

Sun Microsystems, Inc.



SunVision
Sun's Visualization Software Platform
Technical White Paper

June 1990

NOTE: Please send comments or questions about this document to:

Visualization Products Marketing
Sun Microsystems, Inc.
P.O. Box 13447
Research Triangle Park, N.C. 27709-3447

CONTENTS

Chapter 1. Introduction to SunVision

1.1 Overview	1-1
1.2 SunVision Libraries – Application Programming Interfaces.....	1-1
1.3 SunVision Window-Based Tools.....	1-1

Chapter 2. Image Processing

2.1 SunIPLib	2-1
2.1.1 SunIPLib Programming	2-1
2.1.2 List of Functions	2-1
2.2 SunIP	2-3
2.2.1 SunIP Features	2-4

Chapter 3. Volume Rendering

3.1 Background	3-1
3.2 Viewing Modes.....	3-1
3.2.1 Cube Mode.....	3-1
3.2.2 Lightbox Mode.....	3-3
3.2.3 Point Cloud Mode.....	3-4

Chapter 4. High-Quality Rendering

4.1 Using SunART.....	4-1
4.2 SunART Scene Files.....	4-1

Chapter 5. 3-D Interactive Graphics

5.1 File Manipulation.....	5-1
5.2 SunGV Scene Editing	5-1
5.3 SunGV Viewing Parameters and Light Sources.....	5-2

Chapter 6. SunMovie

6.1 SunMovie Features	6-1
-----------------------------	-----

Chapter 7. Tools and Utilities

7.1 The Colormap Editor	7-1
7.2 The File Browser.....	7-2
7.3 The Color Box Editor.....	7-2
7.4 Miscellaneous Widgets	7-3

Chapter 8. The Visualization File Format

8.1 Image File Format.....	8-1
8.2 Volume File Format.....	8-2
8.3 Point Cloud File Format.....	8-2
8.4 Geometry File Format.....	8-3
8.5 Movie File Format	8-4

Figures

Figure 1-1.	SunVision Window-Based Tools Communication	1-2
Figure 1-2.	PMGRLib Functions	1-3
Figure 2-1.	SunIPLib Functions.....	2-1
Figure 3-1.	Volume Data	3-1
Figure 3-2.	Texture Mapping in Cube Mode of SunVoxel.....	3-2
Figure 3-3.	Ray Casting in Cube Mode of SunVoxel.....	3-2
Figure 3-4.	SunVoxel Lightbox Mode.....	3-3
Figure 3-5.	SunVoxel Cloud Mode.....	3-4
Figure 4-1.	SunART Commands	4-2
Figure 5-1.	SunGV Object Attributes	5-2
Figure 5-2.	SunGV Viewing Parameters	5-2
Figure 5-3.	SunGV Light Sources.....	5-3
Figure 6-1.	SunMovie Control Panel.....	6-1
Figure 7-1.	SunVision Colormap Editor.....	7-1
Figure 7-2.	SunVision File Browser	7-2
Figure 7-3.	SunVision Color Box Editor	7-2
Figure 7-4.	Single-Value Slider	7-3
Figure 7-5.	Double-Value Slider.....	7-3
Figure 7-6.	Thumbwheel.....	7-3
Figure 7-7.	Trackball.....	7-4

Chapter 1. Introduction to SunVision

1.1 Overview

SunVision™ software is a set of highly integrated visualization libraries and tools with components for image processing, volume rendering and analysis, photorealistic rendering, and interactive display and manipulation of 3-D geometric data. These components facilitate sharing of images and data among the various display technologies.

SunVision 1.0 consists of:

- SunIPLib — Image-processing library
- SunART™ — Photorealistic rendering program and library interface
- PMGRLib — Library of functions for communication with the parameter manager
- SunIP — Image-processing tool
- SunVoxel — Volume-rendering tool
- SunGV — Interactive graphics tool
- SunMovie — Movie display tool
- SunVIF — Reconfigurable user interface tool
- SunPMGR — Parameter manager program

1.2 SunVision Libraries – Application Programming Interfaces

SunVision 1.0 provides two new libraries for visualization tasks, image processing, and high-quality rendering, as well as a new utility library. SunVision is an application developer's platform. Future releases will include an additional library interface for volume rendering. In addition, SunVision works with Sun's XGL™ 3-D interactive graphics library, providing an integrated graphics and imaging developer's platform.

SunIPLib provides extensive image-processing functionality. It is described in more detail in Chapter 2.

SunART (Advanced Rendering Tool) is software for high-quality rendering. SunART provides two RenderMan® compatible interfaces: a library interface and an interface through RIB™ protocols. SunART can also be accessed through its own script-based interface. SunART is described in detail in Chapter 4.

PMGRLib is a utility library that provides functions for communication between individual visualization programs and SunVision parameter manager program (SunPMGR) and, in turn, communication between SunPMGR and SunVision user interface program (SunVIF). This software architecture is described in the following section.

All SunVision libraries have online manual pages.

1.3 SunVision Window-Based Tools

SunVision provides user access to its component visualization techniques through OpenWindows™ based tools for image processing, volume rendering, interactive graphics, and high-quality rendering. These are augmented by a tool for viewing movie loops and by support utilities such as an interactive colormap editor. SunVision tools are useful as application prototypes for software developers and can be used “as is” by sophisticated end-users. The source code for the image-processing tool is provided as an example for using SunIPLib, SunVIF, and SunPMGR.

Each window-based tool is a completely independent visualization program. All share a common user interface program and parameter manager program. (See Figure 1-1 below.) The parameter manager program is a simple database for storing variables and their current values. When users modify a parameter through the user interface, the PMGRLib functions communicate this to the SunPMGR program. This, in turn, can be relayed to any of several visualization programs using the same library. The PMGRLib functions are based on lightweight message-passing routines.

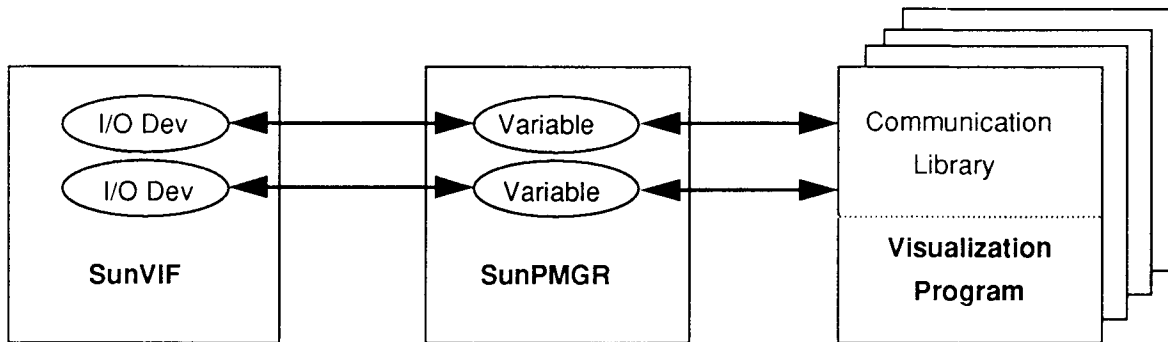


Figure 1-1. SunVision Window-Based Tools Communication

SunVIF user interfaces are also fully configurable at runtime, without any programming. For example, users can cut and paste text and widgets and reassign their associated parameters while the program is running, using simple mouse and keyboard commands. The individualized user interface can then be saved or discarded.

This architecture of several independent — yet integrated — programs running simultaneously offers a number of clear advantages to developers and users:

- SunVision is extensible. Additional programs can be integrated by adding appropriate parameters to SunPMGR and by interactively creating a user interface
- Visualization programs within SunVision are easily extensible. If a new function is added to a visualization program, users need only add the appropriate variables to the parameter database (an ASCII file) and interactively add the appropriate widgets to the user interface
- Programs in SunVision can easily be specialized for particular applications
- User interfaces can easily be prototyped without any modification to the visualization program
- Changes made to the user interface do not require changing—or even relinking—the visualization programs. This means that only the user interface needs to be retested, as the visualization programs remain untouched

Figure 1-2 lists PMGRLib functions.

Function	Description
<i>pmgr_get_handle</i>	gets handle to named parameter
<i>pmgr_get_info</i>	gets database entry information
<i>pmgr_get_value</i>	gets the value of an entry
<i>pmgr_init</i>	initializes connection to a SunVision database program
<i>pmgr_lock</i>	locks access to a specified parameter
<i>pmgr_main_loop</i>	dispatches all notifications as they are received
<i>pmgr_notify_add</i>	adds the calling process to the notification list for the indicated parameter
<i>pmgr_notify_dispatch</i>	allows a database change notification to be dispatched
<i>pmgr_notify_dispatch_all</i>	same as <i>pmgr_notify_dispatch</i> except it dispatches all outstanding messages
<i>pmgr_notify_display_wait</i>	allows a database change notification to be dispatched, but doesn't return until a message has been dispatched
<i>pmgr_notify_get</i>	gets a pointer to the event handler
<i>pmgr_ready</i>	puts a marker in the parameter database to indicate that the specified program has completed initialization
<i>pmgr_save_script</i>	creates file containing current value of listed database entries suitable as input to <i>pmgrscript</i>
<i>pmgr_set_value</i>	sets the value of a parameter
<i>pmgr_set_value_notify</i>	sets value of parameter and calls the notify procedure as if it had been sent from the database
<i>pmgr_unlock</i>	unlocks a database entry that was locked with <i>pmgr_lock()</i>
<i>pmgr_wait_for</i>	returns when the specified program has finished initializing

Figure 1-2. PMGRLib Functions

Chapter 2. Image Processing

2.1 SunIPLib

SunIPLib is Sun's platform library for image-processing applications. It includes C functions for:

- Image analysis
- Arithmetic and logical operations
- Spatial filtering
- Fourier domain processing
- Geometric operations

It also includes library functions to create and manipulate subimages and regions of interest.

SunIPLib functions work on three image data types: unsigned byte, signed 16-bit short, and 32-bit floating point. Images may have multiple bands.

2.1.1 SunIPLib Programming

SunIPLib functions treat images as objects with an underlying data structure. This structure contains information about the image's size, data type, number of bands, location in memory, and associated parent/child images. Each library routine examines this structure to determine what type of processing is appropriate. For example, when *ip_convolve()* is invoked on a multibanded floating-point image, it performs a separate floating-point convolution on each band of the image. From the programmer's point of view, only one convolution routine is called, regardless of the image data type, number of bands, and so on.

2.1.2 List of Functions

Figure 2-1 shows the SunIPLib functions in SunVision version 1.0.

Function	Description
Spatial Filtering	
<i>ip_convolve</i>	convolve image with kernel
<i>ip_edge</i>	enhance edges of image
<i>ip_median</i>	median image filter
Transform	
<i>ip_fft</i>	forward Fourier transform, real image
<i>ip_fft_complex</i>	forward Fourier transform, complex image
<i>ip_fft_convert</i>	unfold conjugate symmetric FFT representation
<i>ip_fft_display</i>	convert output of <i>ip_fft</i> into integer displayable form
<i>ip_iffi</i>	inverse Fourier transform
<i>ip_iffi_complex</i>	inverse Fourier transform, complex image
Geometric	
<i>ip_reflect</i>	reflect or transpose image
<i>ip_rotate</i>	rotate image
<i>ip_translate</i>	translate image
<i>ip_warp</i>	polynomial image warp
<i>ip_zoom</i>	zoom image

Figure 2-1. SunIPLib Functions

Function	Description
Statistics and Analysis	
<i>ip_extrema</i>	find image mean and variance
<i>ip_histogram</i>	compute image histogram
<i>ip_lookup</i>	pass image through lookup table
<i>ip_moments</i>	find image mean and variance
<i>ip_morph</i>	morphological operation
<i>ip_threshold</i>	threshold image against float value
Arithmetic and Logical	
<i>ip_abs</i>	generate absolute values of an image's pixels
<i>ip_add</i>	add two images
<i>ip_add_const</i>	add constant to image
<i>ip_and</i>	bitwise logical AND
<i>ip_and_const</i>	bitwise logical AND with constant
<i>ip_blend</i>	alpha or compositing blend of two images
<i>ip_divide</i>	divide two images
<i>ip_fcos</i>	generate image containing the cosines of a source image's pixels
<i>ip_fexp</i>	generate image containing e* of a source image's pixels
<i>ip_flog</i>	generate image containing the natural logarithms of a source image's pixels
<i>ip_frecip</i>	generate image containing the reciprocals of a source image's pixels
<i>ip_fsin</i>	generate image containing the sines of a source image's pixels
<i>ip_fsqrt</i>	generate image containing the square roots of a source image's pixels
<i>ip_lincomb</i>	interband linear combination
<i>ip_max</i>	pixel-wise max of two images
<i>ip_max_const</i>	pixel-wise max of image and float constant
<i>ip_merge</i>	merge two images
<i>ip_min</i>	pixel-wise min of two images
<i>ip_min_const</i>	pixel-wise min of image and float constant
<i>ip_mul_const</i>	multiply image by constant
<i>ip_multiply</i>	multiply two images
<i>ip_nand</i>	bitwise logical NAND
<i>ip_nor</i>	bitwise logical NOR
<i>ip_not</i>	bitwise logical NOT
<i>ip_or</i>	bitwise logical OR
<i>ip_or_const</i>	bitwise logical OR with constant
<i>ip_rescale</i>	rescale image
<i>ip_shift</i>	shift image bits
<i>ip_subtract</i>	subtract one image from another
<i>ip_xor</i>	bitwise logical exclusive OR
<i>ip_xor_const</i>	bitwise logical XOR with constant
Memory/Object Management	
<i>ip_create</i>	allocate child image
<i>ip_destroy</i>	deallocate image
<i>ip_destroy_kernel</i>	deallocate a kernel
<i>ip_destroy_mtable</i>	deallocate a morphology table
<i>ip_destroy_roi</i>	deallocate a region of interest

Figure 2-1. SunIPLib Functions (continued)

Function	Description
Utility	
<i>ip_base_image</i>	return the base parent of an image
<i>ip_copy</i>	copy image, with data type conversion
<i>ip_create</i>	allocate a root image
<i>ip_display</i>	display image
<i>ip_gather</i>	reassemble bands in multiband image
<i>ip_getband</i>	gets single band from multiband image
<i>ip_getchild</i>	get child image relative to parent
<i>ip_getpixel</i>	get information about specific pixel
<i>ip_image_check</i>	compare number of bands, size, and pixel type of two images
<i>ip_image_gen</i>	generate test image
<i>ip_init</i>	initialize image-processing library
<i>ip_load_file</i>	load image from file
<i>ip_load_kernel</i>	load convolution kernel from file
<i>ip_load_mtable</i>	load morphology table from file
<i>ip_put_pixel</i>	put per-pixel information into image
<i>ip_read</i>	read data from image
<i>ip_roi_from_image</i>	get region of interest from image
<i>ip_save_iff</i>	write image to a file
<i>ip_save_kernel</i>	write convolution kernel to a file
<i>ip_undisplay</i>	disassociate image from display
<i>ip_write</i>	write data to image

Figure 2-1. SunIPLib Functions (*continued*)

2.2 SunIP

SunIP is an OpenWindows tool that lets users interactively process images, extract and process subimages and regions of interest, display separate bands of multiband images (such as true color), and analyze images. SunIP enables users to display and concurrently manipulate images in multiple windows.

SunIP is part of the SunVision family of visualization tools, all of which use the SunVIF user interface program. The SunIP interface can be tailored with the SunVIF interface management system. The source code for this tool is included.

The image-processing operations that are available interactively through SunIP are based on the SunIPLib image-processing library.

2.2.1 SunIP Features

- Permits up to six image windows for source and destination of operations and for display
- Loads images in visualization file format (*.vff*), *TIFF*, *FITS*, and Sun raster formats; saves images to *.vff* files
- Displays 8-bit images, single 8-bit bands of multiband images, or 24-bit images dithered for 8-bit display
- Provides interactive display of pixel coordinates and values
- Offers interactive access to functions
 - Analytical (statistical and morphological operations)
 - Arithmetic and logical
 - Spatial filtering and edge detection
 - Fourier domain processing (FFT, transcendentals)
 - Geometric operations (scale, rotate, warp, transpose)
 - Miscellaneous and utility operations
- Imports images from other SunVision components (e.g., slices from volumes, high-quality renderings)
- Provides image editing
- ROI
- Undo

Chapter 3. Volume Rendering

The SunVision volume-rendering component, SunVoxel, is an interactive tool for generating images from *volume data*. Volume data can be defined as a three-dimensional grid of data points. Volume data may come from many different application areas, including X-ray crystallography, fluid-flow analysis, seismic interpretation, and a variety of medical imaging technologies such as computed tomography (CT) and positron emission tomography (PET).

SunVoxel consists of window-based rendering and analysis functions, as well as data conversion filters. It lets users manipulate and view volume data in two modes: extract and view on-axis or oblique 2-D slices of volume data; and manipulate the entire (possibly clipped) volume and view internal structures as 3-D objects. SunVoxel supports unsigned byte data on uniform rectangular grids. Data is stored in visualization file format (*.vff*).

3.1 Background

A voxel is a volume element; a u, v, w location within a volume and an associated value. It can be considered as the three-dimensional equivalent of a pixel. The image on the right in Figure 3-1 is a close-up of voxels within a volume data set. Currently in SunVoxel, voxels are scalar values, usually representing density information, but theoretically they could also be vectors of data associated with each spatial location.

One convenient way to think of a volume of data is as a sequence of 2-D arrays of data, or a sequence of slices of data. This paradigm works particularly well for medical images collected as a sequence. Slices are “stacked” into a volume, and the data elements become voxels. The picture on the left in Figure 3-1 shows a set of slices that make up a volume data set. SunVision includes utilities that create volume *.vff* files from ordered slices of data.

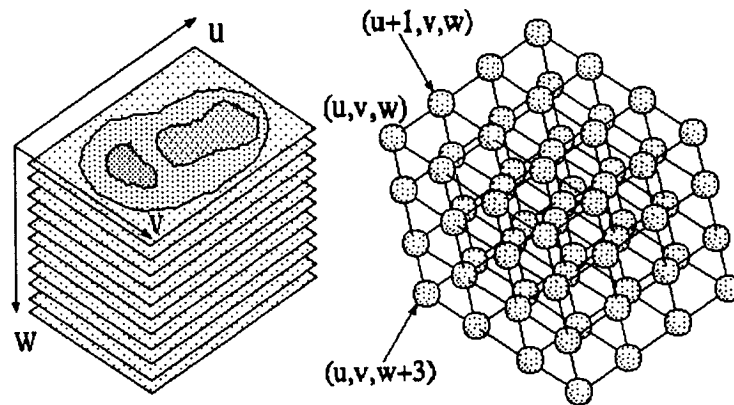


Figure 3-1. Volume Data

3.2 Viewing Modes

3.2.1 Cube Mode

Cube mode offers several methods of viewing the volume as a three-dimensional cube:

Texture mapping maps volume data values onto the surface of the volume. This is the simplest of the direct rendering methods. In this mode, the data appears as a rectangular solid that users can cut away or slice on the six faces of the volume and on one oblique axis. The data values on the exposed sliced faces, as well as the values

of the unsliced faces, are texture-mapped onto the resultant solid. This method, shown in Figure 3-2, is also referred to as multiplanar reprojection.

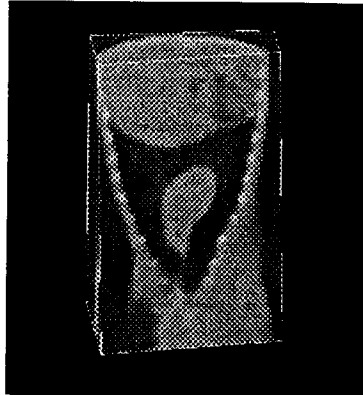


Figure 3-2. Texture Mapping in Cube Mode of SunVoxel

Another direct rendering method is *ray casting*, which displays the volume as a solid composed of multiple structures or substances. *Maximum value ray casting* gives the X-ray effect shown in the image on the left in Figure 3-3. SunVoxel spawns rays along the view vector into the volume and displays the maximum voxel value encountered along each ray.

Ray casting is also used to produce images that show the interior structures of the volume as semi-transparent surfaces. The volume data is sorted or classified into the substances (e.g., skin, muscle, bone) that it represents. Users assign each substance a color and opacity value. Multiple internal structures can be seen by varying the opacities, producing an image as on the right in Figure 3-3. Another viewing mode enables users to interactively vary color and opacity values of the volume's substances and to see the changes very rapidly. This mode requires a preprocessing step that restricts users to one viewing position.

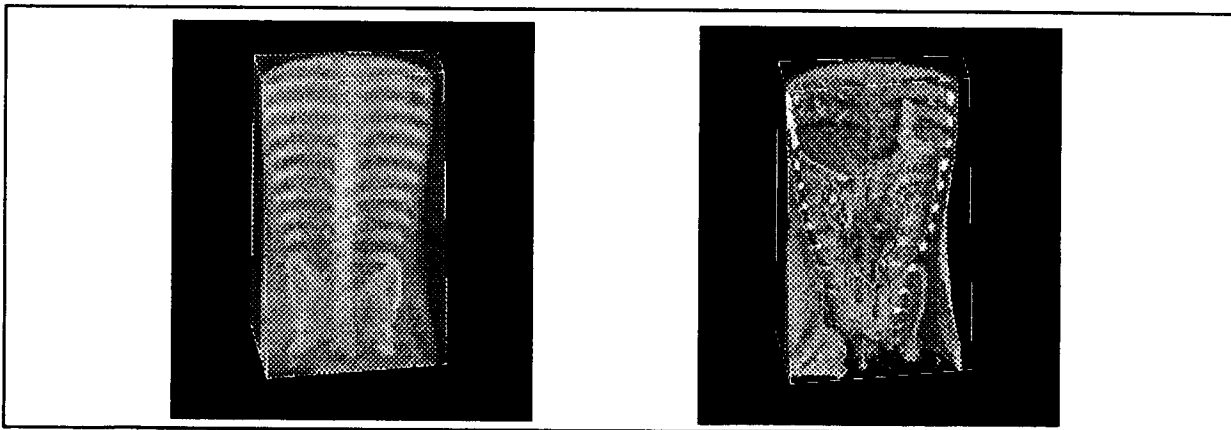


Figure 3-3. Ray Casting in Cube Mode of SunVoxel

Cube Mode Features

- Wireframe display for interactive manipulation and viewing of the volume
- Texture-mapped viewing of sliced and clipped volume solid
- Maximum value ray casting
- Ray casting with transparency and light-source shading to view selected interior surfaces of volume data with previously assigned substance classifications and properties (classifications and properties can be changed interactively)
- Interactive modification of substance colors and opacities for rapidly viewing selected interior surfaces
 - enables skipping layers of a substance to see internal layers
 - includes function to preprocess data for this display mode
 - loads, modifies, and saves files of substance classification data
- Nearest neighbor and trilinear interpolation sampling
- Load and save files of rendering and viewing parameters
 - to recreate images
 - for movie generation
- Arbitrary image size; limited only by CPU memory and swap space

3.2.2 Lightbox Mode

Lightbox mode uses the same format volume data as cube mode but treats the volume as a series of two-dimensional slices. It extracts sequential planar slices of the raw data to let users pan through the entire volume, one slice at a time, along any of the three major axes. Lightbox mode also supports the extraction of an oblique slice. The picture below shows three orthogonal slices, with the oblique slice (lower right) defined by the cut line appearing in the other views.

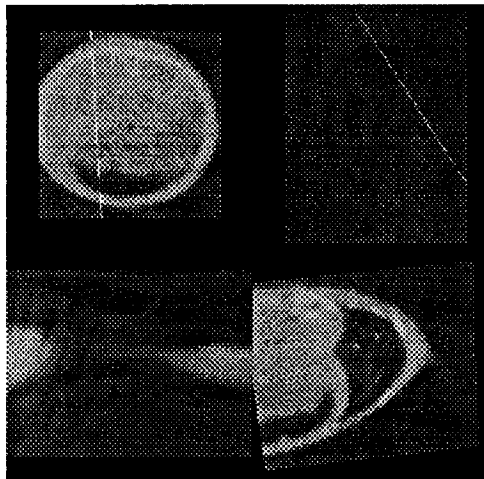


Figure 3-4. SunVoxel Lightbox Mode

Lightbox Mode Functions

- Display orthogonal and arbitrary oblique angle slices through volume
- Pan through slices of volume along major axes
- Select by number slice to be displayed
- Read and modify voxel values in model space
- Nearest neighbor and/or trilinear interpolation sampling
- Scale displayed slice with arbitrary coefficients
- Edit volume data by clipping a slice or range of slices to a polygon (region-of-interest/extrusion clipping)
- Load and save edited volumes

3.2.3 Point Cloud Mode

SunVoxel's Point Cloud mode displays data stored in point cloud format. Very dense clouds of point data are often used to display surface data extracted from CAT scans, points lying on a terrain, solvent accessible surfaces of a molecule, and mathematically defined surfaces. Point Cloud mode is also useful for displaying surfaces that cannot be easily defined mathematically, but can be generated as a set of points in 3-D space. Each data point consists of a 3-D location, normal, gradient information, and optional color. Points are a geometric primitive and can be transformed and shaded using traditional graphics techniques.

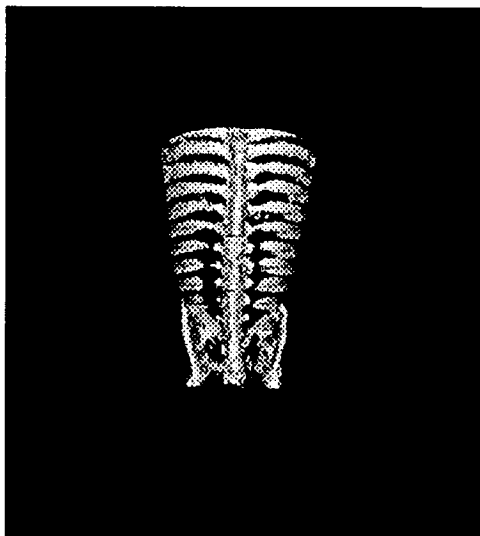


Figure 3-5. SunVoxel Cloud Mode

Chapter 4. High-Quality Rendering

The photorealistic rendering tool provided with SunVision is Sun's Advanced Rendering Tool, SunART. It is a script-driven, photorealistic rendering package that can produce high-quality images of three-dimensional geometric data. SunART includes a rendering program and associated utility programs for preparation of data from various modelers. Features include:

- Very high quality lighting and shading capabilities, including the ability to accommodate user-defined shading routines and user-defined attributes
- Texture mapping, bump mapping, and reflection (environment) mapping
- Anti-aliasing by stochastic supersampling

SunART also has a RenderMan compatible interface. Both the RenderMan subroutine library and RenderMan Interface Bytestream (RIB) interfaces are supported.

4.1 Using SunART

To use SunART, users must supply certain information, which may be provided in several forms:

Scene File Scene files contain SunART commands that control the creation of an image. It is also possible to give SunART commands from standard input. This method can be useful for rendering multiple-frame sequences

Data Input Files Input geometry and images must be provided in the visualization file format (*.vff*). SunART also provides utilities to convert AutoCAD™ *.dxf* and *movie.byu* format files into *.vff* format

The SunART Tool Menu options let users select a scene filename and determine how SunART runs. For example, users can set the size of the output image and the anti-aliasing quality using a successive refinement method. It is also possible to run SunART from the UNIX command line

The RenderMan Subroutine Library Applications can be built by linking with the RenderMan compatible subroutine library

The RIB Protocol The RIB protocol interpreter takes a stream of data from the standard input, interprets the stream, and issues commands to SunART to produce the output image

4.2 SunART Scene Files

A SunART scene file is a text file consisting of SunART commands that load data, control the transformation and viewing of data, define the lighting model, specify various attributes of the data, and update the rendered image. Scene files are ASCII files, with each line containing one SunART command followed by its arguments. Figure 4-1 lists the SunART commands.

Data Input/Output

LOAD_OBJECT *filename*
 LOAD_BKG *filename*
 LOAD_TEXTURE *filename*
 LOAD_REFLMAP *filename*
 LOAD_BUMPMAP *filename*
 SAVE_IMAGE *filename*
 SAVE_ZBUF
 CLR_BKG *r g b*
 CLR_ALPHA *alpha*
 CLR_ZBUF
 RENDER
 PATCH_SUBDIVISIONS *ps*

Viewing and Projection Transformation

EYEPOINT *x y z*
 GAZEPOINT *x y z*
 ROLLANGLE *theta*
 NEAR_CLIP *nc*
 FAR_CLIP *fc*
 PERSPECTIVE
 FIELD_OF_VIEW *theta*
 ORTHOGRAPHIC
 LR_CLIP *lc rc*
 TB_CLIP *tc bc*

Shading Commands

SHADER *shader_name surface_parameters*
 SHADE_INTERPOLATE *flag*
 ATMOSPHERE_SHADER *shader_name atmosphere_parameters*

Commands Used from Standard Input

ANTIALIAS *a*
 EXIT
 FILTER
 SAMPLECOUNT *s*

Model Transformation

MODEL_TRANSLATE *tx ty tz*
 MODEL_SCALE *sx sy sz*
 MODEL_ROTATE *ax ay az theta*
 PUSH_MODEL_MATRIX
 POP_MODEL_MATRIX
 RESET_MODEL_MATRIX
 SET_MODEL_MATRIX
 CONCAT_MODEL_MATRIX

Lighting Model

NUM_LIGHT_SOURCES *nl*
 LTSRC_TYPE *n type*
 LTSRC_WEIGHT *n weight*
 LTSRC_COLOR *n r g b*
 LTSRC_DIR *n x y z*
 LTSRC_POS *n x y z*
 LTSRC_ATTEN *n base scale exponent*
 LTSRC_SHAPE *n concentration cone_angle cone_delta_angle*

Object Attributes

OBJECT_COLOR *r g b*
 TRANSLUCENCY *r g b*
 OPACITY *r g b*
 AMBIENT_COEFF *ka*
 DIFFUSE_COEFF *kd*
 SPECULAR_COEFF *ks*
 SPECULAR_EXP *n*
 TEXTURE *flag*
 REFLMAP *flag*
 REFLMAP_COEFF *krc*
 REFLECT_CUBE_SIZE *size*
 BUMPMAP *flag*
 BUMPMAP_COEFF *kbm*
 MATTE_SURFACE *flag*
 BACKFACE_CULL *flag*

Figure 4-1. SunART Commands

Chapter 5. 3-D Interactive Graphics

SunVision interactive Geometry Viewer, SunGV, is used to interactively view 3-D graphics. It can also be used to interactively generate scene files for SunART, SunVision photorealistic renderer.

In SunVision 1.0, SunGV provides wireframe display of line, polygon, and patch data types stored in *.vff* format files. Shaded displays will be added in SunVision 1.1.

5.1 File Manipulation

These functions deal with transferring geometry, image, and scene files to and from SunGV. Users can:

- Load individual objects (*.vff* files), which can then be individually transformed and assigned surface properties
- Load and save scene files containing objects and information for high-quality rendering, including model and viewing transformations, surface properties, and lighting information
- Export scenes to the SunVision clipboard, import them into SunART, and perform high-quality rendering according to the scene information

5.2 SunGV Scene Editing

SunGV includes functions to enable users to easily build up complex scenes from simple objects. Scenes are composed of a series of objects that are loaded into SunGV, positioned, and assigned attributes. Objects can be organized hierarchically in a tree structure. Users can “step through” the tree structure to transform individual nodes on the tree.

The edit functions let users select, copy, paste, cut, and delete objects in the object hierarchy. Tree-structured object hierarchies can be created and manipulated.

Each object in a scene can undergo individual object transformations such as scale, rotate, and translate. In addition, users can assign “object attributes” to each object as listed in Figure 5-1.

SunGV: Object Attributes			
Color	R	G	B
	1.00	1.00	1.00
Opacity	1.00	1.00	1.00
Ambient coeff.	01.00		
Diffuse coeff.	00.70		
Specular coeff.	00.70		
Specular exp.	30.00		
Bumpmap coeff.	01.00		
Refl map coeff.	01.00		
Refl cube size	01.00		
<input type="checkbox"/> Texture	[]		
<input type="checkbox"/> Bump	[]		
<input type="checkbox"/> Reflect	[]		
Patch subdivision	08		
Patch deviation	000.00		

Figure 5-1. SunGV Object Attributes

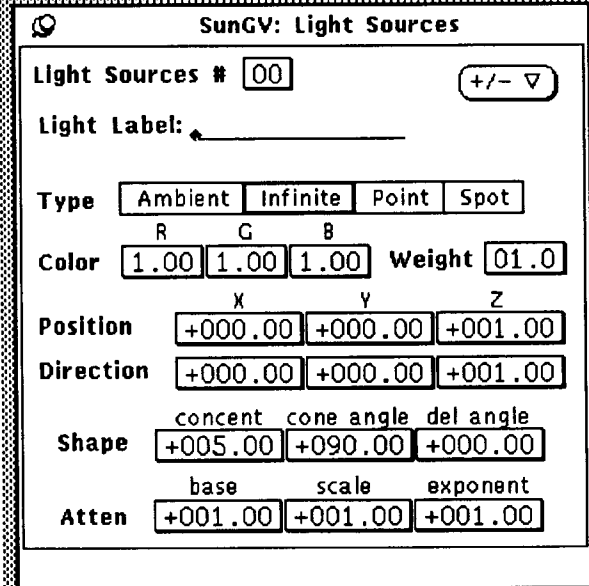
5.3 SunGV Viewing Parameters and Light Sources

The viewing functions can be used to change viewing and projection parameters, as well as the lighting model. The general viewing parameters are those shown in Figure 5-2. These parameters correspond to the viewing parameters used by SunART. Notice that a background image can be specified in the scene description.

SunGV: Viewing Parameters			
Eye	X	Y	Z
	+000.00	+000.00	+005.00
Gaze	+000.00	+000.00	+000.00
Near	000.50	Far	100.00
Left	000.00	Right	000.00
Top	000.00	Bottom	000.00
Rollangle	+000.0	FOV	050.00
Proj	<input type="checkbox"/> Perspective <input type="checkbox"/> Orthographic		
Bkg Color	0.00	0.00	0.00
Bkg Image	[]		

Figure 5-2. SunGV Viewing Parameters

In addition, users can specify up to 32 light sources. Each light source can be assigned the properties shown in Figure 5-3.



The image shows a dialog box titled "SunGV: Light Sources". It contains several fields for configuring a light source:

- Light Sources #**: A numeric field with the value "00" and a +/- button.
- Light Label**: A text input field.
- Type**: A group box containing four radio buttons: "Ambient", "Infinite", "Point", and "Spot".
- Color**: Three numeric fields for Red (R), Green (G), and Blue (B) components, each with the value "1.00". A **Weight** field with the value "01.0".
- Position**: Three numeric fields for X, Y, and Z coordinates, with values "+000.00", "+000.00", and "+001.00" respectively.
- Direction**: Three numeric fields for direction components, with values "+000.00", "+000.00", and "+001.00" respectively.
- Shape**: Three numeric fields for "concent", "cone angle", and "del angle", with values "+005.00", "+090.00", and "+000.00" respectively.
- Atten**: Three numeric fields for "base", "scale", and "exponent", each with the value "+001.00".

Figure 5-3. SunGV Light Sources

Chapter 6. SunMovie

The SunVision image/movie display component, SunMovie, is a tool for displaying image and movie data. Images and sequences of images generated by other components of SunVision can be viewed using this tool. It supports the display of 8-bit grayscale or dithered full-color movies. In Movie mode, users can run movie sequences; in Image mode, users can display single images.

6.1 SunMovie Features

Image or image sequences appear in the SunMovie display window. This window contains the following control buttons:

- **File** — Controls image data file operations (load, save, import, export)
- **View** — Selects a viewing mode: Movie or Image
- **Props** — Sets rendering and control properties (colormap or movie control windows)
- **Stop** — Quits SunMovie

The Movie control panel, shown below, contains the cine loop controls used while displaying movies.

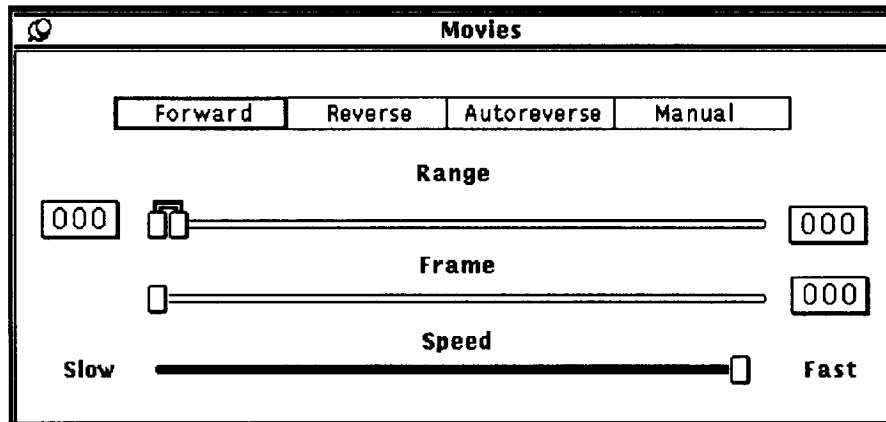


Figure 6-1. SunMovie Control Panel

- **Forward** — Run movie from low to high frame (default)
- **Reverse** — Run movie in reverse sequence
- **Autoreverse** — Run movie frames from low to high, then reverse
- **Manual** — Manually select the frame number to display by changing the Frame slider or thumbwheel
- **Range** — Double-ended slider, and corresponding thumbwheels, setting low and high frame numbers. Note that it is not possible to “wrap around” frame 0
- **Frame** — Current frame (applies to Manual mode only)
- **Speed** — Update rate

Movie data consists of single-band, 8-bit images. SunMovie requires *.vff* format data files described in Chapter 8.

Chapter 7. Tools and Utilities

SunVision includes window-based tools for controlling the colormap, browsing/loading/saving files, and selecting substance colors (in SunVoxel). The implementation of all the window-based tools includes three new widgets: double-value sliders, thumbwheels, and trackballs.

7.1 The Colormap Editor

The components of SunVision share a common colormap. Of the possible 256 display values available with an 8-bit frame buffer, SunVision reserves 40 locations for use by the window system and the color box editor. SunVision uses the remaining 216 colormap entries to display images in this environment. All images are rendered into a virtual 32-bit frame buffer if true color is needed (for transparency), or into a virtual 8-bit frame buffer if the image is suitable for grayscale or pseudocolor representation. 32-bit images are dithered before displaying. 8-bit images are linearly mapped from 256 values into the 216 range before displaying.

The colormap editor (Ctool) lets users modify the colormap (except for the 40 reserved locations) by loading preselected colormaps or by interactively drawing a colormap. (See Figure 7-1.)

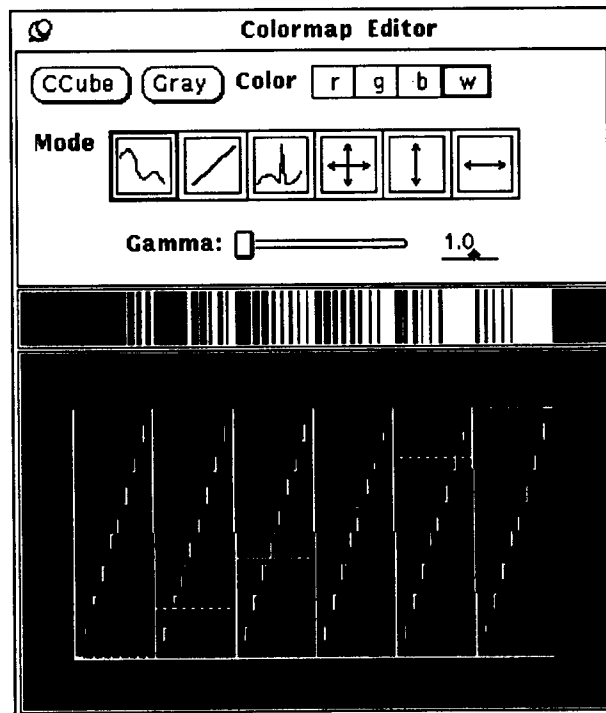


Figure 7-1. SunVision Colormap Editor

The colormap editor provides the following functions:

- **CCube** — Loads a dithered color cube colormap for displaying dithered true-color images
- **Gray** — Loads a linear ramp (grayscale). This colormap provides a full 0-to-256 ramp in the 216-value colormap space
- **Color** — Selects the color component to be modified: r = red; g = green; b = blue; w = red/green/blue
- **Mode** — Drawing mode for colormap modification (similar to a draw program). From left to right, the modes are: *freehand*, *rubber-banding*, *spikes*, *shift up/down and left/right*, *constrained shift (up/down only)*, *constrained shift (left/right only)*

7.2 The File Browser

The File Browser lets users browse among directory listings and load a file. Users can either type in a filename or click on a name in the directory listing. If a directory is selected, the File Browser switches to that directory and lists its files and subdirectories. The File Browser recognizes environment variables. (See Figure 7-2.)

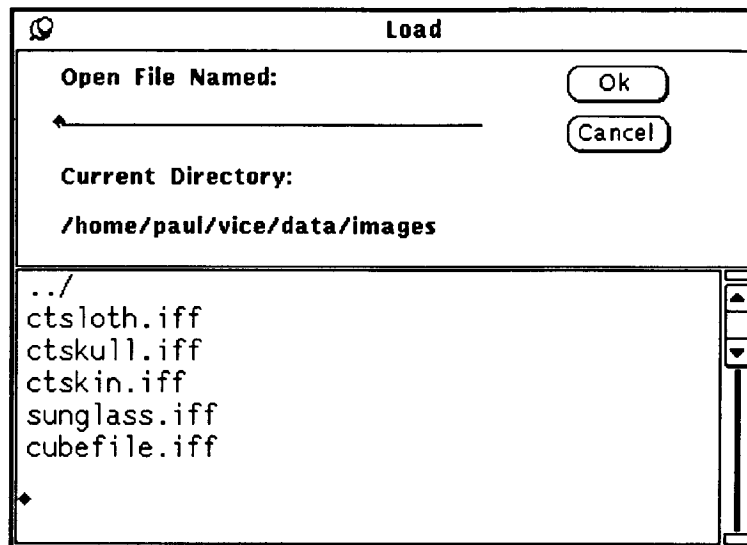


Figure 7-2. SunVision File Browser

7.3 The Color Box Editor

Color boxes are used only by SunVoxel to edit and display the current color assigned to a volume substance. The color box editor lets users interactively modify the color selected for a substance. (See Figure 7-3.)

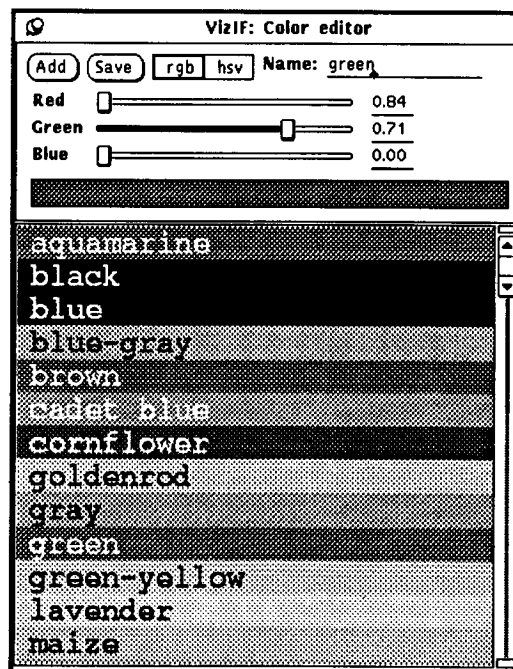


Figure 7-3. SunVision Color Box Editor

Users can select from a palette of previously selected colors in the bottom of the panel, or can create a new color using the sliders to alter color components in either RGB or HSV mode. Users may also add a color to the list of previously defined entries. Modifying a color writes it into the frame buffer's colormap, using one of eight locations reserved for these colors.

7.4 Miscellaneous Widgets

Sliders let users set a value within a specified range. This is a single-value slider. (See Figure 7-4.)

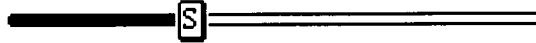


Figure 7-4. Single-Value Slider

Figure 7-5 shows a double-value slider, which lets users set minimum and maximum values. This slider is used by SunVoxel to set the front and back clipping planes. Users may also set both values simultaneously, maintaining the distance between them by selecting the bar over the slider (the drag bar) rather than the drag box, and dragging it to the desired setting.

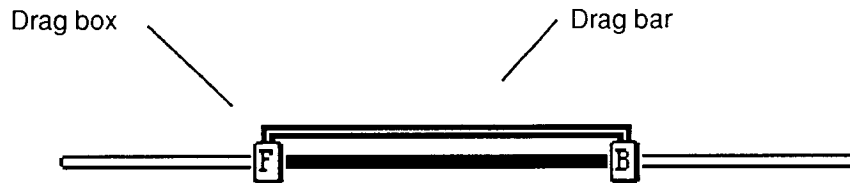


Figure 7-5. Double-Value Slider

Thumbwheels display values and let users set new values by changing any of the digits in the number displayed. In Figure 7-6, to change the value to 103.60, users would position the cursor over the "2" and press the left mouse button. Clicking the middle mouse button on a digit decrements the digit. Users may also position the cursor over a digit and drag the cursor down to increase the value, up to decrease the value — as if rotating a thumbwheel.

Thumbwheels are often tied to sliders, so that moving a slider changes the value in a corresponding thumbwheel, and vice versa. (See Figure 7-6.)

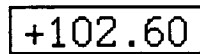


Figure 7-6. Thumbwheel

Trackballs control the 3-D orientation of an object or light source. (See Figure 7-7.) The trackball controls the orientation of a volume in SunVoxel. To rotate the volume, users simply put the cursor inside the trackball window and hold down the mouse Select button while moving the cursor.

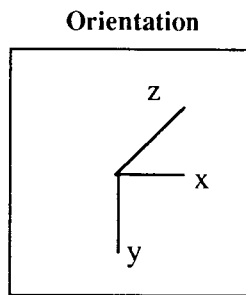


Figure 7-7. Trackball

Chapter 8. The Visualization File Format

All components of SunVision use *.vff* format. This is an extensible format for images, geometry, and volume data sets. All *.vff* files are composed of an ASCII header followed by binary or ASCII data, with a Ctrl-L character separating the two.

The header begins with a “magic number,” the string *ncaa*, identifying the file as *.vff*. The magic number is followed by a series of keyword/value pairs of the form

```
keyword=value [value...];
```

Keywords are defined by the programs that read the data files. The following sections describe particular visualization files used by SunVision programs.

8.1 Image File Format

Image file headers must contain the following keyword/value pairs (*title* is optional):

```
ncaa  
type=raster;  
rank=2;  
size=xsize ysize;  
rawsize=number_of_bytes;  
bands=number_of_bands;  
bits=number_of_bits_per_band;  
format=base;  
title=string;  
^L
```

<i>ncaa</i>	is the <i>.vff</i> magic number
<i>type</i>	refers to the kind of data in the file; it should be <i>raster</i> for image files
<i>rank</i>	is the dimensionality of the data; it should be 2 for image files
<i>size</i>	<i>xsize</i> and <i>ysize</i> are non-negative integers representing the size of the image
<i>rawsize</i>	is the number of data bytes in the file (excludes the header)
<i>bands</i>	is the number of bands in the image (1 for mon, 3 for RGB, etc.)
<i>bits</i>	is the number of bits per band; SunVision images must be 8, 16, or 32
<i>format</i>	is the format of the image. Images must be in <i>base</i> format
<i>title</i>	is an ASCII image title
^L	separates the ASCII header from the binary data

Binary data follows the header. Multiband images are stored in pixel order (e.g., a typical 32-bit image is stored in base format as *ABGRABGR*).

8.2 Volume File Format

Volume data file headers may contain the following keyword/value pairs:

```
ncaa
rank=3;
type=raster;
format=slice;
size=usize vsize wsize;
origin=x0 y0 z0;
extent=x1 y1 z1;
aspect=xsize ysize zsize;
rawsize=number_of_bytes;
bands=1;
bits=8;
title=string;
^L
```

<i>ncaa</i>	is the <i>.vff</i> magic number
<i>rank</i>	is the dimensionality of the data. It must be 3 for volumes
<i>type</i>	refers to the kind of data in the file. It must be <i>raster</i> for volumes
<i>format</i>	is the format of the image. SunVoxel currently requires <i>slice</i> format
<i>size</i>	<i>xsize</i> and <i>ysize</i> are the size of a single slice; <i>wsize</i> is the number of slices
<i>origin</i>	is the floating-point model space coordinate of the upper left corner of the volume
<i>extent</i>	is the floating-point model space coordinate of the lower right corner of the volume
<i>aspect</i>	specifies scale factors applied to the volume before viewing
<i>rawsize</i>	is the number of data bytes in the file (excludes the header)
<i>bands</i>	refers to the number of bands in the data; currently, bands must be 1
<i>bits</i>	is the number of bits per band in the data; currently, this must be 8
<i>title</i>	is an ASCII title for the volume
<i>^L</i>	separates the ASCII header from the binary data

Origin, *extent*, *rawsize*, *aspect*, and *title* are optional keyword/value pairs for volumes.

Binary data follows the header. The data is *column*, *row*, *slice* order packed bytes, and must be aligned on a double-word boundary.

8.3 Point Cloud File Format

Point cloud data is stored in groups called buckets. Each bucket can be thought of as a cube of the volume, with the bucket *x*, *y*, *z* coordinate representing a “base point” for the lower left corner of the cube. Buckets can have a size of 4, 8, or 16. Following the bucket information is a list of offsets from the base point.

Point cloud files require the following header information:

```
ncaa
format=cloud;
type=bucket;
display=type_of_file;
bucket_count=number_of_buckets;
point_count=number_of_points;
bucket_size=size_of_bucket;
bounding_cube=greatest_value;
rawsize=number_of_bytes;
comment=string;
^L
```

<i>ncaa</i>	is the .vff magic number
<i>format</i>	is the format of the image. SunVision point clouds must use <i>cloud</i> format
<i>type</i>	refers to the kind of data in the file. Point cloud data must be of type <i>bucket</i>
<i>display</i>	can be pseudocolor (<i>SHADEPSEUDO</i>) or grayscale (<i>SHADEGRAY</i>)
<i>bucket_count</i>	specifies the number of buckets in this file
<i>point_count</i>	specifies the number of points in this file
<i>bucket_size</i>	refers to the <i>x</i> , <i>y</i> , <i>z</i> size of each bucket. It must be 4, 8, or 16
<i>bounding_cube</i>	is the absolute value of the largest <i>x</i> , <i>y</i> , or <i>z</i> in the data file
<i>rawsize</i>	is the number of data bytes in the file (excludes the header)
<i>comment</i>	is an optional ASCII comment field
<i>^L</i>	separates the ASCII header from the binary data

Point cloud data follows the header; the data must be aligned on a double-word boundary.

8.4 Geometry File Format

Polyhedra and bicubic patches are currently supported. Polyhedra are described as follows:

```
ncaa
type=vertices;
components=x y z normal_x normal_y normal_z;
{
    {x0, y0, z0, nx0, ny0, nz0},
    {x1, y1, z1, nx1, ny1, nz1},
    ...
    {xn, yn, zn, nxn, nyn, nzn},
}
type=connectivity;
components=variable.i;
{
    {vertex list for polygon 0},
    {vertex list for polygon 1},
    ...
    {vertex list for polygon N},
}
```

Bicubic patch descriptions require the following:

```
ncaa
type=vertices;
components=x y z;
{
    {x0, y0, z0},
    {x1, y1, z1},
    ...
    {xn, yn, zn},
}
name=patch0;
type=nurb patch;
patchtype=nurb;
vertices=ctlpts;
num_ctlu=number of control points in u;
num_ctlv=number of control points in v;
uorder=order_in_u;
vorder=order_in_v;
klenu=length_of_knot_vector_in_u;
klenv=length_of_knot_vector_in_v;
uknot=u knot vector list;
vknot=v knot vector list;
```

Both the polyhedra and patch formats are expandable to include user-defined attributes at polygon vertices or patch control points. For details, see the *SunVision Reference Manual*.

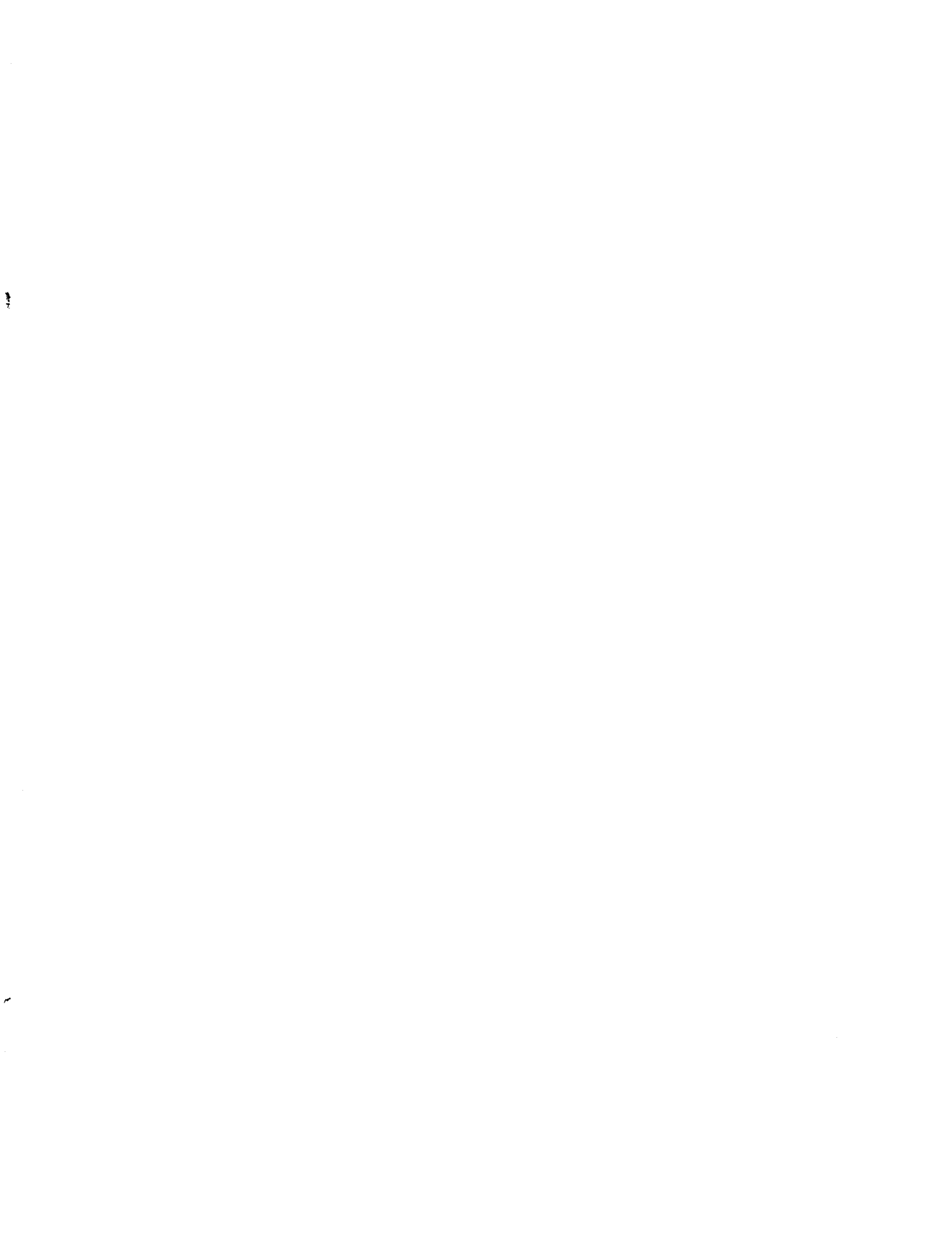
8.5 Movie File Format

Movie file headers contain the following information:

```
ncaa
rank=3;
type=raster;
format=slice;
size=xsize ysize zsize;
rawsize=number_of_bytes;
bands=1;
bits=8;
title=string;
^L
```


<i>ncaa</i>	is the <i>.vff</i> magic number
<i>rank</i>	specifies the dimensionality of the data; for movies, rank must be 3
<i>type</i>	is the type of data in the file; movies are type <i>raster</i>
<i>format</i>	refers to the format of the data; movies are in <i>slice</i> format
<i>size</i>	<i>xsize</i> and <i>ysize</i> are the size of each frame, and <i>zsize</i> is the number of frames
<i>rawsize</i>	is the number of bytes of data in the file (excludes the header)
<i>bands</i>	is the number of bands per frame; currently, this must be 1
<i>bits</i>	is the number of bits per band; currently, this must be 8
<i>title</i>	specifies an optional ASCII movie title
<i>^L</i>	separates the ASCII header from binary data

Binary movie data follows the header. The raw data is in *frame, row, column* order and must be aligned on a double-word boundary.





Sun Microsystems, Inc.
2550 Garcia Avenue, Mountain View, CA 94043
415 960-1300 FAX 415 969-9131

For U.S. Sales Office
locations, call:
800 821-4643
In CA: 800 821-4642

European Headquarters
Sun Microsystems Europe, Inc.
Bagshot Manor, Green Lane
Bagshot, Surrey GU19 5NL
England
0276 51440
TLX 859017

Australia: (02) 413 2666
Belgium: 32-2-759 5925
Canada: 416 477-6745
France: (1) 40 94 80 00
Germany: (089) 95094-0
Hong Kong: 852 5-8651688
Italy: (39) 6056337
Japan: (03) 221-7021

Korea: 2-563-8700
New Zealand: (04) 499 2344
Nordic Countries: +46 (0)8 7647810
PRC: 1-8315568
Singapore: 224 3388
Spain: (1) 5551648
Switzerland: (1) 8289555
The Netherlands: 033 501234

Taiwan: 2-7213257
UK: 0276 62111
Europe, Middle East, and Africa,
call European Headquarters:
0276 62111
Elsewhere in the world,
call Corporate Headquarters:
415 960-1300
Intercontinental Sales

Specifications are subject to change without notice.

Sun Microsystems and the Sun logo are registered trademarks of Sun Microsystems, Inc. SunVision, SunART, XGL, and OpenWindows are trademarks of Sun Microsystems, Inc. All other products or services mentioned in this document are identified by the trademarks or service marks of their respective companies or organizations. This product is protected by one or more of the following U.S. patents: 4,777,485; 4,668,190; 4,527,232; 4,745,407; 4,679,041; 4,435,792; 4,719,569; 4,550,368 in addition to foreign patents and applications pending.

© 1990 Sun Microsystems, Inc.

Printed in USA 6/90 FE275-0/10K