22 February 1990

ILLUSTRATED BY ANDRZEJ DUDZINSKI

MULTIMEDIA

Is the Message

With so much recent attention surrounding it, the term "multimedia" seems to be on everyone's lips. What is multimedia? Is it a breakthrough in mixing traditional media in a revolutionary new way, or just some fancy juggling of old concepts in new packaging? What software already exists to create multimedia productions? And what new programs are on the horizon that may give the multimedia concept more than just passing-fad status?

Like "plastics" to Dustin Hoffman in *The Graduate*, "multimedia" is the current hot buzzword and wave of the future. And like another pop-culture phenomenon of that era—the "global village" of Marshall McLuhan—multimedia has quickly built up a coterie of proponents and a veil of conveniently confusing hype to spice up its appeal. Not only does it seem like multimedia is the solution to all informational needs, but also in some kind of McLuhan redux, multimedia is the *message* itself.

Despite the fact that some computer makers and the press act as if they invented the concept just last week, the multitasking Amiga has been combining graphics, text, animation, and sound in desktop-video applications for some time now. The Director—software that is kind of like a multimedia programming language—has been providing the "glue" to integrate all these media for over two years. (See "And For Best 'Direction'. . . ," p. 25, for a look at The Director in action.) Whether they call it desktop video, interactive presentation, or hypermedia, people in the community know that multimedia, while not exactly old hat, isn't a brand new Easter bonnet either!

What is new is the great interest in and attention being paid multimedia at present. The positive results are new activity and new products in the field. Innovators are looking past desktop video to applications that can reference data on CD-ROM disks and show images from a laser disk. (New modules in The Director program and the icon-based authoring system VIVA—see "Author! Author!", p. 30—are active here.) A lot of new and more sophisticated thinking is also going into interactive (user-directed) programs that can access a wide variety of information sources and different media. (See "Play Your Best Hand," p. 38, for a look at UltraCard's "stack" approach to designing interactive multimedia presentations.)

Brave New Worlds of Multimedia

In addition to the existing Amiga multimedia packages already mentioned (which will be covered in detail elsewhere in this issue), a number of new programs are waiting in the wings. So in the remainder of our "multimedia overview" we will preview five packages—some of which may be out or just about to be released by the time you read this—to see if we can get an idea where multimedia is going.

The **Commodore Authoring System** will allow you to create a fully interactive multimedia presentation using only the mouse and a large selection of icons. Although the program is actually a graphic, object-oriented programming language, the interface should make the system simple to use for non-programmers.

Using icons that represent elements as diverse as video, pictures, animations, text, and digitized and synthesized sounds, you build a graphic flowchart of the presentation. User input can be added from the mouse, keyboard, or even a touch screen. Decisions ►

By Oran Sands III and Louis R. Wallace

can be made based on user input by employing one of several control icons that perform such programming-style functions as IF-THEN-ELSE, GOTO and LOOP. Other flow modules can be accessed using subroutine calls, and it is even possible to execute external programs via an ARexx port.

With a video-disk player and a genlock, you can combine canned video with the Amiga's graphics and animation in a highly controlled manner, specifying exactly how many frames of video to display. When combined with overlayed graphics, this allows full motion video images to be accessed from within any application. A large number of wipes and transitions are available.

Again using only icons and requesters involving little keyboard input, you will also be able to use the program to create a simple database. You can use it to store information from users during a session or to supply information to users when requested. Once you finish any application, you can distribute it by creating a runtime version of it to use as your presentation or course.

ShowMaker (Gold Disk) is designed to be an interactive multimedia presentation system that will allow you to combine Amiga-generated sound, graphics, and animation with video from laser disks, video tape recorders, and cameras. By loading the next elements of your presentation from disk while simultaneously displaying the current sequence, you will be able to create long-playing productions.

One of ShowMaker's strengths is its ability to precisely synchronize sound and music with specific graphic or video events. Other notable features include built-in titling software, several dozen wipes and transitions, and support for Anim, RIF and MovieSetter format animations. On the audio side, it supports MIDI, SMUS, and 8SVX. ShowMaker also supports laser-disk players.

From Very Vivid Inc., creators of The Mandala, comes a "hypermedia presentation system" called **Interactor**, which uses the idea of a theatrical production as a metaphor for the process of creating its applications. Your presentation is thought of as a "play" (also called a stack), which in turn consists of a number of "scenes." Scenes contain "backdrops" (pictures) and "actors" (objects), each of which have certain "roles" (or states of being).

Interactor supports low-, medium-, and high-resolution interlaced modes, overscan display, and both single- and double-buffered modes. You are offered a variety of fonts and "softstyles" for text, and you can include such effects as color cycling and fades in your scripts.

Besides being able to import a variety of graphics such as pictures and brushes into its presentations, Interactor provides its own animation capabilities that enable you to move sprites and brushes on the screen with full collision detection. This allows the presentation to take specified set actions if certain objects come in contact.

Interactor currently supports genlocks and laser disks; additional modules from Very Vivid will enable you to add other hardware devices in the future.

One of the earliest programs to allow the multiple and interactive uses of sound, graphics, and animation, DeluxeVideo will soon be available in a revised version from Electronic Arts. **DeluxeVideo III** adds new features and improves on many of the older ones. No longer using the "dual-playfield" format for its display, which limited the number of colors and resolutions users had access to, DVideo III now supports all display modes—including HAM, Extra_Halfbrite, hi-res, overscan and SuperBitMap.

A major strength of DVideo III is its full compatibility with DeluxePaint III's new animation types, which makes it possible to include DPaint Anim and Animbrush files, as well as more standard graphics and brushes, in your videos. And unlike the earlier version, DVideo III saves all data separately from the script, so it can be used in other videos as well. (If you wish, however, you can convert your video directly into a standard Anim file, using the MakeAnim utility supplied with the program.)

DeluxeVideo III supports genlocks and MIDI devices via an ARexx port. It also offers HyperCard-like object-oriented user control. Object motion has been improved with a refined MovePath routine enabling you to define the motion of objects with the mouse. Also new is a relative-motion option, allowing you to attach one object to another. Moving the main object causes the second to follow.

Other features include better font control, background patterns, more cut-and-paste options, interactive options for mouse and joystick, and easy video appending.

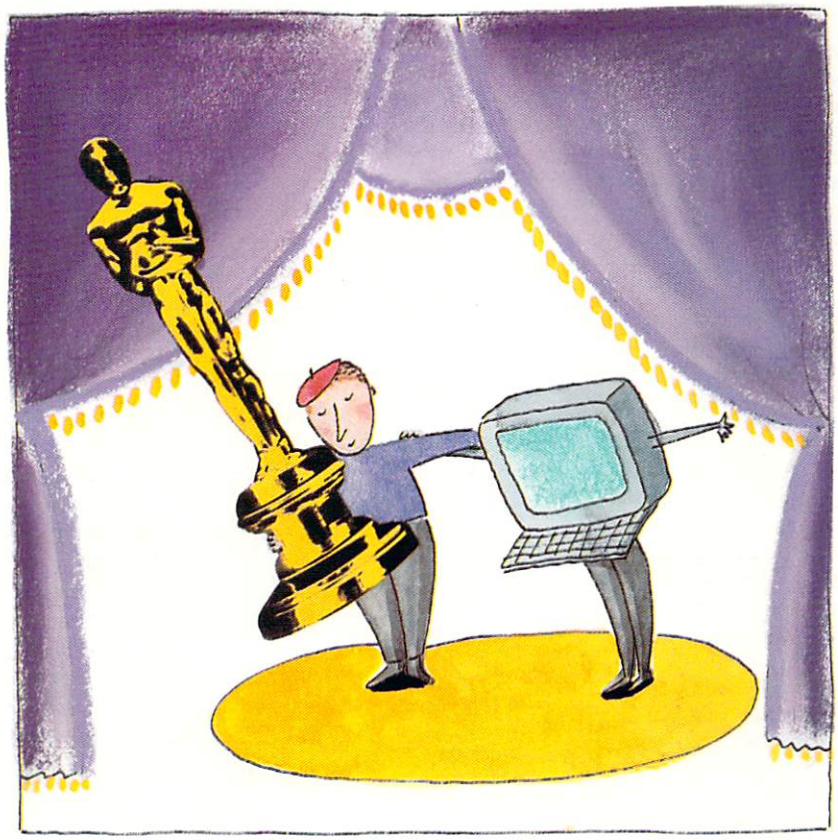
The **CanDo** (INOVAtronics) authoring system offers many of the same features as the Commodore authoring program. With CanDo, however, you combine objects and events into "cards," then assemble the cards into a "deck," which is the final application or presentation.

Objects can be standard graphics, text, brushes, animations, or sounds. They can be presented in a variety of screens and windows, and in conjunction with several opportunities for user interaction—via buttons, requesters, "hot spots," and the like.

CanDo supports external video and audio hardware, it is ARexx-compatible, and it supports DPaint III's BrushAnim format. Completed CanDo decks can be saved as independently executable applications that you may distribute or sell without any licensing fees. ■

Oran Sands III is a video producer and engineer for Methodist Hospital of Indiana. Louis R. Wallace is AmigaWorld*'s Senior Editor, Technology. Write to them c/o* AmigaWorld, *Editorial Dept., 80 Elm St., Peterborough, NH 03458.*

"And For Best 'Direction,' The Winner Is..."



One of the creators of The Director demonstrates how it can help you create your own one-man multimedia show.

By Joel Hagen

IF YOU NEED to create interactive or stand-alone presentations using the Amiga, chances are you're talking about The Director (Right Answers Group, $69.95). Indeed, it occupies a unique niche as the only animation and presentation language for the Amiga. The Director offers powerful control over the Amiga's hardware strengths in graphics, animation, and sound—without forcing a cumbersome structure on an individual's creativity.

With the addition of The Director's Toolkit and some other new modules, the combined power of the program also provides the user with software control over Laser disk players, VCRs, CD players, camcorders, and other external devices. In this article we will focus on how to create interactive and multimedia presentations on your Amiga with the help of The Director.

So What Can You Do for Me?

In terms of the content of presentations, The Director allows you to combine and sequence just about any combination of pictures, text, and sounds you can create with your arsenal of Amiga software. The Director supports a choice of animation approaches including playback of Anims created with Deluxe-Paint III (Electronic Arts), VideoScape 3D (Oxxi), FrameGrabber (Progressive Peripherals & Software), or anything that creates standard IFF Anim files. You can add sound to these Anims and change playback speeds. More advanced users might want to load multiple Anims into memory and chain them together for playback, or superimpose text or other effects on a frame by frame basis.

Page-flipping animation using either full or partial screens is another option. This allows a small number of IFF image pieces to be recombined in new ways in response to specified conditions. For instance, an animated character whose arms, legs, and body parts occupy only three screens could walk around and point to different locations for twenty minutes, ►

ILLUSTRATED BY MERLE NACHT

whereas an Anim file of identical size would have to end or repeat in just seconds. Similarly, you can reposition a single picture element repeatedly. For instance, you could have a pointing finger move endlessly around the screen indicating various information without taking up much valuable disk or RAM space.

Another memory-efficient presentation technique is to avoid using IFF screens for titles or text information. Instead, The Director's TEXT command allows you to print directly to the presentation display or to the CLI either from the script or from an external file. You can use any number of fonts, and by changing PEN colors and repositioning the coordinates of a TEXT statement, you can create on the fly such effects as drop shadows, and embossed, extruded, or stenciled text. For hi-res overscan presentations, this memory-saving feature may be critical. In any resolution, it allows the user to concentrate disk and RAM space resources on images, Anims, and sound files.

Let's Look at Some Scenarios

Interactive presentations with the Amiga are an excellent alternative to slide shows, flip charts, or chalk talks. At a simple level, you can write presentations that pause at predetermined spots while the speaker elaborates a point being illustrated on screen. The program waits for a keystroke or a mouse click to trigger the next sequence. This is an easy way for a speaker to deliver a presentation making use of the mouse as a hand-held remote button. At more complex levels, you can create "hot spots" or buttons on graphic screens that recognize a mouse click within a defined area. The click triggers branching decisions based upon the x,y position of the pointer. This kind of interaction is typical in kiosk informational programs, where the user clicks multiple-choice boxes branching to a selection of restaurants, theaters, or shops.

Similarly, the program can wait for a particular keystroke to trigger a branching event, or it can compare an input word or phrase to a stored string to determine correct or incorrect response. Interactive educational programs often make use of this capability. A quiz on films, for example, could wait for the user to select a category such as Mystery with a mouse click in a box, then use the TEXT command to ask randomized questions such as, "Who played Sam Spade in the Maltese Falcon?" The program would then compare the typed answer with the string "Humphrey Bogart" to determine the next event, and catalogue the result for final scoring.

Instead of a mouse or keystroke, a touch screen can be used as the interactive interface to the program. The Future Touch touch screen from Amigo Business Computers ($1095 with monitor, $795 with kit to modify existing monitor), for example, emulates the mouse and allows the user to simply touch an area of the screen designated as a hot spot with a finger rather than move a pointer with a mouse. This is excellent for interactive programs for young children, or for presentations where equipment is exposed to heavy public use where a mouse would be unfamiliar or soon damaged. Director-driven kiosks are being used in airports and hotels in just this manner.

Getting Started

Basic Script Writing

AS AN ANIMATION and presentation *language* for the Amiga, The Director's syntax is patterned after BASIC to make it as familiar as possible, with specific commands added to simplify the creation of presentations and animations. The overall program size is small for efficient use of memory. Weighing in at less than 60K, The Director is one of the few programs that will actually function well in a 512K environment, although it will take full advantage of any expansion RAM present. A comprehensive video tutorial course on The Director is available from Right Answers for $39.95.

The Director works from the CLI with any text editor, even ED in your C directory. A simple slide show could be written like this:

```
LOAD 1,"landscape"
LOAD 2,"clouds"
LOAD 3,"dog"
LOAD 4,"trees"
LOAD 5,"portrait"

DISPLAY 1:PAUSE 100
DISPLAY 2:PAUSE 100
DISPLAY 3:PAUSE 100
DISPLAY 4:PAUSE 100
DISPLAY 5:PAUSE 100
```

This sequence loads five pictures into chunks of memory called "buffers" and then displays them one at a time, pausing 10 seconds (100 tenths) for each display. The more memory you have, the more buffers you can fill with pictures, sounds, and Anims.

The example below gives a precise pathname for locating images, and changes the script from a slide show into a looping page-flip animation by simply reducing the pause from 10 seconds to one-tenth of a second:

```
LOAD 1,"df1:wink1"
LOAD 2,"df1:wink2"
LOAD 3,"df1:wink3"
LOAD 4,"df1:wink4"
LOAD 5,"df1:wink5"

10 DISPLAY 1:PAUSE 1
   DISPLAY 2:PAUSE 1
   DISPLAY 3:PAUSE 1
   DISPLAY 4:PAUSE 1
   DISPLAY 5:PAUSE 1

   GOTO 10
```

A line number has been added to the first DISPLAY command as a label. The last command tells the program to "go to" that labeled line and begin again. The result is an endless loop of the pictures in buffers 1 through 5 page-flipping at animation speed.

If the pictures are a logical sequence, like someone winking, the effect will be smooth motion. More complex presentations can be built in logical stages from simple elements such as this one. *—JH*

The Director's Toolkit ($39.95) contains a number of handy additions to the original Director program that can be called from within Director scripts. Of interest to those involved in multimedia presentations is the MIDI input module. MIDI is the Musical Instrument Digital Interface, a communications network for interfacing musical instruments with computers. Using the Toolkit's MIDI input module with a Director presentation allows a musician to trigger screen events with musical notes. The Director gets note on/off information (actually note and velocity) as it listens to one of 16 MIDI channels. This information can be used in the same way as a mouse click or keystroke to change the screen display, or to present text information in real-time interaction with the performer.

MIDI input can also respond to devices other than instruments. A SMPTE (Society of Motion Picture and Television Engineers) to MIDI converter can send information via the MIDI input to The Director. While a video plays, the program waits for a specified frame to send its unique SMPTE signal. This can then trigger an Amiga event either on screen or genlocked over the video. In this way, the Amiga could overlay appropriate floor-plan and price information of houses as video of a housing development plays. Prices could be updated in a Director script in minutes without the need of altering or reshooting the video.

Hooking up With Some Heavy Hitters

In another multimedia vein, Right Answers has just released an experimental driver for Pioneer laser disk players that allows program control over all laser disk player functions. This driver, called LVIDEO, is compatible with Pioneer series 4200, 2200, and 8000, as well as with its industrial LC-V330 auto changer. Via the RS-232 serial interface, a Director script can Play, Pause, single step, forward and reverse, seek to frame number, superimpose text on screen, control speed, eject, and trigger any other remote functions. The LVIDEO module is available on PeopleLink, from Right Answers for a nominal handling charge, or as part of the *AmigaWorld* Animation Software Library (a two-disk set costing $14.95).

Utilizing the kind of hot-spot screen buttons described above, an interactive computer/laser disk tool or presentation could be created. Figure 1 shows a Planetary Image Library main menu offering the user an initial choice of planets to view. Each planet's selection box is defined as a button by the x,y coordinates of its boundary (see Example 3 in the "Up and Running" sidebar). Clicking the Mars box could trigger branching to another screen such as the MARS menu (Figure 2) in which selection buttons are defined as the quadrangles of a USGS Mars map. Selecting the appropriate sector could then trigger branching to other options, such as "Surface view" or "Orbital view" and eventually to an LVIDEO module command. This could seek to a specified frame number on a laser disk of NASA images, and display the appropriate Viking lander image of the Martian surface (Figure 3). (As a kid who grew up watching "Space Patrol" in an era when the most powerful computer I owned was my Secret Squadron decoder badge, it frankly amazes me to have casually written this last paragraph.)

Another interesting multimedia combination with The Director involves using the MediaPhile 1.3MP Infrared Control Unit from Interactive Micro Sys- ►

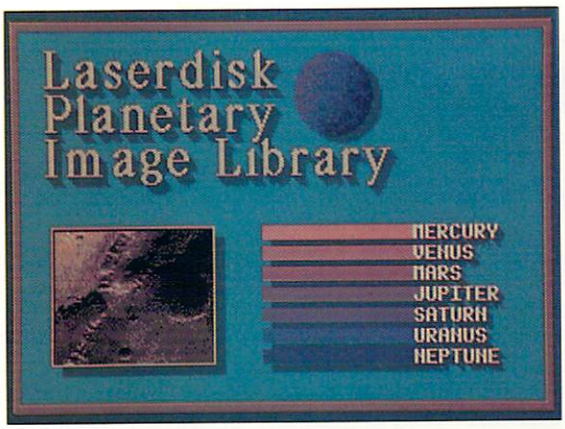


Figure 1. The Planetary Image Library main menu offers a choice of planets to view.

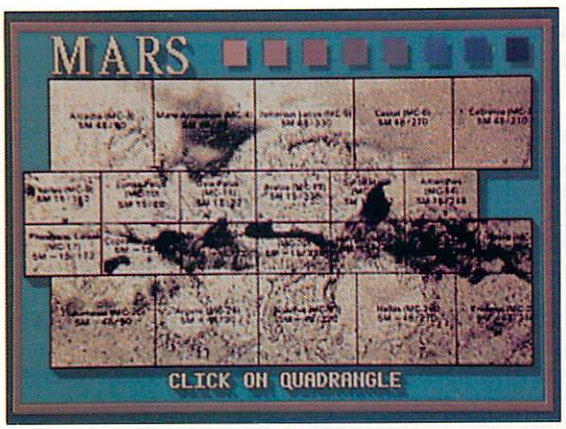


Figure 2. Clicking on "Mars" above triggers branching to this MARS menu.

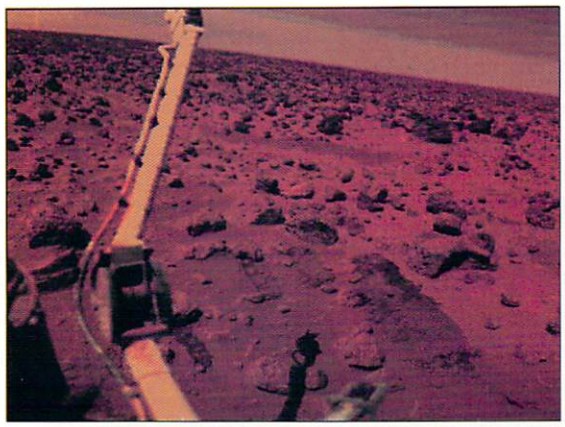


Figure 3. The user finally arrives at the desired single frame of the Viking lander image of the Martian surface.

tems ($195 with software—see May '89, p. 68, for a complete review) and any infrared-controlled device such as VCR, CD, TV, or laser disk players. On the MediaPhile Programmer's Toolkit disk ($149) is a Director module allowing program control of multiple devices using commands relayed via MediaPhile's own infrared LED. Also, you can plug any Sony device with an S port directly into the MediaPhile controller. The control unit connects to the Amiga's second mouse port and is also capable of sending signals to a camcorder or to some film cameras. There are wonderful possibilities for linking several devices to the Amiga and The Director for sophisticated interactive displays at trade shows or science fairs.

The quality and level of graphic sophistication that is now possible in presentations is remarkable. Digitizers and frame grabbers coupled with powerful paint programs make it possible to create beautiful graphics and animations for the computer screen. 3-D and ray-tracing software, CAD, and math-function plotters all add to the range of images and information one can present. The philosophy of The Director is not to replace any of this software, but rather to provide a medium through which it can all be used in concert to create better results. It is now

Up and Running

A "Conversation" with The Director

IN THE MAIN part of this article, we concentrated mostly on *what* The Director could do in terms of presenting and combining multimedia elements. Here we focus on *how* the program operates. The examples that follow are designed to show the various commands and statements and their parameters—the very language of The Director itself—in action.

Flipping Over Animation

Full-screen page-flipping animation can be as simple as the example outlined in the "Getting Started" sidebar. Creating partial-screen page-flip sequences, however, is a bit more complex and relies on The Director's BLIT command (see "Accent on Graphics" #5, p. 50, June '89, for more on BLIT). BLIT, like many other Director commands including WIPE and DISSOLVE, is followed by a series of parameters indicating which portion of the screen to change. For instance:

Example 1

```
REM  Partial-Screen
REM  Page-Flipping Test

LOAD 1,"df1:screen"
LOAD 2,"df1:sequence"
TRANSPARENT 1

50 FOR q=1 to 8
   BLIT 2,10,5,45,130,20,50
   BLIT 2,40,5,45,130,20,50
   BLIT 2,70,5,45,130,20,50
   NEXT
```

This simple animation script looks at a three-image sequence of 20-by-50-pixel rectangles in buffer 2 and transfers them from their x,y locations to location 45,130 in the current destination buffer. The TRANSPARENT statement prevents BLIT from transferring color 0. This allows objects to be "removed" from the background when transferred. A FOR/NEXT loop for the BLIT commands runs the animation sequence eight times before proceeding. Page-flip animation normally BLITs from a series of different source coordinates to a constant destination coordinate as shown. Conversely, moving an object over a background would commonly BLIT from constant source coordinates to a series of different destination coordinates.

Expressive Text

The TEXT command also needs screen coordinates. These are usually provided through the MOVE command, which provides a location for the text to begin. For instance:

Example 2

```
REM  Text Test Program

LOAD 1,"df1:screen"
LOADFONT 1,12,"big.font"

60 DRAWMODE 0
   PEN 1,15
   SETFONT 1
   MOVE 20,110
   TEXT "Who was Asta?"
```

LOADFONT lets you preload as many fonts as you like into RAM, assigning each a font number to be used by SETFONT later. PEN 1 is the foreground color, and is set to palette position 15. MOVE sets the beginning position of the text in the image window of the illustrated screen. Repeating the TEXT command with a darker PEN color will give a highlight effect if you offset it with the command MOVE 22,112.

At the Touch of a Button

Interactive presentations often use on-screen buttons for user response. The next example shows how a screen button can be specified and monitored with an IF/ENDIF statement and GETMOUSE command.

Example 3

```
REM  Button Test
REM  Program

LOAD 1,"df1:screen"
LOAD 3,"df1:pix4"
ABORT 2

5 GETMOUSE x,y
  IF x>127&x<175&y>49&y<93
  GOSUB 40
  ENDIF
  GOTO 5

40 BLIT 3,3,2,18,100,284,57
   RETURN
```

This loads the main screen into buffer 1, and a screen of images into buffer 2 (see Figures 1A and 2A). The ABORT statement terminates the program if a key is hit. The interactive statement GETMOUSE waits for the user's mouse click and remembers the x,y location of the pointer at that click. (GETKEY would wait for a keystroke, holding the ASCII code of that key in a variable.) The IF/ENDIF line checks to see if the GETMOUSE x,y location falls within the boundaries of button A on the main screen. If so, the program executes the subroutine at line 40 which BLITs the top image from Buffer 2 into the image area on the main screen. Each button area on the main screen could be similarly identified and checked each time the mouse is clicked. Each button could branch the program to a different subroutine such as the animation sequence of subroutine 50 in Example 1, or the TEXT subroutine 60 in the Example 2.

possible to go a step further and tie the Amiga into a larger network of devices to create integrated multimedia presentations. With all the options now available, maybe Commodore's recent media pitch for the Amiga as the tool for the creative mind is less hype than you might think. ■

Joel Hagen, one of the founding members of the Right Answers Group, is a graphic artist whose credits and projects span a fascinating range—from art to astronomy, and software development to science fiction. Write to him c/o AmigaWorld, *Editorial Dept., 80 Elm St., Peterborough, NH 03458.*

Manufacturers' Addresses

Right Answers Group
Box 3699
Torrance, CA 90510
213/325-1311

Interactive MicroSystems
PO Box 1446
Haverhill, MA 01830
508/372-0400

Amigo Business Computers
192 Laurel Rd.
E. Northport, NY 11731
516/757-7334

The Sound of MIDI

The MIDI module in The Director's Toolkit is a substitute for The Director's regular sound module. It contains enhancements to the original sound commands, and also provides MIDI note input. It can be used to synchronize Director animations with external MIDI instruments, synthesizers, sequencers, and so on. To select the MIDI module, use the command:

```
MODULE "midi"
```

The Director's SOUND command then allows a program to monitor up to 16 MIDI channels, using specific parameters to indicate the channel and note information, as in:

```
SOUND v,"midi",6,0
```

In this particular command, the variable v is ignored. The "midi" parameter invokes the MIDI module, and 6 is the MIDI channel number the module will monitor. The 0 specifies that only note-on commands will be recognized in this case. This allows you to mask out note-off commands for simple triggering.

Other commands can also make use of this monitoring, as in the following example:

```
SOUND v,"wait"
```

This command is similar to GETMOUSE, and will wait for the next note-on in the specified channel. All other MIDI commands will be ignored. Variable v will return both the note and velocity information. To extract this information, use these computations: note = v%256 (% is The Director's symbol for the modulo operation), velocity = v/256. If velocity is non-zero, then it is a note-on command. An IF/ENDIF statement can determine if the new variable, velocity, is non-zero and branch to a subroutine exactly as outlined in the button test program. Obviously, this is a very simple illustration of the potential of MIDI interaction.

Heavy Interaction

Going a step further to interaction with a Pioneer laser disk player, the LVIDEO module could be made addressable from a Director script with the command:

```
MODULE "lvideo"
```

As with the MIDI module, a series of parameters using The Director's SOUND statement will send and receive information from the player to be used in the program or to control the player. For example:

```
SOUND v,"lv","5438SE"
```

invokes the LVIDEO module with the "lv" command and seeks frame number 5438 on a laser disk with Pioneer's search command, SE. Similar operations could be performed with other commands: PL for play, PA for pause, MF for multispeed forward, and so on. On other commands, the v parameter is a variable that can hold information returned from the player. A subroutine that GETMOUSE or GETKEY branches to could seek frame 5438 of a laser disk, and display that image. The MediaPhile Director module works in a similar way. —*JH*

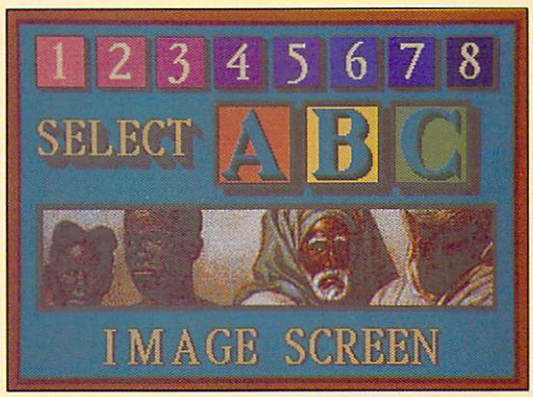


Figure 1A. The main screen of the interactive presentation outlined in Example 3.

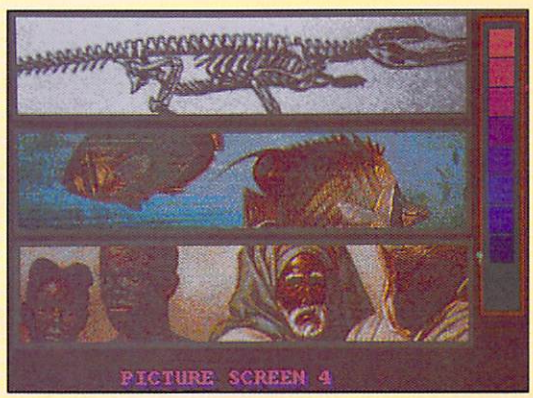
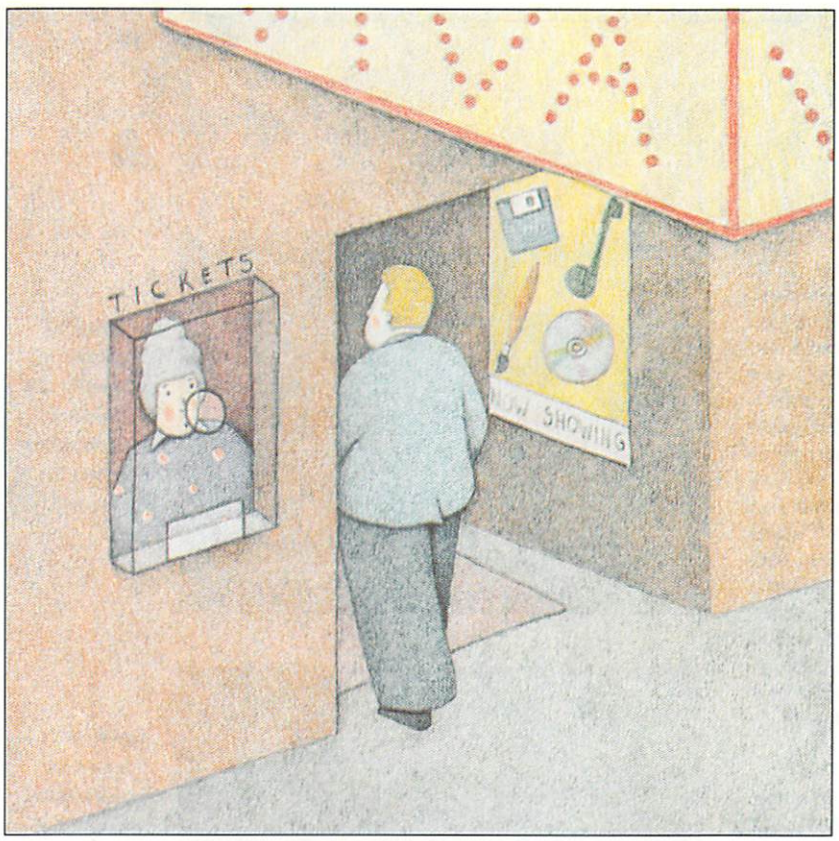


Figure 2A. Screens of images like this one can be triggered by touching various on-screen buttons above.

Author! Author!



With its elegant, easy-to-use interface, the VIVA authoring system could ring down the curtain on other Amiga multimedia-production software. But will the "Hit-in-New-Haven" translate to box-office success on Broadway?

By Geoffrey Williams

THE VIVA AUTHORING package (MichTron, $199.95) represents yet another approach to creating and controlling interactive, multimedia productions with the Amiga. Other methods are covered elsewhere in this issue—from the script-based procedures used by the well-established Director program (p. 25) to the stack-metaphor concept employed by the new UltraCard (p. 38). What sets VIVA apart, however, is its icon interface, which makes creating your own programs very easy. Unlike The Director, which forces you to write script files, VIVA allows you to create presentations by selecting a series of icons.

The ability to work with an icon interface is VIVA's greatest strength, and allows you to create programs in minutes that could take hours if you had to write and debug a script file. The learning curve is very short—I was creating working "stories" (VIVA nomenclature for an interactive presentation) the very first day I started to play with it. While this one feature is such a significant development that it gives VIVA the potential to dominate the Amiga multimedia field, there are, unfortunately, a few serious reservations about the program that must be addressed (and will be in this article).

VIVA: Power to the Users!

VIVA (which stands for Visual Interactive Video Authoring) allows you to write interactive programs that can utilize the images from a laser disk as well as display Amiga graphics and play digitized sounds. By clicking on areas of the screen, blocks of text, or buttons and gadgets you create in a paint program, users can choose what information they want to see or what paths they want to follow.

With the addition of a genlock, you can create programs that combine Amiga graphics and laser disk images. For example, you could display a single frame from the laser disk, and have objects you created in a paint program genlocked on top of it. By clicking on the objects, the user would be sent to a different laser disk image or a different graphic. Because a single side of a laser disk can contain over 50,000 individual images and play any part of them as animations, the possibilities open to you are staggering.

When you first open VIVA, you are presented with ►

ILLUSTRATED BY PAUL YALOWITZ

an upper window, the Storyboard, in which all of the story icons will be placed, and a lower window from which you select the various action icons. These are broken down into several categories, and the bottom row of buttons allows you to select the category you want to work with. When you change categories, a new selection of icons appears in the lower window.

To create a script, all you have to do is click on a series of icons and set their individual parameters. They will then appear in the Storyboard. If there are more icons in the Storyboard than can fit in the window, you can scroll down through them. Figure 1 shows the VIVA main screen with an assortment of story icons pertaining to a wine promotion campaign in the upper window (Storyboard), while the action icons appear in the lower window.

Editing an icon in the Storyboard is easy. Click on it, and a popup menu will appear that allows you to Delete, Move, Edit, Insert, Copy, change the name (each icon is given an individual default name, but you can change the name to anything you want), and test the icon to see what it does. This makes editing very fast, as the menu selections are right where the icon is so that you do not have to go back up to the menu bar. When you decide to delete an icon, the program even warns you if other icons are dependent upon it before proceeding.

Because there are different icons for different functions, it is simple to go through the Storyboard and find the functions you want to edit. The mind can identify a simple shape much faster than it can decode text, so it is much easier to edit in this icon environment than it is with a text-based script file.

There are also three icons at the top of the screen that let you print the Storyboard script, run the Storyboard (which you can do at any time to see how things are progressing), and pop up a laser disk controller. The latter looks just like a VCR controller, with Scan (fast forward or reverse), Step (single-frame forward or reverse), and Play (plays a sequence on the disk). You can also play back in slow and fast motion, mute the right or left audio channels, and search for a specific frame by its frame number or by a chapter number (a laser disk is divided into chapters). You use the controller during the planning stages to find the specific frames you want to use in your presentation.

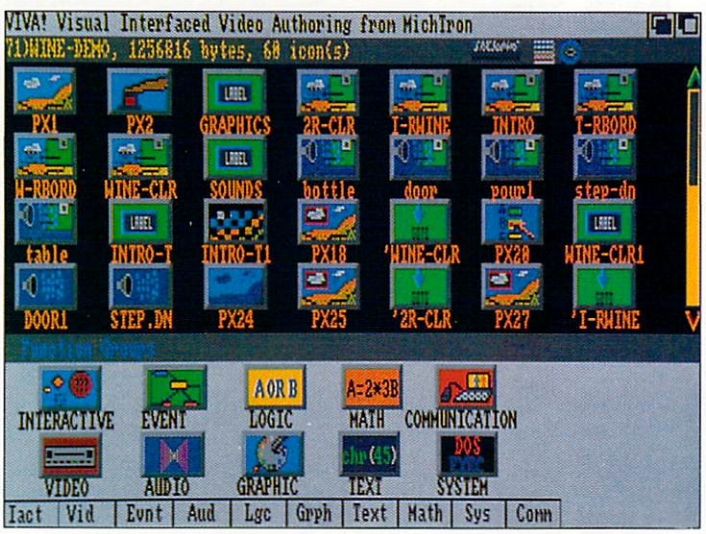


Figure 1. VIVA main screen. Note the sample story icons in the upper window (or Storyboard) and the action icons in the lower window.

The user interface is very elegant, and it is obvious that a lot of time went into its development. I am thrilled with its ease of use and its ability to show stills and animations from a laser disk, but I was disappointed by its poor Amiga graphics support. While it can handle all resolutions including overscan, it does not support animation or color cycling, and has only a few transitions (and not all of them work very well). This is a real shame, as the power of Anim brushes, brush moves, compressed animation, and other exciting Amiga features would be a tremendous enhancement to the power of the laser disk.

The Icon Approach to VIVA Stories

VIVA offers ten basic functional groups of icons from which to choose. The first are found under the **Interactive** icons. These offer a number of ways to let the user respond to on-screen events.

The *Text Hot Spot* icon allows you to create text that can be clicked on to execute a specific function. This provides an easy way to make menu options. You can choose the font type from any in your fonts directory (except ColorFonts), and you can select the text and background colors. You can also have a box appear around the text and/or a text color change to show that it has been clicked on. Although the manual claims that the text can be up to 32 characters long, I was unable to enter more than 16.

The icon that follows the Text Hot Spot icon in the Storyboard determines the action that will take place when that text is clicked on. You can have as many as 1000 text hot spots on a single screen, and you can have text appear over IFF or laser-disk images.

The *Area Hot Spot* icon allows you to draw an invisible box on the screen, so that area becomes a hot spot. This could be over graphics or laser video. A major failing of the program, however, is that you can make only rectangular hot spots. For instance, a common educational application might be to show a cross-section of an object such as a plant and allow the user to click on various parts of the plant to access information about that part. If the individual components are curved, it would be very difficult to create hot spots that do not overlap other parts. This is severely limiting, and free-form hot spots should be included in future updates.

Another missing feature I would like to see is the ability to make an alternate image appear when a hot spot is clicked, so you can have things like buttons that appear to be pressed when you click on them ►

The Essence of Platinum!

Scribble! Platinum Edition

- ☐ 104,000+ word **Spellchecker**
- ☐ Scientific and Technical Supplements
- ☐ **Spell As You Type**
- ☐ Full User Dictionary maintenance
- ☐ 470,000+ word **Thesaurus**
- ☐ Multiple windows
- ☐ Color, Interlace & Overscan support
- ☐ Cut and Paste among documents
- ☐ **Mail Merge**
- ☐ Print **IFF Graphics**
- ☐ Clipboard Compatible
- ☐ Cartridge Font support
- ☐ 512K Required
- ☐ Not copy protected
- ☐ Free Technical Support
- ☐ User Friendly Manual

The Works! Platinum Edition

- ☐ **Includes Scribble! Platinum module**
- ☐ **Full featured spreadsheet module**
- ☐ Lotus 123 wks file compatible
- ☐ Macro-language
- ☐ 40+ built-in functions
- ☐ 68881 math co-processor support
- ☐ 8 graph types
- ☐ Sideways print utility
- ☐ **Flat file manager (database) module**
- ☐ Extensive math capability
- ☐ **Includes OnLine! Platinum module**
- ☐ Clipboard Compatible
- ☐ 512K Required
- ☐ Not copy protected
- ☐ Free Technical Support
- ☐ User Friendly Manual

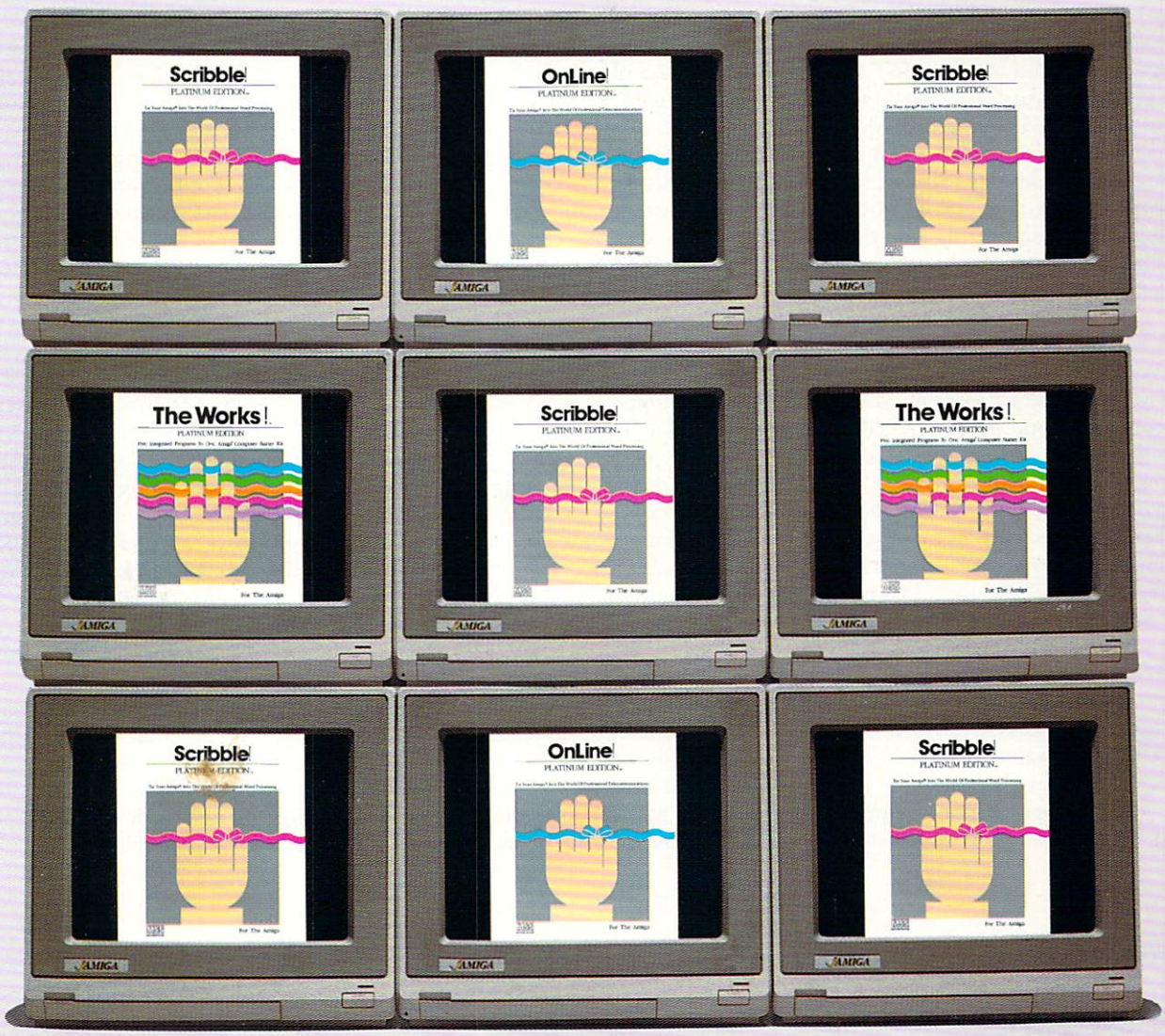
OnLine! Platinum Edition

- ☐ **ARexx support**
- ☐ New Sadie Protocol (simultaneous chat and 2-way file transfers)
- ☐ Color, Interlace & Overscan support
- ☐ VT-100, -52, -102, TTY, ANSI-BBS, Tektronics 4010 emulations
- ☐ X-, Y-, Z-, WX-Modem, CIS-B, Quick-B, Kermit protocols
- ☐ 300 - 57600 bits per second
- ☐ **Multiple Serial Ports**
- ☐ **Internal Modem support**
- ☐ **Full Script Language**
- ☐ **User defined Macro keys**

MSS
Micro-Systems Software

12798 Forest Hill Blvd., Suite 202
West Palm Beach, Florida 33414
407-790-0770 FAX: 407-790-1341

Committed to excellence since 1978



Circle 95 on Reader Service card.

A Working Guide to VIVA

IN WORKING WITH Tony Gomez, who heads the Valley Video Workshop in North Hollywood, CA, the first VIVA project we decided to create was an interactive tour of the solar system based on images from the Voyager I and II missions. (The results of this work, incidentally, will be presented at an upcoming meeting of the International Interactive Communications Society—IICS.) In creating the project I devised the following "8 Step Guide" that may be helpful in preparing a VIVA interactive presentation.

Step 1: The Story

Once you know the topic of your interactive project, you need to make a complete outline of all of the points you want to cover. You will find that an outlining program such as Flow New Horizons Software, $99.95) is very helpful, as it allows you to organize your material as heads and subheads, displaying only certain levels at a time. This lets you make a rough simulation of the way the material will be organized in your interactive presentation.

Step 2: The Laser Disk

Obviously, you will need a laser disk with which to interact. Creating your own is a complex proposition, requiring very high-quality video-source material, careful planning, and considerable expense in having the laser master disk made. For most people, it is much more practical to use an existing laser disk. This is known as "repurposing," and the VIVA manual lists several Videodisc manufacturers with discs available on a variety of subjects.

For our tour of the solar system project, we chose a laser disk created by the Optical Data Corporation that sells for about $100. It contains thousands of images from the Voyager missions, along with many exciting animated sequences.

Step 3: Choosing the Images

Next, we went through the laser disk to find images and animations that would illustrate our main points. Laser-disk players give you the option of displaying the frame numbers, so when we found an image we wanted to use, we wrote down its frame number and a short description. For animation sequences, we also jotted down the frame number we wanted the animation to end on.

Creativity is important, as you will probably not find the precise picture you want to use on the laser disk. You may be able to genlock Amiga graphics over an image, masking out the parts that you do not want to show. We found some great images that showed size comparisons of the planets and moons, but the top half of the screen was filled with technical information that had no place in our story. We genlocked graphic images over these data sections, which gave us screens that we could use.

Step 4: Creating a Flowchart

First, find a large piece of paper. A flowchart of even a simple story can get very large very quickly. The flowchart will be your road map to the design of your presentation.

The flowchart in Figure 1A shows only the complete branchings for one of the planets, in this case Jupiter. The organization of how the different screens interact is known as the program logic, and it is important to keep careful notes to avoid confusion. The numbers in each box signify the laser-disk frame number for that screen.

We decided to standardize the different screen types. We called our first screen, consisting of three planets, the Main screen. Most of the other screens contain a button labeled "New Planet" that will return you to this Main screen. By clicking on one of the planets, you bring up the appropriate Planet Data screen. Each of those three screens includes information about the selected planet as well as three buttons along the bottom: "New Planet," "Planet," and "Moons." "Planet" takes you to a Planet Option screen with two large buttons offering a "Narrated Flyby" and a "Planet Rotation." These two buttons play the corresponding animations. At the end of the animations, the laser disk freezes on the last frame and a genlocked set of buttons is displayed: "Repeat," "Moons," and "New Planet." "Repeat" replays the animation, while "Moons" takes you to the Moons Option screen.

While there are three Planet Option screens and several Moons Option screens, each screen type has a consistent design: The same buttons take you in similar directions in every instance. As you can see from the partial flowchart (Figure 1A), things get crowded very quickly, but because all Moon Option or Planet Data screens work in the same way, we can easily figure out how to program them if we lay out one complete path in the flowchart. This makes it much easier to check and debug program logic.

Step 5: Creating the Graphics

Working from the flowchart, make a list of all of the graphics screens you will need to create. It is important to devise a standardized way of naming the graphics. The name should tell you what the picture is, and what type of screen it should appear on. For example, the graphic with the buttons to be genlocked on the Jupiter data screen is called "Jupiter.PD.buttons," the PD signifying that it is a Planet Data screen.

Try to design buttons, text screens, and other graphics elements so that there is a consistent overall design, and so that the graphics match the style of the laser-disk material. All the elements should look like they belong together.

The only way to properly create the graphics is to use the laser controller utility in VIVA to bring up a video frame, and then genlock your paint program over it. This lets you align graphics properly and to see what they look like in NTSC. If you are working with DeluxePaint III (Electronic Arts, $149.95) in overscan, make sure you hit the F10 function key to eliminate the title and tool bars; otherwise you will be looking at an image that is shifted down several scan lines from the way it will appear in your story.

VIVA multitasks easily with paint and other programs. By selecting a pull-down menu from within VIVA, you can type in a pathname for a specific program you want to run when you access that pull-down menu later. There are options for setting seven different programs in pull-down menus.

If you want to do wipe transitions between graphics screens, the smartest thing to do is to use the same color palette for the entire presentation. Images with different palettes will give you strange colors as the second picture's color palette becomes active during wipe effcets. While you may start by carefully planning to use only pictures with the same color palettes together, if you later decide you need to rearrange the images, your careful planning will go out the window.

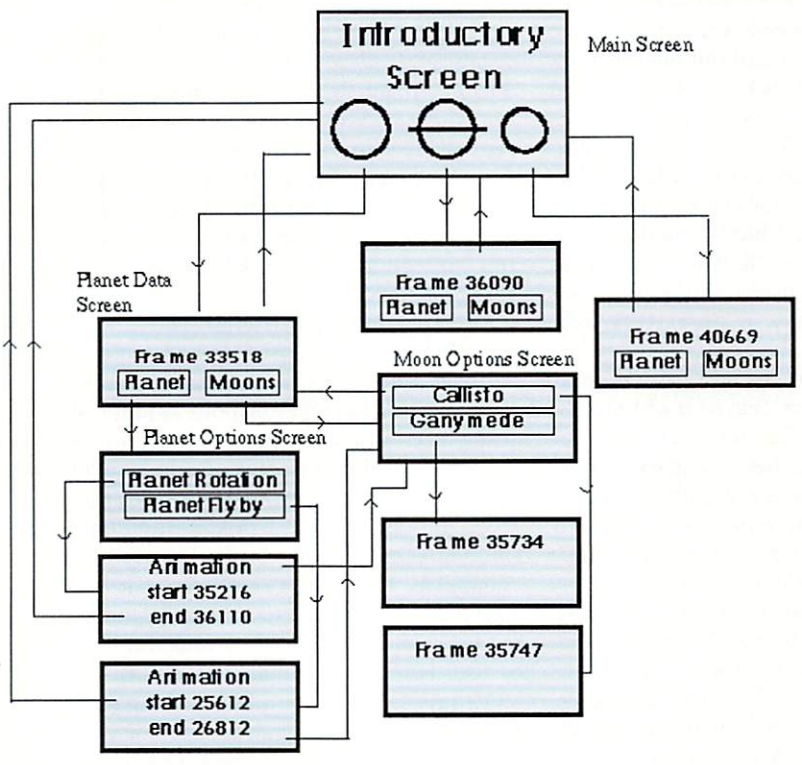


Figure 1A. Using a flowchart (such as this one showing the complete branchings for the planet Jupiter) will make organizing how the different screens in your presentation interact—the "program logic"—much easier.

Step 6: Programming in VIVA

The manual contains lots of good advice on programming structure. Pay special attention to labeling, and read the "Tips and Tricks" section in the Appendix.

Step 7: Fine-Tuning

Once you have your basic story working in VIVA, you can start adding extra features. For example, we decided that some people might not want to watch a complete animation sequence, so we made the entire screen a hot spot during animation playback. By clicking on the screen, the user can stop the animation and return to the Planet Data screen.

It would have been nice here to be able to use a frame grabber, which would have enabled us to capture graphics from the laser disk and manipulate them as graphics files. This would have made it possible to create graphics composites, such as of the three planets on our Main screen, that resemble the digitized images from the laser disk. Instead, we had to create a relatively crude graphics screen for the Main screen.

Step 8: Real-World Testing

While *you* know that your presentation works well technically, it still may not be easy to use for someone who does not know how it is set up. Test your presentation on several people who are unfamiliar with interactive media and your subject matter. Observe how easily they are able to move around, notice if there are sections that are frequently skipped over, and get feedback from them on what they like and dislike. As perfect as you might have thought it was, chances are you will need to make additional changes based on this feedback. Such testing will help insure that your finished product is a useful and exciting interactive presentation. □ ***—GW***

and other simple forms of animation. This is more of a frill than a necessity, but the more graphics sizzle you can add, the better.

To create questions on the screen to which you want a keyboard response, you first create the question as text in an IFF image. The *Ask* icon allows you to set the question as true/false, yes/no, or multiple choice, and to place a prompt, such as "Enter T or F only," anywhere on the screen. After giving each question a name, you then put an *Answer* icon for each possible answer to that question elsewhere in the script. If the user responds with a "y" for yes, the program jumps to the section of the Storyboard containing the Answer icon with the "y" response set for that named question. Everything after that Answer icon is then executed. (For those who program, this is like a conditional GOTO—but with VIVA you need not worry about such things.)

The **Video** icons offer you a range of controls over the laser disk player. You can play animations from the laser disk at a variety of speeds, play them to a specific frame and stop (even while other parts of the Storyboard are being run), and show single frames.

The **Graphics** icons allow you to load and display Amiga graphics, and there are a few transitions possible. *Fade* with adjustable speed works well, while *Tile* provides effective spiral in and out effects and a good-looking checkerboard transition. In the initial release, the *Wipe* effect did not work well, but it is very smooth in version 1.02. You can wipe right, left, up, and down. The *Push* transition is jerky and has too much "artifacting" (additional colors appear that do not belong) to be useful, while I could not get *Dissolve* to function properly at all. The *Blinds* effect is slow and you do get some minor artifacting when you use it to wipe in a picture; it works very well, however, when wiping out to black.

The **Text** icons allow you to display text. *Place Text* works exactly like the Hot Spot Text icon without the hot spot. You can load text in and display it in a scrolling window. The problem here is that users must understand how to work the scroll gadget, and to click on the close gadget when done—there ought to be a better way to handle this. A *Print Text* icon enables you to supply users with hardcopy of the text.

The *Speak Text* function has an odd habit of ignoring periods and leaving out the pauses. It also exhibits the more commonly encountered problem of pausing at the end of each line rather than at the end of each sentence. These two quirks combine to make it sometimes difficult to understand. A workaround would be to make sure that your sentences end at the end of each line. Each line of text is displayed as a small window at the bottom of the screen as it is being read, so you cannot get away with phonetic pronounciation. The easiest solution is to get a copy of the public-domain program BetterSpeech, which allows you to create an exception table that improves the pronunciation. It would also be nice if the program allowed you to display a graphic as text is spoken.

There are also several icons for putting in pauses or delays: The *Keyboard* icon waits for a specific key to be pressed; *Keywait* puts the words "press the escape key" on the screen and then waits for you to do so; *Time* waits for a specified amount of time or for a specified number of video frames to go by; and *Time Wait* branches to a specified part of the Storyboard if the time runs out before the user responds.

The **Event** icons give you a basic range of operational functions such as If, Else, Goto, Return, End, Until, and Break. *Masterloop* will return the program to the beginning, while *Label* allows you to label different sections of your Storyboard. The **Logic** icons provide operations such as Greater Than, Equal To, And, Or, Xor, Negate, Less, and Less Equal, while the **Math** icons provide a dozen math functions.

VIVA Summary

VIVA provides you with all of the basic tools to create effective interactive presentations. Its real power, though, comes from its ability to control a laser disk. To take advantage of this, you will need a professional unit with an RS-232 port, such as the Pioneer LD-4200 player (about $1000).

VIVA'S 263-page manual is thorough and well organized (with an index). It also provides an introduction to interactive hypermedia and a glossary. Appendices offer suggestions for further reading, a vendor list with sources for laser disks, players, and even an Amiga-compatible touch screen, and a list of support organizations, including the IICS (International Interactive Communications Society—which you can contact at 2410 Charleston Rd., Mountain View, CA 94043, 415/922-0214).

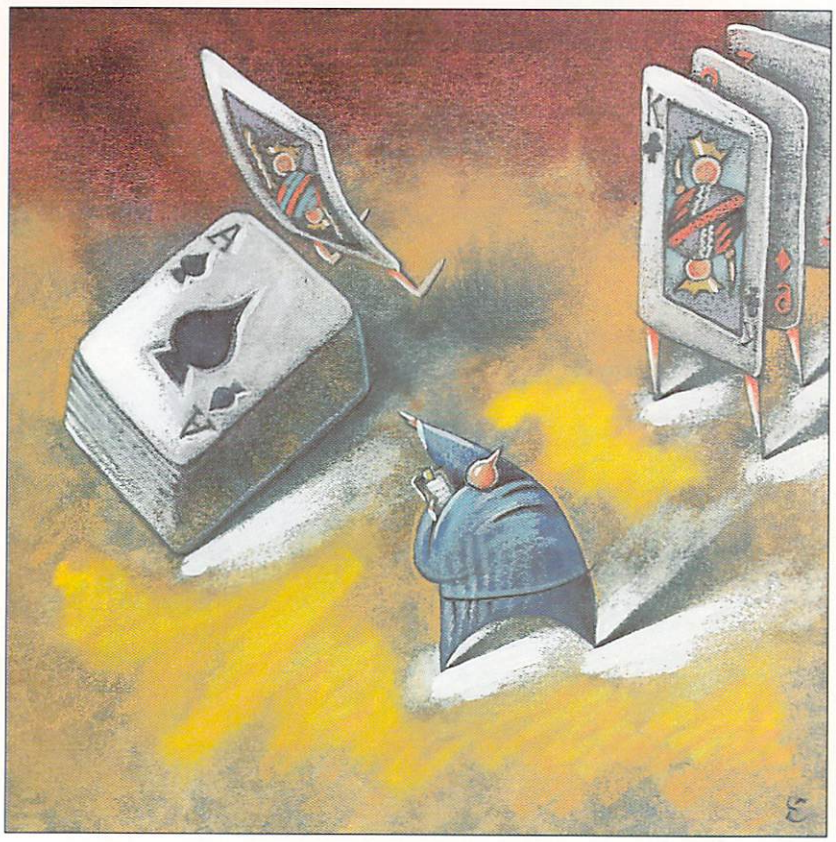
A recent announcement that The Director will soon provide laser-disk support means that it will become an even more powerful multimedia tool. The program already has an enormous amount of flexibility and power when it comes to manipulating Amiga graphics. I feel that the complexity of programming Director presentations, however, still puts it out of reach for many. While VIVA may not (and probably should not try) to offer the kind of sophistication and complexity found in The Director, it does need to offer a wider range of graphics and animation options to fully capitalize on its marvelous simplicity and ease of use. Fortunately, thanks to the elegance of the VIVA interface, adding additional capabilities should not make it more difficult to learn or use.

As it is, VIVA is one of the easiest interactive programs to work with on any computer. With further enhancements, I feel it could be one of the best interactive development tools available. ■

Geoffrey Williams is Executive Producer for Creative Business Communication and head of the Amiga Video-Graphics Guild. Write to him c/o AmigaWorld, *Editorial Dept., 80 Elm St., Peterborough, NH 03458.*

Play Your Best Hand:

Building a Presentation with UltraCard



Stack the deck in your favor with these tips on designing a multimedia project.

By Michael Hanish

FOR A WINNING presentation, you must combine ordinary information in extraordinary ways. UltraCard (Intuitive Technologies, $50) gives you the tools you need. With it, you design an interface to communicate with external programs and present text, graphics, sounds, and speech. You can specify every detail of how the information is to be displayed—link screens in information trails, manipulate data and move it between programs. In effect, you can create a new program without being a programmer. And to prove it, just follow along in the sample project below.

Like its Apple cousin Hypercard, UltraCard is based on the stack metaphor. Imagine a stack of file cards, all different in appearance and content. Each card can hold pictures, text, sounds, numbers, or any combination of these. You can flip through the cards, viewing them in any order. In UltraCard, a card is called a frame and is made up of a frame layer overlayed on a backdrop layer, like a sheet of acetate. The backdrop can include IFF pictures and objects (buttons, text fields, and so forth) that remain constant in several frames. The frame layer contains the objects that are specific to each frame. When clicked on, objects can do everything from jumping among frames and stacks to performing complex calculations to describing the screen's contents. You design what the object will look like and, through a script associated with the object, the action it triggers.

Know the Rules

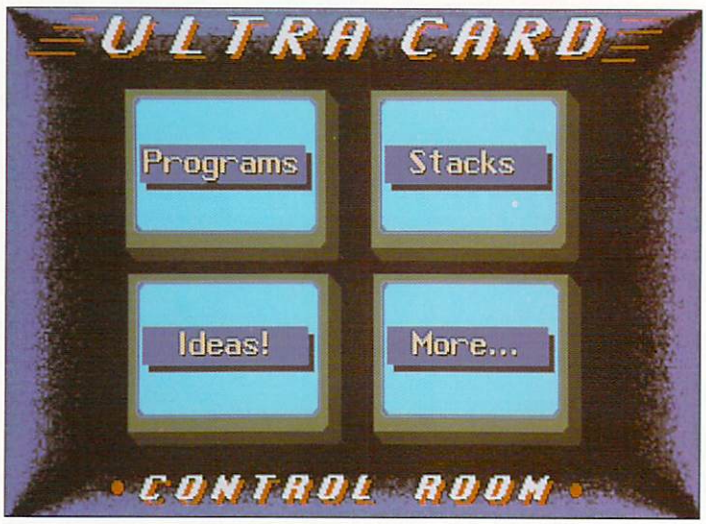
For an example project, I designed an interactive stack to use in the basic reading and writing courses I teach for adults. Each frame shows a picture, says the word associated with the picture, and asks the student to type it. After checking for correct spelling, the frame shows a list of related words that lead to ►

ILLUSTRATED BY RICHARD STEADHAM

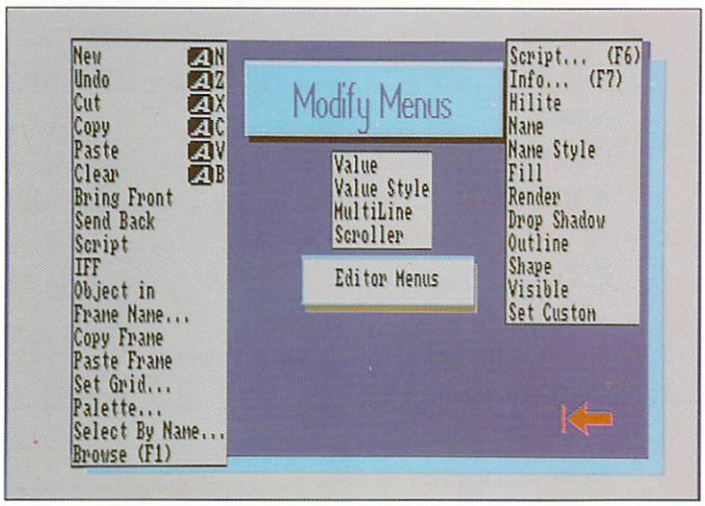
other frames. For example, from the word "round" a student could jump to a picture of a bicycle, a basketball, or a pizza. I also included an index and a help frame.

I quickly learned advance planning is crucial to success. Start by thinking of what you want to communicate. List all the information, then start drawing lines to link the various bits. You not only must decide what information you want to display but also how it should interrelate with the rest. Information need not be presented in only one preset sequence. In a stack, users can follow an information trail through all sorts of paths, according to their interests. Directions for fixing a car engine should progress step by step. Details about how the parts of an engine work can be presented and discussed in a number of sequences. Keep this concept in mind when designing your stack and deciding on links.

Each time you run UltraCard, the program opens into the Control Room, an overview stack. If



The Control Room points you in the right direction . . .



. . . and the on-screen help shows what you can do once you're there.

UltraCard is on your hard drive, you must set search paths so the program can find other stacks and programs. UltraCard offers no drawing tools, but does let you specify a path to and preferences for a paint program, so you can jump to it automatically, work on a backdrop, then import the results back into UltraCard and your stack. A similar option lets you jump to a text editor. If you wish, you can redesign the Control Room with the built-in functions. The help stack is never further away than the Help key.

UltraCard's two main modes are Browse and Modify. In Browse mode you move around in a stack while creating it or navigate through a finished stack to use it. Modify mode contains all the creation and editing tools. You can switch between the two with F1 (to Browse) and F2 (to Modify).

My design called for one basic backdrop that would display a graphic in the center and have buttons for moving around the stack. The first step was to make a backdrop, an IFF picture in any resolution with up to UltraCard's maximum of 64 colors. I experimented with Extra_Halfbrite mode (64 colors) and discovered the program is unstable working with that many colors. Keep to a 32-color maximum. In version 1.4, lo-res overscan mode garbles the backdrop. Word from Intuitive Technologies is that both these features will be fixed in version 1.4.2.

A test of the graphic fill feature, which fills an object with part of an IFF picture, showed that the backdrop palette dominates the frame. Make several backdrops with varying palettes that are compatible with the graphic fills you plan to use.

Builder's Permit

To begin making a stack, select New Stack from Browse mode's Project menu, name the stack, and set the resolution and number of colors. The program will grind away for a while setting up the file for the new stack, then present you with a message directing you to move into Modify mode. Follow the prompt, then chose IFF/Import from the Edit menu to load the first backdrop. You can give each frame a unique name, which makes it easier to keep track of what is where in a complex stack. Frame naming takes effect, however, only after a round trip through Browse mode.

Press ESC to place an object in the frame layer or SHIFT-ESC to place one in the backdrop layer. The cursor will turn into a small set of cross hairs; click and drag out a rectangle where you want the object to be. Objects have properties (position on screen, appearance, fill, shadow, name, and so on) and value (text-string or numerical, which stores the data for manipulation). Choose these by multiple trips to the menu bar while still in Modify mode.

After you decide how the object will look and what (text, an IFF image, or nothing) it will hold, you must tell it what to do. I wanted the buttons at the bottom and in the upper-left corner of the screen to take the

user to another specific frame. When the user clicked on the graphic fill, I wanted a voice to say the name of the picture and ask for the user to spell the word. You issue your instructions via UltraTalk, the scripting language for objects, frames, and stacks.

In UltraTalk's built-in editor, you can type commands or cut and paste them from existing stacks. To move from frame to frame, use the JUMP statement, which demonstrates the importance of frame names. The program names each frame in numerical order as you create them and uses these names (or ones you substitute) for the jump addresses. (For a further discussion of UltraTalk, see the accompanying sidebar.)

When you have the first frame suitably equipped, you can easily reuse the same backdrop for the next frame. Highlight the Frame Add selection from Browse mode's Edit menu. Rename this copy of the previous frame, clear any buttons that do not carry over to the new frame, and add what you need. To add a new backdrop, use the Frame Add New BD selection from the same menu. This creates an empty frame and positions the cursor on it, so you can start from scratch. You can shift objects between layers by clicking on the object then choosing Object In from the Edit menu in Modify mode. If you intend to use the same backdrop and many of the same objects from frame to frame, however, place the repeated objects in the backdrop layer to avoid having to redo them for every frame. To temporarily store frames ►

An Instructive Talk

THE POWER BEHIND the scenes in UltraCard is the scripting language, UltraTalk. Every time you create an object, the program links it automatically with a script that provides the backstage directions for the object's performance. UltraTalk boasts 80 statements (BASIC-like commands), 23 expressions (arithmetic and logical operators that combine constants and variables), and 25 functions, as well as the possibility for user-defined functions.

You access the UltraTalk editor from Modify mode by double-clicking on an object or by selecting it (a dotted outline will appear around it) and then pressing F6 or choosing Script from the Properties menu. The best way to get a feeling for how UltraTalk works is by looking at some of the sample scripts included with the program. Choose an object that does something you find interesting and review the script statements that control it.

You can test statements of your own in Chat mode, which you enter via the Go menu in Browse mode or by pressing RIGHT-AMIGA-T. A one-line window will open at the bottom of the screen. Type in a statement and press RETURN to see what happns when it executes. Chat mode is handy for previewing commands to be sure they work before you commit them to your script.

One of the most frequently used statements, JUMP takes you to a designated frame, possibly even in another stack. Use it with the PUT statement to set a transition between frames or stacks. To do so, you use PUT to assign the number of the transition you want (15 variations are listed in the manual) to the global variable Visual.Effect. Use PUT twice more to place numbers (from 1 to 10) into Effect.Speed and Effect.Amount, which is the number of pixels from the new image that will appear on the screen at each step of the transition.

To branch to other programs, you use the CLI, WORKBENCH, and AREXX statements. CLI and WORKBENCH allow you to run independent programs as you would from the Command Line Interface or Intuition environment. AREXX starts an ARexx script.

The various FILE commands (GET, OPEN, READ and CLOSE) display text files in a multiline object, but you can make your presentations more vocal with the SAY and SOUND statements. SAY invokes the Amiga's speech synthesizer to read a specified string into the translator (where you take your chances with phonetic pronunciation), or if the string is preceeded by a tilde (~), sends the phonemes directly to the narrator device. The SOUND statements (LOAD, PLAY, STOP, WAIT, UNLOAD) let you play back IFF or FutureSound samples, with complete control over their rate, volume, and number of repeats. By using the ASYNC option while playing back a sample, you can execute another statement, such as a display, simultaneously.

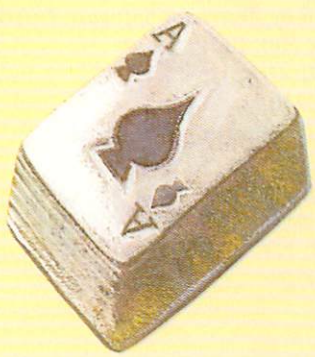
Consider the following button script from one of the sample stacks:

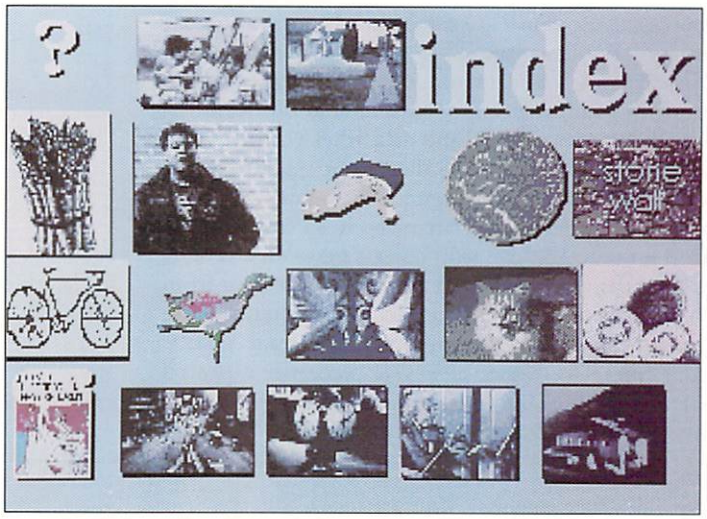
```
PLAY buzzer
PUT 1 into visual.effect
CURSOR wait
PUT 1 into fade
JUMP to frame 10 with effect
PUT 0 into fade
CURSOR ready
```

When you click on the button, the program executes the lines above: plays a digitized sound called buzzer, fades out the current screen, jumps to the destination, and fades it in.

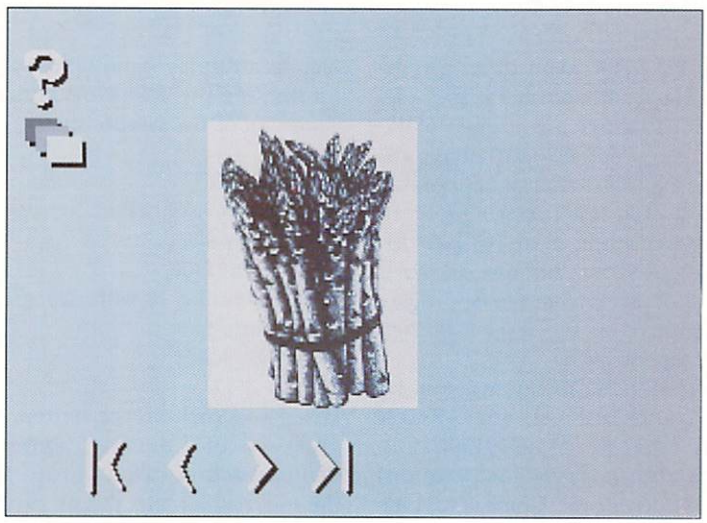
These are only a few of the possibilities offered by UltraTalk. Some familiarity with elementary programming techniques is helpful, but because the language is so straightforward, you can quickly discover how things work. □

—MH





The example stack's index card: Click on a picture to jump to a frame.



A typical screen in the sample project.

Manufacturers' Addresses

Intuitive Technologies
distributed by American Software Distributors
RR 1 Box 290, Bldg. 3
Urbana, IL 61801
217/643-2050
408/646-9147 (technical support)

Bantam Books
666 Fifth Ave.
New York, NY 10103
212/765-6500

and objects, copy them into a clipboard and then paste them into place when needed.

For my project, I made and linked each frame in the order I wanted them. I then returned to each frame, adding UltraTalk's SAY statements, devices to check spelling input, and statements to display the list of associated words. Working in this order allowed me to make sure that each step of the process worked before moving on to the next.

Each time you edit a stack, it grows in size. By using the Compact Stack choice from the Project menu in Browse mode, you can eliminate a stack's wasted space, sometimes reducing its size by half. When you save the compressed version, it takes the original file's name while the fat stack's filename gains the suffix .old.

Generation Gap

Version 1.4 of UltraCard suffers from its ancestors. Version 1.1 barely worked and had a minimal manual. I'm glad to report 1.4 has almost all the kinks worked out and an expanded manual. Be warned, however: Incompatibilities among versions cause some stacks created with earlier editions of UltraCard to corrupt when you load them in a newer version. The developer, Mike Lehman, promises to fix any corrupted stacks you send his way.

The program is modeled after, and begs comparison with, Apple's Hypercard. Overall, UltraCard bears up well to the test. It is more flexible than Hypercard because of the Amiga's multitasking and color graphics. On the negative side, the program sometimes feels a bit slow getting information to the screen, and too frequent trips to the menu bar impede the process of creating a stack.

The manual could use a major expansion to explain many of the terms that will be unfamiliar to nonprogrammers. Until then, study the assortment of stacks that comes with the program. Find the features that do what you want, then cut and paste them into your own stacks or model your features after them. A less obvious source of information is *The Complete Hypercard Book, Second Edition* by Danny Goodman (Bantam Books, $29.95). While the book is specific to Hypercard, many concepts and features are similar to UltraCard.

Quibbles aside, UltraCard is relatively easy to work with for such a complex program. Its price of $50 is a bargain for the multitude of things the program can do for and with you. ■

Michael Hanish uses his Amiga for video and graphics work with both his performance group, The World Turned Upside Down, and his adult literacy students. Write to him c/o AmigaWorld, *Editorial Dept., 80 Elm St., Peterborough, NH 03458.*