Shading in Valve's Source Engine

SIGGRAPH 2006

Jason Mitchell
Valve

SIGGRAPH2006

Introduction

- World Lighting
  - Radiosity Normal Mapping
  - Specular Lighting
- Model lighting
  - Irradiance Volume
  - Half-Lambert
  - Phong terms
- High Dynamic Range Rendering
  - Tone mapping
  - Autoexposure
- Color Correction

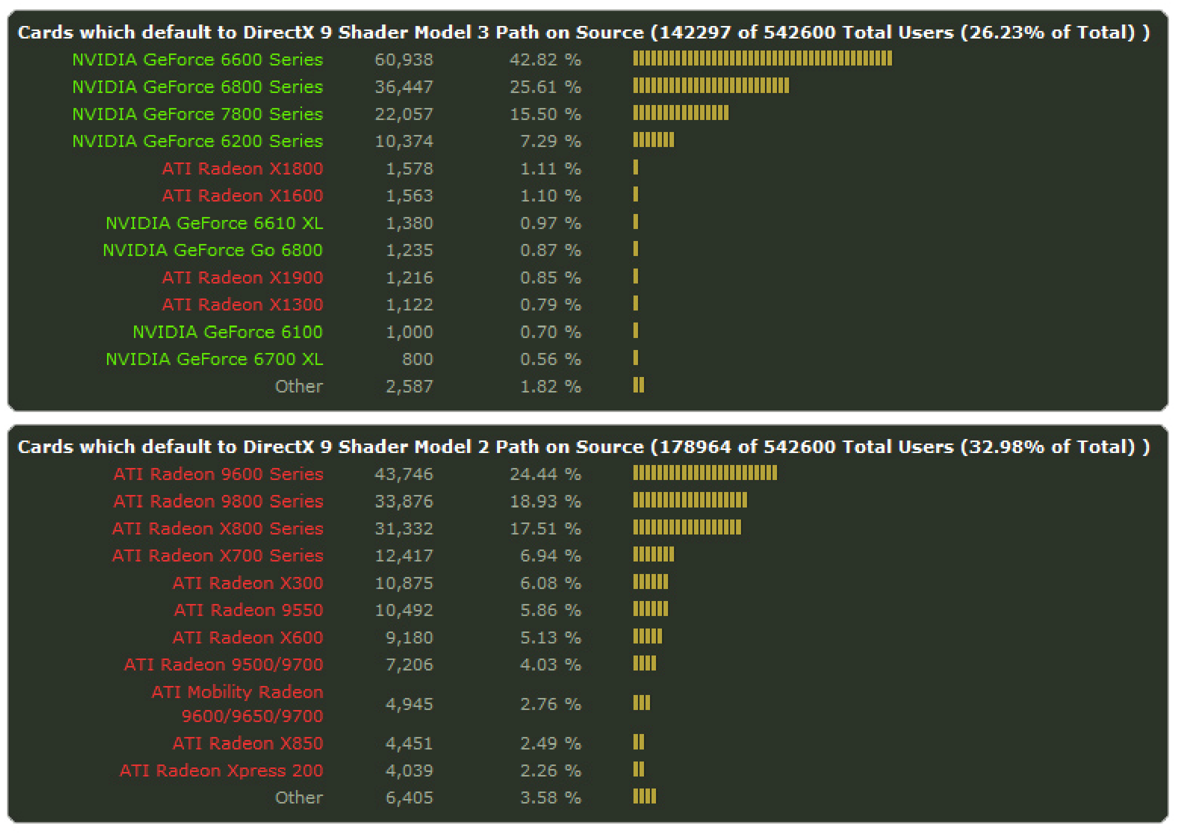
Timeline

- Half-Life 2 and Counter-Strike: Source (November 2004)
    - Baseline shading model we’ll present today
- Lost Coast (October 2005)
    - High Dynamic Range Rendering
- Day of Defeat: Source (September 2005)
    - Color Correction and HDR
- Half-Life 2: Episode One (June 2006)
    - Phong shading

Episodic Content ➔ Incremental Technology

- The extensions to the baseline Half-Life 2 technology have been shipped in subsequent products in the year and a half since HL2 shipped.
  - More efficient way to develop technology
  - Delivers value to customers and licensees
- Shorten the delay between hardware availability and game support

Marketplace Context



Cards which default to DirectX 9 Shader Model 3 Path on Source (142297 of 542600 Total Users (26.23% of Total))

NVIDIA GeForce 6600 Series	60,938	42.82 %
NVIDIA GeForce 6800 Series	36,447	25.61 %
NVIDIA GeForce 7800 Series	22,057	15.50 %
NVIDIA GeForce 6200 Series	10,374	7.29 %
ATI Radeon X1800	1,578	1.11 %
ATI Radeon X1600	1,563	1.10 %
NVIDIA GeForce 6610 XL	1,380	0.97 %
NVIDIA GeForce Go 6800	1,235	0.87 %
ATI Radeon X1900	1,216	0.85 %
ATI Radeon X1300	1,122	0.79 %
NVIDIA GeForce 6100	1,000	0.70 %
NVIDIA GeForce 6700 XL	800	0.56 %
Other	2,587	1.82 %

Cards which default to DirectX 9 Shader Model 2 Path on Source (178964 of 542600 Total Users (32.98% of Total))

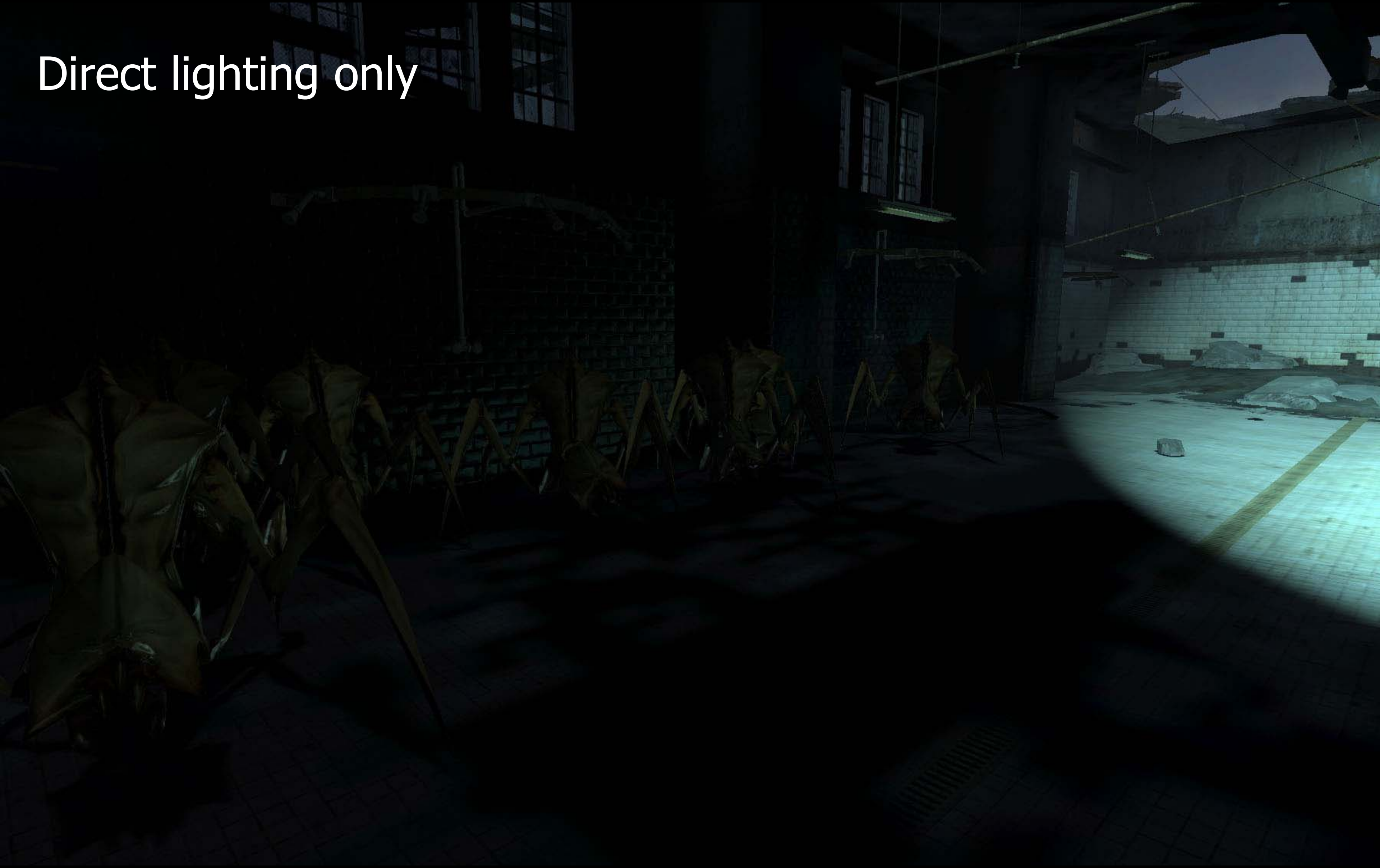
ATI Radeon 9600 Series	43,746	24.44 %
ATI Radeon 9800 Series	33,876	18.93 %
ATI Radeon X800 Series	31,332	17.51 %
ATI Radeon X700 Series	12,417	6.94 %
ATI Radeon X300	10,875	6.08 %
ATI Radeon 9550	10,492	5.86 %
ATI Radeon X600	9,180	5.13 %
ATI Radeon 9500/9700	7,206	4.03 %
ATI Mobility Radeon 9600/9650/9700	4,945	2.76 %
ATI Radeon X850	4,451	2.49 %
ATI Radeon Xpress 200	4,039	2.26 %
Other	6,405	3.58 %

See full survey at http://www.steampowered.com/status/survey.html

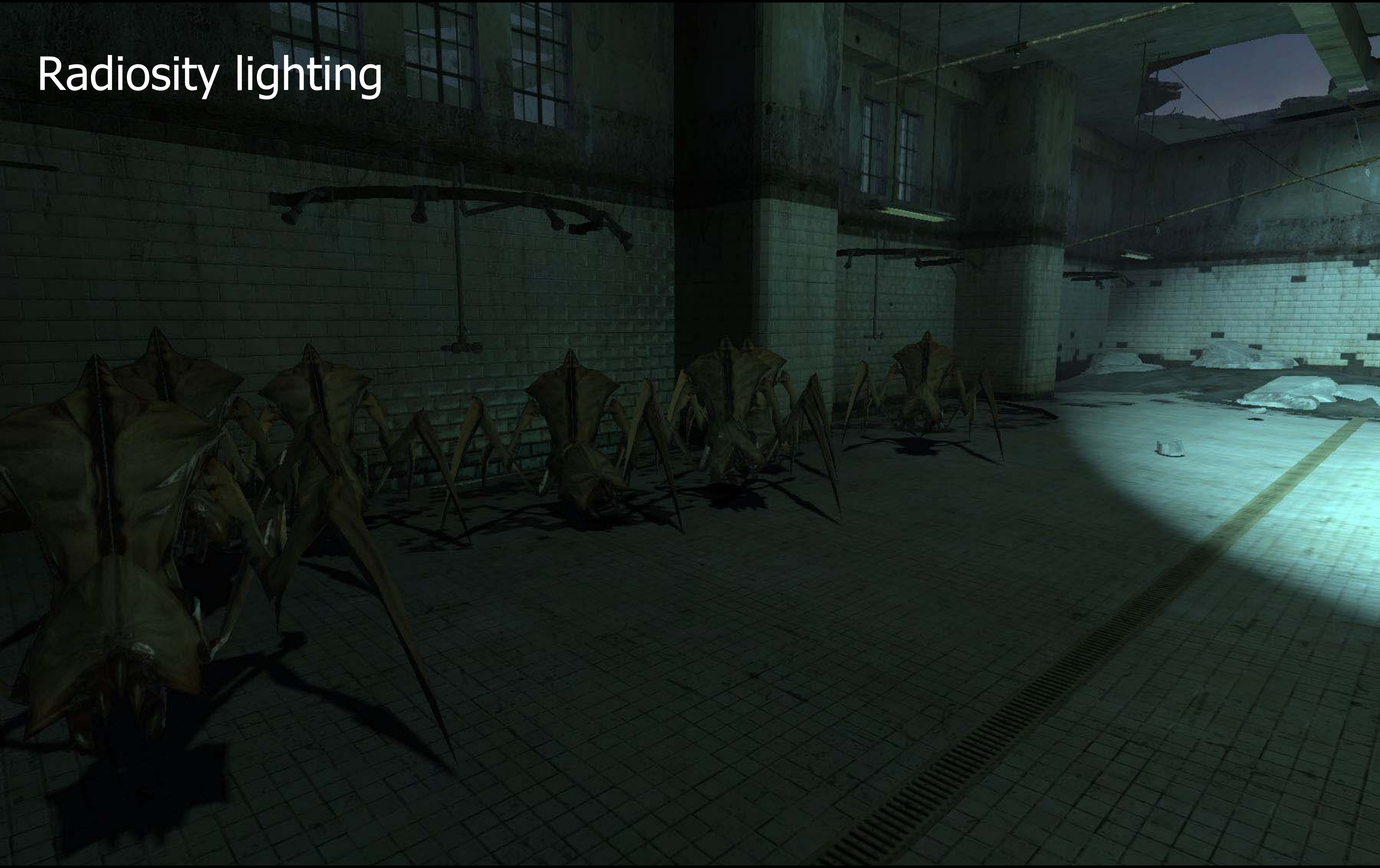
World Lighting with Radiosity

- Realism
- More forgiving than direct lighting only
  - Avoids harsh lighting situations
  - Less micro-management of light sources for content production
  - Can’t tune lights shot-by-shot like movies. Don’t know what the shots are, and don’t want to take the production time to do this

Direct lighting only



Radiosity lighting



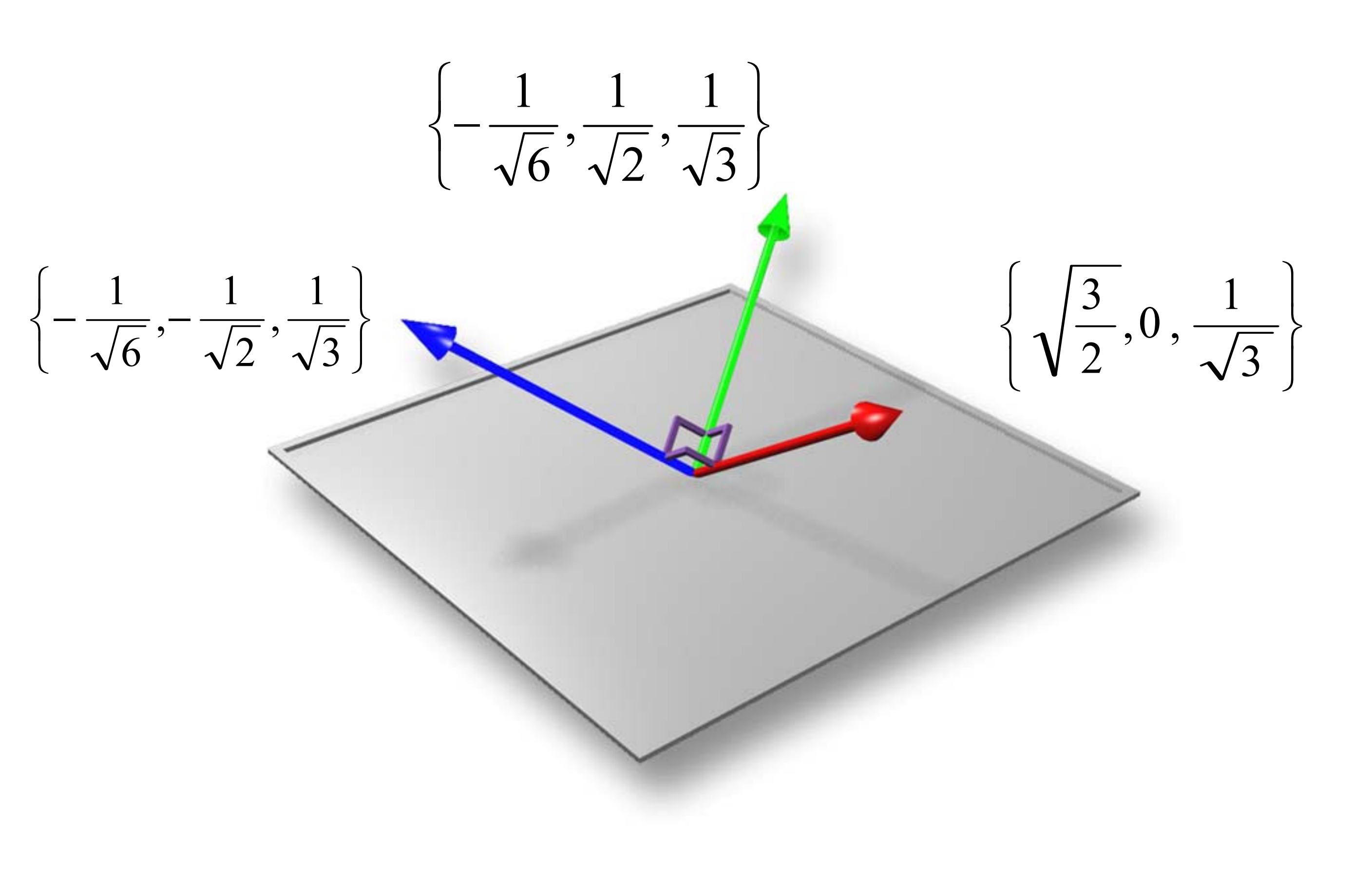
Normal Mapping

- Reusable high-frequency detail
- Higher detail than we can currently get from triangles
- Works well with both diffuse and specular lighting models
- Can now be made to integrate with radiosity

Radiosity Normal Mapping

- We created this technique to efficiently address the strengths of both radiosity and normal mapping
- In other engines, per-pixel lighting terms are often accumulated one light at a time
  - Multiple lights handled by summing multiple light contributions within or between passes
- We express the full lighting environment in our novel basis in order to effectively perform diffuse bump mapping with respect to an arbitrary number of lights
- This is the key to shading in Valve's Source engine

Basis for Radiosity Normal Mapping



$$\left\{-\frac{1}{\sqrt{6}},\frac{1}{\sqrt{2}},\frac{1}{\sqrt{3}}\right\}$$

$$\left\{-\frac{1}{\sqrt{6}},-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{3}}\right\}$$

$$\left\{\sqrt{\frac{3}{2}},0,\frac{1}{\sqrt{3}}\right\}$$

Computing Light map Values

- Traditionally, when computing light map values using a radiosity preprocessor, a single color value is calculated
- In Radiosity Normal Mapping, compute light values for each vector in our basis
    - This triples light map storage, but we are willing to bear this cost for the added quality and flexibility

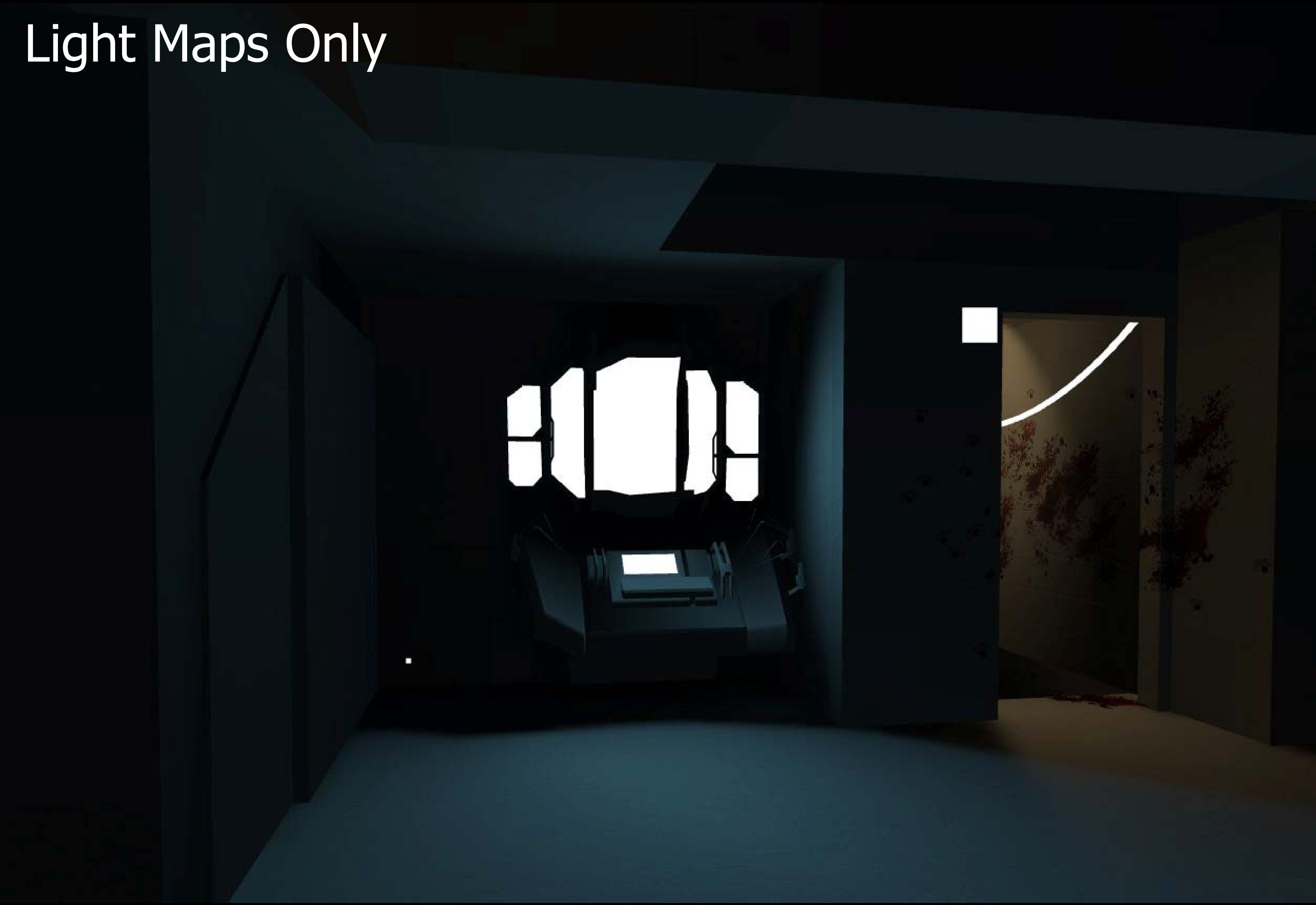
At the pixel level. . .

- Sample three light map colors, and blend between them based on the transformed vector

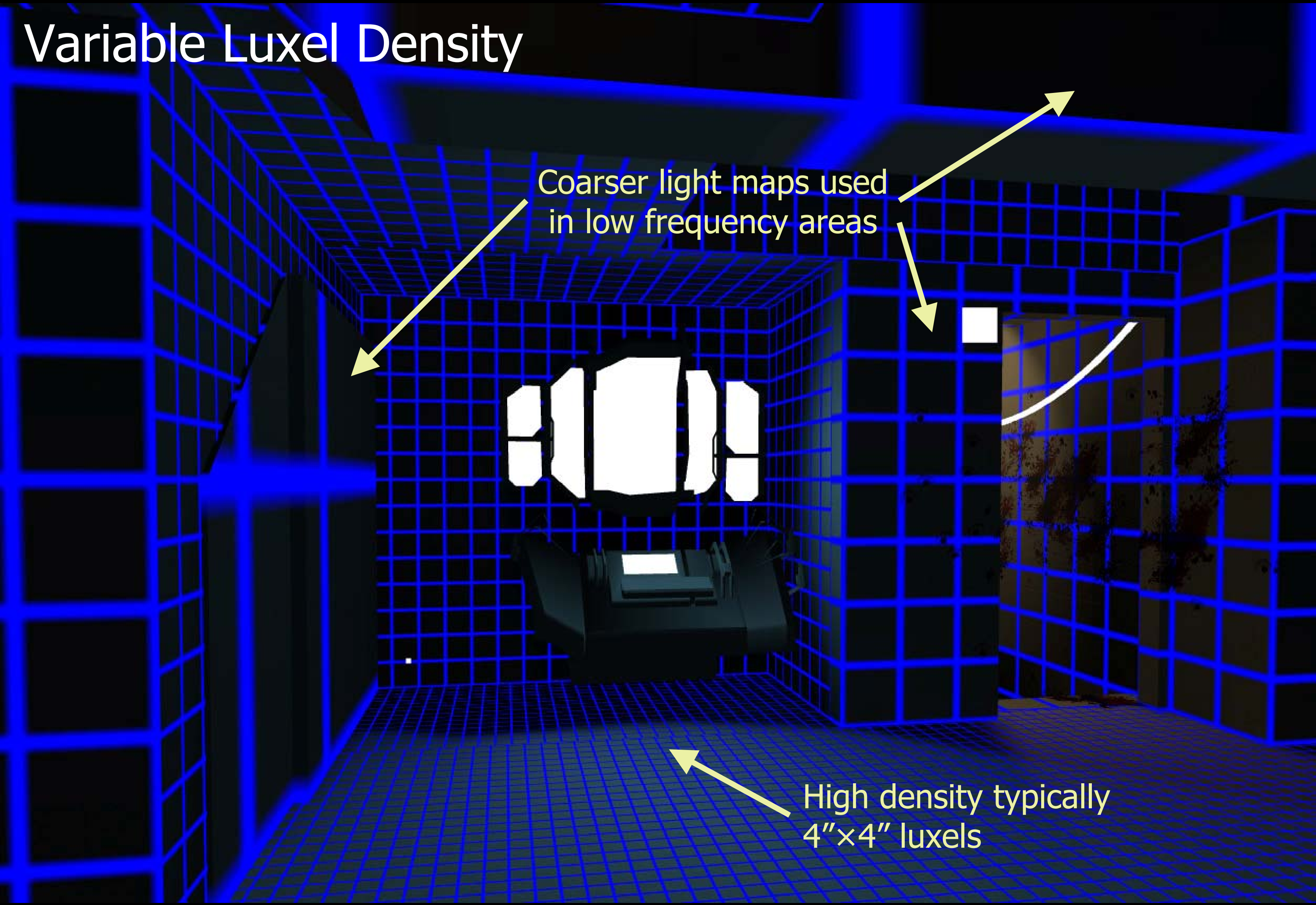
```
float3 dp;
dp.x = saturate( dot( normal, bumpBasis[0] ) );
dp.y = saturate( dot( normal, bumpBasis[1] ) );
dp.z = saturate( dot( normal, bumpBasis[2] ) );
dp *= dp;

diffuseLighting = dp.x * lightmapColor1 +
                  dp.y * lightmapColor2 +
                  dp.z * lightmapColor3;
```

Light Maps Only



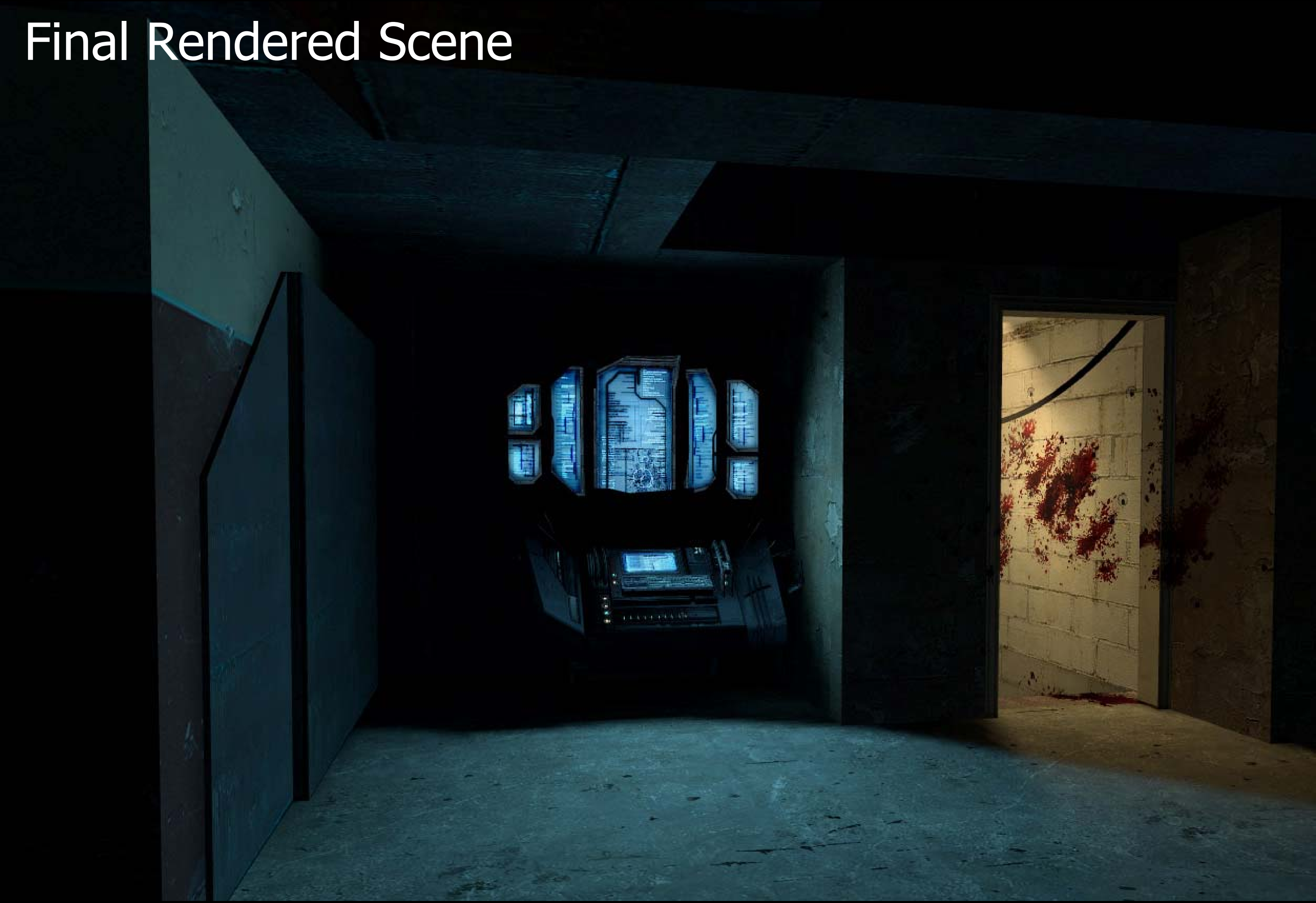
Variable Luxel Density



Coarser light maps used in low frequency areas

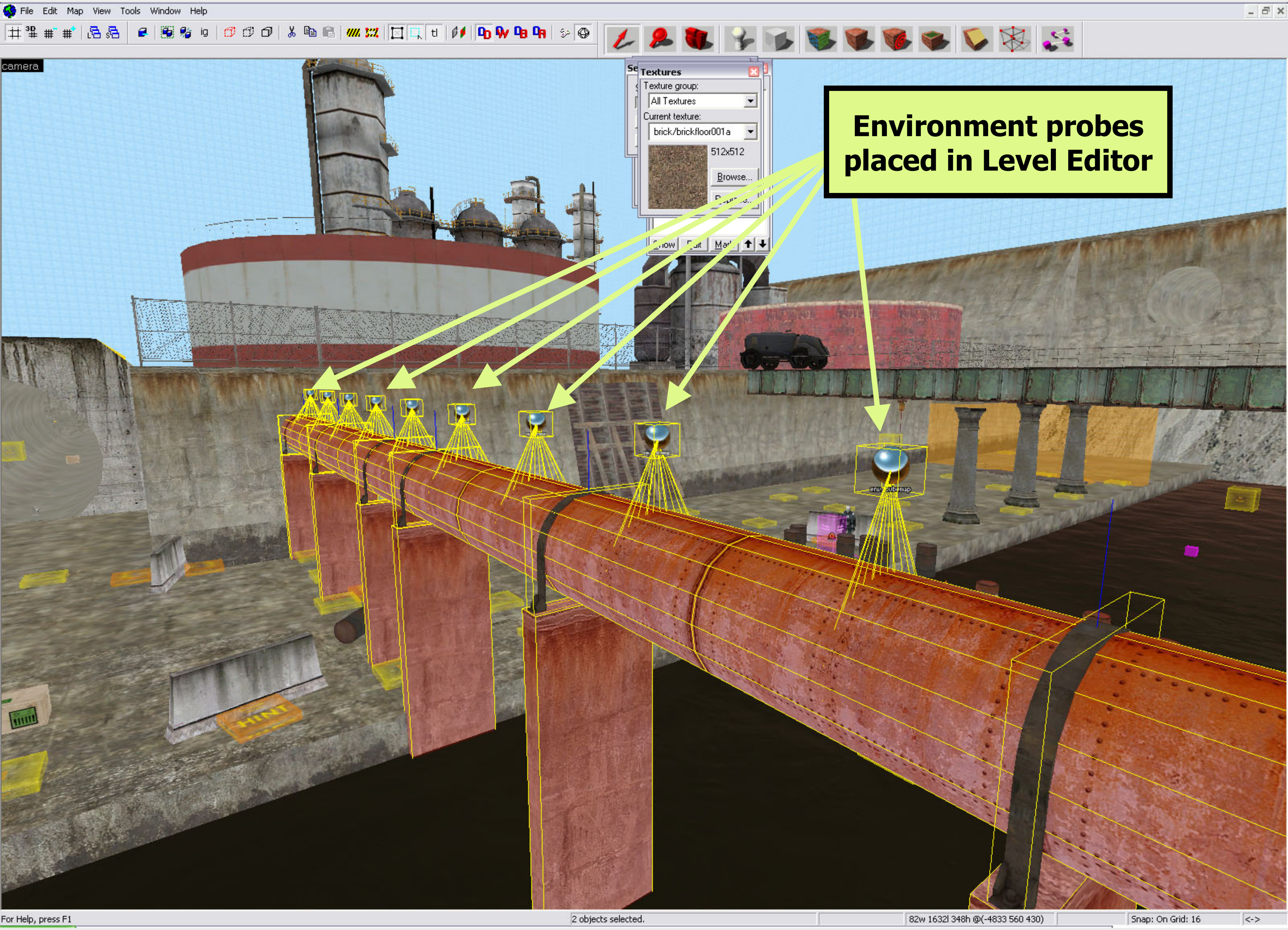
High density typically 4″×4″ luxels

Final Rendered Scene



World Specular Lighting

- We use cube maps for specular lighting the world
- Designers place the sample locations for specular lighting
- Cube maps are pre-computed in-engine from the level data using rendering
  - Especially interesting for HDR, which we'll discuss later
- World surfaces pick up the "best" cube map, or cube maps can be manually assigned to surfaces to fix boundary problems



Texture group: All Textures

Current texture: brick/brickfloor001a

512x512

Environment probes placed in Level Editor

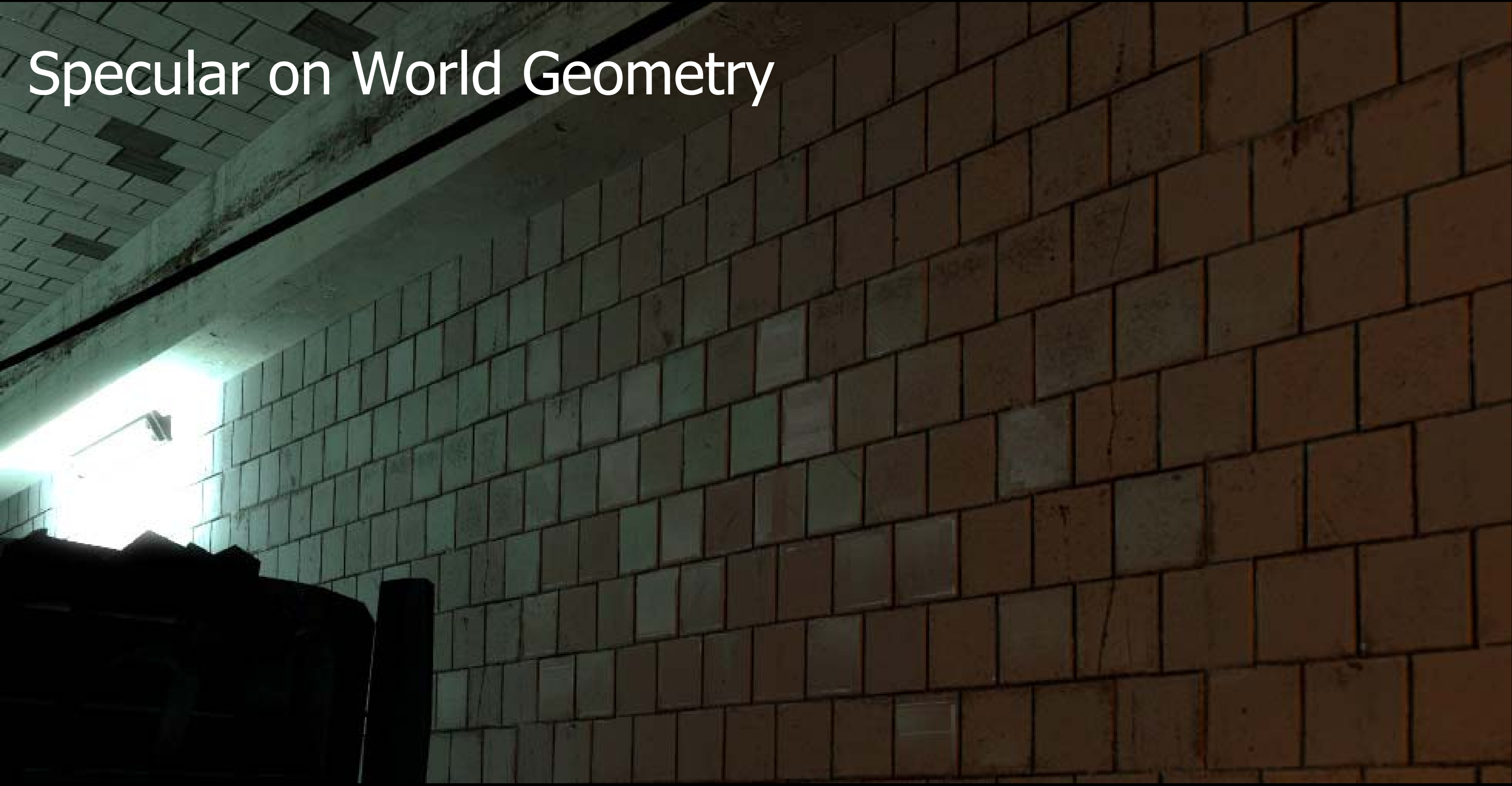
For Help, press F1

2 objects selected.

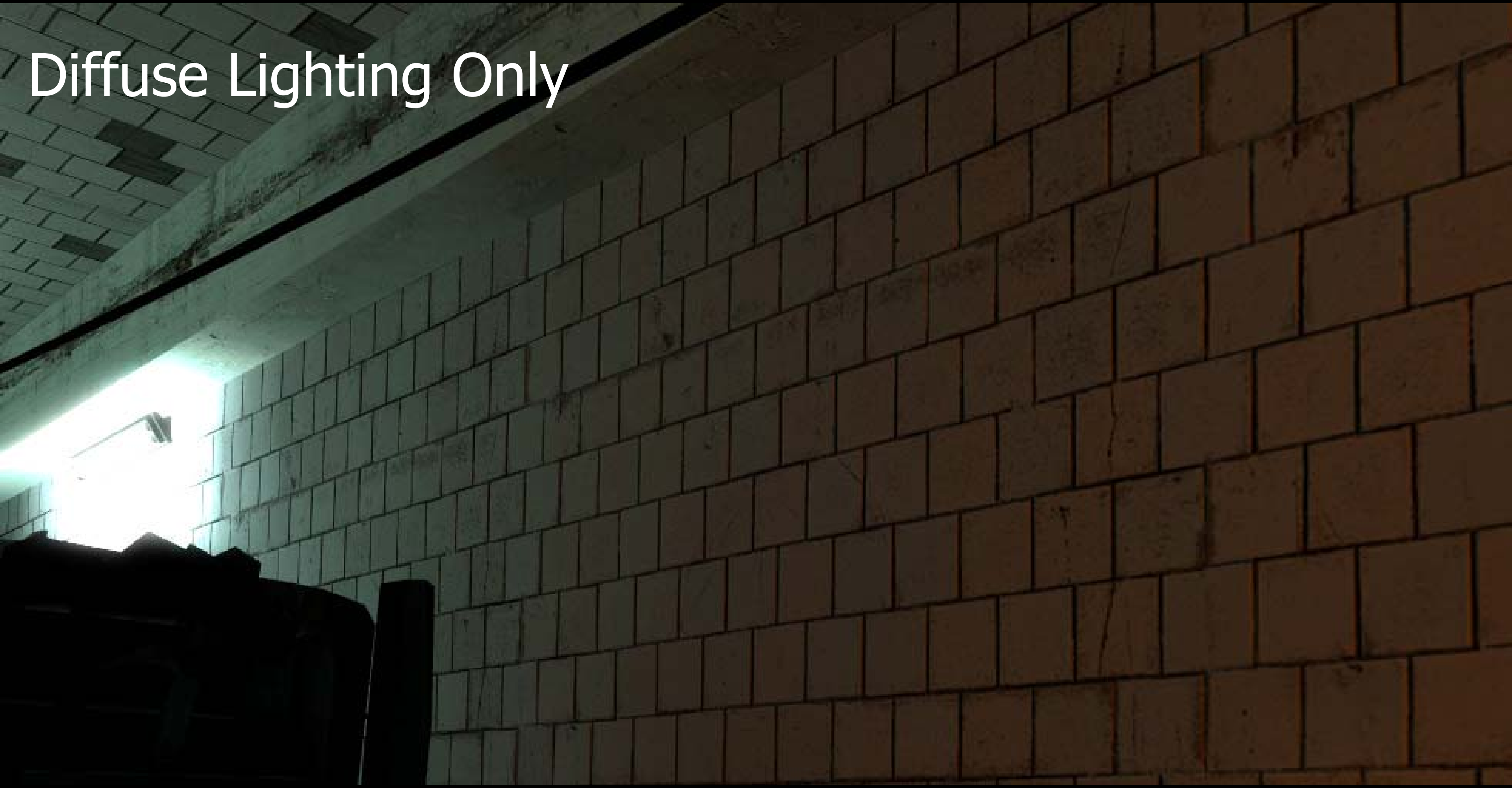
82w 1632l 348h @(-4833 560 430)

Snap: On Grid: 16

Specular on World Geometry



Diffuse Lighting Only



Specular from cube maps

Other World Lighting

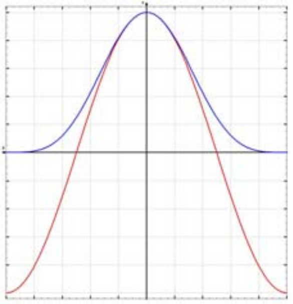
- There are a few other special cases
  - Reflective and refractive water
  - Special effects such as the Citadel Core in *Episode One*
  - Some simple geometry which is just vertex lit

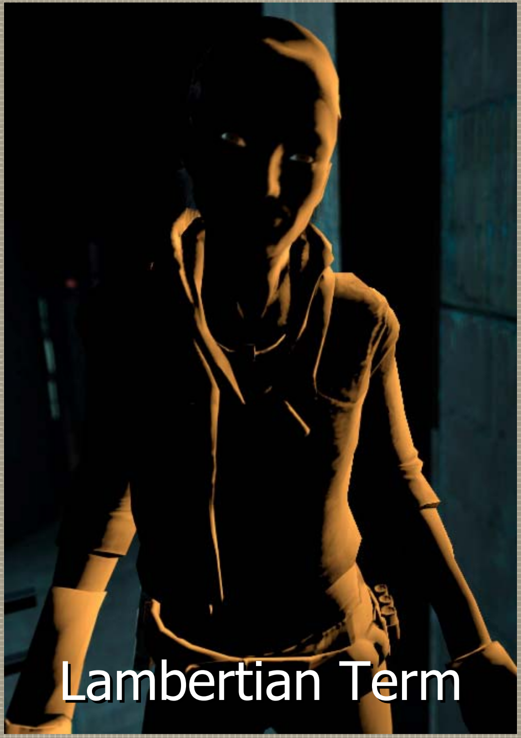
Model Lighting

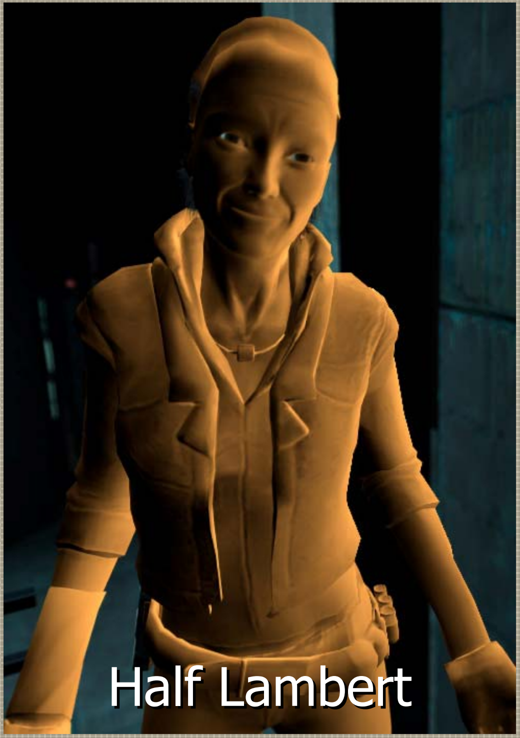
- Prior to *Episode One*, models (objects which aren’t part of the immovable *world* geometry) used primarily view-independent lighting terms:
  - Two local diffuse lights with Half-Lambert
  - Directional ambient term (ambient cube)
- Also cubic environment mapping terms
- These terms maxed out ps_2_0, which was our high end when we shipped *Half-Life 2* in November 2004

Half Lambert

- Typically clamp N·L to zero at the terminator
- Half Lambert scales the -1 to 1 cosine term (red curve) by ½, biases by ½ and squares to pull the light all the way around (blue curve)
- Another example of choosing a forgiving lighting model
- Similar to Exaggerated Shading apaper by Rusinkiewicz et al

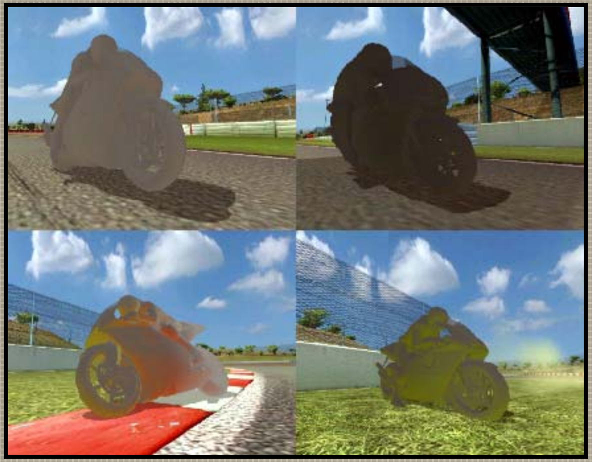



Lambertian Term


Half Lambert

Indirect Illumination in Games

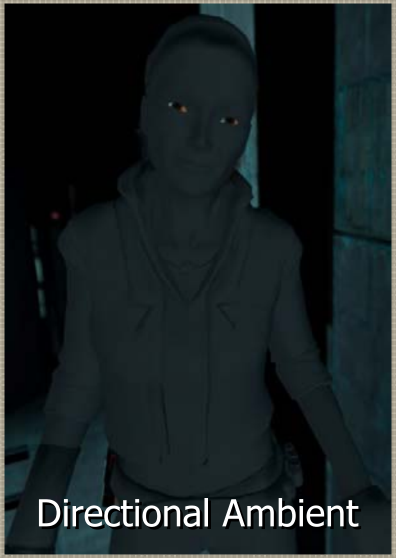
- Quake III used a filtered sample from a 3D array of ambient terms to provide a constant ambient for a given character [Hook99]
- MotoGP used a hemisphere lighting model where bounced light from the ground varied spatially throughout the race track [Hargreaves03]
- Max Payne 2 stored linear 4-term spherical harmonic coefficients for use in lighting characters and other moving geometry [Lehtinen06]



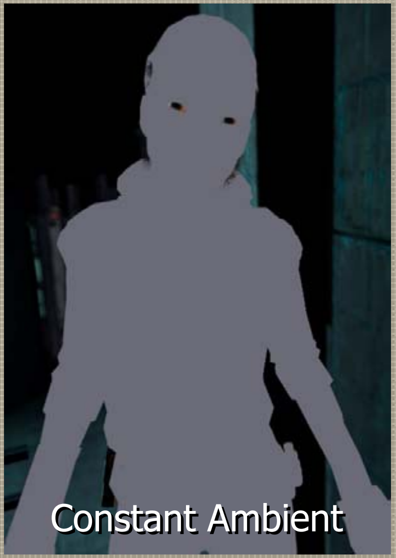
From [Hargreaves03]

Ambient Cube

- Pre-compute irradiance samples throughout the environment
- 4′×4′×8′ density
- The set of samples is referred to in the literature as an *irradiance volume* [Greger98] and each sample defines an irradiance environment map [Ramamoorthi01]



Directional Ambient

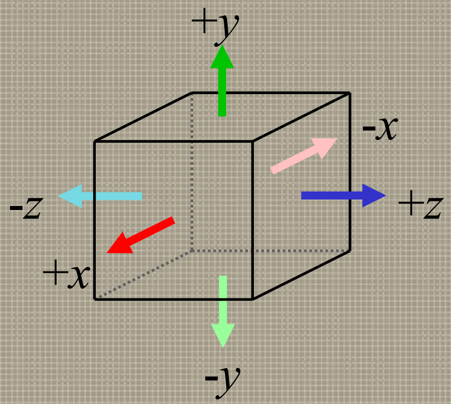


Constant Ambient

- Directional ambient term which includes indirect light
- Lights beyond the first two can be added to the ambient cube

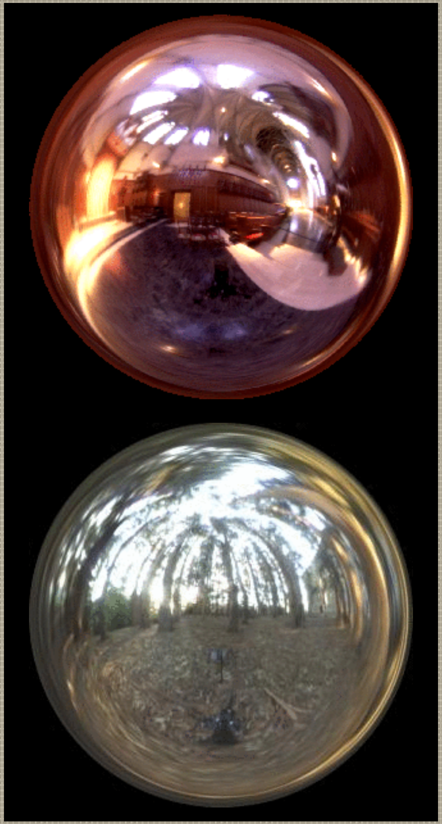
Ambient Cube Basis

- Six RGB lobes stored in shader constants
- More concise basis than first two orders of spherical harmonics (nine RGB colors)
- Pixel shader constant store is currently a bottleneck, though this eases quite a bit on ps_3_0
- Developed in parallel with other techniques such as [Ramamoorthi01]

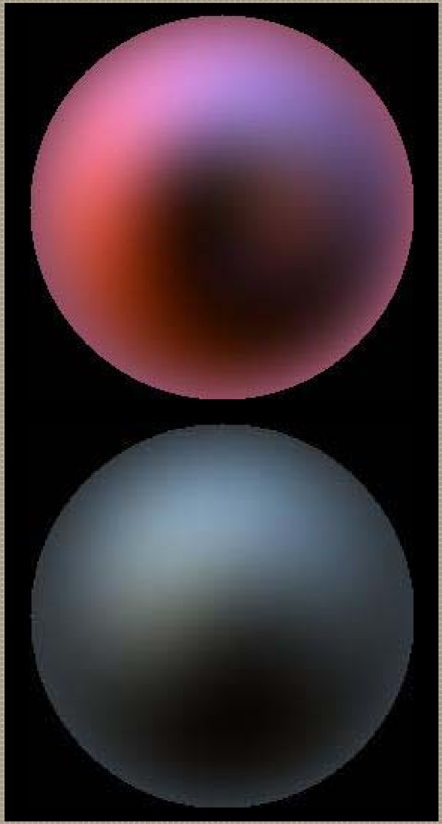


```
float3 AmbientLight( const float3 worldNormal )
{
  float3 nSquared = worldNormal * worldNormal;
  int3 isNegative = ( worldNormal < 0.0 );
  float3 linearColor;
  linearColor = nSquared.x * cAmbientCube[isNegative.x]   +
                nSquared.y * cAmbientCube[isNegative.y+2] +
                nSquared.z * cAmbientCube[isNegative.z+4];
  return linearColor;
}
```

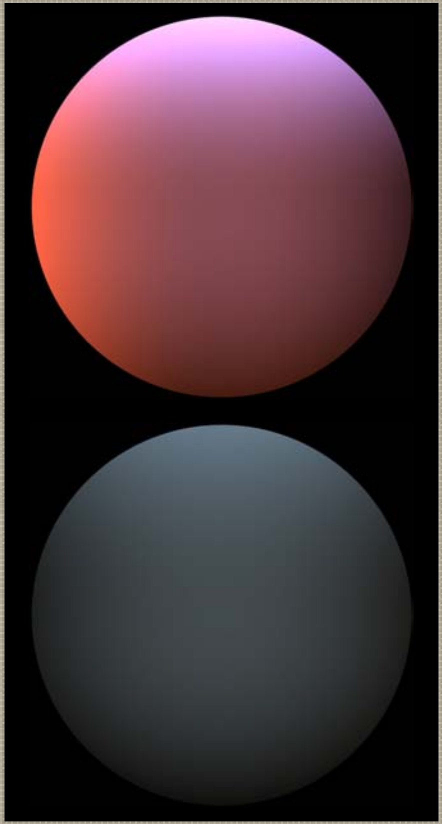
Comparison of Ambient Cube to Spherical Harmonics



Original Light Probes

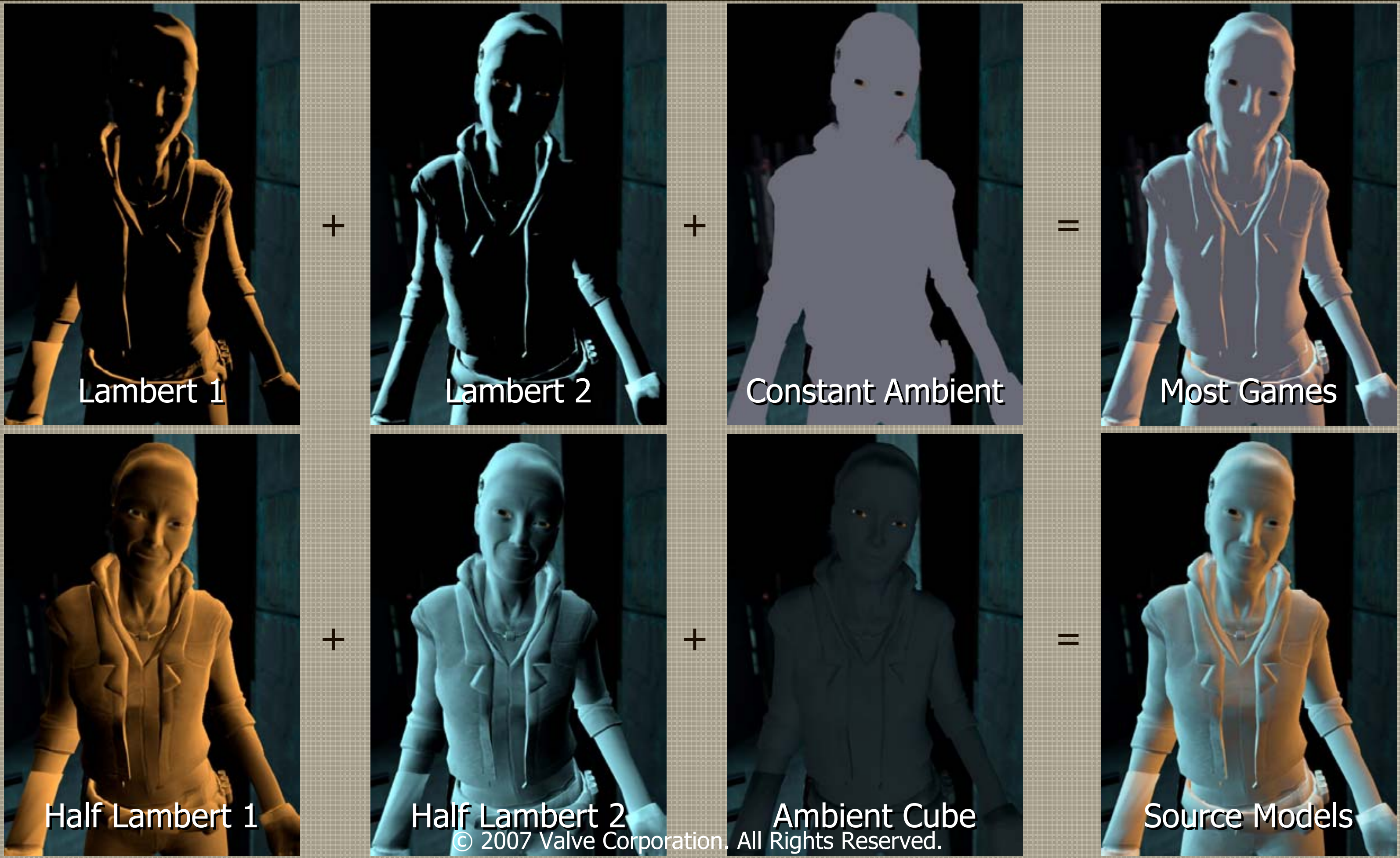


[Ramamoorthi01]

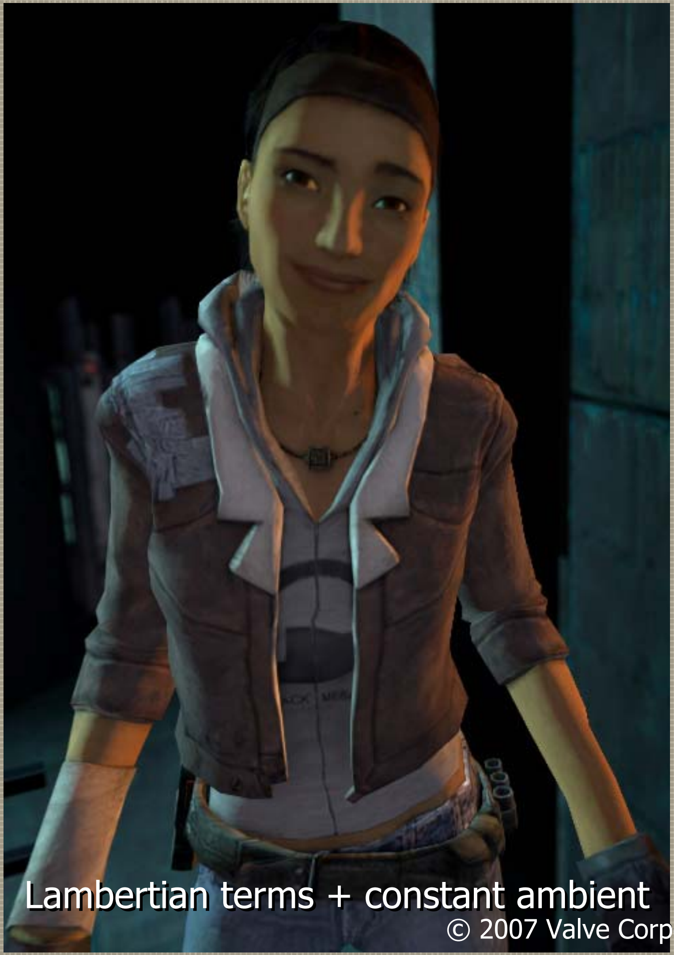


Valve Ambient Cubes

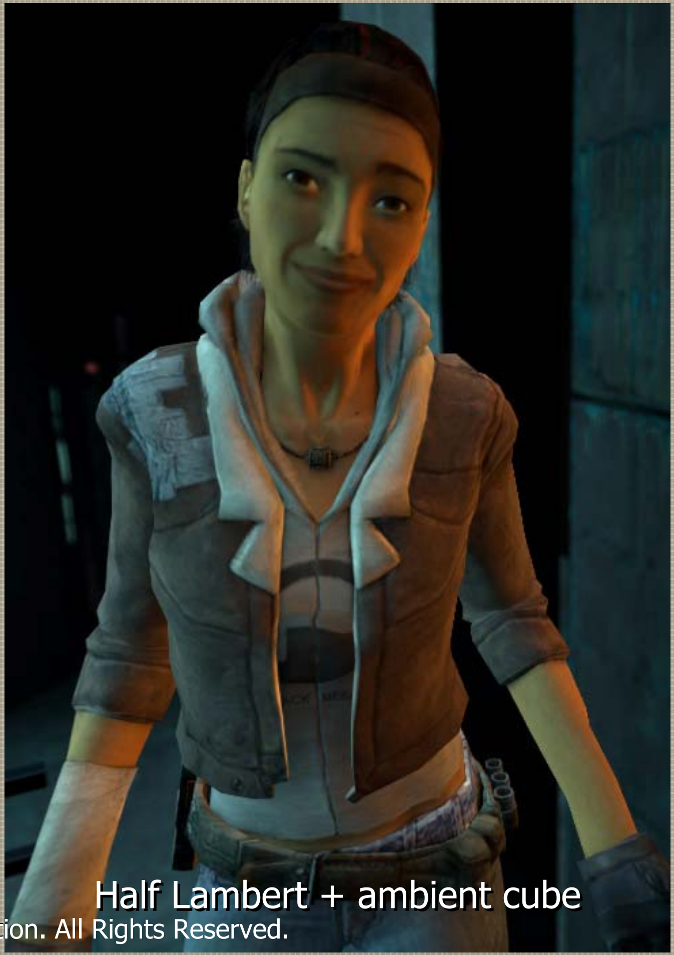
Benefits of Half Lambert and Ambient Cube



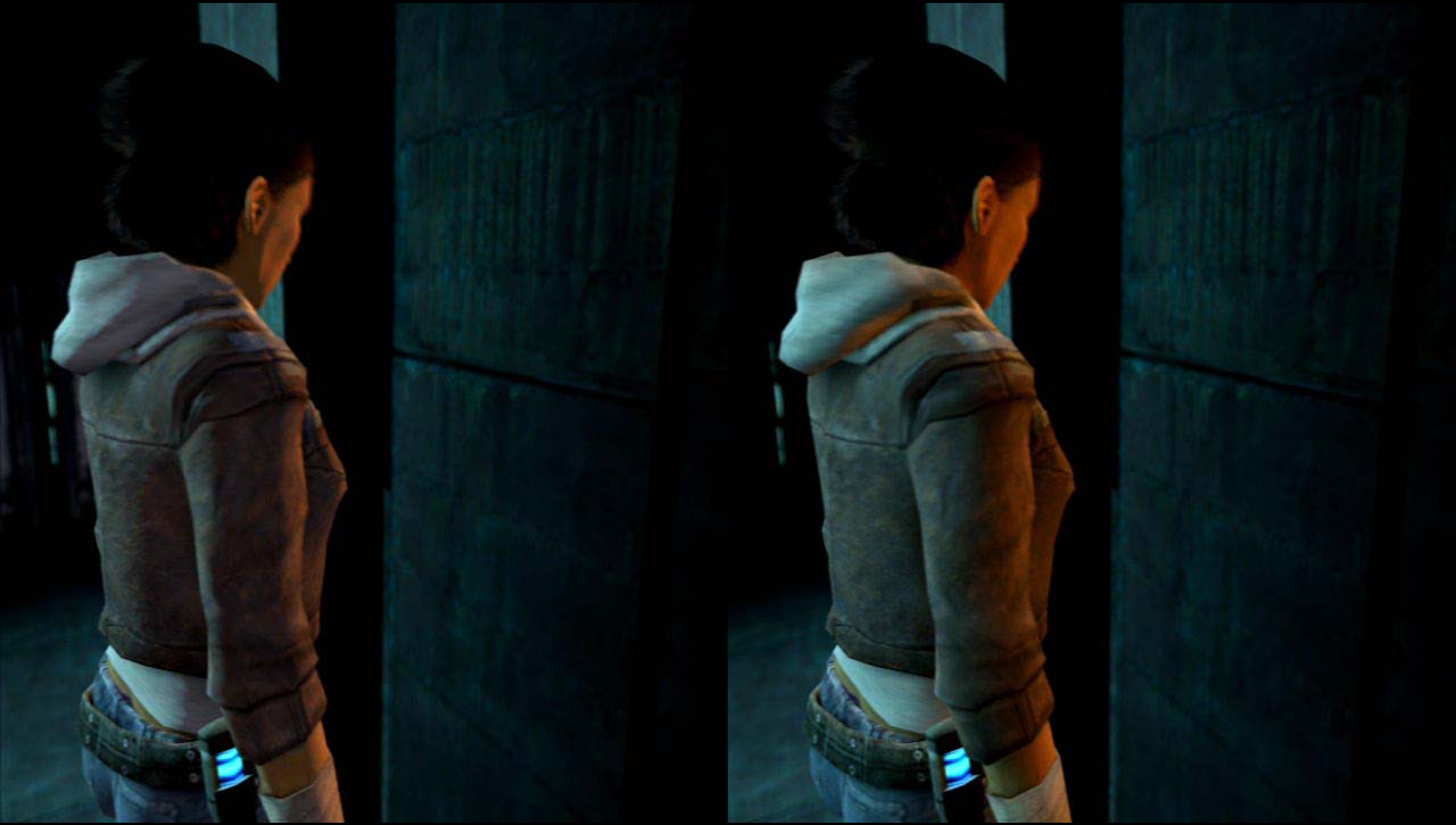
Benefits of Half Lambert and Ambient Cube



Lambertian terms + constant ambient



Half Lambert + ambient cube



Extensions to Earlier Model Lighting

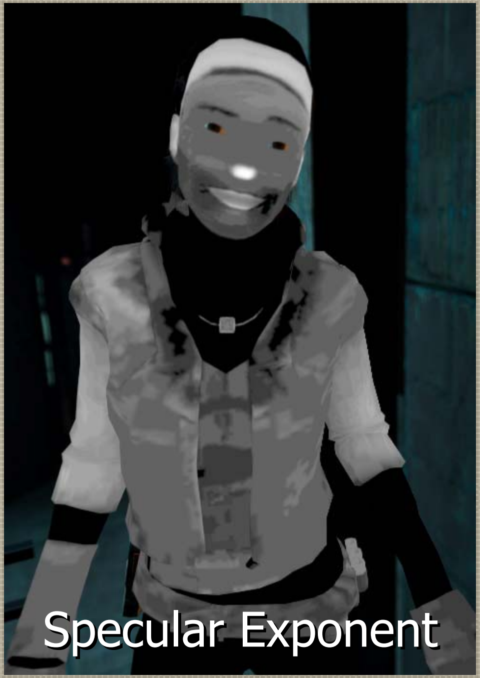
- When we shipped Half-Life 2, ps_2_0 was the most advanced pixel shader model available
- In the spring of 2006, 40% of our users have hardware which supports ps_2_b or higher
  - Of the people who have played *Episode One* in the two months since it has shipped, **62%** have such hardware
- For *Episode One* and a recent update to *Day of Defeat: Source*, we used these longer ps_2_b shaders to add Phong terms to model lighting

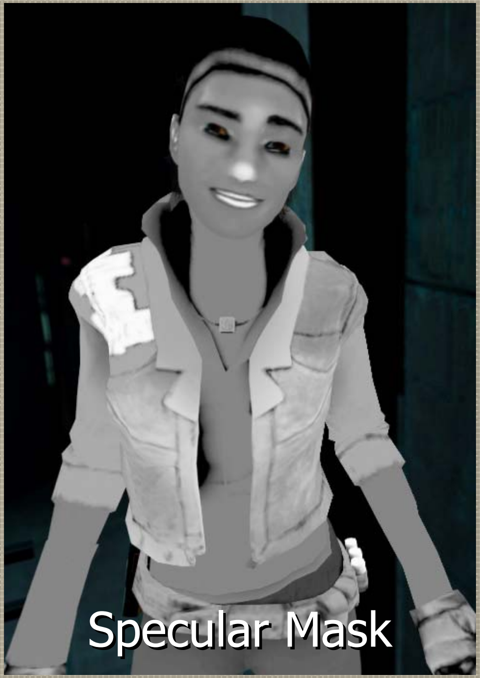
ps_2_b

- Shader model on ATI RADEON X*x*00 parts
- Same programming model as ps_2_0 but with longer shaders
  - ps_2_0 – 64 alu and 32 texture
  - ps_2_b – 512 total…any mix of alu and texture
- ATI RADEON X*x*00 & X1*x*00 series
  NVIDIA GeForce 6*x*00 & 7*x*000 series

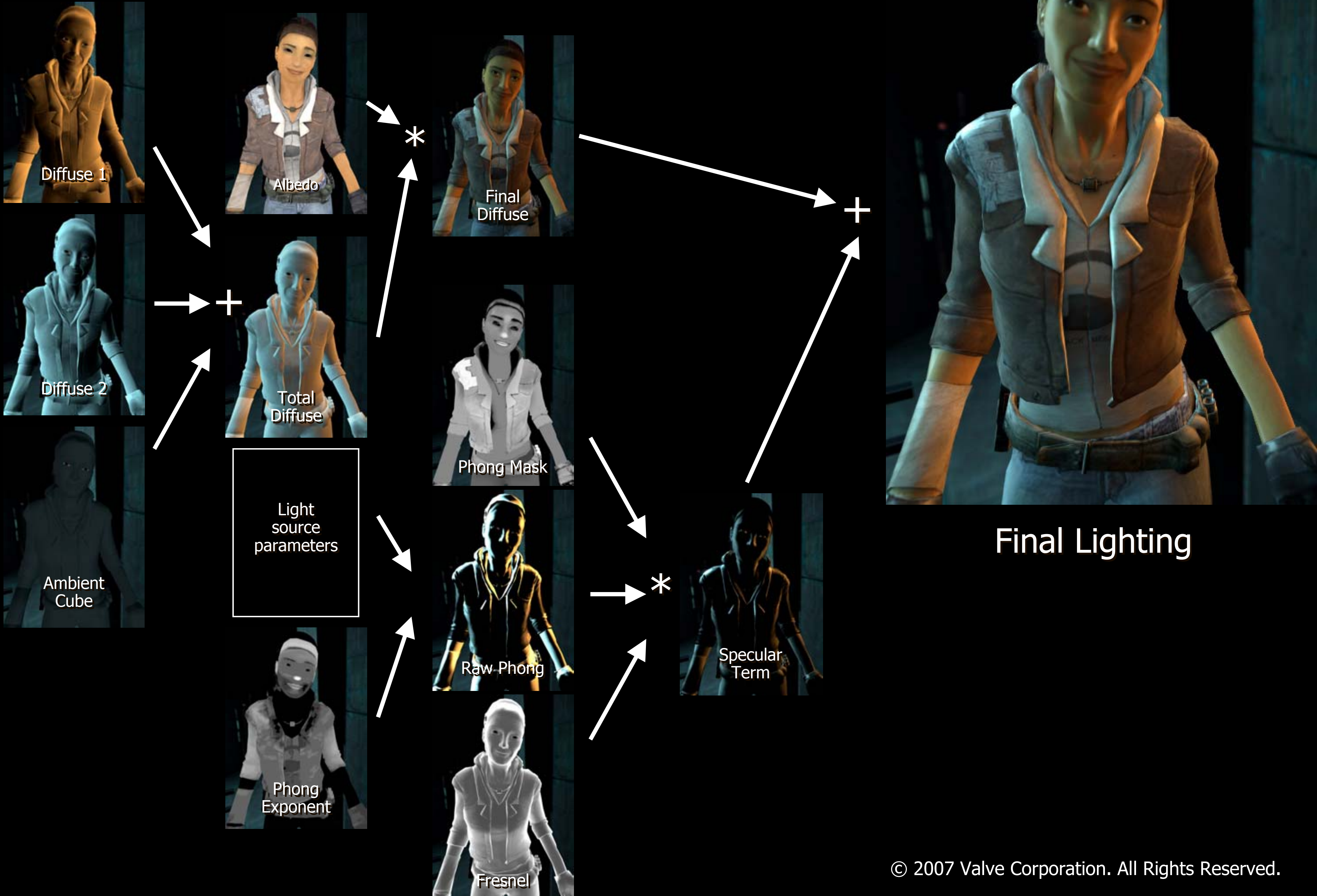
Phong Terms

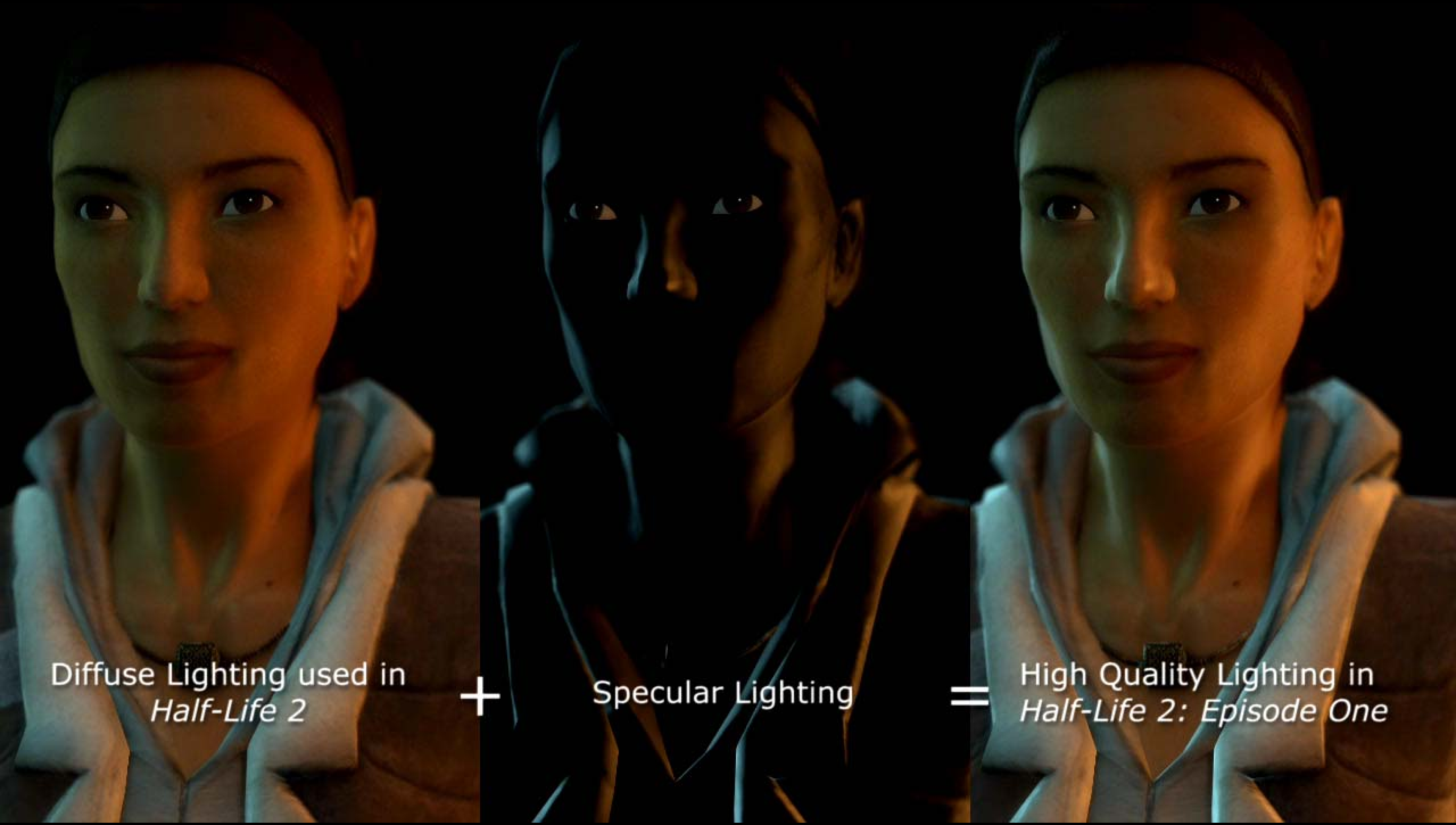
- Compute Phong terms for up to two local lights
- Can specify specular exponent variations across a model
  - Currently, map blends between specular exponent of 1 and 150
- Modulate with scalar mask channel and tweakable Fresnel term


Specular Exponent

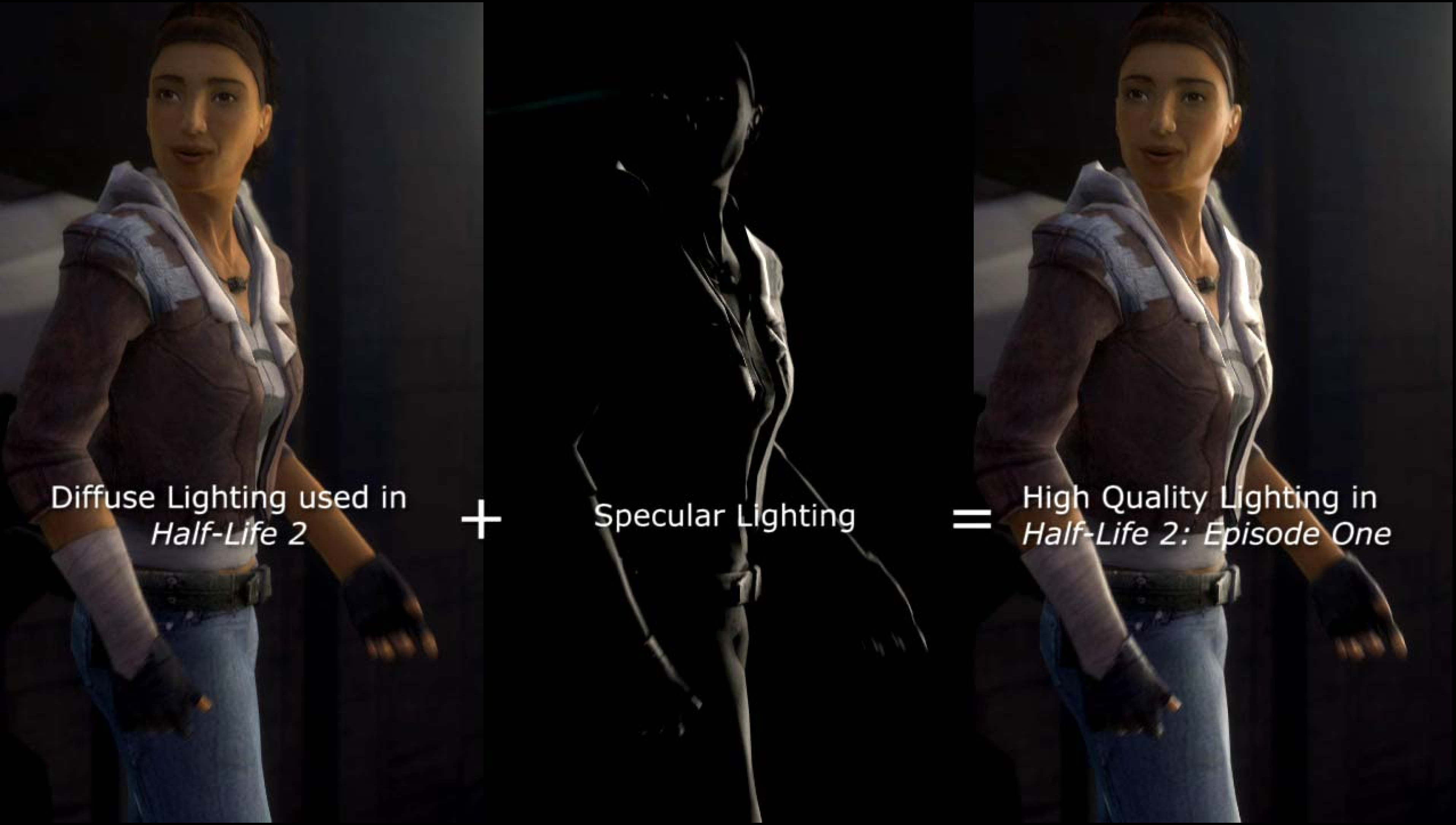

Specular Mask

Episodic Alyx Shade Tree





Diffuse Lighting used in *Half-Life 2* + Specular Lighting = High Quality Lighting in *Half-Life 2: Episode One*



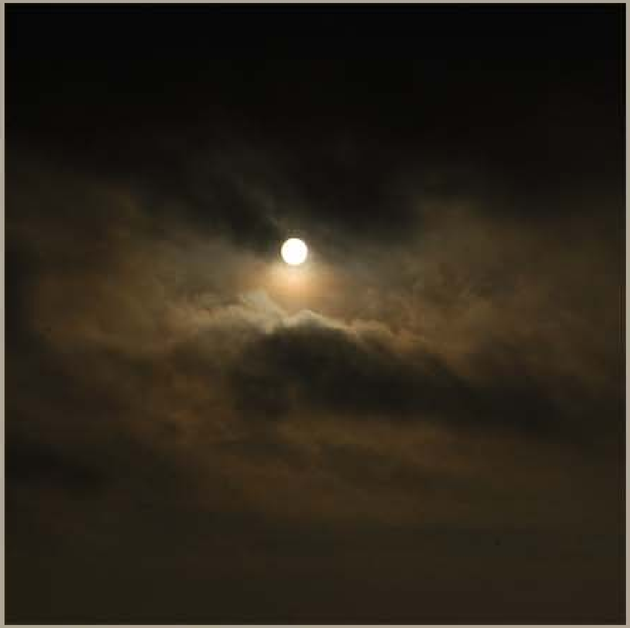
Diffuse Lighting used in *Half-Life 2* + Specular Lighting = High Quality Lighting in *Half-Life 2: Episode One*

High Dynamic Range Rendering in the Source Engine

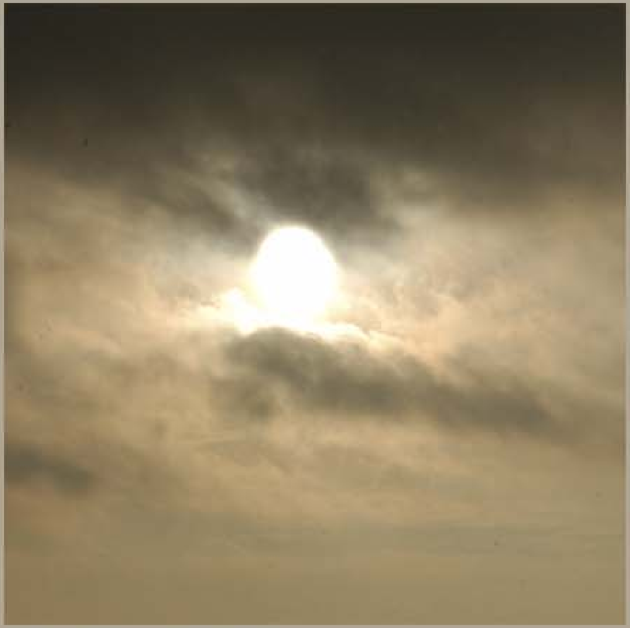
High Dynamic Range

- A High Dynamic Range image is an image that has a greater contrast range than can be shown on a standard display device, or that can be captured with a standard camera with just a single exposure
- High Dynamic Range rendering performs intermediate operations such as shading in high dynamic range space
- The rendered HDR results are then mapped to a limited-range display

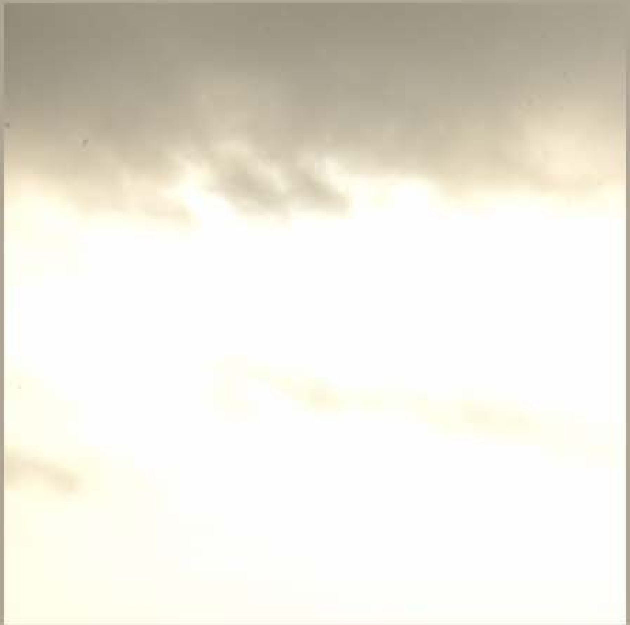
Real-World Sky at Multiple Exposures



f22 @ 1/1600th second

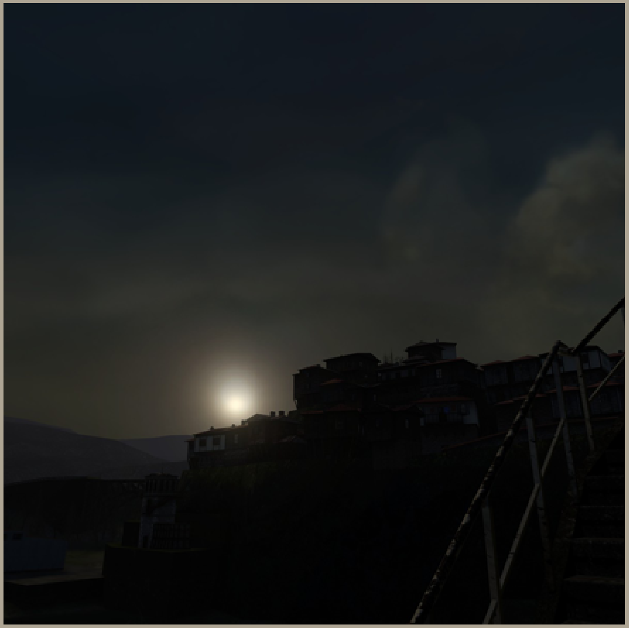


f22 @ 1/250th second

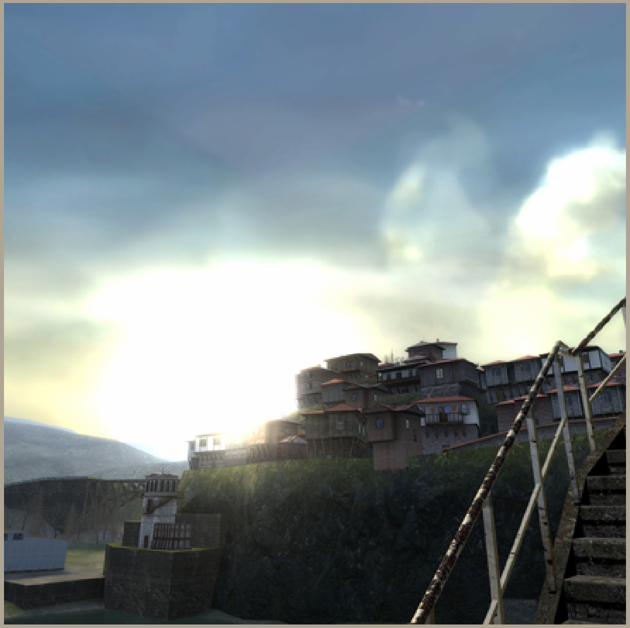


f22 @ 1/40th second

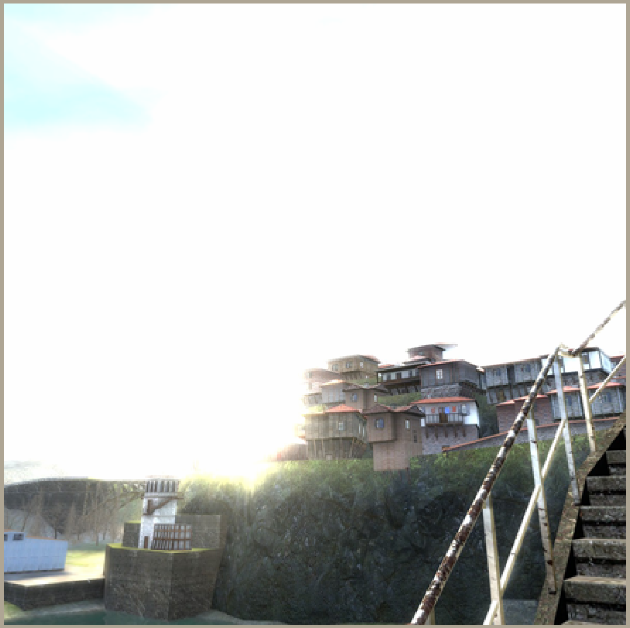
Scene from *Lost Coast* at Multiple Exposures



Tonemap scale = 0.05



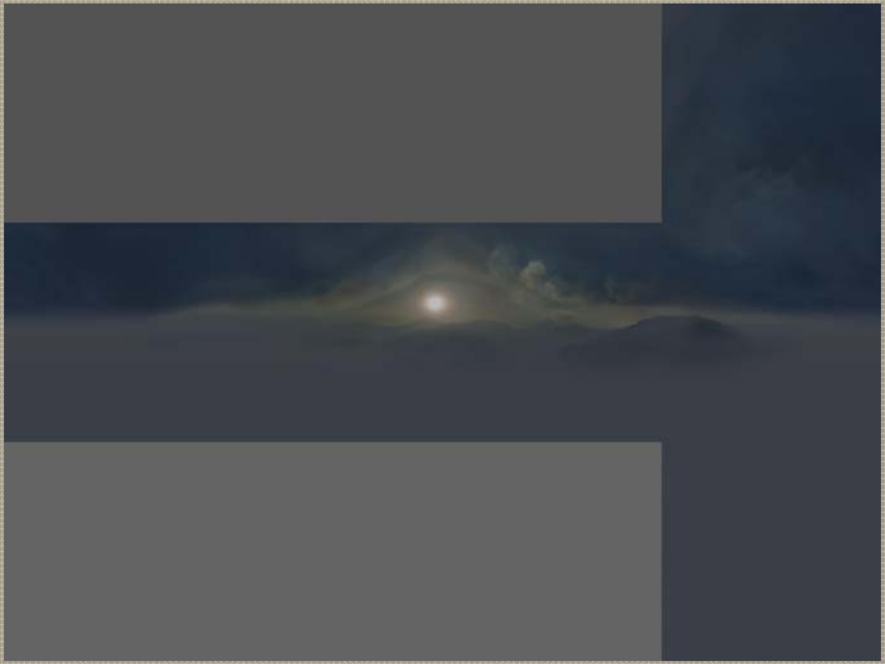
Tonemap scale = 1

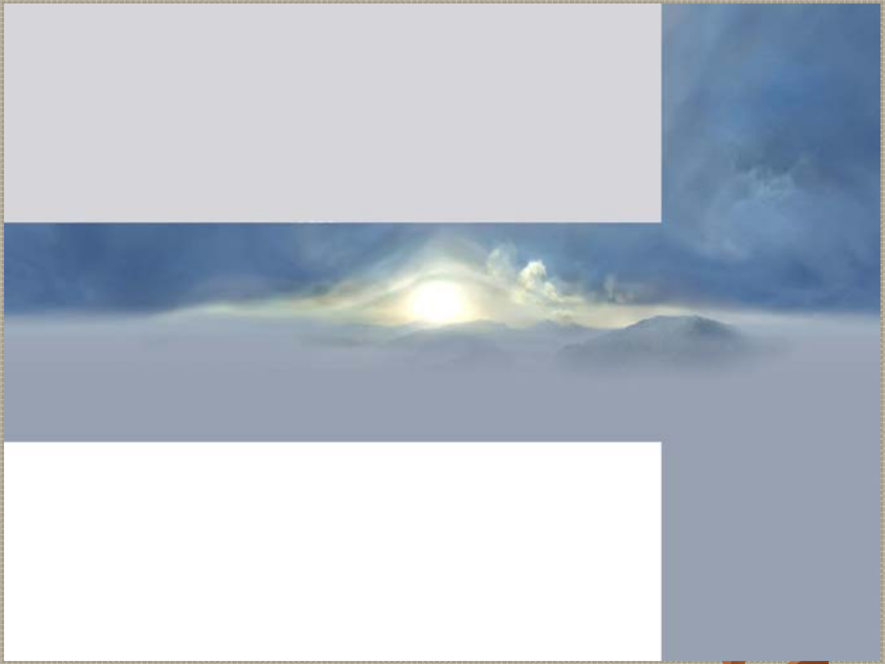


Tonemap scale = 4

HDR Skybox

- Authored by painting multiple exposures of the sky to allow for real-time exposure adjustment





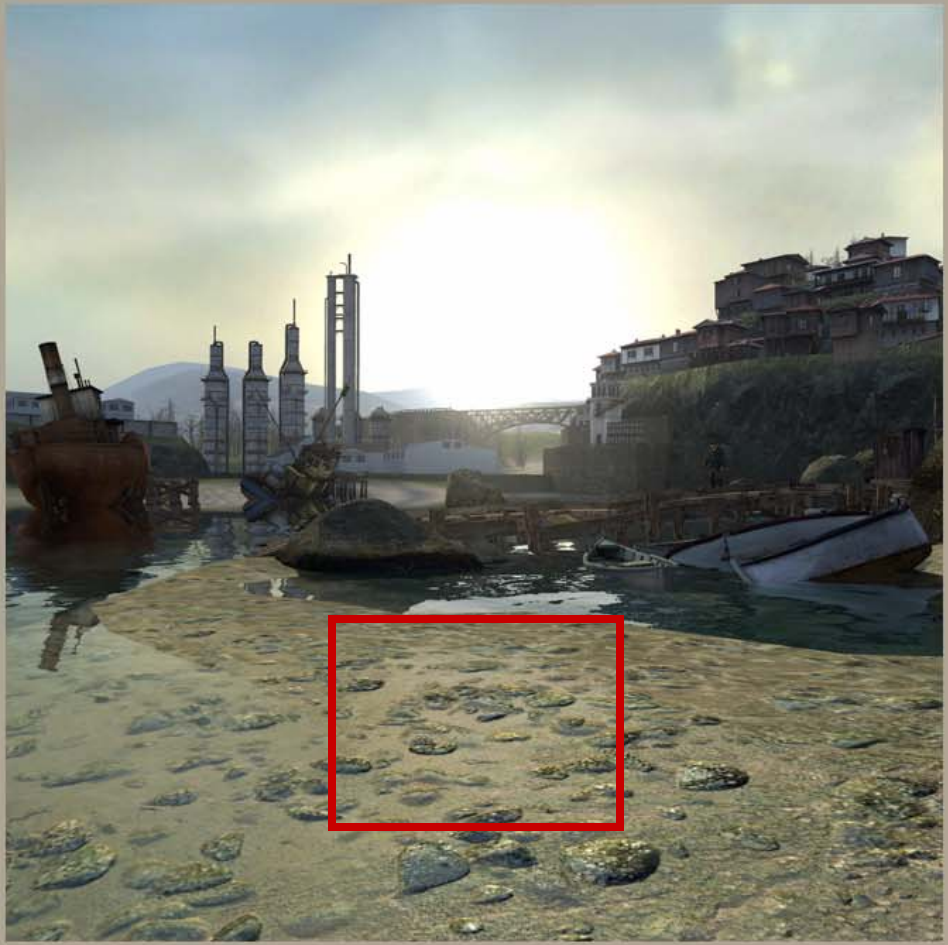
Authored Sources of HDR Illumination

- HDR sky box authored by painting multiple exposures of the sky
  - Start with seamless LDR sky texture and paint HDR information from there
  - We use a combination of Photoshop and HDRShop to make edits / touchups at different exposure levels
  - The industry has not addressed this issue adequately at this point and there is an opportunity for a tool vendor to step up and fix this
- Local lights with HDR values placed in levels

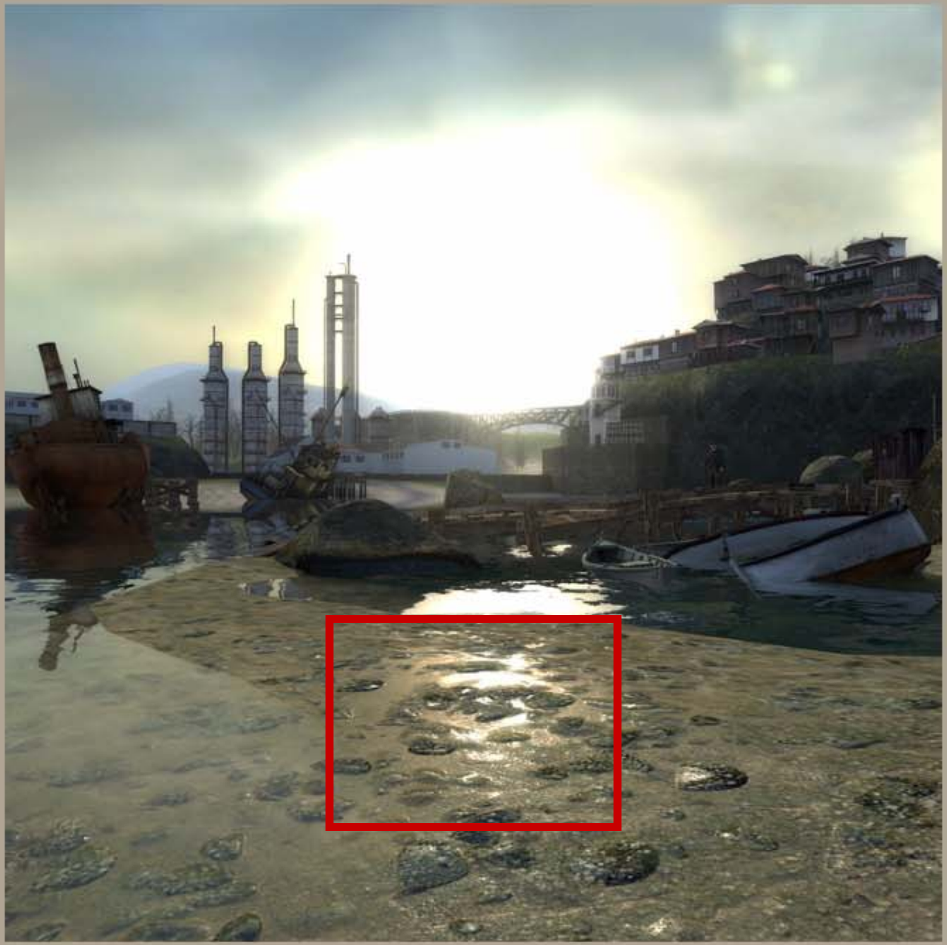
Generated Sources of HDR Illumination

- HDR light maps are generated offline using our radiosity solver and stored in RGBE format. At run time, these are stored in a 16bit per channel format
- HDR Cube maps are generated by the engine using the HDR skybox as well as the HDR light sources and the HDR light maps
  - The result is that an object which reflecting the sun or some other bright part of the scene, you will see this in the full effect of the brightness in the reflection, which is one of the key advantages of HDR rendering

HDR water reflection and refraction



LDR



HDR

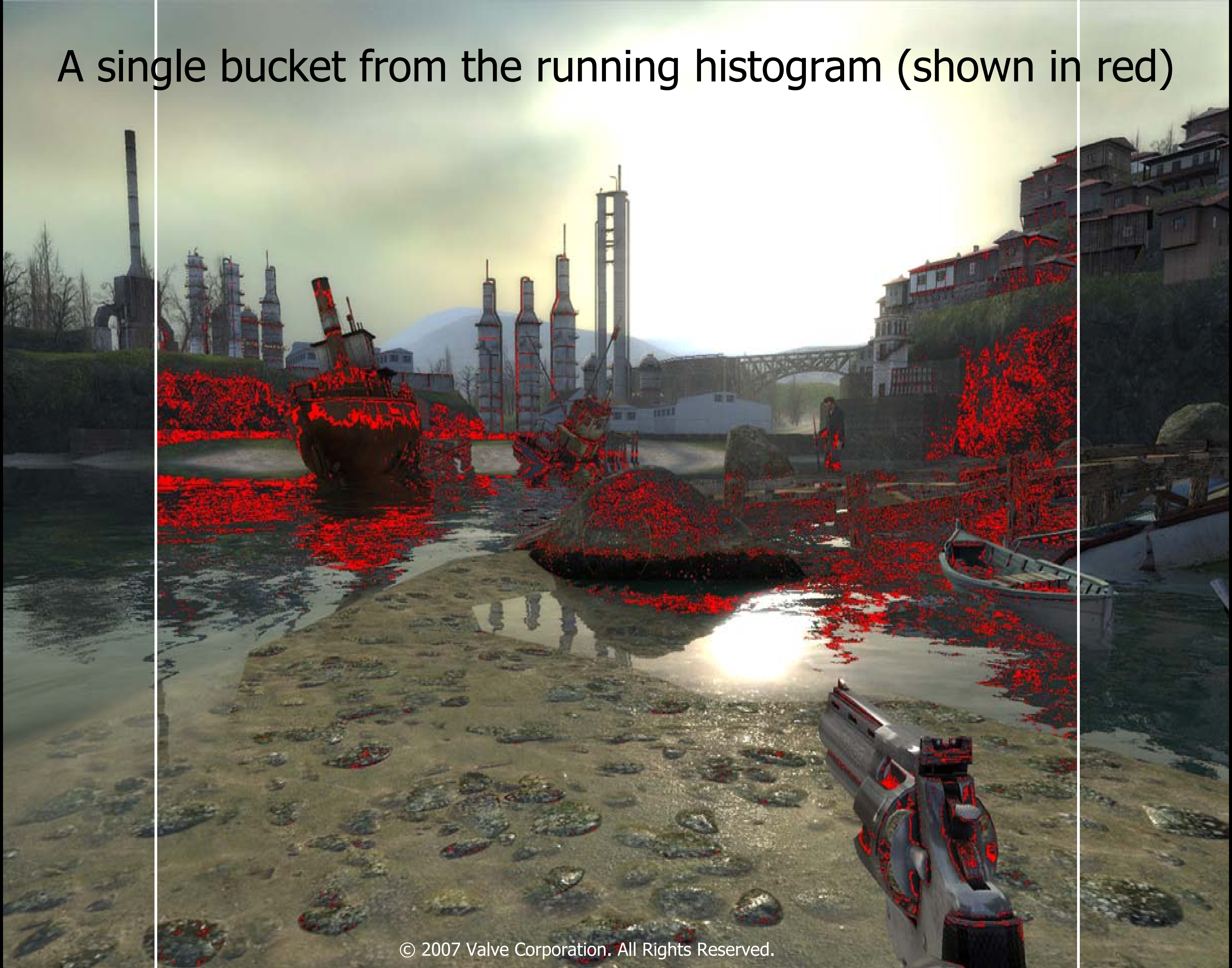
Tone Mapping

- Very careful with linear versus gamma color spaces
- All shaders in our DirectX 9 code path are single-pass
- As a result, tone mapping can be performed during normal rendering
  - Simple subroutine added to the end of all pixel shaders
- This allows us to use 32-bit RGBA render targets
  - No increase in render target memory footprint
  - MSAA supported
  - 60% of our total users see HDR rendering
    - 82% of those who have played *Episode One* so far!

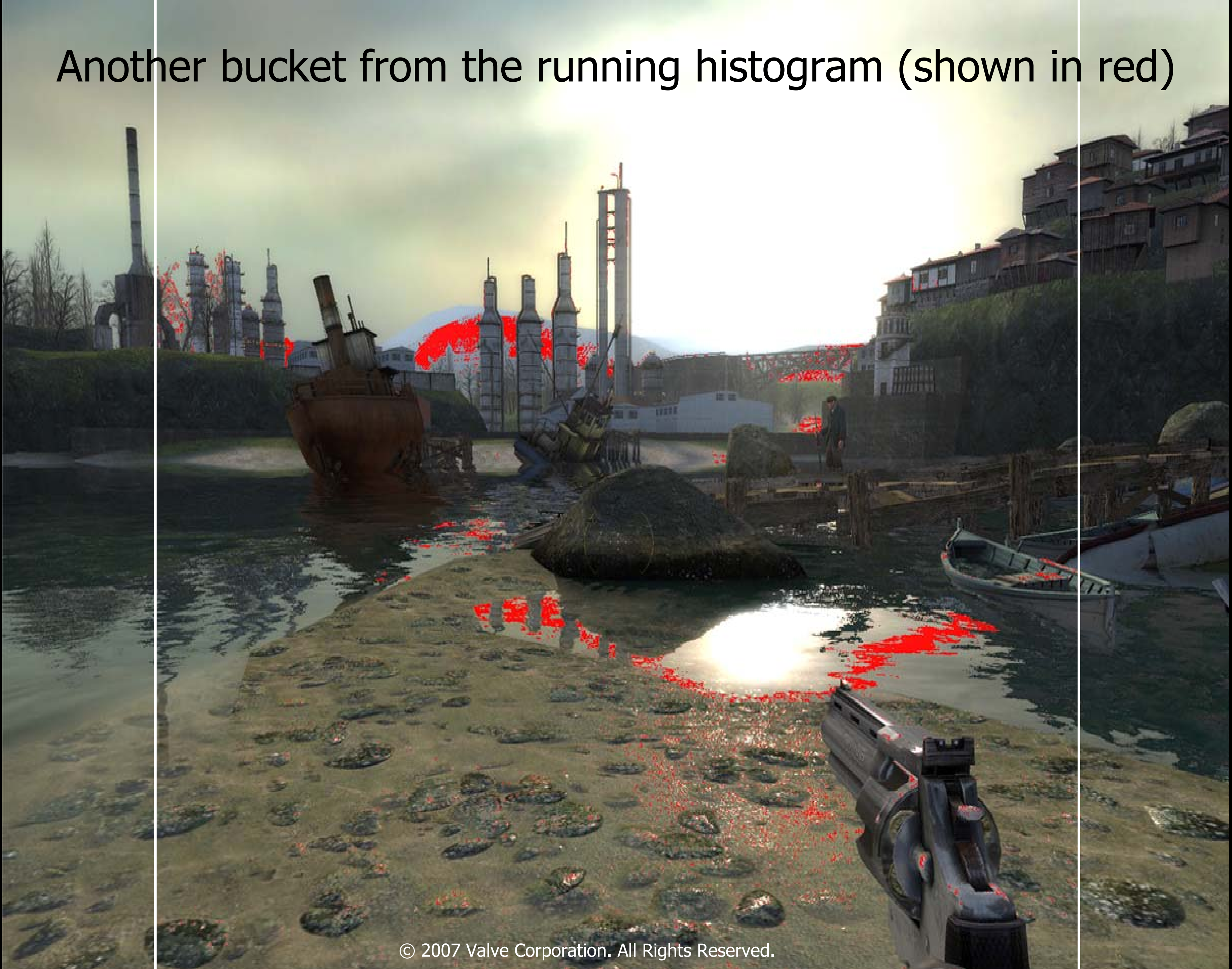
Auto Exposure

- Running luminance histogram of post-tonemapped frame
  - Update one bucket per frame
  - More histogram buckets for low luminance values
- Each frame, we determine which output pixels fall within a given range, tagging such pixels in the stencil buffer
- Use an asynchronous occlusion query to count pixels in range
- Full running histogram (not just a single average luminance) is available to the CPU with no stalls
- Time averaging is used smooth out auto exposure adjustments
- Designers can also modify auto exposure and bloom across different areas of a level or in connection with game events

A single bucket from the running histogram (shown in red)



Another bucket from the running histogram (shown in red)



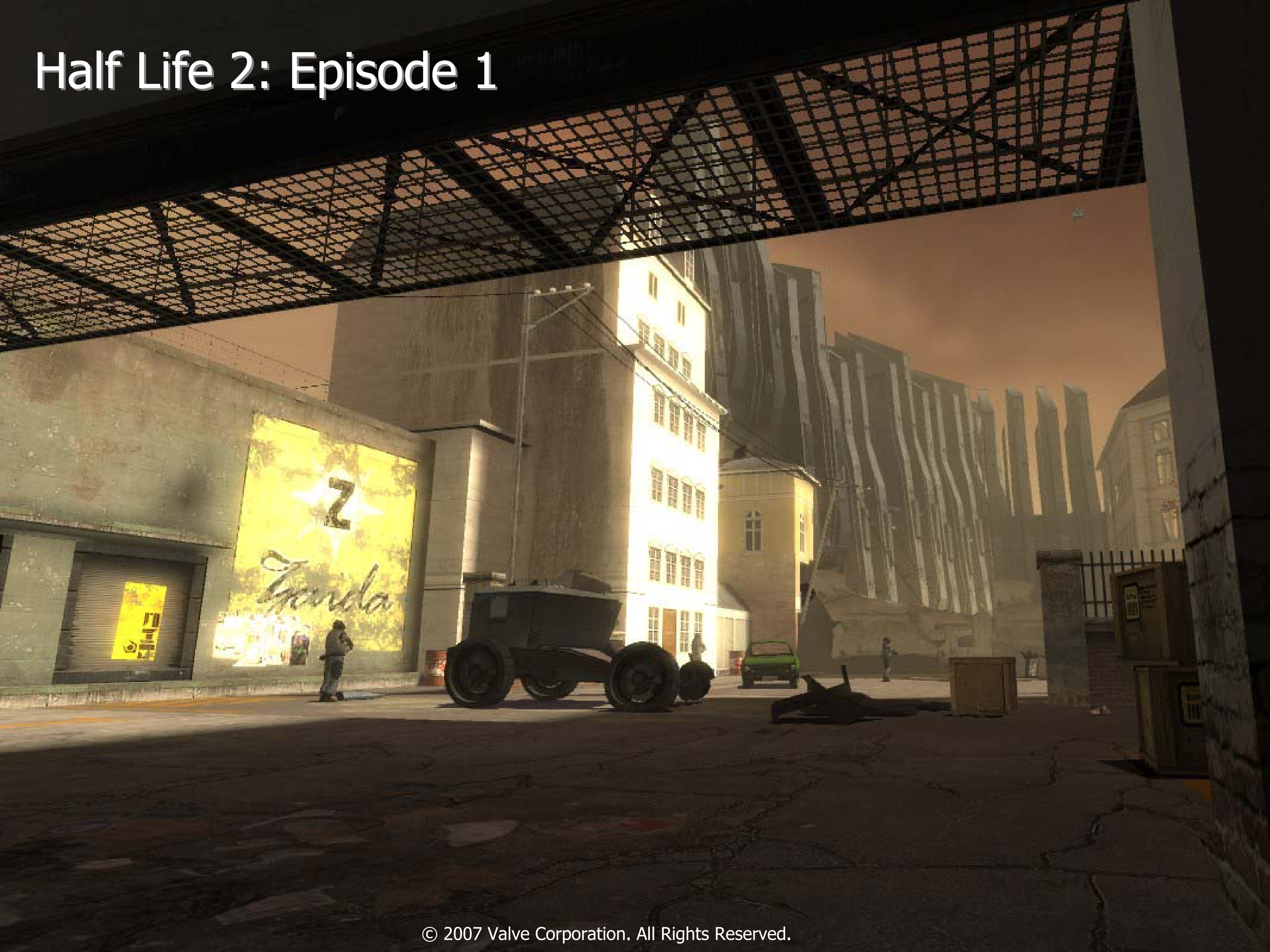
Implemented Multiple HDR Paths

- Floating point
  - HDR textures and render targets are fp16
  - Shipped this path, but it is not the one typically used
- Integer
  - Render to 32 bit RGBA integer frame buffer
  - Store HDR textures in fp16 (linear color space) when filtered fp16 textures are supported.
  - Store HDR textures in 4.12 linear color space otherwise.
  - Had to implement this version along with floating point version to support more hardware

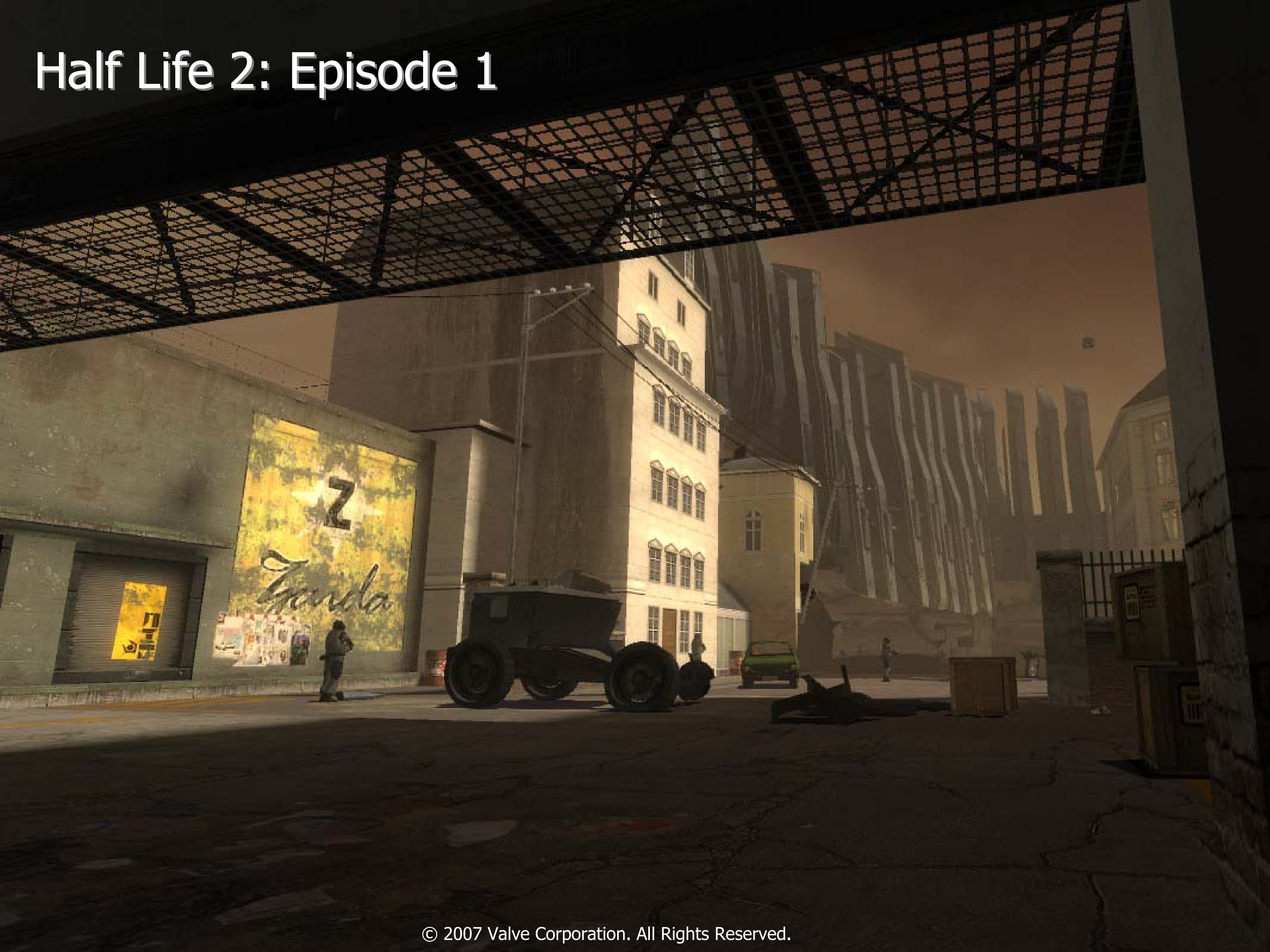
Float vs. Integer HDR Tradeoffs

- Floating Point
  - + Pre-tone-mapping data available for blooming
  - + General refraction mapping techniques preserve HDR information
  - - Requires floating-point alpha blending
  - - Hardware lacks precision to perform fixed function pixel fog
- Integer
  - + Compatible with Multisample Antialiasing (MSAA)
  - + Works on all DirectX 9 hardware (runs well even on ATI RADEON 9600)
  - + Small performance hit relative to LDR

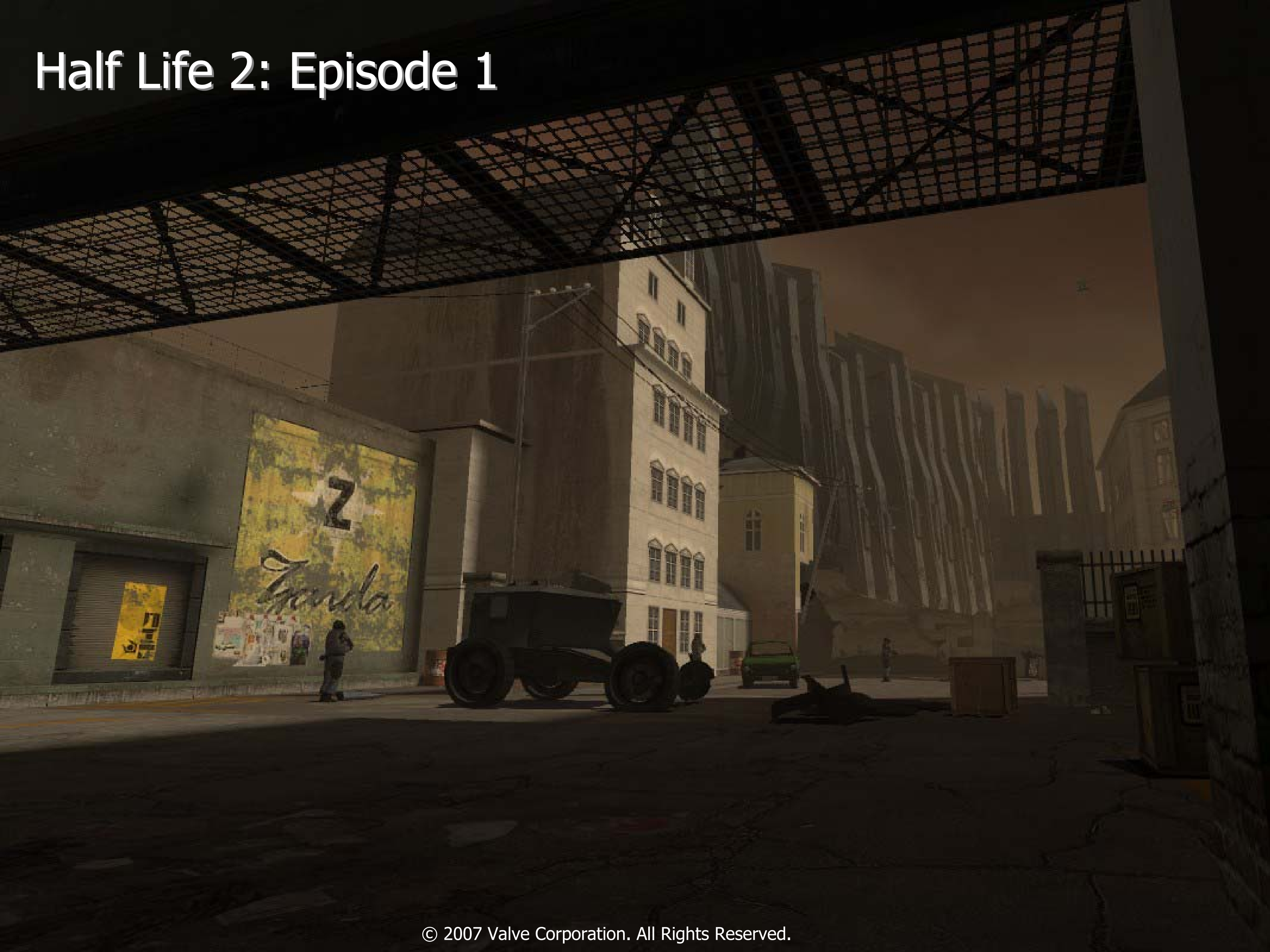
Half Life 2: Episode 1



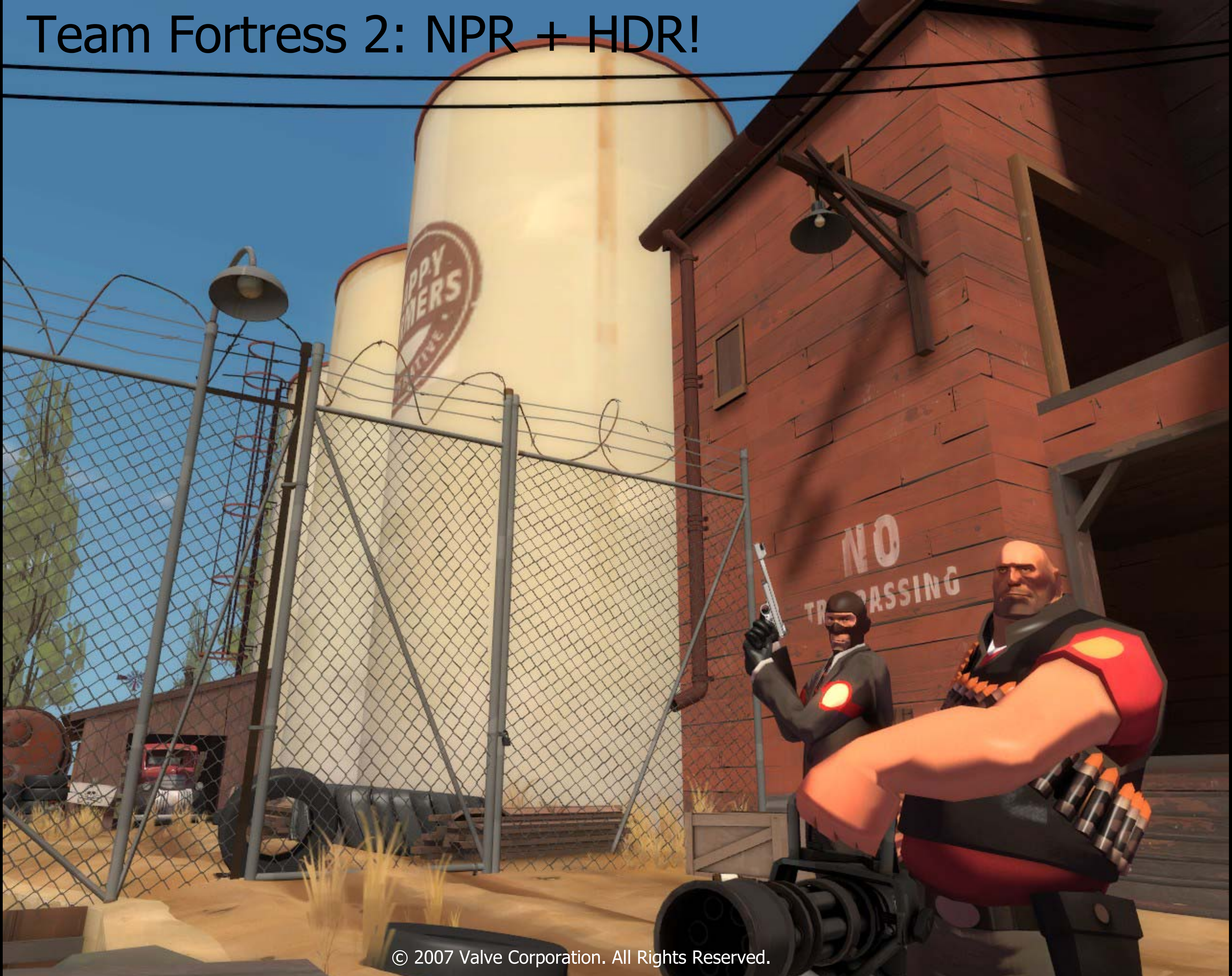
Half Life 2: Episode 1



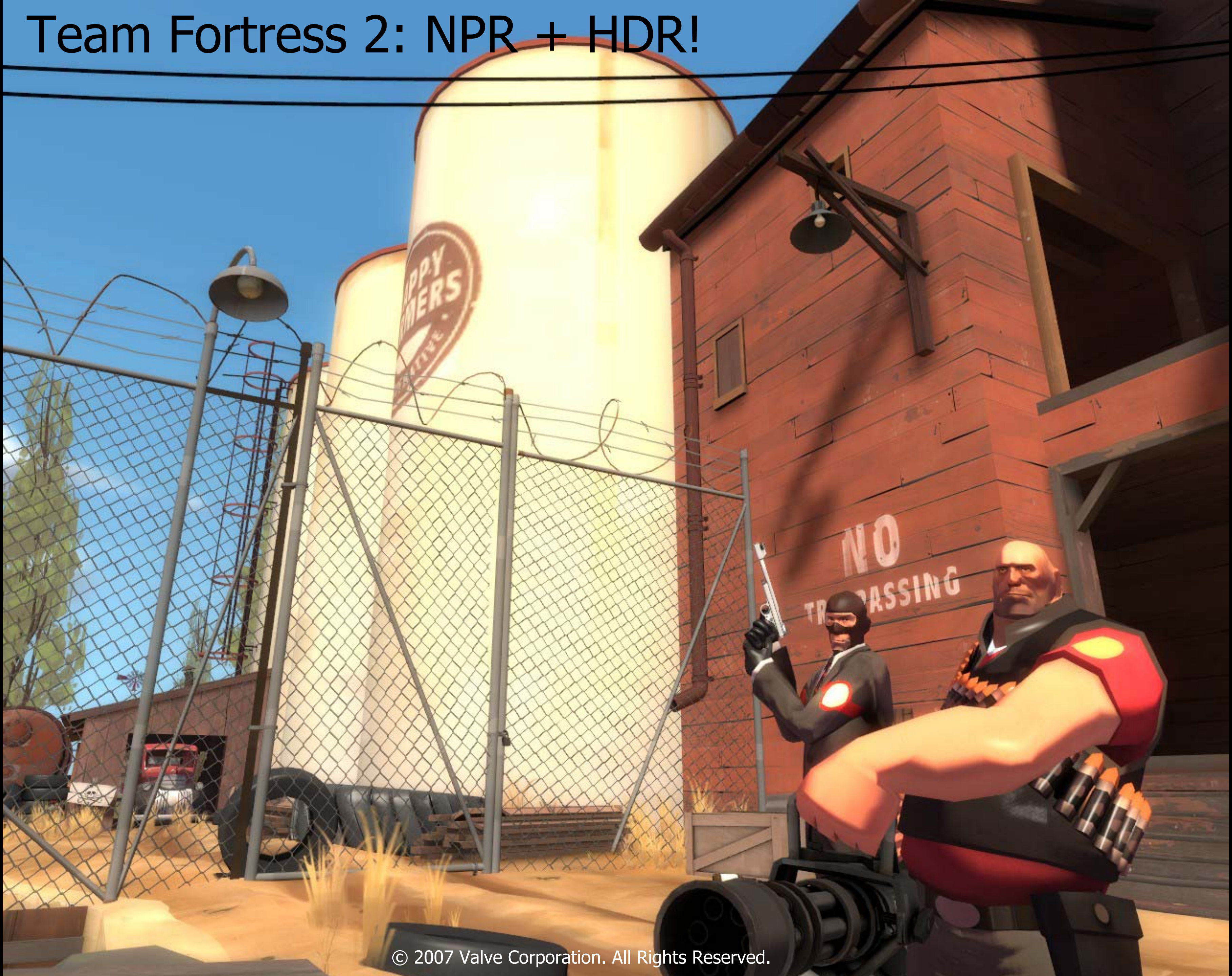
Half Life 2: Episode 1



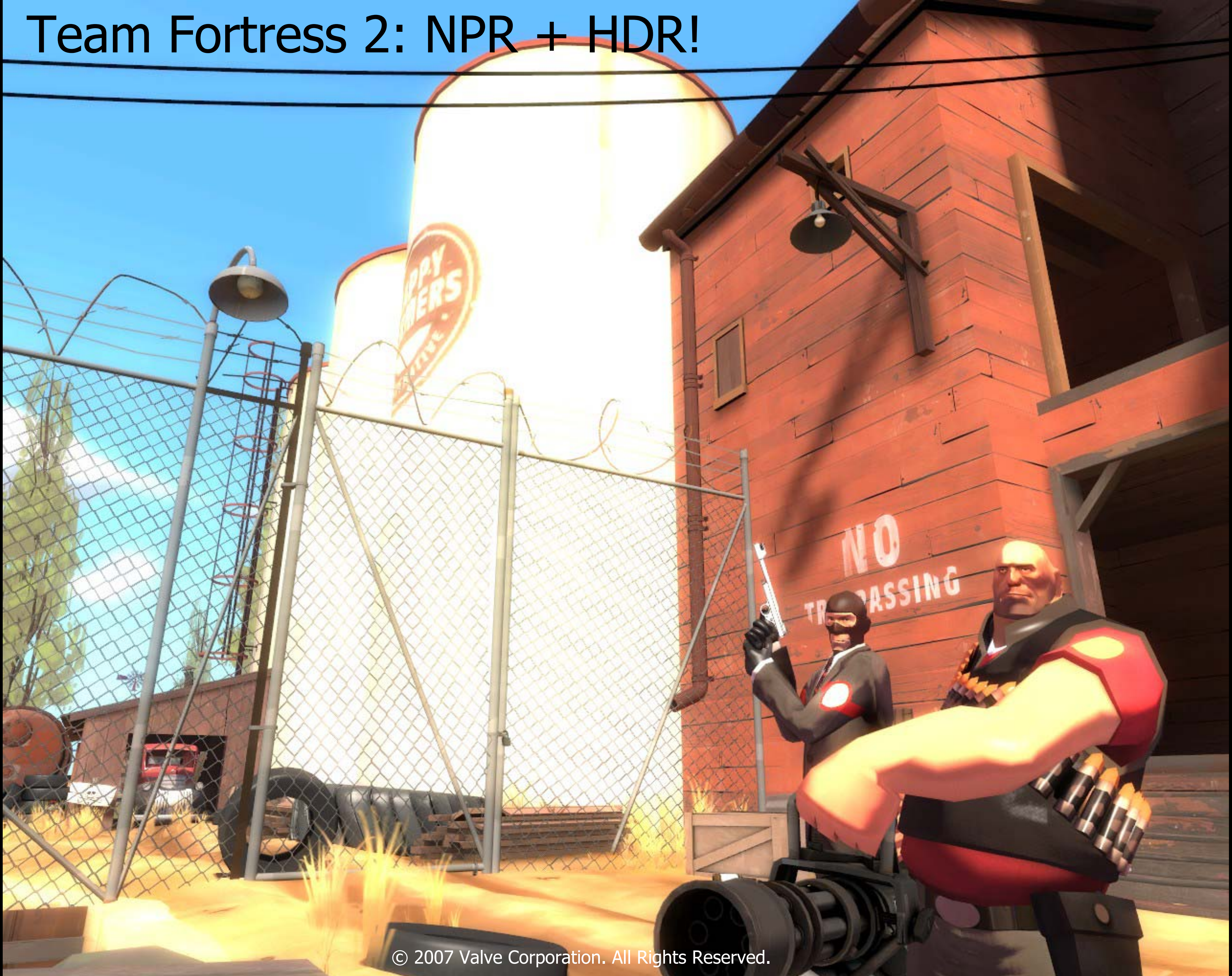
Team Fortress 2: NPR + HDR!



Team Fortress 2: NPR + HDR!



Team Fortress 2: NPR + HDR!



Lost Coast Demo

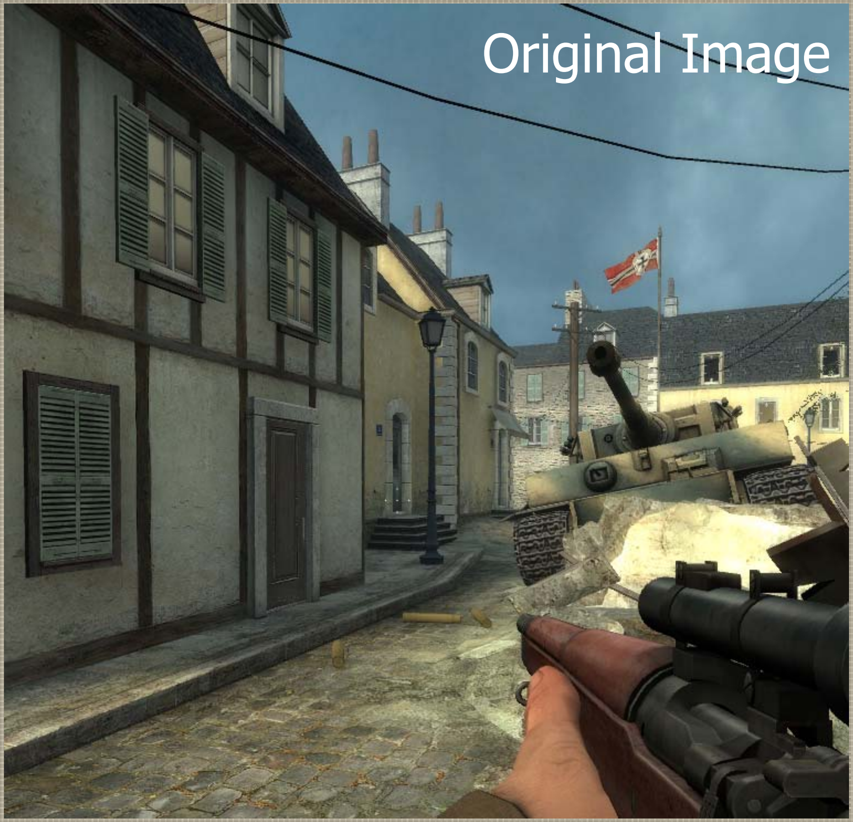
Color Correction

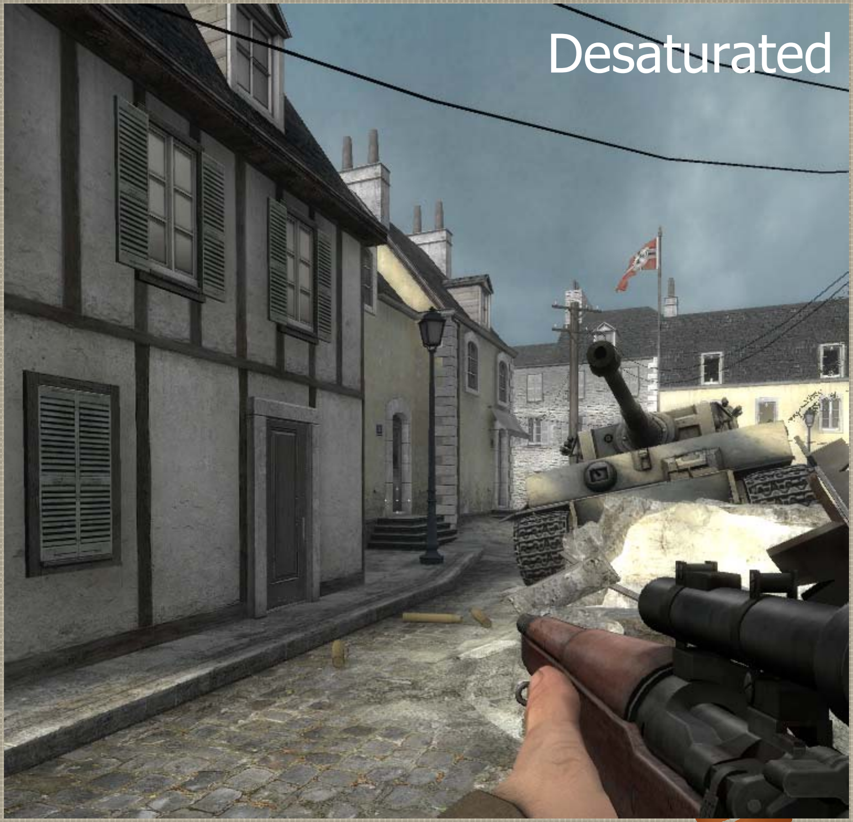
- Color has been used in visual arts to evoke particular emotional responses for millennia, and we can do the same in video games
- It is very natural to apply color correction to rendered images as a post process, since we're doing lots of image processing already
  - Simply define a mapping from a one set of RGB triples to another
  - We do this with a 32^3 volume texture lookup (sometimes several)
- Decouples color correction from the art or lighting of a given setting
- Useful for a variety of purposes
  - Stylization
  - Day for night
  - Gameplay

Color Correction in Source

- Color correction in games can go beyond what is possible in film
- The dynamic nature of games is both a curse and a blessing
  - More difficult to tweak to specific scenes
  - Far more potential to exploit the dynamic nature of games
  - Strong feedback loops with the player
  - Additional sideband communication with the player
- Color correction is a powerful tool for art direction
  - Mod authors and licensees can differentiate their titles

Desaturation





Future Directions

- Improvements to irradiance volume sampling
- Shadow Mapping
- Foliage Rendering
- Soft Particles
- Non-Photorealistic rendering in *Team Fortress 2*

Conclusion

- **World Lighting**
  - Radiosity Normal Mapping
  - Specular Lighting
- **Model lighting**
  - Irradiance Volume
  - Half-Lambert
  - Phong terms
- **High Dynamic Range Rendering**
  - Tone mapping
  - Autoexposure
- **Color Correction**

SDK & Academic Licensing

- Publicly available SDK
- Academic licenses provide
  - Access to Valve games
  - Source code
    - HLSL shaders, Radiosity and visibility calculations
    - AI system, path finding
    - Animation system, acting system, inverse kinematics
  - Production quality art and sound assets
  - Useful level and modeling tools
    - Hammer level editor, Faceposer, Model viewing utilities

`academiclicensing@valvesoftware.com`

Acknowledgements

- Many thanks to Gary McTaggart, Chris Green, James Grieve, Brian Jacobson, Ken Birdwell, Bay Raitt and the rest of the folks at Valve who contributed to this material and the engineering behind it
- Thanks also to Paul Debevec for the light probes used in the comparison with Ramamoorthi’s Irradiance Environment Maps

References

- [Debevec00] Paul Debevec, Personal Communication, 2000.
- [Diefenbach97] Paul J. Diefenbach, "Multi-pass Pipeline Rendering: Realism For Dynamic Environments," Proceedings, 1997 Symposium on Interactive 3D Graphics, 1997.
- [Franke06] Scott Franke, Personal Communication, 2006.
- [Green06] Chris Green and Gary McTaggart, "High Performance HDR Rendering on DX9-Class Hardware," Poster presented at the *ACM Symposium on Interactive 3D Graphics and Games*, March, 2006.
- [Greger98] Gene Greger, Peter Shirley, Philip M. Hubbard and Donald P. Greenberg, "The Irradiance Volume," IEEE Computer Graphics & Applications, 18(2):32-43, 1998.
- [Hargreaves03] Shawn Hargreaves, "Hemisphere Lighting with Radiosity Maps", Gamasutra, August 2003.
- [Hook99] Brian Hook, "The Quake 3 Arena Rendering Architecture," Game Developer's Conference 1999.
- [Lehtinen06] Jaakko Lehtinen, Personal Communication, 2006.
- [McTaggart04] Gary McTaggart, "Half-Life 2 Shading," GDC Direct3D Tutorial 2004.
- [Ramamoorthi01] Ravi Ramamoorthi and Pat Hanrahan, "An Efficient Representation for Irradiance Environment Maps," SIGGRAPH 2001, pages 497-500.
- [Rusinkiewicz06] Szymon Rusinkiewicz, Michael Burns, and Doug DeCarlo. "Exaggerated Shading for Depicting Shape and Detail," ACM Transactions on Graphics (Proc. SIGGRAPH). 25(3) July 2006.