8 Logs and Scripts

SciAn has a simple scripting mechanism which stores sequences of user operations in files. Script files are normal ASCII text files containing lines of text. Usually, one line is used per operation. In a few cases, such as dragging and dropping icons, more than one line is required to complete the operation.

A script can keep track of every operation that can be done within windows, through menus, and by direct manipulation. Scripts refer to windows on the screen and objects within windows by name. They reflect user interface operations, such as setting a control to a certain value or moving an icon from one window to another.

The words “script” and “log” really refer to the same thing: a sequence of SciAn operations kept in a text file. They are called “scripts” when the file is being read into SciAn and are called “logs” when SciAn is recording interactions into a file, but when speaking of the files themselves, the words are interchangeable. Although it is possible to write scripts by hand, it is most effective to let SciAn save a log and then modify the log afterward if necessary, such as to make a movie. Usually SciAn does about 95 per cent of the work and the user contributes the remaining five per cent.

Making logs

To have SciAn log its operations to a file automatically, include the `-l` switch with a file name on the SciAn command line. For example,

```
scian -l test.log
```

Everything which is done within SciAn will be logged to `test.log`. You can use the `-l` switch as many times as you like on a command line, and the commands will be logged to all files. If the file name is a single hyphen (−), then the log will go to standard output. This is sometimes useful for debugging scripts.

Reading scripts

To have SciAn read a prepared script, include the `-s` switch followed by the name of the script file on the SciAn command line. For example,

```
scian -s test.log
```

SciAn will read the lines within the script.

It is possible to read from a script and save the log to another file at the same time. This is useful when revising scripts. Just include an `-s` argument and an `-l` argument on the command line.

Note *When reading scripts, it is a good idea to place the* `-s` *switch first on the command line. This is because subsequent arguments, such as file names, may cause new windows to be opened. Once SciAn has seen the* `-s` *switch, it will place windows automatically rather than manually.*

While the script is running, you may not interact with SciAn. To abort a script while it is running, simply click in a window. You will see a dialog window asking if you want to abort or continue the script.

Normally, errors in scripts cause the script to abort. This can be overridden by the `-C` flag on the command line. (For a complete description of command line arguments, see Chapter 2 of the reference manual.)

A sample script

The following is a sample script, generated automatically by SciAn during a session very much like the quick tour in Chapter 1. Annotations in the form of comments beginning with a hash mark (#) have been added by hand after each command to explain what it does. Comments are ignored by SciAn when it is reading a script.

This example should give you enough of a feel for how scripts work to prepare you for the next chapter on making movies. A more detailed discussion of each script command is given in the reference manual.

```
# The first line in a script gives the version of SciAn which
# generated the script. You will get a warning message if you
# try to read a script with an incompatible version of SciAn
Version 0.661

# The second line gives the time the script was generated. This
# line is totally ignored by SciAn. It's just there to give you
# a record of when the script was written
time Mon Oct  5 14:22:51 1992

# All script operations happen within a certain window. The
# window command specifies which window to use for subsequent
# operations. In this case, the title window, which is called
# "SciAn," is to be used.
window SciAn

# The user has selected New File Browser from the File menu. This
# script command does the same thing, which will generate an Ask
# dialog window
Show FilesWindow

# The user has entered "demo" in the box in the dialog window and
# has pressed the OK button. These script lines do the same
# thing. Note that the text entry and button pressing is done on
# the level of setting the value of named objects rather than
# storing each keypress or mouse click.
window Ask
```

```
set value Reply "demo"
set value OK 0

# Go to the resulting file browser. The backslashes (\) are escape
# characters, which force the next character to be included as
# part of the name.
window \/scri4d\/a\/users\/pepke\/sciantest\/demo

# Move the scroll bar to show more files
set value VScroll -51.4615

# Open three files: W.stf, Terrain.stf, and Z.stf
open W\.stf, Terrain\.stf, Z\.stf

# Go to the Datasets window and visualize the selected datasets.
# The user has actually selected the three datasets and pressed
# the "Visualize" button, but only the actual act of visualizing
# is stored in the log. When reading scripts, by default
# selections do not appear. This behavior can be overridden
# by the -S command line argument.
window Datasets
visualize W, Terrain, Z

# Over to the new visualization window
window Visualization_1

# The user has rotated the window to get a better view. The
# observer has made a snapshot of itself, which is a way of
# setting several variables as one unit. These four variables
# uniquely define the position and orientation of the observer,
# as well as its distance of focus. Angle of view and clipping
# planes are specified elsewhere
begin snapshot observer
    set variable LOCATION [0 -4.73743 1.59899]
    set variable FORWARDVECTOR [0 0.947486 -0.319798]
    set variable UPVECTOR [0 0.319798 0.947486]
    set variable FOCUSDIST 5
end snapshot

# Turn off rotation inertia. Intertia is specified by a speed,
# in degrees per real-time or video-time second, and an axis of
# rotation, specified as a unit 3-vector. When the speed is 0,
# the axis doesn't matter, so an arbitrary unit vector is there.
set rotation 0 [-1 0 0]

# Show the controls for the observer in the visualization window.
# Again, this resulted from selecting the observer icon and
# choosing a menu item, but only the actual action is saved to
# the script.
show controls Observer_1

# Go to the observer window and bring the object closer. Changing
# the perspective control simultaneously moves the observer and
# changes the focus distance, effectively moving the observer
# toward the focus point. The four numbers in the vector are
# focus distance, angle of view, and near and far clipping planes.
window Observer_1
set value Perspective_Control [2.86885 25 0.1 8]

# Show the controls of Isosurface 2 (for Reflectivity or Z)
# and go to the window. The underscore (_) stands for a space.
# To include an explicit underscore, use backslash-underscore (\_)
```

```
window Visualization_1
show controls Isosurface_2
window Isosurface_2

# Press the Color icon button, which goes to the Color controls
set value Color 1

# Turn on transparency and pull down the brightness a bit
set value Transparent 1
set value Brightness 0.676471

# Show the controls for the other isosurface (for upward velocity
# or W) and go to the control panel. Again, go to the color
# controls.
window Visualization_1
show controls Isosurface_1
window Isosurface_1
set value Color 1

# Set the value of the color wheel to a bright red
set value Fixed_Color [1 1]

# Go back to the visualization and select the annotation tool.
# This isn't really needed when the script is read, but it is
# included for completeness.
window Visualization_1
set value Space_Tools 4

# Create a new annotation called Annotation 1 at the specified
# coordinates. This is the result of the user dragging out
# an annotation after having selected the tool
annotation Annotation_1 [36 378 308 390]

# Go to the panel selection, or "finger" tool
set value Space_Tools 3

# Change the text in the annotation. This results from the
# user typing in the text.
set value Annotation_1 "Socorro thunderstorm"
```

9 Animation

SciAn is designed to be able to make simple animated movie scenes quickly and easily. Although it currently lacks many of the features of animation packages developed for the entertainment industry, such as keyframe editing, it is straightforward to make a movie scene which only involves time progression and rotation of an object. More complex camera moves can be done with a little more effort.

SciAn uses a technique called frame-by-frame recording. With this technique, SciAn is set up to produce a series of frames, each of which is a single image of the visualization at a point in time. It may take anywhere from a second to a half minute to generate and save each frame. Once all the frames have been collected, they can be played back in real time, usually at 30 frames per second.

Most commonly, this is done by means of a video recorder connected to the video output of the workstation. Video recorders are controlled by SciAn objects called recorder drivers. It is also possible to use film recorders rather than video recorders, but this has not been done yet with SciAn.

Alternately, the frames can be saved to a series of IRIS RGB image files. Image files are created with a screen snapshot recorder driver as well. The image files can then be sent off to a remote recorder driver or film recorder or can be shown in sequence on the screen. There is at least one free program available via anonymous ftp from `sgi.com` to show a sequence of images on the screen.

The process of creating an animation involves setting up a script. First SciAn is run to save to a log the commands needed to set up the visualization correctly. Then the script is modified by hand to include recording commands. After making sure that the script is correct, the script is run with the workstation connected to the recorder to produce the animation.

The remainder of this chapter describes each of these steps in detail.

Connecting a video recorder

The first step in connecting a video recorder to a workstation for animation is to acquire a video recorder. The recorder must be frame accurate. This means that it must be possible to record single frames accurately, one after the other. It must accept the video output of the workstation. You may need a scan converter to do that. Finally, it must be possible to control the video recorder directly through a serial port connection to the workstation.

Most home and industrial videotape recorders do not have these capabilities. There are some videotape recorders which do, but these can be awkward. Most of them require several seconds of roll before recording each frame. Not only does this take a lot of time, but it is hard on the recorder mechanism and on the tape. A much better choice is a write-once laser videodisc recorder. These devices are more expensive, but they record high-quality images very quickly and are much easier to scan for playback. SciAn currently supports two such videodisc recorders: the Sony LVR-5000 and CVR series, and the Panasonic TQ-2026F.

In general, a separate animation controller is not required. SciAn can control the recorder directly.

Important *If you are not the system manager of the workstation you intend to use for recording animations, ask the system manager to read this section and help connect the recorder.*

Connecting the serial control signal

SciAn controls the video recorder through a serial port connection between the workstation and video recorder. A cable goes from one of the serial ports on the back of the workstation to a serial port on the back of the video recorder. The exact wiring of the cable is specific to the systems being used. Consult the workstation and recorder driver documentation to find out about this.

SciAn communicates through the serial port via a UNIX device driver. Device drivers are located in the `/dev` directory. Typically, there is one or more device drivers for each physical port on the back of the machine. The device used by the recorder driver is given by an entry in the recorder driver's control panel, described later. The control panel also allows the baud rate of the connection to be set.

Note *The serial parameters other than the baud rate are set within the recorder driver. Existing recorders use 8 data bits, no parity, 1 stop bit when the baud rate is 1200 or higher, 2 stop bits when it is 300. This can be changed only by editing the recorder driver source code.*

Consult your workstation documentation to help you decide which serial port and device driver to use. As a general rule, device drivers with names like `/dev/ttyd#`, where # is an integer, seem to work best. Once you have decided on a port, you may also need to kill its `getty` process, or set up the system so that it never starts `getty` for the port. The `getty` process for a port listens on the port expecting a connection from a terminal. Workstations are usually shipped with all of the ports enabled with `getty` processes, making it easy to add terminals. This interferes with SciAn's use of the port, so it must be stopped.

The default port device can be set using the `SCIAN_RECORDER_DEV` environment variable. See Chapter 10, "Customizing SciAn."

Video overview

The trickiest part of the whole operation is getting a clear video signal from the workstation to the recorder. To understand the problems and options involved, it is helpful to understand some basic concepts of video systems.

The video system used in North America and Japan is called NTSC and was invented many years ago. NTSC provides a series of images, each of which nominally contains 525 lines. Perhaps 450 of these lines are visible on the monitor, and most videotape recorders can only guarantee about 200 distinct lines. Each frame is made up of two separate fields. The fields are scanned interlaced, which means that the first field does every other line, and the second field fills in the missing lines. There are 30 frames, or 60 fields per second.

Note *Actually, there are 29.97 frames per second. Broadcast facilities have an elaborate mechanism to take care of the fractional part. One could conceivably use 29.97 as a frame rate in SciAn, but in general, it is best to ignore the difference.*

Europe uses the PAL video system, which has 625 lines per frame at 25 frames per second. PAL has some color improvements over NTSC (which has been nicknamed Never Twice the Same Color), but the basic ideas are similar. The following discussion will focus on NTSC.

NTSC video uses a composite video signal, which puts luminance, color, and frame and line synchronization information on the same cable. It is not meaningful to talk about the number of pixels per line, because the density of pixels varies depending on the color and luminance values used. The average number could be anywhere between 300 and 700 pixels per line.

It is also possible to have separate red, green, blue, and sync channels at the same frequency as the NTSC signal. The timing standard is called RS-170A, but many sources, including Silicon Graphics documentation, use the term NTSC incorrectly to describe the timing. This is of higher quality than composite video, because the red, green, and blue signals are kept separate rather than compressed into one signal. Sync may be transmitted on two separate cables (one for vertical and one for horizontal), on one cable, or combined with the green signal. A single sync cable is the most common. This is sometimes called RGBS, but RGB is more commonly used even if the sync signal is on a fourth cable.

Not all RGB signals are equal, of course. The timing and resolution are also important. If your video recorder has an RGB input and your workstation has an RGB output, don't automatically assume that you can plug them together and expect them to work.

Note *RGB is one kind of component video. Systems such as S-VHS provide another kind of component video which, like RGB, keeps color and luminance signals separate. Like RGB, these provide higher quality than composite video. This discussion will focus on RGB, because it is more common and more easily understood.*

One thing must be realized and accepted at the outset. No matter how much money you are willing to spend or how much work you are willing to do, there is just no way to get the total amount of information on a large high-resolution computer display onto video. The amount of information must be reduced drastically before it can be displayed, and there are good ways and bad ways to do the reduction. This hardware discussion describes some of the decisions that need to be made, and the section "Designing an animation" describes some more.

Connecting a video signal

The process of getting a video signal from the workstation to the video recorder has two steps:

- Getting a signal with a correctly sized image and RS-170A timing
- Converting the signal to composite NTSC video or S-VHS for recording and/or display

Choosing how to do the steps will require a little video engineering. Depending on how you decide to solve the problem, both steps may be done in one machine or separately. You will need to put together some combination of the following:

- Video recorders
  These may have RS-170A RGB, component video, or composite video inputs and outputs.
- NTSC encoders
  These have RS-170A RGB inputs and composite video outputs and serve only to make composite video from an RGB signal with the correct timing.
- Video boards
  These are located inside the workstation and may provide RGB, component video, or composite video outputs.
- Scan converters
  These take high resolution computer display RGB signals and convert them to RS-170A RGB, component video, or composite video.

All these kinds of equipment vary tremendously in quality.

As a rule of thumb, the best way of doing the conversion is with a good scan converter. A scan converter takes the whole screen and compresses it into a video frame. The best scan converters have clever filters which take into account the properties of NTSC video and blur the image in such a way that artifacts of interaction with NTSC are minimized. (Remember that, because the amount of information is being reduced, there is *no way* to avoid blurring, but there is good and bad blurring.)

If you do not have access to a scan converter, some Silicon Graphics workstations can produce RS-170A RGB or composite NTSC signals. This may require an add-on board. The reduction of information is typically done by using only the bottom left corner of the screen and throwing the rest away. (The Window menu has an item to place a window in this portion of the video screen.) Because no filtering is done on the signal, the techniques described below in "Designing an animation" are even more important. It is also possible to use the filters in the Renderer control panel, possibly combined with the Double Video menu items, to improve the quality even more.

A typical configuration

Figure 9-1 shows a typical configuration, one we have used to produce several movies.

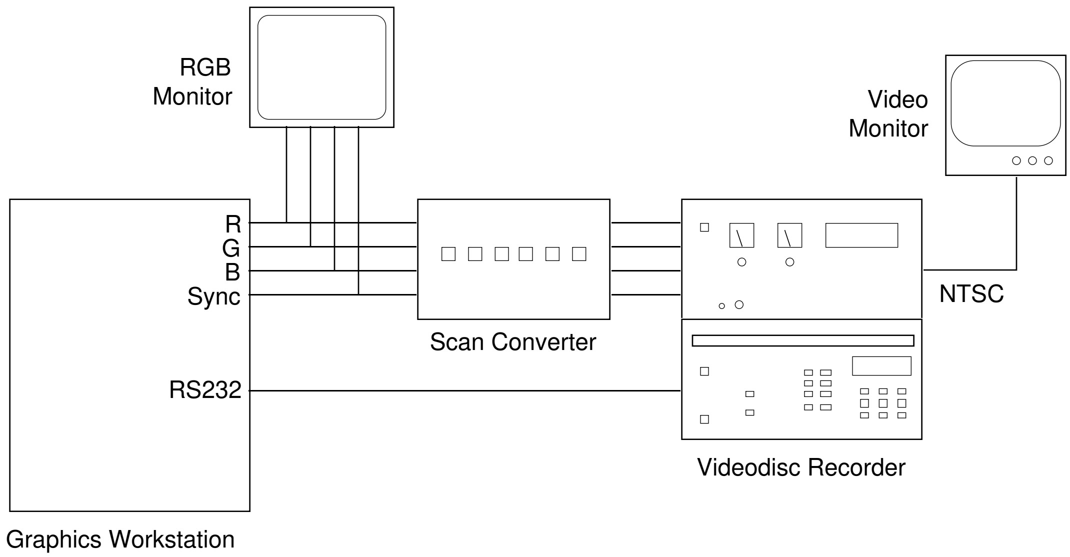

Figure 9-1. *Typical configuration*

On the left is the graphics workstation which is used to generate the images. The RGB and Sync signals go into a scan converter. A monitor is shown connected to the same lines to indicate that these are the same signals used to drive the monitor. In practice, connecting both may make it difficult to get a clean video signal, so it may be a better idea only to connect the monitor or scan converter at any one time. There are inexpensive switches which make it easy to switch back and forth between the two.

The scan converter converts the high resolution RGB and Sync signals to a smaller picture. The picture emerges via RGB and Sync signals at RS-170A timing that go directly into the videodisc recorder. The recorder pictured here is a Sony LVR-5000, which has RGB, NTSC, and composite video inputs. Recording from the NTSC output of the scan converter will also work, but the quality is not quite as good. Some videodisc recorders, however, do not have component video inputs, so for them it is necessary to use composite video.

The workstation is also connected directly to the videodisc recorder via an RS-232 serial control line. This allows SciAn to control the videodisc recorder to snap frames from the screen.

Figure 9-2 shows a typical configuration without a scan converter, which we used exclusively before acquiring a scan converter and which we still use for many animations.

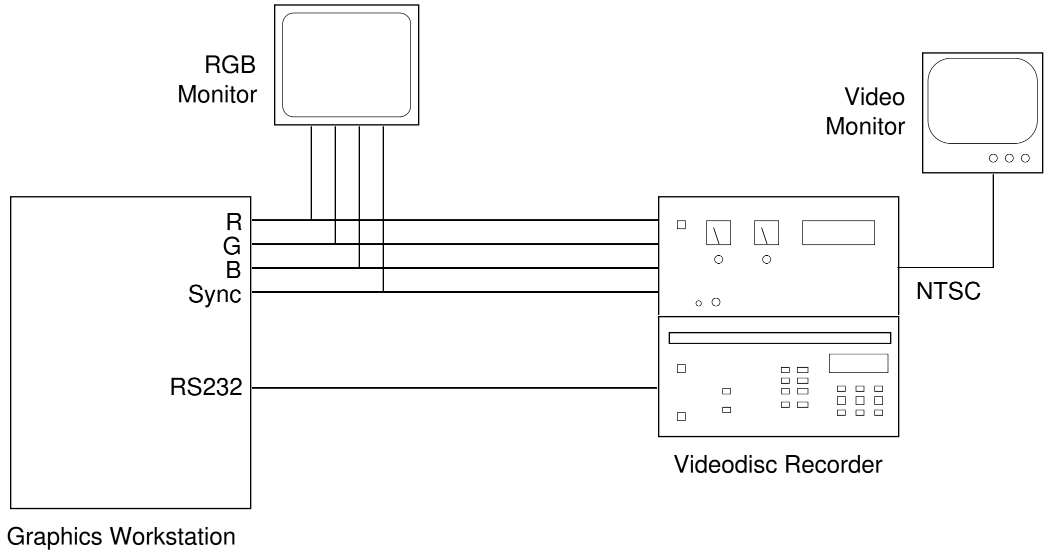

Figure 9-2. *Another typical configuration*

The only difference here is that the RGB and Sync lines go directly from the workstation to the videodisc recorder without going through a scan converter. In order for this to work, the workstation must produce RS-170A timings. The Silicon Graphics 240/GTX, for example, does this without any additional hardware by means of the `setntsc` command.

In this configuration, it is absolutely vital that the signals be set to RS-170A timings *before* they are connected to the videodisc recorder. The videodisc recorder synchronizes its motor to the signals, and improper signals can damage the motor and servo mechanism.

Recorder drivers

Every video recorder within SciAn is driven by a recorder driver. Recorder drivers are SciAn objects which contain information on how to connect to video recorders, start them up, and snap individual frames. To see all the recorder drivers within SciAn, choose Show Recorder Drivers from the Animation menu. You will see a window like Figure 9-3.

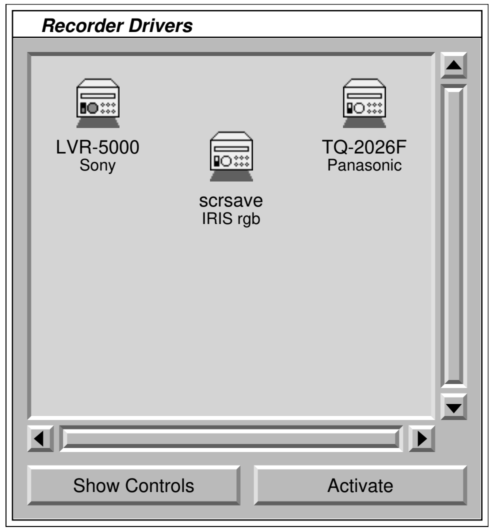

Figure 9-3. *Recorder drivers window*

All the recorder drivers available within SciAn appear as icons in the central icon corral. At any one time, exactly one of the recorders may be active. The active recorder is the one which will be used for all subsequent recording. Its icon is always marked by a bright red pilot light on the icon. To activate another recorder, select its icon and press the Activate button.

Some of the parameters of a recorder driver can be changed within its control panel. To see the control panel for a particular recorder, select its icon and press the Show Controls button. A control panel like Figure 9-4 will appear.

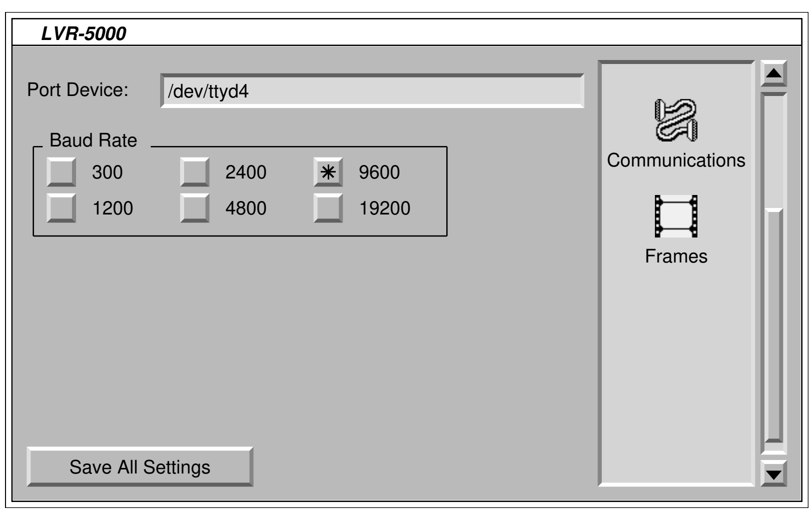

Figure 9-4. *Recorder Communications control panel*

The control panel is very much like the control panels of visualization objects. At the right is a series of icon buttons, each of which selects a group of attributes. To see a particular group of attributes, press its icon button.

In Figure 9-4, the communication controls are selected. Every recorder driver which can communicate with the workstation via a serial port has this set of controls. At the top is a text box which controls the serial device driver used to communicate with the recorder driver. Below it is a group of radio buttons which control the baud rate of the port. Both of these controls are explained in "Connecting a serial control signal" earlier in this chapter.

All video recorders also have a set of controls which control various frame parameters such as the frame rate and frame size. See Figure 9-5.

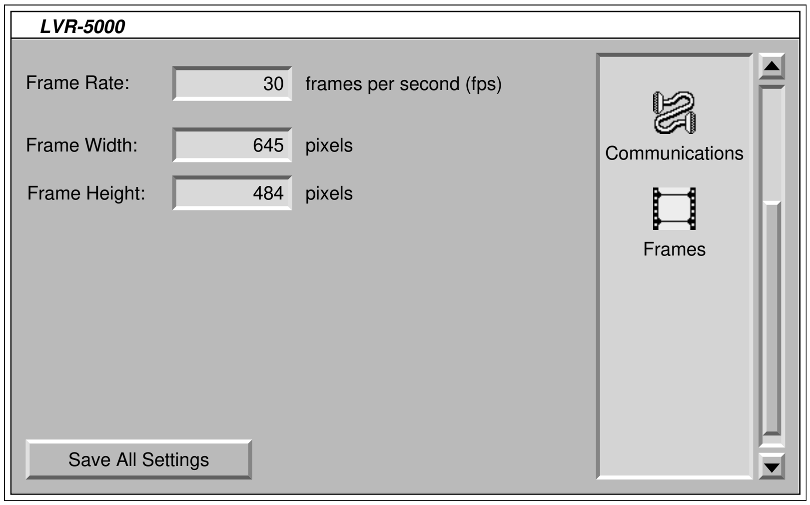

Figure 9-5. *Recorder Frames control panel*

At the top is a text box giving the frame rate in frames per second. You can set this to any real number greater than zero. Below are two text boxes giving the width and height of the video screen. These numbers define a region at the bottom left of the screen which is used for video recording. Some drivers, such as the image pseudorecorder driver, use these numbers to calculate the portion of the screen to save. Drivers which use the actual video signal just keep these numbers for reference. The width and height of the video screen of the active recorder driver defines the area for the Video Screen menu item.

At the bottom left of all recorder driver control panels is a Save All Settings button. This button becomes active as soon as you change any setting. Normally, settings that you change remain changed only for the rest of the session with SciAn. The next time you start up SciAn the settings will go back to the way they were. If you press this button, however, all of the settings for this recorder driver will be saved in a subdirectory called `.scianSettings` in your home directory. The next time you start up SciAn, all these settings will be read and used.

Note Save All Settings *only saves the settings within a particular recorder driver. Other information external to the recorder driver, such as information on which driver is currently active, is not saved.*

Each recorder driver is described in detail in the reference manual.

It is possible to write your own recorder drivers. For more information see the reference manual or the downloadable technical notes, or send mail to us at `scian-info@scri.fsu.edu`.

Designing an animation

Once your hardware and software is set up to be able to produce animations, you can start designing an animation of a particular scientific visualization.

At the present time, there are only two attributes of the visualization which are easy to animate: time and viewpoint. The easiest way to animate viewpoint is to do simple rotation around the focus point. It is possible to animate more complex parameters, but currently this takes quite a great deal of work. In future versions of SciAn, a keyframe editing mechanism will make this easy. For now, it is best to limit animations to time advance and rotation. Fortunately, a wide variety of scientific animations work well under these constraints.

The first thing to decide is exactly what you want to show in the animation. Is there a single visualization object or are there multiple ones? Are the datasets time-dependent? Do you want to fly around to see the visualization better, or is it satisfactory to keep it stationary? The best way to decide on the answers to these questions is to play around with the visualization interactively before you even think about starting on the script to produce the animation.

Whether you are going through a scan converter or not affects the size of the screen. With a scan converter, usually the full screen is used. Without a scan converter, the bottom left quarter of the screen is used. Use the space of the screen carefully. Remember that you will never get close to the complexity of what you can see on the computer screen. At best, you will be able to get one to four groups of visualizations and one to four big titles.

Remember to make everything big, especially the text and solid lines. Use text fonts large enough to read easily across the room. Never use single pixel lines; always make them at least two and possibly three pixels wide. Single pixel lines may be sharper than the video hardware can handle and may cause color bleeding.

Choose colors wisely. Some color combinations, such as dark blue on black or blue next to red or green, simply don't work with NTSC. In general, it is best to use muted colors such as blue or black for the background and bright colors such as yellow, green, or white for the foreground. Also beware of short color transitions, especially quick changes along a horizontal line. The NTSC video channels which carry color can change much more slowly than the channel that carries luminance. The best way to discover which color combinations work best together is to experiment.

Most video monitors overscan, which means that the edge of the image will be lost on all four sides. A good rule of thumb is to keep a border of 1 to 2 cm around the edge for the video screen and 2 to 4 cm around the edge for the full screen.

An animation is not limited to a single visualization window. You can have several windows side by side. The Tile items in the Arrange submenu of the Windows menu provide a quick way to arrange several similar visualization windows in a regular pattern on the screen. You will need to hide the window frames after doing this so that they will not appear in the final product. Another useful technique is to have a close-up view inset into a larger view. To do this, place the window with the close-up over the overview window, again hiding the window frame. Be careful, though. It is easy to clutter up the screen so much that it is hard to make sense of the animation.

Avoid going overboard with the length of the animation. The first impulse is to want to get as much as possible into the animation no matter how long it is. In practice though, animations should be one to three minutes in length. Much over three minutes and the audience will start to lose interest, no matter how good the science.

Another decision to make is how fine the animation should be. The best animation is animated "on ones," where every video frame is used. This results in a smooth animation at 30 frames per second. It is also possible to animate "on twos," where the videodisc is played back at half speed for 15 frames per second. Animation on twos is only slightly less smooth than animation on ones, and it takes only half the time to record. Animation on threes is noticeably choppy.

When calculating the time to record, remember that you will be recording every single frame. If each frame takes three seconds to draw and you are recording one minute at 30 frames per second, the entire animation will take 3*30*60=5400 seconds, or an hour and a half to record.

Producing an animation

Once you have decided on what to animate and how to animate it, you can go on to producing the actual animation. This is a process of progressive refinement, involving several steps, which may need to be repeated:

- Generate a visualization, logging the user interface functions to a script file.
- Try out the script file by running it through SciAn.
- Modify the script file with a text editor to include recording commands.
- Run the script on the screen as a dry run.
- Run the script with the recorder active to make the movie.

Preparing a script

A movie starts with a good script. As described in the previous chapter, SciAn can log actions within SciAn to a log file, which can be fed back into SciAn as a script. The first step in making a movie is to use the logging feature to produce a script that sets up the visualization the way it should look for the animation.

As described in the previous section, there are two things which can be animated to make a movie: time and viewpoint.

Time is animated using the clock control panel. Use the animation controls near the bottom of the window as described in Chapter 6 to set time to advance at a certain rate. When recording the video, SciAn will not advance the clock until the recording actually begins, and the clock will be advanced at a rate to ensure the correct rate at playback. For example, if the clock is set to advance 2 seconds for every 1 second in real time, and the video is set to 30 frames per second, then the clock will be advanced 2/30 or 1/15 second every frame.

Sequences with the time stopped and running at various rates can be mixed in a script. This is useful, for example, to have a second of still recording at the beginning and end of a sequence. To do this, go through all the sequences in the control panel of the clock. Stop the clock, then start it, then stop it again. Later, when you add the recording commands, you will add them between these events.

The easiest way to animate viewpoint is to do a smooth rotation around the object by turning on Rotation Inertia in the Preferences window and giving the space a spin. This will produce a `set rotation` command in the script, which is of the form

`set rotation` *speed* `[`*xAxis yAxis zAxis*`]`

where *speed* is the speed of rotation in degrees per second, and `[`*xAxis yAxis zAxis*`]` is the axis about which rotation is to occur. You can modify the speed in the script later by changing the number. Alternately, you can use the `smoothrots.c` program in the utilities directory to produce a sequence of rotations about a principal axis that smoothly accelerates from and decelerates to rest. There are other utility programs which allow the adventurous to do more complex viewpoint animation.

The flight simulator currently cannot be used to change viewpoints for an animation. However, it is possible to write sections of a script that do complex things with the viewpoint. See Chapter 5, "Scripts," in the reference manual.

As you are developing your script, remember that you can run SciAn reading a script and logging the results to another log file as you use the program. This is handy to add operations at the end of an existing script.

Modifying the script

Once a script has been made, it usually needs to be modified by hand to adjust various numbers, such as clock advance settings and rotation speed. After this has been done and has been verified to work, then the recording commands must be added to the script.

All recording is done within a recording block in a script. The format of a recording block follows:

```
begin recording nSeconds
   ... script commands
end recording
```

When SciAn gets ready to record, first it looks for a contiguous block lasting *nSeconds* seconds on the videodisc. Set *nSeconds* to a number that is at least as large as the total number of seconds in this scene of the recording. It may be larger than the number of seconds actually used, but it may not be larger than the time remaining on the disc.

Within a recording block, the `snap` and `record` commands record video. The `snap` command snaps a single frame. The `record` *nSeconds* command records a number of seconds.

Usually, there is only a single recording block in a script, although there could be more. The recording commands are most commonly at the end of the script, but the entire script could as easily be enclosed in a recording block, with short groups of `snap` and `record` commands distributed throughout.

It may also be handy to use the `setup block` to speed up operations. Simply use

```
begin setup
   ... script commands
end setup
```

to enclose all the script commands that need to be executed before recording. Within a setup block, changes to controls will only do a minimal number of window redraws. The windows will look ugly until the end of the setup block, but it will take much less time to run through that portion of the script. Do not use a setup block within a recording block or vice versa. See Chapter 10 for more information.

As the last step before doing the dry run, you may want to put a `quit` command at the end of the script. This will cause SciAn to quit at the end of the script rather than simply go into interactive mode.

Doing a dry run

Once the script has been set up, do a dry run without driving the video recorder to see if there are any errors in the script. SciAn is set up to do dry runs by default to minimize the possibility of error. When you want to make the actual recording you will have to do something extra as described in the next section.

When you run your script through SciAn (`scian -s` *name*), SciAn will go through all the motions of producing a movie without actually setting up the recorder. Instead, you will see on the screen the images as they will appear on the video and will see messages on the console whenever frames are to be snapped.

Be absolutely sure that your script is doing the right thing during the dry run before proceeding to the next step.

Making the movie

When you are satisfied that the script is set up the way you want to record, you are ready to make the movie.

First set up a console or terminal window with a font large enough so that you will easily be able to see it through the video monitor.

If you are recording using the NTSC mode of the Silicon Graphics IRIS, enter the `setntsc` command. If you are recording through a scan converter, you will not need to do this.

SGI *The* `setntsc` *and* `set60` *commands should be available on your system somewhere, possibly in* `/usr/sbin`. *If you cannot find them, you can press the* F10 *key while in SciAn to toggle in and out of NTSC mode. You can also compile the programs in the utilities directory distributed with SciAn. These programs only work on some Silicon Graphics workstations.*

Next, connect the recorder to the workstation. On the Sony LVR-5000 it is important only to do this after the video signal has been set to the correct scanning mode. If it is hooked up to a signal it cannot synchronize to, the motor will hunt for a speed and may damage the servo mechanism. Make sure any RS-232 cables are connected as well and the switches on the recorder are set to enable remote control.

Now rerun SciAn. Include the `-v` flag on the command line to enable video recording. Instead of a dry run, SciAn will go ahead and record the video. If you included the `quit` command as suggested in the previous section, SciAn will then quit.

When SciAn is done recording the video, disconnect the recorder and then set the workstation back to 60 Hz mode with the `set60` command.