Image-Space Horizon-Based Ambient Occlusion

SIGGRAPH2008

Louis Bavoil, Miguel Sainz

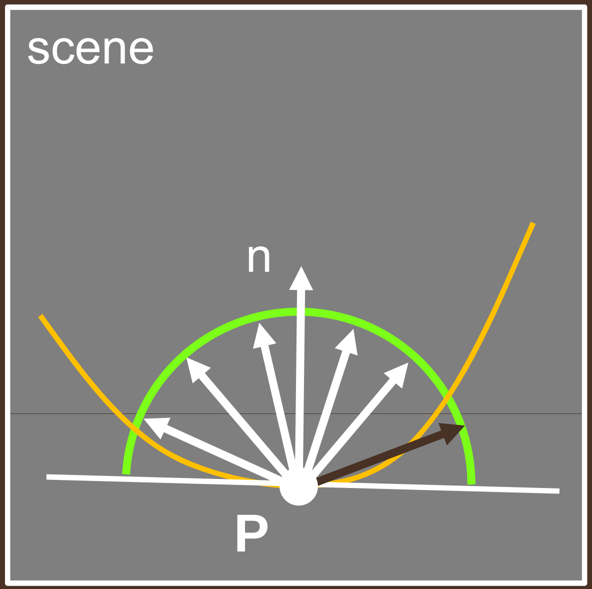
NVIDIA

Sky Light

- Simplest form of Ambient Occlusion
  - Light source = sky (sphere light)
  - Two definitions of AO
    - AO = diffuse illumination from the sky [Landis 02] [Christensen 03]
    - AO = shadow from the sky illumination [Pharr and Green 04] [Hegeman et al. 06]
- Limited to outdoor scenes

Ambient Occlusion

- Light = local hemisphere
  - Centered at current surface point
  - Radius = user parameter
- Can be rendered with ray-tracing
  - [Gelato] [Mental Ray]



local sphere light

Ambient Occlusion

- Gives perceptual clues of depth, curvature and spatial proximity

Without AO

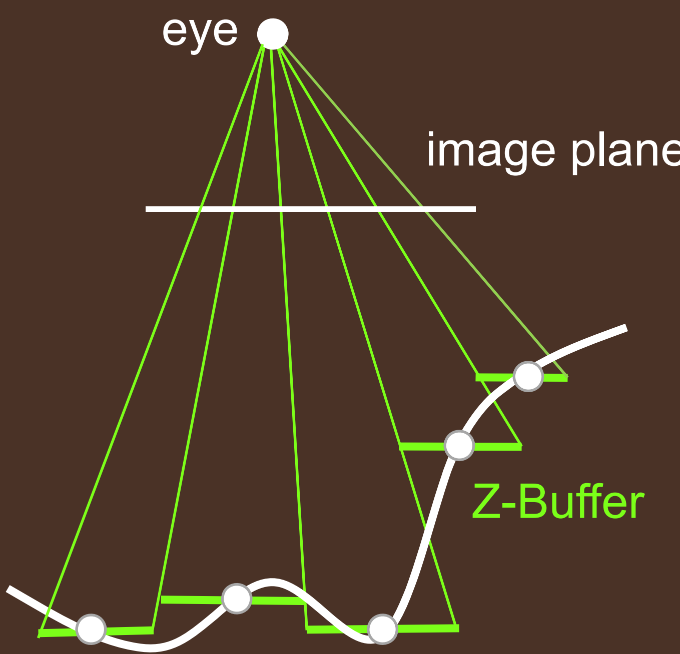
With our AO

Screen Space Ambient Occlusion

- Approach introduced by

  [Shanmugam and Orikan 07]
  [Mittring 07] [Fox and Compton 08]

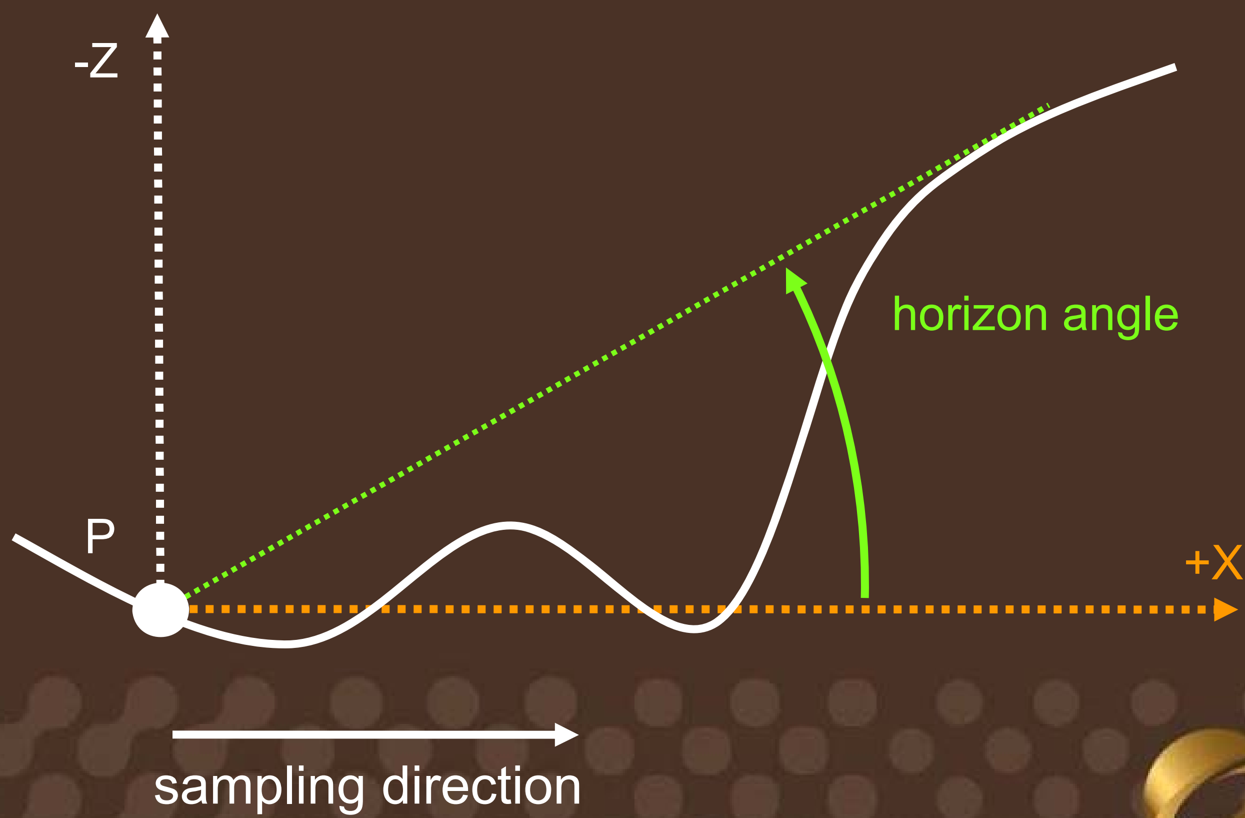
- Input = Z-Buffer + normals
  - Render approximate AO for dynamic scenes with no precomputations
- Z-Buffer = Heightfield
  - z = f(x,y)



SIGGRAPH2008

Horizon Mapping [Max 86]

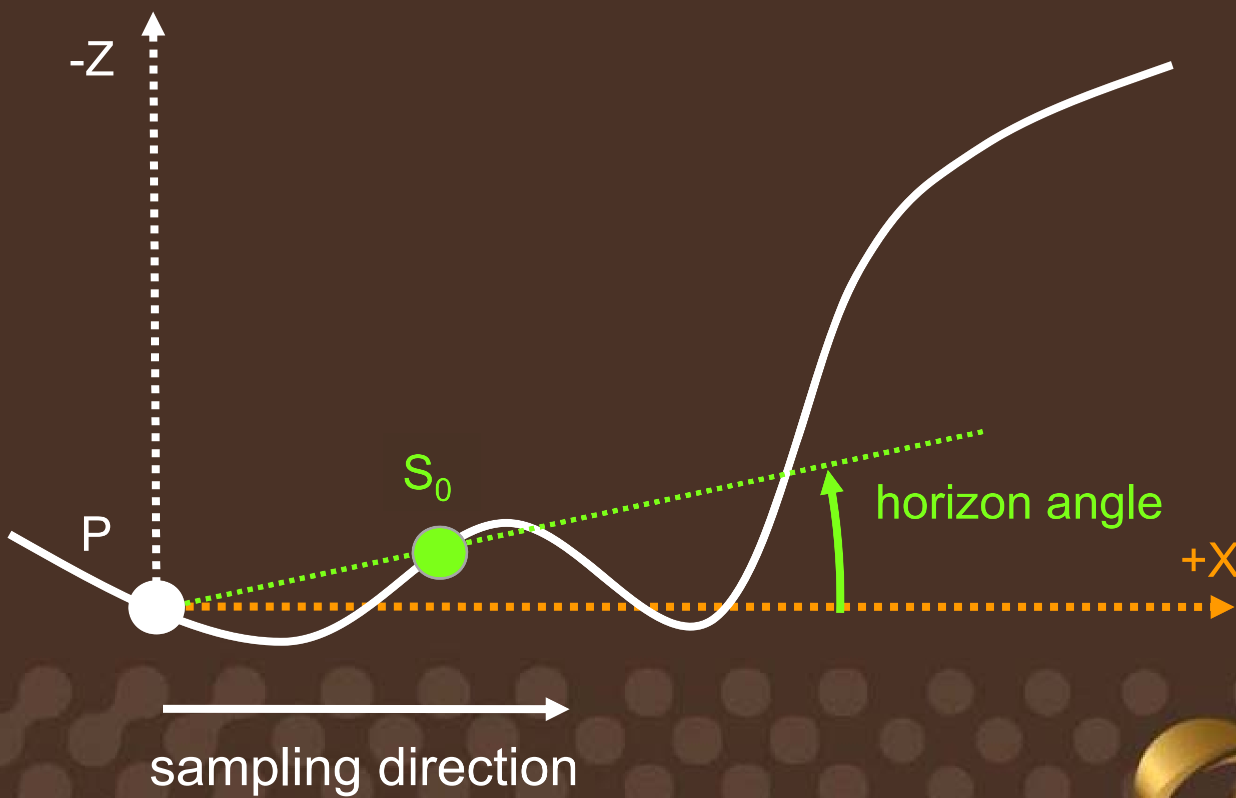
- Given a 1D heightfield



SIGGRAPH2008

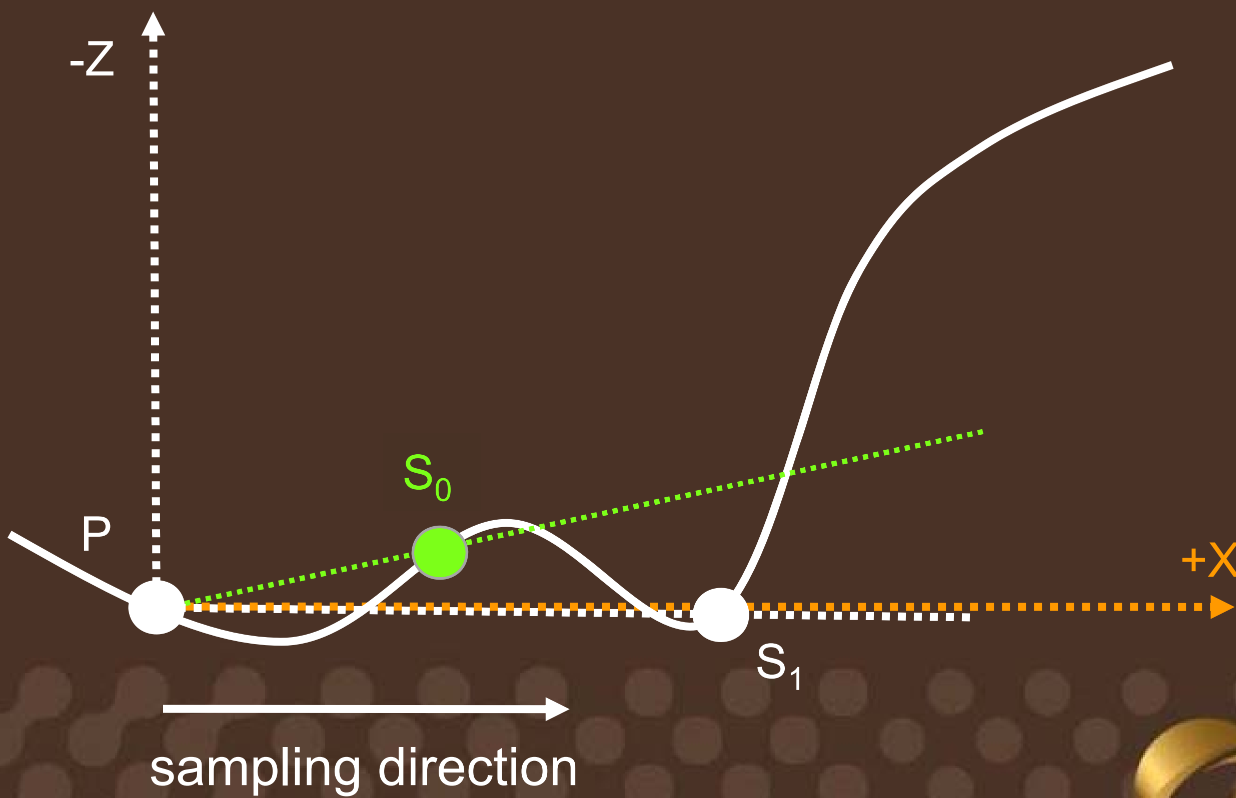
Finding the Horizon

- Marching on the heightfield



Finding the Horizon

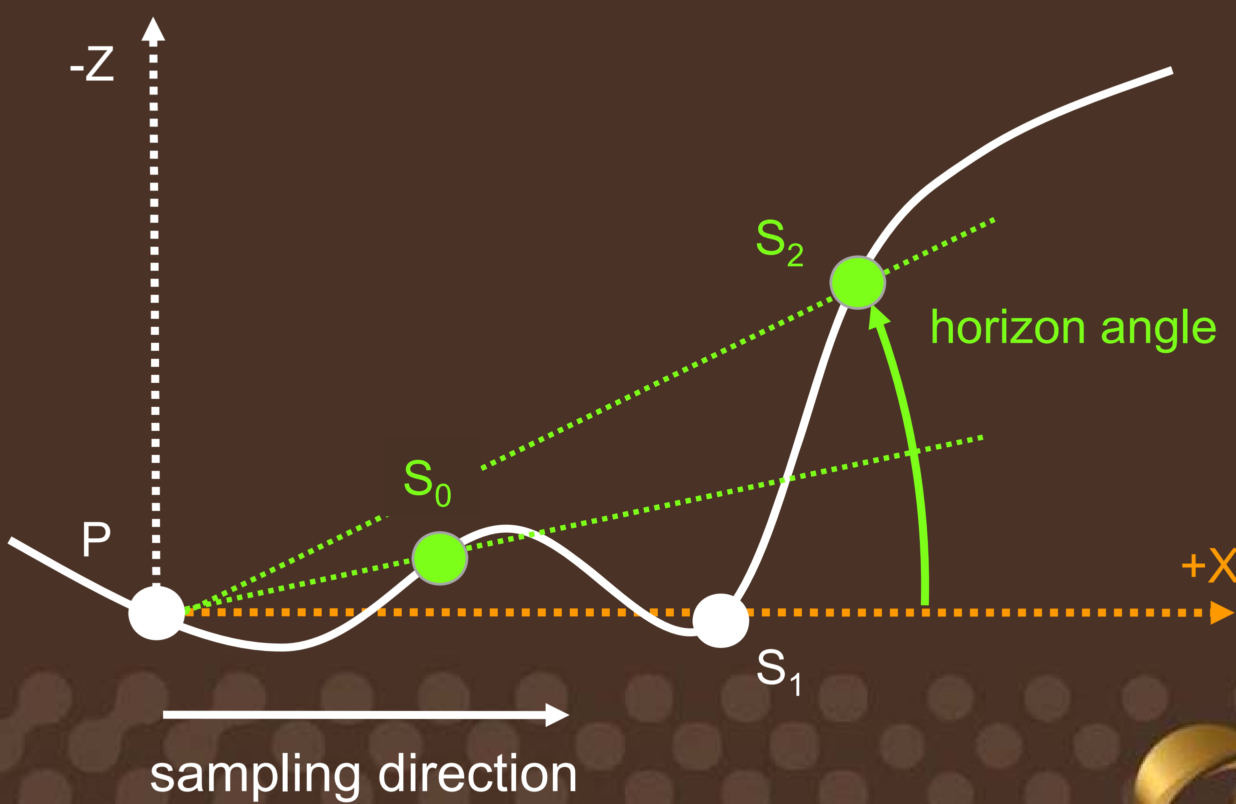
- Marching on the heightfield



SIGGRAPH2008

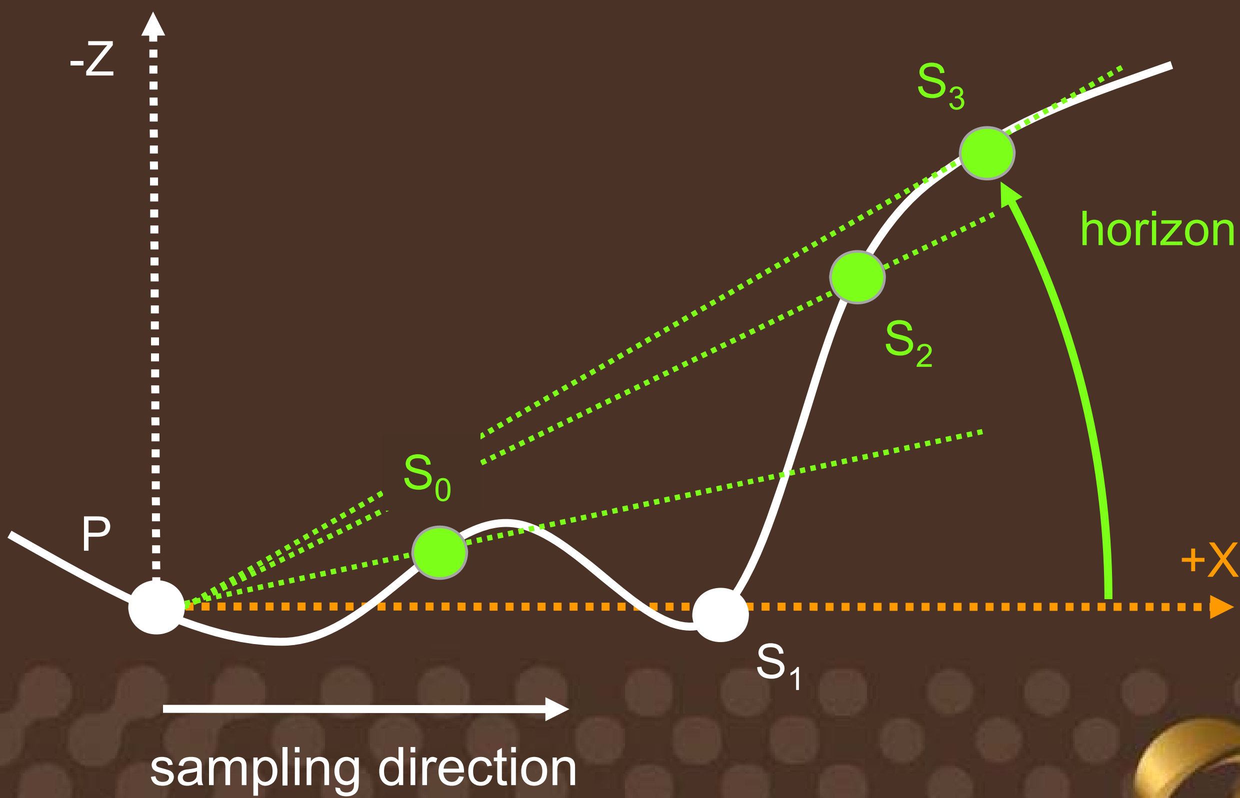
Finding the Horizon

- Marching on the heightfield



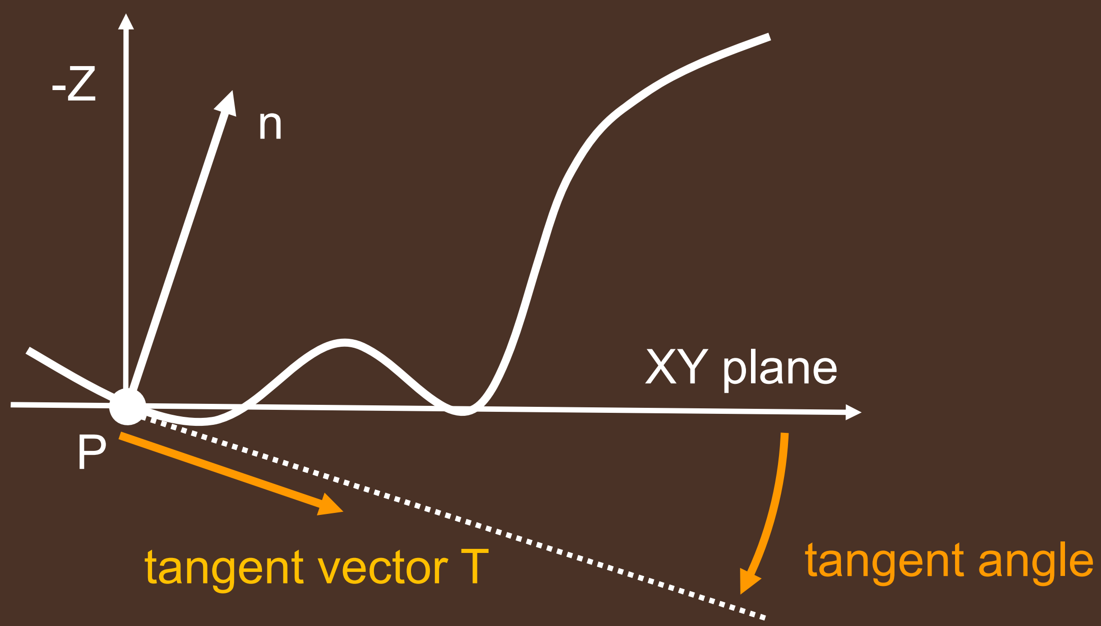
Finding the Horizon

- Marching on the heightfield

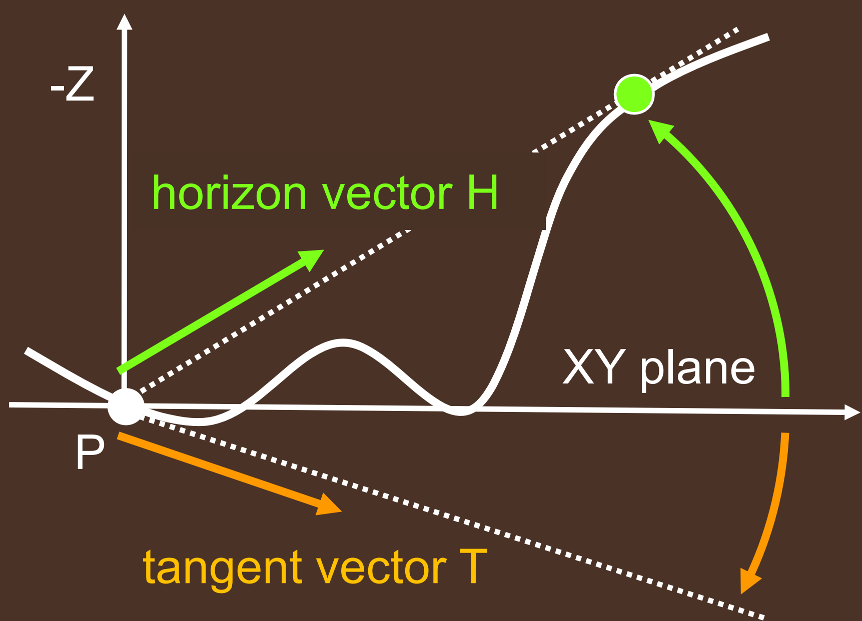


Tangent Plane

- Given point P and its normal n



Horizon-Based AO



horizon angle in $[-\pi/2, \pi/2]$

$h(H) = \mathrm{atan}(H.z \,/\, ||H.xy||)$

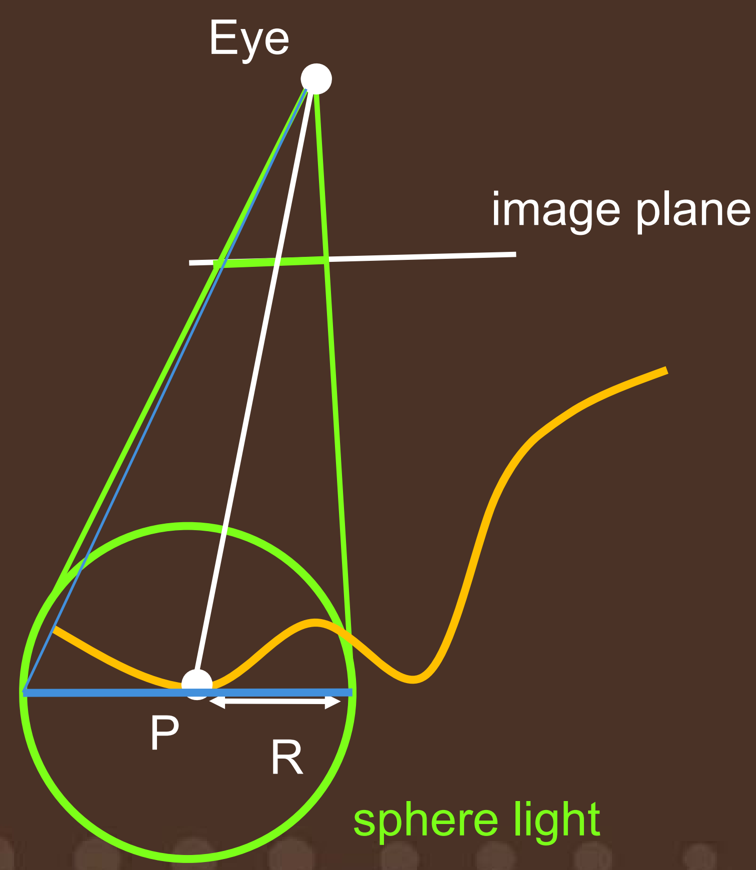
tangent angle in $[-\pi/2, \pi/2]$

$t(T) = \mathrm{atan}(T.z \,/\, ||T.xy||)$

$AO = \sin h - \sin t$

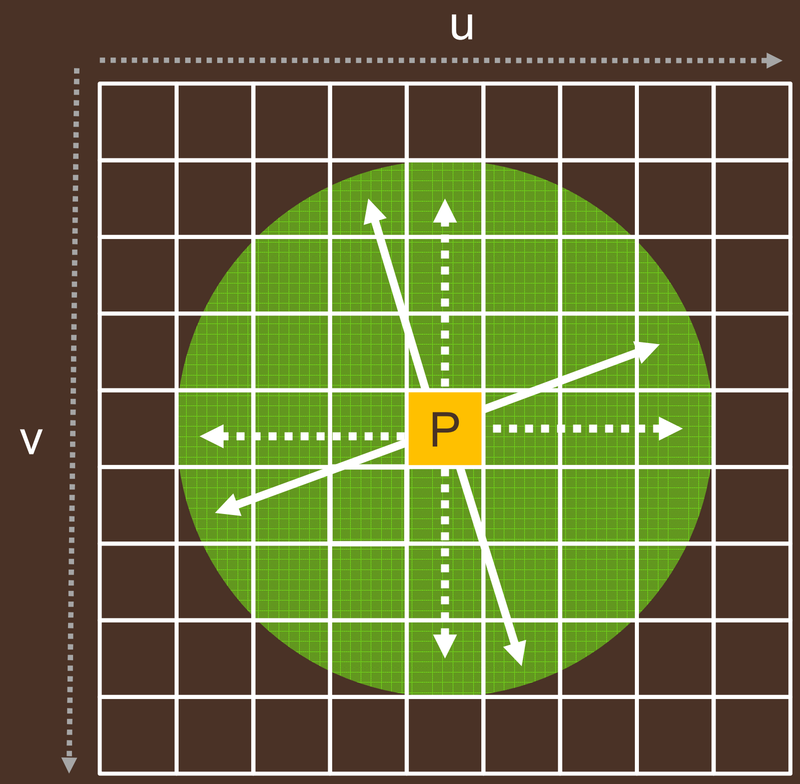
Ambient Occlusion Radius

- Ambient occlusion radius defined in eye space
  - Scene = depth image
- Project light sphere into texture space
  - Approximate projection of the sphere by a disk
  - Project disk onto uv space



Sampling the Depth Image

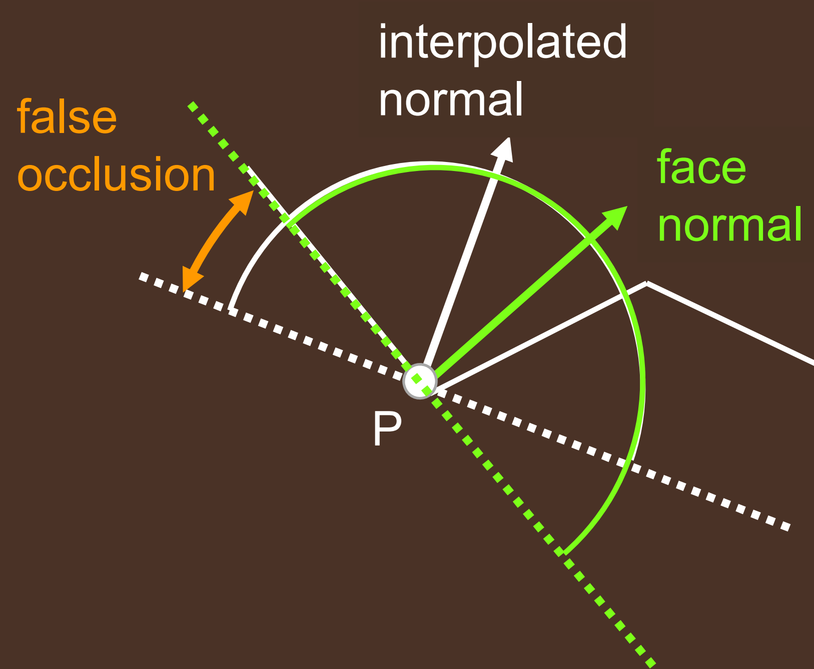
- Use uniform distribution of directions per pixel
  - Fixed num samples / dir
- Per-pixel randomization
  - Rotate directions by random per-pixel angle
  - Jitter samples by a random offset



Example with 4 directions / pixel

Normals

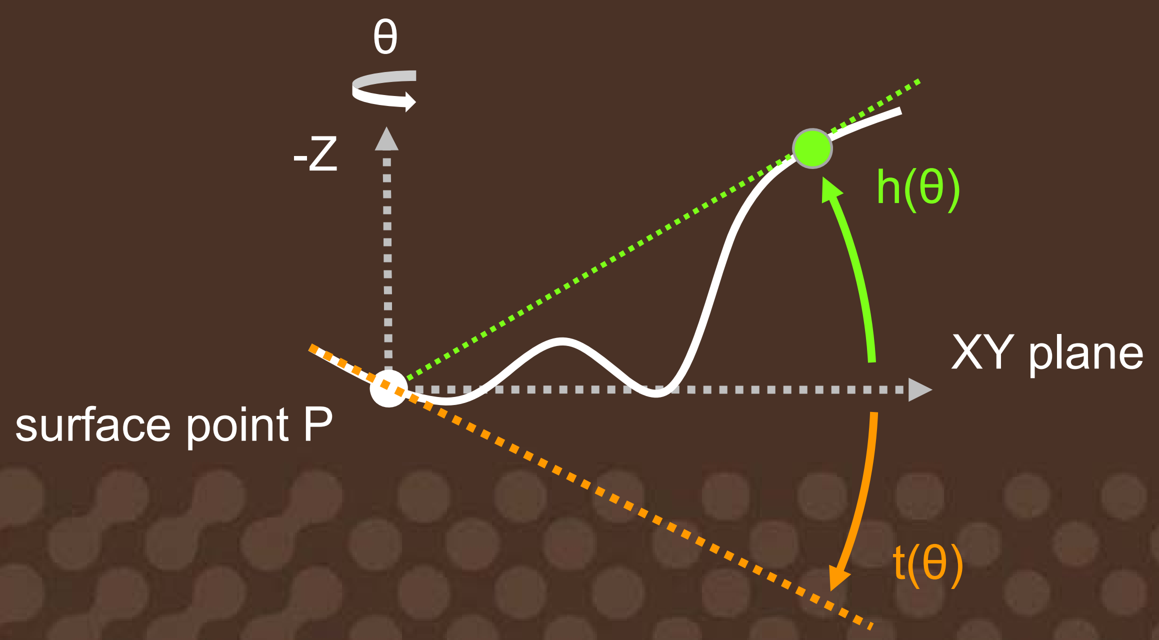
- We store per-pixel normals
  - Not interpolated normals
    - Would generate false occlusion
  - But face normals
    - Using ddx/ddy instructions on eye-space coordinates in the geometry pass



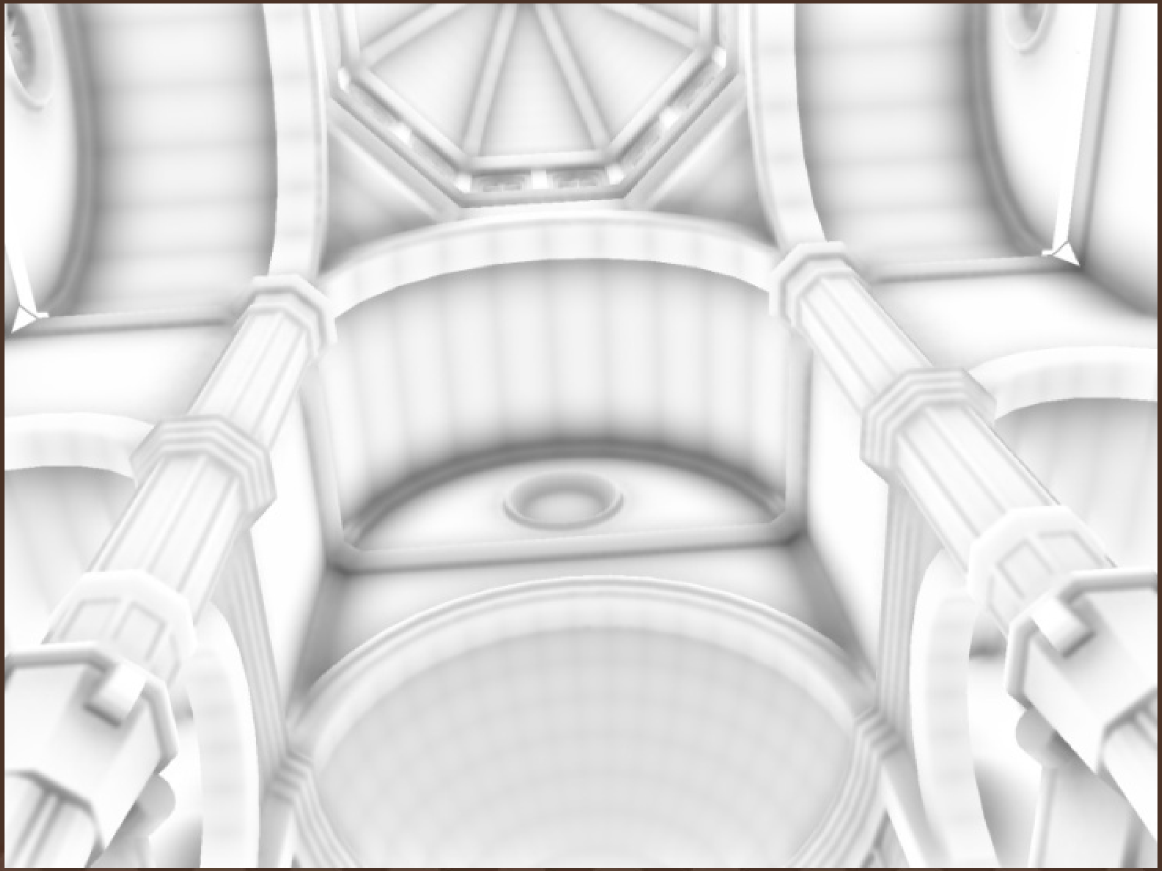
SIGGRAPH2008

Core Algorithm

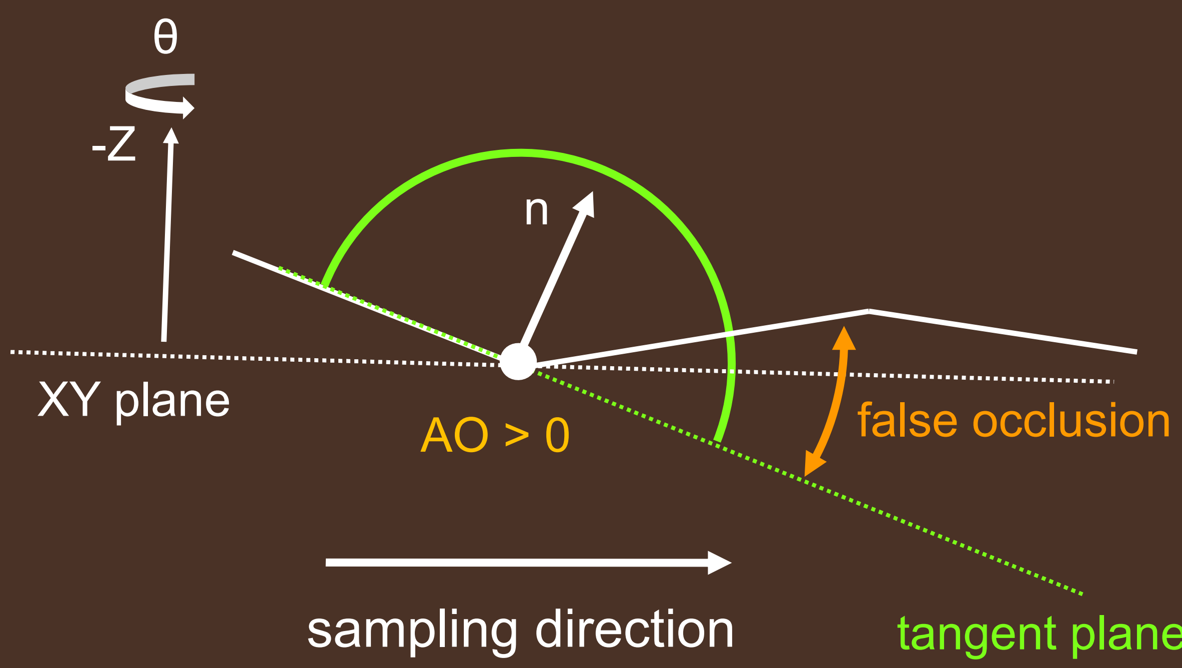
- Integrate AO in 2D
  - Average AO over multiple 2D directions θ
  - AO(θ) = sin h(θ) - sin t(θ)



Ambient Occlusion in Creases

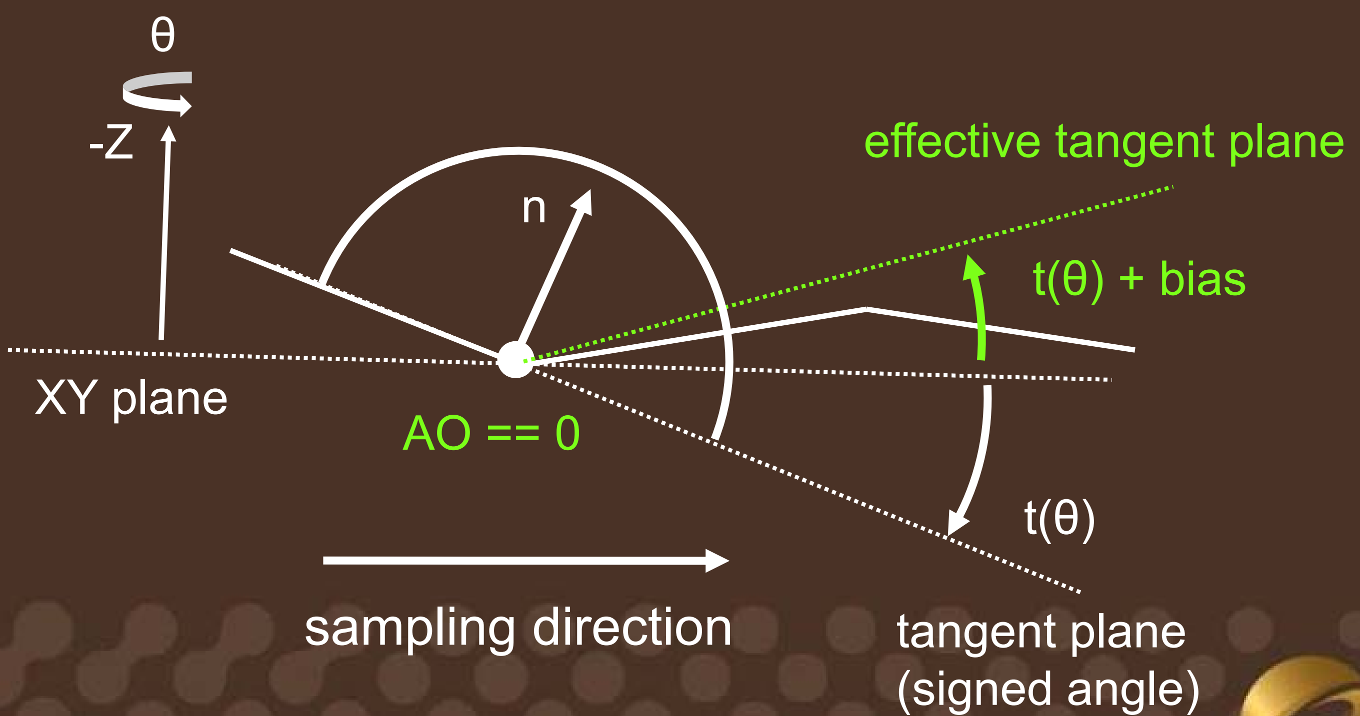


Low-Tessellation Problem

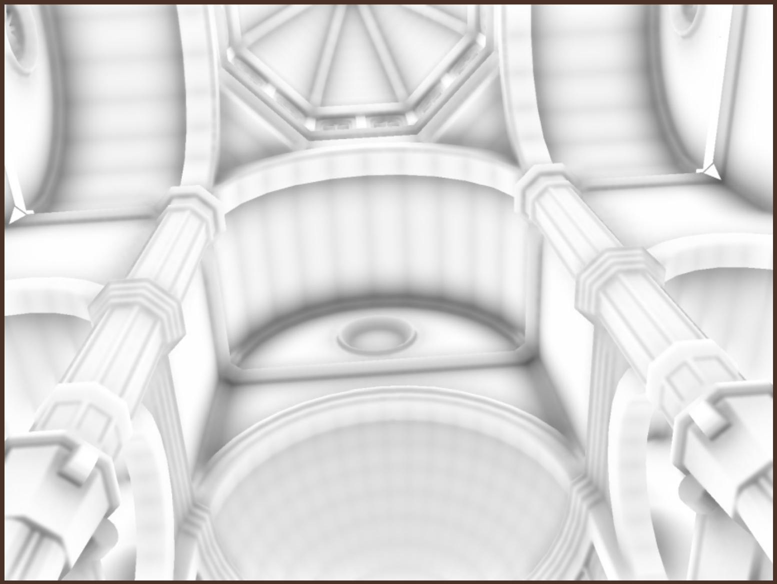


Solution: Angle Bias

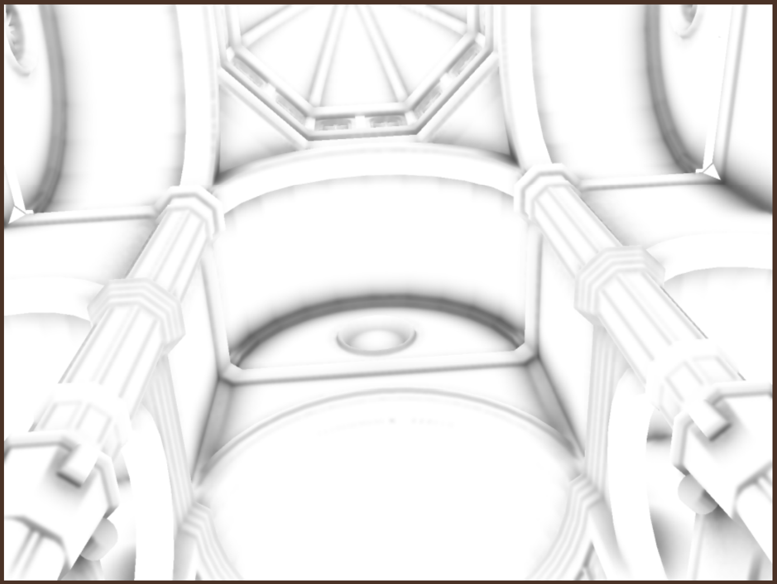
- Similar to “spread” parameter in [Mental Ray]
- Ignore occlusion near the tangent plane



The Angle Bias in Action



Without angle bias



With angle bias = 30 deg

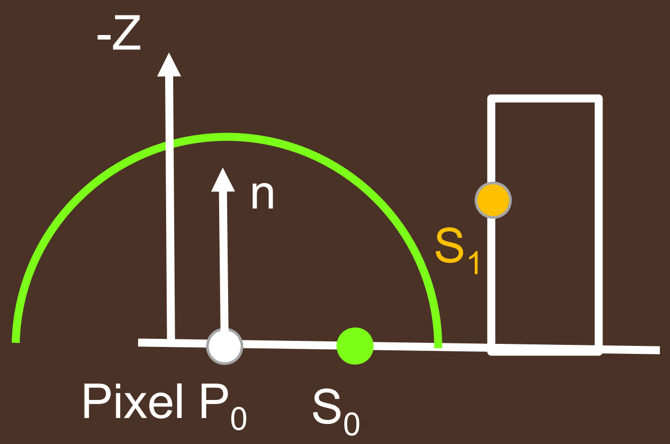
Sampling Outside the Screen

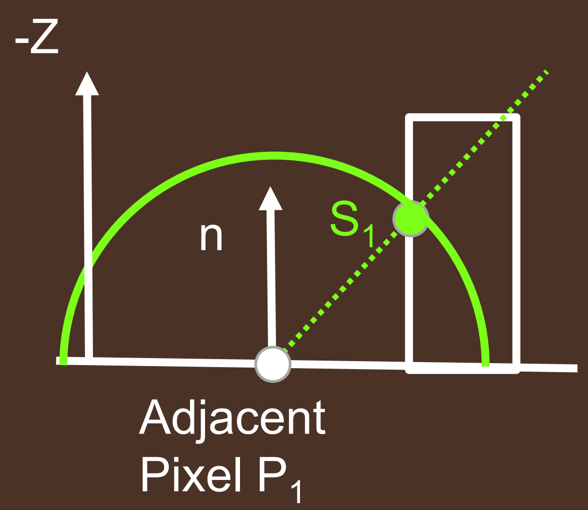
- No scene information outside view frustum
  - We remove false occlusion by using clamping to edge and an angle bias

angle bias = 0

angle bias = 30 deg

Discontinuity Problem





$AO(P_0)$ = sin h – sin t
= sin 0 – sin 0 = 0

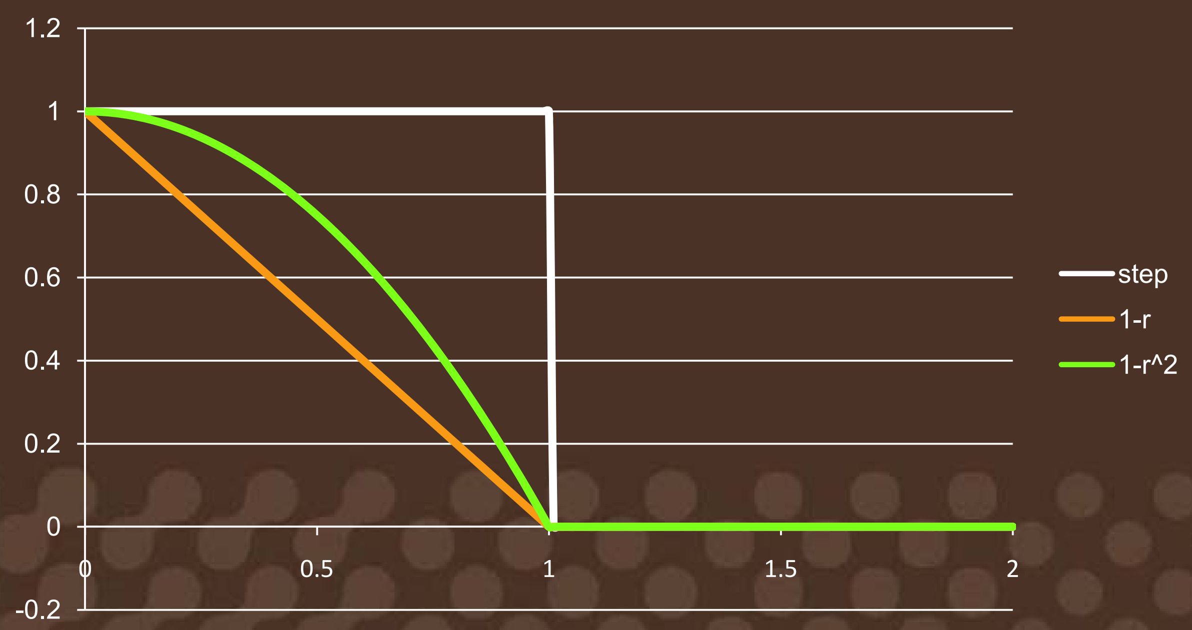
$AO(P_1)$ = sin h – sin t =
= sin(45deg) – sin 0 = 0.7

➔ Large AO discontinuity between P_0 and P_1

SIGGRAPH2008

Attenuation Function

- Weight AO by a radial function W(r)
  - Similar to obscurances [Zhukov et al. 98]
  - "Falloff" in [Gelato] and [Mental Ray]

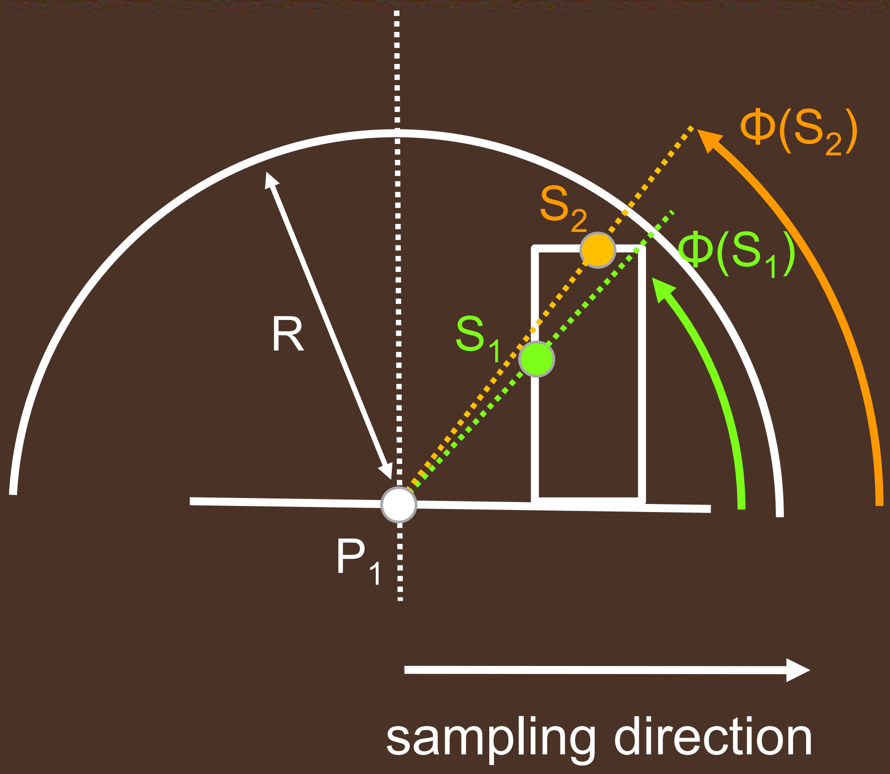


Normalized distance
$r = ||S - P|| / R$

We use the attenuation
$W(r) = 1 - r^2$

Per-Sample Attenuation

- Initialize WAO = 0
- After sample S_1
  - $AO(S_1) = \sin \Phi(S_1) - \sin t$
  - $WAO \mathrel{+}= W(S_1)\ AO(S_1)$
- After sample S_2
  - If $\Phi(S_2) > \Phi(S_1)$
    - $AO(S_2) = \sin \Phi(S_2) - \sin t$
    - $WAO \mathrel{+}= W(S_2)\ (AO(S_2) - AO(S_1))$



With and Without Attenuation

With Attenuation
$W(r) = 1 - r^2$

Without Attenuation
$W(r) = 1$

Noise

- Per-pixel randomization generates noise

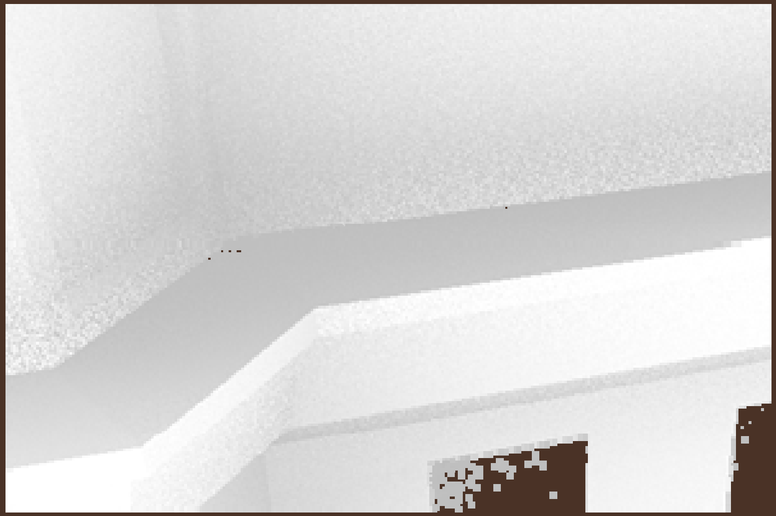
AO with 6 directions x 6 steps/dir

Cross Bilateral Filter

- We blur the ambient occlusion
- Depth-dependent Gaussian blur
  - [Petschnigg et al. 04]
    [Eisemann and Durand 04]
  - Reduces blurring across edges
- Although it is a non-separable filter, we apply it separately in the X and Y directions

Cross Bilateral Filter

- Depth-dependent Blur



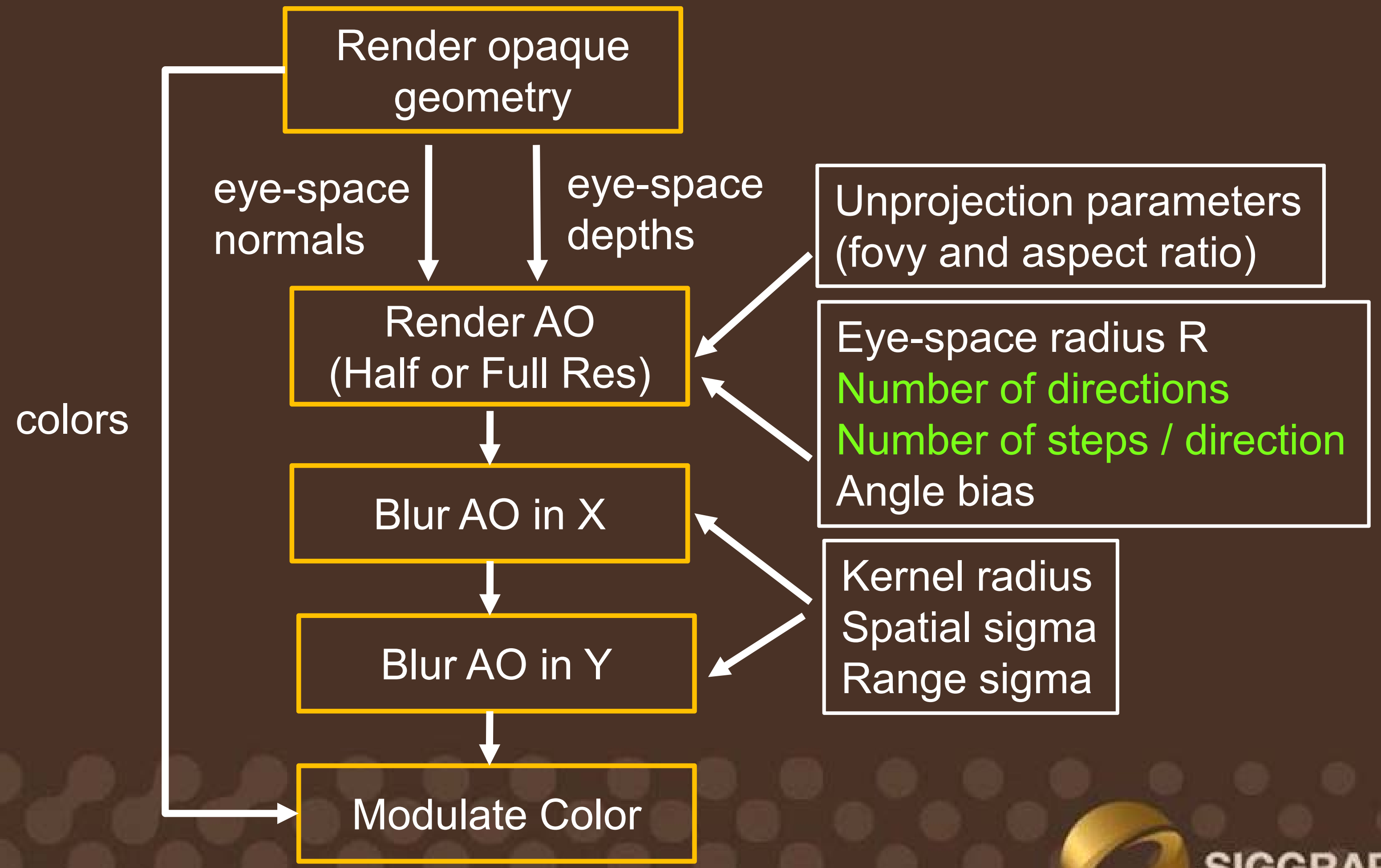
Without Blur

With 15x15 Blur

Half-Resolution AO

- AO is mostly low frequency
  - Can render the AO in half resolution
    - Source half-resolution depth image
- Still do the blur passes in full resolution
  - To avoid bleeding across edges
  - Source full resolution eye-space depths
    - [Kopf et al. 07]

Rendering Pipeline

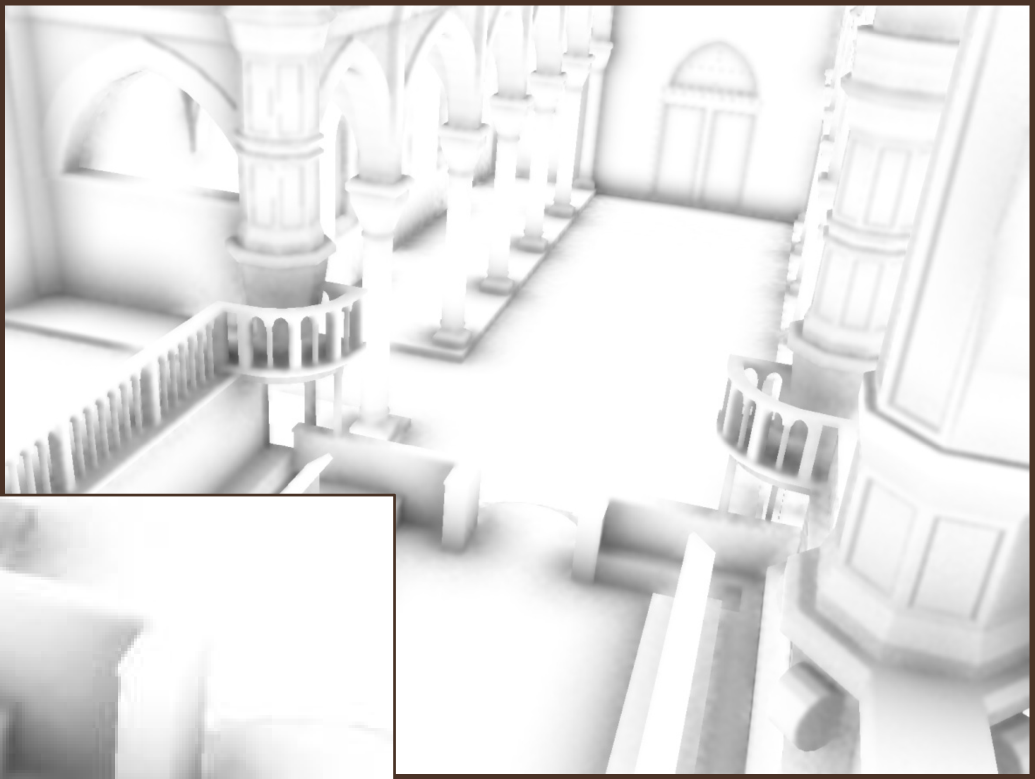


Demo

Performance

- Depends on
  - Screen Resolution
  - Ambient Occlusion Resolution
  - Number of samples (directions * steps)
  - Blur Size

Half-Resolution AO



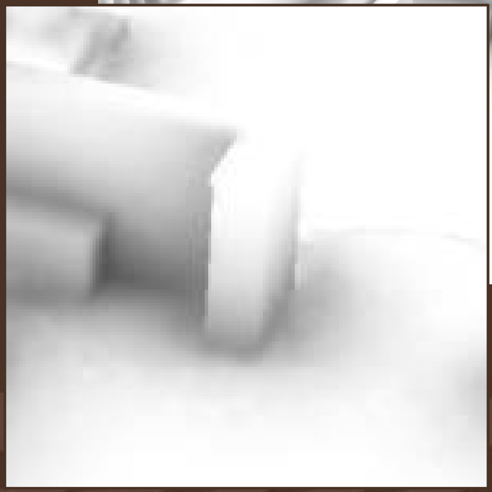


Image Size	1600x1200
AO Resolution	800x600
Blur Resolution	1600x1200

Half-Res AO	GeForce GTX 280
Geometry	1.0 ms
AO	3.5 ms
Blur	2.5 ms
Total	7.0 ms 143 fps

6 directions per pixel
6 steps per direction
15x15 Blur Size

Full-Resolution AO

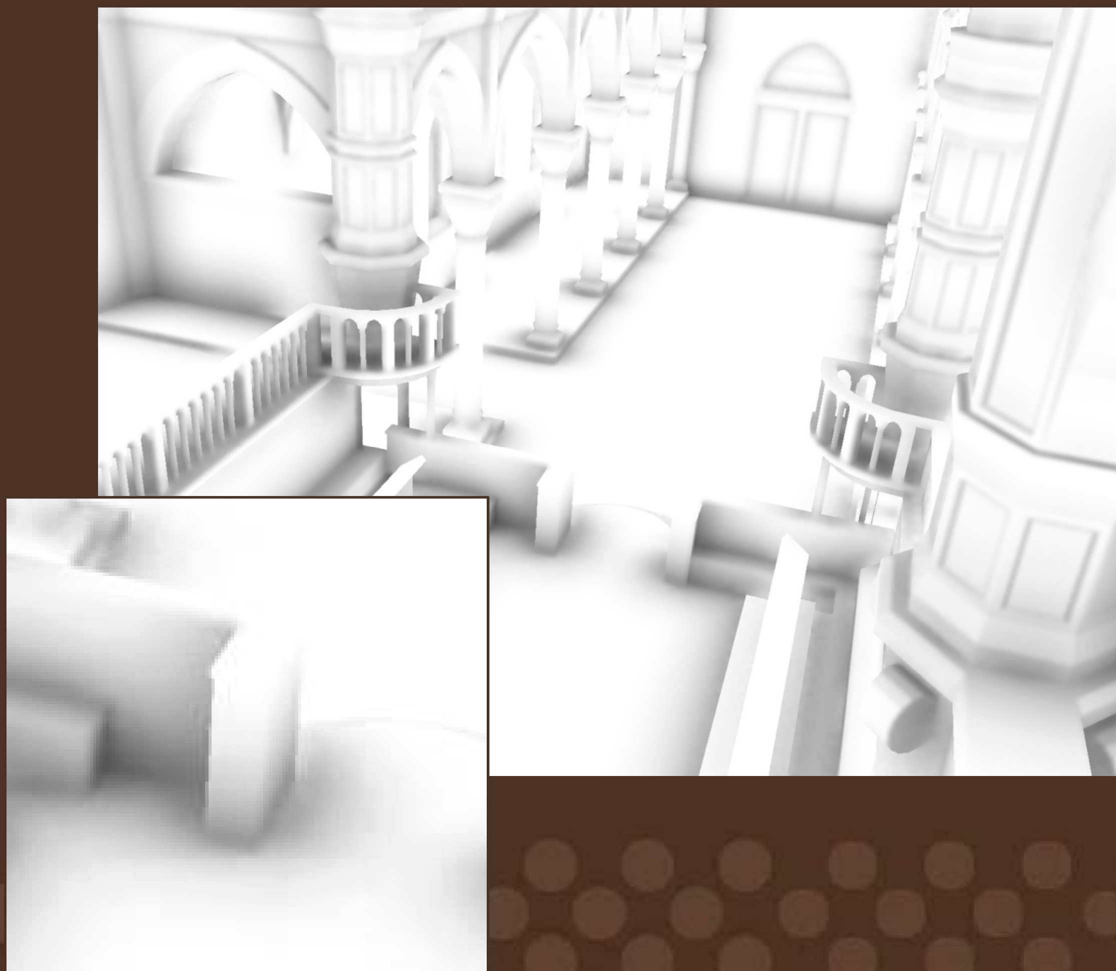


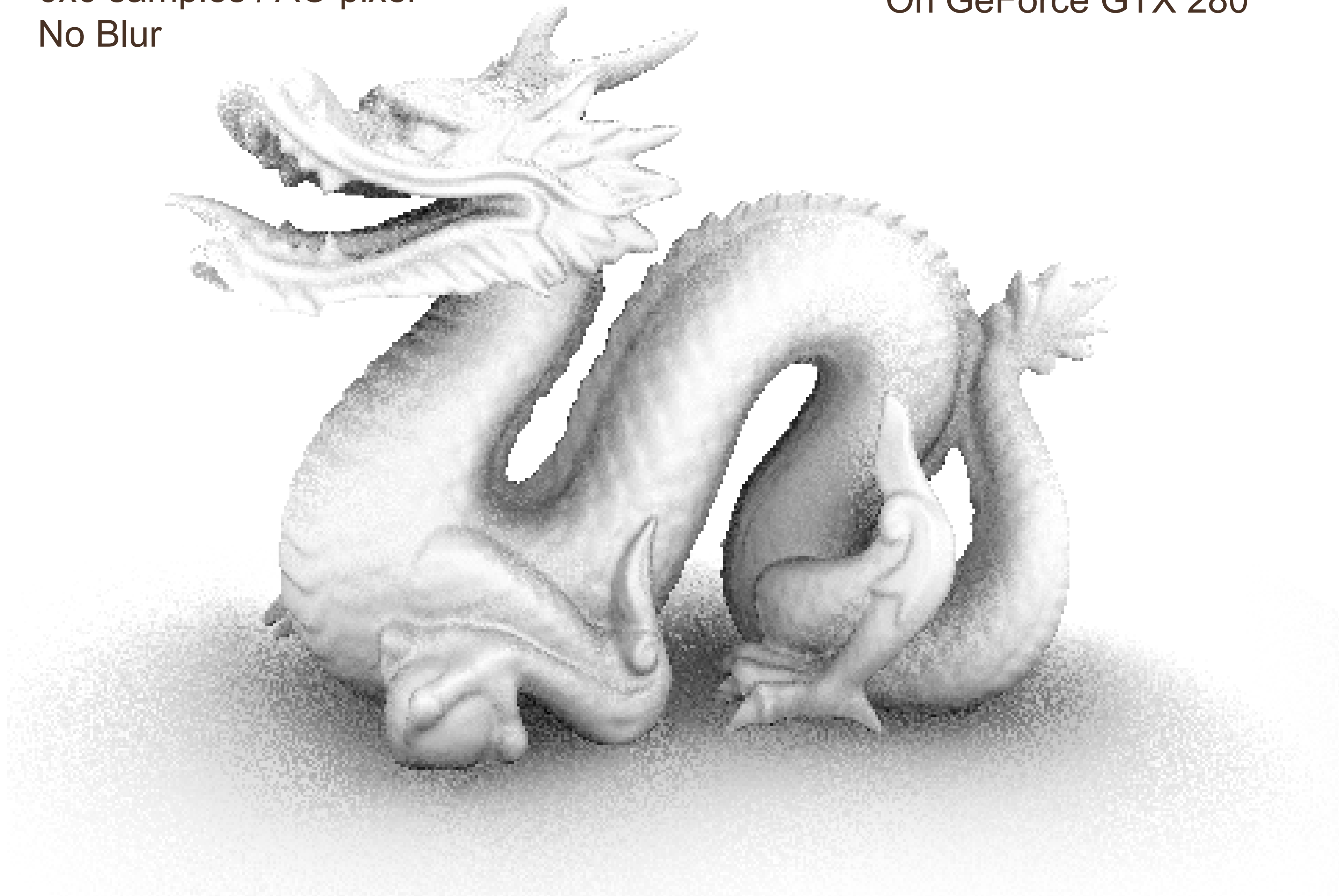
Image Size 1600x1200
AO Resolution 1600x1200
Blur Resolution 1600x1200

Full-Res AO	GeForce GTX 280
Geometry AO Blur	1.0 ms 30.0 ms 2.5 ms
Total	33.5 ms 30 fps

6 directions per pixel
6 steps per direction
15x15 Blur Size

Half-Resolution AO
6x6 samples / AO pixel
No Blur

AO = 3.5 ms @ 800x600
On GeForce GTX 280

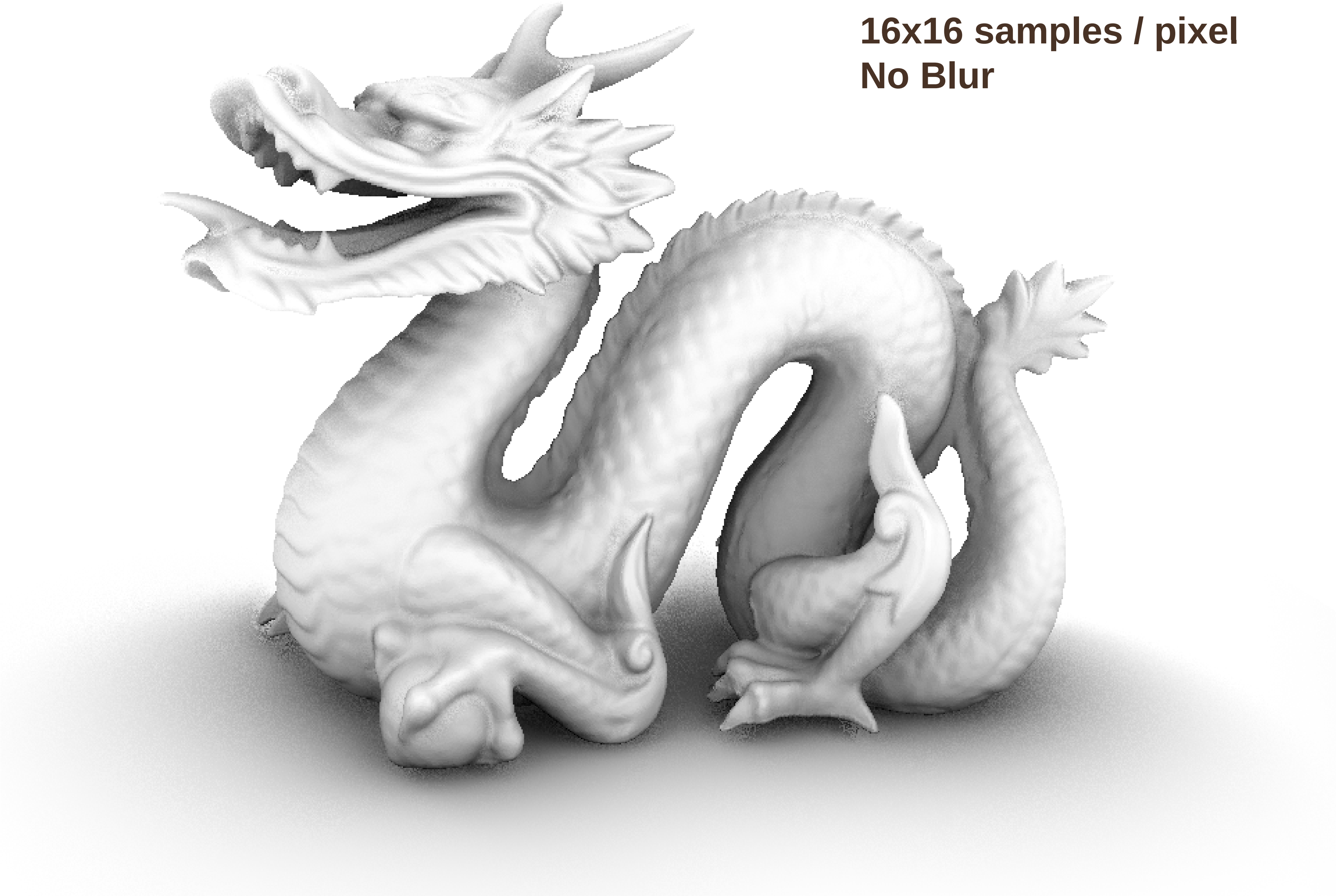


Half-Resolution AO
6x6 samples / AO pixel
15x15 Blur

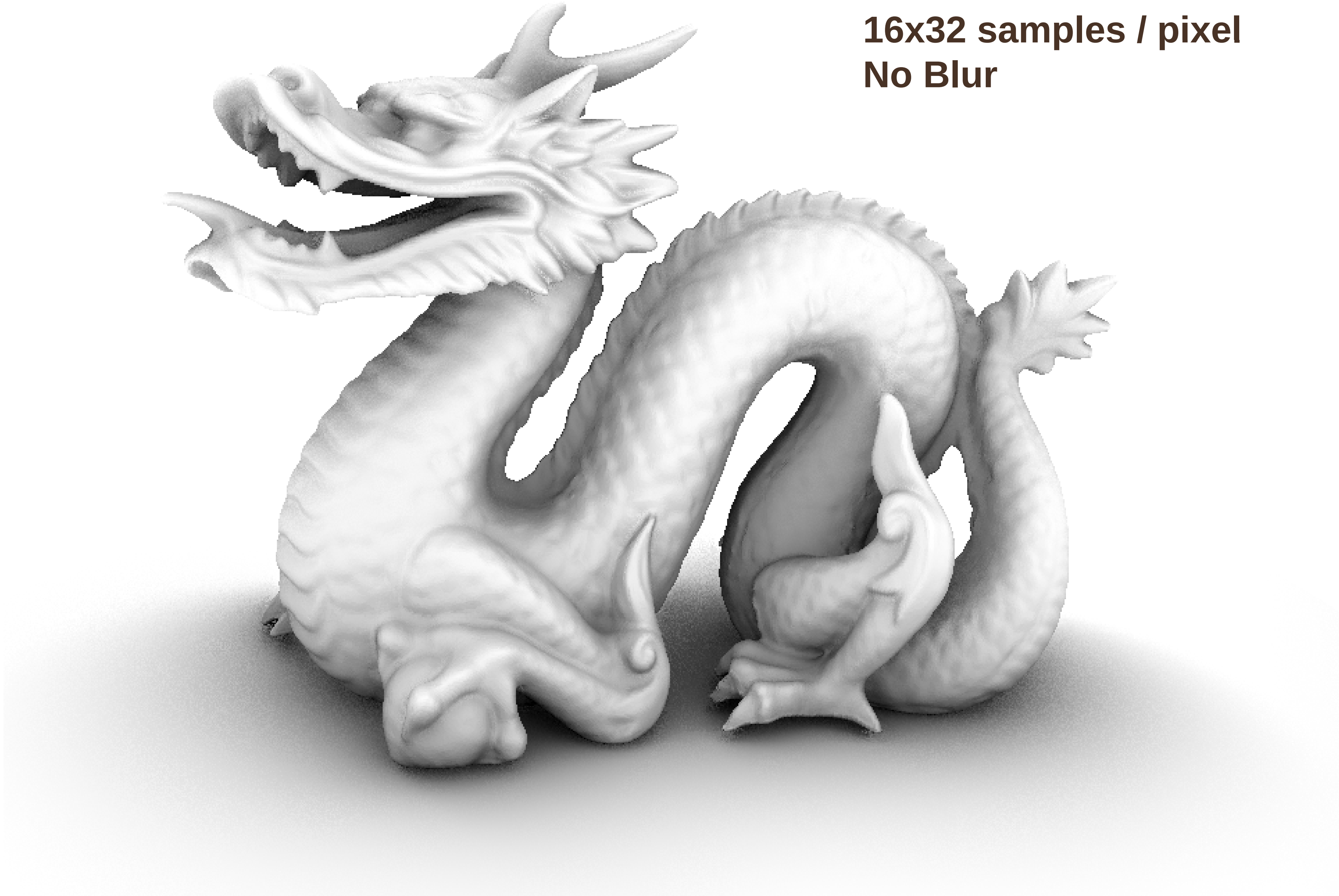
AO = 3.5 ms @ 800x600
Blur = 2.5 ms @ 1600x1200
On GeForce GTX 280

Full-Resolution AO
6x6 samples / AO pixel
15x15 Blur

AO = 30 ms @ 800x600
Blur = 2.5 ms @ 1600x1200
On GeForce GTX 280



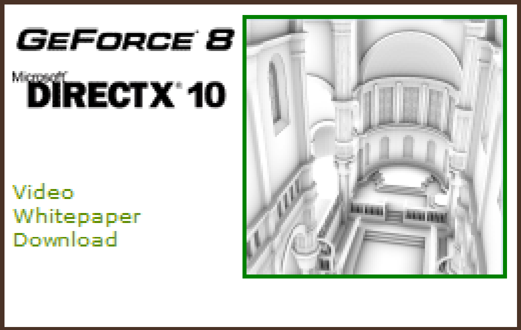
Full-Resolution AO
16x16 samples / pixel
No Blur



Full-Resolution AO
16x32 samples / pixel
No Blur

Conclusion

- DirectX10 SDK sample
  - Now available on developer.nvidia.com
  - Including video and brief whitepaper



- Easy to integrate into a game engine
  - Input Data = eye-space depths and normals
  - Rendered in a post-processing pass
- More details in ShaderX^7 (to appear)

Acknowledgments

– NVIDIA

- Rouslan Dimitrov, Samuel Gateau, Michael Thompson, Ignacio Castano, the demo team

– Models

- Dragon - Stanford 3D Scanning Repository
- Sibenik Cathedral - Marko Dabrovic

Questions?

Code sample available on
http://developer.nvidia.com

lbavoil@nvidia.com
msainz@nvidia.com

References

- [Max 86] MAX, N. L. 1986. "Horizon mapping: Shadows for bump-mapped surfaces." In Proceedings of Computer Graphics Tokyo '86 on Advanced Computer Graphics.
- [Zhukov et al. 98] Sergej Zhukov, Andrej Inoes, Grigorij Kronin, 1998. "An ambient light illumination model." In Rendering Techniques '98, G. Drettakis and N. Max, Eds., Eurographics, 45–56.
- [Landis 02] Landis, 2002. "Production-Ready Global Illumination," In ACM SIGGRAPH Course #16.
- [Christensen 03] Christensen, P. H. 2003. "Global illumination and all that". In ACM SIGGRAPH Course 9.

References

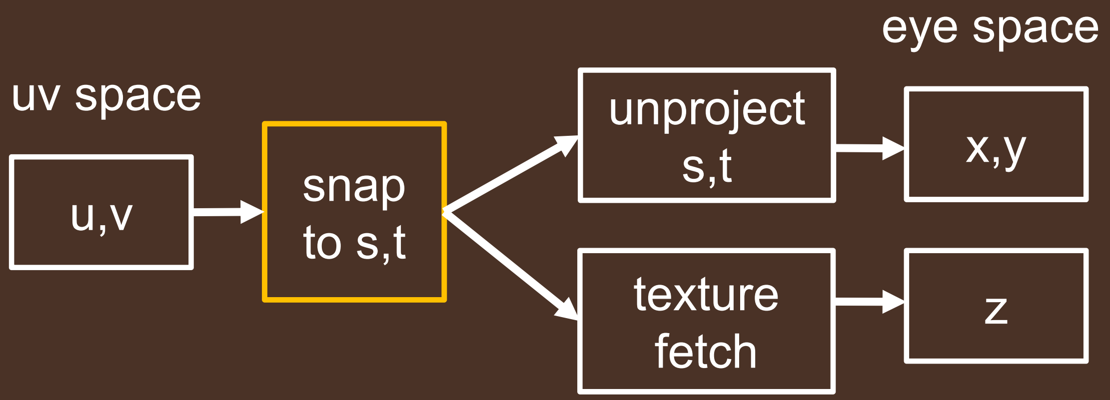
- [Eisemann and Durand 04] Elmar Eisemann and Frédo Durand, "Flash Photography Enhancement via Intrinsic Relighting", ACM SIGGRAPH 2004.
- [Petschnigg et al. 04] Petschnigg, Szeliski, Agrawala, Cohen, Hoppe, Toyama, "Digital photography with flash and no-flash image pairs." ACM SIGGRAPH 2004.
- [Schüler 05] Christian Schüler, 2005. "Eliminate surface acne with gradient shadow mapping", ShaderX^4: Advanced Rendering Techniques.
- [Gelato] Gritz, L. 2006. "Gelato 2.1 Technical Reference". NVIDIA.
- [Mental Ray] "mental ray Shader Reference", November 2007.

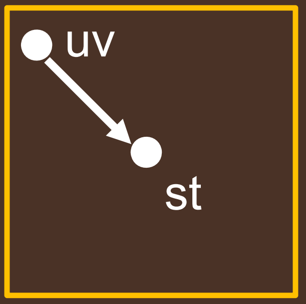
References

- [Mittring 07] MITTRING, M. 2007. “Finding next gen: Cry Engine 2”. In SIGGRAPH ’07: ACM SIGGRAPH 2007 courses.
- [Shanmugam and Orikan 07] SHANMUGAM, P., AND ARIKAN, O. 2007. “Hardware accelerated ambient occlusion techniques on GPUs”. In I3D ’07: Proceedings of the 2007 symposium on Interactive 3D graphics and games.
- [Kopf et al. 07] Johannes Kopf, Michael F. Cohen, Dani Lischinski, Matt Uyttendaele, “Joint Bilateral Upsampling”, SIGGRAPH '07: ACM SIGGRAPH 2007 papers.
- [Fox and Compton 08] Megan Fox and Stuart Compton, “Ambient Occlusive Crease Shading”, Game Developer Magazine, March 2008.

Computing Eye-Space Positions

- For a given sample S at (u,v)





snapping to
pixel center

  - Snap uv to avoid mismatch between reconstructed (x,y) and sampled z

Tangent Angle

- Given a tangent vector T
  - T(θ) on the plane
  - $t(\theta) = atan(T.z / ||T.xy||)$
- Compute plane basis
  - Basis = (dP/du,dP/dv)
  - $T = dP/du\ \Delta u + dP/dv\ \Delta v$
  - Similar to gradient shadow mapping [Schüler 05]