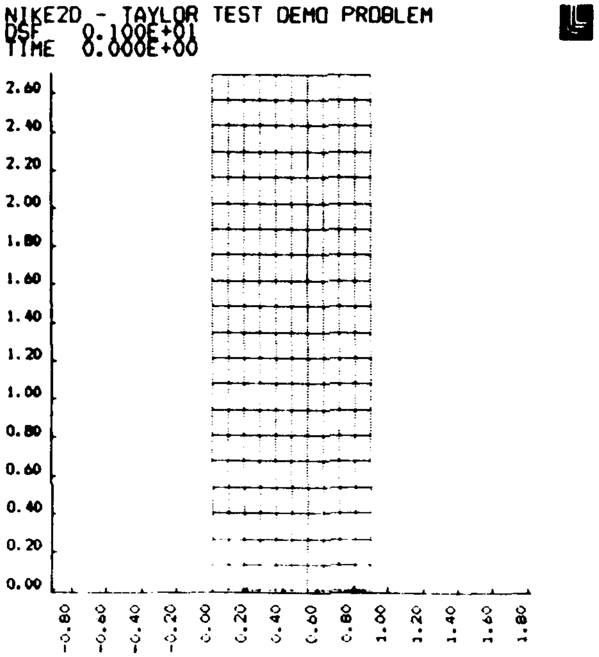


Figure 3.2

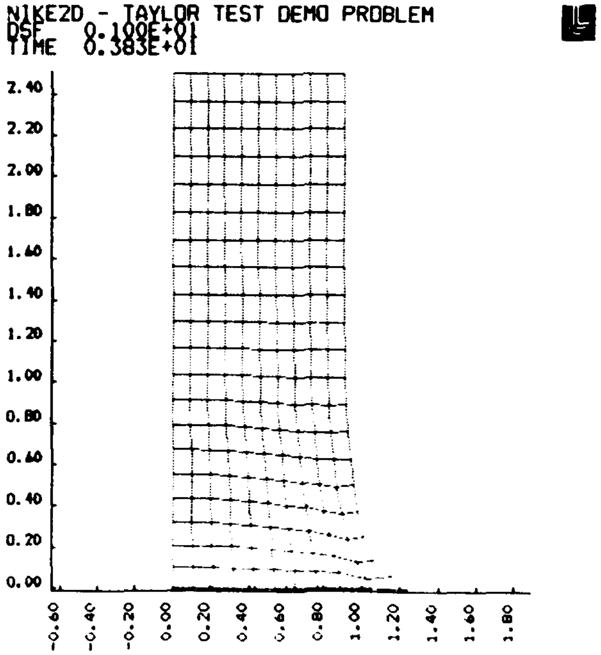


Figure 3.3

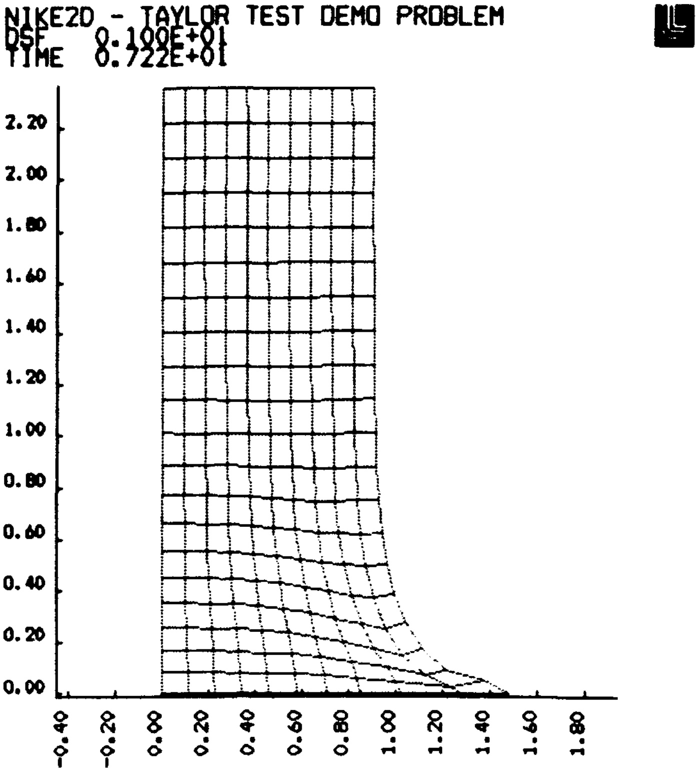


Figure 3.4

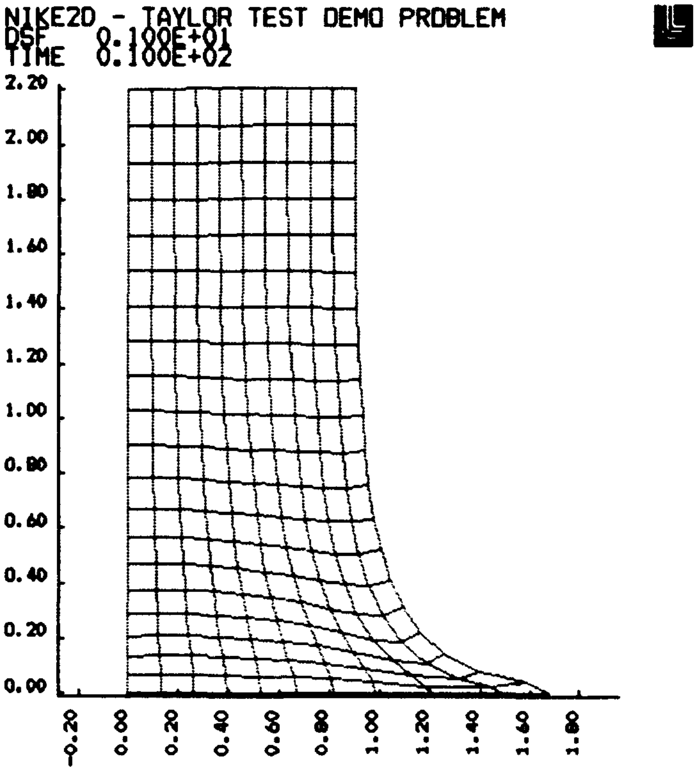


Figure 3.5

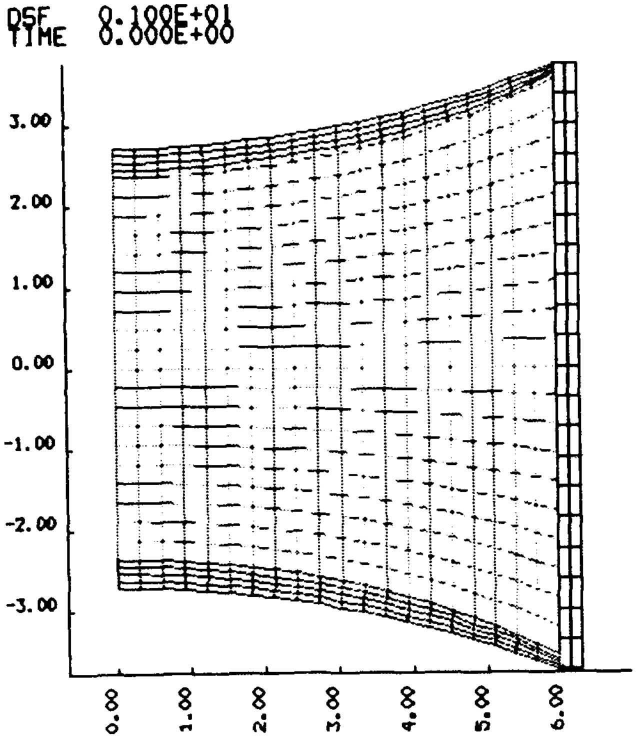
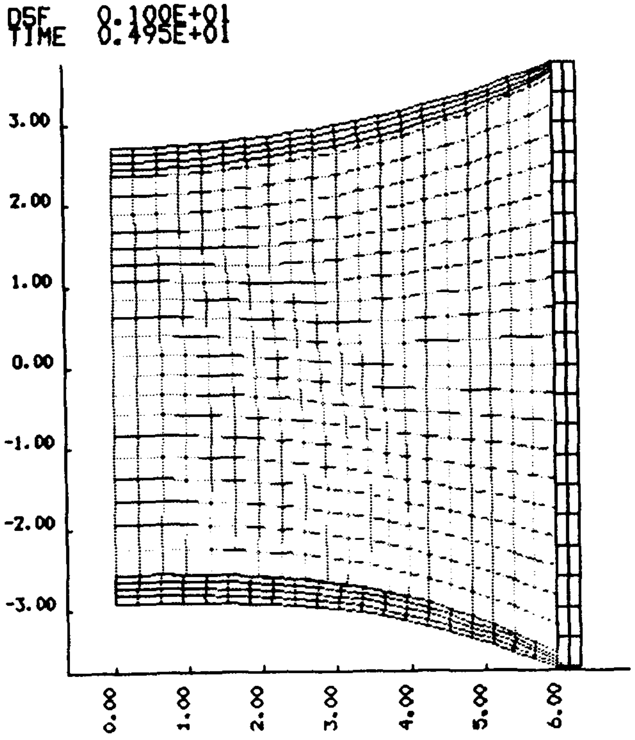


Figure 3.6



4. SUMMARY AND CONCLUSIONS

This brief computer coding endeavour has demonstrated the power, versatility, and cost effectiveness of desk-top personal computers in the field of explosives and dynamic material analyses. Those with the need to reduce the cost of computer hardware and the expense of performing hydrocode calculations are encouraged to investigate adapting their hydrocode models to 486 machines. This report has shown that both Eulerian and Lagrangian codes, in particular the HELP, NIKE and DYNA software packages, can operate efficiently and accurately on a personal computer. The author suspects that similar results will be obtained with other hydrocodes available today, such as MESA and EPIC.

REFERENCES

1. L. Hageman, D. Wilkins, R. Sedgwick, J. Waddell, *HELP: A Multi-material Eulerian Program for Compressible Fluid and Elastic-Plastic Flows in Two Space Dimensions and Time.* System, Science and Software, LaJolla, California. 1975

2. J. Hallquist, *ORION: An Interactive Post-Processor for the Analysis CODES NIKE2D, DYNA2D, and TACO2D.* University of California, Lawrence Livermore National Laboratory. 1985.

3. J. Hallquist, *NIKE2D: An Implicit, Finite-Deformation, Finite-Element Code for Analyzing the Static and Dynamic Response of Two-Dimensional Solids.* University of California, Lawrence Livermore National Laboratory. 1979.

4. J. Hallquist, *DYNA2D: An Explicit Finite Element and Finite Difference Code for Axisymmetric and Plane Strain Calculations.* University of California, Lawrence Livermore National Laboratory. 1988.

5. J. Hallquist, *MAZE: An Input Generator for DYNA2D and NIKE2D.* University of California, Lawrence Livermore National Laboratory. 1983.

6. H. Brand, *Documentation for DIGLIB Device Independent Graphics Library.* University of California, Lawrence Livermore National Laboratory. 1984.