Future graphics in games

Anton Kaplanyan
Lead researcher

Cevat Yerli
Crytek CEO

Agenda

- The history: Crytek GmbH
  - Current graphics technologies
  - Stereoscopic rendering
  - Current graphics challenges
- Graphics of the future
  - Graphics technologies of the future
  - Server-side rendering
  - Hardware challenges
  - Perception-driven graphics

The Past - Part 1

- **March 2001 till March 2004**
  - Development of Far Cry
  - Development of CryEngine 1

- **Approach:** A naïve, but successful push for contrasts, by insisting on opposites to industry. size, quality, detail, brightness
  - First right investment into tools - WYSIWYPlay

Past - Part 1: CryEngine 1

- Polybump (2001)
  - NormalMap extraction from High-Res Geometry
- First „Per Pixel Shading“ & HDR Engine
  - For Lights, Shadows & Materials
  - High Dynamic Range
- Long view distances & detailed vistas
  - Terrain featured unique base-texturing
- High quality close ranges
- High fidelity physics & AI
- It took 3 years, avg 20 R&D Engineers

CRYTEK

CryENGINE 2

The Past – Part 2 – CryENGINE 2

- **April 2004 till November 2007**
  - Development of Crysis
  - Development of CryEngine 2
- **Approach:** Photorealism meets interactivity!
  - Typically mutual exclusive directions
  - Realtime productivity with WYSIWYPlay
  - Extremely challenging, but successful ☺

CryEngine 2 - Way to Photorealism

The Past - Part 2: CryEngine 2

- CGI Quality Lighting & Shading
- Life-like characters
- Scaleable architecture in
  - Both content and pipeline
  - Technologies and **assets** allow various configurations to be maxed out!
  - Crysis shipped Nov 2007, works on PCs of 2004 till today and for future... ☺

The Present - CryEngine 3

- CryEngine 3 is build with next-gen in mind
- Scales through many-core support
- Performs on PC, Xbox360, PS3, DX11
- Built by avg. 25 people over 3 years

CryENGINE 3 architecture

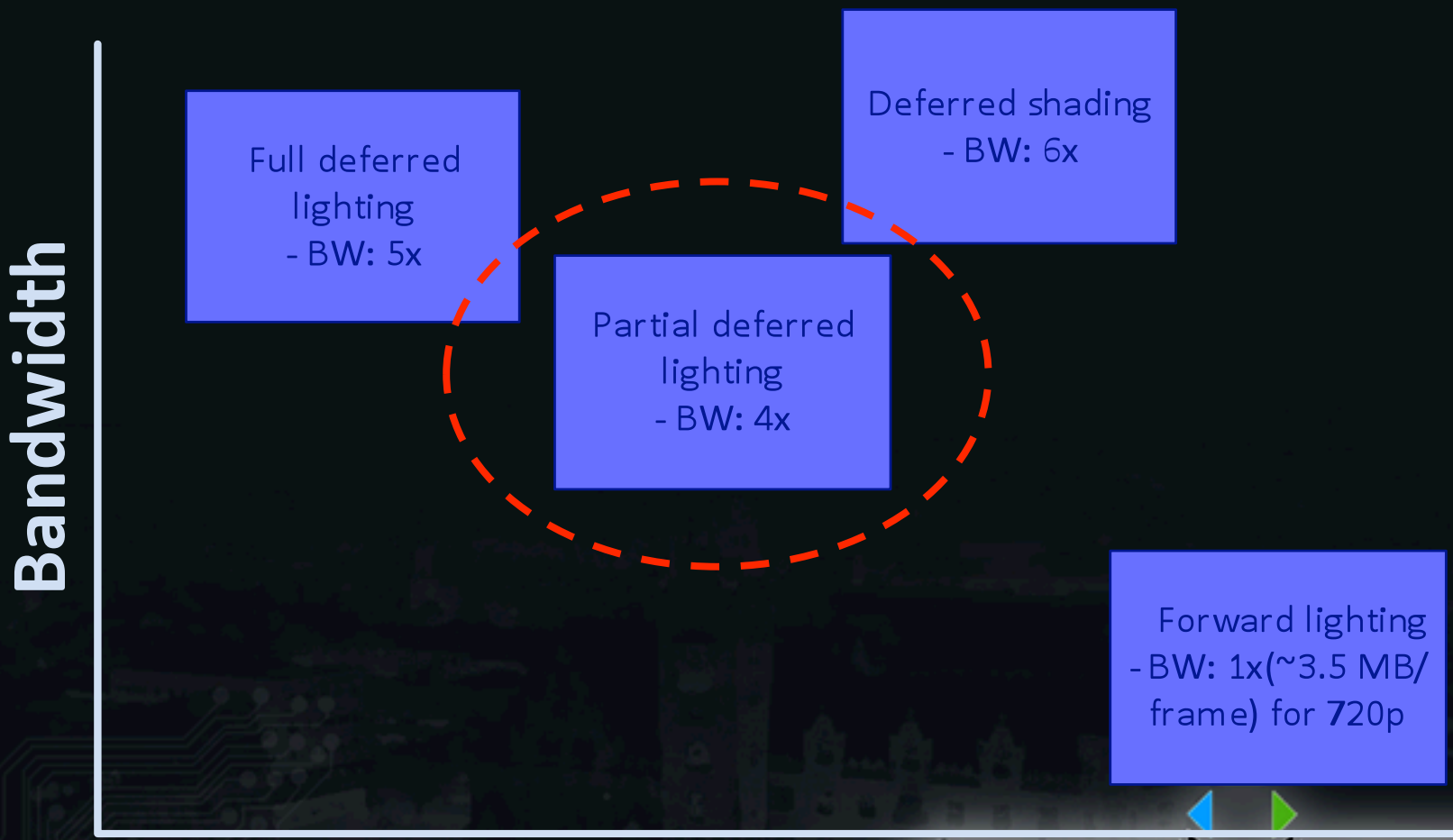
- CryENGINE 2 successor, but now we do
  - Deferred lighting (aka Light Prepass)
  - Lighting in linear space
  - Indirect lighting
  - Coordinated dynamic and precomputed lighting
  - Advanced color correction (artists-driven color charts)
  - Streaming rendering assets (geometry, textures, animation)
  - Run on both consoles (Xbox 360 and Playstation 3)
  - Compressed and minimized bandwidth and memory requirements

Why deferred lighting

Deferred lighting

- Good decomposition of lighting
  - No lighting-geometry interdependency
- Cons:
  - Limited material variations
  - Higher memory and bandwidth requirements
  - Shading problems
    - 2x2 tiles for mip computation fail for any kind of deferred texturing (projective light textures, decals etc.)

Deferred pipelines bandwidth



Feature

How to design for the future?

- Facts
  - Fixed Resolution for Gaming till 2012
  - HD 1920 x 1080 @ 60 fps
    - Stereoscopic 3D experience: 30 fps per eye
  - Limited by current consoles hardware
  - Risk of „Uncanny Valley“ for content
    - Perception-driven approaches!
  - Till 2012 majority of games must use artistic style, physics and AI to differentiate!
    - What‘s the current artistic style? Desaturate colors?

Graphics architecture

- Breakthroughs in rendering architecture are not easy
  - Proved multiple times by hardware vendors
    - **Especially multiple recent tries with software renderer**
  - Trails along with a huge infrastructure
    - Outcome of a many-years development experience
- Graphics architecture will be much more **divergent**
  - Do we really want to write our own software renderer?
  - Coming back to old good techniques like voxels, micropolygons etc.

How to design for the future?

- Alternatives that will brand some games in future:
  - Point Based Rendering
  - Ray Tracing
  - Rasterization, as usual
  - Micropolygons
- Data representations:
  - Sparse Voxel Octrees (data structure)
  - Sparse Surfel Octrees

Graphics in Future

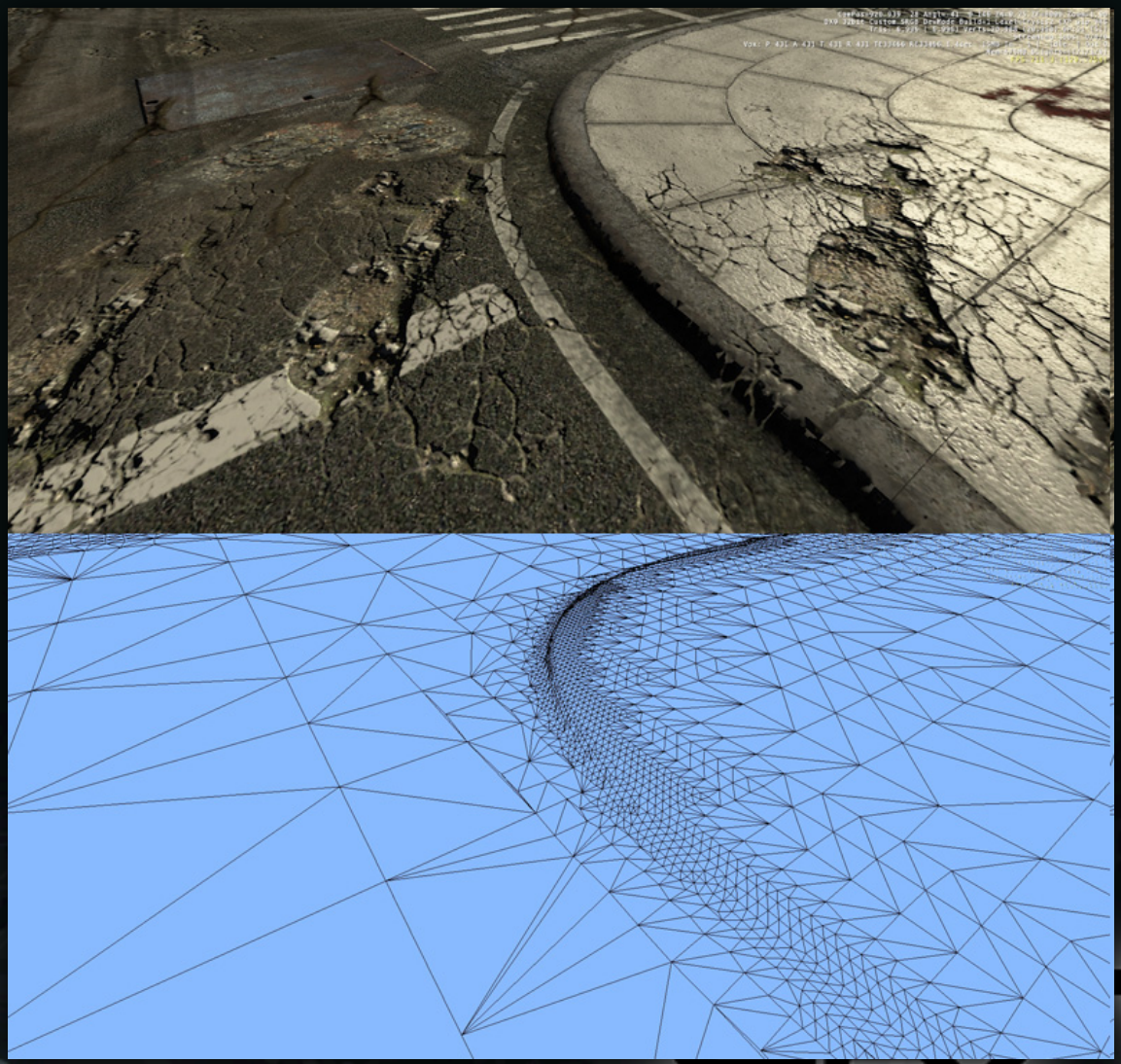
- Sparse Voxel Octrees (Datastructure)
  - Pros
    - Data structure is future proof for alternative rendering
    - Very good fit for unique geometry & texture
      - Geometry and texture budgets become less relevant
    - Artistic freedom becomes true
    - Naturally fits to automatic LOD schemes
  - Cons
    - Neither infrastructure nor h/w
    - Slightly memory intensive
    - Fits nicely to Ray-tracing, but is still too slow

Sparse Voxel Octree Usage in CryEngine 3

- We already use it in production!!
  - Used during level export to bake geometry and textures
  - Stored in a sparse octree of triangulated sectors
  - Very easy to manage and stream geometry and textures
  - No GPU computations required (despite virtual texturing)
  - Automatic correct LOD construction
  - Adaptive geometric and texture details
    - Depending on the gameplay
- Huge space on disk for each level!
  - Use aggressive texture compression
  - Bake wisely, not the whole world

CRYTEK

Sparse Voxel Octree Usage in CryEngine 3



Opportunities in Future

- Short-term user impact opportunities till 2012
  - The delta in visual opportunities is limited, BUT...
  - for the next 3 Years: Huge gains are possible in Physics, AI and Simulation of Special Effects
    - ➔ Focus around that knowledge can lead to very different designs
- Mid-erm 2013+ creative opportunities
  - Future console generations
    - ➔ New Rendering Methods will become available
    - ➔ The renaissance of graphics will arrive
  - Allows new visual development directions that will rival full CGI feature films qualty
  - Action point: Link yourself to console cycle

PERCEPTION-DRIVEN GRAPHICS

Perception-driven graphics

- PCF-based soft shadows
- Stochastic OIT
- Image-based reflections
- Ambient Occlusion (SSAO, prebaked etc.)
- Most posteffect (DoF, motion blur approximations)
- Light propagation volumes
- Many stochastic algorithms
- most of assumptions in real-time graphics
- *All that works because of the limited human perception*

Real-time graphics *is* perception-driven

- Human‘s eye has some specialities
- ~350 Mpixel spatial resolution
  - Quite hard to trick it in this area
- ~24 Hz temporal resolution
  - **Very** low, a room for techniques
  - We don‘t notice the flickering @ **> 40Hz**
- *We don‘t create an image for another machine, our target customer is a human*

Under-sampling / super-sampling

- Spatial
  - Undersampling
    - Inferred shading
  - Depth of field
    - Decoupled sampling
- Temporal
  - Temporal anti-aliasing
  - Motion blur
- Mixed
  - Spatio-temporal anti-aliasing

Hybrid rendering

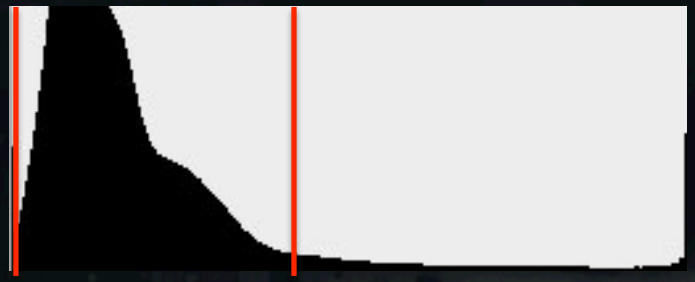
- There is no panacea rendering pipeline
  - Even REYES is not used in its original form for movies
- Hybrid pipeline is possible on the current gen GPUs
  - Will be even more topical for new generation of consoles
- Usually combines everything that matches and helps
  - Ray-tracing for reflections and shadows
    - Could be triangles / point sets / voxel structures / etc.
  - Voxels for better scene representation (partially)
  - Screen-space contact effects (e.g. reflections)
  - Much much more (a lot of ideas)

Recent trend

STEREOSCOPIC RENDERING

3D stereoscopic rendering

- Technique was there for a long time
  - Becomes popular due to technologies, in games too
  - No new concepts, similar to photography art though
    - One golden rule: don't make the audience tired
- Crysis 2 already has a 1st class 3D Stereo support
  - Use the depth histogram to determine the interaxial distance:



CRYTEK

Supported stereo modes in CryENGINE 3

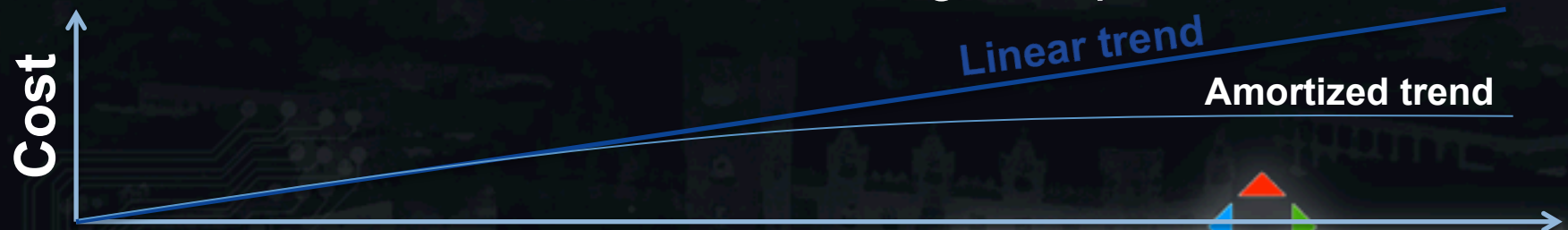
- Stereo rendering modes
  - Brute-force stereo rendering
  - Central eye frame with reprojection
  - Experimental stochastic rendering from one of eyes
- Stereo output modes
  - Anaglyph (color separation)
  - Interlaced
  - Horizontal joint images
  - Vertical joint images
  - Two monitors

Stereo video

SERVER-SIDE RENDERING

Server-side rendering

- 4G networks have a good ground for that
  - Low ping – a strong requirement for real-time games
  - Will be widely deployed in 5-7 years
- Compression of synthesized video
  - Temporally decompose the video details
  - Use perception-based importance
    - Salience maps + user-side eye-tracking
- Need to amortize cloud-rendering cost per user:



Example of perception-driven graphics

Image

Per-object importance map

Saliency map

Courtesy of Matthias Bernhard
TU Vienna

- Example of perception-driven rendering
  - They use eye-tracking system to build importance map
    - Can be provided by the game itself
- Adaptive video compression is possible along with adaptive rendering

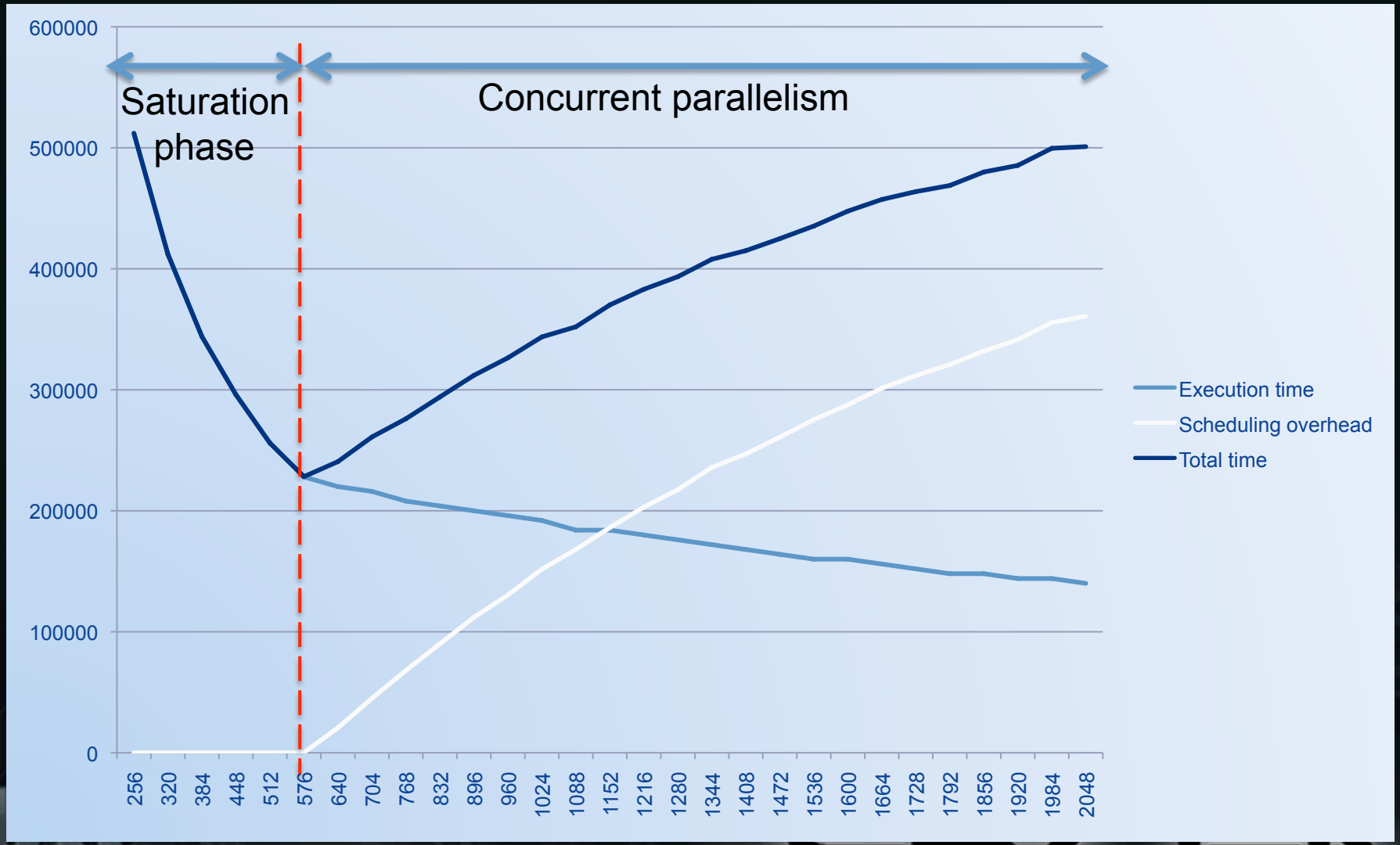
CRYTEK

CURRENT PROBLEMS OF HARDWARE ARCHITECTURE

Highly parallel scheduling

- Small synthetic test (simulate GPU behavior)
  - 512 cores (could be interpreted as slots of shared cache too)
  - 32k small identical tasks to execute
    - Each item requires 1 clock on one core (so synthetic)
  - Within a range of 256 to 2048 threads
  - Scheduling overhead is taken into account in total time
    - Task feeding
    - Context switches
    - Overhead weight is not important

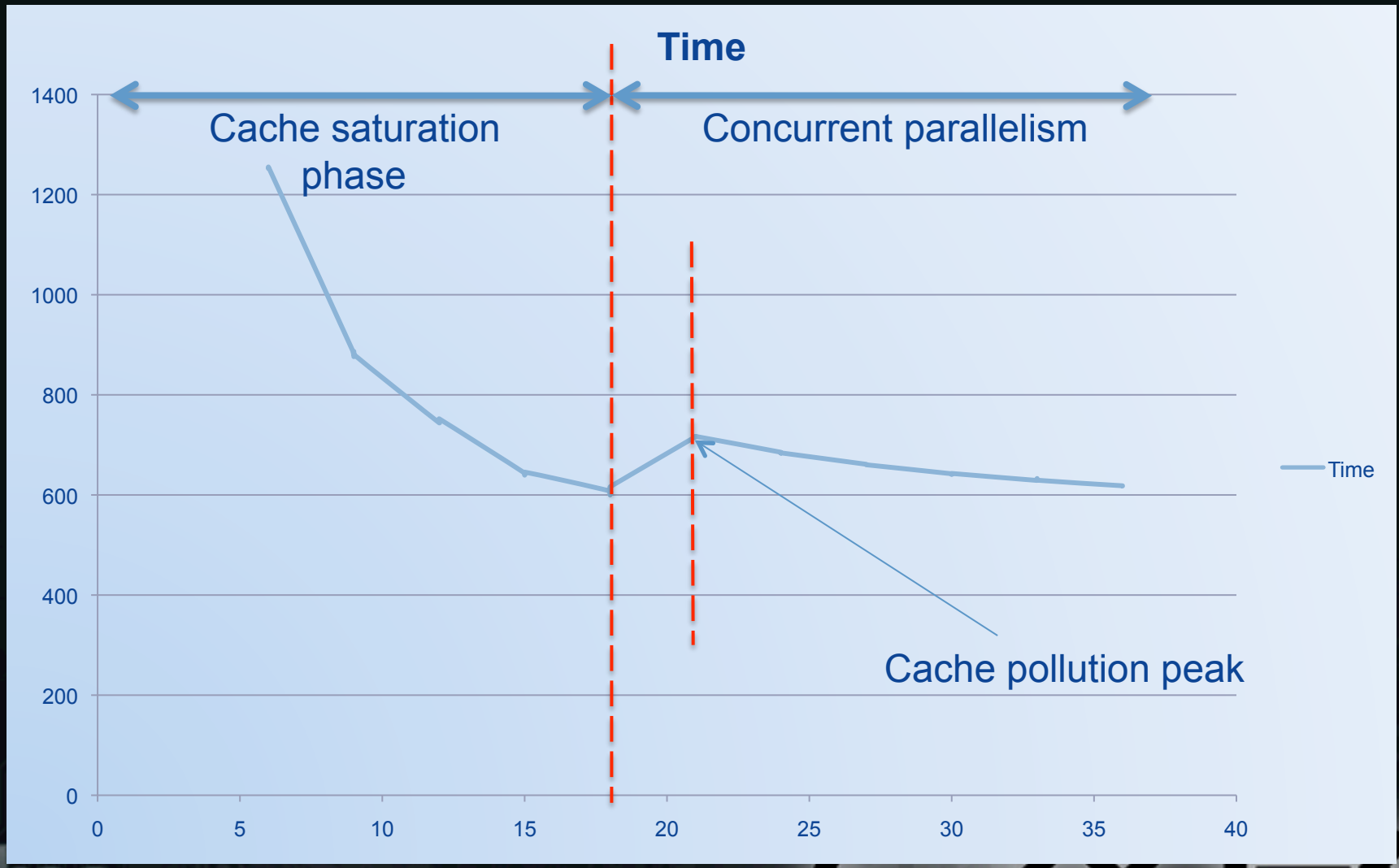
Highly parallel scheduling



Highly parallel scheduling

- Another test
  - Real GPU!
  - Screen-space effect (SSAO)
  - Bandwidth-intensive pixel shader
    - Each item requires 1 clock on one core (so synthetic)
  - Within a range of 5 to 40 threads
  - Cache pollution causes a peak right after the saturation state
    - The time reaches the saturation performance with more threads asymptotically

Highly parallel scheduling



Highly parallel scheduling

- Scheduling overhead can be a problem
  - Parallel scalability
  - With homogenous tasks it comes to maximum at saturation
    - What about heterogeneous workload?
    - The existence of the minimum depends on the performance impact of scheduling
  - We need to reduce it
    - **Configurable hardware scheduler!**
      - GRAMPS-like architectures are possible with it
      - Ray tracing becomes much faster and SoL with bandwidth bottleneck

Atomics

- Atomics came from CPU hardware
  - Used to build synchronization primitives in Oses
  - Works only on integers
  - Provides result of operation
- We need absolutely **different** atomics!!!
  - We use it mostly for gather/scatter operations
  - MUST work on floating point numbers instead!
  - In most cases no result needed
    - Improve atomics w/o read-back (fire-and-forget concept)
    - Operation should be done on memory controller / smart memory side
  - We need order of magnitude faster performance for **graphics atomics**

Future performance

- PS3 and Xbox 360 are in-order
  - “by optimizing for consoles we also speed up PC”
  - not really, we invest only into current consoles
- What’s the next generation of consoles?
  - Larrabee 2 and Fermi ARE in-order
  - Should we rewrite the architecture again?
- Death of Out-of-Order architecture?
  - No way! Game platform will remain **heterogeneous**
    - **Related to different game subsystems (game code vs rendering)**
- Many new parallel languages and paradigms
  - **OpenCL, GRAMPS, C++0x, OpenMP, TBB, ConcRT, Ct**
- **Backwards scalability is a challenge**

Future performance

- Mostly graphics, as it's scalable without pain
  - Doesn't affect game-play
- Assets processing
  - Texture compression becomes an issue as well
    - Both decompression **AND** **compression** complexity should be respected
  - Shaders development
    - Compilation is too slow and not flexible
      - Still not solved by DX11 DSL
      - Getting worse with ComputeShaders
    - Debugging / profiling is still not there for compute shaders
    - Developing a huge system might become a hell

Textures

- Quantization / color depth?
  - BC6/7 delivers, but DX11-hw only

DXT1-compressed

Original

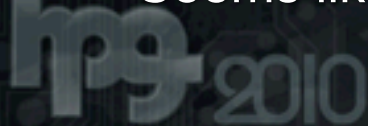
Challenges of Future

- **Technology challenges**
  - Switching to a scaleable codebase
    - Think of parallelism & async jobs
    - Multithreading, scheduling
    - Larger codebases, multiple platforms & APIs
- **Production challenges**
  - Cost of assets increase by ~50% annually
    - Content, besides quality increases, gets more & more „interactive“
    - Think to improve Tools, Pipelines & Bottlenecks to counter-effect , automate Source Back-Ends ➔ Resource Compilers
    - The better the tools, the cheaper and/or the better your output

Efficiency

- We spend too much of computational power per frame!
  - Precision is mostly redundant
    - No need to compute colors in 32-bits floating points
    - Even h/w rasterizers was 12-bits of fixed precision in good old times
  - Humans do not notice the most of the picture in real-time graphics
    - It is a gameplay video rather than a still image
    - Neither we watch it like a movie, games are usually challenging
    - The importance of a particular technology is perception-driven

      How important are the fully accurate rather very glossy reflections
- Graphics hardware should challenge incoherent workloads
- What about profit / development cost ratio?
  - Seems like we already fall into **uncanny valley** in graphics technologies



Scopes

- Content costs will increase
  - If nothing changes ➔ Tools must adapt
  - Smarter & automated pipelines & tools will provide better, faster & valid content data
  - Think procedural content creation
- 5y...gaming graphics will change,
  - but insignificantly in the next 3 years
- Today‘s technologies will drive the next 3 years in visual development. The look is still about creativity and using the given resource powers of today
- 5y...realtime gaming graphics will approach to current CGI offline rendering

Conclusion

- Real-time rendering pipeline renovation is around the corner
  - Hardware improvements are required
  - Evolution of current techniques for production real-time rendering
    - Prepare to new representations and rendering pipelines
  - Better infrastructure for parallel development
  - Tools and authoring pipelines needs modernization
  - Consider **server-side rendering**: could change the direction drastically
- Perception-driven real-time graphics is a technology driver
  - Avoid uncanny valley in graphics technologies


Questions?
Cevat@Crytek.de
AntonK@Crytek.de