

Remaining challenges for real-time ray tracing systems
William R. Mark, University of Texas at Austin



SIGGRAPH2005

Course #38 – Introduction to real-time ray tracing



SIGGRAPH2005

What should we do next?



SIGGRAPH2005

Overview

- What works now
- The missing pieces
- Promising approaches
 - For specific challenges
 - For overall system architecture



SIGGRAPH2005

Consider the whole system

- System components:
 - Algorithms
 - Software architecture and implementation
 - Hardware
- System must solve the right problems
 - Performance – 60 Hz @ 1280x1024
 - Functionality – better than a Z buffer



SIGGRAPH2005

Basic capabilities work now

- Eye rays
- Shadow rays
- Material shaders
- Whitted reflected & refracted rays (sort-of)
- Static scenes, plus rigid moving objects

Huge progress compared to a few years ago.

But not enough to displace Z buffer in mass market.



SIGGRAPH2005

What's missing?

- Inadequate performance for:
 - Deformable objects
 - “Scattered” secondary rays
 - E.g. hemisphere sampling
 - Anti-aliasing
- Need algorithmic fixes, not just raw compute
- Work in progress w/Gordon Stoll and others



SIGGRAPH2005

Challenge #1: Deformable objects

- Two possibilities:
 - Rebuild acceleration structure
 - Update acceleration structure
- Avoid updates to invisible objects
- Use lazy update/rebuild
 - Simple in concept
 - Complex in a real system



Image from Okan Arikan

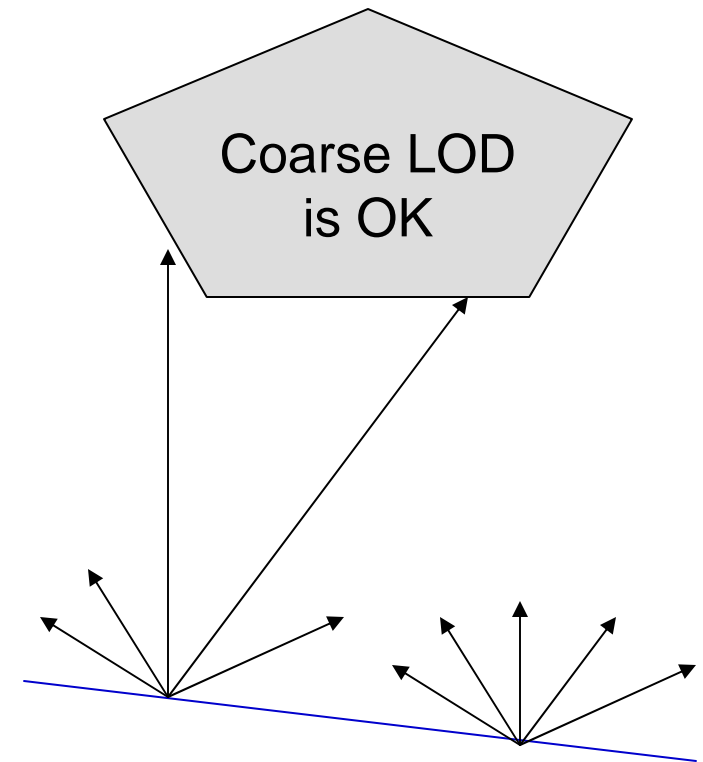
References: [Ar02, Luque05,...]



SIGGRAPH2005

Challenge #2: Scattered secondary rays

- Many rays
- Detailed geometry not needed!
- Adjustable geometric level-of-detail improves efficiency
[Christensen et al 2003]





SIGGRAPH2005

Ray tracing LOD is different

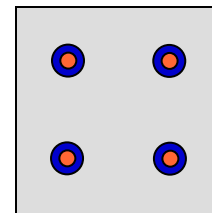
- LOD is per ray; not specified by single viewpoint
- Different rays may access same object with different LODs...
 - issues w/ cracking, tunneling, etc.
- Integration into complete system



SIGGRAPH2005

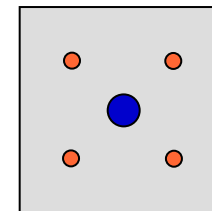
Challenge #3: Efficient anti-aliasing

- Current ray tracers tie material shading to visibility
- Result is excessive shading costs when anti-aliasing
- Compares poorly to Z buffer multisampling



Current ray tracers

- = shading sample
- = visibility sample



Z-buffer multisampling



SIGGRAPH2005

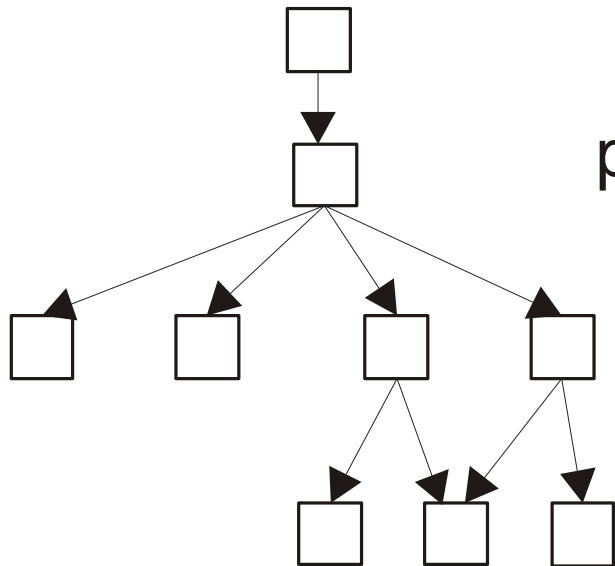
Must decouple visibility from material shading

- Z-buffer multisampling and REYES do this
- Similar technique needed for ray tracing.
- Basic idea: Extend REYES to ray tracing
- Generalize approach to tracing secondary rays

Meeting challenges requires new system organization

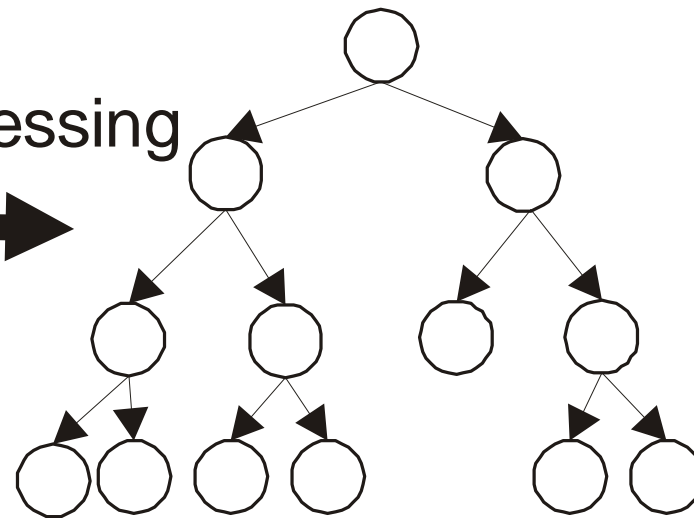
Classical Ray Tracing System

Scene graph
for scene management



Spatial acceleration structure
for visibility/rendering

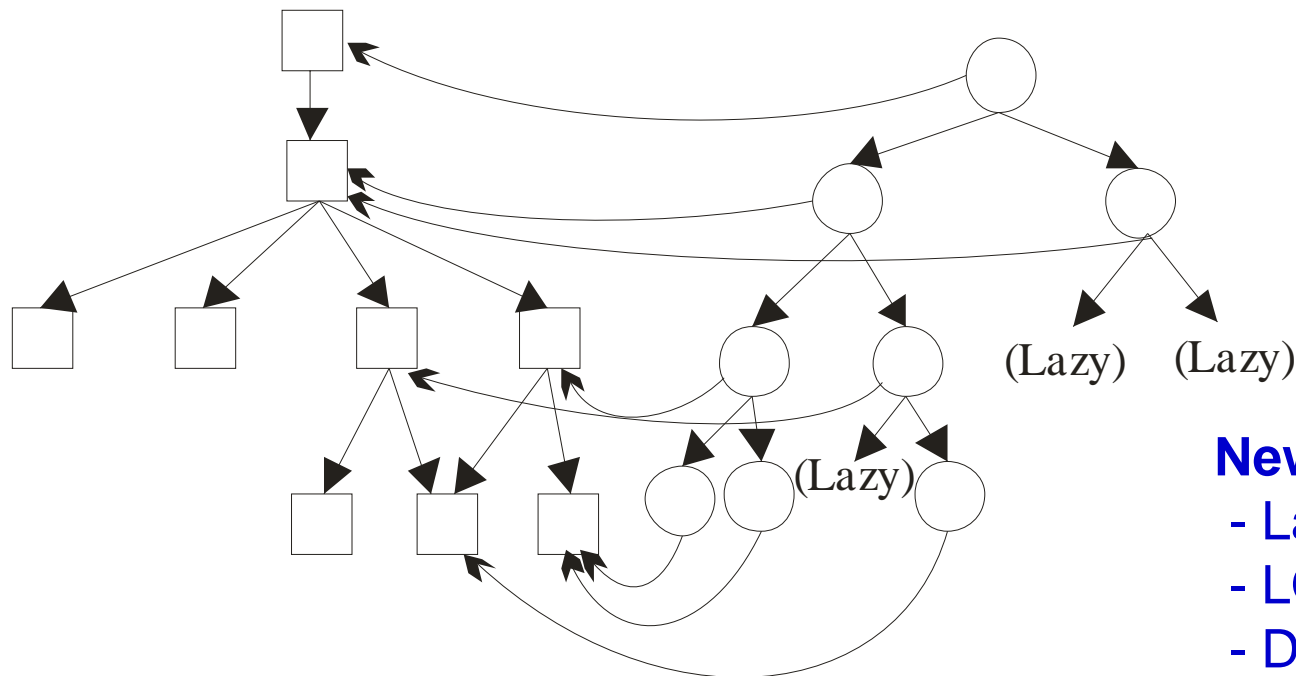
preprocessing





Meeting challenges requires new system organization

**Proposed ray tracing system:
Integrate scene graph and acceleration structure**



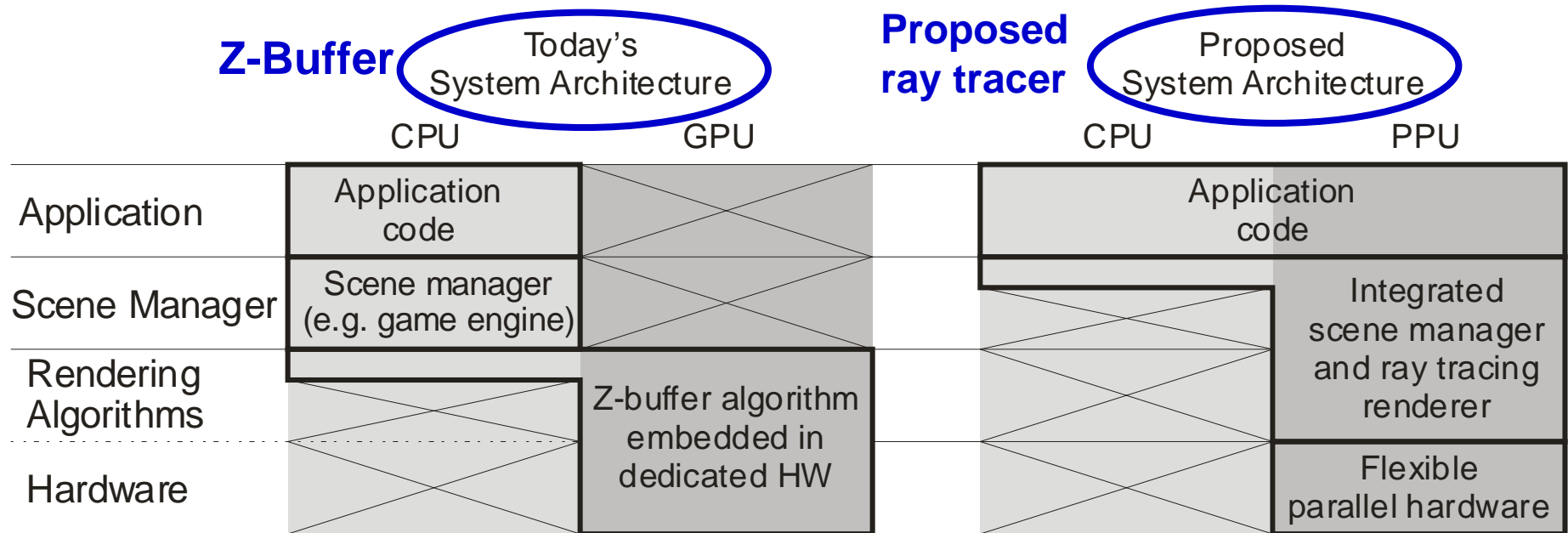
New design facilitates:

- Lazy updates
- LOD management
- Decoupling of visibility from shading



SIGGRAPH2005

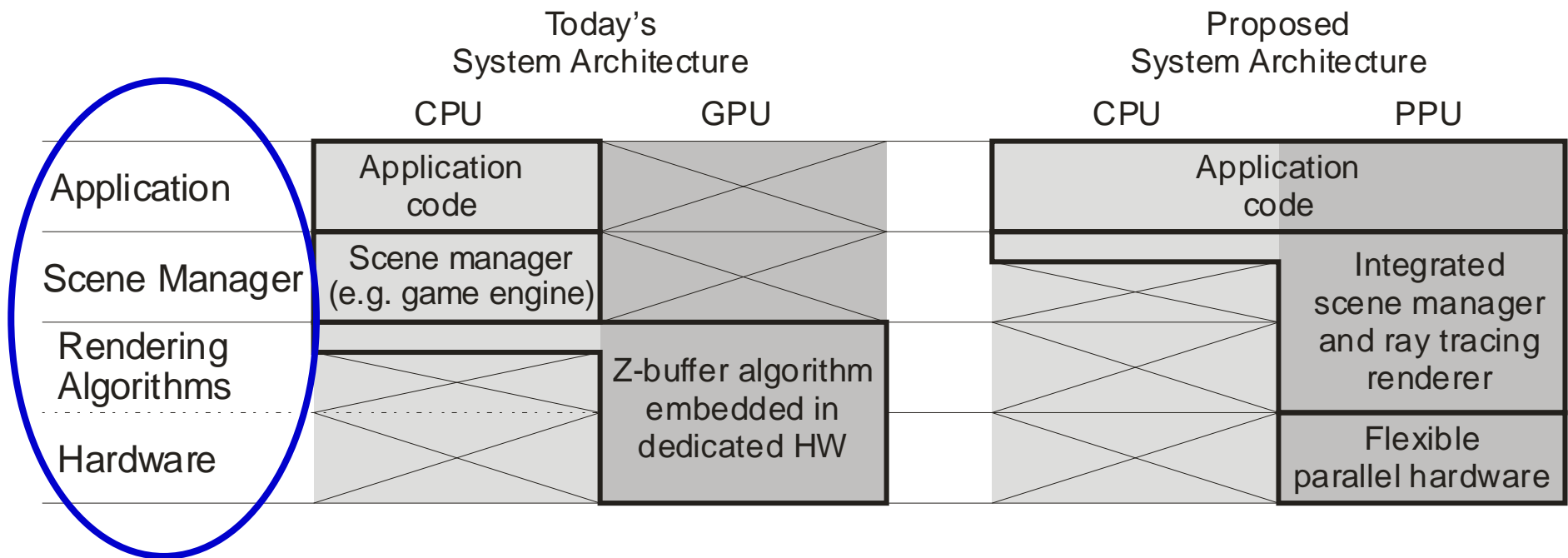
Changes to both HW & SW compared to today's GPUs





SIGGRAPH2005

Changes to both HW & SW compared to today's GPUs

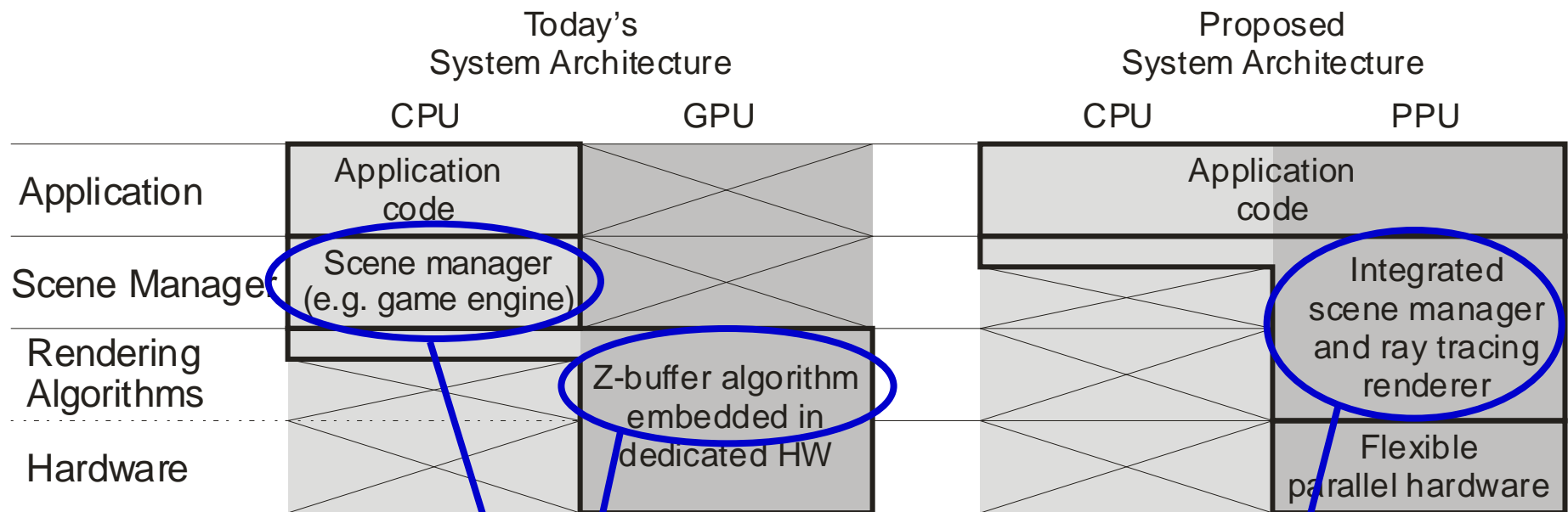


System Layers



SIGGRAPH2005

Changes to both HW & SW compared to today's GPUs



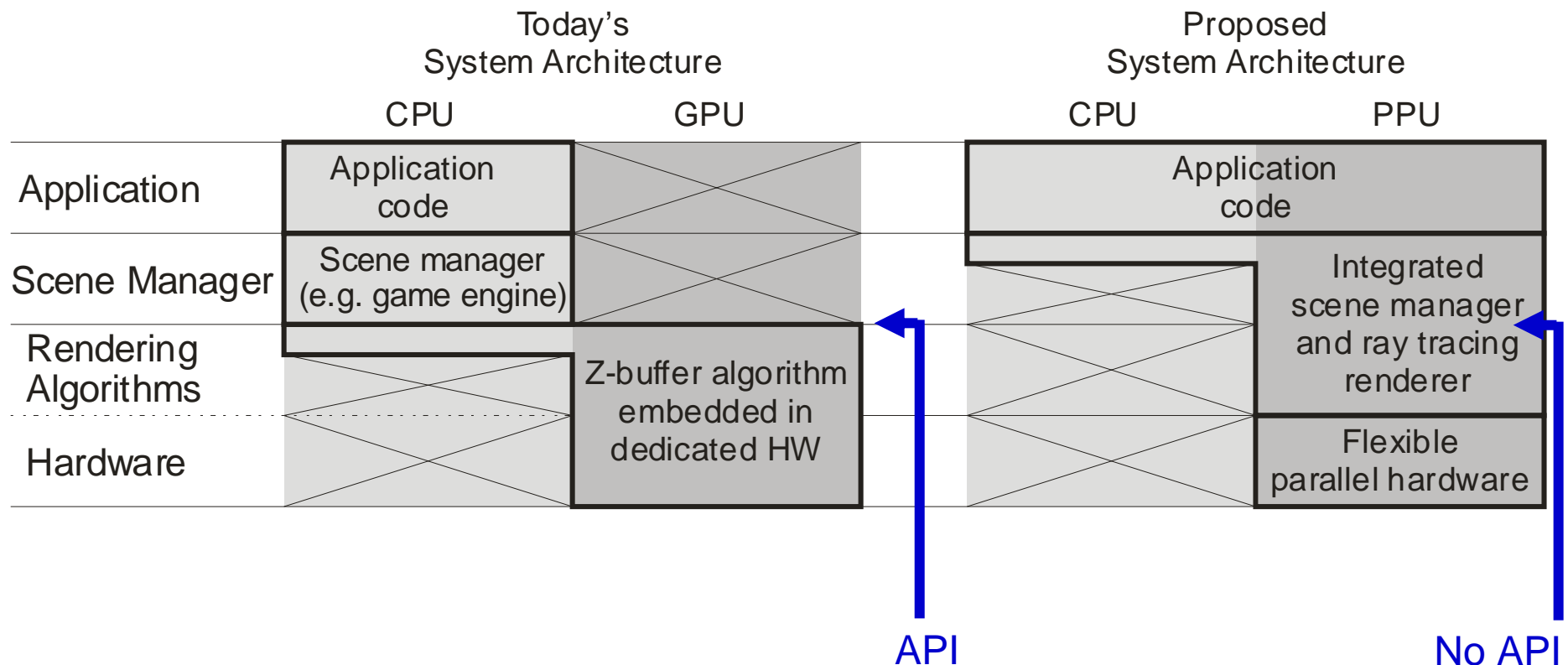
Scene graph and visibility are separated

Scene graph and visibility are unified



SIGGRAPH2005

Changes to both HW & SW compared to today's GPUs



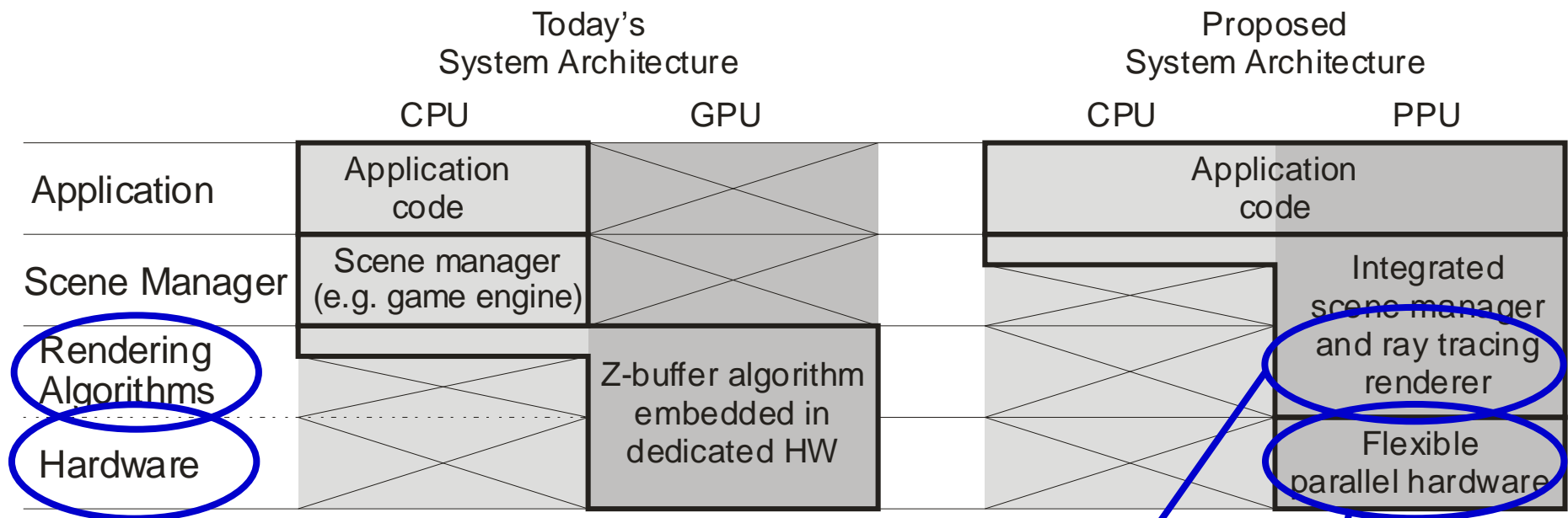
(it's all just "game engine")

Perhaps too extreme, but a useful way to think about the system



SIGGRAPH2005

Changes to both HW & SW compared to today's GPUs



A return to software rendering...

... but on highly-parallel hardware

Again, perhaps a bit too extreme. But Saarbrücken work is moving in this direction, and systems supporting deformable objects must go further.



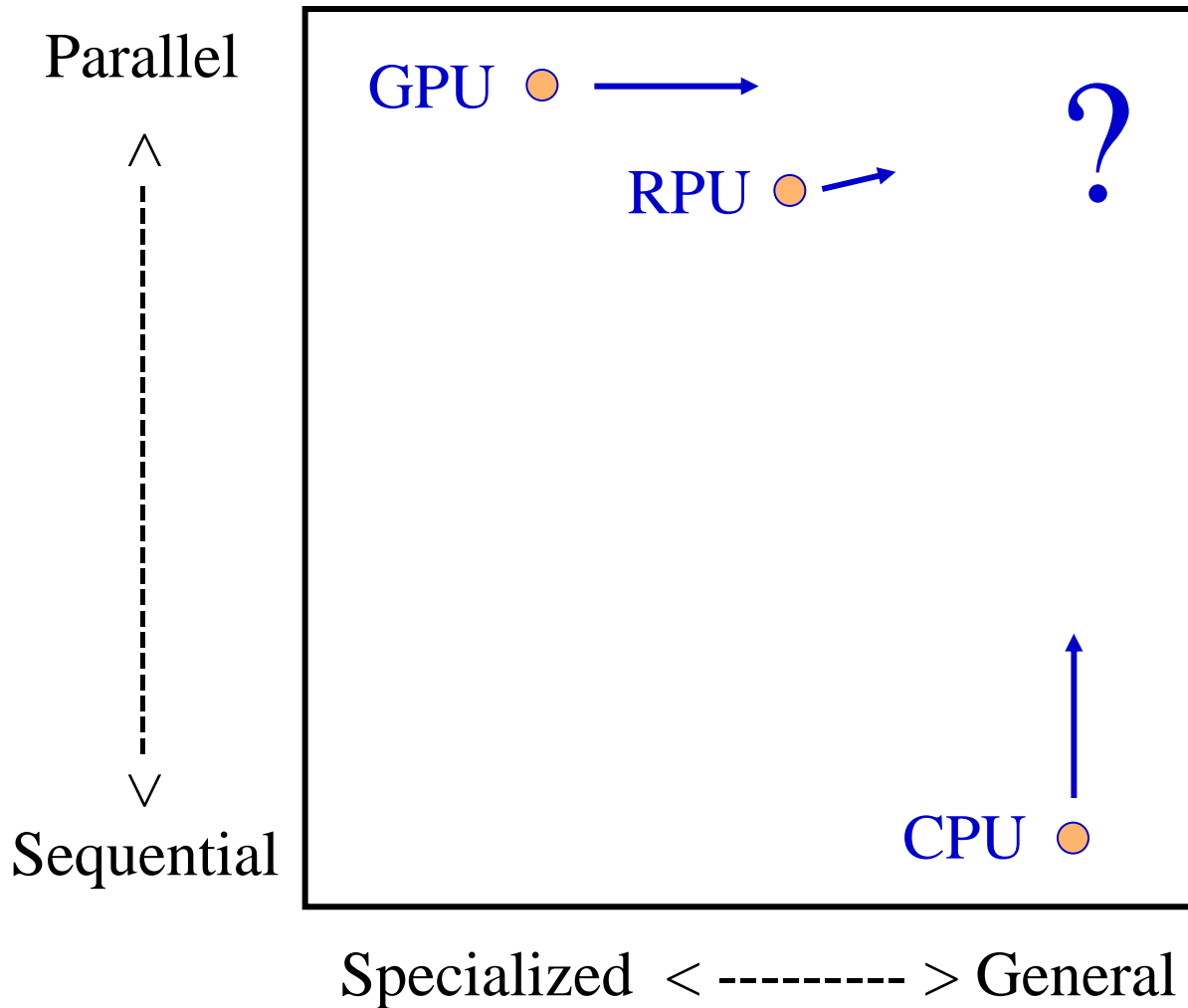
SIGGRAPH2005

Implications for hardware

- HW is parallel but flexible
 - Allow application-specific scene manager code
 - Some specialization for ray/surface intersection, etc.
 - Maybe two or more kinds of cores
- HW must build acceleration structure
 - Parallel construction of irregular data structures is hard
 - Lots more work to do here, for SW and HW
- The basics:
 - Multi-core, multi-threaded, MIMD, caches, ... -- like RPU



Evolutionary path for HW not yet clear



No one is there yet



SIGGRAPH2005

Summary:

Status of real-time ray tracing

- Enormous recent progress
- Open issues:
 - Deformable objects
 - Scattered secondary rays
 - Efficient anti-aliasing
- Need algorithmic progress, not just raw FLOPS
- Must reconsider system organization
- HW will have to be very flexible



SIGGRAPH2005

More information

- Overall system organization
 - “Real-time rendering in 2010”, Mark and Fussell
(in course notes)
- More next year?
 - Ongoing work by:
Gordon Stoll, myself, Don Fussell, Peter Djeu,
and others.