

Equalizer

Quickstart and Demonstration Guide

Building Equalizer

- Install a binary version **or** build the subversion source tree:
- Linux, Mac OS X:
 - `cd src; make`
 - set library path as printed by make
- Windows:
 - Build Solution `src/vs2005/Equalizer.sln`

Running the Server

- Linux / Mac OS X:

```
cd src/build/[Linux|Darwin]/bin  
./eqServer [config]
```
- Windows:
 - run: Debug 'Equalizer Server' project
 - **OR:** `build\vs2005\win32\debug\eqServer`

Running the Example Application

- Linux:

`src/build/Linux/bin# ./eqPly`

- Mac OS X:

`src/build/Darwin/bin# ./eqPly.app/Contents/MacOS/eqPly`

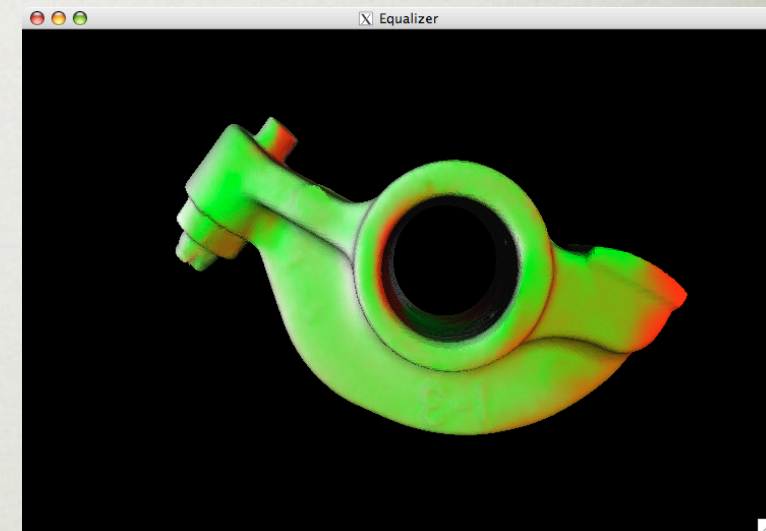
- Windows:

- debug 'eqPly Example'

- **OR:** `build\VS2005\win32\debug\eqPly.exe`

Running the Example Application

- eqPly runs now with default config
 - one window, one pipe thread, one process
- Left mouse button rotates
- Middle mouse button zooms
- Right mouse button moves
- Exit by pressing <Esc>, all three mouse buttons or using window close button



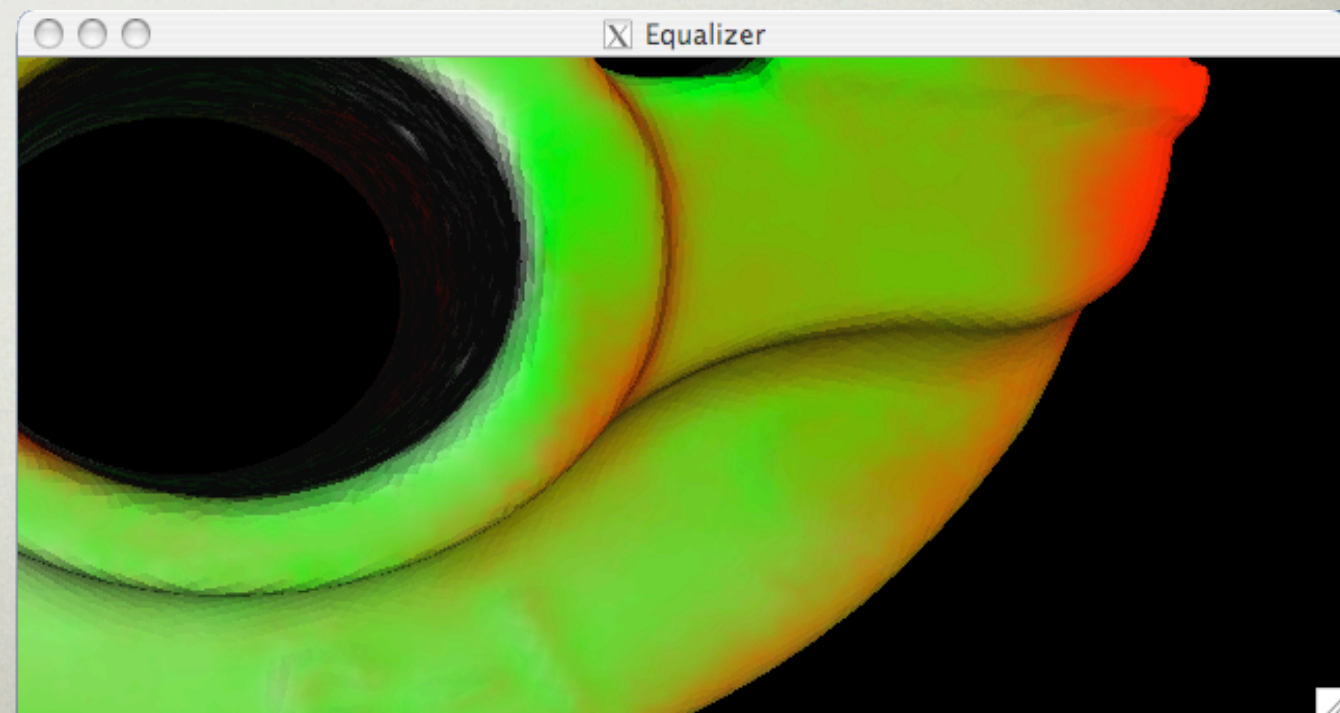
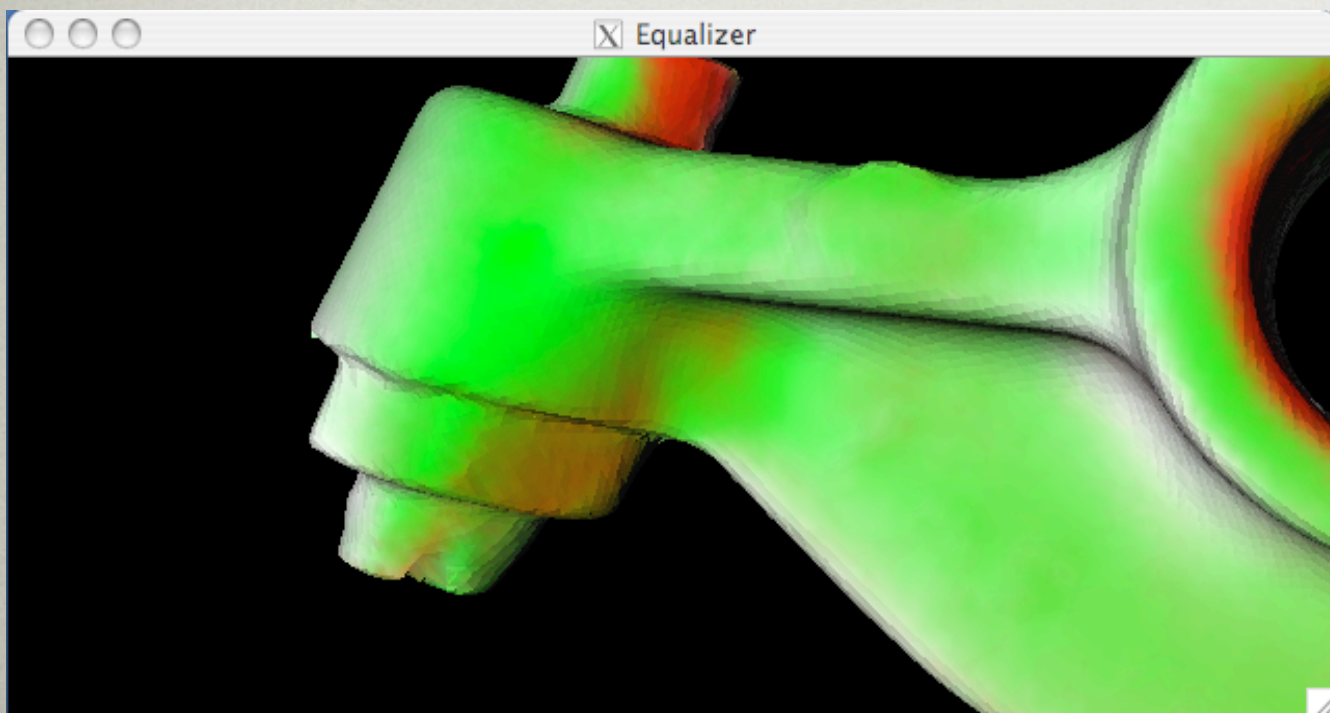
Exploring Equalizer

- To use a different config:
 - exit eqPly; stop server
 - start server with new config:

```
./eqServer <examples>/configs/2-window.eqc
```
 - run eqPly again
- Load model with '--model <name>'
 - Sample Models at www.cyberware.com

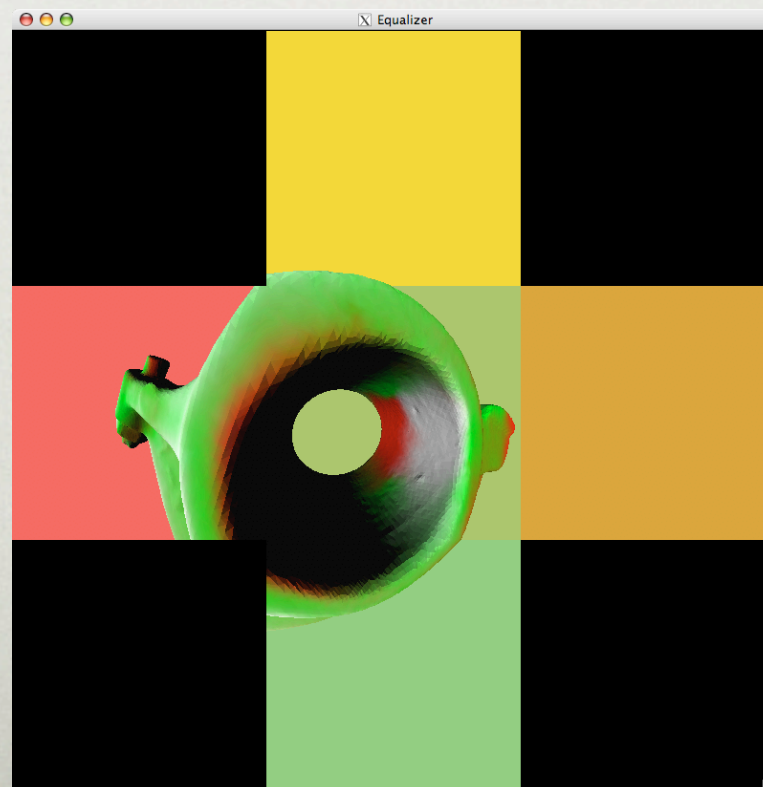
2-window

- Two windows, one pipe thread
- Compound wall descriptions produce side-by-side image



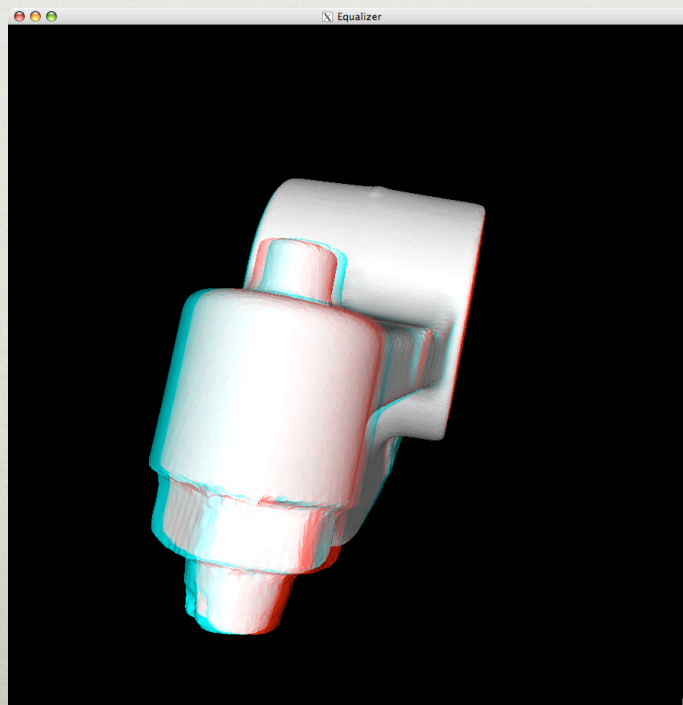
2-window

- Set EQ_TAINT_CHANNELS to get channel background colors
- One window, five channels
- Simulate a CAVE™ on a single PC



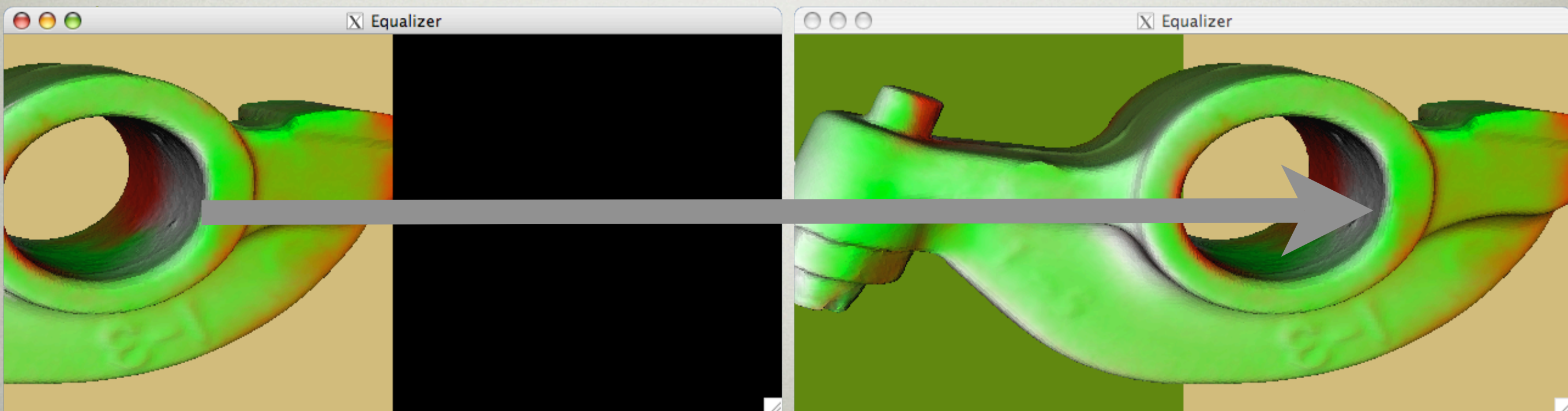
1-pipe.stereo.anaglyph

- Start eqPly with option -b
- Use anaglyphic (colored) glasses
- Two sequential eye passes
- Support for active (quad-buffer) stereo



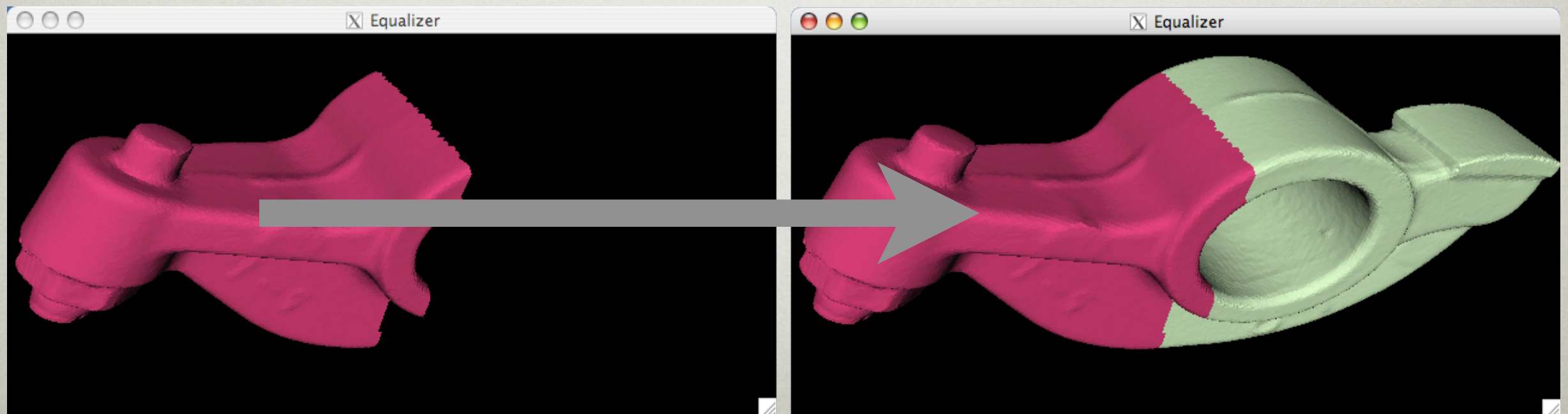
2-window.2D

- Left window renders half of the viewport for right window
- For deployment, windows are on separate pipes (GPUs) for scalability



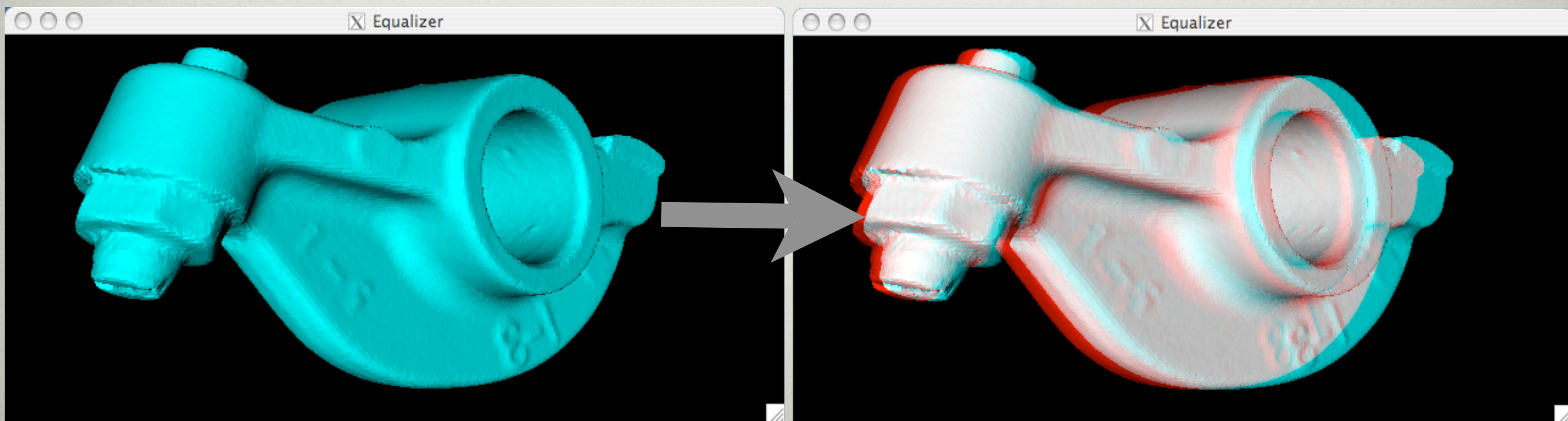
2-window.DB

- Left window renders part of the database for the right window
- Coloring is implemented in eqPly
- Data is combined using Z-Buffer information



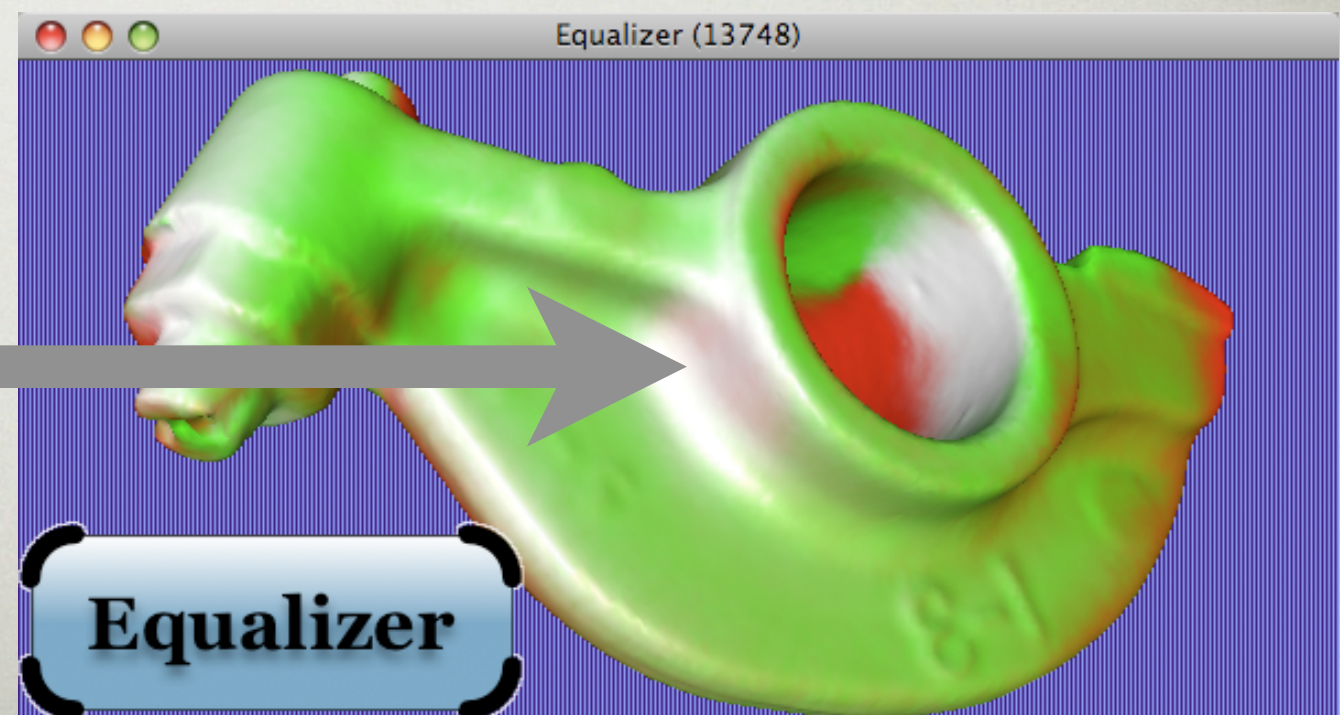
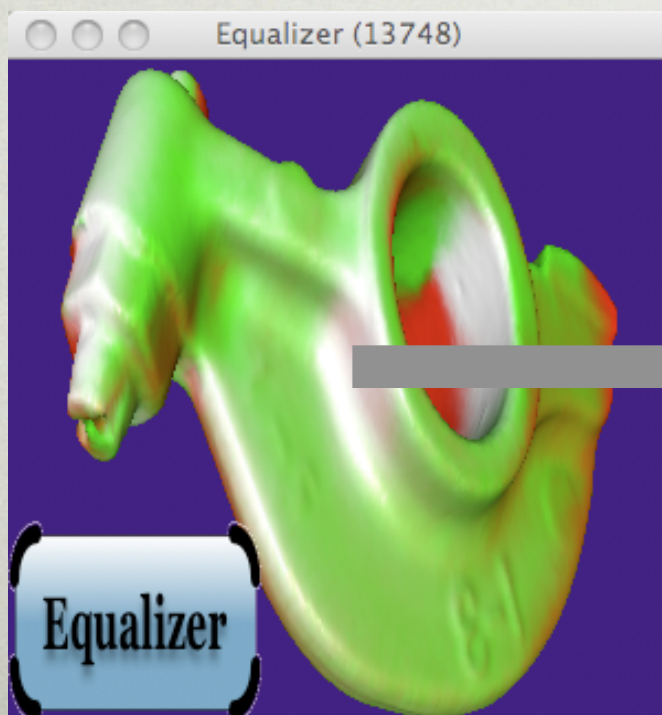
2-window.EYE.anaglyph

- Left window renders right eye
- Right window renders left eye
- Very good scalability on two pipes
- Also works for active stereo



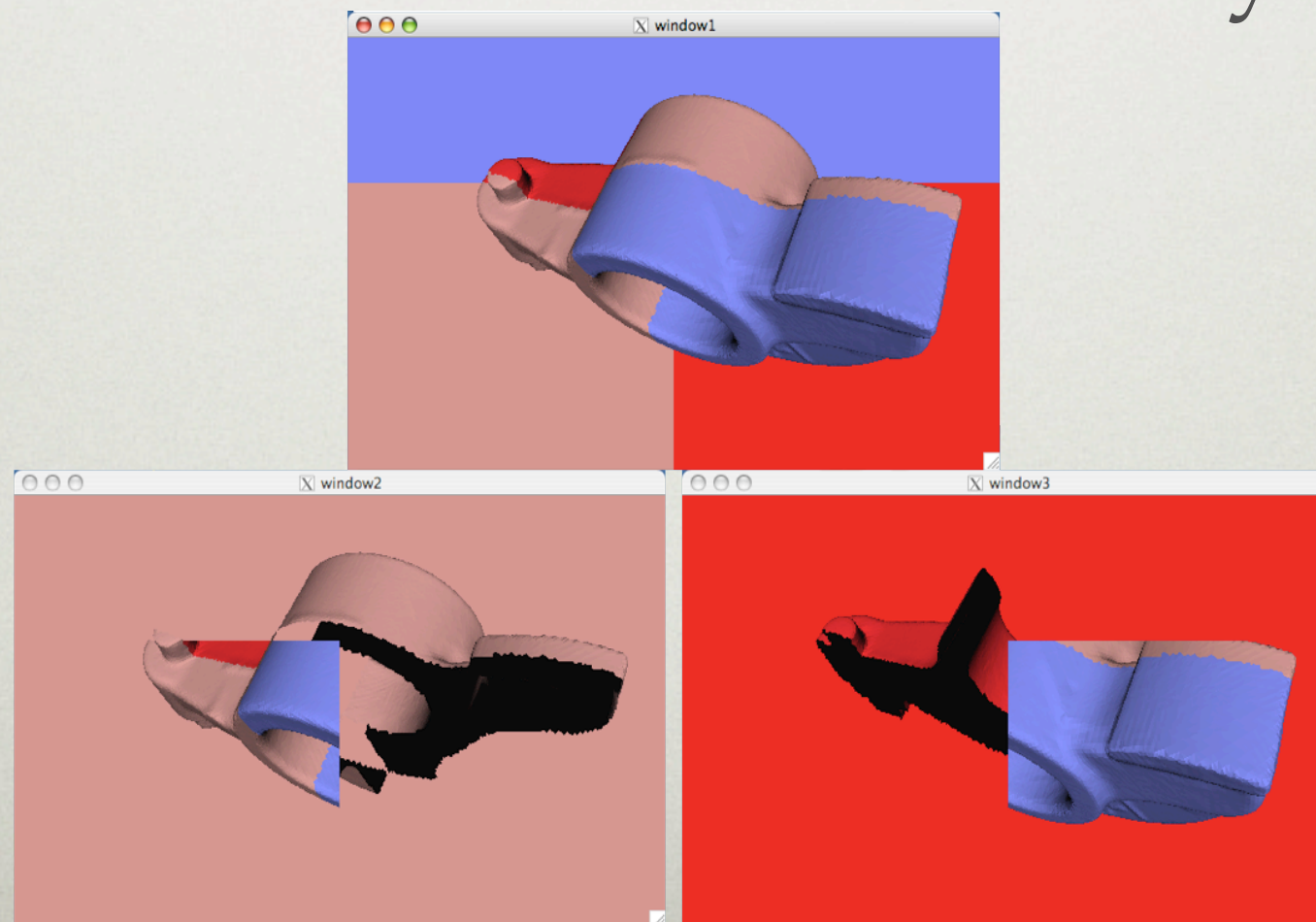
2-window.PIXEL

- Left window renders even lines
- Right window renders odd lines
- Very good scalability for fill-limited applications, e.g., raytracing



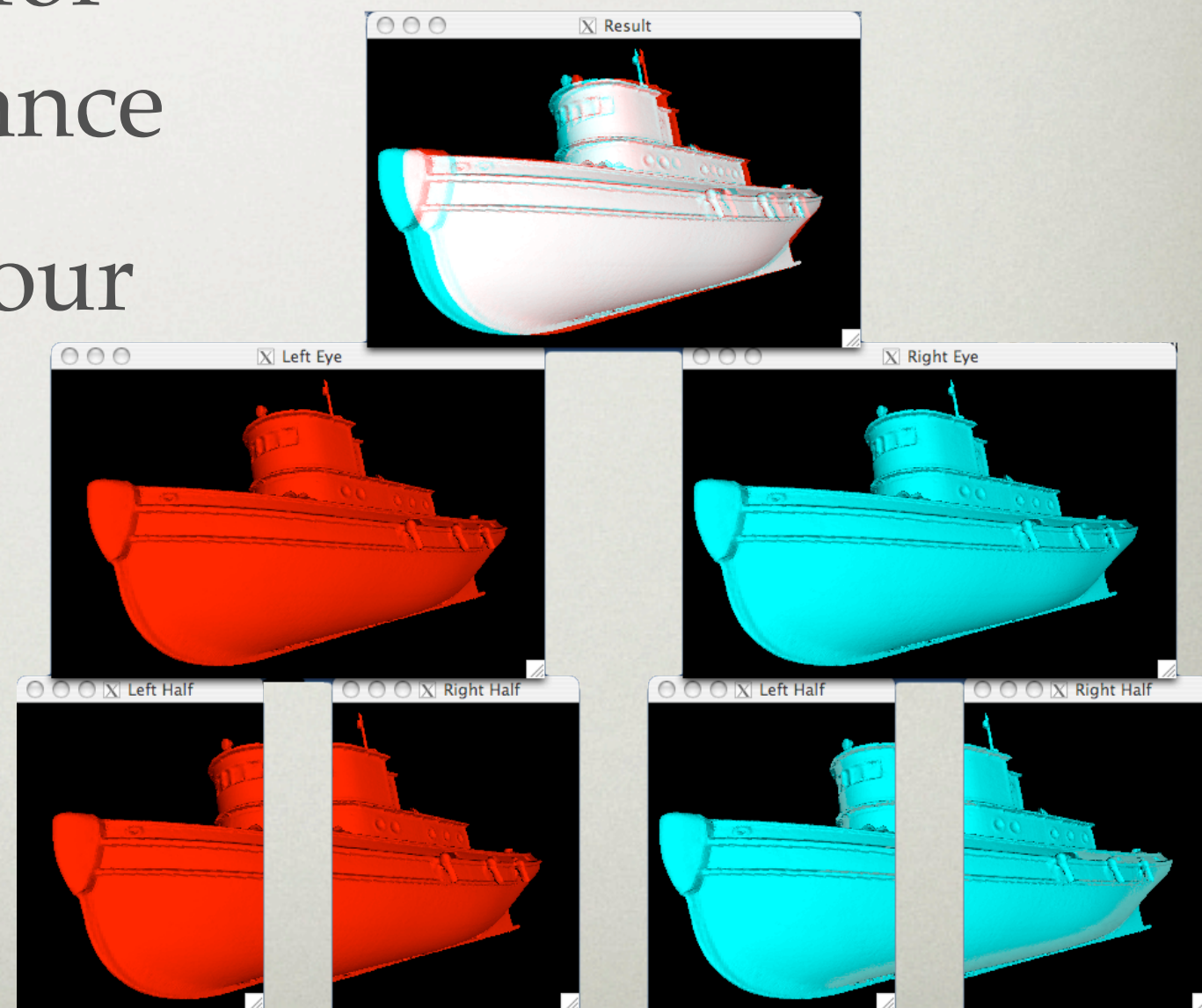
3-window.DB.ds

- Parallel compositing (direct send)
- Each channel renders and composites
- See 4-window.DB.bs for binary swap



7-window.EYE.2D

- Multilevel configuration
- Combine modes for optimal performance
- Deployment on four pipes



Next Steps

- Cluster example configurations are named *n-node.*.eqc*
- Password-less ssh setup needed
- Change hostnames to reflect your setup
- ConfigTool creates some configurations
- Active stereo requires stereo visuals
- Read configuration file specification