
SIGGRAPH2007

Finding Next Gen CryEngine2

SIGGRAPH2007

Martin Mittring

Lead Graphics Programmer

Crytek GmbH

Far Cry

Shipped March 2003
Publisher: Ubisoft
Platform: PC

Crysis

Not released yet
Publisher: Electronic Arts
Platform: PC

Introduction

- Far Cry (CryEngine 1) shipped March 2003:
  - Outdoor with long view range, paradise look
  - Gameplay: AI, physics, non linear, explore
  - Technology: Per pixel shading, realtime shadows, HDR, up to shader model 2.x/3.0

… we wanted to push the technical limits again

Goals (1/2): three different environments

- Jungle paradise
  - Many objects, height map, ocean, view distance, sky and sun light

- Alien indoor
  - Many point lights, dark, huge room like sections, geometry occlusion, fog volumes

- Ice environment
  - Ice Material layer, subsurface scattering, Ambient lighting

Goals (2/2):

- Cinematographic quality rendering
- Dynamic light and shadows
- MGPU & MCPU
- Game design requested 21km x 21km
- Target GPU from SM 20 to SM 40 (DirectX10)
- High Dynamic Range
- Dynamic environment (breakable)
- Developing game and engine together

Concept Phase

Blur Studio www.blur.com

Overview

- Shaders and shading
  - Shader combinations problem, 3Dc™ for normal maps, Per pixel scene depth, World Space Shading
- Direct Lighting and Indirect Lighting
  - Direct: SM, soft SM, Shadow mask, deferred, VSM
  - Indirect: 3D Transport Sampler, RAM, SSAO
- Level of Detail
  - Dissolve
  - Ocean

Shaders and shading

Shaders and shading
CryEngine 1

- VS and PS up to SM 30
- Fallback: T&L and register combiner
- Über-shader with many #ifdef for specializations
  (multiple lights, fog, shadow, CM reflection, bump, …)
- Shader cache manually created (NVIDIA & ATI)
- DirectX and unsupported OpenGL
- Supported HLSL and CG
- Functionality like FX techniques

Shaders and shading CryEngine 2 (1/3)

- Shader combinations problem
  - dynamic branching
  - reducing combinations and accepting less functionality and less performance
  - separating into multiple passes
  - Distributed Job System to compile the shader cache
- 3Dc™ for normal maps
  - reconstruct z in pixel shader
  - TIF to DDS
  - DXT5 fallback, BC5 for Direct3D10 (XY swap in shader)

Shaders and shading CryEngine 2 (2/3)

- Per pixel scene depth
  - Early z pass (more draw calls)
  - Format: R16,R16G16, R32 or native
  - WS position [with offset] in 1 instruction
  - Use: Shadow Mask, Global Fog, Volumetric Fog, Soft Z Buffered Particles, DOF, Motion Blur, Beach/ Ocean, EdgeAA, Sun Rays

Shaders and shading CryEngine 2 (3/3)

- World Space vs. Tangent Space Shading
  - better shading quality
  - less interpolators (Pos, Tangent, Binormal, Orientation) with multiple lights
  - unified shader (like Cubemap reflection)
  - per scene constants instead of per object constants
  - longer pixel shader
  - Less appropriate for low spec

Direct Lighting

Shadowing Approach in CryEngine 1

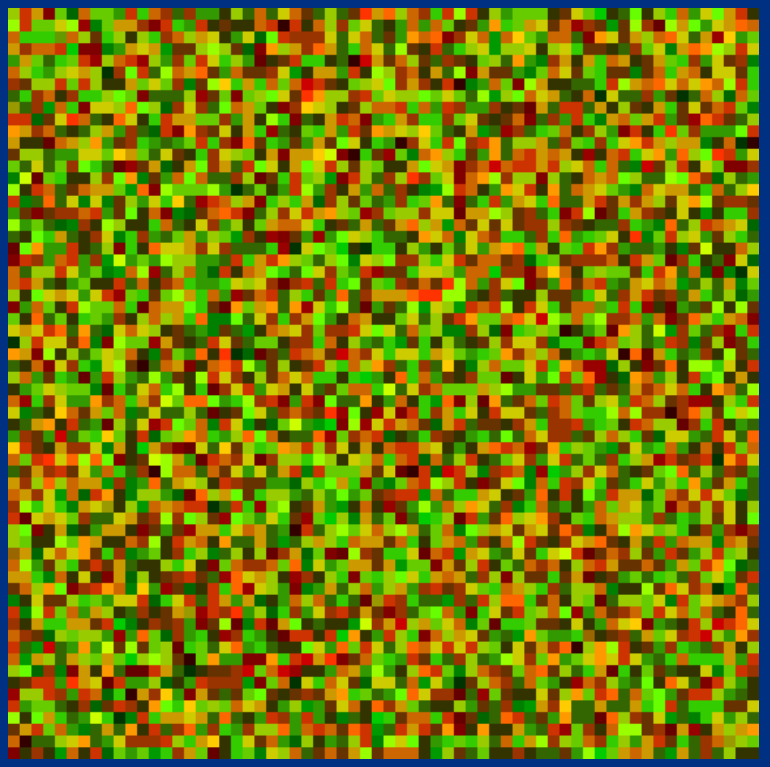
- Shadow maps and projected shadows for sun
- PCF if available, some aliasing
- Blurry precomputed vegetation shadows
- Stencil shadows for point lights (CPU performance)
- Dot3Lightmaps (normal mapped Lightmaps) patent, efficient and simple for static diffuse case like a lightmap, stores: avg. light direction (in TS) and colour with blend value to blend between diffuse and ambient case

Experiments after Far Cry

- Static occlusion maps
  - Like light maps but texture channels store light occlusion
  - per pixel lighting including specular for static geometry
  - slower than light maps but faster than dynamic solutions
- Dynamic occlusion maps
  - updated with shadow map render2texture (dilation)
  - double aliasing and limited resolution
- Static occlusion maps with shadow maps
- Limited success with adjusted penumbra size

The plan for CryEngine 2

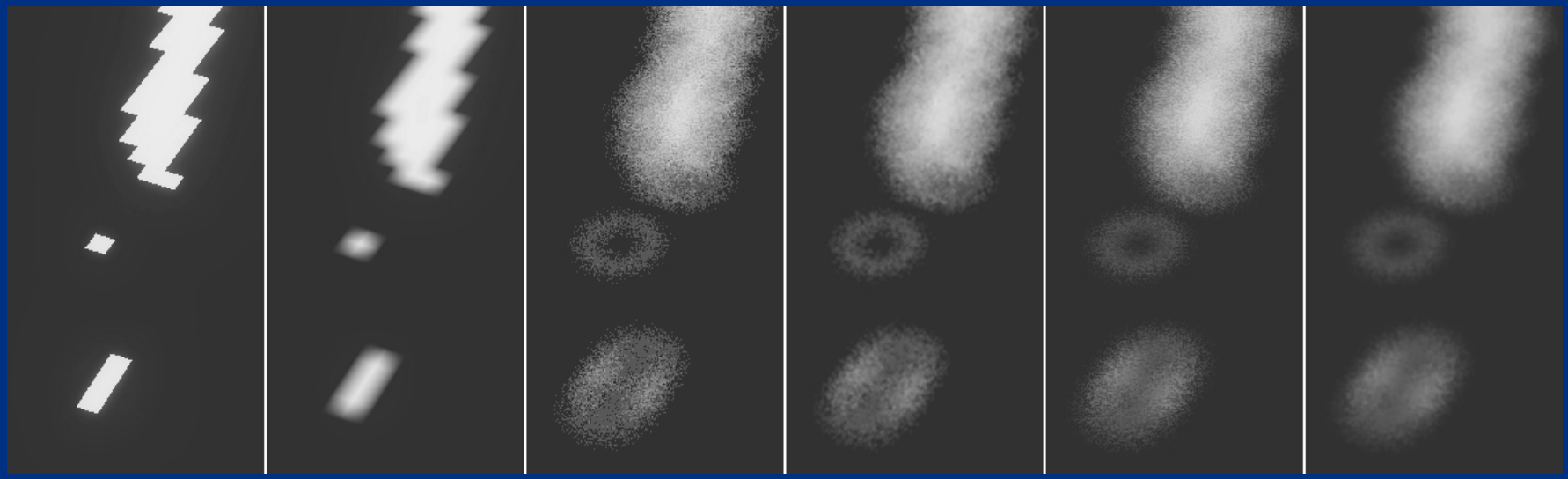
- clean unified shadow system
- drop stencil shadows
- Approach direct and indirect lighting with specialized solutions
- Shadow maps with multiple lookups



Random 2D noise texture

Screen space randomized lookup

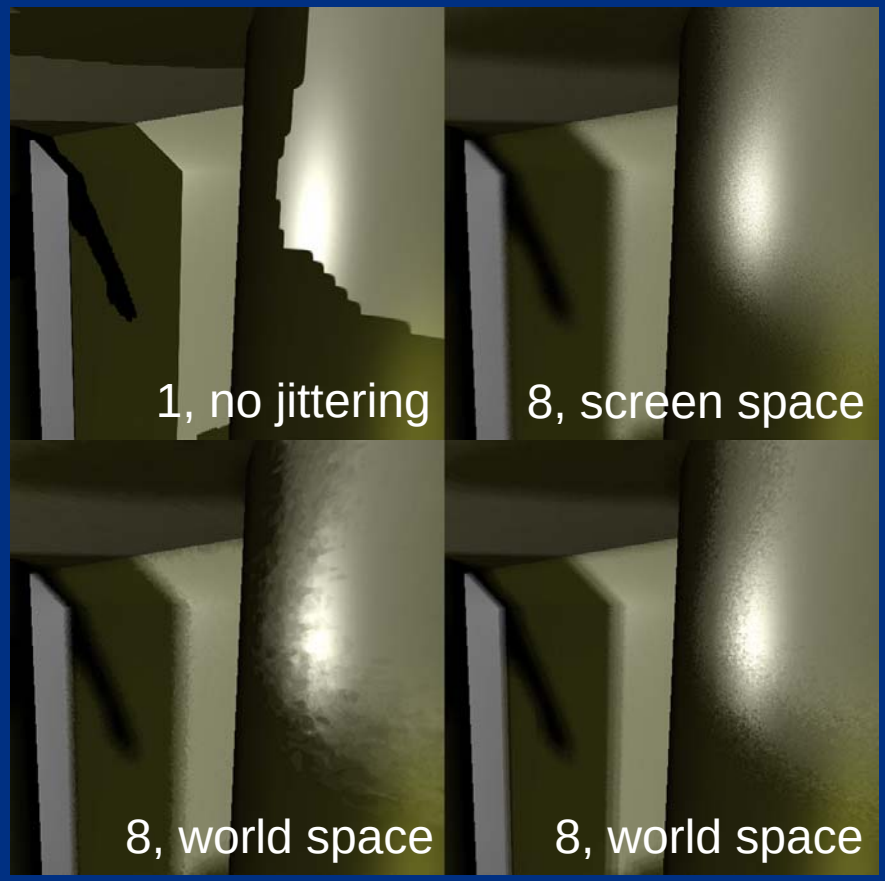
- 2D: rotated disk samples (random 2d base)
- 2D: construct samples (random 2d vectors)
- 3D: plane mirroring (random normals)



plain, PCF, 8 samples, 8 samples+blur, PCF+8 samples, PCF+8 samples+blur

Random Texture projection

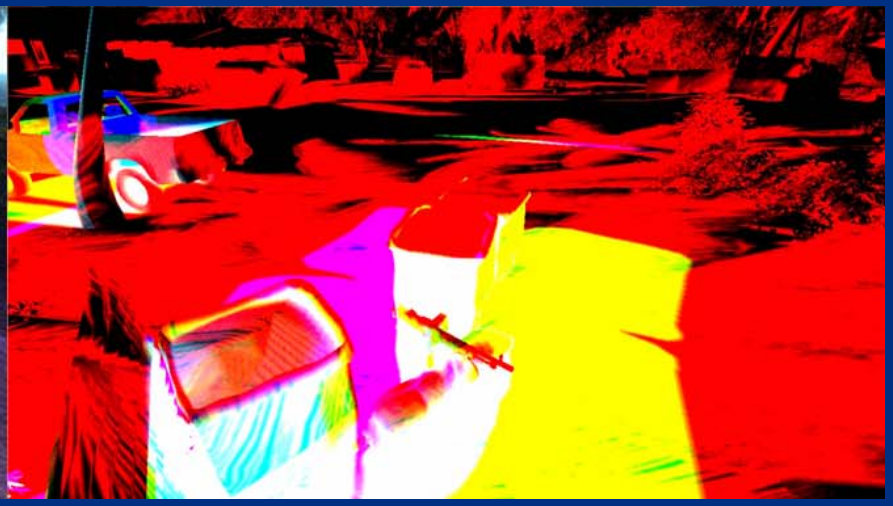
- Screen space
  - Tiled 64x64, no mips
  - Animated noise ?
- Light space
  - Tiled 256x256, mipped
  - Moving with camera
  - Snapping for stable results
  - Filtered



Shadow mask texture

- One ARGB texture allows up to four lights per pixel
- Separate shadow from shading
- Multiple shadow casting lights per pass

final rendering with sun
and two shadow-casting lights

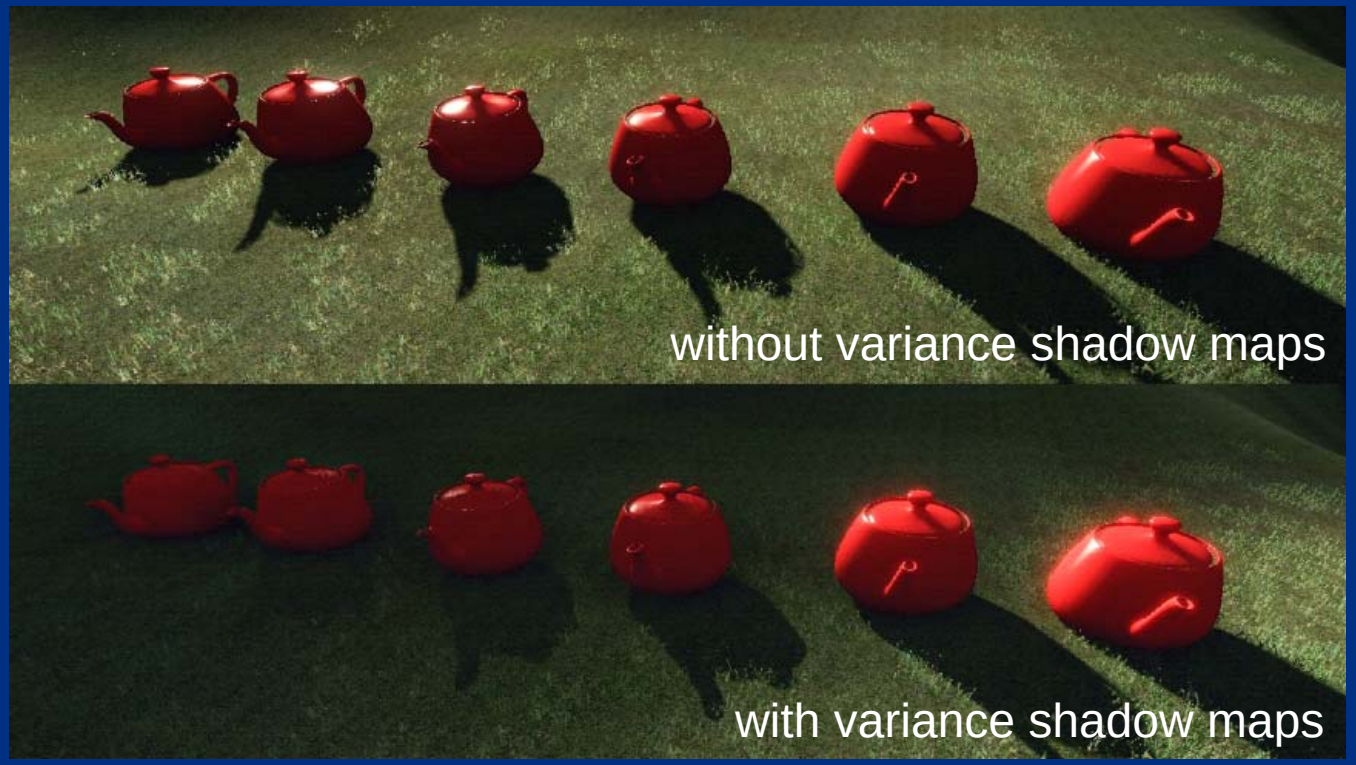


light mask texture with three lights
in the RGB channels

Deferred shadow mask generation

- Opaque only (including alpha test)
- Based on EarlyZ pass, less draw calls
- Lookup center position in 1 instruction
- Sun: 2-4 parallel shadowmap slices
- Omni light: 6 shadowmaps pyramids
- Shadow mask generation restricted to volumes marked in the stencil buffer

Variance shadow maps



Variance shadow maps

- Inexpensive very soft shadows
- Leaking with multiple casters in front of each other
- Suits the terrain shadows properties
- Alternatives are less preverable
- Only terrain is casting VSM
- Deferred applied to the shadow mask

Indirect Lighting (3DTS, RAM, SSAO)

Plan: 3D Transport Sampler

- A tool to compute Global Illumination data
- MultiCPU, distributed, extendable
- Implementation: Photon mapper
- Unwrapping based on existing UV mapping
- Output:
  - Lightmap, [Dot3Lightmap]
  - Lightmap with 4 normals oriented to the surface
  - RAM

Real time ambient maps (1/2)

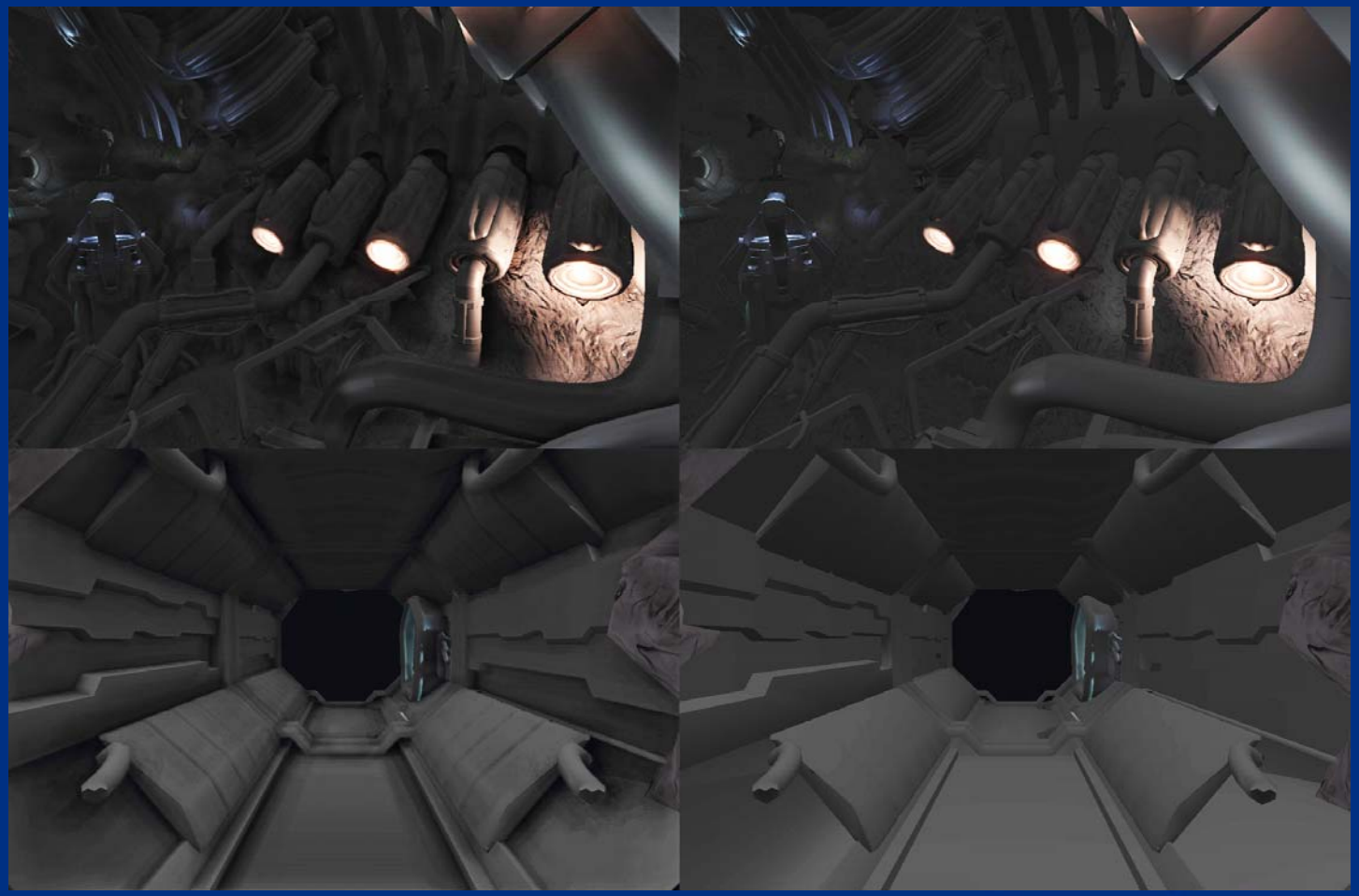
Real time ambient maps (2/2)

- One scalar ambient occlusion value per texel
- Reconstruction in the shader based on:
  - texel with the occlusion value
  - light color, relative position and attenuation radius
  - surface normal
  - Artist tweakable values (bounce color, radius)
- Normal map, portals, multiple lights combined
- Drawbacks: preprocess, static, resolution, memory

Screen Space Ambient Occlusion (1/2)

- 3D sample distribution scaled with distance
- Samples tested against the depth buffer, far samples with less influence
- AO is a function of the number of exposed samples
- Per pixel rotation with 4x4 texture (reflect), Smart blurring (depth buffer)
- Fully dynamic, view dependecy rarely visible, texture cache trashing
- Results depend on asset tessellation and tweaking

Screen Space Ambient Occlusion (2/2)



Level of Detail

LOD (Level of Detail)

- Indoor requires good occlusion but
  dense outdoor environment requires LOD
- Objects need to look good in all distances
- Only PS scales nicely with distance

- Specialized Solutions:
  ocean, static LOD, vegetation sprites

Dissolve

- LOD/LOD LOD/Sprite transitions, Objects fading
- Per pixel dissolve in EarlyZ pass only,
  further passes use ZEqual
- Dissolve texture projected in screen space
- Texkill, alpha-test or A2C
- Post processing EdgeAA hides artifacts
- Transition phase should be finished quicky
  (for performance and look)

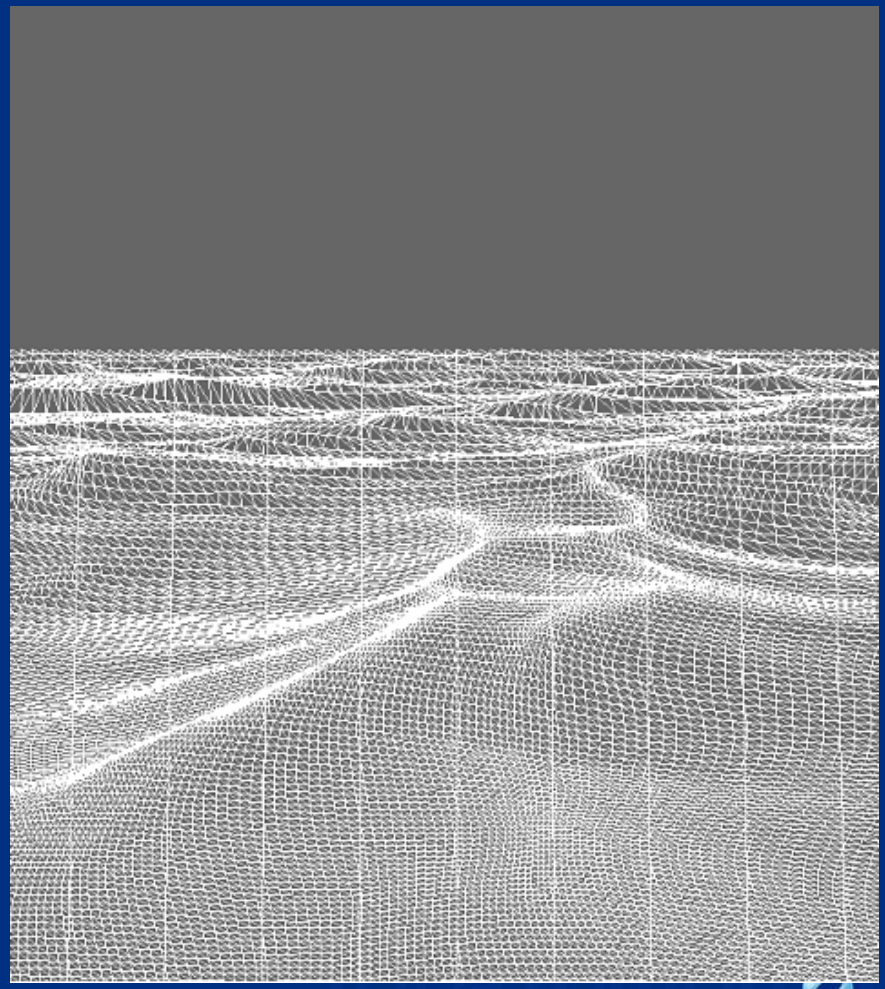
Water surface LOD

Square water sectors

- FFT based water animation for a small sector
- LODs with static IB but shared VB
- Sharing VB degrades quality because of aliasing
- Faded perturbation for distant vertices and limited the perturbation in the near
- Shoreline based on vertex animated static mesh
- Used in ATI tech demo „the Project“
- Drawbacks: CPU, aliasing, LOD popping

Screen-space tessellation (1/3)

- Brute force approach
- Screen space tessellated quad
- Water plane projection
- Perturbed in VS
- Vertex cache
- Z buffered



Screen-space tessellation (2/3)

- Vertices moved in XYZ
- Edge attenuation to fix the borders
- Also works with camera roll

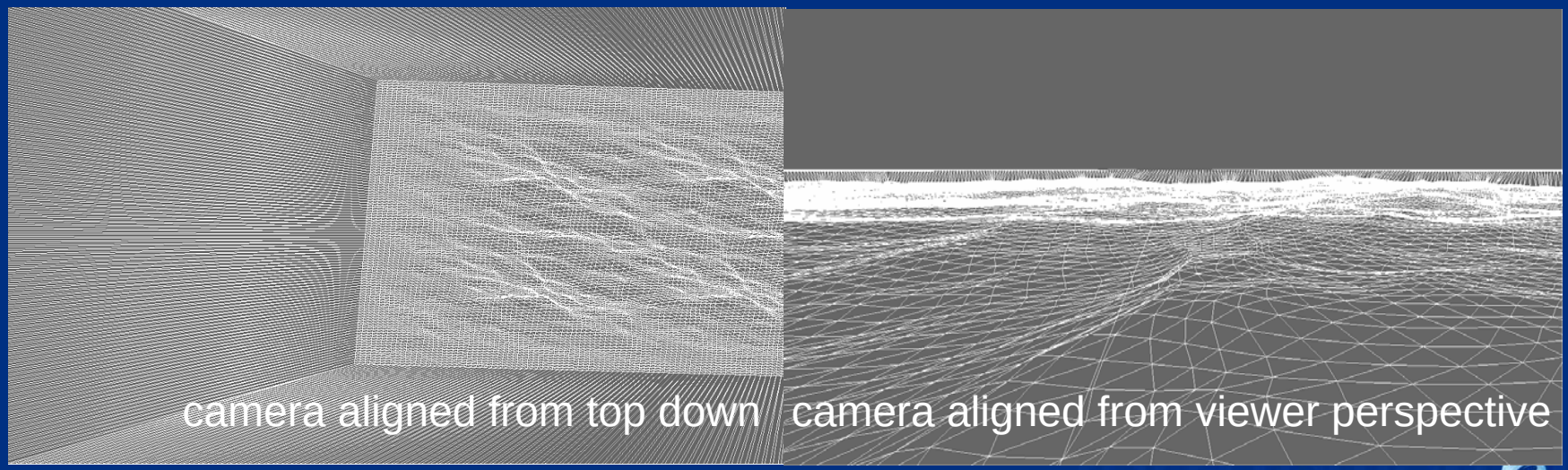
Screen-space tessellation (3/3)

- Lower tessellation to adjust quality/speed ratio
- FFT via vertex texture (if available)
- Shading: procedural shore, fog, shadow receiving, RGB independent refraction, Caustics

- Drawbacks: aliasing, view-dependent because of attenuation, no physics interaction

Camera aligned tessellation (1/2)

- Static mesh with top down projection
- Moving with camera (snapping to reduce aliasing)
- Extruded horizon border
- Attenuation to fade out perturbation before horizon



Camera aligned tessellation (2/2)

- Not view dependent (compare borders)
- Proper physics iteraction
- Less aliasing (compare horizon)
- Drawbacks: small triangles, vertex texture lookup

Conclusion

- Crysis will be shipped soon
- Leaving the path leads to new challenges
- Iterations are necessary
- More creation time for less improvements
- Big scale production
- Wide range of hardware means compromises
- Depth from EarlyZ good for current generation

Acknowledgements

- Based on the passionate work of many programmers, artist and designers
- Vladimir Kajalin, Andrey Khonich, Tiago Sousa, Carsten Wenzel and Nick Kasyan
- Intensive special support namely Miguel Sainz, Yury Uralsky and Philipp Gerasimov
- Additional thanks to Natasha Tatarchuk and Tim Parlett

Questions?

Martin@Crytek.de