sgi™

OpenGL Performer™

Real-Time 3D Rendering for High-Performance
and Interactive Graphics Applications

OpenGL Performer is a high-performance 3D rendering toolkit for developers of real-time, multiprocessed, interactive graphics applications. OpenGL Performer dramatically simplifies development of complex applications such as visual simulation, simulation-based design, virtual reality, interactive entertainment, broadcast video, CAD, and architectural walk-through while providing a high-performance portability path across the entire IRIX® and Linux® product lines from SGI.

An OpenGL Performer application can make automatic and optimal use of available system features and components on any SGI™ platform—including peak performance rendering and use of multiple CPUs. On IRIX platforms, OpenGL Performer also makes optimal use of multiple graphics pipelines and real-time scheduling features.

Feature Highlights

OpenGL Performer provides all of the basic features and tools to build a complete image-generation solution, including:

- Maximum graphics performance across the entire IRIX and Linux product lines from SGI and support for OpenGL® rendering
- Transparent multiprocessing and efficient use of multiple CPUs, video channels, and graphics pipelines, without requiring application code or compilation changes
- Support for Digital Video Multiplexer Option (DPLEX) on InfiniteReality™ series configurations for dome distortion correction and other high frame rate applications
- Real-time fixed frame rate operation for truly immersive simulations
- Automatic scene and load management features; view frustum culling, level of detail (LOD) evaluation, and dynamic video resolution (DVR)
- Run-time and real-time profiling for use in debugging, tuning, and load management
- Asynchronous paging of database geometry and imagery, including fast-loading formats
- Automatic paging and management of huge textures, up to 8 million x 8 million texels in a single texture
- Active surface definition for automatic paging of terrain with continuous LOD evaluation
- Asynchronous database intersection and user database processing
- Dynamic animated geometry and general data
- Light points and calligraphic points for visual simulation
- Atmospheric effects for visual simulation, including fog and haze, and support for implementing range/angle-correct layered fog and patchy fog
- Visual simulation effects, including rotorwash
- Double precision coordinate system support for very large databases
- Video textures
- Database interoperability with run-time linking with database loaders for loading data files of any file format or multiple file formats
- EventView performance analysis tool
- Enhanced realism through a multipass rendering infrastructure utilizing output from OpenGL Shader™

Sample applications and viewers are provided with source code. More than a gigabyte of utility libraries, file loaders, examples, and data, much of it contributed by third-party suppliers, is also included with the product. Extensive programming guides and online documentation tools are also included.

Entering Data into OpenGL Performer: Database Independence Ensures Interoperability

OpenGL Performer is database-independent, supporting asynchronous loading and paging of arbitrary file formats. A run-time look-up mechanism is used for finding loaders for arbitrary file formats that, upon loading, can be combined into a single database. It is shipped with more than 50 different file loaders and most of the respective sources. Included in the collection are native csb OpenGL Optimizer™ and iv Open Inventor™ loaders for easy import of files.

Additionally, OpenGL Performer provides a fast-paging file format specifically designed to speed up the loading and storing of scenes (usually 100 times faster than a standard loader) for use in run-time database paging.

Application Layer Architecture

The OpenGL Performer libraries extend the functionality of the SGI system-level software. OpenGL Performer supports OpenGL/GLX™ operation through a unified application program interface (API). On IRIX, optimum performance is achieved through REACT™ real-time extensions to the IRIX operating system. REACT is used both for real-time scheduling control and real-time system profiling. XFS™ filesystem features are used for large database paging on the IRIX operating system. OpenGL Performer leverages the system capabilities to provide high-level application control while allowing for simple and efficient user access to these complex features. Applications can take advantage of the API completely or in part while accessing any system libraries such as OpenGL, OpenGL/GLX, IRIX, and Linux as needs dictate.

Library Architecture: Rapid Rendering and Peak Performance

OpenGL Performer was designed for seamless integration with all SGI graphics visualization systems, ensuring peak rendering performance across the product line. The product is a layered library consisting of low-level primitives, higher level multiprocessed scene management, and application-level utilities provided in source code. The bottom level is the rapid-rendering layer, libpr, to which applications have complete access and which employs more than 4,500 specialized rendering loops to optimize graphics features such as lighting, multitexturing, and transparency. Libpr also performs back-end state management. The top level is libpf, which sets up a multiprocessed environment for full application processing—providing the high-level functionality critical to establishing real-time frame rates.

The large majority of applications work at the libpf level. There are several libraries in the application utility layer for optimizing databases (libpfdu), loading different file formats (libpfdb), handling device input and implementing user interfaces (libpfui) and GUIs (libpfutil), and rendering special effects (libpfutil). Finally, source code to a full-sample application, the perfly demo program, is provided for use in the development of custom applications.

Rendering Architecture: App, Cull, and Draw

OpenGL Performer provides a multiprocessed rendering architecture. At its core lie multithreaded, parallel rendering pipelines for per-frame scene management and image generation for output to graphics pipelines. The software rendering pipelines are each split into three major pieces to handle critical path operations:

- App: specification of viewpoint and object positions
- Cull: LOD management, view frustum culling, and sorting (for state optimization and rendering special effects)
- Draw: the final rendering of the scene

Each software pipeline uses one graphics pipeline and can render an arbitrary number of output channels.

The user has full control over the configuration of the App, Cull, and Draw tasks, including:

- Putting the tasks in a single process
- Dividing the tasks arbitrarily between multiple processes
- Running the tasks synchronously or asynchronously

OpenGL Performer can also automatically make process configuration decisions at run time based on the run-time hardware platform.

Additional asynchronous processes are available for user customization and for certain predefined tasks such as database paging, intersection testing, dynamic geometry evaluation, and computation of complex light-point characteristics.

Running Real Time: IRIX REACT

In situations where a guaranteed fixed frame rate is required, OpenGL Performer uses the REACT extensions to the IRIX operating system to control process scheduling and process priority management. REACT guarantees real-time predictable behavior from the IRIX operating system by locking the API's processes to specific processors and maintaining nondegrading priority for them.

Performance Monitoring for Load Management: EventView, LOD, and DVR

OpenGL Performer provides a full suite of diagnostic statistics, including graphics pipeline hardware statistics for extremely accurate measurements of rendering time. These statistics are used for tuning and real-time monitoring of full system performance for load management and for direct use with other system monitoring tools, such as IRIXview™.

OpenGL Performer includes a time event analysis tool called EventView. It allows tracing time events generated by Libpf internals as well as user-generated time events on a logic-analyzer-style display. It is useful for measuring how the duration of various Performer execution blocks varies across time and as a function of user application events.

There are two automatic mechanisms for load management. The first mechanism, LOD management, adjusts object complexity in accordance with scene quality and performance considerations set by the database or user. This mechanism allows objects with low contribution to scene quality (far from the eyepoint, small in scene, or based on custom parameters) to be rendered at a lower level of complexity—thus reducing polygonal and graphics state loads. The second mechanism is targeted at pixel-fill or raster-load management. On the InfiniteReality series of graphics subsystems, raster load is managed with DVR, which allows each display channel to be automatically rendered with fewer pixels (determined on a per-frame basis based on per-channel load). The resulting image can be zoomed up to full output resolution, using bilinear interpolation without added latency or loss of performance.

Texturing with Large Imagery: Virtual Clip Texturing

Traditionally, large geographic areas were textured with separate tiled textures. This required significant modeling effort, complex application management of the texture paging, and a substantial amount of texture memory.

Supported by OpenGL Performer on the InfiniteReality series of graphics subsystems, cliptexturing (clipmapping) is a superior alternative because it virtualizes the texture and allows the entire texture to be specified in a single coordinate system. Only a small fixed amount of these virtualized textures, called clipmaps, need to be kept in hardware texture memory. The InfiniteReality series of graphics subsystems features specialized hardware that can map texture coordinates from the original virtual space into this "clipped" texture space. This allows texture and geometry of large textures to be defined more independently than is possible with texture paging. With clipmapping, large-area geospecific imagery, such as satellite and aerial photographs, can be easily mapped onto terrain geometry with minimal database creation effort. This clipped part of the texture is actually a subset of the clipmapping pyramid usually associated with MIPmapping and is centered at a point of interest in the virtual texture.

The size of the clipped area needs to be only as big as the number of high-resolution texels that can fit on the screen at one time and is completely decoupled from the size of the virtual texture. For a virtual texture of size 8 million x 8 million texels, less than .0000003 percent (under 45MB) of the actual potential full virtual texture is kept in hardware texture memory. The user can choose a smaller clipped space and thus use even less hardware texture memory. The virtual texture space can also be very sparsely populated with high-resolution insets. Lower-resolution versions of the image data will automatically be used where high-resolution data is unavailable.

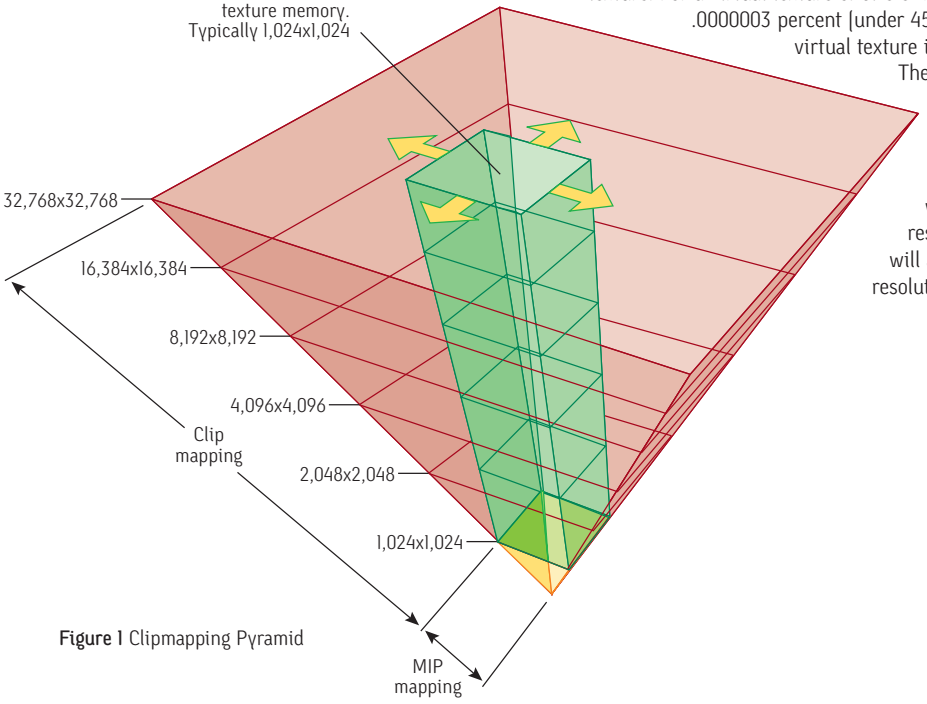


Figure 1 Clipmapping Pyramid

OpenGL Performer manages the virtualization of clip-mapped textures, the update of the center of interest based on viewer position, and the automatic paging of texture data to keep the clipped space up-to-date. A two-level look-ahead caching scheme is employed in order to minimize disk paging latency and improve download bandwidth into texture memory—thereby wasting a minimal amount of both host memory resources. Load management controls are provided to control the texture and paging resources. OpenGL Performer also contains support for the efficient management of multiple graphics pipelines in a system viewing a single cliptexture and management for multiple cliptextures. Utilities are provided to convert image data to clip-mapped texture files for optimal texture paging speed.

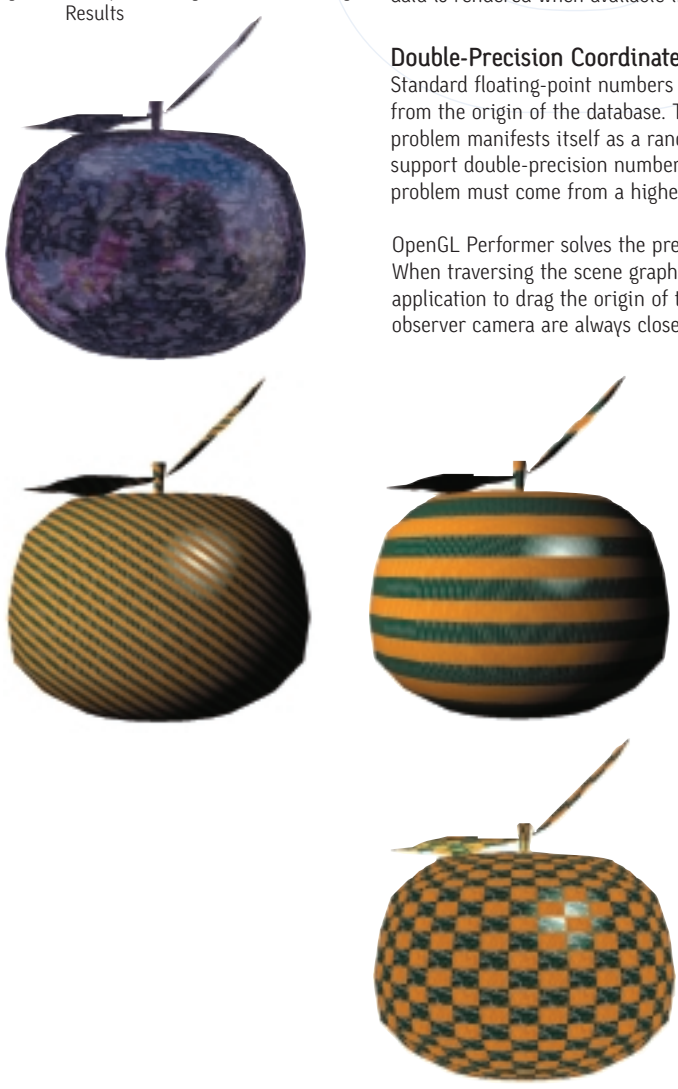
Rendering Large Geometric Surfaces: Active Surface Definition

The rendering of very large or heavily tessellated surfaces presents many image-quality and load-management challenges. OpenGL Performer solves these problems using an approach called Active Surface Definition (ASD). ASD provides an efficient, multiprocessed framework for the evaluation and paging of geometry over precomputed levels of detail based on user-specified evaluation, quality, and load-management constraints. Transitions between different levels of detail are made smoothly, on a per-triangle basis, with no visible spatial or temporal artifacts.

Dynamic Data Buffers: Engine and Flux

OpenGL Performer includes several features for the representation and evaluation of dynamic data. Engines allow the description of operations, such as morphing, blending, and bounding box computation, to be performed on specific objects or buffers of data. Fluxes are dynamic evaluated objects, the contents of which can be computed by engines and used as geometry or transformations any place where fluxed data is allowed. Asynchronously generated data is rendered when available in a frame-accurate manner.

Figure 2 Example of Programmable Shading Results



Double-Precision Coordinate Systems

Standard floating-point numbers fail to provide enough precision for representing position information very far from the origin of the database. This is a problem when rendering very large terrain models (e.g., the earth). This problem manifests itself as a random jitter in the position of database elements. Current OpenGL hardware does not support double-precision numbers for vertex coordinates and matrices; therefore, the solution to the precision problem must come from a higher level layer.

OpenGL Performer solves the precision problem by providing an intermediate double precision coordinate system. When traversing the scene graph, OpenGL Performer allows specifying double precision matrices. It allows the application to drag the origin of the database along with the observer camera. This means that objects near the observer camera are always close to the origin of the database; therefore, the precision problem does not exist.

Enhanced Realism through Programmable Shading

Using multiple rendering passes to compute a shading function on some geometry is a well-known trick. However, implementing the multipass rendering algorithms at the application level has always been a very difficult job. OpenGL Performer provides a new infrastructure for specifying a multipass shading algorithm on some geometry. It reads the output of OpenGL Shader, a toolkit for converting shading specifications into a multipass specification. It then constructs a list of passes for the geometry that it affects. At the rendering stage, it renders the geometry multiple times as specified in the shader description.

Figure 3 Example of Patchy Fog

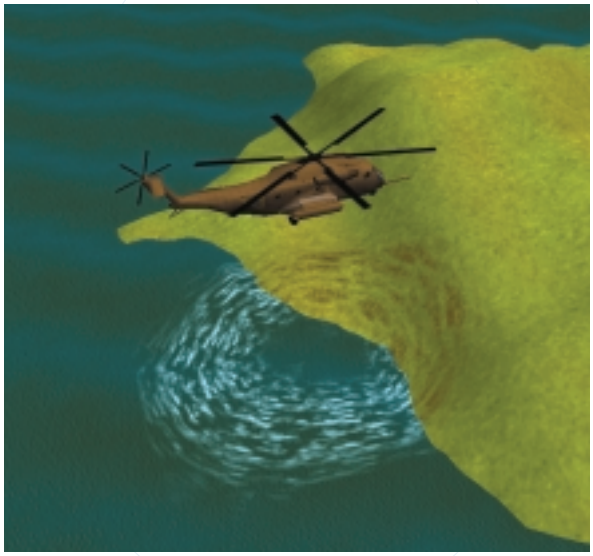
Rendering Features for Realistic Visual Simulation

OpenGL Performer includes special features for enabling realistic environments and visual simulation. One of these features is the layered atmospheric model that includes fog, haze, and an earth/sky model with graduated sky color and horizon glow. On all OpenGL architectures, a multipass algorithm enables layered and patchy fog visual effects. Additional support is provided for implementing a layered angle and range-correct fog model using multipass rendering on the InfiniteReality series of graphics subsystems. OpenGL Performer also offers sophisticated lighting effects, including shadows, and local projected lights such as landing lights and headlights.

OpenGL Performer 2.4 introduces support for a rotorwash effect for visual simulation applications. The effect is generated on any scene-graph geometry and may vary depending on the material properties of the geometry. By automatically detecting the underlying material properties and geometry, the rotorwash effect adjusts the color and appearance of the dynamic texture to achieve the appropriate visual effect.

OpenGL Performer incorporates wide support for visible, nonilluminating light points—essential for accurate renderings of a given view that might include such lights as stars, runway lights, visual approach slope indicators, precision approach path indicators, and even street lights when viewed from a great distance. The computation for these lights can be done in a separate process in parallel with the main rendering process, which can be multithreaded. OpenGL Performer contains full support for calligraphic light points and the management of calligraphic hardware, available from Electronic Image Systems, Inc., for the entire InfiniteReality family of graphics subsystems.

Figure 4 Example of Rotorwash



Increasing Frame Rate by Using DPLEX

On InfiniteReality systems with multiple rendering pipes and a DPLEX option, OpenGL Performer provides support for time-multiplexing the output of the different pipes into a single screen. For example, a five-pipe system that can render a complex model at 12 Hz can now render the same model at 60 Hz. Each one of the five pipes starts drawing its frame at a different time, and the resulting images are multiplexed into the output screen. The final result is that the output screen sees a new image 60 times per second even though each one of the pipes can only produce 12 new images per second.

Scene Graph and Basic Object Types

Libpf Node Types

ASD	Active surface definition evaluates continuous level of detail of terrain
Billboard	Rotates geometry to face eyepoint for efficient rendering of symmetric geometry
DCS	Dynamic coordinate system—applies changeable transformation to the coordinate system of its children
FCS	For asynchronous (fluxed) evaluation of coordinate system transformation
Geode	Contains geometry described with GeoSets and GeoStates
Group	Groups with zero or more children
Layer	Renders co-planar geometry (e.g., pictures on a wall)
LightSource	Invisible but illuminating light source
LOD	Level of detail—selects one or more children based on distance from eyepoint, viewport pixel size, and field of view
Partition	Spatially partitions geometry beneath it into an efficient data structure
Scene	Root node of a visual database
SCS	Static coordinate system—applies static transformation to the coordinate system of its children
Sequence	Sequences through its children for sequenced animation effects
Switch	Enables/disables traversal of children nodes in a group

Additional Libpf Object Types

FrameStats	Holds per-frame timing, computation, and rendering statistics
LODState	Represents custom LOD parameters and priority classes for individual objects or arbitrary groups of objects
MPClipTexture	Manages the paging of a cliptexture for a gfx pipe
Pipe	Software rendering pipeline to hardware gfx pipe
PipeVideoChannel	A video output channel of the gfx pipe
PipeWindow	A window on a gfx pipe
Rotorwash	Enables the vis-sim specific rotorwash effect
ShaderManager, Shader	Containers for multipass rendering specifications
VolFog	Encapsulates layered and patchy fog rendering algorithms

Libpr Objects

Calligraphic	Configures/manages a calligraphic channel for rendering calligraphic light points
Engine	Encodes a standard or custom user-defined operation on a fluxable object
Flux	An object or buffer for dynamic and optionally asynchronous update
GeoSets	Hold geometric primitives: points, lines, line strips, triangles, triangle strips, triangle fans, and quadrilaterals
GeoStates	Hold state description for geometry: texture, material, lighting, transparency, and fog
LpointState	Holds description of light-point illumination behavior

Release and Compatibility Information

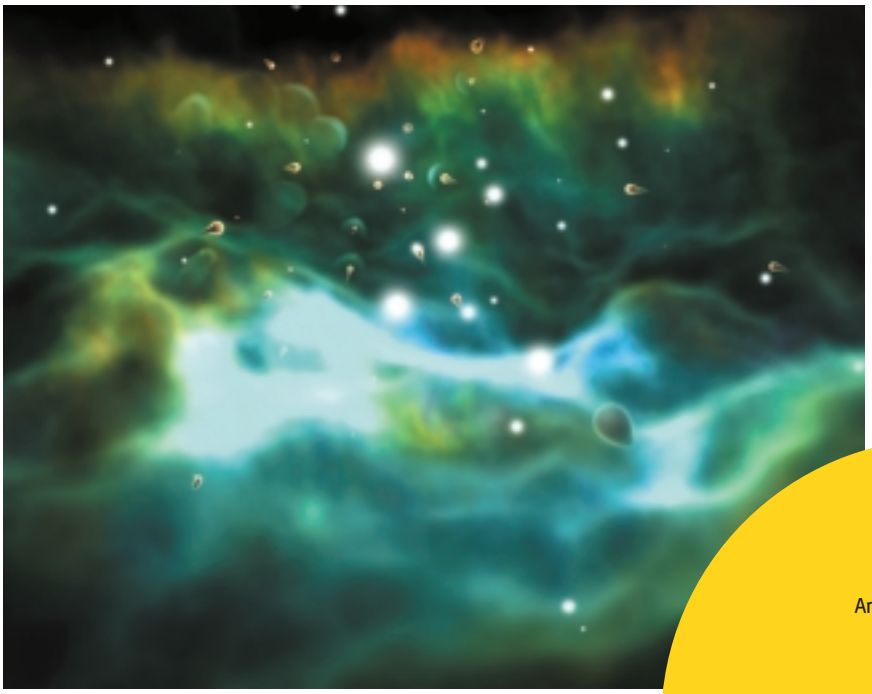
Compatible with all platform releases of OpenGL Performer. Requires IRIX 6.2 or a later IRIX release.

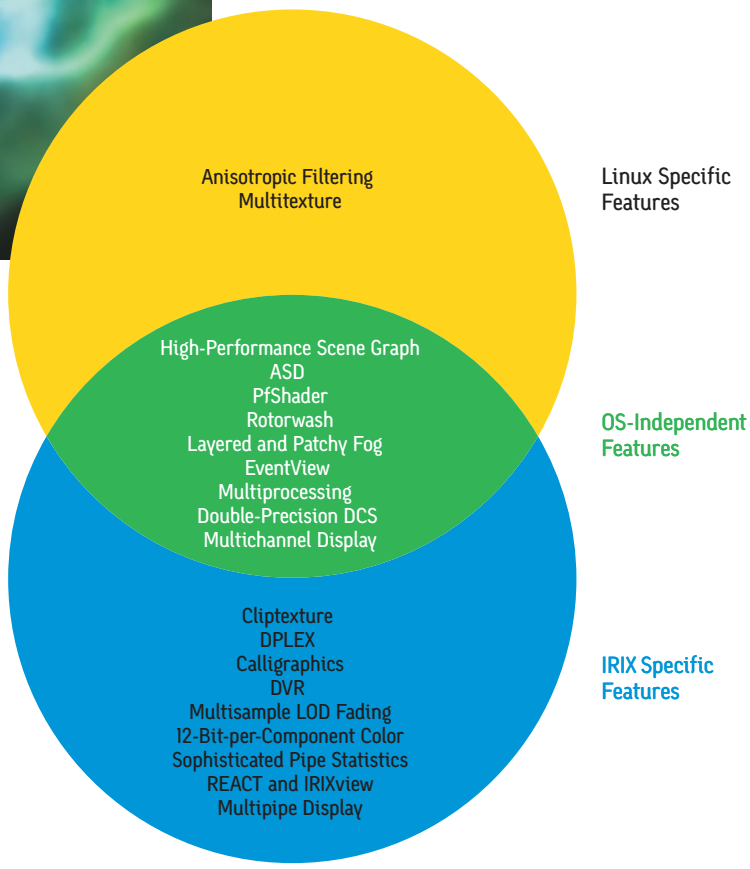
- Supports OpenGL on all platforms
- Supports all ABIs: O32, N32, N64

Includes compatibility execution environments for previous releases:

- 2.0.7—binary-compatible upgrade for 2.0
- 2.1.5—binary-compatible upgrade for 2.1

All execution environments are included in IRIX 6.5.





Further Information

For additional information on OpenGL Performer, visit the Web site at *www.sgi.com/software/performer/*.

Corporate Office
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
(650) 960-1980
www.sgi.com

North America 1(800) 800-7441
Latin America (52) 5267-1387
Europe (44) 118.925.75.00
Japan (81) 3.5488.1811
Asia Pacific (65) 771.0290

3060 (4/01)

J12484