CS 380 - GPU and GPGPU Programming
Lecture 28: GPU Virtual Texturing

Markus Hadwiger, KAUST

King Abdullah University of Science and Technology
KAUST

Reading Assignment #11 (until Nov 18)

Read (required):

- Look at Vulkan *sparse resources*, especially *sparse partially-resident images*
  - `https://docs.vulkan.org/spec/latest/chapters/sparsemem.html/`
- Read about shadow mapping
  - `https://en.wikipedia.org/wiki/Shadow_mapping/`
- Look at Unreal Engine 5 virtual texturing
  - `https://dev.epicgames.com/documentation/en-us/unreal-engine/virtual-texturing-in-unreal-engine/`

Read (optional):

- CUDA Warp-Level Primitives
  - `https://developer.nvidia.com/blog/using-cuda-warp-level-primitives/`
- Warp-aggregated atomics
  - `https://developer.nvidia.com/blog/cuda-pro-tip-optimized-filtering-warp-aggregated-atomics/`

Next Lectures

Lecture 29: Thu, Nov 14: 10:00-11:30 (on Zoom)

Lecture 30: Mon, Nov 18: Quiz #3

GPU Virtual Texturing

Virtual Texturing

Example #1:

ARB Sparse Textures (originally: AMD Partially Resident Textures)

ARB_sparse_texture / ARB_sparse_texture2

`https://www.khronos.org/registry/OpenGL/extensions/ARB/ARB_sparse_texture.txt`

`https://www.khronos.org/registry/OpenGL/extensions/ARB/ARB_sparse_texture2.txt`

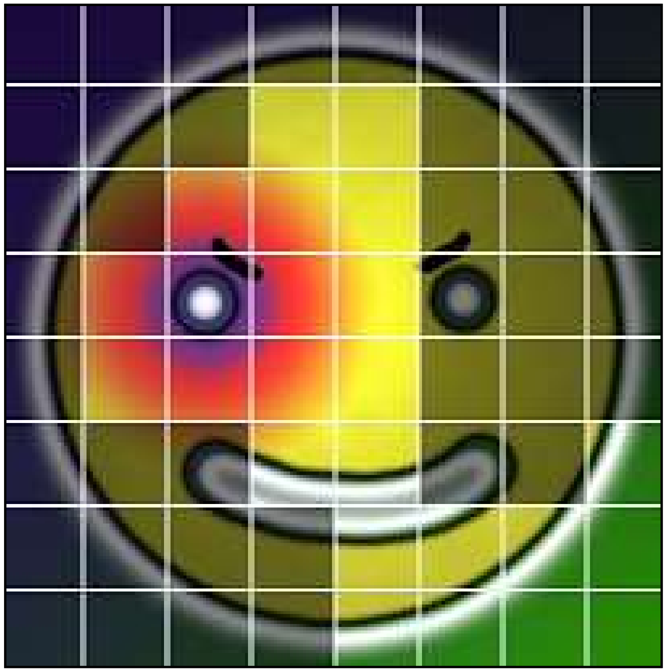
Hardware Virtual Texturing, Graham Sellers,
from SIGGRAPH 2013 course “Rendering Massive Virtual Worlds”

`https://cesiumjs.org/hosted-apps/massiveworlds/downloads/Graham/Hardware_Virtual_Textures.pptx`

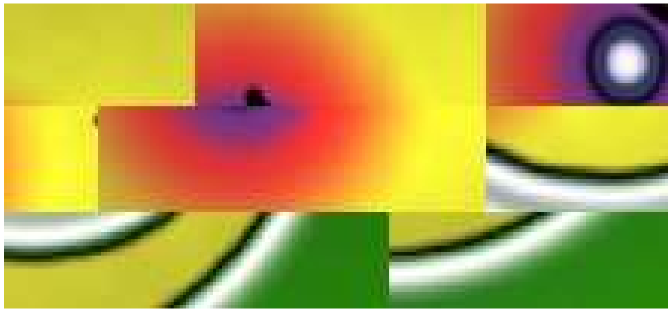
Virtual Texturing

Divide texture up into tiles

- Commit only *used* tiles to memory
- Store data in separate physical texture



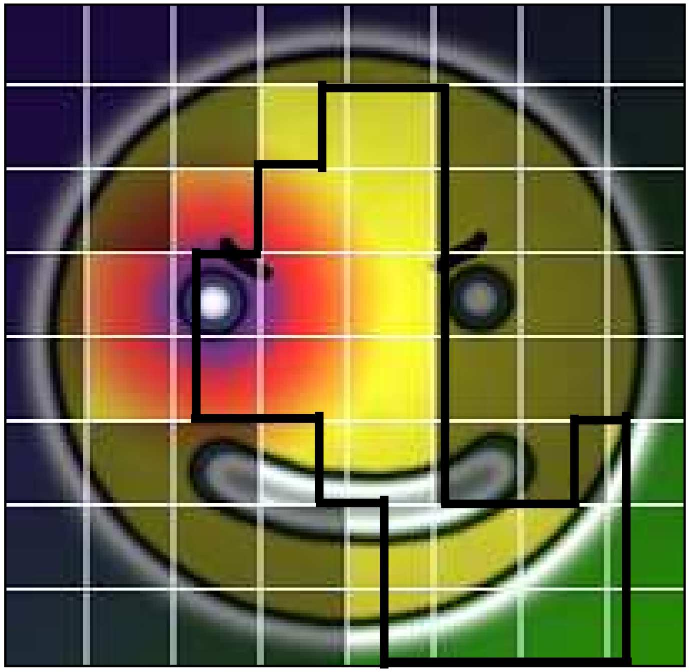
Virtual Texture



Physical Texture

Virtual Texturing

Memory requirements set by number of resident tiles, not texture dimensions



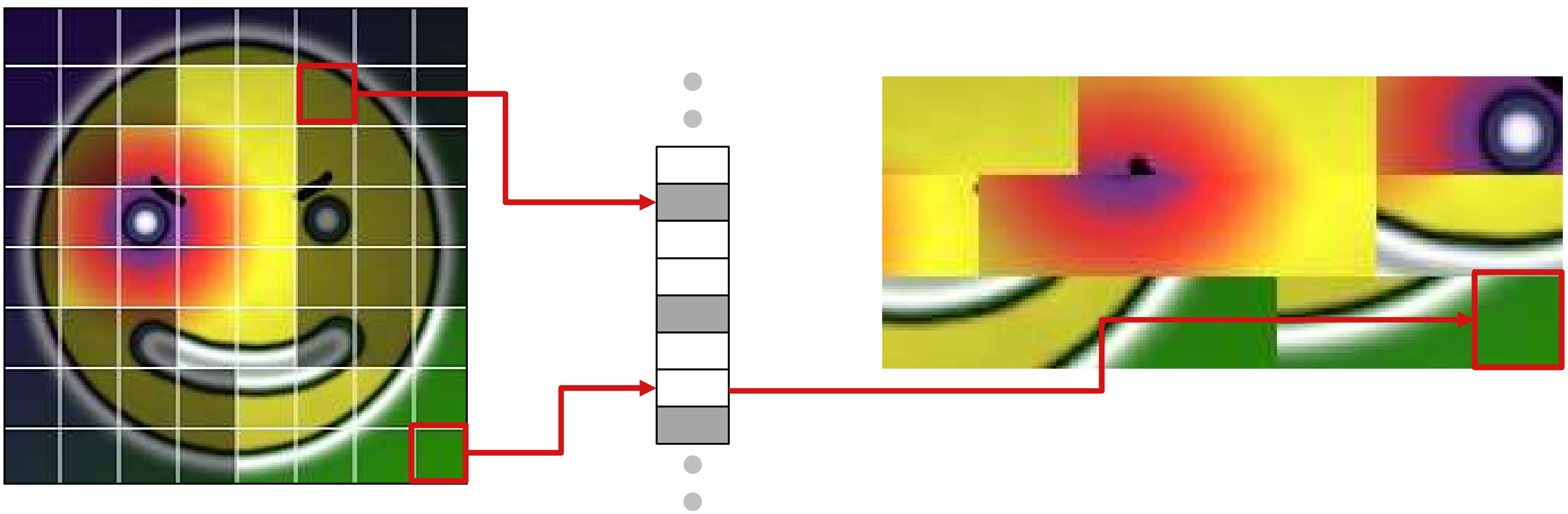
RGBA8, 1024x1024, 64 tiles

	Virtual	Physical
Memory	4096 kB	1536 kB

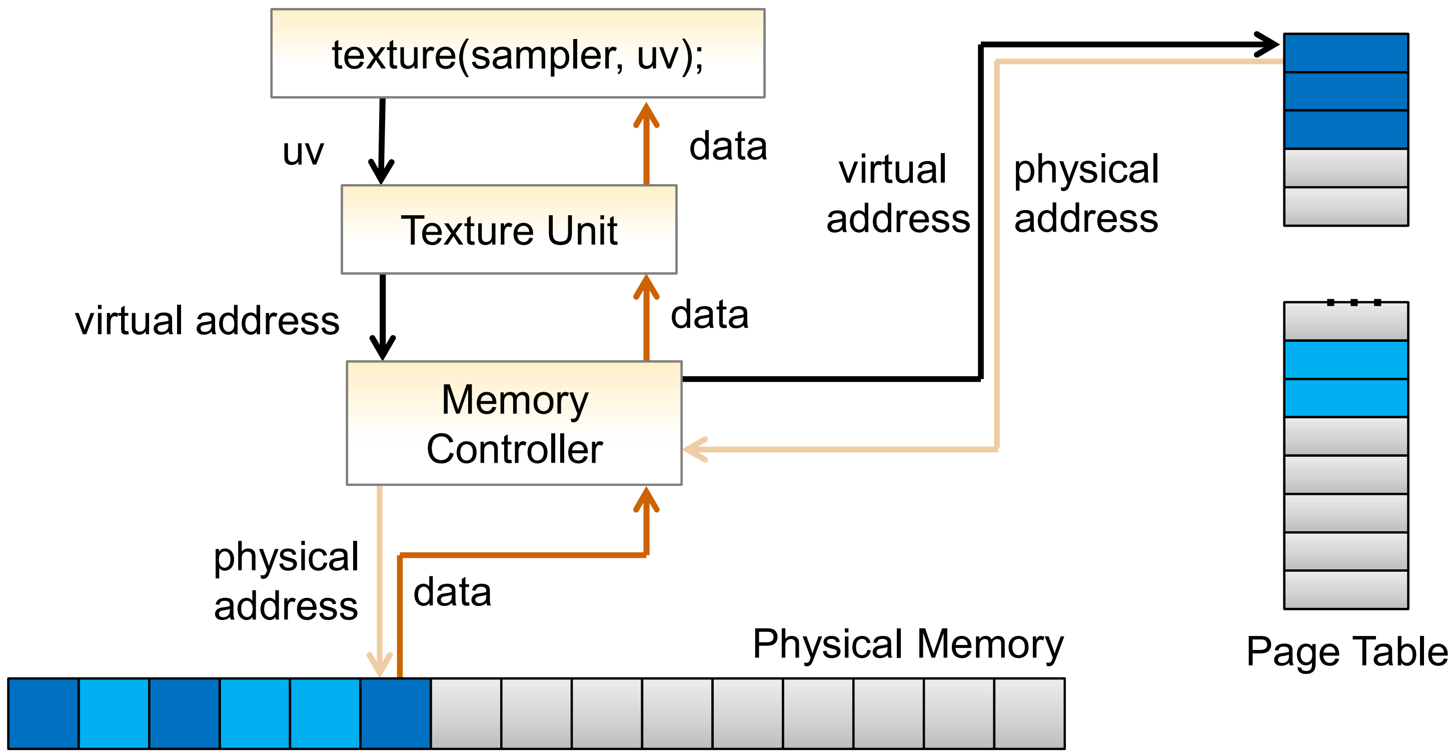
Virtual Texturing

Use indirection table to map virtual to physical

- This is also known as a *page table*



GPU Virtual Memory



Summary (Shader vs. Full Hardware Support)

	SVTs	HVTs
Address translation	Shader code	HW page table
Filtering	HW + shader code	HW only
# of texture fetches	2, dependent	1
Supported formats	The ones implemented	All supported by HW
Supported texture types	The ones implemented	All supported by HW

Virtual Texturing

Example #2:

Adaptive Shadow Maps (ASM)

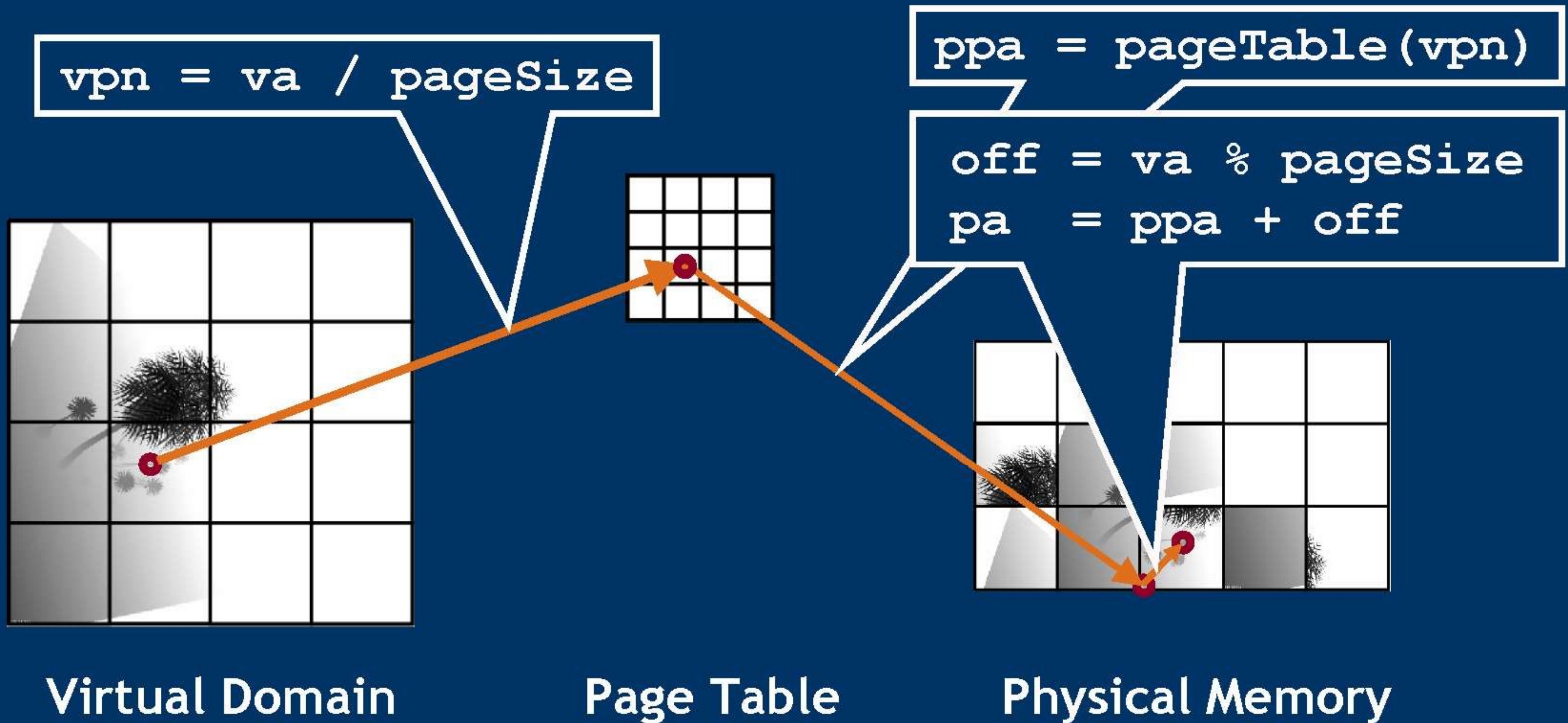
- On CPUs: Fernando et al., ACM SIGGRAPH 2001

Resolution-Matched Shadow Maps

- On GPUs: Aaron Lefohn et al., ACM Transactions on Graphics 2007

ASM Data Structure (Adaptive Shadow Maps)

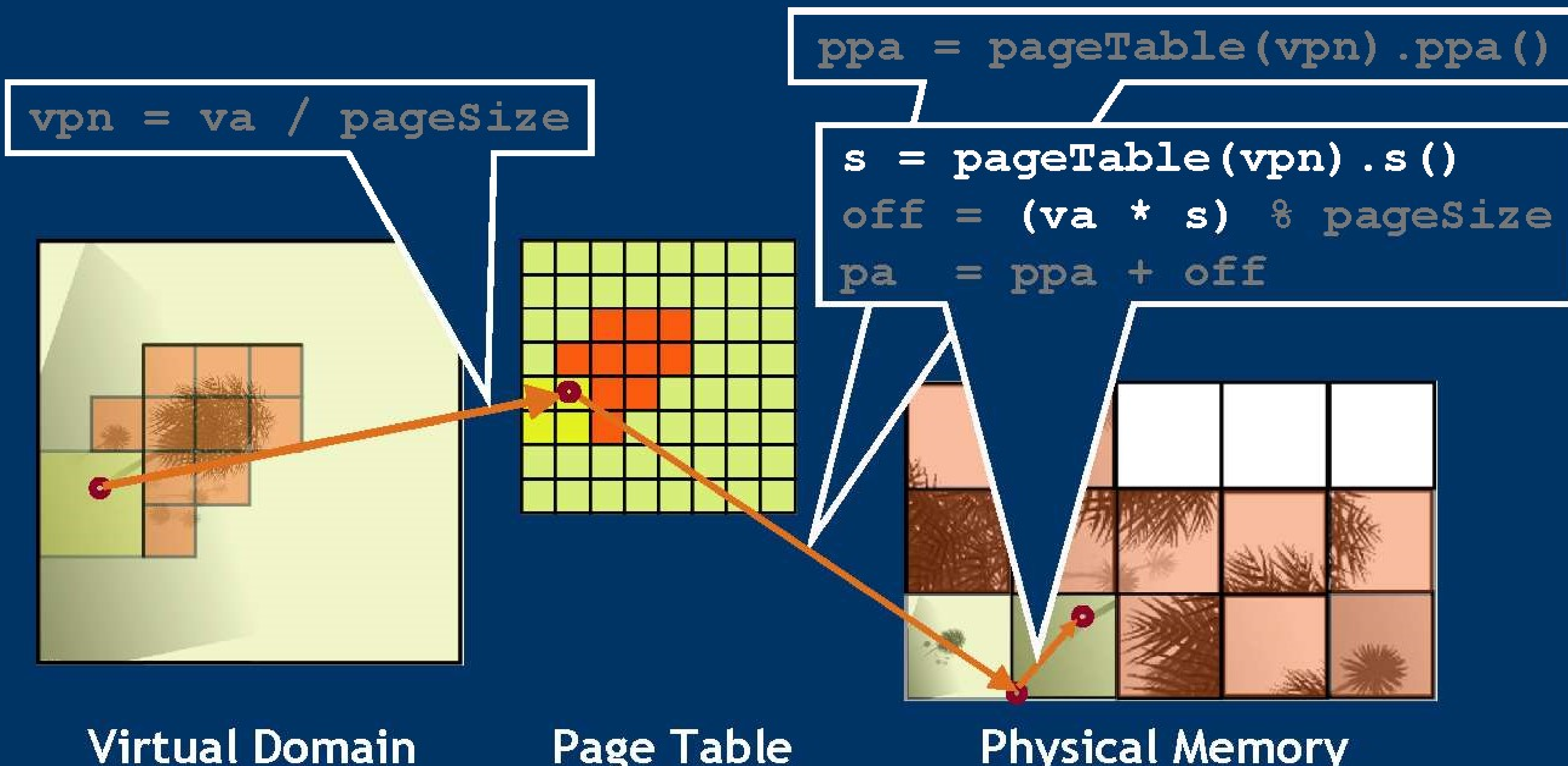
- Page table example



Aaron Lefohn
University of California, Davis

ASM Data Structure (Adaptive Shadow Maps)

- Adaptive Page Table
  - Map multiple virtual pages to single physical page



Aaron Lefohn
University of California, Davis

Virtual Texturing

Example #3:

id Tech 5 Megatextures, id Software

Rage

- Virtual Texturing in Software and Hardware, van Waveren et al., SIGGRAPH 2012 course notes + slides

`http://www.jurajobert.com/data/Virtual_Texturing_in_Software_and_Hardware_course_notes.pdf`

`http://www.mrelusive.com/publications/papers/Software-Virtual-Textures.pdf`

`http://www.mrelusive.com/publications/presentations/2013_siggraph/hq_sw_hw_vts_12.pdf`

Virtual Texturing

Rage / id Tech 5 (id Software)

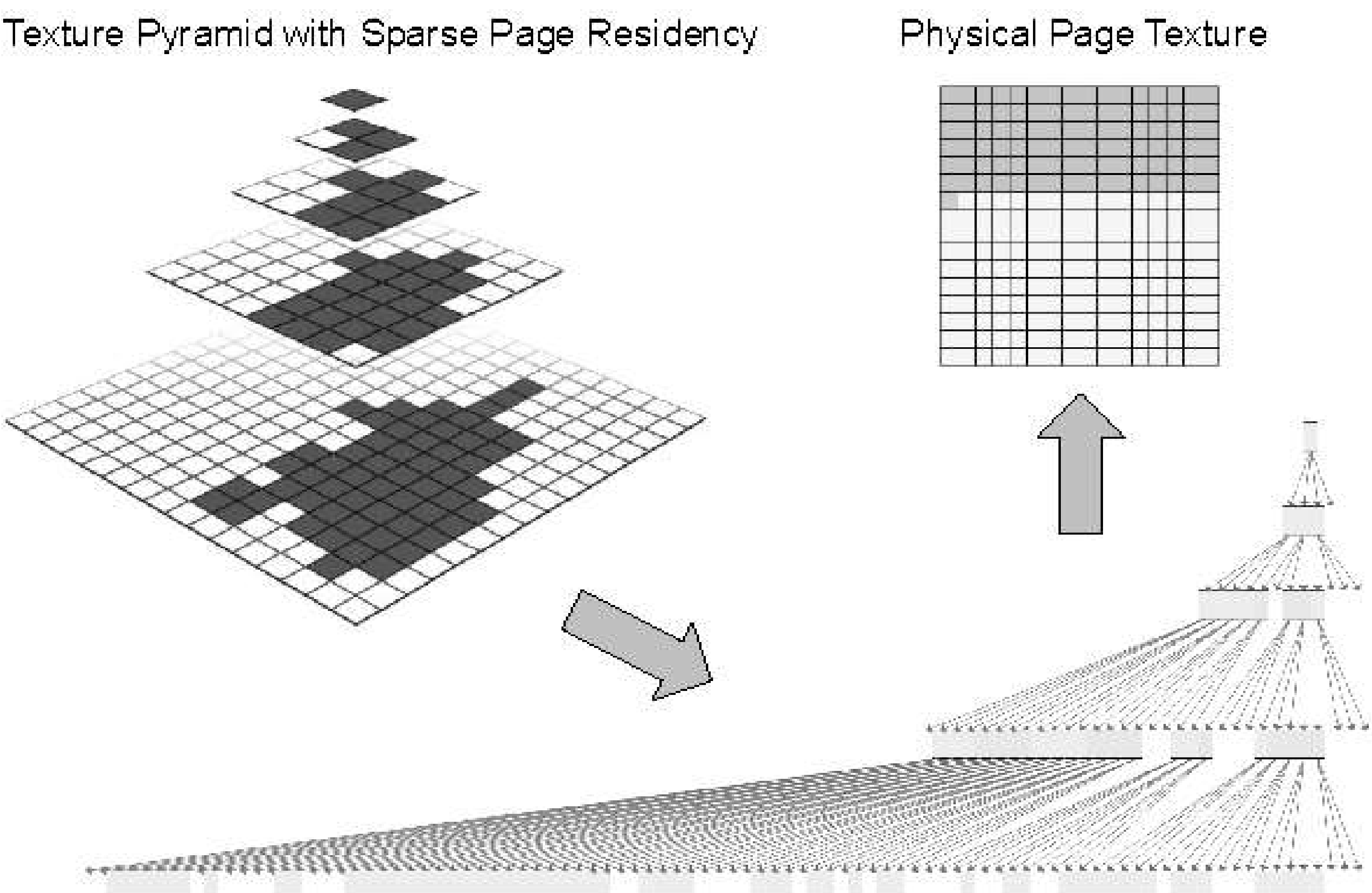
Virtual Texturing

- Unique, very large virtual textures key to id tech 5 rendering
- Full description beyond the scope of this talk

Virtual Texturing

Virtual Texturing

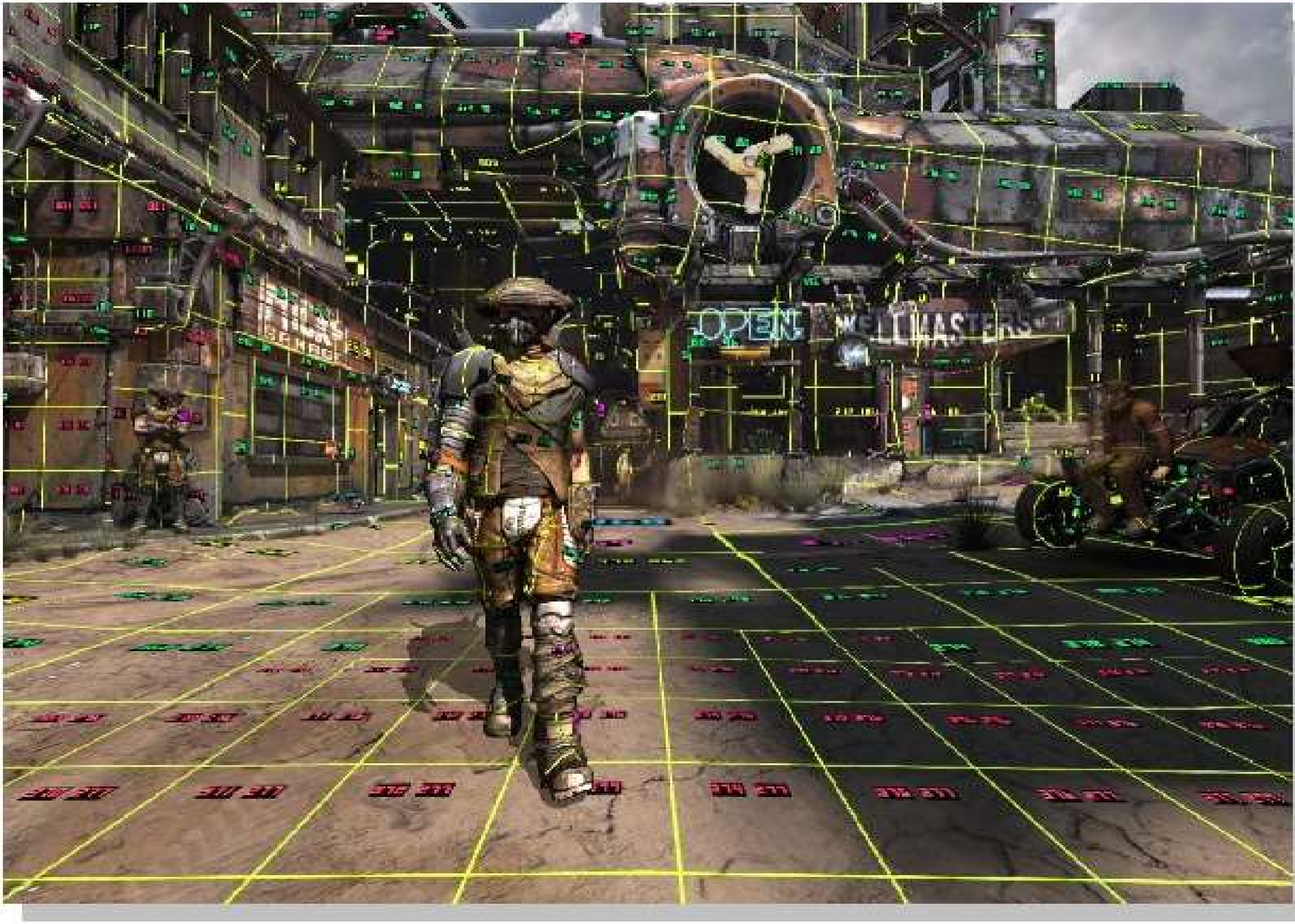
Virtual Texturing



Quad-tree of Sparse Texture Pyramid

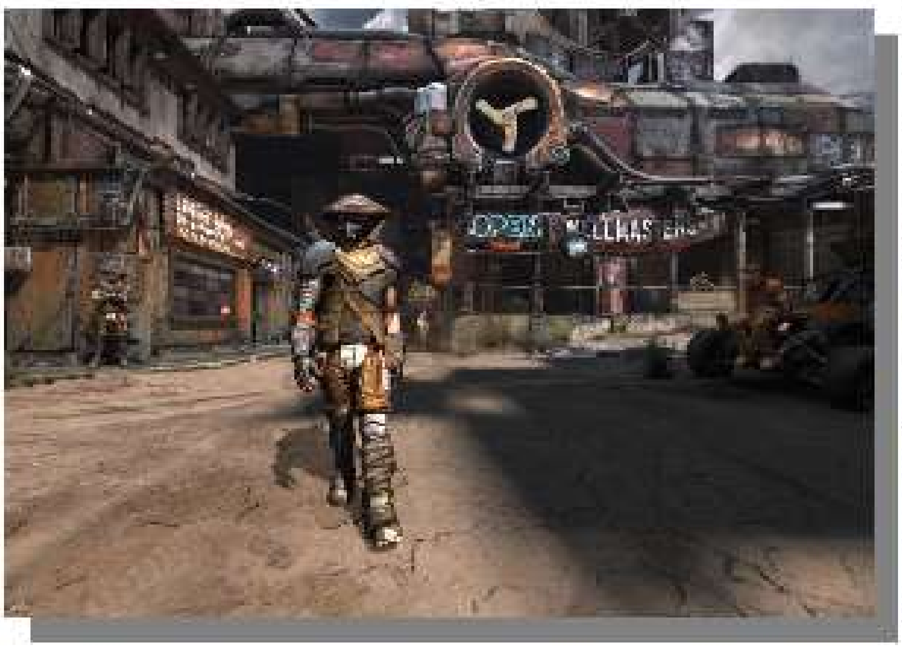
Virtual Texturing

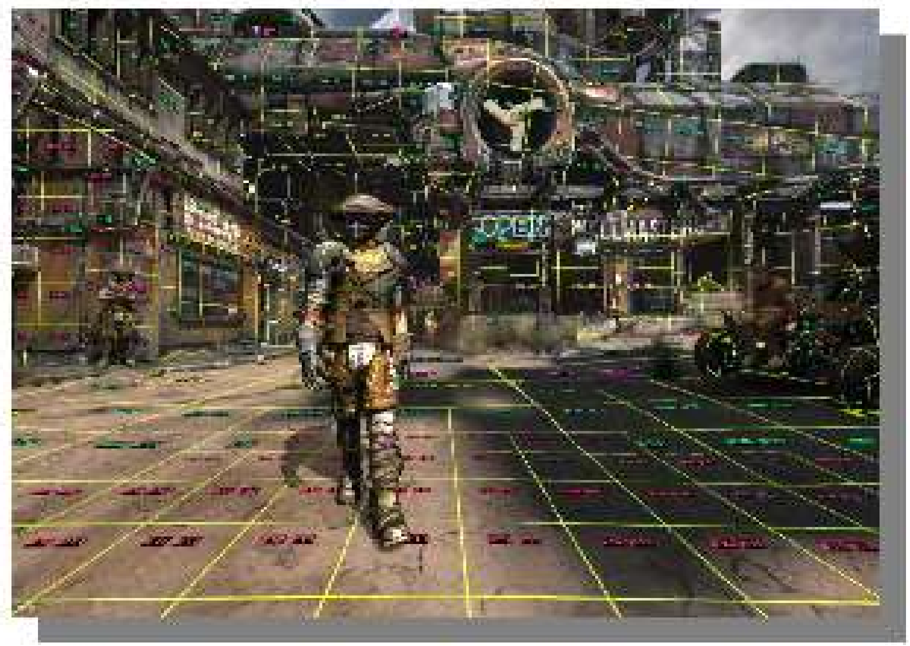
Virtual Texturing



Virtual Texturing

A few interesting issues...
- Texture filtering
- Thrashing due to physical memory oversubscription
- LOD transitions under high latency

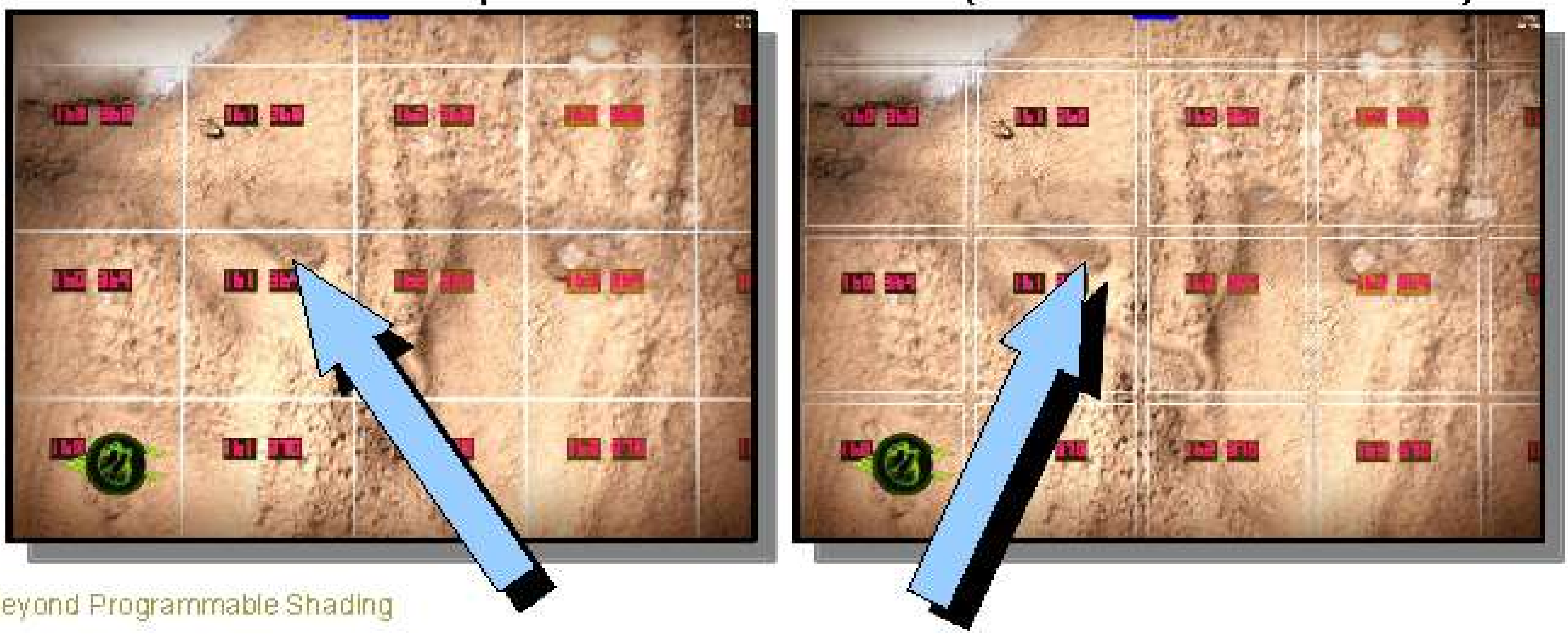




RAGE with PRTs (Image courtesy of id Software)

Virtual Texturing - Filtering

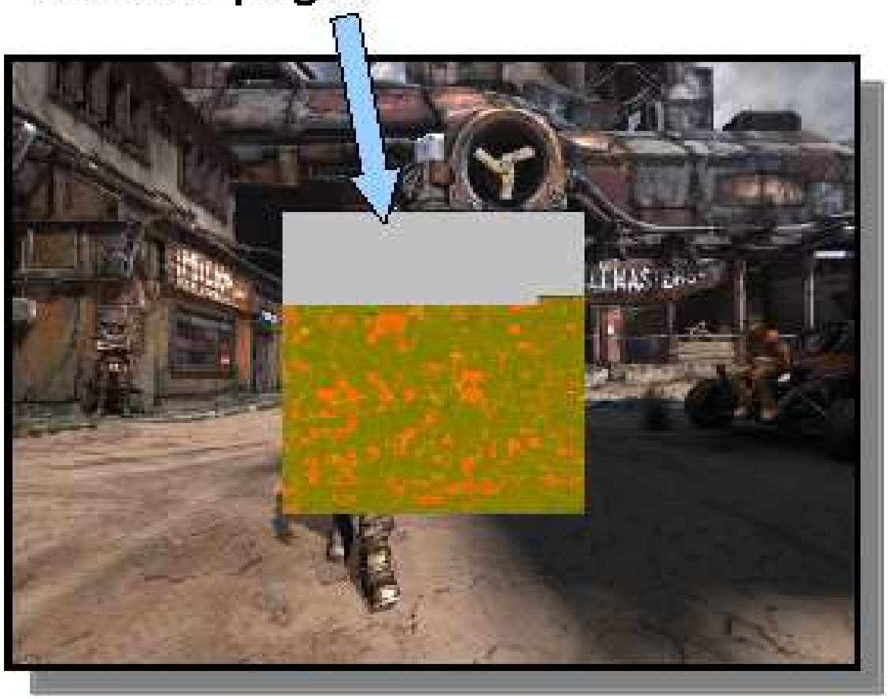
- We tried no filtering at all
- We tried bilinear filtering without borders
- Bilinear filtering with border works well
- Trilinear filtering reasonably but still expensive
- Anisotropic filtering possible via TXD (texgrad)
  - 4-texel border necessary (max aniso = 4)
  - TEX with implicit derivs ok too (on some hardware)



Virtual Texturing - Thrashing

- Sometimes you need more physical pages than you have
- With conventional virtual memory, you must thrash
- With virtual texturing, you can globally adjust feedback LOD bias until working set fits

32 x 32 pages



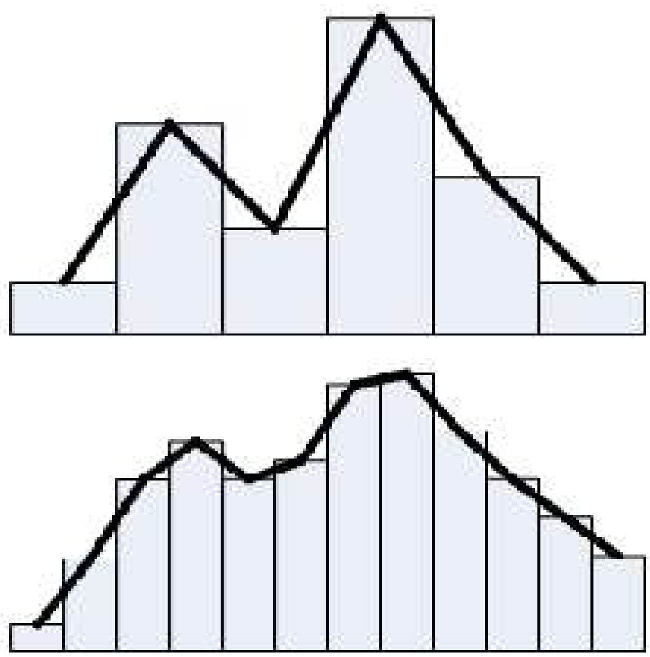
1024 Physical Pages

8x8 pages

64 Physical Pages

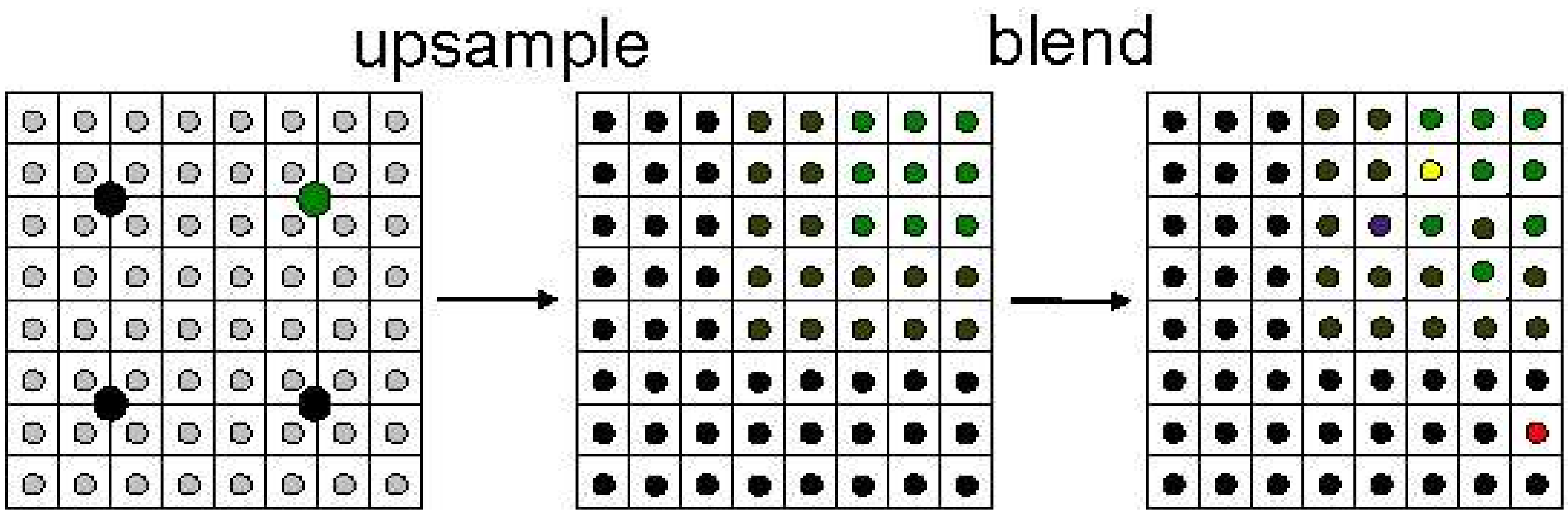
Virtual Texturing – LOD Snap

- Latency between first need and availability can be high
  - Especially if optical disk read required (>100 msec seek!)
- Visible snap happens when magnified texture changes LOD
- If we used trilinear filtering, blending in detail would be easy
- Instead continuously update physical pages with blended data



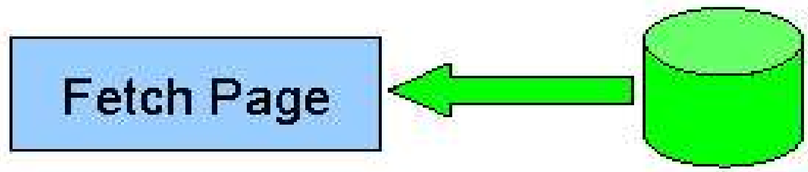
Virtual Texturing – LOD Snap

- Upsample coarse page immediately
- Then blend in finer data when available



Virtual Texturing - Management

- Analysis tells us what pages we need
- We fetch what we can



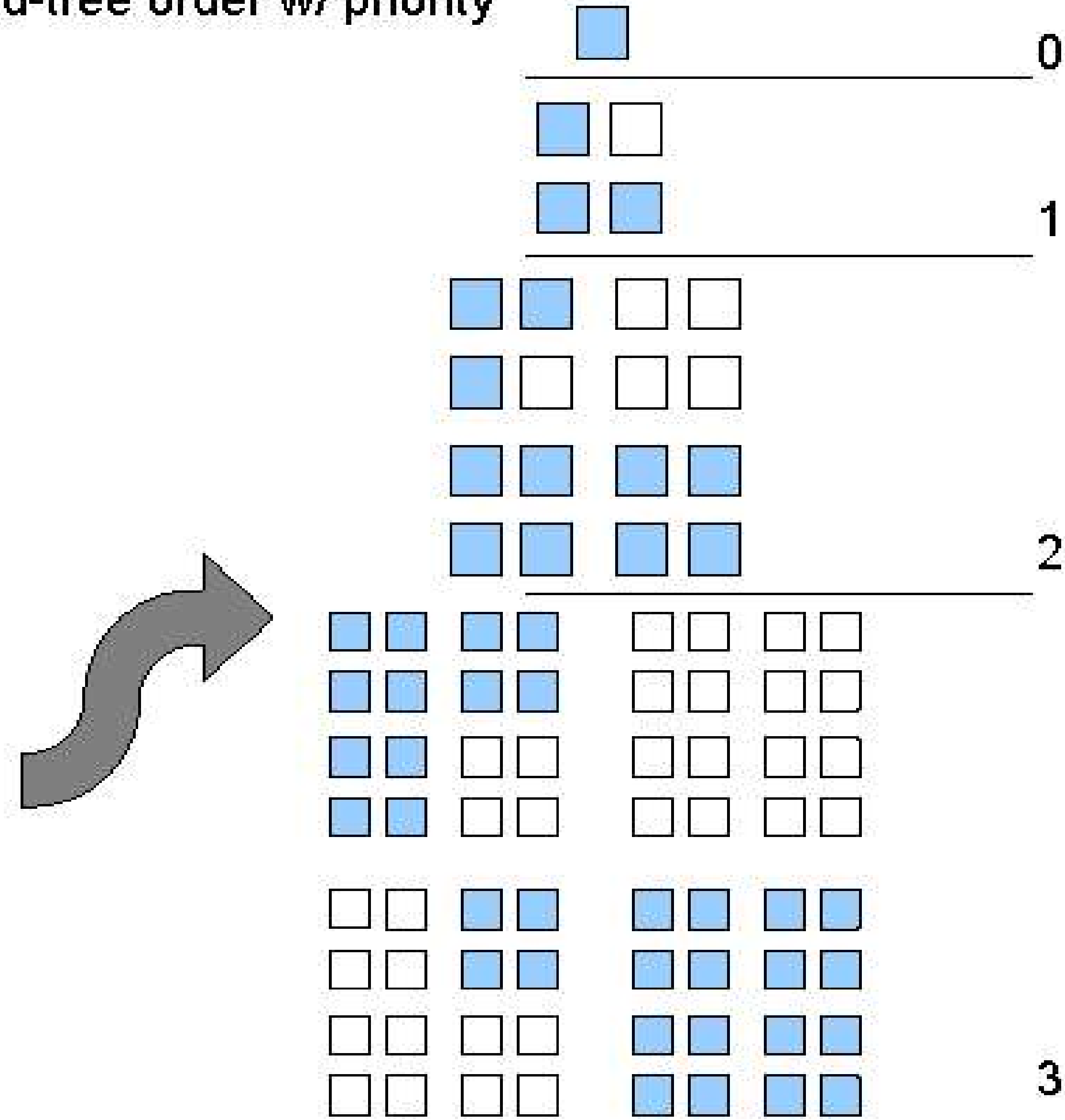
- But this is a real-time app... so no blocking allowed
- Cache handles hits, schedules misses to load in background
- Resident pages managed independent of disk cache
- Physical pages organized as quad-tree per virtual texture
- Linked lists for free, LRU, and locked pages

Virtual Texturing - Feedback

- Feedback Analysis
  - Gen ~breadth-first quad-tree order w/ priority

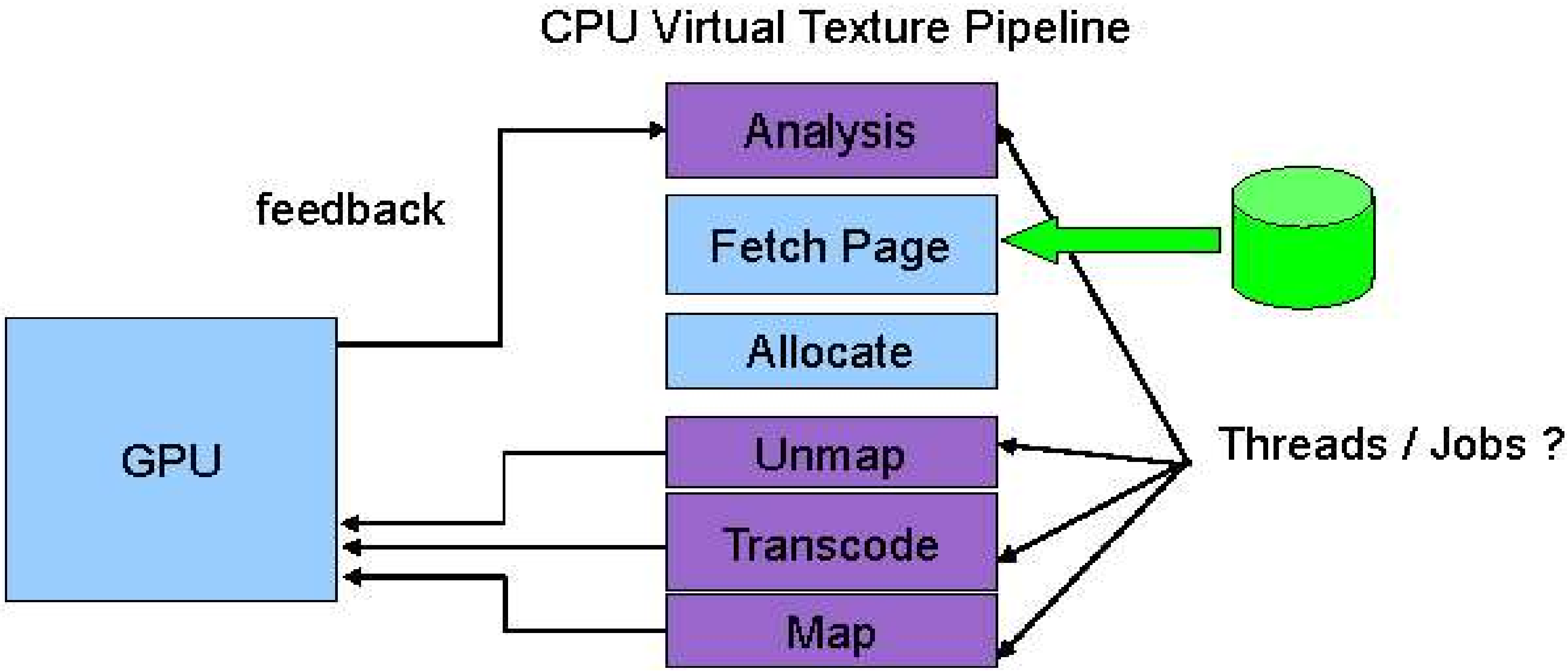
Color Buffer

Feedback Buffer



Virtual Texturing - Pipeline

- Compute intensive complex system with dependencies that we want to run in parallel on all the different platforms



Virtual Texturing

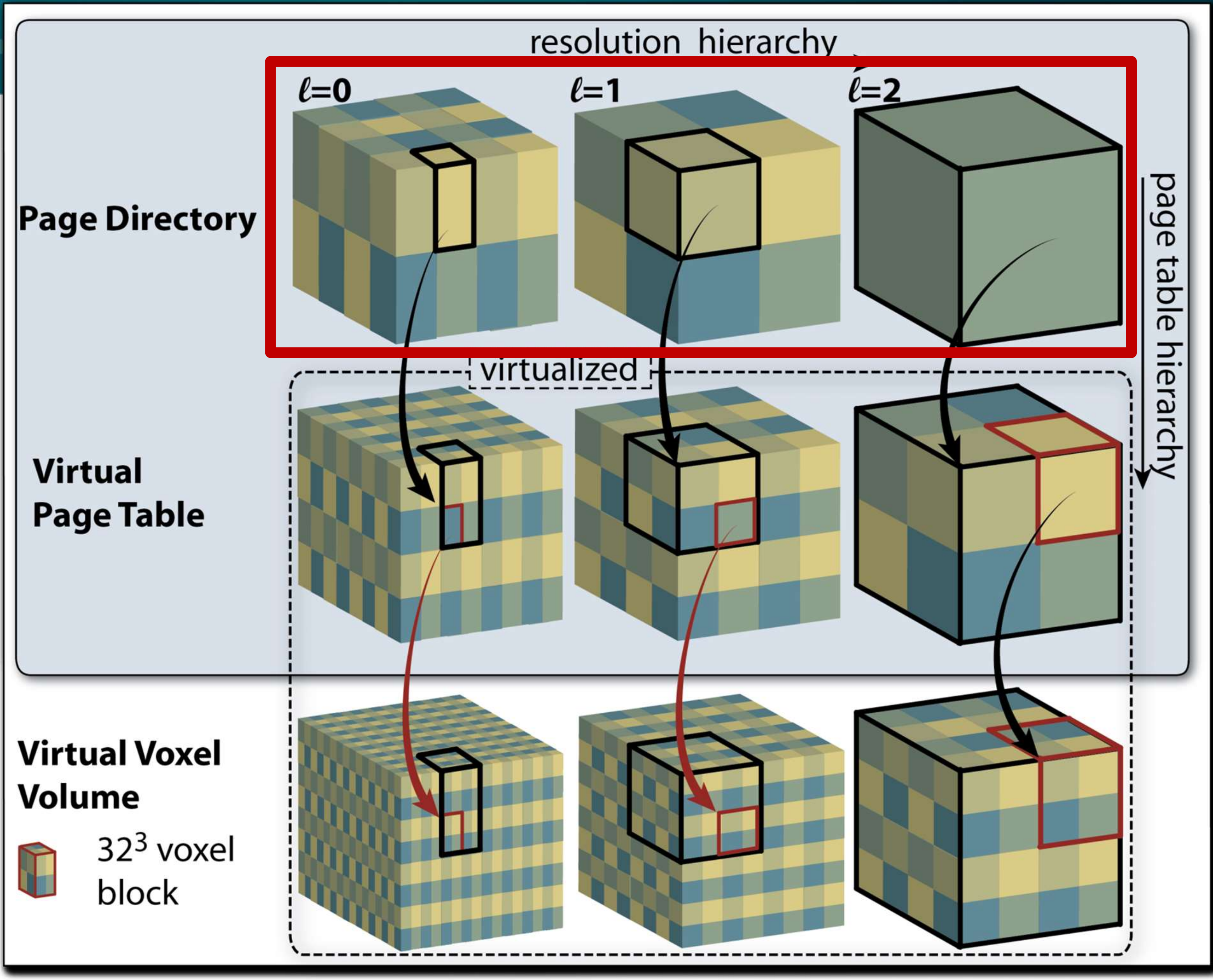
Example #4:

Petascale Volume Rendering

- Interactive Volume Exploration of Petascale Microscopy Data Streams Using a Visualization-Driven Virtual Memory Approach,
  Hadwiger et al., IEEE SciVis 2012

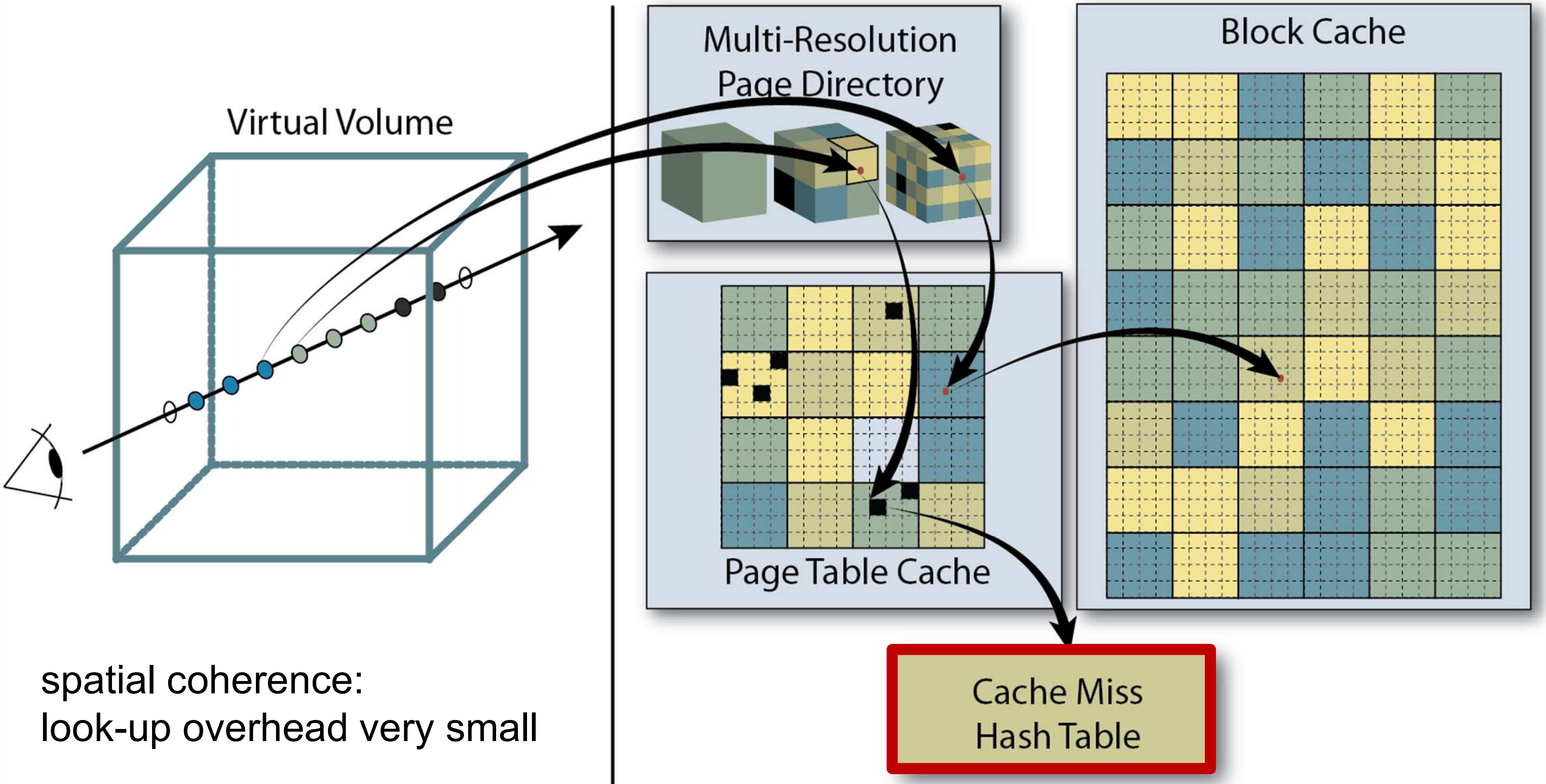
`http://dx.doi.org/10.1109/TVCG.2012.240`

Petascale Volume Rendering



multi-resolution page directory

Petascale Volume Rendering



spatial coherence:
look-up overhead very small

Thank you.