Real-Time Graphics Architecture

Kurt Akeley

Pat Hanrahan

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

About Kurt

Personal history

- B.E.E. Univeristy of Delaware, 1980
- M.S.E.E. Stanford, 1982
- SGI co-founder, chief engineer, CTO, 1982 – 2000
- Lots of SIGGRAPH involvement

Currently

- Reinstated as Ph.D. student here (almost ;-)
- Working at NVIDIA on Fridays
- Book on this subject

Other Notes

OpenGL

- Lots of history with this
- Good framework for understanding

Glossary

- On-line soon

Feedback

- Yes!

Outline

Introductions (done)

Evolution of Graphics Systems (Kurt)

Future Evolution (Pat)

Lecture Schedule (Pat)

Brief Introduction to Perception (Kurt)

Course Logistics (Pat)

Evolution of Real-time Graphics

Don't have a genealogy chart

- (This would be a great project)

Some important phases

- Early research
- Flight simulation
- GL-like: Terminal -> SGI -> PC

We'll focus on GL-like

- Attempt to credit research and simulation results

Axes of Improvement

Performance

- Triangles / second
- Pixel fragments / second

Features

- Hidden surface elimination
- Image mapping
- Antialiasing

Quality

- Bits of numeric resolution
- Image filters

Relationships of Axes

Software is the base line

- Performance is inversely proportional to algorithm complexity

Hardware behavior differs from software

- Performance is invariant to complexity, or
- Performance falls off catastrophically
- If use of feature X is "free" (meaning that it imposes no penalty in performance) then rendering without using feature X is too slow!

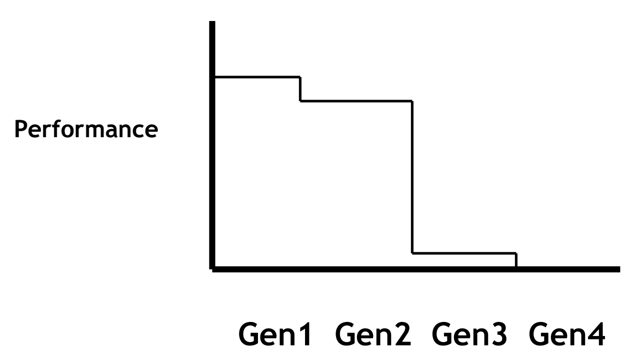
Pipelining leads to "free" features

- Traditional parallelism typically doesn't

Generations (GL-like machines)

Generations are defined in terms of feature sets

Only features that are included in the performance plateau count



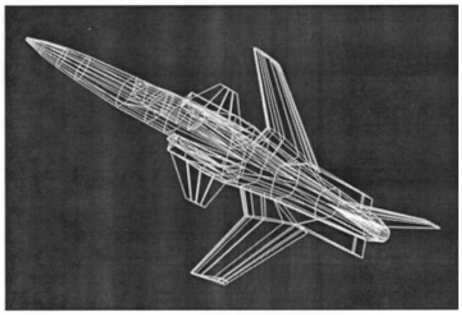
First Generation - Wireframe

Vertex: transform, clip, and project

Rasterization: color interpolation (points, lines)

Fragment: overwrite

Dates: prior to 1987



Second Generation - Shaded Solids

Vertex: lighting calculation

Rasterization: depth interpolation (triangles)

Fragment: depth buffer, color blending

Dates: 1987 - 1992

Third Generation - Texture Mapping

Vertex: texture coordinate transformation

Rasterization: texture coordinate interpolation

Fragment: texture evaluation, antialiasing

Dates: 1992 - 2000

SGI Historicals

Year	Product	Fill rate	Yr rate	Tri rate	Yr rate
1984	Iris 2000	46M	-	10K	-
1988	GTX	80M	1.2	135K	1.9
1992	RealityEngine	380M	1.5	2M	2.0
1996	InfiniteReality	1000M	1.3	12M	1.6
			1.3		1.8

SGI Historicals

Gen	Year	Product	Fill rate	Yr rate	Tri rate	Yr rate
1st	1984	Iris 2000	46M	-	10K	-
2nd	1988	GTX	80M	1.2	135K	1.9
3rd	1992	RealityEngine	380M	1.5	2M	2.0
3rd	1996	InfiniteReality	1000M	1.3	12M	1.6
				1.3		1.8

SGI Historicals (Depth Buffered)

Gen	Year	Product	Zbuf rate	Yr rate	ZTri rate	Yr rate
1st	1984	Iris 2000	100K	-	0.8K	-
2nd	1988	GTX	40M	4.5	135K	3.6
3rd	1992	RealityEngine	380M	1.8	2M	2.0
3rd	1996	InfiniteReality	1000M	1.3	12M	1.6
				2.2		2.2

Yearly Growth well above Moore's Law

*n*VIDIA Graphics growth (225%/yr)

Season	Product	Process	# Trans	Gflops	32-bit AA Fill	Mpolys	Notes
2H97	Riva 128	.35	3M	5	20M	3M	Integrated 2D/3D
1H98	Riva ZX	.25	5M	7	31M	3M	AGP2x
2H98	Riva TNT	.25	7M	10	50M	6M	32-bit
1H99	TNT2	.22	9M	15	75M	9M	AGP4x
2H99	GeForce	.22	23M	25	120M	15M	HW T&L
1H00	GeForce2	.18	25M	35	200M[1]	25M	Per-Pixel Shading
2H00	NV16	.18	25M	45	250M[1]	31M	230 Mhz DDR
1H01	NV20	.15	55M	80	500M[1]	30M[2]	Programmable

Essentially Moore's Law ***Cubed***.

1: Dual textured
2: Programmable

*n*VIDIA Historicals

Season	Product	Fill rate	Yr rate	Tri rate	Yr rate
2H97	Riva 128	20M	-	3M	-
1H98	Riva ZX	31M	2.4	3M	1.0
2H98	Riva TNT	50M	2.6	6M	4.0
1H99	TNT2	75M	2.3	9M	2.3
2H99	GeForce	120M	2.6	15M	2.8
1H00	GeForce2	200M	2.6	25M	2.8
2H00	NV16	250M	1.6	31M	1.5
1H01	NV20	500M	4.0	30M	1.0
			2.5		2.2

Yearly Growth well above Moore's Law

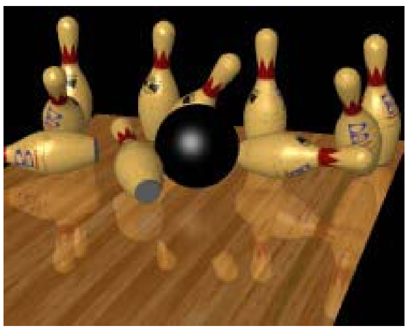
Fourth Generation - Programmability

Programmable shading

Image-based rendering

Convergence of graphics and media processing

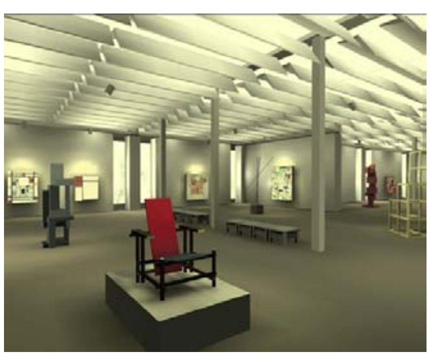
Curved surfaces

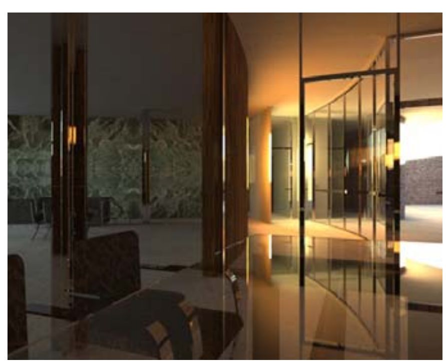


Fifth Generation - Global Evaluation

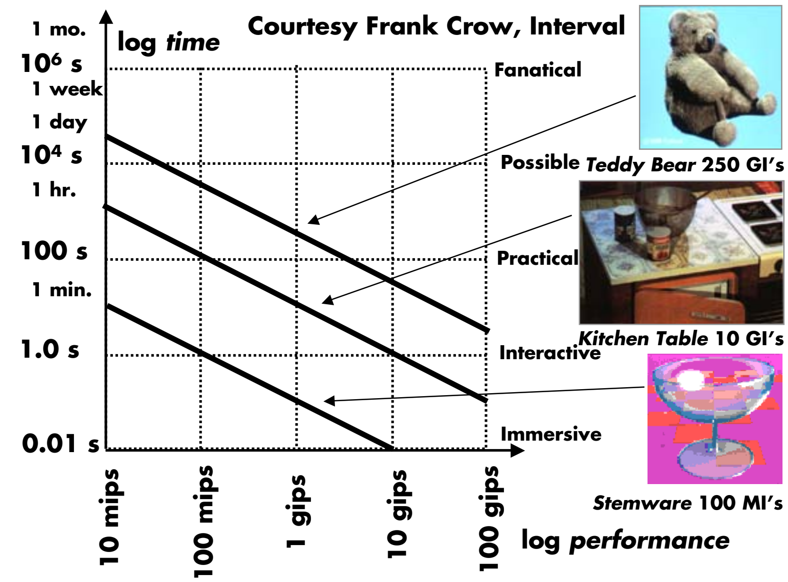
Ray tracing: visibility and integration

True shadows, path tracing, photon mapping





From Batch to Interactive



CS448 Lecture 1 Kurt Akeley, Pat Hanrahan, Fall 2001

Perception

Interactive graphics is all about eye candy

- Not going to discuss bridge design

Good designers know their customers

- Understand visual perception
- NTSC is a great example

References

- *Foundations of Vision*, Brian Wandell
- *A Technical Introduction to Digital Video*, Charles Poynton

CS448 Lecture 1 Kurt Akeley, Pat Hanrahan, Fall 2001

Perception Issues

Intensity

Motion

Latency

Color

Resolution

....

Demo

Intensity and motion

Intensity

Eye has nonlinear response to intensity

- Minimum visible (static) contrast ratio is 1%
- Brightness = k Intensity $^{0.4}$

CRT has nonlinear response to input signal

- Intensity = a Input ^{gamma} + b

Combined response is near-linear

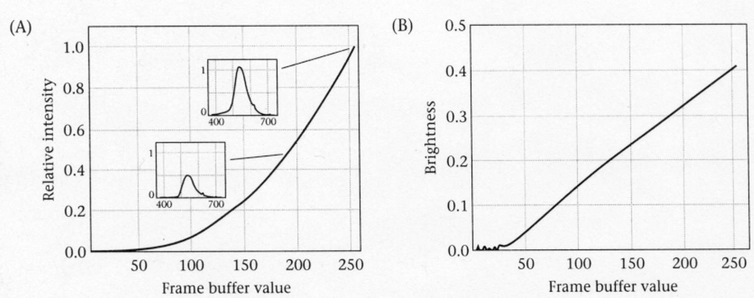
- 0.4 (2.2) ~= 1.0
- Suggests that 8-bit DAC would handle 100x range

But …

- Color arithmetic is done in linear intensity

Graphs

Foundations of Vision, Wandell, p. 416



B.2 FRAME-BUFFER VALUE AND DISPLAY INTENSITY. (A) The curve measures the intensity of the emitted light relative to the maximum intensity. The data shown are for the green phosphor. The insets in the graph show the complete spectral power distribution of the light at two different frame-buffer levels. (B) The curve describing the relative intensities is replotted, using Stevens's power law, to show the linear relationship between the frame-buffer value and perceived brightness.

Gamma Correction

Store image linear in brightness

- Best use of available storage precision
- 256 representable levels are enough
- Requires conversion for each pixel operation
- Historically unusual design choice

Store image linear in intensity

- Native arithmetic format
- Requires conversion during display
- Large brightness steps at low intensities
- 256 representable levels are not enough!
- Historically typical design choice

Motion

Eye is sensitive to motion and change (only)

- Rule 1: Avoid substantial frame-to-frame changes

Animation

- No flicker detection above 80Hz or so
- Sequence of frames is interpreted as continuous
- Corollary to rule 1: Evaluate sequences of images

Eye/brain combination tracks motion

- Image doubling if render and display rates differ
- Interlace artifacts if render and field rates differ
- Separation if colors are displayed sequentially

Latency

Latency is a *critical* system issue

- GPU is just a link in the latency chain
- Latency "budget" is sum of all delays

Human latency thresholds

- Hand-eye (fixed display) is ~100ms
- Head-eye (head-mounted display) is ~10ms
- Regan, Matthew and Pose, Ronald, *An Interactive Graphics Display Architecture*, Proceedings of IEEE Virtual Reality Annual International Symposium, 18-22 September 1993, Seattle USA

Color

Three cone types (S, M, L)

Color can be represented as a 3-tuple

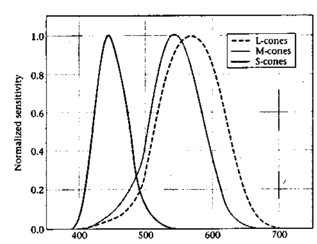
- RGB is convenient for display (L,M,S)
- Other tuples for other purposes

Cone densities differ

- S (blue) cones low density
- L,M (red, green) cones higher density

Color arithmetic

- Independent R, G, and B calculations are wrong
- 3x3 matrix arithmetic is required



Resolution

Eye's resolution is not evenly distributed

- Foveal resolution is ~20x peripheral
- Can track direction of view
- Flicker sensitivity higher in periphery

Static and dynamic resolutions differ

- Think of screen-door effect

One eye can compensate for the other

- Research at NASA suggests high-resolution dominant display

....

Real-Time Graphics Architecture

Kurt Akeley

Pat Hanrahan

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