# 3D Graphics Hardware

Hannes Kaufmann

Interactive Media Systems Group (IMS)

Institute of Software Technology and Interactive Systems

Thanks to Dieter Schmalstieg and Michael Wimmer for providing some slides/images/diagrams

# Motivation

- Last week: Various VR Application Areas
- For development of VR environments = Hardware configurations + Applications detailled knowledge is needed about
  - Hardware: 3D Graphics, Input Devices & Tracking, Output Devices
  - Software: Standards, Toolkits, VR frameworks
  - Human Factors: Usability, Evaluations, Psychological Factors (Perception,…)

# 3D Graphics Hardware - Development

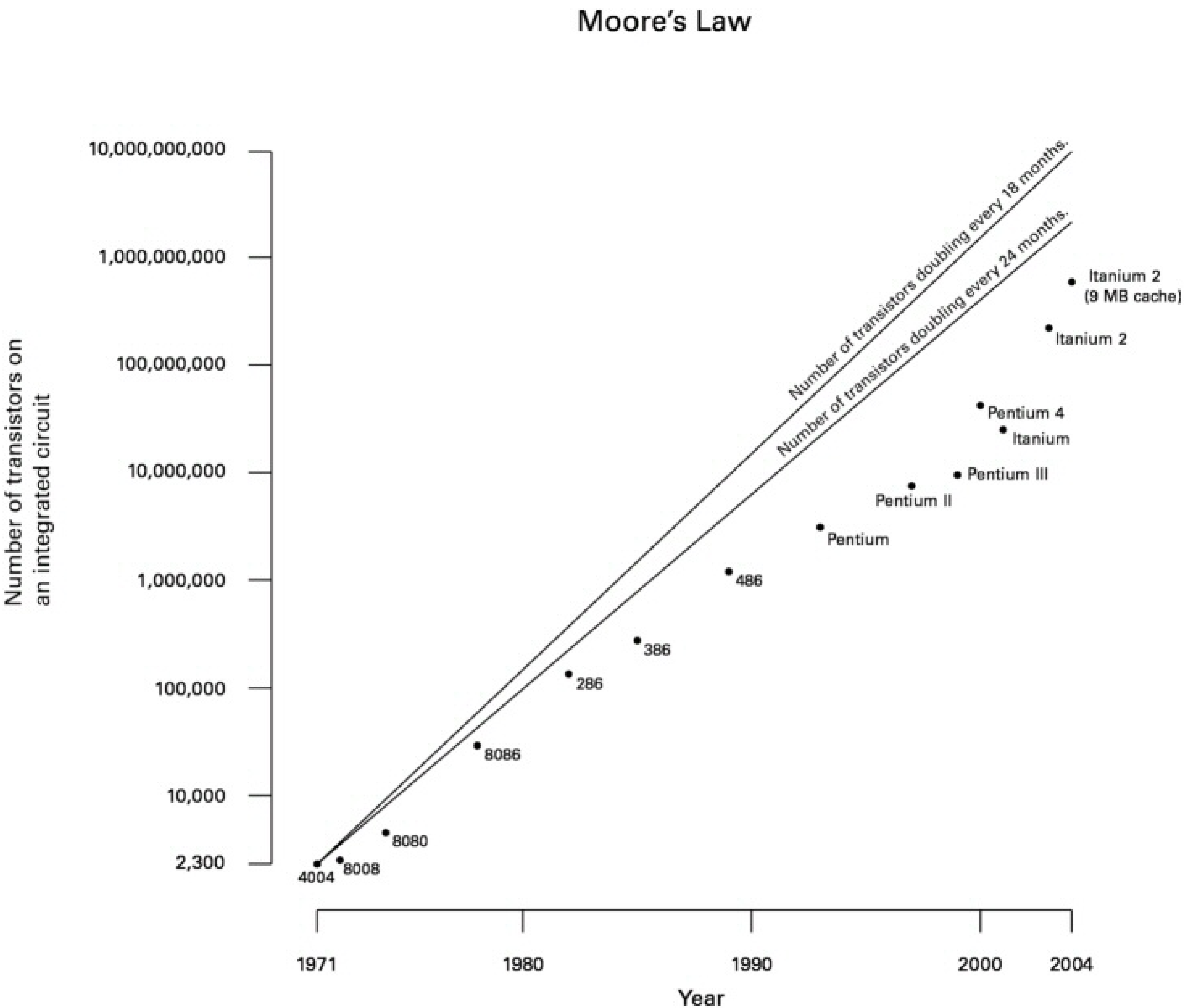
- Incredible development boost of consumer cards in previous ~10 years
- PC graphics surpassed workstations (~2001)
- Development driven by Game Industry

# Consumer Graphics – Major Points

- Up to 1995
  - 2D only (S3, Cirrus Logic, Tseng Labs, Trident)
- 1995 Scanlines (Proprietary APIs)
- 1996 3DFX Vodoo (first real 3D card); Introduction of DX3
- 1997 Triangle rendering (… DX5)
- 1998 Triangle setup (…DX6)
- 1999 Multi-Pipe, Multitexture (…DX7)
- 2000 Transform and lighting (…DX8)
  - finally caught up to full 3rd generation!
- 2001 Programmable shaders
  - PCs surpass workstations, 4th generation
- 2002 Full floating point
- 2004 Full looping and conditionals

# Moore‘s Law

- Gordon Moore, Intel co-founder, 1965
- Exponential growth in number of transistors
- Doubles every 18 months (holds for CPUs)
  - yearly growth: factor 1.6
  - But: stagnation lately (2.8GHz available since December 2002); now: Dual Core CPUs


Moore's Law
10,000,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
Number of transistors on an integrated circuit
Number of transistors doubling every 18 months.
Number of transistors doubling every 24 months.
4004
8008
8080
8086
286
386
486
Pentium
Pentium II
Pentium III
Pentium 4
Itanium
Itanium 2
Itanium 2 (9 MB cache)
1971
1980
1990
2000
2004
Year

# SGI Development (Workstations)

| Gen | Year | Product | Fill rate (no Z) | Yr rate | Tri rate (no Z) | Yr rate |
|---|---|---|---|---|---|---|
| 1st | 1984 | Iris 2000 | 100K (46M) | - | 0.8K (10K) | - |
| | | | | | | |
| 2nd | 1988 | GTX | 40M (80M) | 1.2 | 135K | 1.9 |
| | | | | | | |
| 3rd | 1992 | RealityEngine | 380M | 1.5 | 2M | 2.0 |
| | | (Cost: 1,000,000$, size of a domestic fridge!) | | | | |
| 3rd | 1996 | InfiniteReality | 1000M | 1.3 | 12M | 1.6 |
| | | | | | | |
| | | | | 1.3 | | 1.8 |

- Yearly tri rate growth above Moore's law!

Hannes Kaufmann
Virtual Reality

# Nvidia Development 1/2

| Season | Product | Proces | # Trans | MHz | 32-bit AA[1] Fill | MPolys | New Features |
|---|---|---|---|---|---|---|---|
| 2H97 | Riva 128 | .35 | 3M | ? | 20M | 3M | Integrated 2D/3D |
| 1H98 | Riva ZX | .25 | 5M | ? | 31M | 3M | AGP 2x |
| 2H98 | Riva TNT | .25 | 7M | 100 | 50M[2] | 6M | 32-bit, Tri setup |
| 1H99 | TNT2 (Ultra) | .22 | 9M | 150 | 75M | 9M | AGP4x |
| 2H99 | GeForce256 | .22 | 23M | 120 | 120M[3] | 15M | HW T&L |
| 1H00 | GeForce 2 GTS | .18 | 25M | 200 | 200M[4] | 25M | Per-Pixel Shading |
| 2H00 | Geforce 2 Ultra | .18 | 25M | 250 | 250M | 31M | |
| 1H01 | GeForce 3 | .15 | 57M | 200 | 417M[5] | 25M | Vertex Shading |
| 2H01 | GeForce 3 Ti500 | .15 | 57M | 240 | 500M | 30M | |
| 1H02 | GeForce 4 | .15 | 63M | 300 | 625M | 75M[6] | (Dual display) |
| 1H03 | GeForce FX 5800 | .13 | 125M | 500 | 1041M | 375M | Floating Point |
| 2H03 | GeForce FX 5900 | .13 | 130M | 450 | 938M | 338M | (faster shading) |
| 2H04 | GeForce FX 6800 | .13 | 222M | 400 | ~2000M | 600M | looping |

# Nvidia Development 2/2

| Season | Product | 32-bit AA Fill | Yr rate | MPolys | 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 | TNT 2 | 75M | 2.3 | 9M | 2.3 |
| 2H99 | GeForce256 | 120M | 2.6 | 15M | 2.8 |
| 1H00 | GeForce 2 GTS | 200M | 2.6 | 25M | 2.8 |
| 2H00 | Geforce 2 Ultra | 250M | 1.6 | 31M | 1.5 |
| 1H01 | GeForce 3 | 416M | 2.5 | 25M | 0.6 |
| 2H01 | GeForce 3 Ti500 | 500M | 1.4 | 30M | 1.4 |
| 1H02 | GeForce 4 | 625M | 1.6 | 75M | 6.3 |
| 1H03 | GeForceFX 5800 | 1041M | 1.7 | 375M | 5 |
| 2H03 | GeForceFX 5900 | 938M | 0.8 | 338M | 0.8 |
| 2H04 | GeForceFX 6800 | ~2000M | 2.1 | 600M | 1.8 |
| | (Cost: 500 Euro) | | 2.1 | | 2.5 |
| 2H05 | GeForce 7800 | | | 860M | |

Almost Moore's law squared (^1.5-2.0)

Performance doubles every 9-12 months!

# Mobile Graphics

| | GeForce Go 6800 Ultra (10 months ago) | GeForce Go 7800 GTX (Today) |
|---|---|---|
| **Shader Performance (Pixels per Clock)** | 1X | 2X |
| **Geometry Performance** | 1X | 1.6X |
| **Transistors** | 202 Million | 302 Million |
| **Power Consumption** | 1X | 1X |
| **Package** | Identical | Identical |
| **Memory Interface** | 256-Bit | 256-Bit |
| **Clock Frequencies (Core/Memory)** | 450MHz/550MHz | 400MHz/550MHz |
| **DirectX 9.0** | Shader Model 3.0 | Shader Model 3.0 |
| **Transparency AA** | No | Yes |
| **Programmable Video Processor** | PureVideo | PureVideo |
| **Power Management** | PowerMizer 5.0 | PowerMizer 6.0 |
| **Host Interface** | PCI-Express | PCI-Express |

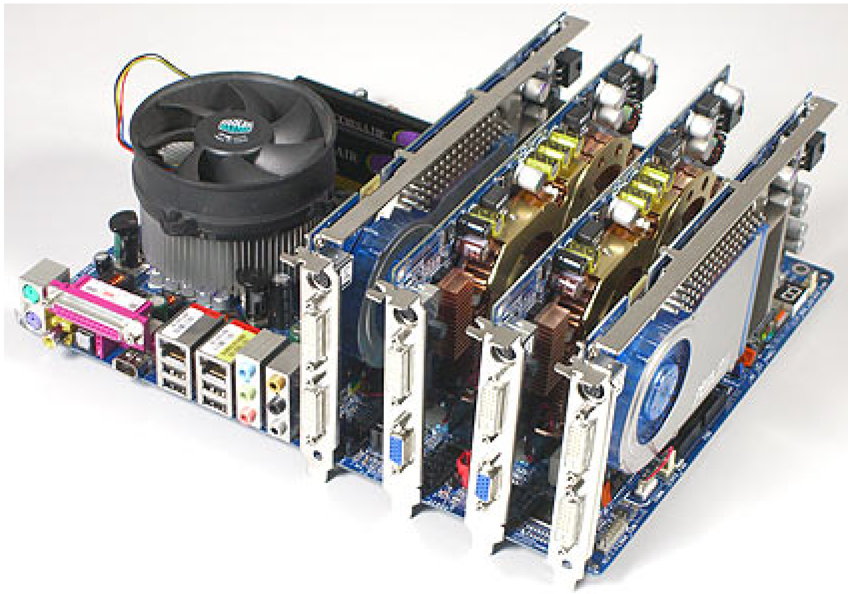
Complies to Moore‘s Law too

Similar performance increase

Hannes Kaufmann
Virtual Reality

# And it goes on and on….

- Performance increase expected to continue within the next few years
  - Smaller chip production processes possible (currently 110nm)
  - More pixel pipelines
  - Multi-core GPUs
  - Multiple graphics cards in a PC
- General purpose computing on GPUs



# 3D Card High End Model

- nVidia Quadro FX 4500 (~ € 1700.- ), 512MB
- Based on G70 chip (such as GeForce 7800)
- 24 Pixel Pipelines, 8 Vertex Pipelines
- 33.6 GB/s Bandwidth
- 10.32 billion pix/sec.
- Vertices: 860 M/s
- optimized OpenGL drivers (comp. to consumer card)

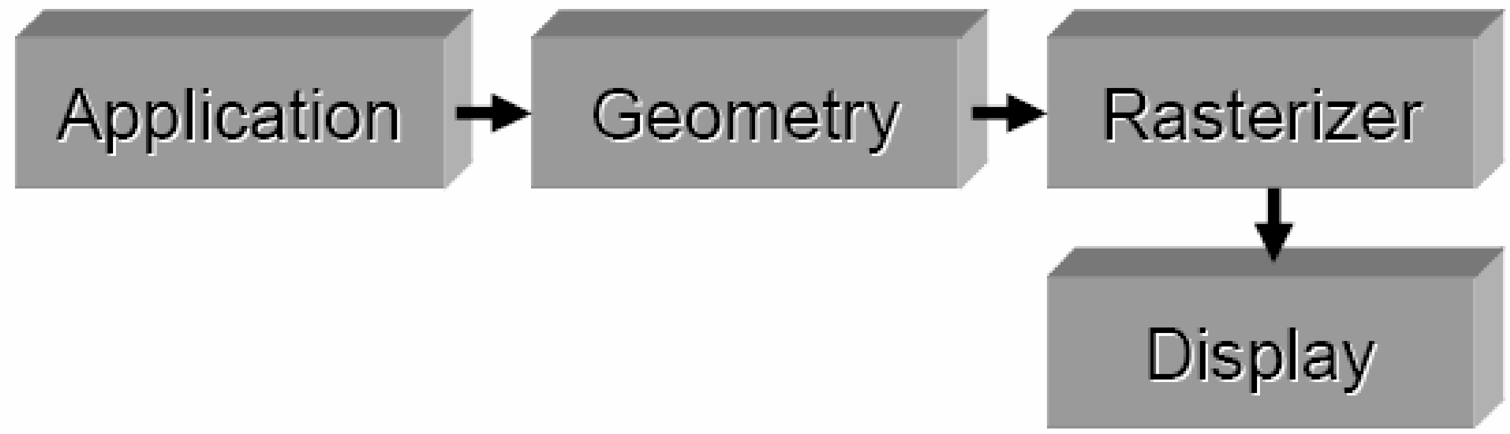
# Some Relevant Features (for VR)

- Memory size: 5[illegible]MB
- Dual DVI / VGA Output (up to [illegible]40x2400)
- Multi-Display support
- OpenGL quad-buffered stereo (3-pin sync connector)
- Scalable Link Interface (SLI™) Technology
- Edge Blending for Powerwalls (supported by driver)
- G-Sync Option Board with Framelock and Genlock
- Quality: Full-Scene Anti-aliasing (FSAA); OpenGL 2.0; Shader Model 3.0,…

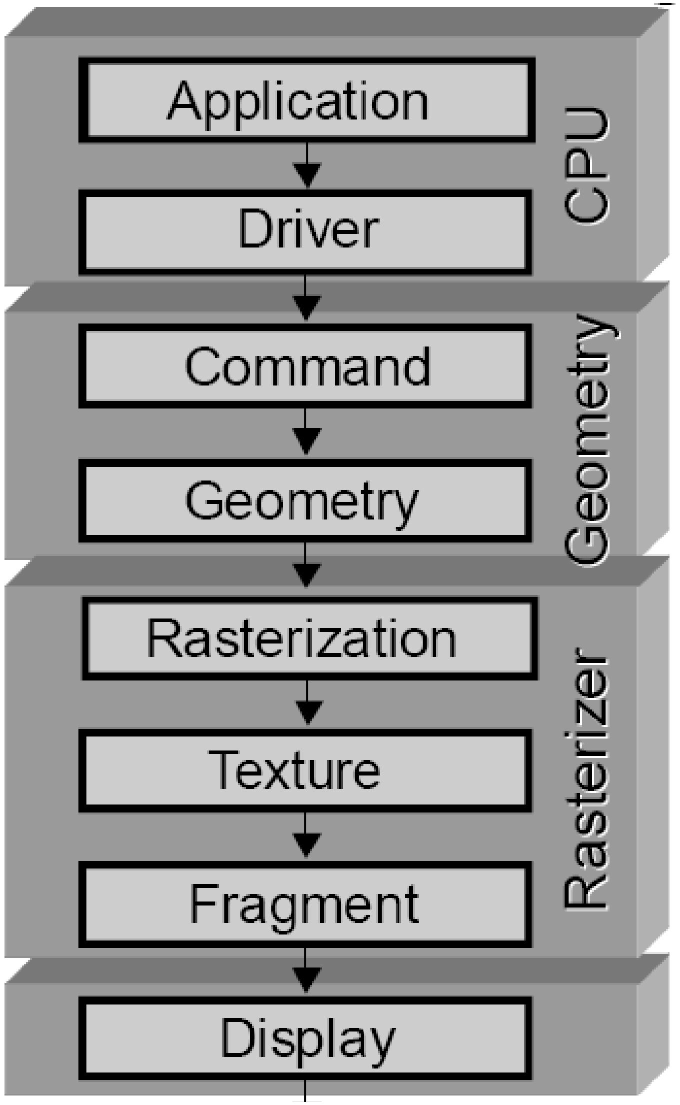
# Explanations
# &
# Back to the Basics

# 3D Graphics
# Basics

# The Graphics Pipeline



# Modern Graphics Pipeline




- Nowadays Geometry and Rasterizer Stage completely HW accelerated

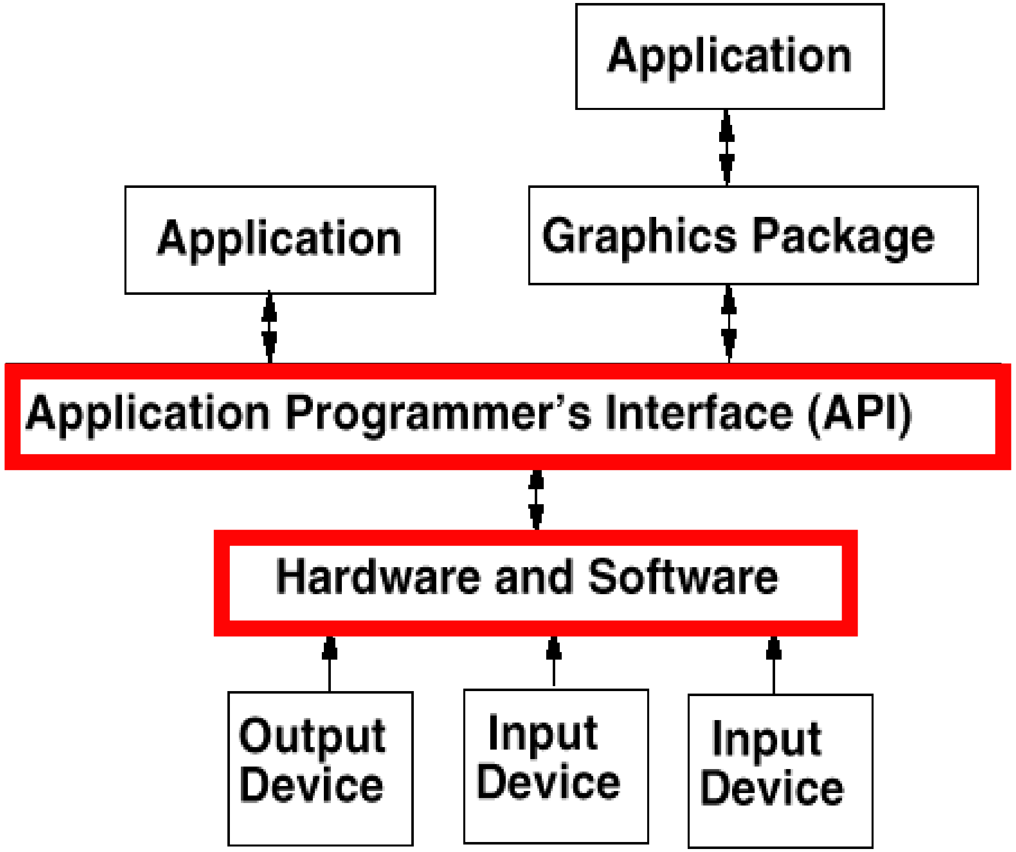
- Fragment: „pixel“ with additional information (alpha, depth, stencil,…)

# Application Stage
# 3D Graphics Programming

3D Application Programmer‘s Interfaces (APIs)

- Access to Hardware
- Standards: OpenGL, Direct3D
- Language: C (mostly)
- Higher Level APIs (OpenInventor, Java3D, Performer,…) based on Standards

# Application Programmer's View



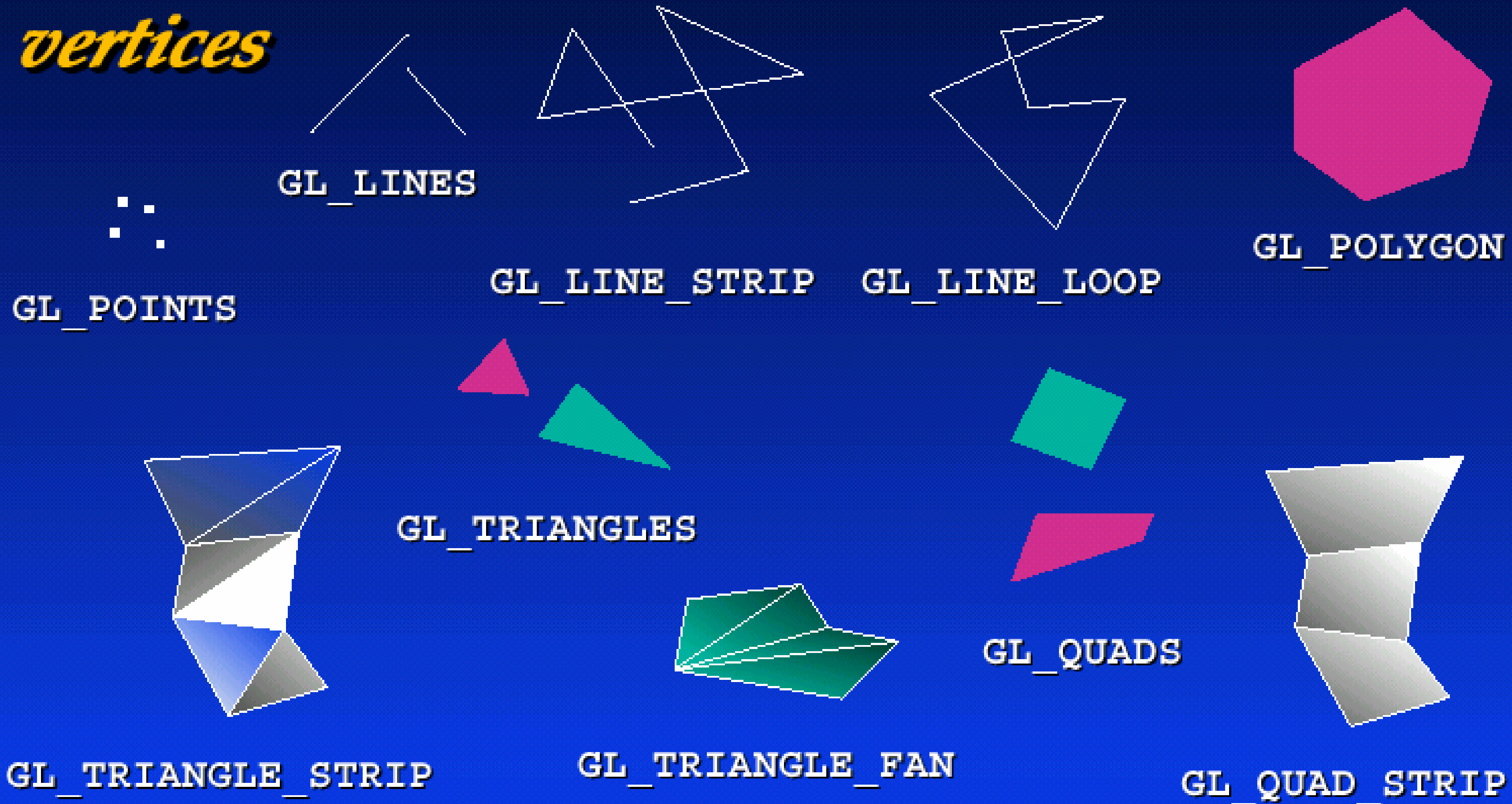

Hannes Kaufmann
Virtual Reality

# OpenGL – Hello World

```
#include <GL/glut.h>

void display(void) {
    glClear (GL_COLOR_BUFFER_BIT);
    /* draw white polygon (rectangle) with
    corners at (0.25, 0.25, 0.0)
    and (0.75, 0.75, 0.0) */
    glColor3f (1.0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f (0.25, 0.25, 0.0);
    glVertex3f (0.75, 0.25, 0.0);
    glVertex3f (0.75, 0.75, 0.0);
    glVertex3f (0.25, 0.75, 0.0);
    glEnd();
    glFlush ();
}

void init (void) {
    glClearColor (0.0, 0.0, 0.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE
    GLUT_RGB);
    glutInitWindowSize (250, 250);
    glutInitWindowPosition (100, 100);
    glutCreateWindow ("hello");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

# OpenGL Geometric Primitives

***All geometric primitives are specified by vertices***



# Application Stage

- Generate geometric primitives, camera properties, lightning, object properties (color, texture,…)
- Input event handling
- Database traversal
- Optimizations possible: Build hierarchy, Level of Details, Culling Techniques, Impostors, Triangulation,…

# Not yet in Hardware

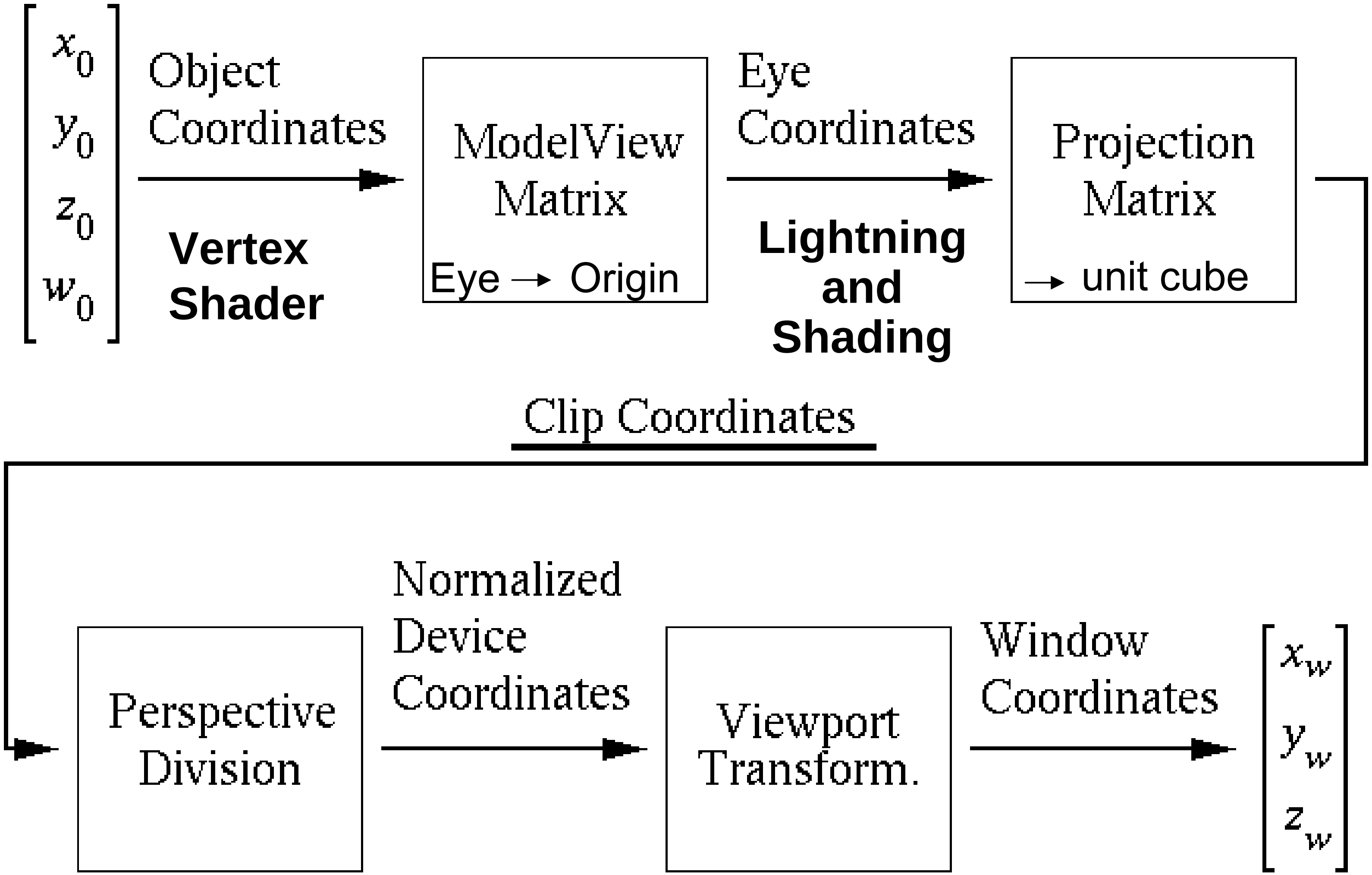
- Evaluation of polynomials for curved surfaces
- Create vertices (tesselation/triangulation)

# Driver

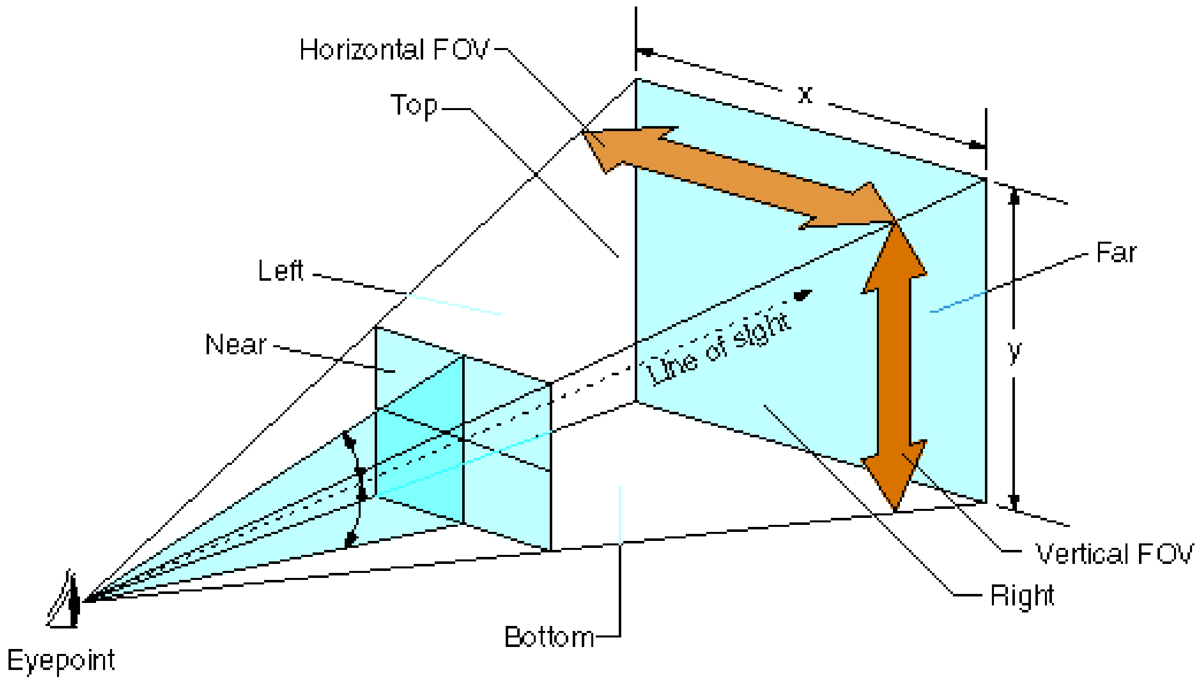
- Command interpretation/translation
  - Host commands → GPU commands
- Handle data transfer
- Memory management
- Emulation of missing features

Hannes Kaufmann
Virtual Reality


# The Geometry Stage (Simple)

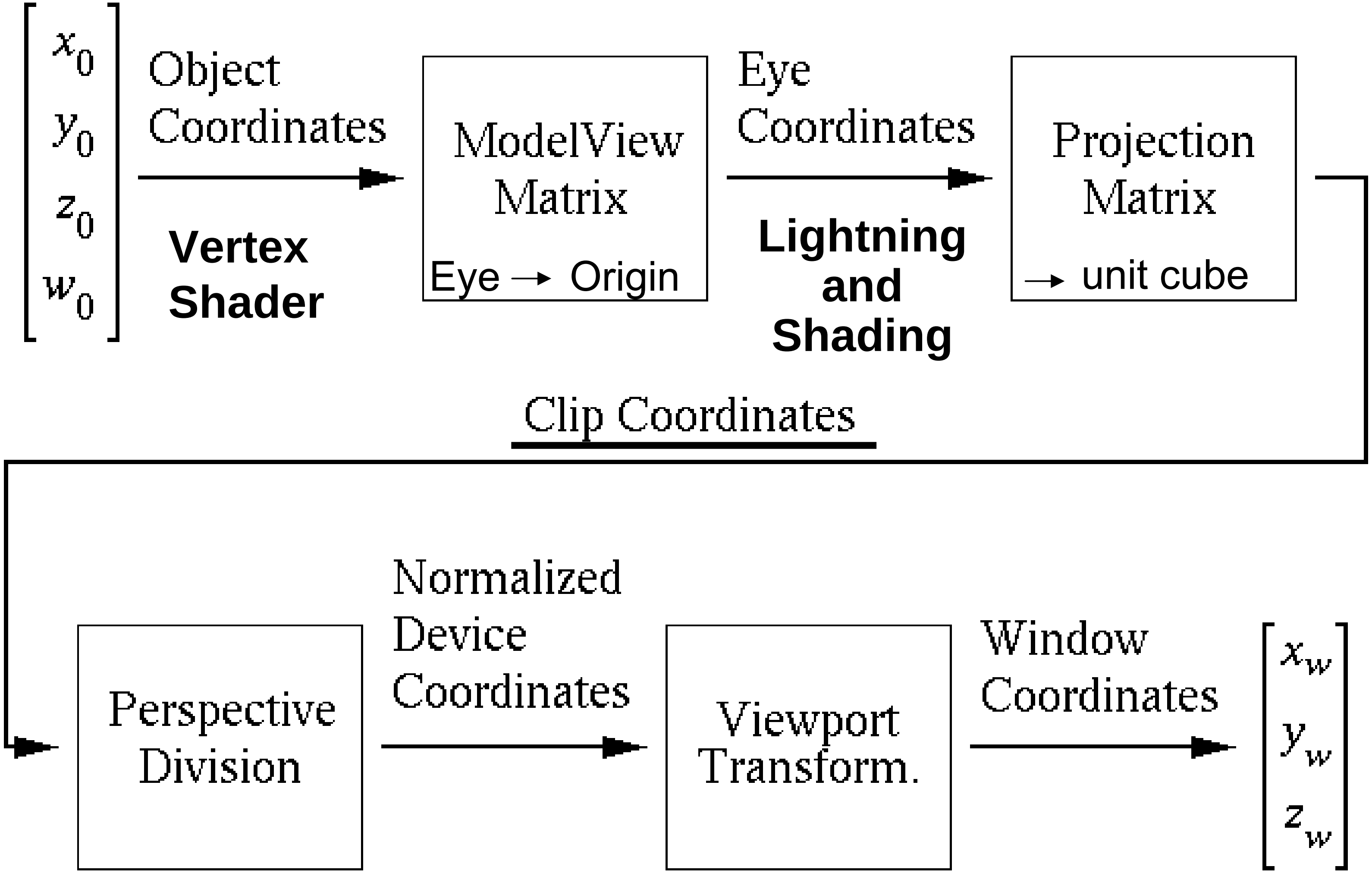


# Viewing Frustum




$$\text{Aspect Ratio} = \frac{y}{x} = \frac{\tan(\text{vertical FOV}/2)}{\tan(\text{horizontal FOV}/2)}$$

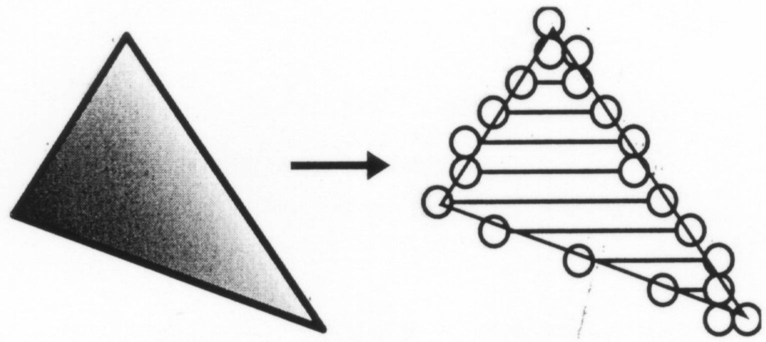

# The Geometry Stage (Simple)

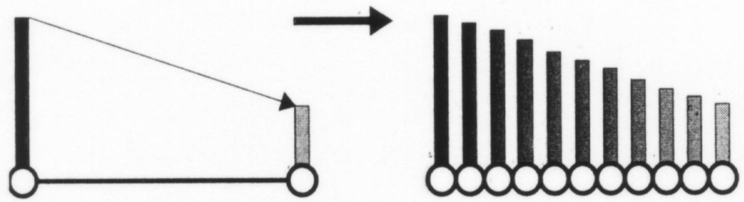



Kaufmann Reality

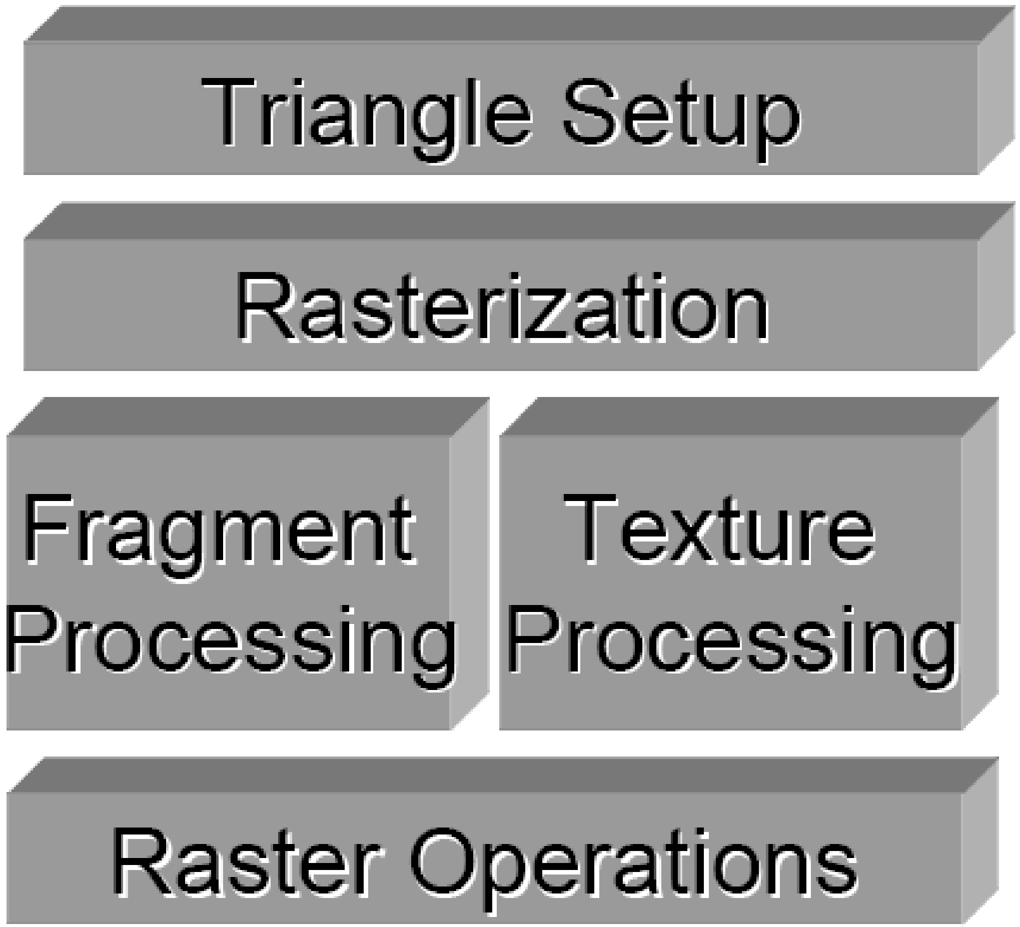
# Rasterization Stage (1)

- Input: 2D Geometric Primitives (Points, Lines, Polys, Bitmaps)
- Primitives needed!
- 1st step output: Fragments (Pixel-Coord. + Color + Depth + Texture-Coord.)
- Polygons are decomposed (various methods)


Polygon decomposition to spans


Span iteration (interpolating depth and color)

# Rasterization Stage



# Rasterization Stage (2)

- Per-Fragment Operations
- Pixel Ownership Test (Window visible?)

Buffers:

- Depth Buffer Test (z-Buffer)
- Stencil Buffer
- Accumulation Buffer
- Color Buffer + Alpha channel

# Rasterizer/Display Stage

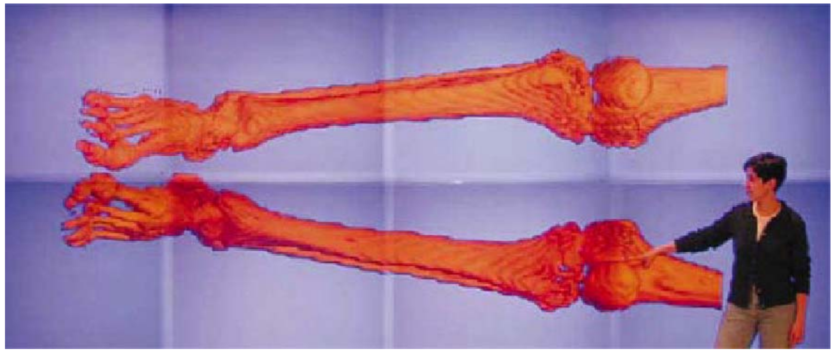
- Framebuffer pixel format: RGBA vs. indexed (colormap)
- Bits: 32, 24 (true color) 16, 15 (high color), 8
- Double buffering, Triple Buffering
- For Stereo: Quad buffer (Left/right x 2)
- Overlay planes (extra bitplanes)
- Per-window video mode (e.g. stereo, mono)
- **Display**: frame buffer -> screen

# Bottlenecks

- Application limited: App cannot create drawing primitives fast enough
- Interface limited: Not enough bandwidth CPU->graphics card (e.g., AGP)
- Geometry limited: Not enough vertex transform capacity (Test: remove lights)
- Pixel fill limited: Not enough pixel fill capacity (Test: decrease resolution)
- Memory limited: Not enough (texture) memory bandwidth

# Large Scale Visualizations – Some Examples

Mechanical visualization
CAVE, SGI Onyx (8 CPUs, 6 pipes)



Princeton Display Wall
3x8 projectors, 24 PC cluster

# Parallel Graphics

Overcome bottleneck by parallel computation

Types of parallel graphics:

1. On-chip / on a graphics board
2. Multiple boards (former: graphics supercomputer)
3. PC cluster with standard network
4. PC cluster with special hardware

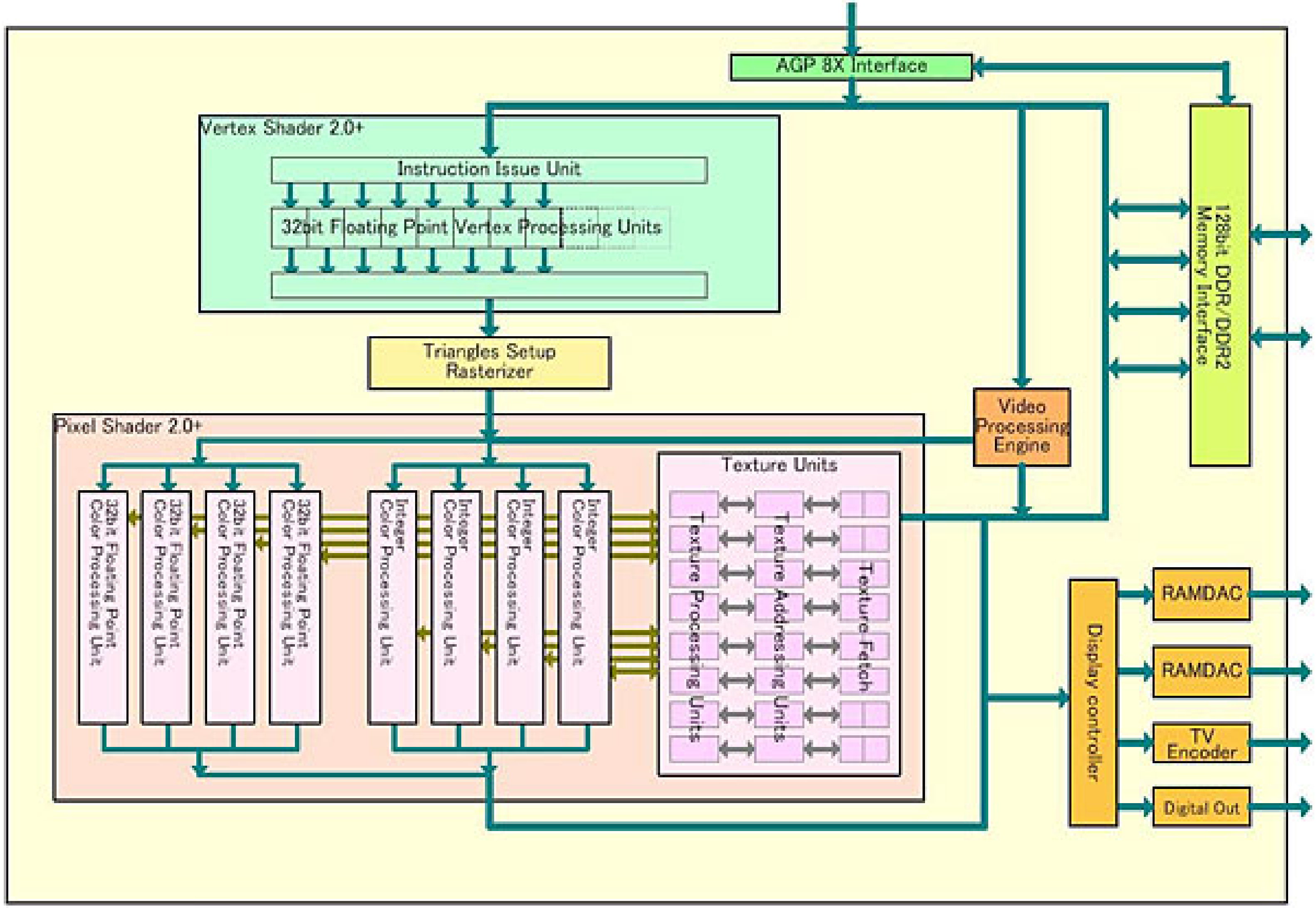
# Multiple Graphics Pipelines


- Pipelines fully realized in HW
- Multiple independant pipelines can be parallelized
- Modern GPUs process up to 24 Pixel-Pipelines (Rasterizer) and 8 Vertex-Pipelines (Geometry) in parallel

Hannes Kaufmann
Virtual Reality

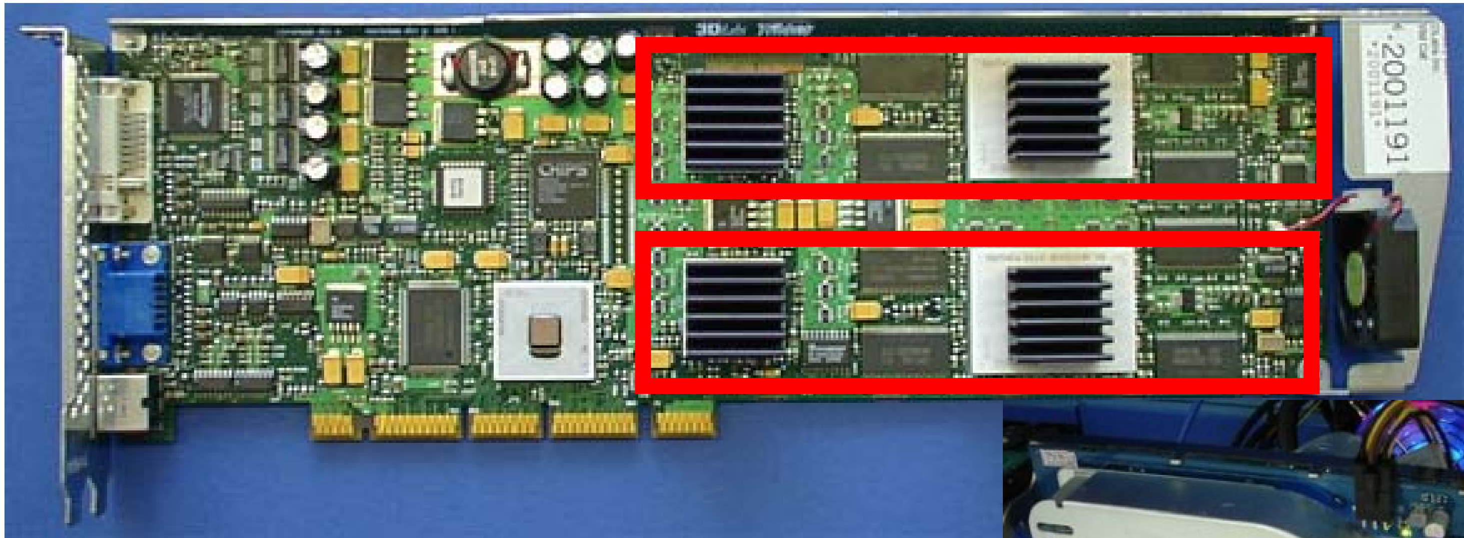
# Parallel On-Chip

NVidia NV30



# Parallel On-Board

- Examples:
- 3DLabs Wildcat III: 2 Pipes



- ASUS Dual GeForce 7800GT (Nov. 05)

# Parallel Graphics

Types of parallel graphics:

1. On-chip / on a graphics board
2. **Multiple boards (former: graphics supercomputer)**
3. PC cluster with standard network
4. PC cluster with special hardware

# Multiple graphics boards

Parallel graphics:

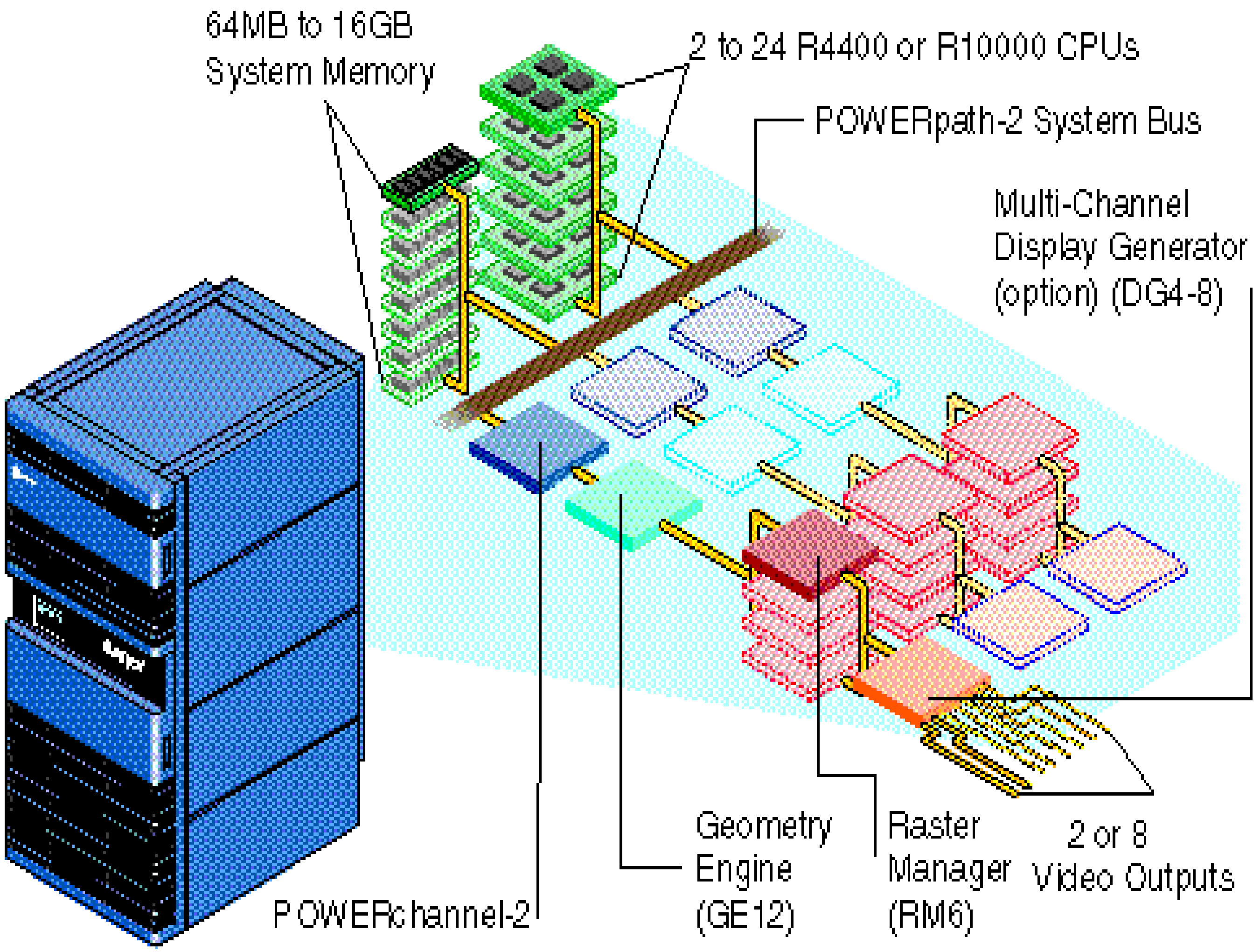
- Graphics Supercomputer
- PC with SLI or CrossFire

Different (!):

Multiple display support (not synchronized):

- PC with multiple unconnected cards (via PCIe or APG+PCI)

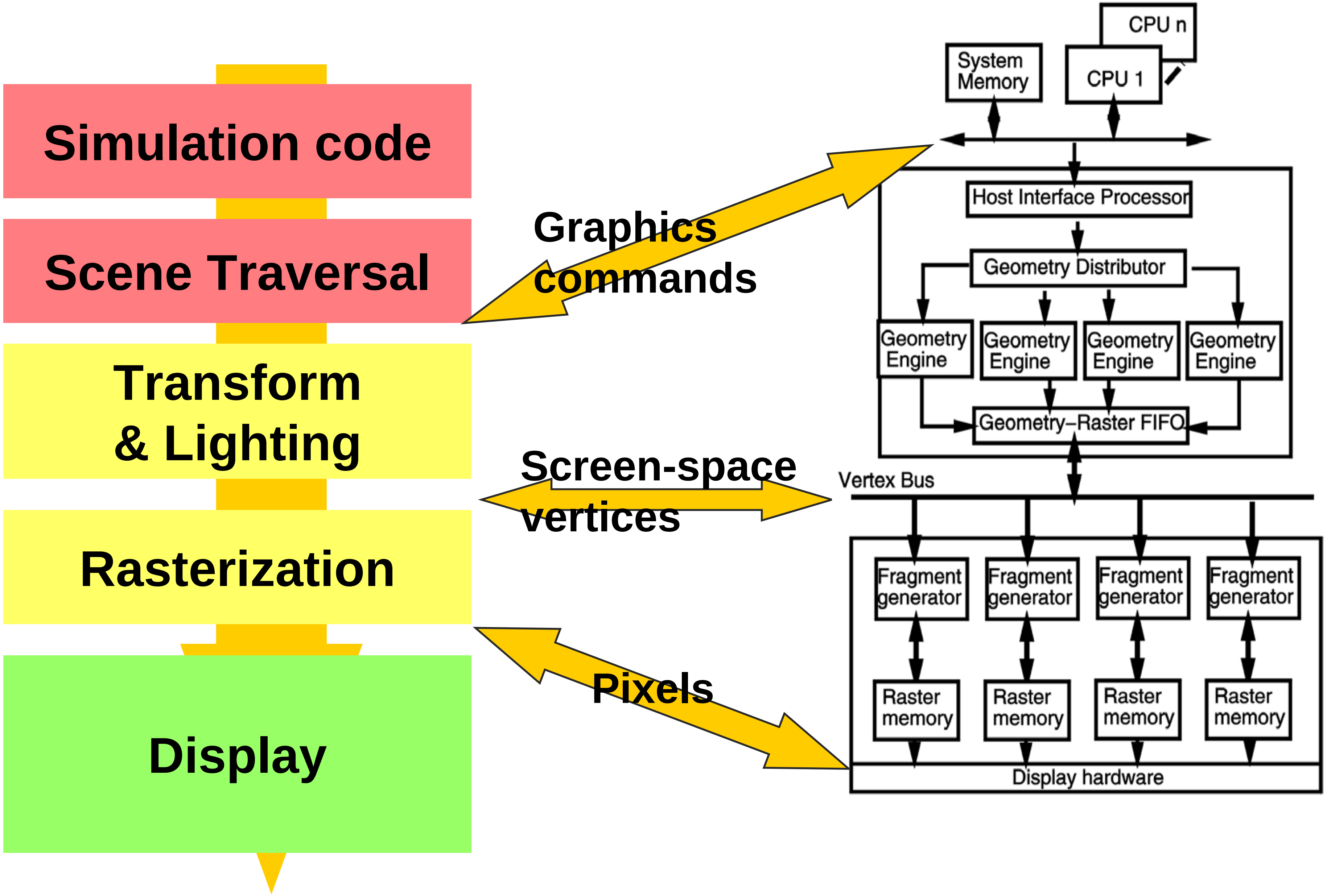
# Graphics Supercomputer (1)




SGI Onyx with
Infinite Reality 3

Hannes Kaufmann
Virtual Reality

# Graphics Supercomputer (2)



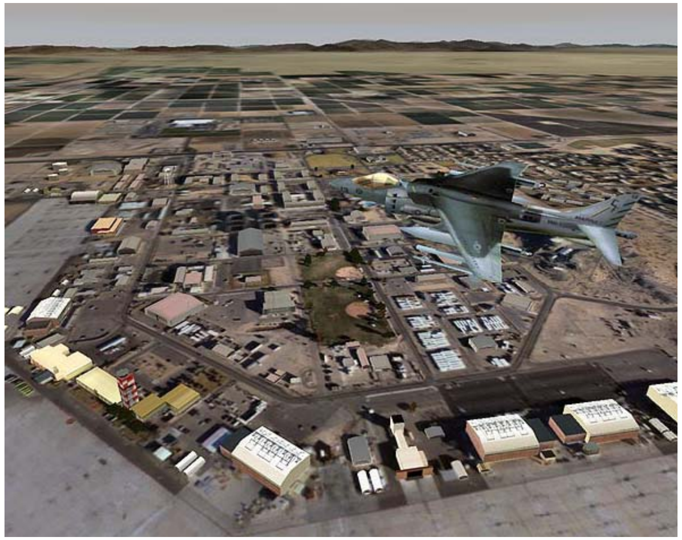
# SGI Onyx 3000 & Infinite Reality 4

G-Brick:

- 4 RasterManager Boards
- 1.3 Gpixel/s/Pipeline
  - (8 subsample/full scene/AA)
- 1 GB Texturspeicher
- 10 GB Framebuffer
- 192 GB/s Bandbreite
- Kombination bis zu 16 IR4

# Supercomputer – Application Areas

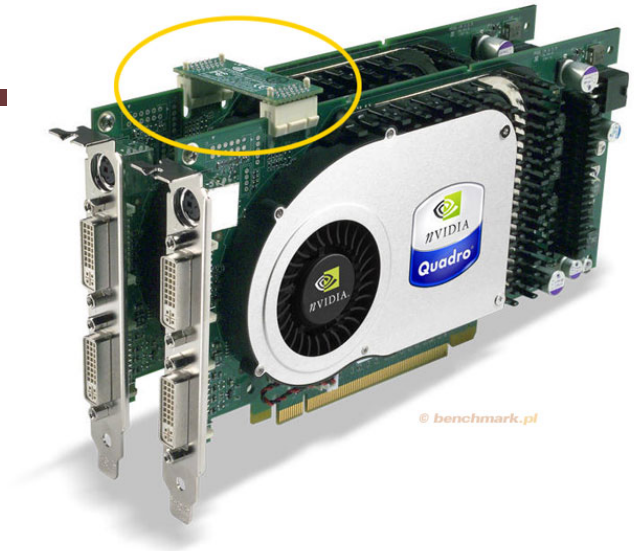
- Theme Parks (DisneyQuest – CyperSpace Mountain)
- Flight Simulators
- Military Applications
- CAVEs / Large setups



# Graphics Supercomputer (3)

- Multiple CPUs
- Multiple Geometry Engines
- Multiple Rasterization Engines
- Genlocking
- Multiple Pipes (=graphics cards)
- Multiple Channels (=display outputs)
- Highly configurable
- Now used: standard Nvidia/ATI graphics chips
- Now on PC: Scalable Link Interface (Nvidia) or CrossFire (ATI) for PCI Express

# SLI™ (Nvidia)




- originally Scan Line Interleave (3Dfx Vodoo 2) – odd/even lines
- Nvidia calls it Scalable Link Interface

2 Modes:

- Split Frame Rendering (SFR) - Scissors: Splits each frame and sends half the load to each of the graphics cards
- Alternate Frame Rendering (AFR):
  Frame 1 – Card 1, Frame 2 – Card 2, alternating
  problems in some games because of blending over frames
- New modes coming

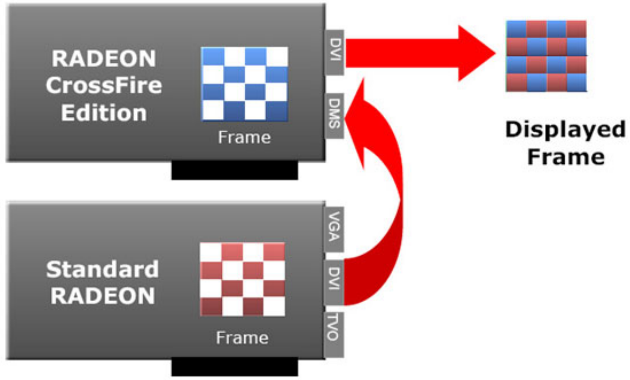
- Optimal performanc increase: 1,8 max.

# CrossFire (ATI)

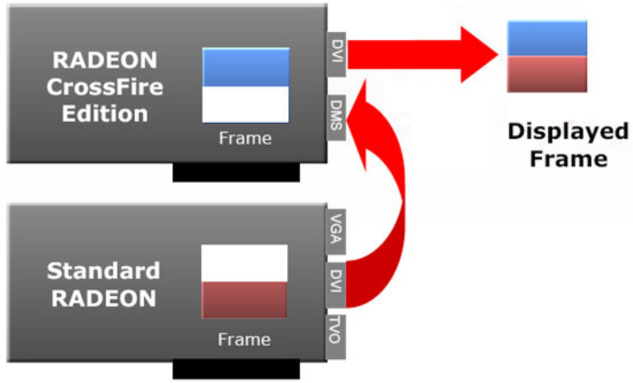
3 Modes:

- Supertiling
- Scissors
- Alternate Frame Rendering

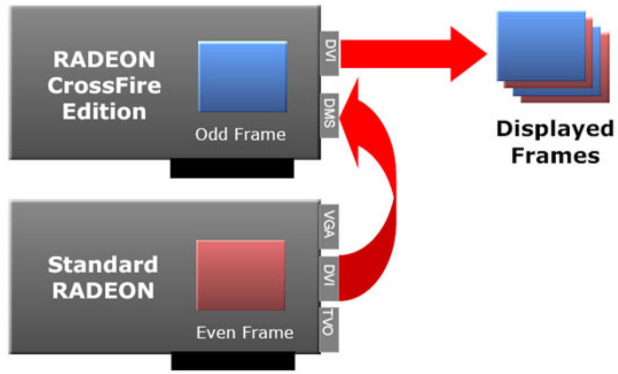
Additional AA Mode


*Supertile Mode*




*Scissor Mode*




*Alternate Frame Rendering (AFR) Mode*

# SLI / CrossFire

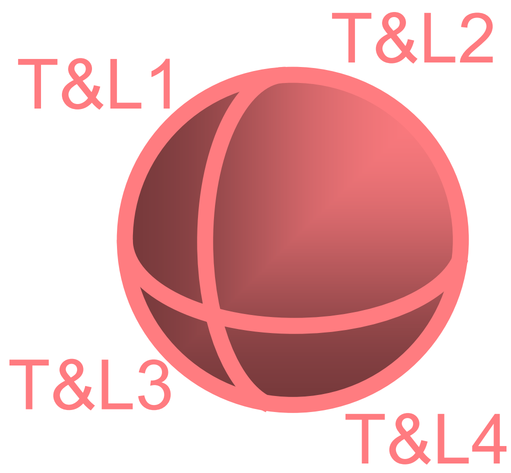
- Mainboards with SLI or CrossFire support needed
- Master/slave setup
- SLI: Connection via separate bridge (soon bridgeless – PCIe communication)
- CrossFire: Inter-GPU connector, compositing on Master board
- CrossFire SuperTiling efficient
- Only 2 cards can be connected yet
- Stereo SLI Modes: Alternate Left/Right (?)

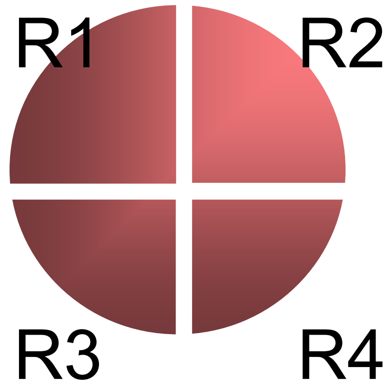
# Vertex and Pixel Load Balancing

- Problem with parallel rendering
  - Load balancing of vertices
    → 3D (object space) problem
  - Load balanding of pixel (rasterizers)
    → 2D (screen space) problem

# Parallel Rendering as Sorting (1)

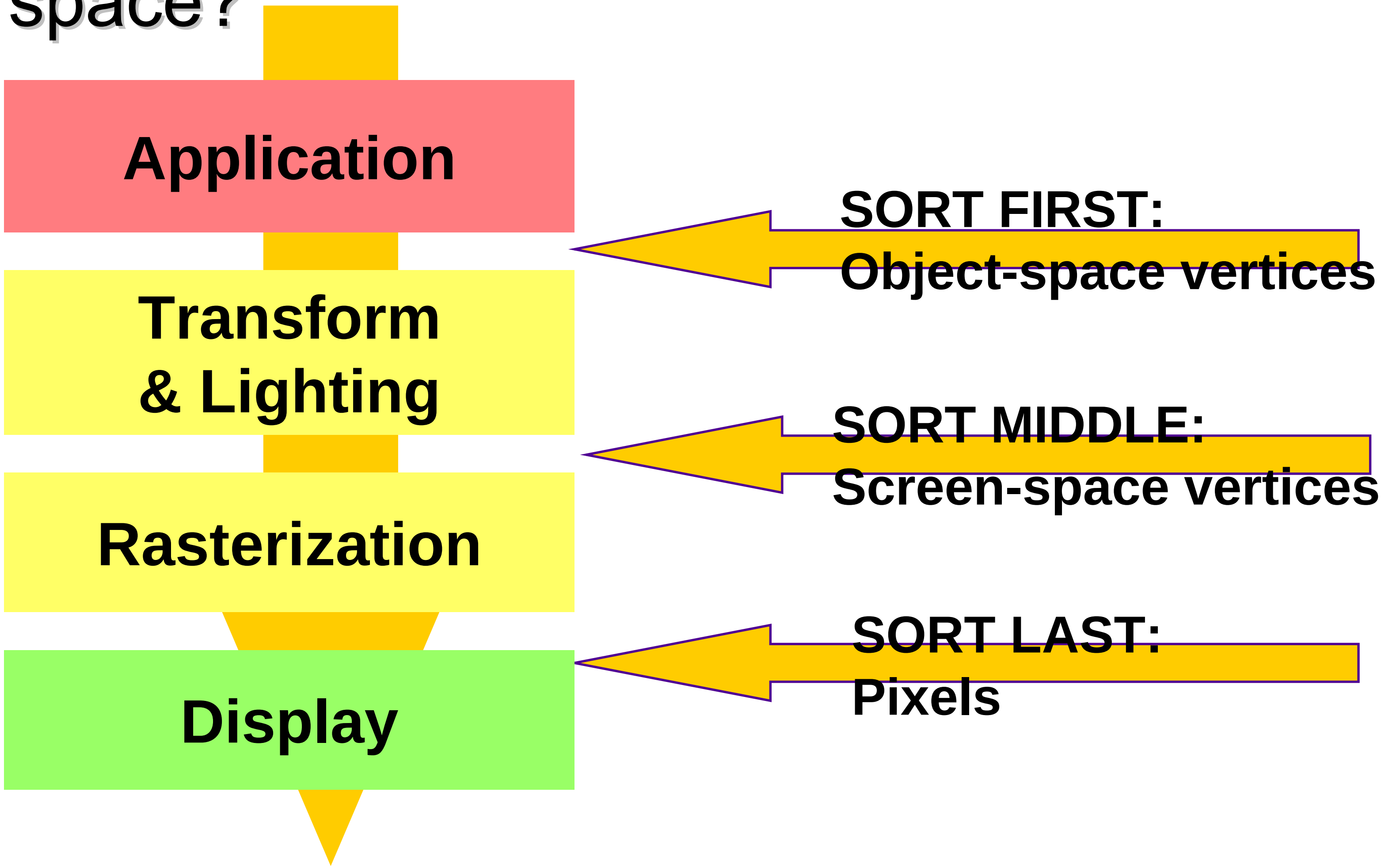
- Parallel T&L
  - Cut 3D model into pieces with equal number of vertices
  - Assign one piece to one T&L unit
- Parallel rasterization
  - Cut destination image into tiles
  - Assign (triangles contained in) one tile to one rasterizer
- → Need to SORT transformed 2D triangles
- Shared common memory



# Parallel Rendering as Sorting (2)

Where to sort from object-space to screen-space?



# Parallel Graphics

Types of parallel graphics:

1. On-chip / on a graphics board
2. Multiple boards (former: graphics supercomputer)
3. **PC cluster with standard network**
4. PC cluster with special hardware

# Parallel Cluster Rendering (1)

- PC Cluster
  - Off-the-shelf hardware
  - Network (LAN)
  - Cheap
  - Scalable

# Parallel Cluster Rendering (2)

- power of cluster ≥ power of supercomputer
- Price of cluster << price of supercomputer
- BUT: problems of cluster
  - How to make cluster PCs work together
  - On a single image
    (or consistent set of images)
- → Parallel Execution of Rendering
- → Cluster synchronisation (genlocking)

# Cluster Synchronisation



Q: How to synchronize multiple displays?

(1) Simple: PC + Multiheaded Graphics

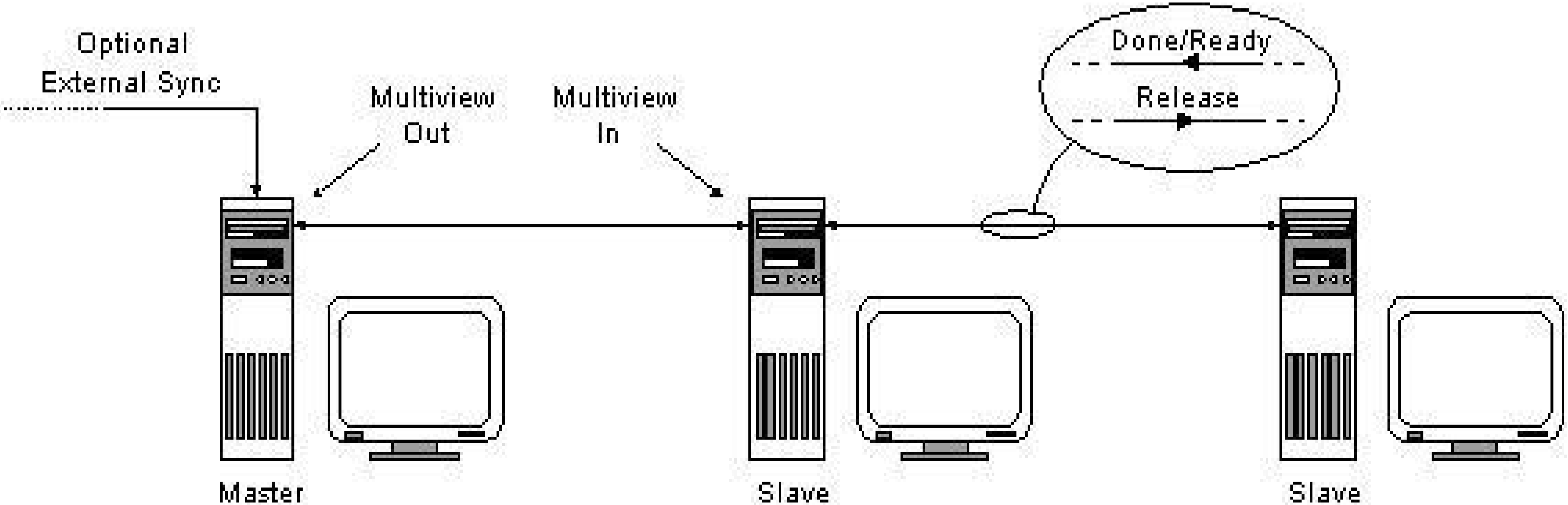
(2) Not so simple: Multiple workstations

# Parallel Graphics

Types of parallel graphics:

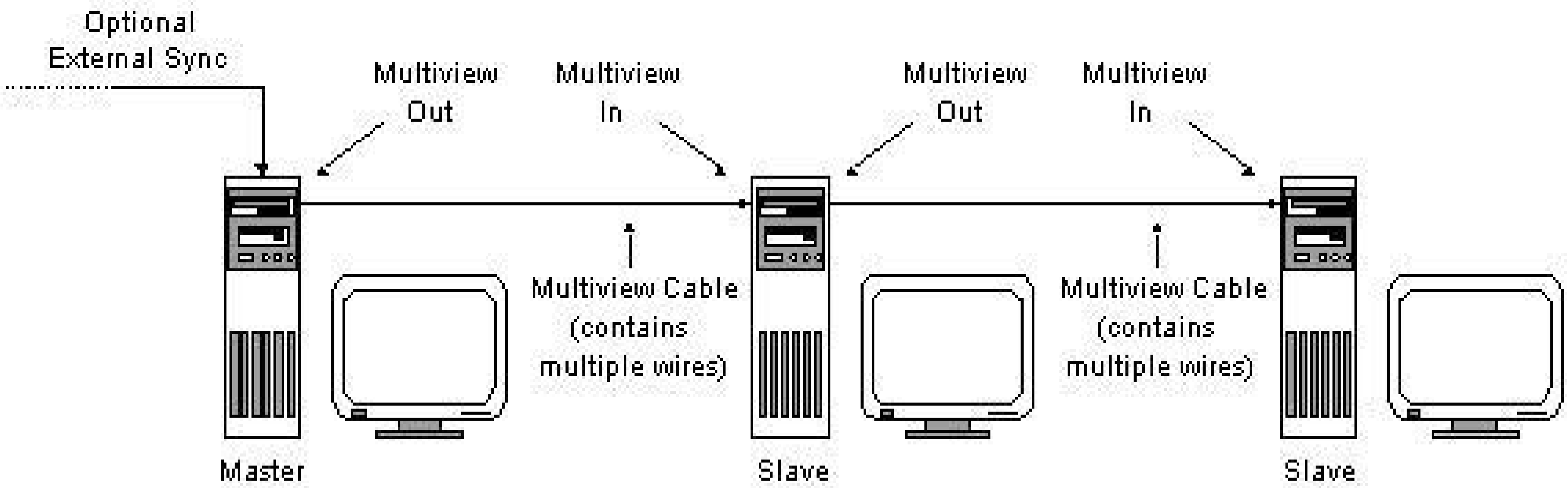
1. On-chip / on a graphics board
2. Multiple boards (former: graphics supercomputer)
3. PC cluster with standard network
4. **PC cluster with special hardware**

# Framelock




Mechanism for synchronizing double buffer swap

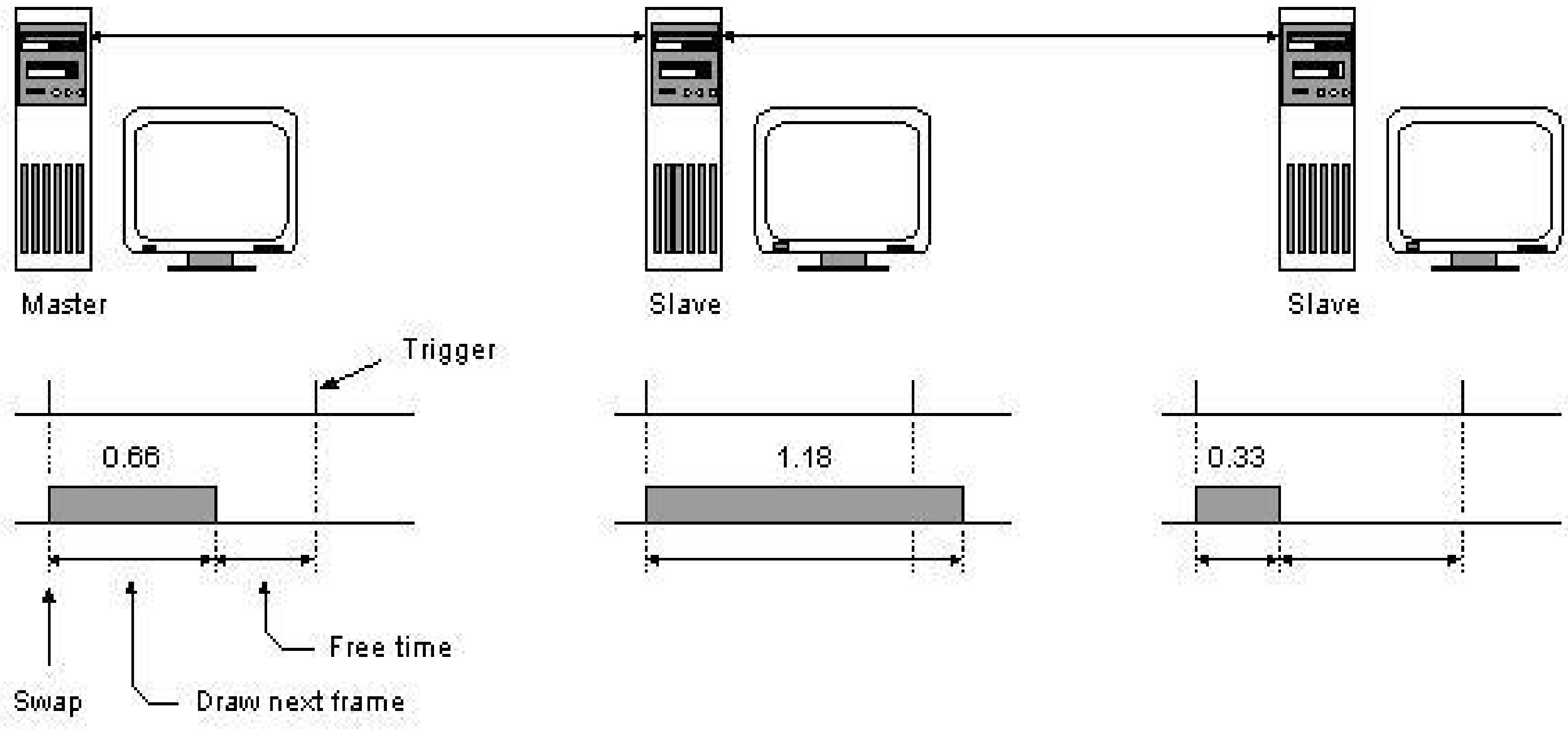
# Genlock




Mechanism for synchronizing video generation by external signal [=hardware]

# Example: Blue-C

# Ratelock




Mechanism to ensure equal rate of multiple applications‘ drawings [=software]

# 3D Card High End Model

- nVidia Quadro FX 4500 (~ € 1700.- ), 512MB
- Based on G70 chip (such as GeForce 7800)
- 24 Pixel Pipelines, 8 Vertex Pipelines
- 33.6 GB/s Bandwidth
- Fillrate: 10.32 billion pix/s.
- Vertices: 860 M/s
- optimized OpenGL drivers (comp. to consumer card)

# Some Relevant Features (for VR)

- Memory size: 512 MB
- Dual DVI / VGA Output (up to 3840x2400)
- Multi-Display support
- OpenGL quad-buffered stereo (3-pin sync connector)
- Scalable Link Interface (SLI™) Technology
- Edge Blending for Powerwalls (supported by driver)
- G-Sync Option Board with Framelock and Genlock
- Quality: Full-Scene Antialiasing (FSAA); OpenGL 2.0; Shader Model 3.0,…

# Nächster VO Termin

9. November ´05

(14:15-15:45)