Real-Time Graphics Architecture

Kurt Akeley

Pat Hanrahan

http://www.graphics.stanford.edu/courses/cs448a-01-fall

The OpenGL® Graphics System

Outline

- Introduction and history
- Block diagrams
- Goals and approaches
- Details
- Successes
- Mistakes
- Lessons
- Future

The OpenGL® Graphics System

Web sites

- www.opengl.org
- http://oss.sgi.com/projects/ogl-sample/
- www.sgi.com/software/opengl/license.html

OpenGL is a registered trademark

- Owned by Silicon Graphics
- Must be used as an adjective!
- This is all SGI owns now

OpenGL is controlled by the "ARB"

- Architecture Review Board
- Named by Bill Glazier after Palo Alto's board ☺

OpenGL ARB Members

1991	Compaq (Digital Equipment Corporation)
	IBM
	Intel
	Microsoft
	Silicon Graphics
1994*	Evans & Sutherland
	3D Labs (Intergraph)
1995*	Hewlett Packard
1996*	Sun Microsystems
1998	nVidia
1999	ATI
2001	Apple

Some of the 70+ OpenGL Licensees

3D Labs
Apple
ATI
AT & T
Barco
Be Inc.
Cirrus Logic
Compaq
Cray Research
Daikin
Digital Equipment Corp.
Elsa
Evans & Sutherland
Fujitsu
Harris Computer
Hewlett Packard
Hitachi
Hummingbird Communications
IBM
Intel
Intelligraphics
Interdimension
Intergraph
Japan Radio Company
Microsoft
Miro
Mitsubishi
NEC
nVidia
RasterOps
S3
Samsung
Siemens Nixdorf
Silicon Graphics
Sony
SPEA
Sun Microsystems
Template Graphics Software
Univel

History of OpenGL

1977*	Jim Clark writes -LDS for E&S Multi Picture System
1982	Silicon Graphics incorporated
1983	IRIS GL ships with IRIS 1000 terminal
1985*	IRIS GL licensed to IBM
1986*	Jim Clark and others propose SGL (Simple GL)
1987	SGI and Pixar consider joint API development
	IRIS GL extended with vector commands (e.g. v3f)
1988	SGI ships GTX and Personal Iris
1989	First GL 5.0 documents
1990	OpenGL development begins
	SGI and Microsoft graphics collaboration begins
1991*	OpenGL ARB created

History of OpenGL (cont.)

1992	OpenGL 1.0 completed (June 30)
	OpenGL / PEX debate (panel) at SIGGRAPH '92
	OpenGL course at SIGGRAPH '92
1995	OpenGL 1.1 completed
1996*	OpenGL specification is made public
	SGI ships OpenGL DLL
1997	Fahrenheit agreement between SGI and Microsoft
1998	OpenGL 1.2 completed
1999*	Apple becomes an OpenGL licensee
2000	OpenGL becomes available as open source
2001	OpenGL 1.3 completed

SGI / Pixar Collaboration

"It seems to me that we are trying to merge troff and Scribe into a single hodge-podge. I don't think that we'll get a good result if we continue."

"It seems that a key to a high-performance interface is choosing the right boundary between what and how. … Even the Pixar language has this boundary in it. It still does not support a description like 'lighted like a sunday afternoon in september'."

"I am in touch with Pat Hanrahan. He will send me an updated copy of a document outline today. I am to be working on what I discuss in this note. What next?"

-- Kurt Akeley (to Forest Baskett)

Fahrenheit

SGI promoting and supporting PC OpenGL

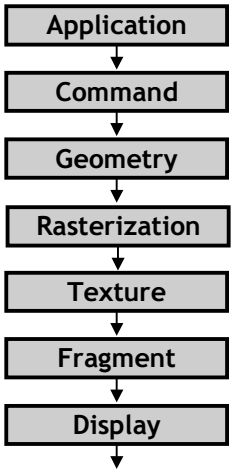
Microsoft asserting control of their own platform

Fahrenheit is the negotiated settlement

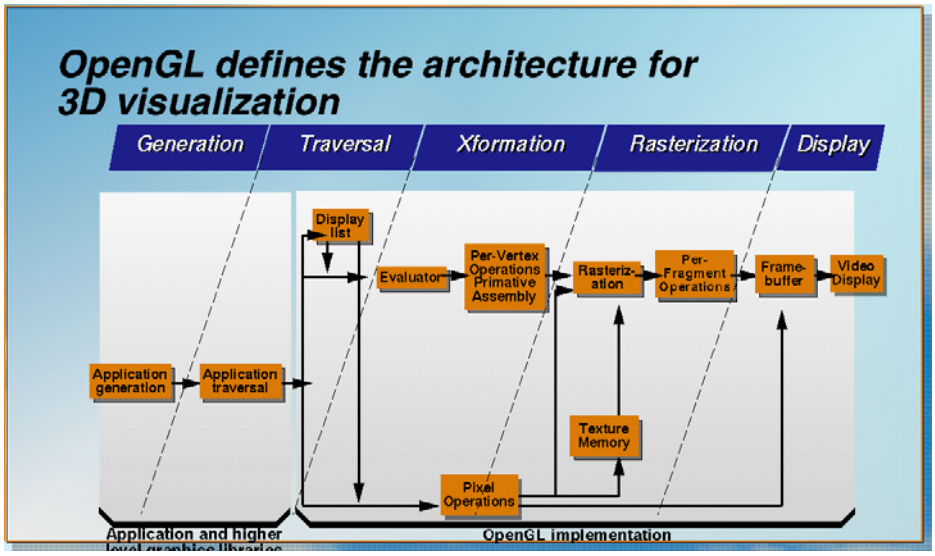
Results:

- Broad reach of agreement failed
  - No new low-level standard, little at scene graph level
- OpenGL still exists on all Windows platforms
- SGI learned a lot about Microsoft's business
- Microsoft learned a lot about graphics
  - Seamus's presentation at Graphics Hardware 2001
- Kurt got mean, stopped dealing with Microsoft

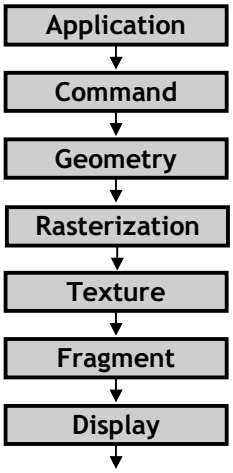
Modern Graphics Pipeline

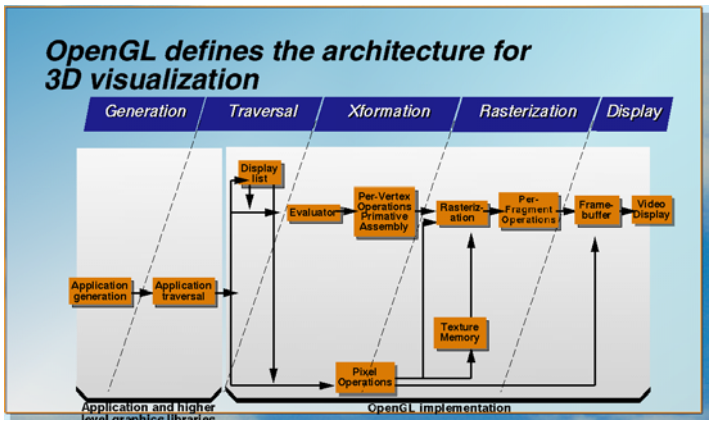


Typical OpenGL Block Diagram

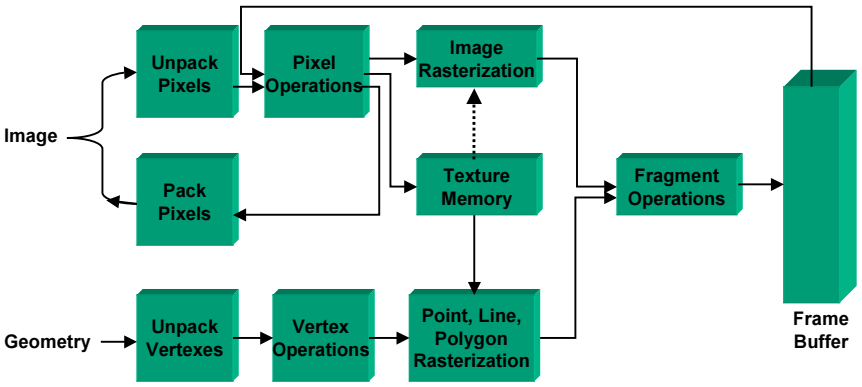


They match





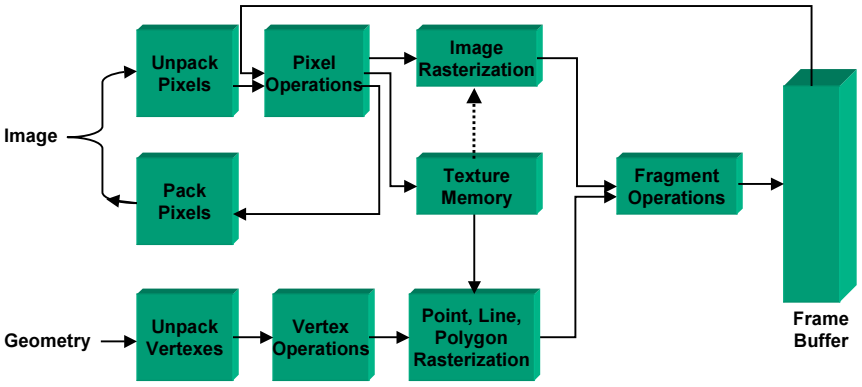
My Favorite OpenGL Block Diagram



My Favorite OpenGL Block Diagram

Symmetry of geometry and imaging paths

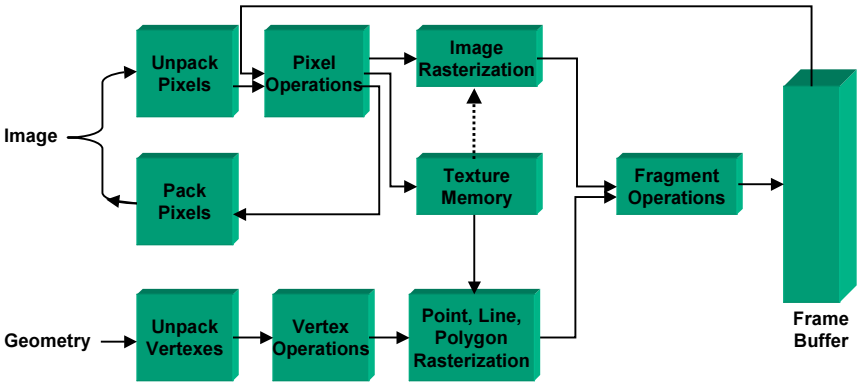
Circulation paths



My Favorite OpenGL Block Diagram

Orthogonal operation

- Vertexes, pixels, fragments, texture



Goals

Goals for OpenGL

Industry-wide acceptance

Consistent implementations

Innovative implementations

Innovative and differentiated applications

Long life

High quality

Non-Goals

Make graphics programming easy

- OpenGL is a power tool

Integrate digital media and 3D graphics

- This is really hard

Goal: Industry-wide acceptance

Avoid compromising performance

- Allow explicit application trade-offs

Get it right the first time

- Make minimum changes from IRIS GL
- Collect lots of input during design
- Do implementation during design

Create an open standard

- Licensing program
- ARB to control future evolution of specification

Goal: Industry-wide acceptance (cont.)

Achieve compatibility with multiple

- Operating systems (MS, Unix, Linux, Mac OS, OS/2, Be, …)
- Window systems (X, Windows)
  - Framebuffer not part of OpenGL state
- Programming languages (C, FORTRAN, Java, …)
  - No pointers, structures, function overloading
  - No 2D arrays at interface (row-major / column-major)

Target Microsoft acceptance

- Omit 2D, window interface, font support
- Allow full application compliance with Windows
- Issue: no driver interface specified

Goal: Industry-wide acceptance (cont.)

Marginalize PEX

- API, not protocol, is the right interface
- But client-server matters too
  - Defined GLX protocol
  - Included server-side display list storage
  - Carefully specified client-side state

Match current hardware capabilities

- Get input from other IHVs

Meet current application needs

- Get input from ISVs

Goal: Consistent Implementations

Tightly written specification

Conformance tests and required verification

Complete implementations

- No sub-setting of 1.*n* specification
- Minimum resource specifications

Required runtime error semantics

- Check and report
- No other side effects

Incentives to share extensions

- Balance desires for consistency and for innovation

Result: portable applications

Goal: Innovative Implementations

Specification not too tight

- Not pixel-exact
- Smooth antialiasing loosely defined
- Object storage abstracted
- Filter attached to texture image

Extensibility

- Key to IHV innovation
- Requires IHV control of entire driver
  - Obvious for systems companies
  - Not at all obvious for Microsoft

Extensibility

SGI maintains registry

- Over 270 extensions so far
- Names, token values, GLX protocol, …

Careful extension documentation

- Extension specification template
- Syntax rules for names
- Suffix/prefix rules
  - Clearly identify all non-core commands and tokens
- Extension numbers
  - Must account for all lower-numbered extensions
  - OpenGL is more than the sum of its parts

Extension Template

Name

Name Strings

Version

Number

Dependencies (list)

Issues

Overview

New Procedures and Functions

New Tokens

Additions to Chapter [2,3,4,5,6]

Additions to the GLX Specification

GLX Protocol

Dependencies (details)

Errors

New State

- Get Value
- Get Command
- Type
- Initial Value
- Attribute Set

New Implementation Dependent State

Extension Syntax Rules

Required abbreviations

Abbreviations specifically not allowed

Compound words

Naming rules

- General
- Procedures (e.g. verb-noun or adjective-noun)
- Defined constants (all caps, underbars)
- Parameters ([0],1,2, <target>, <params>)
- Extensions (prefixes, suffixes)

Parameter order and typing rules

Suffix codes

Extension Categories

Proprietary

- Use corporate prefix/suffix, e.g. SGI

EXT

- Use EXT prefix/suffix
- Must be implemented by at least two licensees

ARB

- Use ARB prefix/suffix
- Specification controlled by the ARB

1.*n*

- No prefix/suffix
- Specification controlled by the ARB

Goal: Innovative Applications

Consistent implementations / tight specification

Design qualities

- Mechanism, not "features"
- Orthogonality
- Circulation
  - Similarity of pixels and texels
  - "Machine Shop" analogy
- Sufficiency of capability (e.g. stencil)

Invariance specification

- Support multi-pass algorithms

Intuitive usability

Goal: Long Life

Extensibility

Anticipate important trends

- Integration of image processing and 3D graphics

Goal: High Quality

Beauty counts

SGI culture, very different from Microsoft!

Avoid design-by-committee

- Build a horse, not a camel

Correct well-known IRIS GL deficiencies

- No command prefixes
- Can't be used by libraries (incomplete save/restore)
- Error reporting via printf() !

Provide documentation

- Specification
- Man pages
- Programming guide

Some Details

Object Ids

- Support display list "editing", scripting

Evaluation parameter semantics

- Allow evaluation to share vertex engine(s)

Direct render to texture

- Would constrain memory optimization

Sifdv interface

- For application convenience and efficiency

Discourage incremental matrix arithmetic

- No pre-multiply
- Stack provided for "undo"

Some Details (cont.)

Application-specified clipping done in eye coordinates

- Allow use of singular projection matrixes

Unit area pixels and texels

- Matches window system notion of pixel ownership
- Makes frustum and viewport calculations obvious
- Makes texture wrapping arithmetic obvious
- Matched pixel and texel addressing supports circulation

Success With Respect to Goals

Industry-wide acceptance

- Great for Unix, Linux, embedded systems
- Good as can be expected with Microsoft ☺
- Poor for game consoles

Consistent implementations

- Good for core features
- Marginal for new features

Innovative implementations

- Industry and academic standard
- Interesting work-arounds for constraints

Success With Respect to Goals (cont.)

Innovative and differentiated applications

- Very successful

Long life

- Almost a decade so far

High quality

- Generally respected
- Story of Carmack endorsement

Technical Successes

Procedural interface

- "Every graphics systems aspires to have a procedural interface"

Generalized texture capability

- All primitives (with pixel texture extension)
- Various dimensions of texture
- Texel and pixel conformance
- Incremental in-place modification
- Explicit format, filter specification, MIP level, …
- Texture matrix

Formalization of fragment

Mistakes

Business

- Taking our eye off Microsoft

Technical

- Display lists as encapsulation for "objects"
  - Re-introduction of texture objects
- Overly-abstract storage management
  - Proxy-priority middle ground is not tenable
- No "fast path" clues in API
  - Too much faith in pure mechanism
  - E.g. DL objects, window position, 2D transform, in-place framebuffer to texture memory transfer, …
- Persistent parameter state between Begin/End
  - But would have broken IRIS GL programs badly

Mistakes (cont.)

Probably should have been omitted

- Texture borders!
- Edge flags
- Polygon antialiasing
- Color index features (lighting, antialiasing, …)

Needlessly complex or wrong

- ColorMaterial()
- Normal transformation
- Texture wrap semantics
- Bit fields

Lessons

It is worthwhile to specify carefully

- Write the spec first, or you'll be sorry!

It is critical to guide system implementers and application developers to a common understanding of how to support and achieve high performance.

Beware Microsoft

People make and evolve standards

- Invest in developing a culture
- Keep competitive issues out of a specification's controlling body.

Future of OpenGL

On the rebound

- Low period in the late 90s

Provides alternative to DirectX/Microsoft API control

Apple commitment is a positive sign

Programmability is an exciting development

- Fits well in OpenGL structure
- OpenGL may truly be a "library" some day

Lots of 2.0 activity, ARB still functioning

Tenth birthday next summer

Real-Time Graphics Architecture

Kurt Akeley

Pat Hanrahan

http://www.graphics.stanford.edu/courses/cs448a-01-fall