Equalizer

Get the best performance from your visualization cluster

Outline

- High-performance visualization
  - Transparent and semi-transparent solutions
  - Programming interfaces
- Equalizer
  - Components
  - Features

What is HPV ?

- High-Performance Visualization - like HPC but for interactive 3D applications
- Address the demand to visualize huge data sets using COTS clusters
- Issue is to *scale* rendering performance using multiple GPU's and CPU's

HPC Analogy

	HPC	HPV
What?	Parallelize computation across multiple CPU's	Parallelize 3D rendering across multiple GPU's and CPU's
How?	Mostly non-interactive batch processing	Highly interactive, real-time rendering
Hardware	Cluster or Supercomputers typically using fast interconnects	Graphic Cluster, Supercomputers, display hardware, input devices

HPV Today

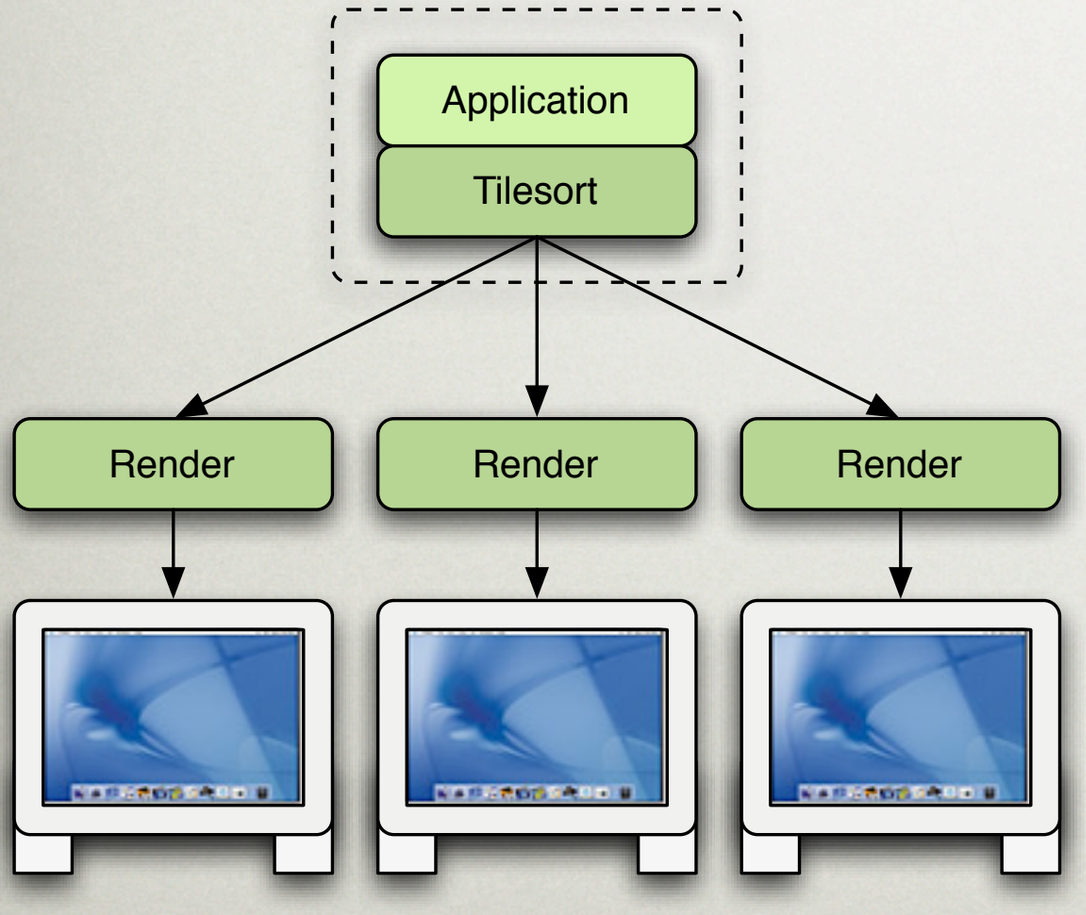
- Transparent and semi-transparent solutions
- Programming interfaces
- Equalizer

HPV Transparent Solutions

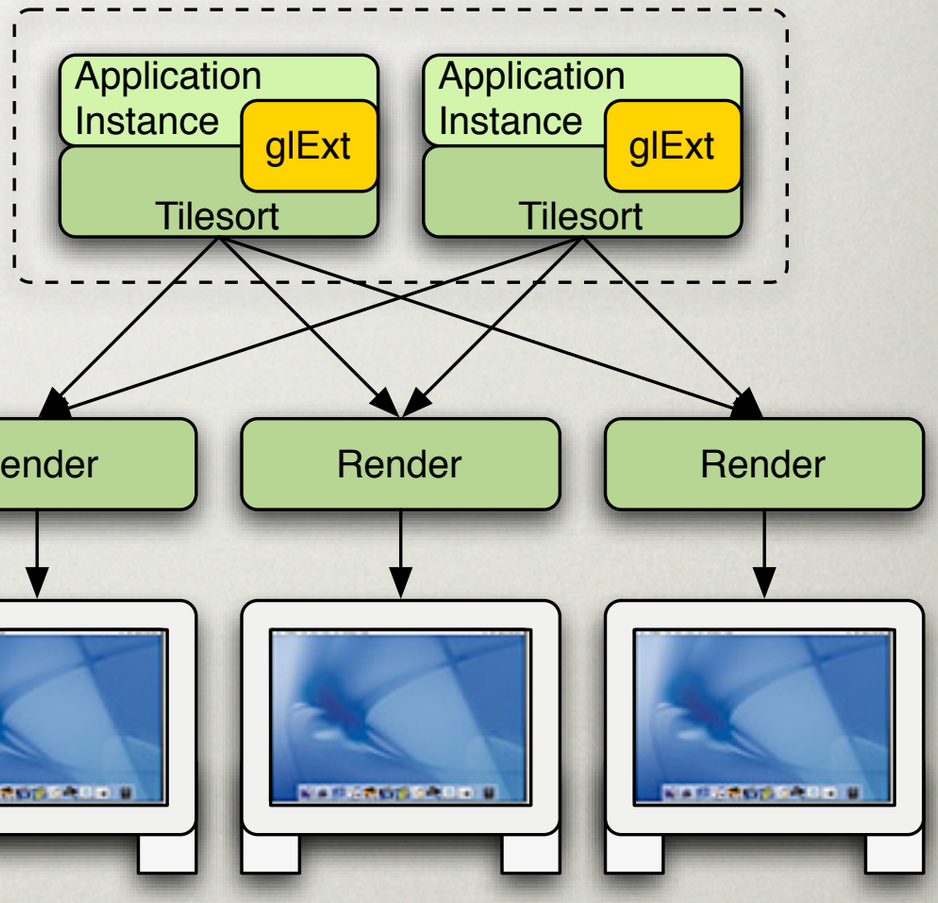
- Chromium, ModViz VGP, OMP
  - Operate on OpenGL command stream (HPC analogy: auto-parallelizing compilers)
  - Provide programming extensions for improved performance and scalability (semi-transparent)

HPV Transparent Solutions

Transparent



Semi-Transparent



HPV Programming Interfaces

- OpenSceneGraph, Vega Prime, VTK
  - Impose overall programming model and data structure (HPC analogy: CFD codes)
  - Best for developing new applications
- Cavelib, NetJuggler, MPK
  - Limited to HPV-critical areas of the code (HPC analogy: MPI, PVM)
  - Best for porting existing applications

Equalizer Components

A Programming Interface

and

Resource Management System

for

Scalable Graphics Applications

Equalizer Programming Interface

Applications are written against a *client library* which abstracts the interface to the execution environment

- Minimally invasive programming approach
- Abstracts multi-processing, synchronisation and data transport
- Supports distributed rendering and performs frame compositing

Equalizer

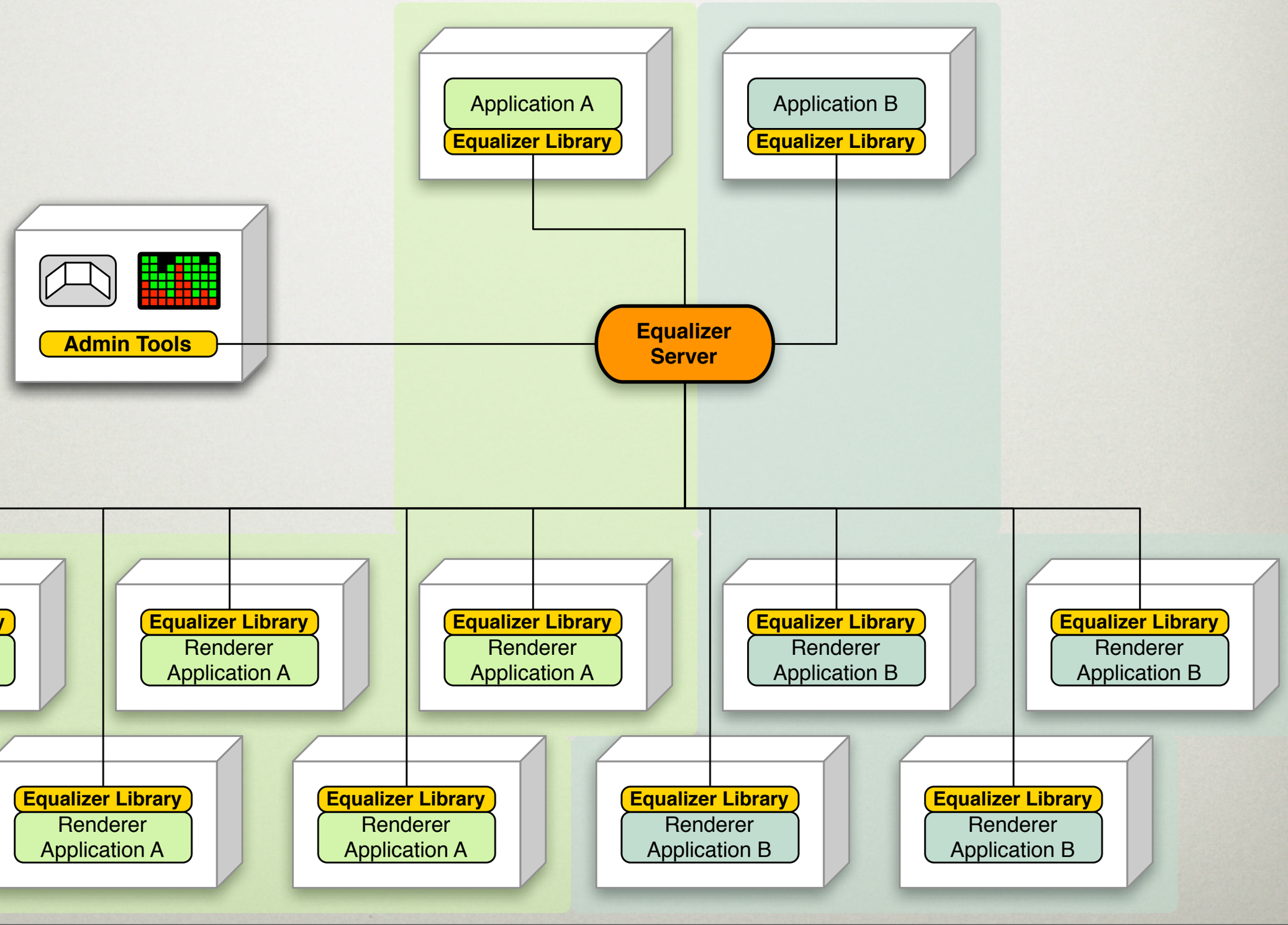
Applications are deployed by a *server* which balances the resource usage across the system

- Centralizes the setup for all applications
- Deploys application render clients
- Dynamic load-balancing of the cluster resources
- ASCII configuration files

Equalizer Tools

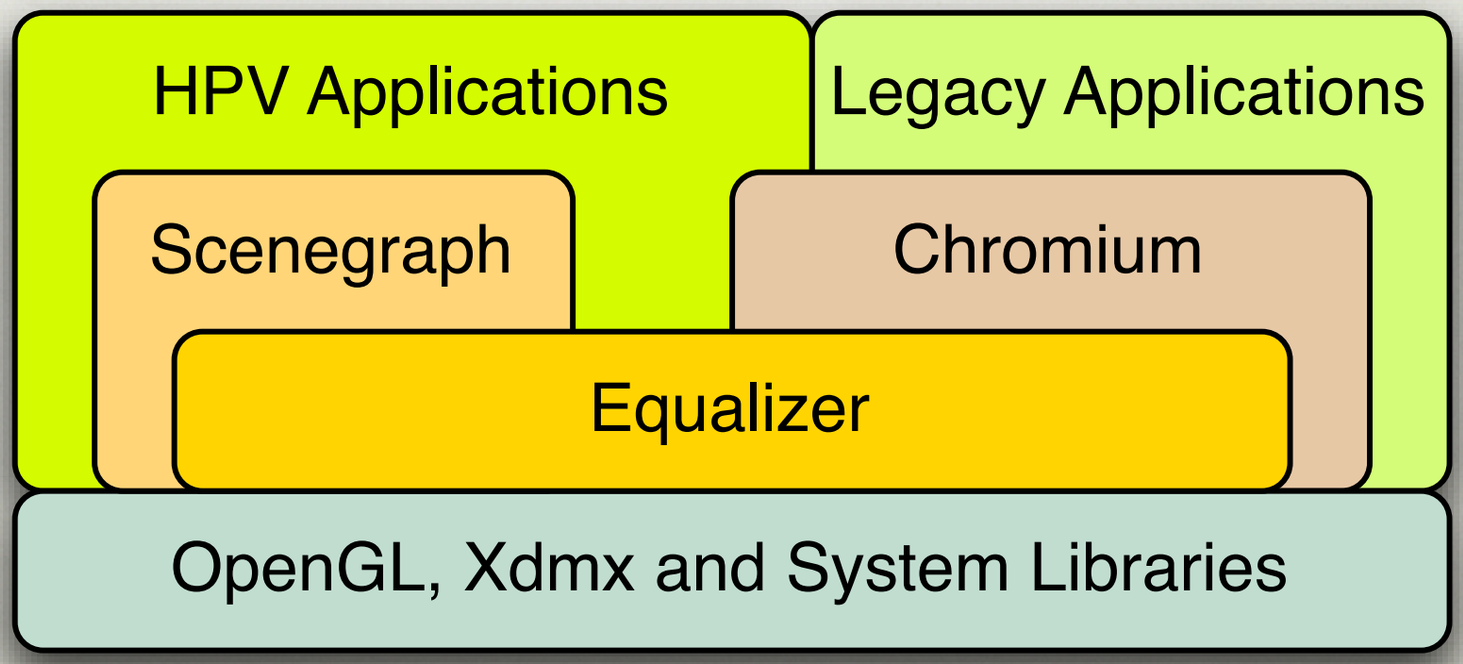
- Configuration
  - Dynamic reconfiguration of the server
- Monitoring
  - Resource usage, bandwidth utilization, statistics
- Profiling
  - Execution flow, data distribution

Equalizer Components



Equalizer Solution

- Xdmx: X11 single virtual screen
- Chromium: OpenGL™ single virtual screen
- Equalizer: Scalable rendering engine



Equalizer Features

- Runtime configuration
- Runtime scalability
- Asynchronous execution
- Remote visualization
- Clusters and SSI
- Open Source

Runtime Configuration

- Hierarchical resource description: Node→Pipe→Window→Channel
  - Node: single system of the cluster
  - Pipe: graphic card
  - Window: drawable and context
  - Channel: view
- Resource usage: compound tree

Runtime Scalability

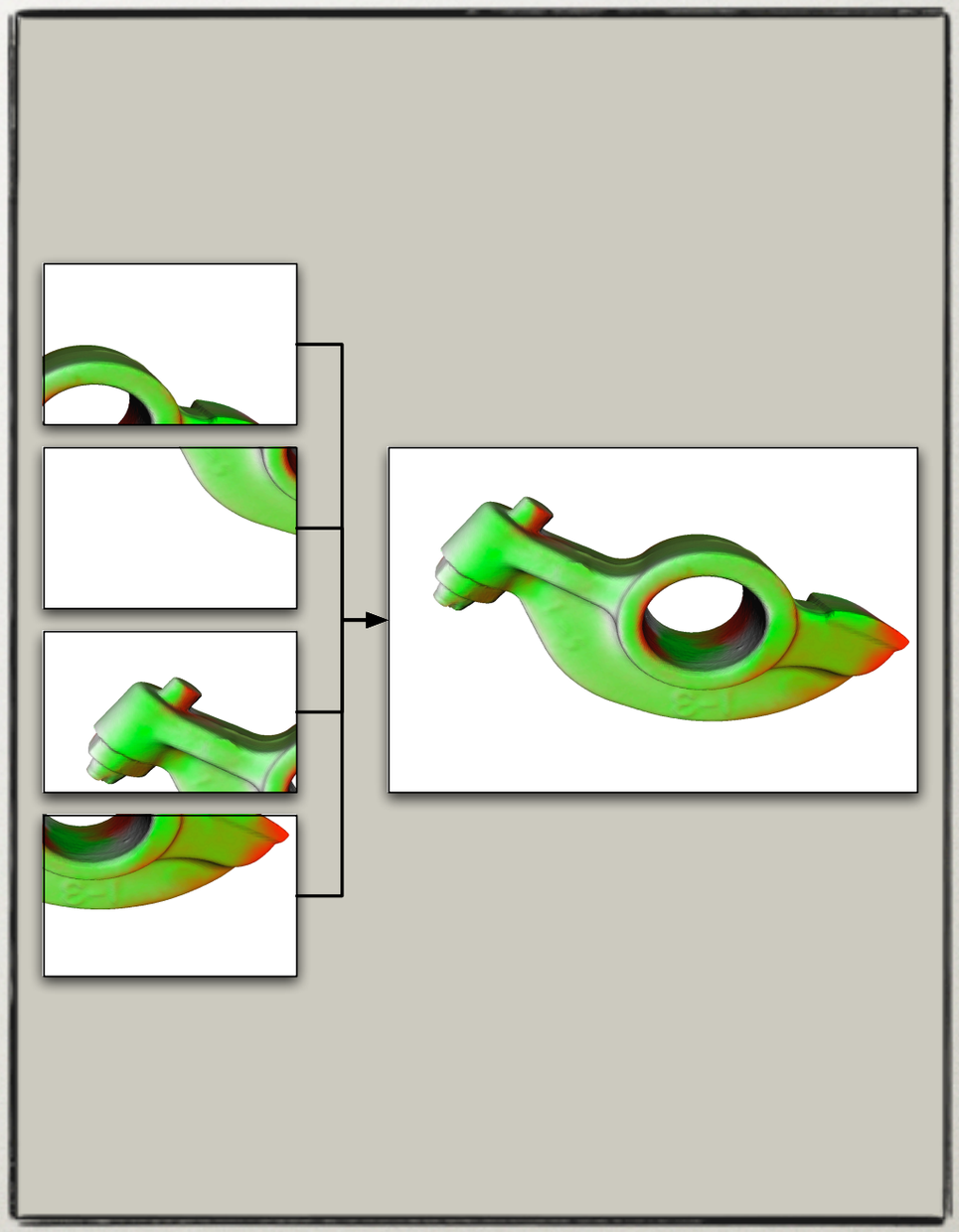
- Parallel execution of the application's rendering code
- One thread per graphic card
- Decomposition of rendering for one view
- Server chooses and adapts configurations based on system resources, topology and load

Runtime Scalability

- Modes: 2D, DB, DPlex, Eye
- Automatic loadbalancing
- Supports compositing hardware
- Hardware optimizations

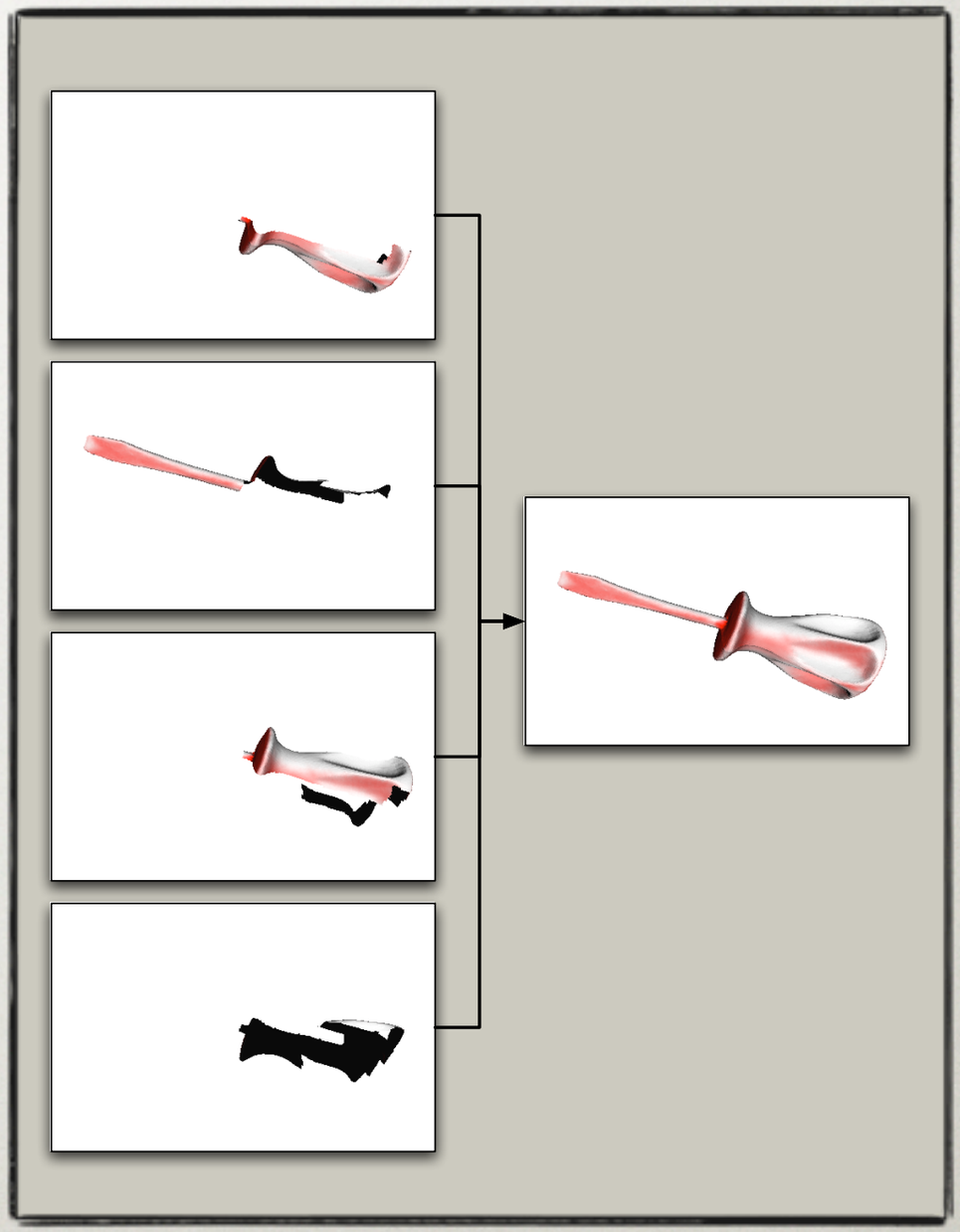
2D Compound

- Sort-first
- Scales fillrate
- Scales geometry when used with view frustum culling



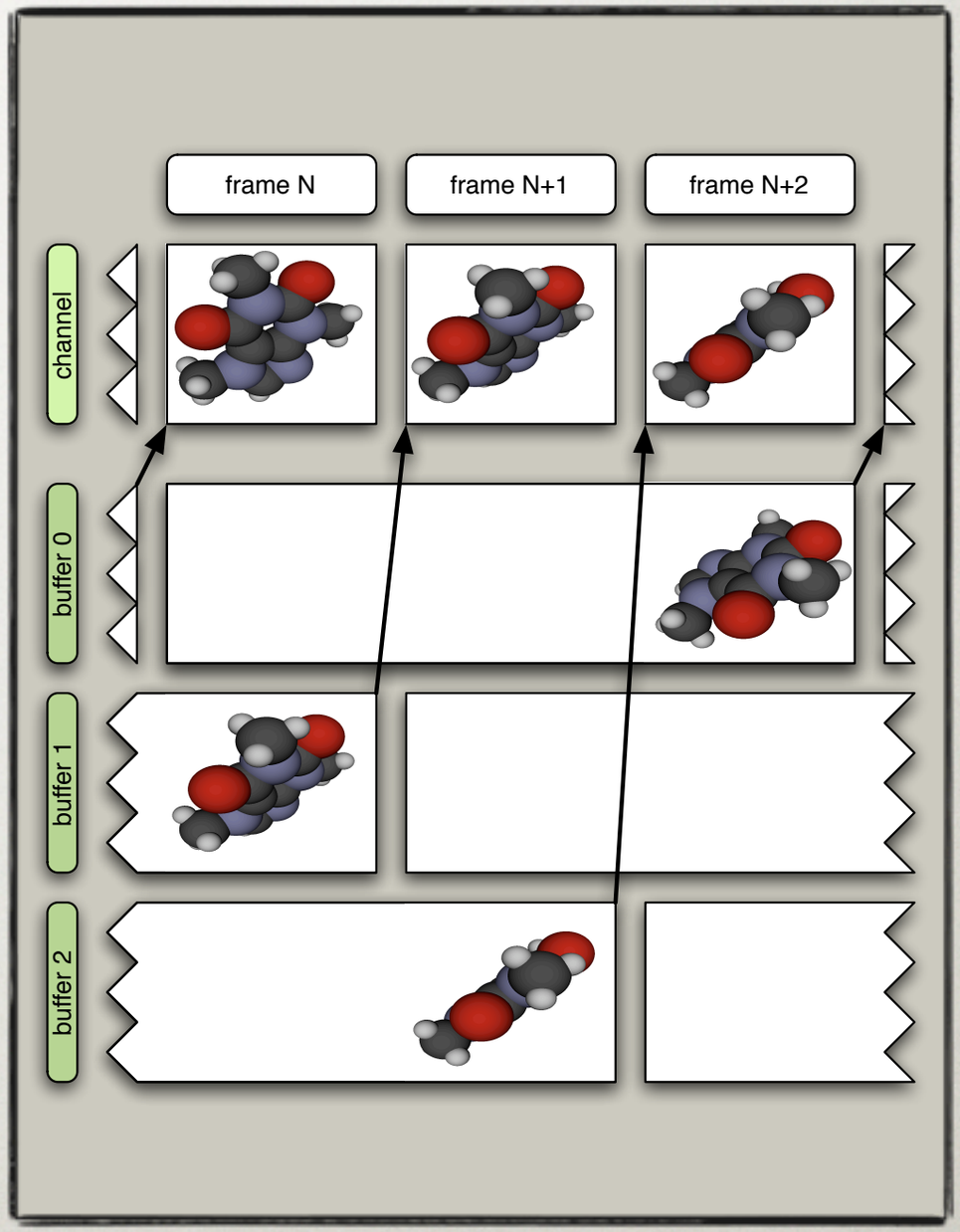
DB Compound

- Sort-last
- Good load distribution
- Application needs to be adapted to render subrange of data



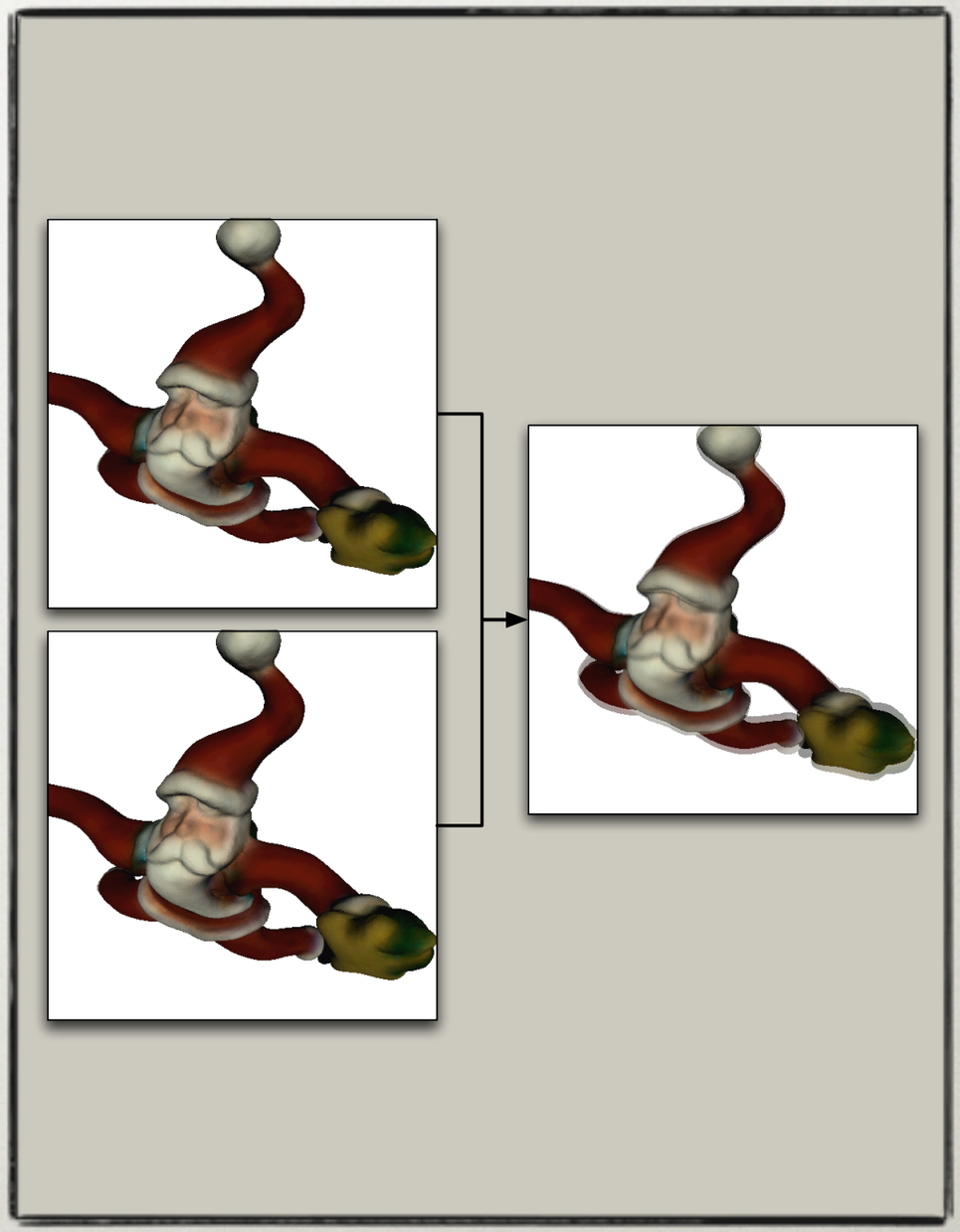
DPlex Compound

- Time-Multiplex
- Good scalability and loadbalancing
- Increased latency may be an issue



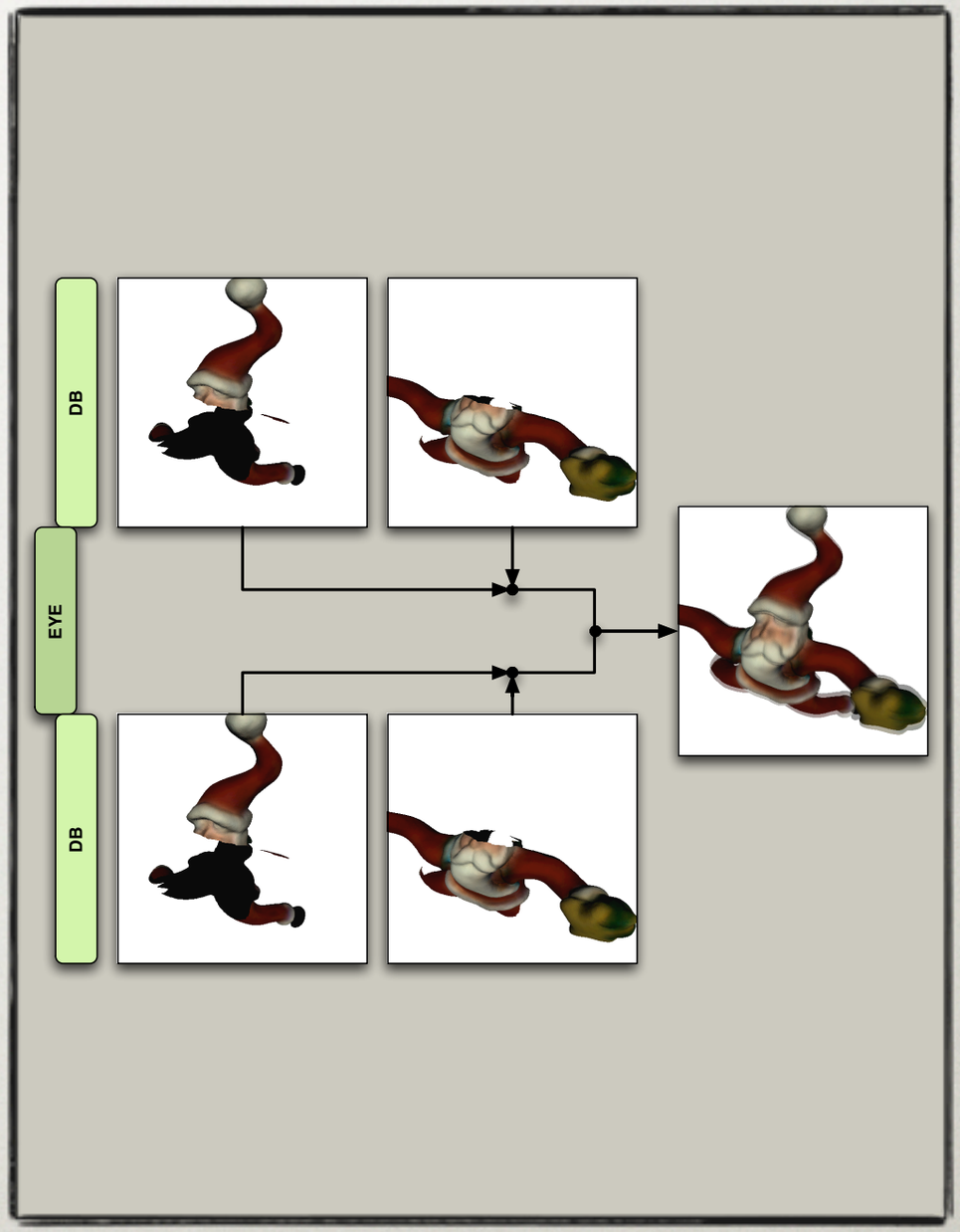
Eye Compound

- Stereo rendering
- Very good loadbalancing
- Limited by number of eye views



Multilevel Compounds

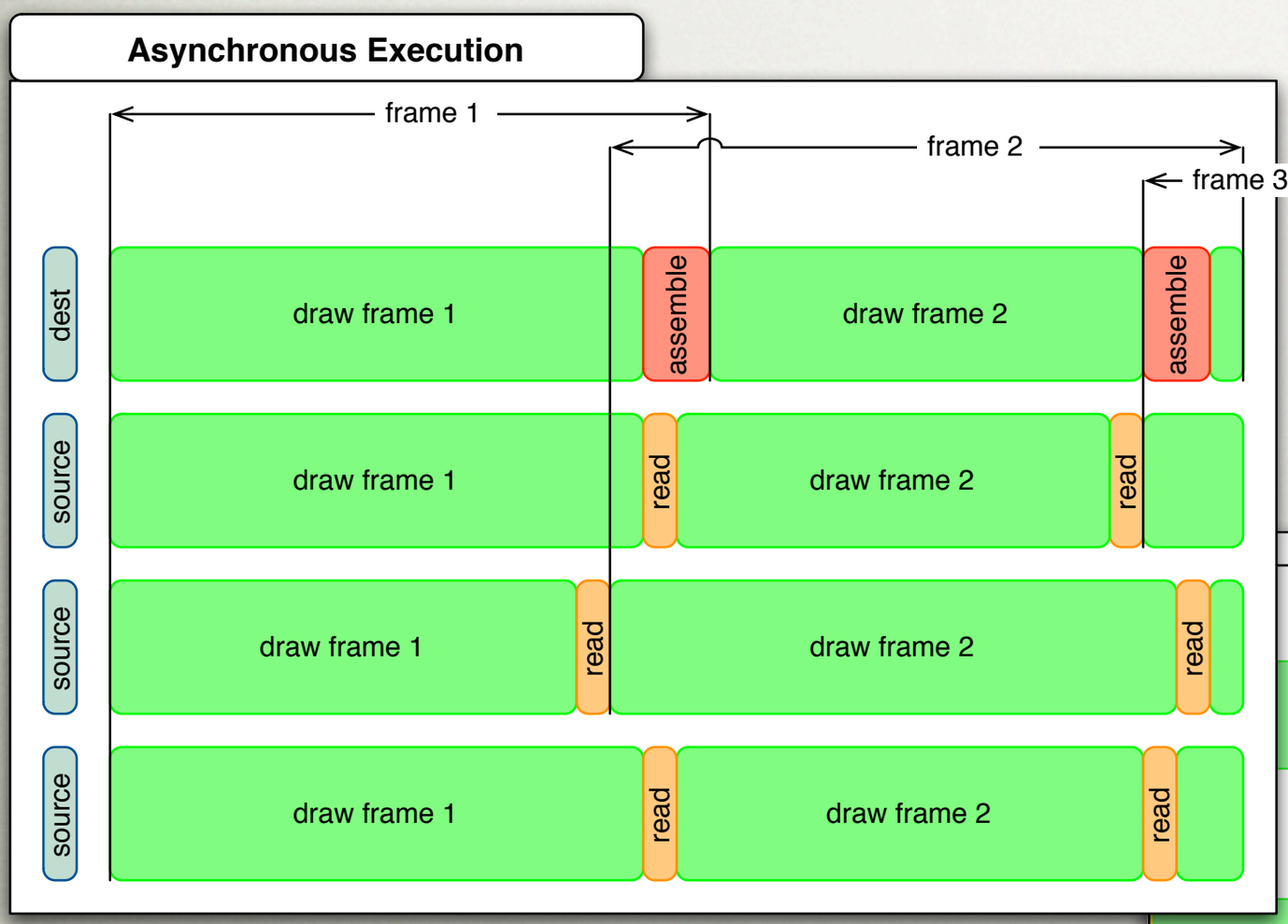
- Compounds allow any combination of Modes
- Combine different modes to address different bottlenecks

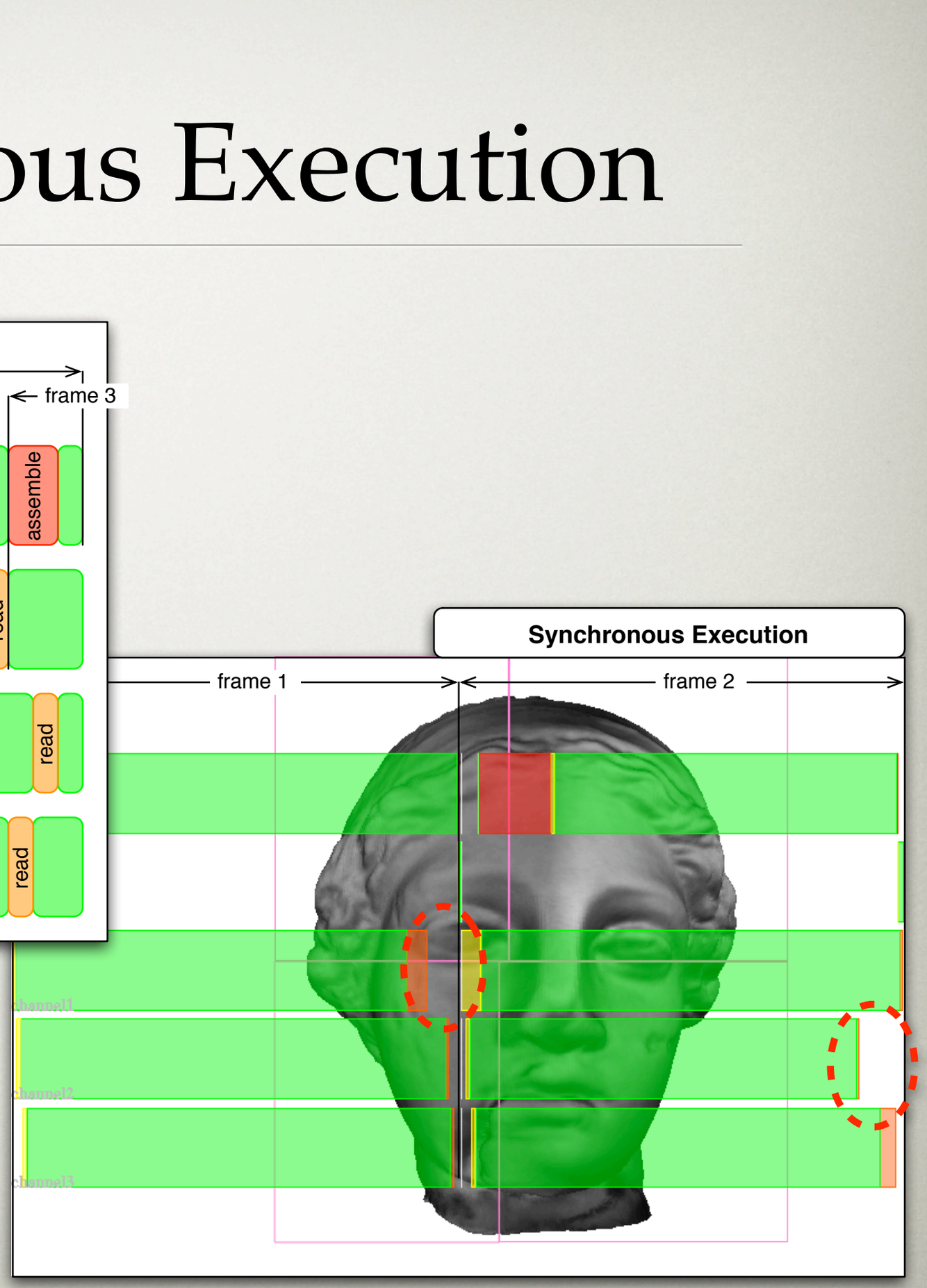


Asynchronous Execution

- Source channels start rendering the next frame early
- Hides imbalance in load distribution
- Only visible channels belonging to the same view are synchronized
- Greatly improves scalability on bigger pipe counts

Asynchronous Execution





Remote Visualization

- Leverages knowledge of the application
  - Frames are often available in main memory
  - Additional frame-transport optimizations
  - Remote workstation is ‘destination channel’
- Loadbalancing of multiple applications on one visualization cluster

SSI and Clusters

- Supercomputers are just tightly integrated clusters
- Equalizer runs on both architectures
- Overall model is the same
- HPC uses the same approach
  - MPI is available on all supercomputers

Open Source

- LGPL license
- Open standard for scalable graphics
- User-driven development
- Alpha version on www.equalizergraphics.com