#154 5/20/80

DR. 1202

ORNL/CSD-54

UCC-ND

NUCLEAR DIVISION

UNION CARBIDE

MASTER

A Guide to Making Time-Lapse Graphics Using the Facilities of the National Magnetic Fusion Energy Computing Center

John K. Munro, Jr.

OPERATED BY
UNION CARBIDE CORPORATION
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

ORNL/CSD-54
Distribution Category UC-32

A GUIDE TO MAKING TIME-LAPSE GRAPHICS USING THE FACILITIES OF THE NATIONAL MAGNETIC FUSION ENERGY COMPUTING CENTER

John K. Munro, Jr.

Sponsor: J. T. Hogan

Date Published - May 1980

COMPUTER SCIENCES DIVISION
at
Oak Ridge National Laboratory
Post Office Box X
Oak Ridge, Tennessee 37830

Union Carbide Corporation, Nuclear Division
Operating the
Oak Ridge Gaseous Diffusion Plant . Paducah Gaseous Diffusion Plant
Oak Ridge Y-12 Plant . Oak Ridge National Laboratory
under Contract No. W-7405-eng-26
for the
DEPARTMENT OF ENERGY

TABLE OF CONTENTS

FIGURES

TABLES

ACKNOWLEDGMENTS

The author wishes to acknowledge the patience and support of J. T. Hogan who wanted to use computer-generated movies to display results from the Oak Ridge Tokamak Transport Code and the criticisms and suggestions of P. W. Gaffney who read the text from the point of view of a person who plans to make some computer movies himself.

This work was funded by the Fusion Energy Division at the Oak Ridge National Laboratory.

A GUIDE TO MAKING TIME-LAPSE GRAPHICS USING THE FACILITIES OF THE NATIONAL MAGNETIC FUSION ENERGY COMPUTING CENTER

John K. Munro, Jr.

ABSTRACT

The advent of large, fast computers has opened the way to modeling more complex physical processes and to handling very large quantities of experimental data. The amount of information that can be processed in a short period of time is so great that use of graphical displays assumes greater importance as a means of displaying this information. Information from dynamical processes can be displayed conveniently using animated graphics. This guide presents the basic techniques for generating black and white animated graphics, considering the aesthetic, mechanical, and the computational problems. The guide is intended for use by someone who wants to make movies on the National Magnetic Fusion Energy Computing Center (NMFECC) CDC-7600. Problems encountered by a geographically remote user are given particular attention. Detailed information is given that will allow a remote user to do some file checking and diagnosis before giving graphics files to the system for processing into film in order to spot problems without having to wait for film to be delivered. Source listings of some useful software are given in appendices along with descriptions of how to use it.

KEY WORDS: Computer movies, Animated graphics, Post-processing, Color movies, 3D plots, File handling procedures, Diagnostics, Graphics commands, Text frames, FR80 format, DD80 format

---

1. INTRODUCTION

1.1. PURPOSE OF THE MOVIEMAKER'S GUIDE

The advent of large, fast computers has brought within reach the possibility of modeling complex dynamical processes such as transport of many different particles and radiation in plasmas and growth rates of various plasma instabilities. Improvements in plasma diagnostic techniques and development of rapid data acquisition systems have given plasma physicists access to enormous amounts of data. The use of a few frames of graphics to portray many effects which occur in complex dynamical systems can be a handicap to visualizing important time-dependent processes. A computer-generated movie permits a person to look at a large number of graphics in sequence and to get a better visual impression of dynamical effects in the system. Large amounts of information can be assimilated very quickly this way. The use of computers to make movies is still a relatively new activity without much written documentation. This guide is an attempt to summarize many bits and pieces of information which are useful for generating movies on the National Magnetic Fusion Energy Computing Center (NMFECC) system at the Lawrence Livermore Laboratory (LLL) in California. Movie-making procedures described here were developed in the course of writing and using a program [Munro 1979] currently being used to display results from the Oak Ridge Tokamak Transport Code (ORTTC).

1.2. SUBJECTS COVERED IN THE GUIDE

Section 2 contains an introduction to the basic movie-making process and is intended to acquaint the user with the terminology and the procedures to be described in more detail in later sections. Section 3 contains information needed for getting ideas and presents suggestions for software development. Section 4 gives descriptions of diagnostic and debugging procedures which may help when one is faced with the tough problems that never fail to arise when making a computer movie.

1.3. ARRANGEMENT OF SUBJECT MATTER

Generally the most widely applicable and useful information will be presented first with more detailed and specialized information appearing later or placed in appendices. Most of the reference material cited is available in the form of on-line documentation and can be obtained by executing the routine DOCUMENT [DOCUMENT Computer

2. THE MOVIE-MAKING PROCESS

2.1. BASIC ELEMENTS

2.1.1. Summary of the Complete Process

In simplest terms one needs to plan carefully what one wants to show, then write a program to generate a sequence of many graphs. The program writes the graphics commands in a family of files which is written onto a magnetic tape. The tape is carried to an FR80 or DICOMED film device and mounted on it for processing into film. The segments of exposed film which are produced are developed and routed to the appropriate user division for distribution into mail slots and subsequent mailing. The film is usually wound on three-inch plastic reels and is mailed with other output several times a week to the various user service centers (USC's). For USC's located in the eastern part of the U.S., turn-around time for getting film is at best one week. Once all the segments of the movie have arrived, they must be carefully inspected for blank or incorrectly drawn frames. If the segments are not all free of problems, then another attempt must be made to get correctly generated film. Once all the segments have been correctly produced, one can splice them together or have any special processing done which one desires in order to enhance the visual presentation.

There are many individual steps to the process summarized above and at least a half dozen different people who have a chance to make an error that will prevent the film from arriving properly prepared. One's efficiency in making movies depends on understanding all the steps of this process and what happens or can go wrong at each step. A suggested logical procedure for making computer movies is summarized in Fig. 2.1. The following sections discuss the details and reasons for such a procedure.

2.1.2. Planning a Movie

Summary. Careful planning is particularly important for developing efficient procedures for producing movies and for making a clear visual impression of the information which one wants to communicate. Planning efficient procedures for producing movies comes largely through experience, though in sections to follow, suggestions will be given to expedite this process. To a degree, learning how to communicate effectively through the visual medium of animated graphics also comes through experience. However, there are obvious elements to consider for which experience is not necessary. These elements include proper allowance for framing rates, length of a movie, arrangement of graphics or text in a frame, use of narration, and some special visual effects. Table 2.1 provides a summary of data required for planning and making a movie. Table 2.2 summarizes basic principles to observe.

Framing rates and movie length. The standard movie projector framing rate is considered to be 20 frames per second. The framing rate will vary between 16 and 24 frames per second. Individual frames can be resolved at a framing rate of 8 per second. This is useful to know if a variable rate, stop-action projector is available. These numbers are the most useful guidelines for planning the number of frames required to produce a movie of a desired duration or in deciding what time scales to consider for displaying the evolution of a process. A recommended length for a movie for use at a technical talk is 2-3 minutes. A 5-minute movie can become a major project and require enormous amounts of computational and editing time. Most movies prepared for technical talks will probably not involve adding a sound track or narration. If narration is used, the movie should be designed to stand alone as an independent unit.

PLAN MOVIE

Write specifications: what goes into movie, order of presentation, type of graphics to be used, special effects or processing desired.

Choose operating mode: is movie to be generated by post-processing an intermediate file or directly at the time of numerical solution of the problem?

PLAN PROGRAM TO GENERATE MOVIE

Design logical structure: build structure around modular form with modules defined in terms of specific tasks or functions.

WRITE GRAPHICS MODULE

Generate individual frames then re-evaluate whether movie is necessary.

WRITE MOVIE PROGRAM

Test program logic before incorporating graphics module.

Incorporate graphics module and make a "quick movie" to use for diagnostic purposes. "Quick movie" need have only 10-20 frames; get hardcopy on Versatec printer/plotter.

Check out all reasonable combinations of special features in "quick movie" mode.

GENERATE MOVIE AND GIVE GRAPHICS FILES TO SYSTEM

Check graphics files with TRIX AC and FR80PLOT for completeness and correct generation.

Give graphics files to system, notify NMFECC personnel, save graphics files.

Wait patiently.

Figure 2.1. Summary and logical flow of movie-making procedure

Table 2.1. Summary of Data Required for Planning and Making a Movie

Description	Value(s)
Framing rates	**$(\text{sec})^{-1}$**
Range of values for most projectors	16-24
Average value for determining movie length	20
Limit for resolving individual frames distinctly	~8
Computation times (typical) required to generate different types of movies	**(min)**
Title and several sets of text (200 frames each)	<1
Three-minute movie using 2-D plots	5-10
Three-minute movie using 3-D plots	5-60
Volume of information: limits and recommended sizes	**(words, decimal)**
Maximum volume for magnetic tapes used for processing graphics files	1,920,000
Default size for FR80 files produced by TV80 software	2^{15}(=32,768)
Recommended range of size for graphics files	2^{17}(=131,072) -2^{18}(=262,144)
Range of cost (typical) for modest amount of commercial processing	**(dollars per foot)**
Remove every other frame and introduce tinting in making a copy	0.50 - 1.00

Table 2.2. Brief Summary of Essential Principles to Observe in Planning and Producing a Computer Movie

PRINCIPLES TO OBSERVE WHEN PLANNING A COMPUTER MOVIE

Recommended movie length for a technical talk: 2-3 minutes

Assume viewer will direct attention to center of screen unless directed explicitly otherwise.

<u>Plan use of</u>

Visual space: don't put too much information on a frame

Timing of action: fast enough to sustain interest but not so fast as to lose viewer

Color (especially when using several in combination): good colors are light blue, green, yellow, and orange, when frame background is black.

Text (limit lines to six per frame)

Minimum size determined by superimposing a 24 × 42 line grid on frame

<u>Avoid</u>

"Quad-plots"

Jerky movements

Abrupt visual transitions

text to graphics
low light levels to high and color changes

Sudden scale changes of any change a viewer might not anticipate

Dark colors on a black frame background: violet, blue, red, magenta

Computation time. The computation time required to generate a movie of 1-3 minutes' duration is typically a few minutes to an hour. Two-dimensional plots are generated much more rapidly than three-dimensional ones. Plotting time for three-dimensional plots can be very long if the surfaces being displayed have a lot of structure and therefore require a fine mesh to show the structure. Surfaces with a lot of structure can be displayed more effectively by varying the viewing angle. The viewing angle can be changed continuously in the movie or if it is to remain fixed, can be found by trial and error by making a series of brief movies to select the best angle. Movies lasting ≥ 5 minutes typically will require hours of computing time since more problems generally arise in a bigger undertaking such as this. Computation time varies considerably depending on the degree to which information on successive frames remains the same and can be generated once and saved in buffers for insertion in successive frames. Use of color likewise will affect computation time significantly, as well as the reliability of the results.

Use of visual space. Do not attempt to overwhelm the viewer with large amounts of information It is easy to make movies which are "too busy". The time and effort required to make a movie will tempt one to make efficient use of space on a frame, e.g., "quad plots". Resist such temptation. This is too much information to digest in a movie. Usually much more eye movement is required to view an image on a movie screen than is required to look at the same image on a sheet of 8 1/2" x 11" paper. The movement and additional concentration required to process a sequence of changing images require significant effort from a viewer. These facts are not always fully appreciated by the beginning movie maker.

Scaling plots. Scaling plots is more difficult to deal with in a movie than in a sequence of graphs on paper because viewing time and conditions are so different. Scale changes should be announced in a test segment of frames preceding the graphics sequence with the new scale since the viewer will most likely miss the change otherwise. Or the viewer may realize that the scale has changed, so he will attempt to locate the information regarding the new scale and perhaps miss the action shown by the animation. If scales are likely to change through many orders of magnitude in the course of an evolution such as is often the case with distribution functions, for example, use a semi-log scale for the axis used to indicate the function value. If the distribution function ranges over 10 orders of magnitude during the evolution of the process being shown, define range and scale to map the function values to fit into the active visual volume of the frame which must be accessible for the duration of the process being viewed.

Avoid jerky movements. Jerky movements occur if there are problems with frame registration or if the time evolution shown on the screen occurs too rapidly for the viewer to be able to fill in the gaps in the animation between successive frames. Also avoid sudden changes in intensity. Sudden changes of any sort require more effort to process by the viewer and decrease his ability to absorb the information presented to him.

Use of time scales. Avoid long periods of little or no action. The value of animated graphics in a technical presentation lies in creating a strong visual impression of the temporal relationships and changes which occur in a dynamical process. Animated graphics can be used to emphasize the interplay of several competing processes which are governed by very different characteristic response times. Considerable experimentation may be required to generate a movie which effectively communicates these relationships. If one anticipates problems in adjusting animation time scales, one should generate a family of output files with results from a computation. This family of files can later be post-processed by a movie generator program as many times as necessary to determine visually appropriate time scales. The family structure also permits processing output in small amounts to avoid using unnecessarily large amounts of computational resources. Animation time scales can be adjusted somewhat by skipping frames or repeating them. However, this method of adjustment should be reserved for "fine tuning".

Use of color. Color is difficult to use effectively because colors are always displayed against a black background. Hence, lines are going to be more difficult to see than shaded areas. Dark colors like blue and violet are very difficult to see against a black background. Red is very easy to see, but also presents a visual spillover effect, i.e., a red line will appear to be much broader than it actually is. This happens because of the visual contrast between bright red and black. Good colors for lines against a black background are light blue, green, yellow, and orange. While red sometimes does not work for drawing lines, it can work well in titles if the character size is large enough that the visual spillover effect does not cause annoying loss of resolution. For good results with color, it is important to experiment with a few sample frames. This requires more time and effort, but should give much more gratifying results in the final movie.

Generating frames of text. Titles and other text information can be produced by graphics calls in the program that generates the graphs, by running a separate program written solely for generating segments of text, or by hand lettering on drawing paper and photographing the result. The first two options have the advantage of low cost and speed, but few options for letter size and style. The last option allows for limitless variation of style, color, size of lettering, and embellishment of the background of the frame, but costs more and requires more time and more processing steps to produce a segment of 15 mm film. Once the frame has been drawn, it has to be photographed using 35 mm film; then this film must be used to make a sequence of copies on 16 mm film stock for the desired viewing duration. For most purposes, text frames produced by graphics software should be quite acceptable.

Visual presentation of text. Text should not be crowded into a frame. Text should be centered in the part of the frame where the viewer's attention is expected to be. Similarly, text may be shifted to direct the viewer's attention to some part of the frame if he should be looking away from the center. Text, as a rule, should not exceed six lines. More text than this gives a visual impression of a busy frame. As with crowding a frame with too much graphical information, the use of too much text requires a lot of eye movement and hence a lot of effort from the viewer to absorb information. As much as possible, the viewer's attention should be kept focused on one area of the frame.

Text character size. Large characters always require less effort to read, so the larger the text, the better. A useful guide to the lower limit of character size may be determined by constructing a grid of equally spaced horizontal lines and equally spaced vertical lines so that the viewing frame is divided into 24 rows and 48 columns. The smallest character size should be large enough to fill one of these grid elements.

Transitions between segments. Transitions between different animation sequences are frequently helpful to allow for physiological optical readjustment time and to give the viewer time to adjust mentally to a new set of visual images. Current graphics software provides no convenient method for generating a sequence of images with variable intensity to produce a fade-in or fade-out effect. Therefore, visual transitions must be provided for by inserting a sequence of 5-10 blank frames between one type of an image (e.g., text) and a different type which follows (e.g., a 3-D plot). Professional fading sequences can be generated on request by a commercial film processor, but the turn-around time for this will typically be 2-4 weeks. Such effects are probably not worth the extra expense and trouble unless other special effects or processing are also being done. Graphics software with at least 64 grey levels is probably necessary to provide adequate fade-in or fade-out.

2.1.3. Organizing Data Handling and Graphics Software

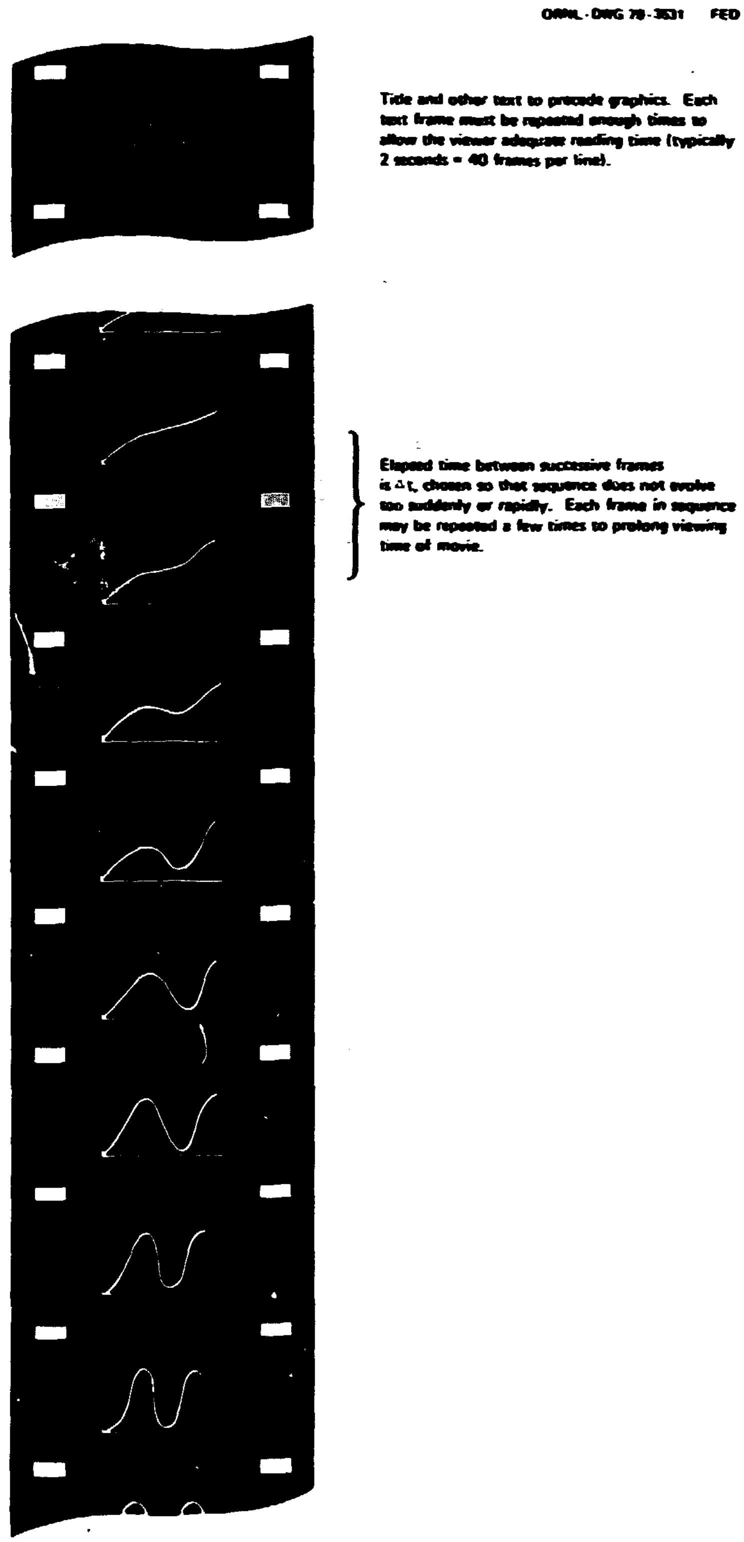
Summary. Making movies at first appears to be a simple extension of the process of generating a single graph by merely repeating the process the desired number of times. Conceptually, this is certainly true. Implementing this extension may not be so trivial. Careful thought to organization of the various tasks, software development, and data files is well worth the investment of time and effort. Making movies with the NMFECC network facilities can require considerable record keeping in order to keep all the steps of the process and the problems associated with each step sorted out so they can be handled with a minimum of confusion. This section gives some suggestions for organizing one's efforts.

Write software to produce a single frame. Debugging difficulties are compounded rapidly if one attempts to write the sections of code to generate an individual frame at the same time one is trying to arrange the logic to repeat frames. Write the section of coding required to produce a single frame and make sure this is working properly before adding any logic to handle the animation. Also try to keep parts of a program which generate graphics collected together in subroutines separate from the parts of the program used to do numerical computations. This will facilitate the transition to more elaborate operations for making movies later. The next step is then to generate a sequence of 5-10 different frames. After this has been done, one should consider again whether a movie is really necessary to show what one needs to see.

Make a "quick movie". Having arrived at the stage of generating a sequence of different frames, one can then add the logic to repeat a given frame any desired number of times. This capability will then permit one to make a simple movie by repeating each frame enough times to provide for whatever viewing time is desired in moving through the sequence of frames. Titles or other text can be generated along with the graphics sequences or they can be generated by a separate program and the film segments spliced in proper order later. A program and procedures for generating text are given in Appendix A. Each different frame of graphics should have information identifying its position in the sequence of frames. Sequencing information allows one to determine quickly the order in which film segments must be spliced and whether all the film segments have been generated correctly. One may also encounter a situation where a segment is lost or delayed in the shipping process. Any problems such as these can be more easily monitored when frame sequence number information is readily available. A simple movie sequence is shown schematically in Fig. 2.2.

Use of post-processing. Making a "quick movie" can become a costly job if an enormous number of computations must be made between each different frame of graphics. Hence one should think in terms of writing results of computations in a file for later processing if one's program requires more than a couple of minutes of execution time. This is particularly important to consider if the graphics calls and movie logic are beyond the most elementary level, since the movie making software can often be difficult to debug. In fact, it is better, when doing a large computation, to write out several hundred data sets in a family of files for later processing into a movie. Most of the movie software development can then be done using only the first member of this family of files. When the software seems to be working satisfactorily, one can then process the entire family of files.

Tape lengths, file sizes and file names. An enormous number of graphics commands are required to produce a movie lasting a few minutes. Therefore file sizes should be as large as possible without



2.2. Simple movie sequence to illustrate the effect of animation in an exaggerated, schematic fashion.

running into hardware limitations or system reliability related problems. The magnetic tapes (2000 feet long) onto which disk files are transferred for later processing on an FR80 or DICOMED film processor hold 1,920,000 decimal (60-bit) words [TV80 computer documentation (on-line)]. The FR80 file naming convention F0AAABBBBB, F1AAABBBBB, ..., FXAAABBBBB allows for a maximum of 11 files in this type of a family so a good file size lies between 2^{17}(=131072) and 2^{18}(=262144) words. File sizes can be specified by using the TV80 routine GFSIZE. The default file size is 2^{15}(=32,768) words. In the file naming convention, AAA identifies the type of film and camera to be used for processing the file; BBBBB is the string of the 5 right-most characters from the first argument in the calling parameter list of the TV80 routine KEEP80. This is the only part of the file name over which one has control to name the family of graphics output files. If one generates a large number of output files it may be necessary to use a sequence of family names. A sequence is easily built on the name of the first family by adding the integer value 1 to the name variable, e.g., NAMVAR=NAMVAR+1. This advances the right-most ASCII character in the name to the next character in the character table (Table 2.3). Frames of graphics generated by a large number of graphics commands frequently are subject to being split across magnetic tapes if the last member of a family of files written on a tape is not closed on a frame boundary, i.e., immediately after a frame advance command (generated by CALL FRAME). This produces partially plotted frames at the end or beginning of a segment of film. If this problem cannot be tolerated, one must devise a method to insure that a complete frame of graphics commands can be written before the end of a tape is

Table 2.3. Representations of Characters
(Internal and External)

(Characters are represented in the compiler as 6-bit quantities.)

Dec	6-bit Oct	Internal ASCII	Printer	HSP	TTY
0	00	space	space	space	space
1	01	exclaim	down-arrow	space	exclaim
2	02	"	#	eta	"
3	03	#	#	#	#
4	04	$	$	$	$
5	05	%	%	pi	%
6	06	&	↑	sigma	&
7	07	'	.ge.sign	tilde	'
8	10	(	(	(	(
9	11	)	)	)	)
10	12	*	*	*	*
11	13	+	+	+	+
12	14	:	,	,	,
13	15	-	-	-	-
14	16	.	.	.	.
15	17	/	/	/	/
16	20	0	0	0	0
17	21	1	1	1	1
18	22	2	2	2	2
19	23	3	3	3	3
20	24	4	4	4	4
21	25	5	5	5	5
22	26	6	6	6	6
23	27	7	7	7	7
24	30	8	8	8	8
25	31	9	9	9	9
26	32	:	:	:	:
27	[illegible]	;	:	delta	;
28	34	<	<	<	<
29	35	=	=	=	=
30	36	>	>	>	>
31	37	?	.le.sign	cycle	?
32	40	at-sign	half-arrow	rho	
33	41	a	a	a	a
34	42	b	b	b	b
35	43	c	c	c	c
36	44	d	d	d	d
37	45	e	e	e	e
38	46	f	f	f	f
39	47	g	g	g	g
40	50	h	h	h	h
41	51	i	i	i	i
42	52	j	j	j	j

43	53	k	k	k	k
44	54	l	l	l	l
45	55	m	m	m	m
46	56	n	n	n	n
47	57	o	o	o	o
48	60	p	p	p	p
49	61	q	q	q	q
50	62	r	r	r	r
51	63	s	s	s	s
52	64	t	t	t	t
53	65	u	u	u	u
54	66	v	v	v	v
55	67	w	w	w	w
56	70	x	x	x	x
57	71	y	y	y	y
58	72	z	z	z	z
59	73	{	{	{	{
60	74	¦	¦	¦	¦
61	75	}	}	}	}
62	76	up-arrow	up-arrow	↑	up-arrow
63	77	left-arrow	right-arrow	half-arrow	left-arrow

reached. One probably must learn the best way to handle this from experience with the particular type of pictures one is generating. One can try to adjust file size or limit the number of frames plotted for a given family of files.

Using a sequence of jobs. One should consider carefully whether to attempt to generate all the graphics files for a movie in one big job or whether to break the task into a sequence of jobs each requiring 10 minutes or less of computing time. One long job makes best use of computational resources and gets the job done quickly, but suffers if there is a continuous stream of jobs in the batch queue in a more favorable classification which allows them to run at a higher priority. A long job is also more vulnerable to system crashes. What a sequence of jobs lacks in computing efficiency is more than compensated for by being less vulnerable to machine crashes, by getting into a higher priority queue, and by permitting one to manage input and output files much more easily. Also, by breaking the task up into smaller units, if the batch control is set up properly and care taken to save the proper files (input and output files, controllee, drop file) at the end of a run, then one can drop back to any point in the sequence of jobs and continue without any difficulty. This is especially useful if something happens to the graphics output files for a particular job so that it has to be re-run. This way one does not have to go all the way back to the beginning of a long movie-making job to recover from a problem.

Organizing job control statements for a sequence of jobs. Probably the best way to stop a job so that it can be restarted in an orderly fashion is to call the ORDERLIB routine OFFMON [Carpenter et al., 1976]. This allows one to exercise enough control over the logic of the program and the job control statements so the program stops after a complete frame has been generated rather than at some arbitrary point in the middle of a frame. If one saves the input and output files and the drop file of the program that has just stopped executing, program execution can be resumed in a following job by retrieving from file storage the input and drop files and by executing the drop file. A time limit for program execution is specified in the job control statement file. This is done by allowing the job to run for a period of time after which a sense switch is turned on. The job is allowed to continue running for a second (shorter) time period sufficient to insure that a periodic testing of the state of the sense switch will have time to occur, at which time the program logic transfers control to the routine OFFMON. Routine OFFMON empties buffers and closes files so that any graphics output generated can immediately be given to User-2 (the system) for processing into film. Input files only need to be saved someplace one time since the drop file maintains a record counter to keep track of the last record read. The drop file for each job in the sequence must be saved in order to permit one to go back to begin re-running jobs at some arbitrary job in the sequence.

2.1.4. Developing Efficient Operating Procedures

Summary. Most of the sections so far have focused on areas which do not require knowledge of a specific computer or specific set of software. This section deals with problems which are more system- and software-dependent, such as file formats for graphics commands, use of frame sequence numbers, reading data from a family of input data files, and design of a basic program to serve as a general framework for making any movie.

File formats. The NMFECC system can generate graphics command files in dd8Ø or FR8Ø format. The dd8Ø format is older and more limited than the FR8Ø format and is being phased out of use. Since the FR8Ø format is the preferred and supported standard, program listings and special techniques presented in this guide will all use logic and software consistent with the FR8Ø format conventions. For people who must use the dd8Ø format, Appendix B contains a summary of useful procedures and information necessary for program debugging.

Repeating information from one frame to the next. Any kind of animation repeats a certain amount of information from one frame to the next without change. Some information may be repeated identically for many frames. With the proper software and program logic, much more efficient use of computing time can be obtained to generate graphics commands for the same information over and over if they can be stored in a buffer which is read as many times as needed. Procedures for storing groups of commands in buffers for repeated use are available from the TV8Ø software library. For FR8Ø formatted files, one uses the routine REFOCO [TV8Ø Computer Documentation (on-line)]. A simple example of its use is given in the listing of subroutine TXTFRM in

Appendix A. Calls to routine REFOCO can be used very effectively in nested DO-loops to repeat selected parts of different sequences of frames. The nesting of REFOCO calls is limited, however, to a depth of eight levels. The REFOCO routine has other, more annoying, limitations which are not mentioned in any of the documentation about it. One should use it to repeat a group of graphics commands only about 20 times. The reason for this becomes clear when one considers how the group of commands is processed by the film device. Each time a group of commands is repeated, the tape is rewound and that group read again. This process is subject to the limits of mechanical reliability of the tape and tape reader. Clearly one also cannot hope to process graphics commands which must be repeated if they span two tapes. So one has the same kind of problem here as discussed previously for the case of a large number of graphics commands required to generate a frame. One must be sure that all graphics commands to be repeated can be contained on a single tape. An additional problem occurs when a group of commands, intended to be repeated using REFOCO, spans two tapes: the family name of the files intended for the second tape is generated by the system. This causes problems when one runs in batch mode and wants to save graphics files in FILEM or desires to manipulate them in some fashion depending on one's assignment of file or family names.

Use of sequence numbers. Each frame in an animation sequence should have information in some form which uniquely identifies its location in the sequence since film is processed and mailed in segments without any attention given to the order in which the graphics commands files were generated. A segment of film will correspond to a sequence of files which was written on a particular magnetic tape. Several tapes will usually be required to transfer all the graphics files from disk storage to the film device. The segments of film which are produced must be spliced together to make the movie. Sequence numbers are essential to determine whether all the frames have been generated correctly and whether the film is complete.

Framework for post-processing software. Post-processing output from complex calculations has potential advantages beyond organizational efficiency in making movies. With experience and practice, one can develop a basic movie-making program and associated software which constitute a framework for making many movies of different things. Such programs can be used to call the primary subroutine in a module of subroutines written to produce a single frame of graphics of some desired function or table of numbers. Such movie-generating programs or drivers can be written to include procedures for operations common to any movie-making problem. Provision can be made at an appropriate place in the program logic to call the primary subroutine in the module of routines written to produce a frame of graphics. HLYWD [Munro 1979], a program used for a couple of years at ORNL, illustrates the use of the driver idea and many other features discussed in this guide.

Common operations required in making different movies. Some of the operations common to many movie-making efforts are

a) frame repetition a specified number of times

b) an option to generate titles or text before a graphics sequence

c) use of variables to specify input and output file names

d) provision for reading data from a sequence of members of a family of files

e) provision to interpolate between sets of data for successive times to slow or smooth motion during processes where quantities change rapidly.

f) rotations of three-dimensional plots through discrete angles (90^{o}, 180^{o}, 270^{o}) to obtain the best viewing direction or through a continuous series of angles between specified limits to display details of a complex surface.

This list will grow with more experience and different applications. The operation of reading data from a sequence of members of a family of files (item d in the list above) occurs frequently enough in operations with large amounts of data that it is handled by a separate subroutine ADVFIL (described in Appendix C).

Job restart provisions. Whether one chooses deliberately to break a movie-making task into a sequence of "10-minute" jobs or not, it is a good idea to have a job restart capability in one's program. The status of sense switch 1 should be checked at least once each minute, preferably after a frame has been completed. The logic should look as follows:

```
     CALL FRAME
     ...
     IF(SENSESWITCH,1) 3Ø, 2Ø
 3Ø  CALL OFFMON                                   This group of statements
     CALL KEEP8Ø (1,3)                             is required to open
     IG(36)=1ØHBOX A55                             graphics output files and
     IG(37)=1ØHJOHN K. MU                          reinitialize procedures
     IG(38)=1ØHNRO  MOVIE                          for writing in them. The
     CALL GFSIZE(3,3ØØØØØ)                         IG array is the main
     CALL FR8ØID(1ØH ORNL FED ,1,3)                (first) array in
     CALL KEEP8Ø(INAME,3)                          COMMON/GOBCOM/.
     CALL MAP(1.,1Ø24.,1.,1Ø24.,Ø.15,Ø.86,Ø.285,1.)
 2Ø  GO TO 21Ø - - - - - - - - - - - - - - - -     (Return to statements in
     RETURN                                        program to allow
     END                                           continuation of gener-
                                                   ating graphics output;
                                                   bottom of loop for
                                                   periodic testing of
                                                   sense switch 1)
```

The variable INAME in the calling parameter list in the second call to KEEP8Ø should contain the right-adjusted Hollerith string identifying the FR80 graphics output files (only the 5 right-most characters are used).

Job restart problems. Sometimes program execution may be terminated unexpectedly before the OFFMON routine can be called to terminate it in a graceful fashion. If this happens and one has managed to save output files and the drop file, check to see whether an output file with the name FYAAABBBBB is present. If so, the drop file can be executed, providing there has been no reading or writing of files on tapes, and it will continue writing in the file with the temporary name FYAAABBBBB until it is full. This file name will then be changed to the name it would have had in the sequence of files being generated prior to the unexpected termination. This procedure does not have to be used often, but it is useful to know about and can save time and wear on the person faced with such unexpected system behavior.

2.1.5. Generating Film Segments

Summary. After the files of FR80 commands have been generated, a surprising amount of effort is still often required to get the film. This section describes accounting and monitoring procedures for keeping track of what happens at as many steps of the remaining processes as one can. It also presents some suggestions for minimizing problems with the systems involved, both machine and human.

File naming conventions. The file handling utility routines and graphics software maintained by the NMFECC usually build families of files in either of two ways. The convention for graphics files in most cases is to use the second character from the left in a 10-character name as the sequence character and to let it range through the values

Ø,1,2,...,9,X. This convention accommodates 11 files. The other convention is to use the right-most character in the file name as the sequence character. If one does not want the system routines to generate a sequence of files but desires to do this oneself, one only needs to use the statement

NAME=NAME+1

where NAME is an integer variable name containing the name of the file member. The effect of this statement is to add 1 to the right-most bit in the word containing the variable NAME. The (6-bit) ASCII character sequence and its octal representation are given in Table 2.3.

Suggestions for naming graphics files. A useful and convenient naming convention for graphics files incorporates the string MMYNN where MM is a two-character month designator, Y the last digit of the year, and NN a sequence number. The FR8Ø file names are generated as described in Section 2.1.3. For 16 mm sprocketed black and white film, the string AAA in FSAAABBBBB, for example, would be P16. A large sequence of families of files can be generated if the sequence number NN ends in Ø and is advanced through each of the sequence of characters in Table 2.3 after 9-1Ø members of the possible 11 (built on the second character from the left) have been generated by the graphics software.

File monitoring. After graphics files have been generated, one can check to see how many there are by executing

FILES ALWITH. F

to get a listing, or

FILES COUNT. ALWITH. F

to get the number of files. This is useful for verifying whether all the files from a job were saved correctly in a subsequent execution of

FILEM [FILEM Computer Documentation (on-line)] and also whether they were all processed. After giving files to the system (User-2), one can check to determine whether they were correctly processed by executing

FILES USER2. ORGUSR. ALWITH. F

to get a listing of all file names beginning with F and the user number from which these files came.

File disposition for processing. The NMFECC graphics library TV80 [TV80 Computer Documentation (on-line)] provides two methods for disposing of graphics files for processing. The first uses only a call to the routine FR80ID and causes the graphics files to be given automatically to the system for processing. This method does not permit one to save graphics files, so is not recommended if one must wait more than a day for film to be returned. The second method uses calls to the routines KEEP80 and FR80ID as shown in Section 2.2.1. Graphics files are now saved on disk and can be saved in a FILEM directory before being given to the system. Graphics files should always be saved, at least until one is sure all the film segments have been correctly generated. One frequently will have to retrieve various members of a family of files and give them to the system (User-2) again for processing because of problems which inevitably occur. On occasion one may even have to go back and get a member a third or a fourth time to give to the system before the film segment is generated correctly. One may also encounter problems later in editing a film and have to generate segments over again. Graphics files are given to the system using the GIVE utility, for example

GIVE ENDWITH. BBBBB 2 K. END

where the option K. keeps a copy of the files in the user's disk area while

GIVE ALWITH. F 2 END

would not keep copies on disk. The ENDWITH. option may be less vulnerable to carelessness or oversight than the ALWITH. option, which one could use if one is sure the only file names beginning with F are FR80 graphics files. Graphics files which have been saved on disk can also be given to the system at the time they are being inspected on a graphics terminal using the GIVE command of FR80PLOT [see the FR80PLOT computer documentation, (on-line)].

Monitoring film delivery. Film frequently is delayed and sometimes lost in the mail, so one should always notify whoever is in charge (currently Max Allison) of mailing output from the NMFECC to the users, describing the amount of film one expects to receive. This procedure alerts someone at Livermore to start looking for film. This can be important because problems have been known to occur in the processing of the graphics tapes or in the development of the film. If the anticipated amount of film does not appear after a reasonable period of time (usually 2-3 days), the person alerted at Livermore can make inquiries there and at the same time alert one to the possibility that a problem has arisen. One should probably also request to be notified when film is mailed and by what means it is to be shipped in order to know when to expect it or to begin making inquiries if it does not arrive after a reasonable period of time (usually 2 weeks).

Keep a film log. Cataloging film segments as they are produced helps with subsequent monitoring of delivery time and in keeping a record of problems encountered when assemblying the segments at the time of splicing and editing. Such a record is particularly helpful when one generates many different segments and must wait frequently as long as two weeks to receive them. The order of receipt usually bears no resemblance to the order in which the graphics files were given to the system. A sample log sheet is shown in Fig. 2.3.

Figure 2.3. Sample Page for Logging Film Segments

NAMES		DATES			CONDITION OF FILM SEGMENT			
File(s) or Family	FILEM directory where files are stored	Graphics File(s) given to User-2	Film segment produced	Film segment received	Frame sequence Identification first	last	Film reel number	Comments

2.1.6. Splicing and Editing Film Segments

Summary. Splicing and editing are not usually difficult, but do take time. This section describes the procedures for splicing, editing, and caring for film.

Inspect film. Inspect each segment for missing or unplanned blank frames and incorrectly generated frames. Sometimes a sequence of several frames will appear to have been plotted on top of each other. Sometimes line intensities vary too much from one segment of film to the next, so a decision will have to be made whether to process the sequence of graphics files for a segment over again.

Add leader and trailer. A film should have at least 2-3 feet of leader and trailer to allow for threading film in a projector. If one anticipates showing the film many times, the leader should be a few feet longer to allow for problems of film jamming, wear, and tear which normally occur in the process of showing a movie. The trailer should be long enough to allow time for one to stop the projector when the end of the film is reached.

Film composition, design and care. Film is usually a cellulose acetate base with a photosensitive emulsion on one side. The side containing the emulsion has a matte finish; the side without emulsion is shiny. Since the layer of emulsion is exposed, it can be scratched easily. If the film is not treated with care, it can acquire a scratchy, abused appearance after only a few showings. Untreated film should hold up adequately for a dozen showings. A measure of protection for film can be obtained by waxing the emulsion side. Commercial services which repair and clean movies can treat film this way to protect and extend its life. A more rugged film stock made from

Mylar is available commercially, but should not be necessary unless one decides to make a copy of a master film for showing many times. The Mylar film is tough enough to bend the synchronous frame advance arm in some projectors if the film is not threaded properly.

Splice film. Two methods for splicing film are commonly used. The low budget method uses presstapes, short strips of adhesive tape (cellophane or Mylar) with sprocket holes which can be used with a film splicer containing pegs to align and hold the segments of film and allow proper positioning of the presstapes. A strip of presstape must be applied to each side of the joint for best results. The more expensive method requires a device for aligning the film segments and a warm metal plate under the area where the splice is made. A quick-drying cement is brushed onto the end of one segment and the end of the other segment is pressed down on top of the first with a slight overlap to join the two ends. The second method requires more skill than the first; but once the technique is mastered, it is quicker.

2.2. WRITING AND ASSEMBLING SOFTWARE

2.2.1. Basic Code Blocks to be Incorporated

Summary. Certain blocks of coding will be common to any movie-making software. This section represents an effort to collect and systematize some basic blocks of coding upon which one can build in the process of making one's own movies. These blocks include the sequence of statements needed to open graphics files and identify the segments of film, read input files one record at a time and check whether the record just read is the last one in the file, write sequence numbers on frames generated by the PLOT3D package, and provide job restart capabilities.

Initialize graphics files. The main routine used to generate graphics output should contain the following basic set of statements for FR80 files (see NMFECC online document TV80):

```
COMMON/GOBCOM/IG(38)
...
DATA MOVIE/5RFILMS/
...

CALL KEEP80(1,3)
IG(36)=10HBOX A55   J
IG(37)=10HOHN K. MUN
IG(38)=10HRO, JR.
CALL GFSIZE(3,300000)
CALL FR80ID(10H ORNL FED ,1,3)
CALL KEEP80(MOVIE,3)
```

The NMFECC document TV80 should be studied carefully to understand the use of the routines in this example and before making other than obvious name, box number, and division changes.

Read input files. The procedure for reading a line and testing for end of file is

```
 10 READ(5,200) (TITLE(I),I=1,3)
200 FORMAT(2A10,A1)
    IF(EOF,5)12,14
 12 CALL ADVFIL(LUN,JNAME,LEN,IOC)
    GO TO 10
 14 IF(TITLE(1) .EQ. TTLEOF) GO TO 499
    ...
    remainder of routine
```

Logic to test for end of file (EOF) immediately follows the READ statement so that if an attempt to read the array TITLE fails, control can be passed to the call which closes the current file being read and attempts to open the next file in the sequence, etc.

If a record or line has N items, i.e., a line from an array of numbers, one must compute a lower and an upper limit sufficient to read the items in a single line all at the same time, then attempt to read the line and test for end of file. This is illustrated as follows:

```
      ITEMS=7
      IL=1
      IU=ITEMS
  40  READ(5,230) (PSAVE(I),I=IL,IU)
 230  FORMAT(3X,7E11.4)
      IF(EOF,5)42,44
  42  CALL ADVFIL(LUN,JNAME,LEN,IOC)
      GO TO 40
  44  IL=IU+1
      IU=IU+ITEMS
      IF(IL.GT.MAXNO) GO TO 54
      IF(IU.GT.MAXNO) IU=MAXNO
      GO TO 40
      ...
      rest of routine
```

Linear array PSAVE must be dimensioned at least as large as the largest anticipated value of ITEMS. Variable MAXNO must be the size of the set of numbers being read in to array PSAVE. In both examples here, routine ADVFIL (described in Appendix C) closes the current file member of the family and opens the next file member.

Write sequence numbers on frames generated by PLOT3D. This topic must be singled out for treatment because the frame boundary used by PLOT3D will not necessarily coincide with the frame boundary one has when using the statements to write on logical unit 100 in other routines. Differences in raster size can be compensated for by drawing a rectangular border at the raster boundary in the routine which calls PLOT3D. The 3D perspective plot will probably extend beyond the raster boundary of the calling routine, but now one can determine where to place the sequence number and any other identifying information out of the way of the 3D picture. Once text location has been determined, the statements which were used to generate the border can be removed to allow the frames to be generated without their interference.

2.2.2. Use of Various Techniques to Improve Visual Quality

Summary. As one gains experience making movies and picks up ideas from other film makers, one can add various algorithms or routines to one's library of software which will allow one to get more professional looking results or give one more flexibiity in displaying the data. Three useful capabilities in this category have been developed for use in making movies with 3D graphics, an option to permit rotating the figure by 90°, 180°, or 270°, and a linear interpolation procedure to allow motion to evolve more slowly between successive sets of data; and logic to allow changing mesh size from fine to any desired coarseness for use in getting qualitative pictures in a greatly reduced amount of execution time.

Figure rotation. When planning a movie that shows the surface determined by the values of a function of two variables, it may not always be clear what the best viewing angle will be. One may also find it advantageous to change viewing angle during the evolution of some process. The routine PLOT3D [J. R. McCall and B. Gumm, 1965] allows for a limited amount of perspective change which can be used very effectively to rotate a figure continuously back and forth through a small range of angles to show what the structure of a surface is like. The useful range of angles, though, is between 15° and 75°. One also needs the capability to rotate successively through increments of 90°. A routine to do this called FARRAY is described in Appendix D.

Change of mesh size. Routine FARRAY also contains logic which allows the mesh size of the figure to be plotted by PLOT3D to be made coarser to speed up the computation time and to allow more convenient use of a CRT graphics terminal to monitor output and to debug a movie-generator program by previewing a selected series of frames using the utility routine FR80PLOT.

Change of time scale. If the data to be plotted changes too rapidly from one data set to the next, the animation will appear jerky or may flash by so quickly as to be lost to the viewer. When it is not convenient to change parameters in the computation to fill in such gaps with more data sets, one can attempt to do a linear interpolation in time. There is one problem to look out for in doing this; that is, the array sizes must remain unchanged and each value in the array must correspond physically to the same thing (some quantity measured at the same spatial point or the same particle moving in space relative to its neighbors) so that interpolation between the value of the same element in an array at one time and the value at a later time physically makes sense. Otherwise, further interpolation to a common basis for comparison at successive times will be necessary.

3. POSSIBILITIES FOR FURTHER DEVELOPMENT

3.1. SOURCES OF IDEAS

3.1.1. Locating Sources

Computer-animated graphics have been generated by a few laboratories, primarily working on government-funded research, for two decades. When bigger and faster computers together with cameras and graphics file processing hardware and software became available in the late sixties, the amount of computer-animated graphics began to increase significantly. Still, however, most of the film has been generated by a relatively small number of institutions. As mentioned in the introduction, there is still very little documentation about the movie-making process. There is a considerable amount of experience and oral tradition, though, among the people who have been making movies for many years. Most of these people have carried over the principles of animated graphics developed by people like Walt Disney and those who have elaborated on his ideas in the movie industry. Graphics groups at Lawrence Livermore Laboratory (LLL), the Los Alamos Scientific Laboratory (LASL), and Bell Laboratories probably have the greatest amount of useful accumulated experience.

Film libraries. While written guidelines are lacking, the products of the efforts of those people involved in computer-animated graphics are frequently available upon request from the film libraries of the institutions where the movies were generated. Addresses of some of these libraries are given in Appendix D. These films not only contain examples of various ideas and techniques for displaying information, but also illustrate what the technical possibilities and limitations are. A few carefully selected movies are well worth renting for a while and studying critically.

Professicnal societies. Access to the oral tradition about how to make movies with computers has been facilitated in recent years by the scheduling of special sessions on this subject at the meetings of the Association for Computing Machinery (ACM) Special Interest Groups on Graphics (SIGGRAPH). An evening session of these meetings is now devoted to the showing of computer-animated graphics.

3.2. USE OF COMMERCIAL PROCESSING

3.2.1. Some of the Services Available

Visual impact of a movie can be enhanced with further processing once the basic movie has been spliced together. This processing can be done by an in-house film processing group or by a commercial film processor. Special effects can be added such as tinting, fading a particular sequence in or out, or repeating or removing frames to give the effect of slowing or speeding up the animation. Hand-lettered titles or text with various background designs or pictures can be photographed and inserted, with an appropriate number of frame repeats, at desired locations in the film. The same thing can be done with photographs of anything else which are available on 35 mm film.

3.2.2. Costs and Turn-Around Time

To copy a film (optically) and add the kinds of special effects enumerated above, one will probably have to pay in the neighborhood of $1-2 per foot. This cost guideline should be useful for most processing needs. If commercial processing cannot be done by a local firm so that shipping becomes a significant part of the processing time, one needs to allow at least 2 weeks for fast delivery service or more than a month if shipped at a lower priority. These guidelines are especially important to consider when one is preparing a movie for use in an oral presentation at a technical meeting or any other meeting for which there is a deadline to be met.

3.3. POSSIBILITIES FOR LOCAL PROCESSING

3.3.1. File Transfers

When all the hardware and software have been assembled and checked, it will be possible to send graphics files from Livermore to the user service center (USC) PDP-1Ø where they can be transferred to magnetic tape. The reels of tape can then be processed on a local FR8Ø or other film device. This capability should reduce turn-around time from a week to a day. An attempt to follow this procedure has been made and the feasibiity of this capability has been established. Further development is currently proceeding at a low priority.

3.3.2. Information Volume and Processing Rates

Use of local facilities can present problems because of the enormous volume of information required to make a movie. Some data based on recent experience can be used as a guideline for planning. The file transfer rate over the 5Ø Kbit line between Livermore and the ORNL Fusion Energy Division (FED) PDP-1Ø is approximately 1Ø blocks per second. During peak load times the information transfer rate of the line has been observed to drop to 4Ø Kbits per second [R. D. Burris, private communication]. For graphics files, one must allow 77

10-character CDC 7600 words per block on the PDP-10, so a 300,000 word FR80 file will require roughly 4000 blocks of disk space on the PDP-10. A 3-minute movie using 3D plots would typically require 100,000 blocks of PDP-10 disk space. So one can see that there are going to be difficulties in using local facilities for producing animated graphics.

3.3.3. File Handling Procedures

The data-handling problems illustrated in the previous section will require the development of file-handling procedures whereby the files are transferred over the 50 Kbit line at a rate which will allow large blocks of data to be stored one at a time temporarily on disc, then written on a tape and finally deleted from disk so the next block can be processed. File handling software to accomplish all this has not been written so one attempting to do processing locally currently is left to his own devices.

4. DIAGNOSTIC METHODS

4.1. USE OF UTILITIES NETPLOT AND FR80PLOT

4.1.1. Output From Versatec Printer/Plotter

Select sections of a graphics file for plotting. Initially the Versatec printer/plotter will probably be the most useful and convenient device to use in diagnosing problems. FR80 files are plotted with NETPLOT [NETPLOT Computer Documentation (on-line)] using the F. option. With some practice and experience, one can soon figure out ways to select sections of graphics files and transfer them into a temporary file for plotting. This permits one to look at parts of a family of files without having to generate enormous amounts of printout to get the information which is of interest. A section of a graphics file can be found and isolated using the TRIX AC [TRIX Computer Documentation (on-line)] editor. The file of interest must be opened with the line

```
TRIX AC!O(filename,10,X)
```

and no other way in order to prevent TRIX from adding format characters to the file or in any way altering it. This procedure defines each word to be a line so that there are then as many lines in the file as there are words. One can then search for the frame advance pattern or inter-record-word pattern using one of the commands CLP or CLS to generate a list of the line (word) numbers in which the pattern or symbol is found. The command TL can then be used to display the list. With this information one can copy a range of lines (words) into a temporary file for plotting. There are other ways to select sections of graphics files, but the method described here is the easiest if one knows the symbol or pattern in the graphics file for which one needs to look. Single frames can be selected and stored in a temporary file using the USC command of the utility routine FR80PLOT. When execution of FR80PLOT is terminated, this temporary file is given to the system for plotting with NETPLOT.

Limitations of NETPLOT and FR80PLOT. Currently neither NETPLOT nor FR80PLOT has a provision for decoding frame repeat information, so neither can be used to check on the results of using the routine REFOCO. Debugging problems due to use of frame repeat commands must be done using a dump utility or an editor. Sections 4.2 and 4.3 describe how to use the TRIX editor for such debugging. Also, NETPLOT cannot process color FR80 files. FR80PLOT can handle color FR80 files but ignores color information and plots everything in black and white. The FR80PLOT routine is written in ALGOL so is difficult for NMFECC programmers to modify. A FORTRAN language version is currently being written [Steve Louis, private communication].

4.1.2. Interactive Graphics Display

A graphics display terminal is very useful for diagnosing problems as well as previewing results. The utility FR80PLOT must be used to look at FR80 graphics files. If one wants to inspect a family of graphics files (with sequence character the second character from the left in the graphics file name), one should open the first member of the sequence using the FAM. option of the OPEN command. The frame index command FI provides a list of the number of frames in the currently opened file and tells whether the frame contains data or is blank. If this list is obtained immediately after a file is opened, one then has a better idea of which frame numbers to enter for display. One thus can display selected frames as a means of checking through a family of files for possible problems. If one wants hardcopy of selected frames, one can use the USC command. If one wants to get film output of the file currently open or of the family of files of which the currently opened file is a member, one can use the GIVE command with appropriate options to give the files to the system (User-2 = 2 = usernumber). One is again reminded that FR80PLOT has no provisions for decoding frame repeat information so cannot be used to check on the results of using the TV80 routine REFOCO to repeat frames. This information can be obtained only by using a file dump utility or an editor (such as EDIT or TRIX AC).

4.2. TRIX SEARCH PROCEDURES

4.2.1. Opening Graphics Files

The TRIX general-purpose editor can be very useful as a graphics file diagnostic tool, but one must use it with care so as not to alter the files being examined. It is generally a good idea to copy the graphics file to a temporary file until one gains experience and works out an operating procedure for using TRIX to diagnose problems. The only way to open a file with TRIX without altering the file contents is to use the line

```
TRIX AC!O(filename,10,X)  .
```

This defines each line in the files to be one word in length so that the number of lines is then the number of words in the file. One use of TRIX in this fashion has already been discussed in Section 4.1.1.

4.2.2. Technique for Searching for Special Characters

A few characters from the A7600 character set table are used as special-purpose record delimiter and control characters. When the (6-bit ASCII) octal bit pattern representing these characters is used in a graphics command to denote control such as frame advance, one needs some way to search for such a bit pattern in order to locate the graphics control commands. The bit patterns for these special-purpose characters may be located by using the special purpose characters in the search string if they are preceded by the character "@".

4.3. STRUCTURE OF FR80 COMMANDS

4.3.1. Basic Format of Commands

The word length of FR80 graphics commands is 18 bits or 3 characters in the (6-bit ASCII) octal character representation. Since the A7600 word length is 60 bits, FR80 graphics commands will, in general, extend across word boundaries. Each 18-bit word can be represented either by its octal representation or by a string of 6-bit ASCII characters. Using a 6-bit ASCII character table, one can translate between the ASCII representation and the octal representation of the bit pattern. If an octal number is indicated by $\{O_i: 0 \leq O_i \leq 7\}$, then a graphics command has the format

$$O_1O_2O_3000B + NB^{*} \ , \ N = 1,2,\ldots,2^n-1$$

where n = 9,10,11 (typically). An example is the frame advance command

---

*Octal numbers are distinguished from decimal by the use of the suffix B, i.e., NB is N in octal representation.

Ø34ØØØB + NB , 1 ≤ N ≤ 1777B

Table 4.1. summarizes some of the commands of greatest interest for diagnostic purposes.

4.3.2. Problems with Word Boundaries

The only utility currently available on the A76ØØ that will search for character strings across word boundaries is the editor TRIX AC. When TRIX AC is used to search for a particular string, the word (line) number given when using TP or the combination FP!L is the word (line) where the pattern begins. Should it ever become necessary to go beyond the capabilities of TRIX AC and to write one's own program to analyze graphics files, one should consider using the CHATR STRUCTURE statement to break the 6Ø-bit words down into more elementary units.

4.3.3. Graphics Commands of Greatest Diagnostic Value

The commands one will likely want to search for most often, in order of priority, are frame advance, frame repeat, and the commands for the various colors. The basic representations of these commands are summarized in Table 4.1. Several of these commands require additional explanation.

Frame advance. The basic frame advance command is Ø34ØØØB, but the most commonly encountered form will be Ø34ØØ1B (value of N is 1) which has the (6-bit) ASCII representation #@! (actually must search for, i.e., type on the terminal, #@@!). If the FR8Ø file is produced by TV8ØLIB routines instead of FROG [FROG Computer Documentation (on-line)] utility), then all frame advance commands are at the

TABLE 4.1. Set of FR8Ø Graphics Commands Most Frequently Used for Diagnostic Purposes

Command	Octal Representation	Range of N	Internal (6-bit) ASCII Representation
Frame advance	Ø34ØØØ+N	1 ≤ N ≤ 1777B	#@" (search string: #@@")
Inter-record-word (IRW)	2Ø3ØØØ		Ø8_[a]
Frame repeat	2Ø1ØØØ+N	2B ≤ N ≤ 511	Ø(_
Color filter (first of two words)	214ØØØ		1@_
Color select (second of two words)			
red	6ØØ2ØØ		p"_
green	6Ø24ØØ		p4_
yellow	4ØØ4ØØ		@$_
blue	67ØØØØ		w_ _
magenta	4ØØØ6Ø		@_p
cyan	42ØØØØ		b_ _
white	6ØØØØ2		p_"

[a] A blank space is denoted by "_".

beginning of a record and therefore always preceded by an inter-record-word (IRW) which has the pattern 203000B. Therefore, one could search for frame advance commands together with the IRW as follows:

TRIX AC!O (filename,10,X)!TP!08 #@@!! ... !<CR>

search pattern

The words printed should be the ones in which the pattern sought begins.

Frame repeats. The basic frame repeat command is 201000B + N, 2 ≤ N ≤ 511. To get the (6-bit) ASCII character representation, one must pair the octal characters (20 10 00). The last two octal characters (00) of the basic command can accommodate 64 frames, after which one then gets 201100. This changes the (6-bit) ASCII character representation, in this instance to "0)<blank>". Table 2.3. gives the representations of characters for reference to use when diagnosing files for frame repeat commands.

Color commands. Color information is stored as a sequence of two 18-bit words. The first word is the color filter command (214000B) and the second the color select command. The color select commands are listed in Table 4.1. The (6-bit) ASCII patterns for the combined two word string can be used to insure that the search selects the correct commands for the color information.

4.4. METHOD FOR MONITORING CREATION OF GRAPHICS FILES

4.4.1. Class of Problems Which Will Benefit from this Monitoring Method

In some respects, the method to be described is a method of last resort. When the methods described previously fail to resolve a problem, one can step through the program, executing graphics calls one at a time, and monitor what is being written into the graphics file with each graphics call. This method is particularly useful for determining the results of calls to the REFOCO routine or any procedures in which the same information must be repeated several times.

4.4.2. The Monitoring Method

One executes the program for generating the graphics output from a terminal by using the utility DBCTRL [DBCTRL Computer Documentation (on-line)] to run the program controllee. One can set breakpoints at the beginning of subroutines or at statement labels and run the controllee until the breakpoint is reached. Then one can switch to a second suffix and use TRIX AC or any other appropriate editor to look at the contents which have been written into the graphics file. After examination of the file is finished, one can go back to the first suffix and continue execution to the next breakpoint, then switch back to the suffix on which the editor is running to examine what additional commands have been added to the graphics file. This technique can be continued in the manner described, for as long as necessary. It is slow and time-consuming, but permits one to examine what is being written into the graphics file in a dynamic mode in great detail.

REFERENCES

E. Carpenter and S. Solbeck, "Livermore Time Sharing System," Part IV: Library Files, Chapter 304: ORDERLIB Subroutine Libr·.·y, Lawrence Livermore Laboratory Document M-026, Edition 2 (1976).

J. R. McCall and Barbara Gumm, "PLOT3D: A FORTRAN Subroutine for Three-Dimensional Plotting". Available through Graphics Coordinator, National Magnetic Fusion Energy Computing Center, Lawrence Livermore Laboratory, Livermore, California 94550.

J. K. Munro, Jr., "HLYWD: A Program for Post-Processing Data Files to Generate Selected Plots or Time-Lapse Graphics." (To be published)

The following documentation is available to NMFECC users as on-line documentation and may be obtained through the DOCUMENT utility:

DBCTRL	FR80PLOT	TRIX
DOCUMENT	NETOUT	TV80
FILEM	NETPLOT	
FROG	TEKPLOT	

APPENDIX A

PROGRAM AND PROCEDURES FOR GENERATING TEXT SEGMENTS IN COLOR

The program MOVIETEXT is a simple program for generating arbitrary text sequences. This program is designed so that blocks of text can be added in a delayed sequence in generating a complete frame. This helps to sustain action and to hold the attention of the viewer. The number of repetitions of a given frame of text can be specified on input or determined automatically by the program which allows a viewing time of two seconds for each line of text. A different color can be specified for each line. All types of text, no matter where it should go in a final movie, can be generated with this program, then spliced into a film at the appropriate place at the time the film is edited. This program can also be incorporated easily into other software, if desired.

The format of the input data is given in Table A1 and can be repeated as many times as there are different sets of text. When checking a new text input data file for centering, balance, and proper text character size, one can set the value of NF=1 and route the output file to a Versatec printer/plotter using NETPLOT with the F. option. Text is automatically centered in the frame in the vertical direction, but the user must center each line in the horizontal direction which is 42 characters wide for the program listed below. When blocks of text are added in a delayed sequence to build up to a final completed frame, each step of this building process must be treated as a separate frame of text when constructing the input data file. The input data file must be named TXTDAT and is read by logical unit 2.

TABLE A1. Input Data Formats for Generating Segments of Text on Film

Line Number	Columns	Description of Variables and Arrays	
1		FORMAT (3I2)	
	1-2	LT	Number of lines of text. If LT=Ø, program execution terminates.
	3-4	LB	Number of blank lines. Total lines on a frame LINES=LT+LB. Number of blank lines must include number of lines of new text to be added later in delayed sequence to complete the frame.
	5-6	NT	Number of lines of new text added in delayed sequence. If no new text is to be added to original block, let NT = -1.
2		FORMAT (I3)	
	1-3	NF	Number of frames generated for a frame which must be repeated. If NF > Ø, the value of NF read in determines the number of frames; if NF ≤ Ø, the number of frames is determined by the program.
3 . . . (2+LINES)		FORMAT (6A1Ø)	
	1-42	TEXT(I) I=1,5	Single line of text. After each line is processed, LINES is decremented by 1 and the next line is processed. After all lines are processed, the program proceeds to the input for the next frame of text.
	51-60	KOLOR	Name of one of the colors allowed by TV8Ø software.

Program Listing

```
      PROGRAM MOVIETEXT (TXTDAT,TAPE2=TXTDAT,INPU,TAPE5=INPU,OUTPUT)
      COMMON /GOBCOM/IG(38)
      CALL CHANGE("+GOTXT")
      CALL KEEP80(1,3)
      IG(36)=10HBOX A55  J
      IG(37)=10HOHN K. MUN
      IG(38)=10HRO, JR.
      CALL FR80ID(10H ORNL FED ,1,9)
      CALL KEEP80(5RTEXTS,3)
      CALL TXTFRM
      WRITE(59,"(""END OF TEXT"")")
      CALL PLOTE
      CALL QUIT
      END

      SUBROUTINE TXTFRM
      DIMENSION TEXT(5)
      DATA H/48./,W/24./,HH/24./,NCAR/4/,NT/0/
      DATA KOLOR/"WHITE"/
      CALL MAP(1.,1024.,1.,1024.)
    5 CALL PLOTEA
      CALL GIVEUP(0)
      CALL SETCHM(1)
      READ (2,200) LT,LB,NT,NF
      IF(LT .EQ. 0)  RETURN
      LINES=LT+LB
      KF=LT
      IF(NT .GT. 0)  KF=NT
      IF(NF .LE. 0)  NF=40*KF
      TH=LINES*H + (LINES-1)*HH
      CR=H + HH
      TM=0.5*(1024.-TH)
      ULB=1024.-TM
      ULM=0.
      LO=5
      IF(NT .GT. 0)  GO TO 9
      CALL FRAME(LO)
    9 CALL REFOCO(NF)
   10 READ (2,201) (TEXT(I), I=1,5),KOLOR
      CALL COLOR(KOLOR)
      CALL SETLCH(ULM,ULB,0,0,3)
      CALL CRTBCD(TEXT(1),NCAR)
      ULB=ULB-CR
      LINES=LINES-1
      IF(LINES .EQ. 0)  GO TO 20
      GO TO 10
   20 CALL FRAME
      CALL REFOCO(0)
      IF(NT .EQ. 0)  GO TO 5
      IF(NT .GT. 0)  LO=5
      CALL FRAME(LO)
      GO TO 5
  200 FORMAT(3I2/I3)
```

```
 201 FORMAT(6A10)
     RETURN
     END
```

Sample Input Data Set

```
4 4 0
 2 FRAMES
EVOLUTION OF FOKKER-PLANCK DISTRIBUTION          GREEN
    FUNCTION SOLUTION FOR SLOWING OF             GREEN
   INJECTED NEUTRAL BEAM OF HYDROGEN             GRSEN
             INTO A PLASMA                       GREEN




6 2 2
 2 FRAMES
EVOLUTION OF FOKKER-PLANCK DISTRIBUTION          GREEN
    FUNCTION SOLUTION FOR SLOWING OF             GREEN
   INJECTED NEUTRAL BEAM OF HYDROGEN             GREEN
             INTO A PLASMA                       GREEN

          AS SIMULATED BY THE                    BLUE

    OAK RIDGE TOKAMAK TRANSPORT CODE             BLUE
2 5 0
 2 FRAMES
      TRANSPORT CODE DEVELOPED BY                BLUE

             JOHN T. HOGAN                       YELLOW




5 2 3
 2 FRAMES
      TRANSPORT CODE DEVELOPED BY                BLUE

             JOHN T. HOGAN                       YELLOW

       FUSION ENERGY DIVISION                    BLUE
     OAK RIDGE NATIONAL LABORATORY               BLUE
        OAK RIDGE, TN  37830                     BLUE
2 5 0
 2 FRAMES
          MOVIE PRODUCED BY                      GREEN

          JOHN K. MUNRO, JR                      YELLOW



5 2 3
```

2 FRAMES	
MOVIE PRODUCED BY	GREEN
JOHN K. MUNRO, JR	YELLOW
COMPUTER SCIENCES DIVISION	GREEN
UNION CARBIDE CORP., NUCLEAR DIV.	GREEN
OAK RIDGE, TN 37830	GREEN
5 0-1	
2 FRAMES	
RESEARCH SPONSORED BY THE	GREEN
OFFICE OF FUSION ENERGY	GREEN
U. S. DEPARTMENT OF ENERGY	GREEN
UNDER CONTRACT W-7405-ENG-26 WITH THE	GREEN
UNION CARBIDE CORPORATION	GREEN
0 0 0	

APPENDIX B

SUMMARY OF PROCEDURES AND INFORMATION FOR USING DD8Ø GRAPHICS FILE FORMAT

B.1. INTRODUCTION

The discussion in this appendix will be brief and presented with the assumption that one has studied the main body of this guide. The order of presentation of material here is basically the same as in the guide.

B.2. THE MOVIE-MAKING PROCESS

B.2.1. BASIC ELEMENTS

B.2.1.1. Organizing Data Handling and Graphics Software

File Sizes and Names. The dd8Ø file default size is 5Ø,ØØØ words. The recommended size range for making movies is the same as for FR8Ø files, 2^{17} - 2^{18} words. The file naming convention is similar to that for FR8Ø files, with the basic form Dsnnnnnnnn where the first character is always a D to indicate the file is of dd8Ø format, s is the sequence character, and nnnnnnnn is a string of the first 8 nonblank characters in the file name. This form is used for saved dd8Ø files. A different naming convention is used if files are given directly to the system. Files given directly to the system are normally processed to produce Xerox output, though 35 mm film can be obtained on request. One cannot get 16 mm film output directly this way. Hence dd80 files must be saved for subsequent processing by the utility FROG.

B.2.1.2. Developing Efficient Operating Procedures

Repeating Information from One Frame to the Next. The method for repeating information on successive frames of dd80 is different from the use of the routine REFOCO. It is more flexible in many ways, but also has limitations which are not present with the use of REFOCO. The routine REFOCO only works for FR80 graphics. To repeat graphics information in dd80 files, one must use the ORDERLIB routines SAVED and DPLOT.

Routine SAVED saves a buffer of plot information. One must call PLOTEA or ENDPL to empty the dd80 buffer before generating the information to be saved, so that it will begin at word zero of the buffer. The length of the array for saving information is determined by the length of the dd80 buffer which is 1200 words. The dd80 buffer is cleared upon transfer of the data to one's designated storage area. Routine DPLOT copies the stored plot commands back into the dd80 buffer. One should study the ORDERLIB document [Carpenter et al , 1976] very carefully on the proper use of these routines. The array used for saving the plot commands must be in LCM and the remaining parameters in the routine calling parameter list must be in SCM. A major limitation on the use of these routines arises when using graphics software that fills the dd80 buffer more than once during execution before returning control to the calling program. If this problem arises, one either must modify the software explicitly by adding calls to SAVED at appropriate locations in the source or use the

ORDERLIB routines SETPLA and SETPLC to write information generated by plot calls into a user-designated array (a different buffer) of whatever length is necessary.

Job Restart Provisions. The restart logic is the same as for programs creating FR80 files, but the calls to re-initialize graphics procedures and open graphics files are a little different. A typical block of coding should look as follows:

```
      CALL FRAME
      . . .
      IF(SENSESWITCH,1) 30,20
   30 CALL OFFMON
      CALL KEEP80(INAME)
      IG(36)=10HBOX A55
      IG(37)=10HJOHN K. MU
      IG(38)=10HNRO  MOVIE
      CALL GFSIZE(2,300000)
      CALL DD80ID(10H ORNL FED ,1)
      CALL MAP(1.,1024.,1.,1024.,0.15,0.86,0.285,1.)
   20 GO TO 210
      RETURN
      END
```

The variable INAME in the KEEP80 calling parameter list should contain the Hollerith string identifying the dd80 output files. This name must be assigned using either the form "NAME" or 4HNAME and be right-justified in the 10-character field.

B.2.1.3. Generating Film Segments

File Monitoring. File monitoring is done the same way as for FR80 files except that one selects file names beginning with D for dd80 files. After the utility FROG has been used to give dd80 files to the system for processing into the FR80 format, then writing onto magnetic tape, one can check on the results using

FILES USER2. ORGUSR. ALWITH.F

as with FR80 files.

File Disposition for Processing. The only way to get 16 mm sprocketed film from dd8Ø graphics files is to use the utility FROG to convert them to the FR8Ø format and give them to User-2. One does this with the command and options

FROG P16MM. 2FAM. DsFILMNAME

where the number 2 in option 2FAM. identifies which character in the name which follows is the sequence character (for giving a family of files to the system) and the letter s in the name of the dd8Ø file represents the sequence number. A 3ØØ,ØØØ (dec.) word dd8Ø plot command file will yield 2 1/2 FR8Ø plot command files after processing by FROG. With the resulting FR8Ø file size, 11 FR8Ø family members will be written on a tape (and presumably fill it) when a family of 4 dd8Ø plot command files is processed by FROG.

B.2.2. WRITING AND ASSEMBLING SOFTWARE

B.2.2.1. Basic Code Blocks to be Incorporated

Initialize Graphics Files. The same differences show up here as are in the job restart provisions example. The basic set of statements should include

```
COMMON/GOBCOM/IG(38)
. . .
DATA INAME/10H  DD8ØFILE/
. . .
CALL KEEP8Ø(INAME)
IG(36)=1ØHBOX A55
IG(37)=1ØHJOHN K. MU
IG(38)=1ØHNRO  MOVIE
CALL GFSIZE(2,3ØØØØØ)
CALL DD8ØID(1ØH ORNL FED ,1)
. . .
```

B.3.3. STRUCTURE OF DD8Ø COMMANDS

B.3.3.1. Frame Advance Command

The frame advance command is identified by the pattern ØØ4ØØØØØØØØØ which has the (6-bit) ASCII character representation "_@____" and can be located using TRIX by entering the search string "_@@____" since @ is a special control character.

APPENDIX C

A COLLECTION OF USEFUL SOFTWARE

ADVFIL

Purpose. This routine closes the current file with name stored in variable JNAME and read by logical unit number LUN and opens the next member of the family of files. Error messages are printed if problems are encountered in attempting to close the old file or open the new file; the error number is given and the name of the file associated with the problem is printed. Errors corresponding to the error number are listed in the ORDERLIB document in the description of the routine DEVICE.

Calling sequence CALL ADVFIL(LUN,JNAME,LEN,IOC)

INPUT LUN, JNAME

OUTPUT LEN, IOC

LEN - length of the new member of file family

IOC - I/O connector number which ties the disk files to the device which is assigned to read them.

Subroutines called

From ORDERLIB: ASSIGN, DEVICE, PLOTE, QUIT

Program Listing

```
      SUBROUTINE ADVFIL(LUN,JNAME,LEN,IOC)
C *** VERSION DATE = YR-MO-DA-HR-MN = 7707201015
C  AUTHOR:   JOHN K MUNRO JR
C            COMPUTER SCIENCES DIVISION
C            UNION CARBIDE CORPORATION, NUCLEAR DIVISION
C            OAK RIDGE, TN  37830
C  PURPOSE...
C       ROUTINE CLOSES CURRENT FILE 'JNAME' BEING READ ON LOGICAL UNIT
C       NUMBER 'LUN' AND OPENS THE NEXT DISK FILE BELONGING TO THE FILE
C       FAMILY NAME.
      DATA IERR/0/,LID/0/
      CALL DEVICE(6HCLOSER,JNAME,IERR,IOC)
      IF(IERR .NE. 0)  WRITE(59,200) IERR,JNAME
  200 FORMAT(1H ,"I/O ERROR",I3," UPON ATTEMPT TO CLOSE ",A10)
      JNAME=JNAME+1
      CALL ASSIGN(LUN,LID,JNAME)
      CALL DEVICE(4HOPEN,JNAME,LEN,IERR,IOC)
      IF(IERR .EQ. 0 .OR. IERR .EQ. 2)  RETURN
      WRITE(59,201)  IERR,JNAME
  201 FORMAT(1H ,"I/O ERROR",I3," UPON ATTEMPT TO OPEN ",A10)
      WRITE(59,"(""END OF RUN"")")
      CALL PLOTE
      CALL QUIT
      RETURN
      END
```

FARRAY

Purpose. To transform a 2D array of numbers (a function of two variables) so, when it is plotted with a 3D plotting routine, the viewing direction will be rotated by 9Ø°, 18Ø°, or 27Ø°. Routine also can reduce the number of function values to be plotted. If NDEXF(6)=2, then every second value in the x- and y-directions of the function stored in the PSAVE array is used; if NDEXF(6)=3, every third value is used, etc. Use of NDEXF(6)=3 reduces computing and CRT plotting time by nearly an order of magnitude. This parameter is very useful for trying new ideas and for general trouble-shooting and program debugging, for one can get a general idea of what a function looks like quickly.

Calling sequence   CALL FARRAY(IOPTØ5,PSAVE,NDEXF,F)

where IOPTØ5 is an option selector variable

= Ø   no rotation

= 1,2,3   rotation 9Ø°, 18Ø°, and 27Ø°, respectively

PSAVE   linear array of function values to be transformed. The array is filled with values for all y-direction nodes for each successive value of the x-direction nodes.

NDEXF   array of indices

NDEXF(2)   number of nodes in the x-direction for PSAVE array

NDEXF(3)   number of nodes in the y-direction for PSAVE array

NDEXF(6)   mesh size scale factor used to reduce the number of function values to be plotted by 3D plotting routine

NDEXF(7)	number of nodes in the x-direction for F array
NDEXF(8)	number of nodes in the y-direction for F array

F — linear array of function values after transformation by rotation or reduction of nodes. Order in which array elements are assigned is variable.

Input   IOPTØ5, PSAVE, NDEXF(2), NDEXF(3), NDEXF(6)

Output   NDEXF(7), NDEXF(8), F

Program Listing

```
      SUBROUTINE FARRAY(IOPT05,PSAVE,NDEXF,F)
C -------------------------------------------------------------
C  AUTHOR:    J K MUNRO JR
C             COMPUTER SCIENCES DIVISION
C             UNION CARBIDE CORPORATION, NUCLEAR DIVISION
C             OAK RIDGE, TN  37830
C  PURPOSE...
C          ROTATE 3-D PLOTS AND ADJUST COARSENESS OF MESH FOR QUICK
C          PLOTTING AT A GRAPHICS TERMINAL
C -------------------------------------------------------------
      DIMENSION PSAVE(1),F(1),NDEXF(1)
      NDX=NDEXF(2)
      NDY=NDEXF(3)
      NP=IOPT05+1
      NS=NDEXF(6)
      NSM=NS-1
      NX=(NDX+NSM)/NS
      NY=(NDY+NSM)/NS
      NDEXF(7)=NX
      NDEXF(8)=NY
      GO TO (100,200,300,400), NP
  100 DO 10  IX=1,NDX,NS
      I=(IX+NSM)/NS
      DO 10  IY=1,NDY,NS
      J=(IY+NSM)/NS
      IDJ=(I-1)*NY+J
      IDJTH=(IX-1)*NDY+IY
      F(IDJ)=PSAVE(IDJTH)
   10 CONTINUE
      RETURN
  200 DO 20  IX=1,NDX,NS
      I=NX-(IX+NSM)/NS+1
      DO 20  IY=1,NDY,NS
      J=(IY+NSM)/NS
```

```
      IDJ=(J-1)*NX+I
      IDJTH=(IX-1)*NDY+IY
      F(IDJ)=PSAVE(IDJTH)
   20 CONTINUE
      RETURN
  300 DO 30   IX=1,NDX,NS
      I=NX-(IX+NSM)/NS+1
      DO 30   IY=1,NDY,NS
      J=NY-(IY+NSM)/NS+1
      IDJTH=(IX-1)*NDY+IY
      IDJ=(I-1)*NY+J
      F(IDJ)=PSAVE(IDJTH)
   30 CONTINUE
      RETURN
  400 DO 40   IX=1,NDX,NS
      I=(IX+NSM)/NS
      DO 40   IY=1,NDY,NS
      J=NY-(IY+NSM)/NS+1
      IDJ=(J-1)*NX+I
      IDJTH=(IX-1)*NDY+IY
      F(IDJ)=PSAVE(IDJTH)
   40 CONTINUE
      RETURN
      END
```

APPENDIX D

SOME FILM LIBRARIES CONTAINING EXAMPLES OF COMPUTER-GENERATED MOVIES

The libraries listed here publish either a catalog or brochures describing the movies they have available. These materials contain brief descriptions of the movies as well as listing titles by categories.

1. Bell Laboratories
   Film and TV Department
   Room 3C-236
   Murray Hill, NJ 07974

2. International Film Bureau, Inc.
   332 South Michigan Avenue
   Chicago, IL 60604
   (312) 427-4545

3. Los Alamos Scientific laboratory
   Attention: Report Librarian
   P. O. Box 1663, Mail Stop 364
   Los Alamos, NM 87545
   (505)667-4446; FTS: 843-4446

The Lawrence Livermore Laboratory (LLL) does not currently have a film library with computer-generated movies, though the LLL Graphics Group has assisted many institutions in computer movie-making projects. For information about the software and movie-generating capabilities which are currently being used, contact

Donald L. Vickers
Computer Graphics Group
Lawrence Livermore Laboratory
P. O. Box 808
Livermore, CA 94550
(415) 532-4231