Real-Time Outdoor Illumination

Naty Hoffman
Naughty Dog

Real-Time Outdoor Illumination

Realistic Real-Time Lighting for Outdoor Scenes

- Source of light
  - Sun
  - Sky
  - Indirect
- Outdoor scene
  - Terrain
  - Other objects

I'll be discussing some of the techniques and practical issues involved with real-time rendering of realistic (dynamic) lighting.

Terrain

Unique challenges and opportunities

- Visually dominant
- High complexity and detail
- Static
- Regular structure

Terrain Lighting

Dynamic Lighting Techniques

- Terrain is static
- If the lighting environment is also static, usually best to pre-compute the lighting
- These techniques can be used to speed precomputation in some cases

Since the terrain itself is assumed to be static, if the lighting is also static, it can usually be precomputed and stored (terrain reflectance is usually primarily diffuse).

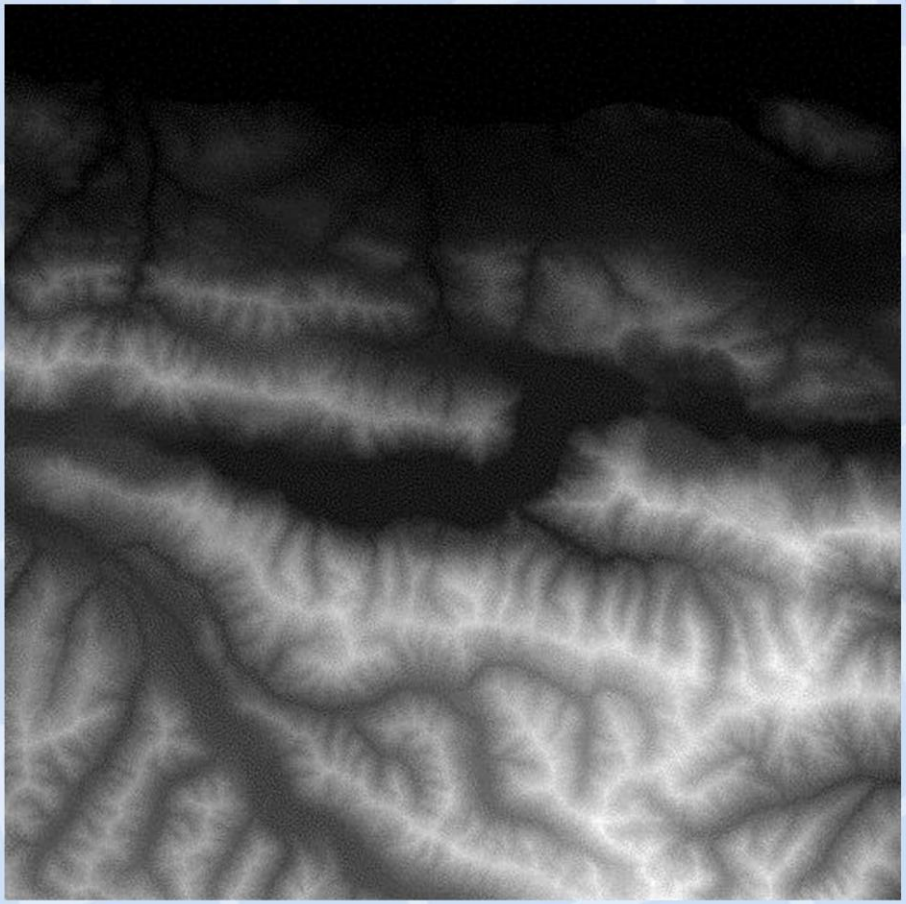
The techniques I discuss are focused on dynamic lighting.

Terrain Lighting

Can split into sub-problems

- Sunlight
  - High radiance
  - Small solid angle
- Skylight
  - Covers entire hemisphere
  - Radiance values angularly variant
- Distant sources – non-local

Terrain Data



The dataset used in the demo was derived from a 10 meter resolution DEM (Digital Elevation Model) of a 26-kilometer-square section surrounding Lake Crescent (the dark area in the middle) in northern Washington state (the dark area on top is the Strait of Juan de Fuca). DEMs for most of the world can be found online. I slightly downsampled the original grid into 2Kx2K before processing. Note that since this heightfield data is used to generate normals and other precision-sensitive data, it needs to have at least 16-bit precision (I used 16-bit).

Sunlight

Direct illumination

- Color Map
- Normal Map
  - *Appearance-Preserving Simplification* (Cohen, Olano, Manocha SIGGRAPH98)
  - Related to bump maps; *Simulation of Wrinkled Surfaces* (Blinn SIGGRAPH77)
- Sun can be treated as a point/directional light source for direct illumination purposes

Normal mapping is simple on terrain, which is specified in world-space coordinates so there are no transforms or changes of coordinate system to worry about. The demo uses a 2Kx2K normal map (generated from the heightfield using D3DX) on a 129x129 mesh grid. The starting camera is looking south from the Strait of Juan de Fuca toward Lake Crescent.

<switch to demo in direct-illumination mode>

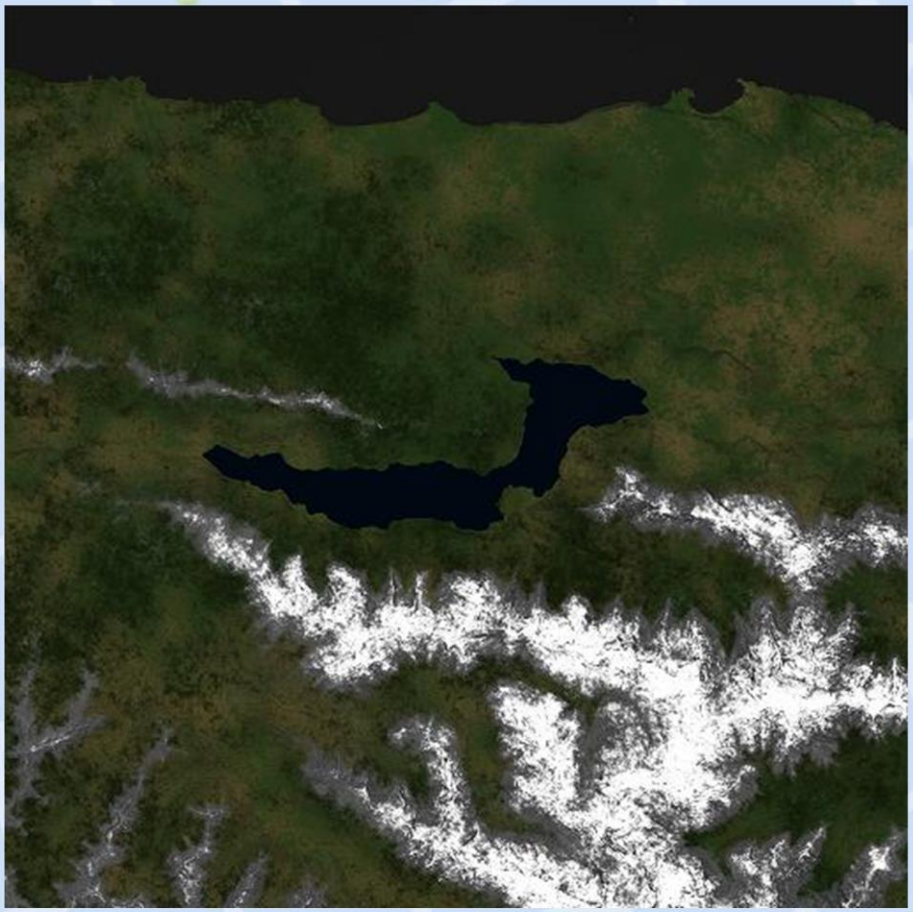
Color Map

Actually a diffuse reflectance map

- Photographic data can be used, but
  - Lighting needs to be factored out
  - Colors need to be scaled to correct range
- Color can be generated procedurally from elevation data
  - Taking elevation and slope into account
  - Terragen, Bryce

Unfortunately, getting reflectance maps of real-world scenes is not as straightforward as getting elevation data. Most available satellite imagery is monochrome or in false-color, and even when true-color images are available they are usually not appropriate for use with dynamic lighting because they have lighting “baked in”.

Color Map



This 2K x 2K color map was generated from the height field using Terragen. This is a large texture, but most hardware supports compression algorithms which are appropriate for color data.

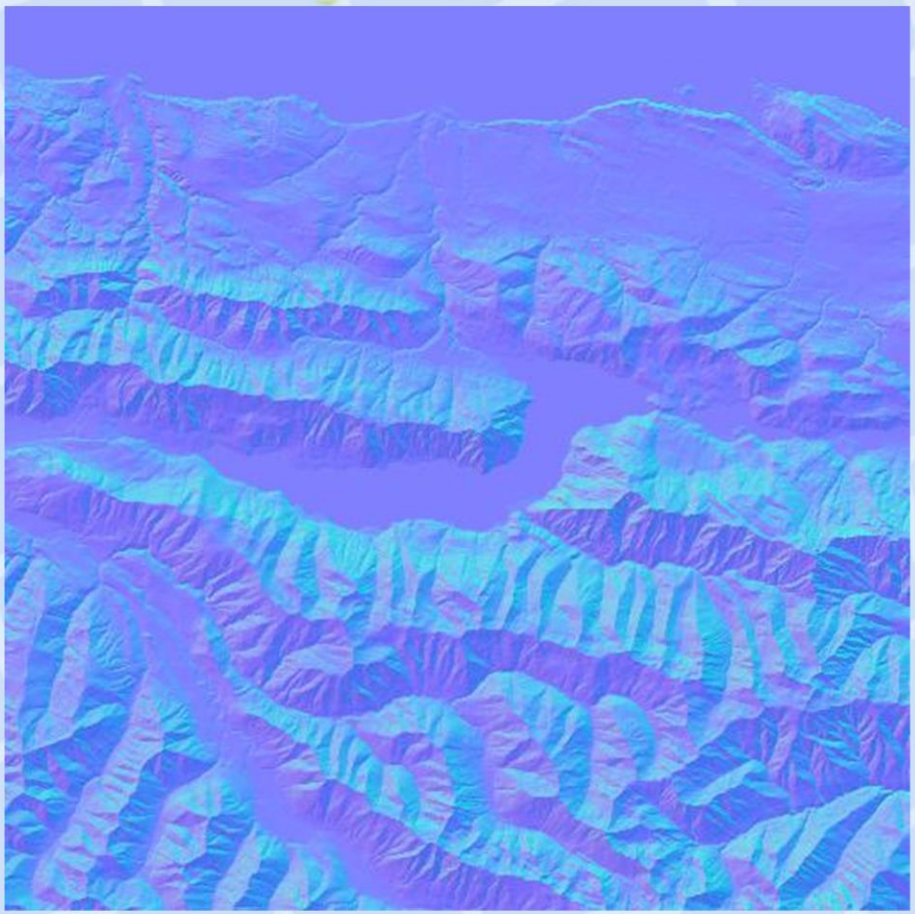
Normal Map

Stored as an RGB image

- 8-bits per channel sufficient for terrains
  - Still expensive – 2048x2048 is 16 MB with Mip-maps!
  - High-end consumer cards today have 128-256MB
- Hardware texture compression schemes produce undesirable artifacts
- Some practitioners have achieved good results with 8-bit palettes

The high memory consumption may be OK, depending on the hardware requirements of your application.

Normal Map



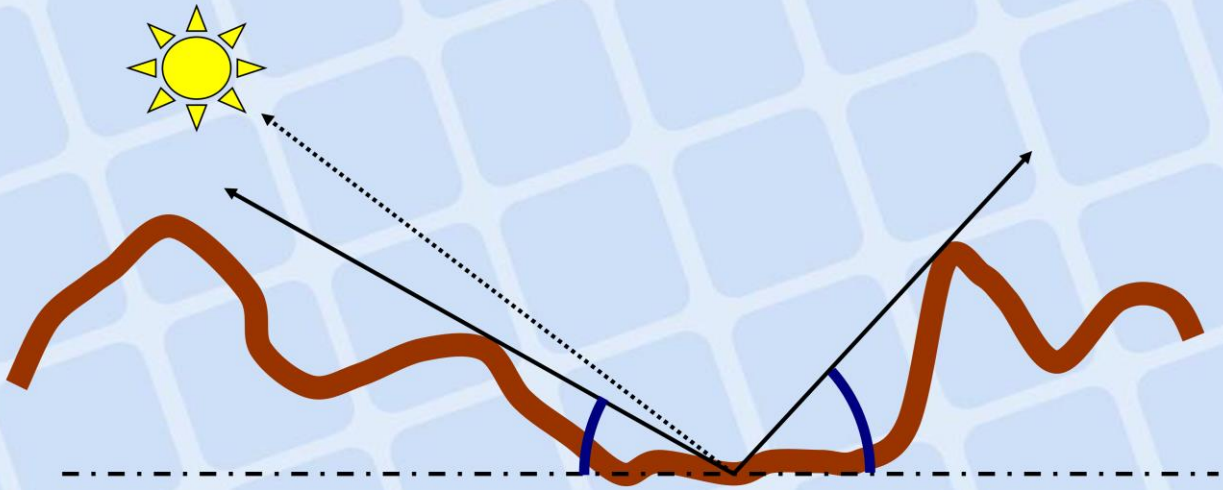
They are stored in a biased format to bring the –1 to 1 range of x, y and z into a 0-1 range. The characteristic pale blue color of the flat areas is (0.5, 0.5, 1).

Sunlight

SIGGRAPH 2003
SAN DIEGO

Terrain Shadows

- *Horizon mapping: shadows for bump-mapped surfaces* (Max, The Visual Computer 1988)

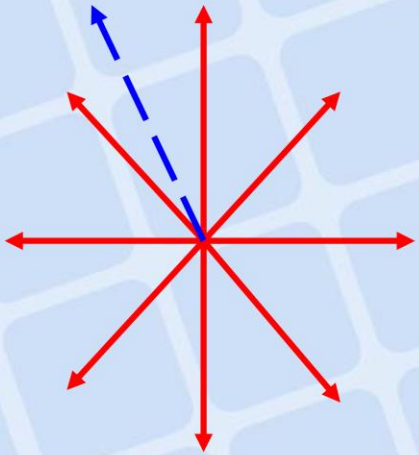


Sunlight

SIGGRAPH 2003
SAN DIEGO

Terrain Shadows

- *Interactive Horizon Mapping* (Sloan and Cohen EGRW2000)

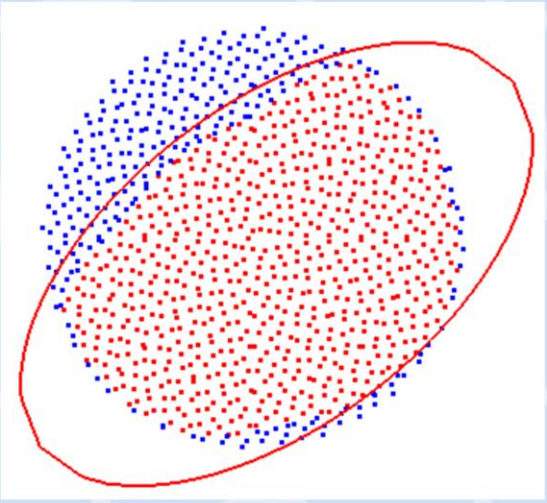


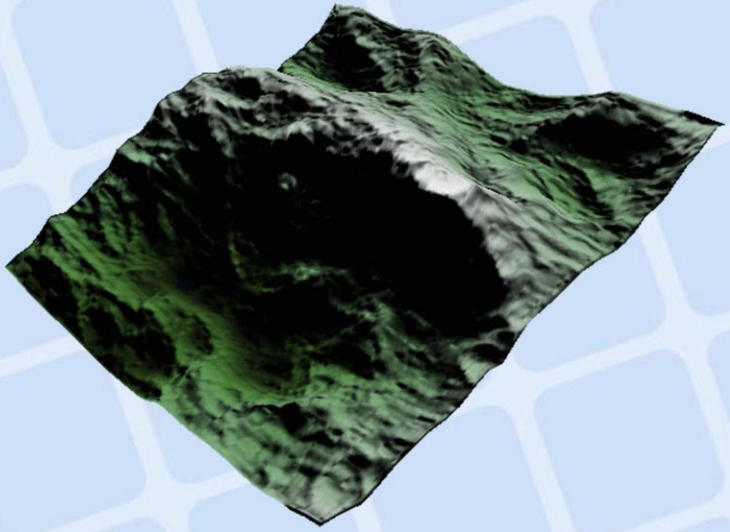
Sunlight

SIGGRAPH 2003
SAN DIEGO

Terrain Shadows

- *Illuminating Micro Geometry Based on Precomputed Visibility (*Heidrich, Daubert, Kautz, Seidel SIGGRAPH00)





A soft-shadow effect can be achieved by modulating the light according to the distance from the ellipse boundary.

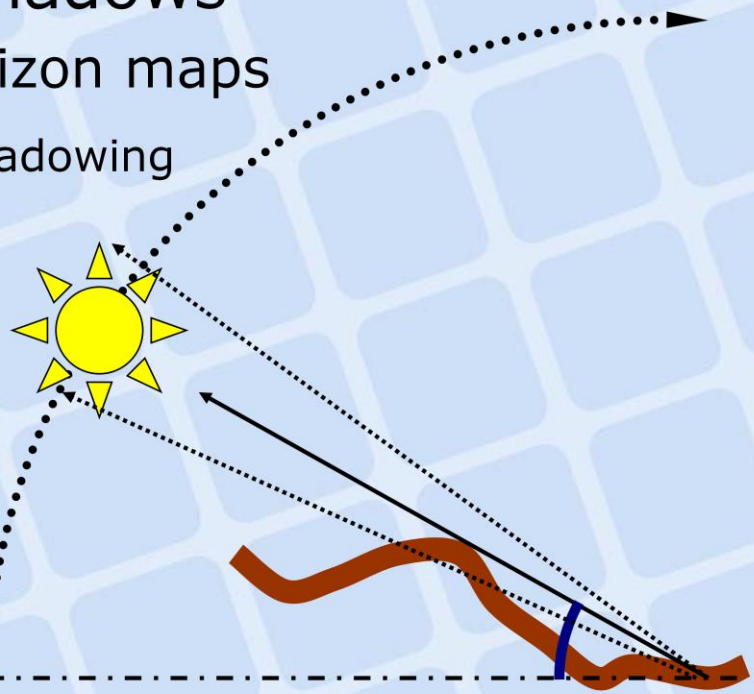
The authors store the six numbers describing the ellipse in two RGB texture maps; the paper describes a hardware-accelerated implementation of the algorithm.

Sunlight

SIGGRAPH 2003
SAN DIEGO

Simple terrain shadows

- Single-axis horizon maps
  - With 1D soft shadowing



Assume that the sun moves in a single axis in a plane perpendicular to the ground.

Then horizon angles can be computed and stored along that direction only. This is easy to implement using a pixel shader.

<Switch to demo. Show direct + shadows (no skylight) – show soft shadow edges (exaggerated for effect).>

Horizon Map

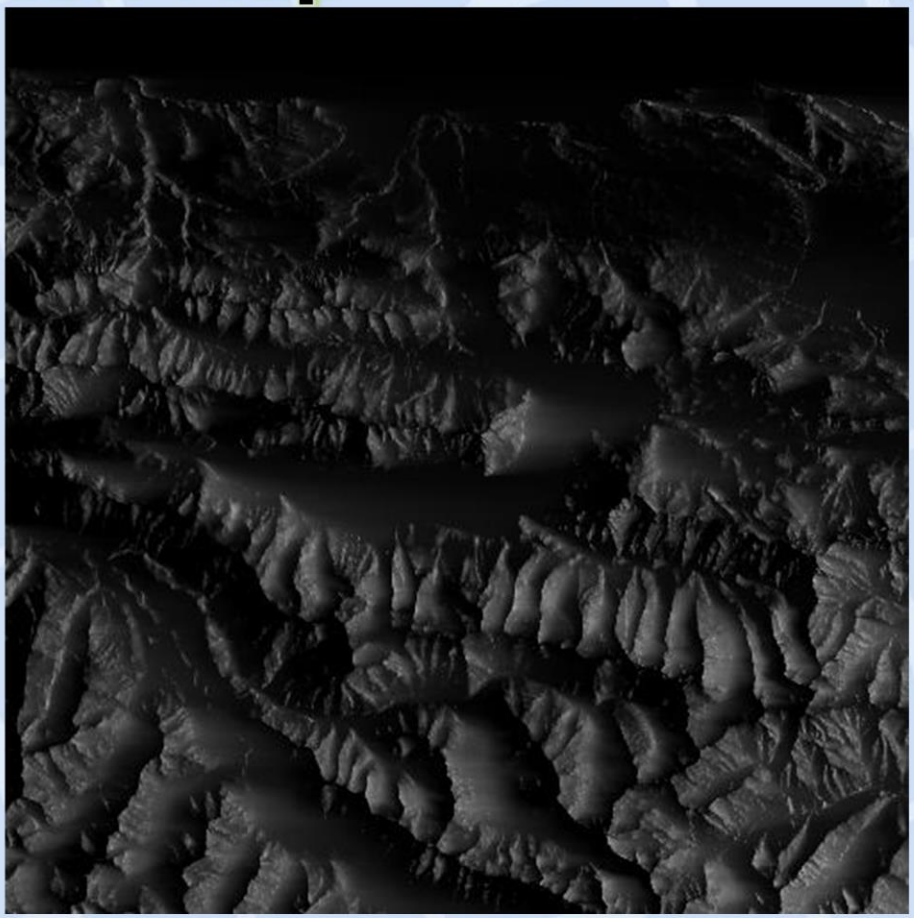
Two separate single-channel images

- Pick one depending on whether the sun is east or west of the zenith
- 16-bit images are preferable for higher accuracy
- As for normal maps, storage can be expensive
  - Good compression options are an open question

The newest graphics hardware can support texture channels with higher precision than 8 bits. Within two years this support will probably be ubiquitous.

I haven't looked much into compression options for horizon maps, so this may be an interesting thing to experiment with.

Horizon Map



The angles are stored scaled so that the range 0 to PI/2 fits in the range 0-1. Brighter values represent larger angles (as measured from the horizon), thus higher occlusion.

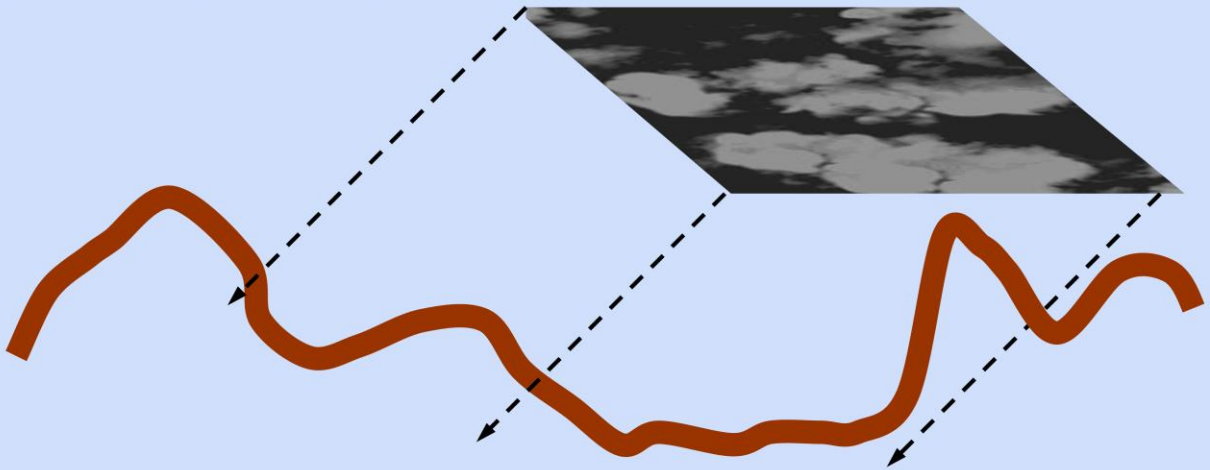
This map stores occlusion in the direction of positive x (to the right).

Sunlight

SIGGRAPH 2003
SAN DIEGO

Cloud Shadows

- Parallel projection of 2D texture on terrain
- Modulate sunlight



<Switch to demo – turn on cloud cover for first time (still no skylight).>

Clouds can be scrolled by merely changing the texture coordinates. Note that the direction of texture projection in this demo does not change as the sun moves—but this is easy to fix.

Cloud Cover Map

Single-channel image

- 8 bits sufficient
- Lower resolution than the other maps
  - Not 1:1 mapped onto terrain
  - Lower frequency content

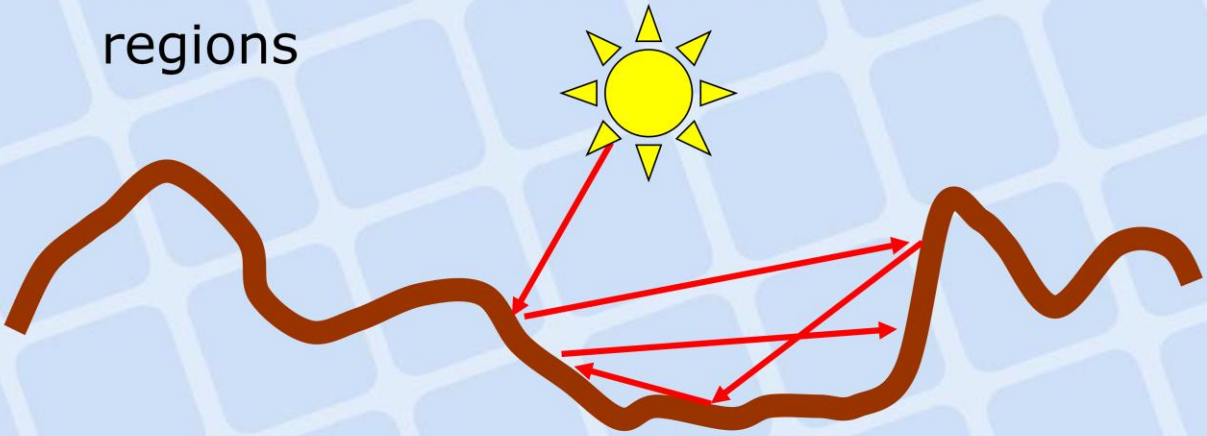
Cloud Cover Map

Sunlight

SIGGRAPH 2003
SAN DIEGO

Indirect Illumination

- Previous techniques all handle direct illumination with occlusion
- But not interreflections between terrain regions



Sunlight

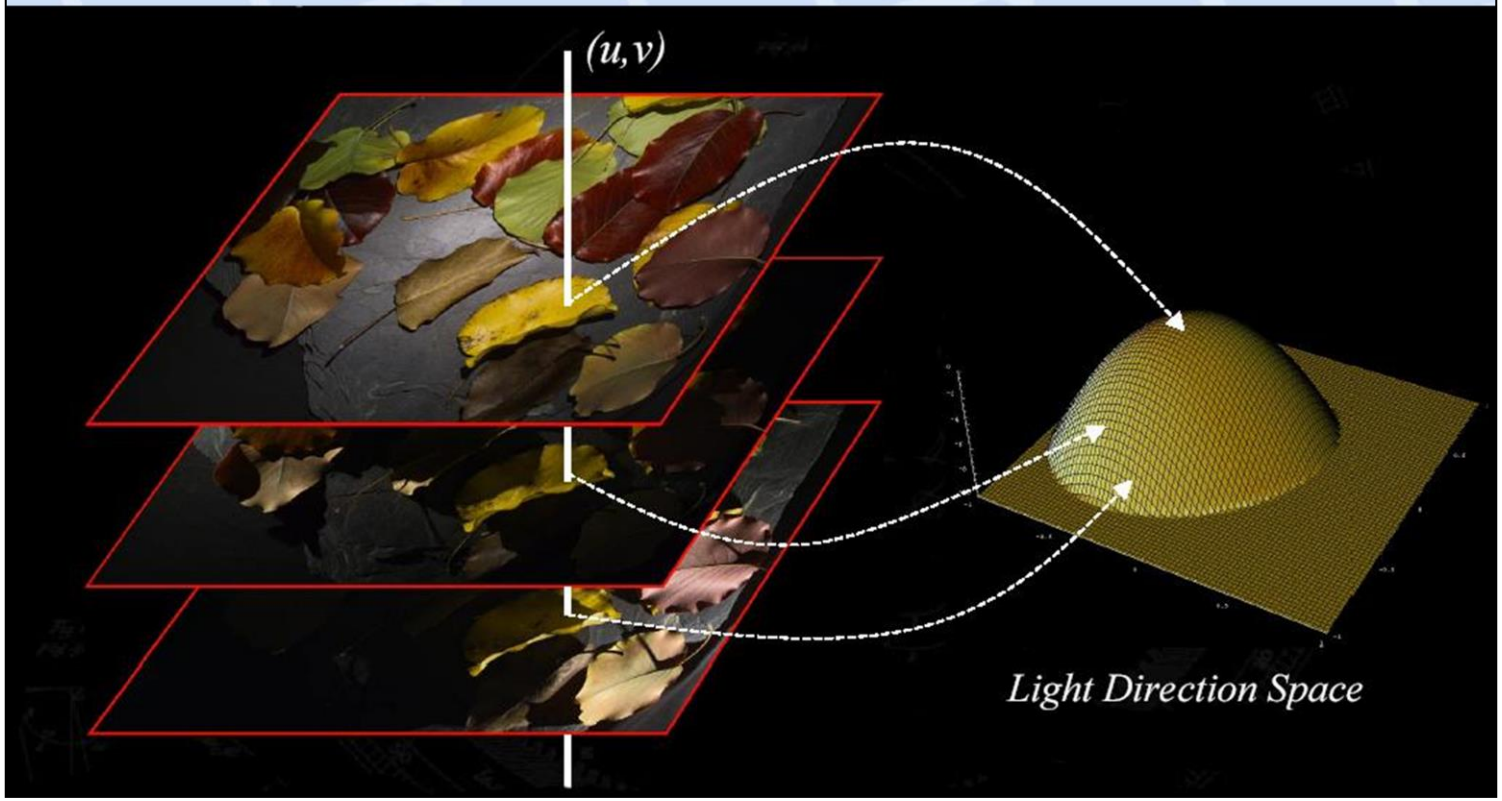
SIGGRAPH 2003
SAN DIEGO

Indirect Illumination

- *Polynomial Texture Maps* (Malzbender, Gelb, Wolters SIGGRAPH01)

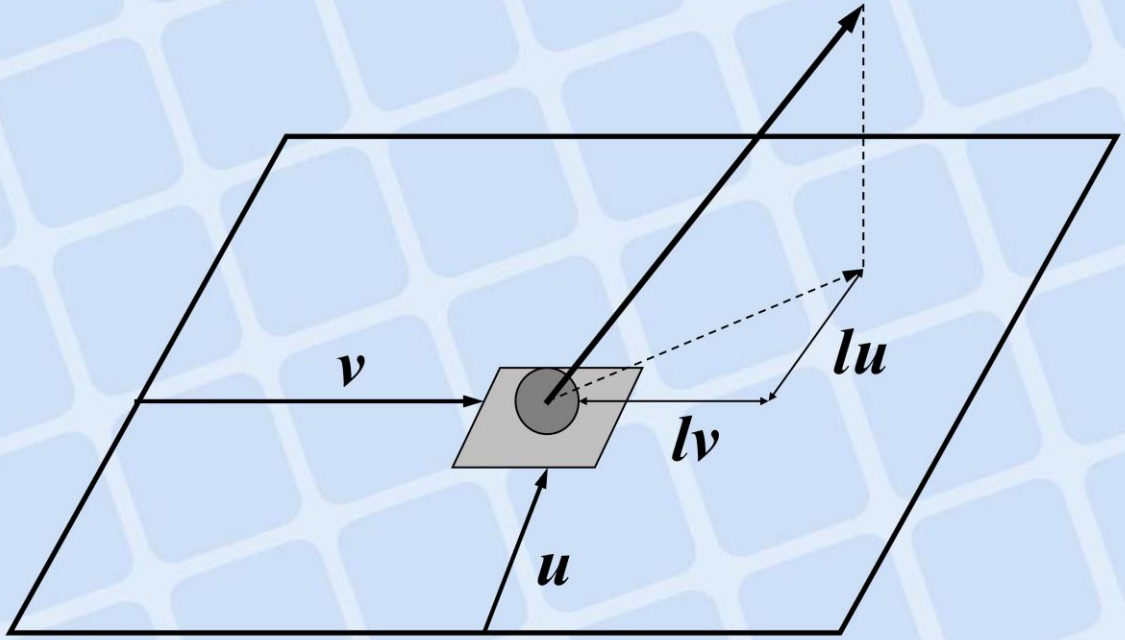
A powerful technique for rendering diffuse surfaces with global illumination effects in real time.

Polynomial Texture Maps



This image (from the original SIGGRAPH presentation and used with the kind permission of Tom Malzbender) explains the basic idea.

Polynomial Texture Maps



Polynomial Texture Maps

Pixel color as function of light direction

- Chromaticity tends to be constant
- Luminance varies smoothly with light direction for a given pixel
  - Approximate with a biquadratic polynomial:

$$L(u,v;l_u,l_v) = a_0(u,v)l_u^2 + a_1(u,v)l_v^2 + a_2(u,v)l_u l_u + a_3(u,v)l_u + a_4(u,v)l_{v+} + a_5(u,v)$$

Polynomial Texture Maps

A series of images with different light directions is required as input

- Constant camera
- Photographs can be used
- Or a high-end renderer can be used to generate images from virtual objects
- HP distributes a tool to fit biquadratic coefficients to sets of images

Polynomial Texture Maps

$$L(u,v;l_u,l_v) = a_0(u,v)l_u^2 + a_1(u,v)l_v^2 + a_2(u,v)l_u l_u + a_3(u,v)l_u + a_4(u,v)l_{v+} + a_5(u,v)$$

- a_0, a_1, …, a_5 stored in two RGB maps
- A third map used for chromaticity (RGB)
- Or a space like LUV can be used, in which case only UV need be stored in addition to a_0, a_1, …, a_5 .
  - Total can be packed into two ARGB maps.
- 8 bits per channel sufficient

<Show PTM Viewer – leaves and seeds, as well as terrains.>

The terrain PTM data in this demo only includes local illumination, but the PTM technique can easily handle any view-independent phenomena such as interreflections, self-shadowing and even subsurface scattering. I plan to generate new data showing those features in future. The polynomial is symmetrical about the Z plane, so backfacing light needs to be handled as a special case.

Polynomial Texture Maps

Sunlight on terrain is a good match for polynomial texture maps

- Coefficients and light directions can be specified in world-space
- No need to deal with $l_z < 0$ case
- However, usually photographic data cannot be used
  - High-quality renders of detailed terrain data can be slow

PTMs can be used on general surfaces, but being able to specify everything in world-space simplifies the implementation.

Also, sunlight is highly directional – PTMs do not handle arbitrary light environments

Skylight

Skylight is very different from sunlight

- Much lower radiance
- Instead of a 1/2° circle, covers entire hemisphere
- Complex directional color /intensity variation
  - Especially at sunrise, sunset

So far, the discussion has been about lighting terrain with sunlight. But sunlight alone is not sufficient (unless you live on the moon) – the sky contributes significant illumination which is particularly noticeable in areas where the sun's light is shadowed.

Skylight

Ambient term

- Just add a constant to the lighting
- The simplest and least accurate approximation of skylight
- Better than no skylight at all…

<Show demo in skylight (but no occlusion factor) mode.>

The demo uses a uniform sky color. The ambient term simply adds this color to the lighting result.

Skylight

Ambient with occlusion term

- During a pre-processing stage, rays are cast from each terrain point
  - The ones that are not occluded are counted (weighted by the cosine with the normal)
  - The final result is divided by the total # of rays
- Approximates the sky's irradiance contribution, assuming uniform radiance

$$E = \int_{\Omega_{SKY}} L_{in}(\theta)\cos\theta \, d\Omega_\theta = L_{in} \int_{\Omega_{SKY}} \cos\theta \, d\Omega_\theta$$

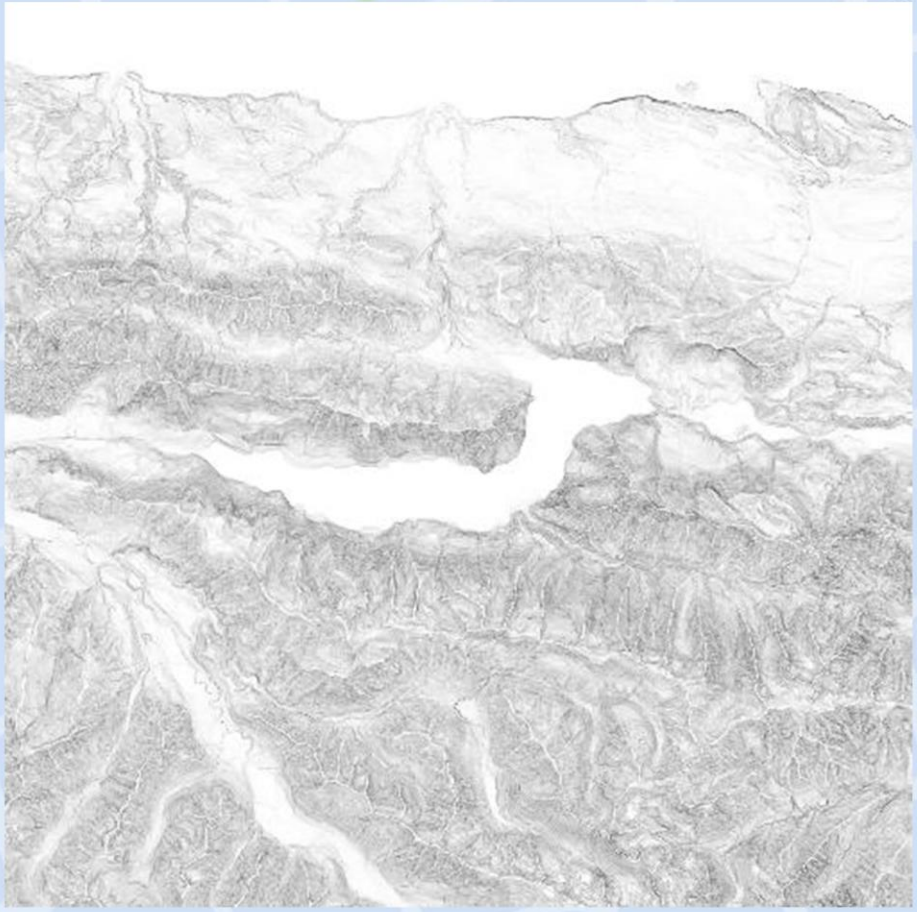
Occlusion Map

Stores sky occlusion term for each texel

- Single-channel texture
  - 8 bits sufficient
  - Can be packed with normal map into a single ARGB texture
- Multiplied with sky color in pixel shader
  - Resulting skylight added to sunlight

<Switch to demo: now in full mode. Show difference between occlusion map and ambient term.>

Occlusion Map



This occlusion map was generated from the height-field by the library function in Direct3DX. It doesn't use the full ray-cast occlusion algorithm but rather a fast approximation of it.

Skylight

SIGGRAPH 2003
SAN DIEGO

Occlusion maps don't account for terrain interreflections

Skylight

Fully accounting for interreflections requires a global illumination solution

- For uniform sky radiance, a solution can be precomputed with a sky emitting unit radiance.
  - Store the result in a texture
  - Used just like a sky occlusion map

Skylight

Sometimes a full global illumination solution is not practical

- Not needed for interreflections under uniform hemispherical illumination
  - In *Towards Accurate Recovery of Shape from Shading Under Diffuse Lighting* (Stewart and Langer IEEE PAMI97) it was found that under these conditions, it can be assumed that incident radiance reflected from other points is equal to the exitant radiance at the current point
  - Yields a closed-form approximation which takes interreflections into account

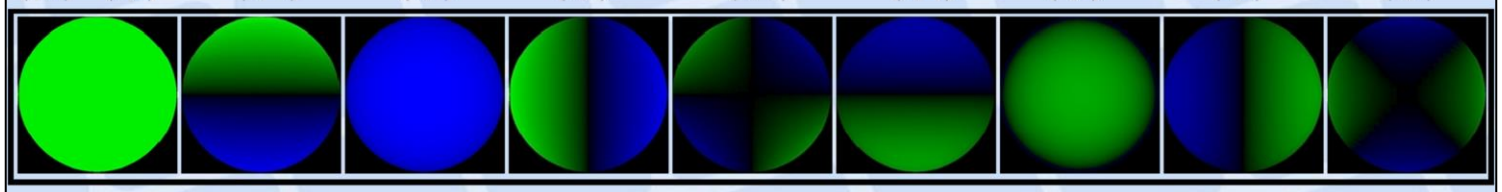
In cases where running a full global illumination solution is not practical (for example, if there are infrequent local deformations of the terrain), I recommend using Stewart and Langer's approximation.

Skylight

Uniform radiance is a very limiting assumption for skylight

- Unless the day is extremely overcast
- Skies usually have a rich variety of light intensities and colors
- We can represent the sky lighting using basis functions and pre-compute a solution for each one ("relighting"); *Efficient Re-rendering of Naturally Illuminated Environments* (Nimeroff, Simoncelli and Dorsey EGWR94)
  - Combine at runtime

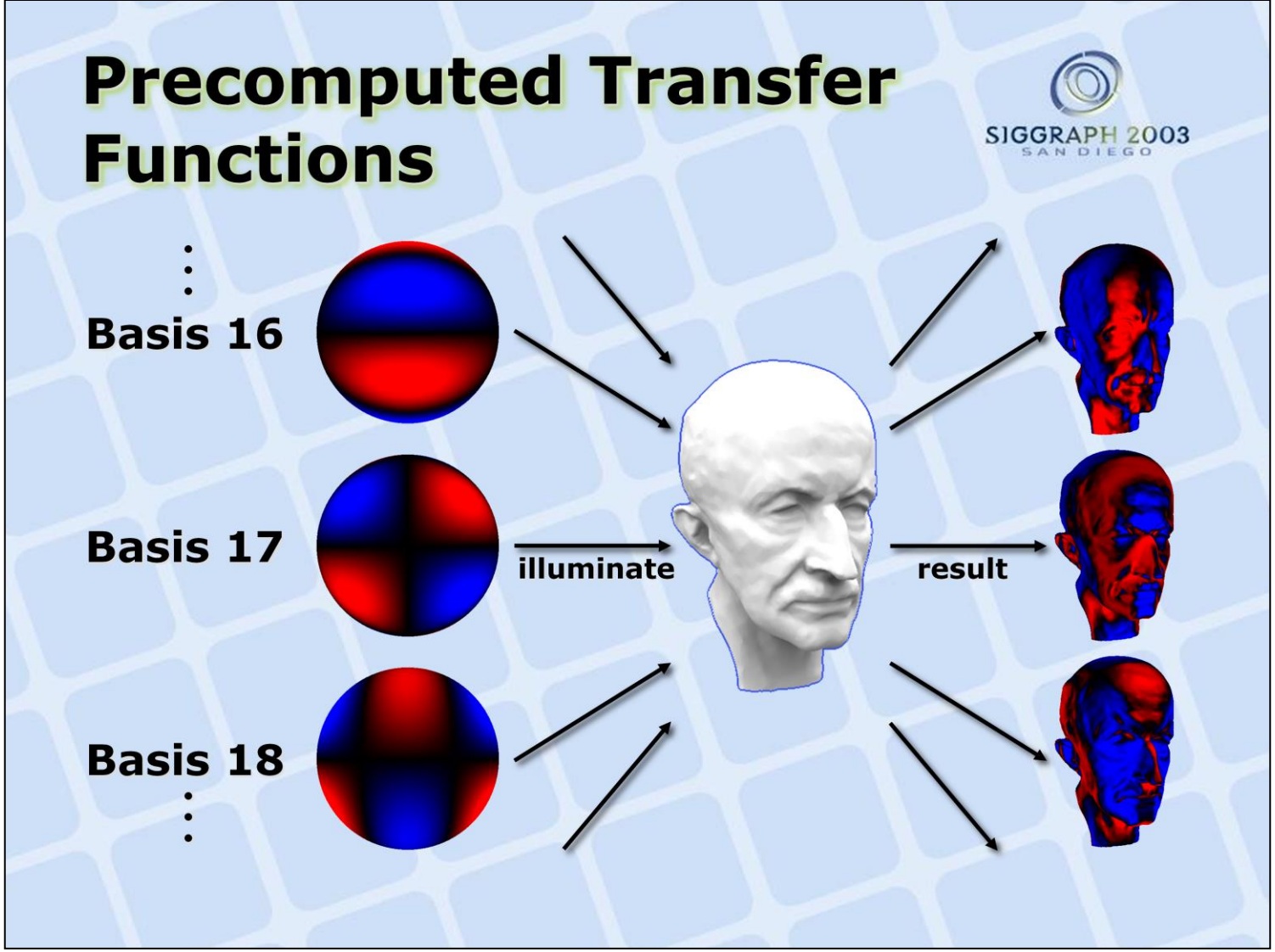
Skylight



Spherical Harmonics

- Form a useful basis for representing incident lighting
  - Rotationally invariant, orthonormal
  - Can be used for a relighting approach: *Precomputed Radiance Transfer for Real-Time Rendering in Dynamic, Low-Frequency Lighting Environments* (Sloan, Kautz and Snyder SIGGRAPH02)

Here we see the first nine spherical harmonics.



This is an example of the precomputation process. Essentially we are doing global illumination solutions for the object, with the SH basis functions as light environments. One way to do this is with global illumination software which can handle negative lights. Otherwise, the basis function can be split into negative and positive parts and each part solved separately and the results combined. Later during runtime, these various ‘solutions’ are weighted by the lighting SH coefficients and added together. (image courtesy of Peter-Pike Sloan)

Precomputed Transfer Functions

Vectors of SH coefficients

- [Sloan02] recommends 16 or 25
- Can be stored on vertices or in textures
- As for polynomial texture maps, the color can be factored out of the radiance transfer
  - To avoid separate transfer functions for R, G, B
- At runtime just perform dot-products
  - Lighting coefficients with transfer coefficients

Precomputed Transfer Functions

- Can reproduce all global illumination effects
  - Soft shadows, interreflections, subsurface scattering
- Can use arbitrary (low-frequency) lighting
- Good match for skylight on terrain
  - Simplest (and fastest) on diffuse surfaces
  - Lighting and transfer functions specified in world-space – no need to rotate SH coefficients
  - Skylight is low-frequency in nature

Precomputed Transfer Functions

- An active area of research
  - Dedicated paper session at SIGGRAPH this year
- Recent developments relevant for outdoor lighting
  - Data compression and performance improvements: *Matrix Radiance Transfer* (Lehtinen & Kautz SI3D03), *Clustered Principal Components for Precomputed Radiance Transfer* (Sloan, Hall, Hart & Snyder SIGGRAPH03)
  - *Bi-Scale Radiance Transfer* (Sloan, Liu, Shum & Snyder SIGGRAPH03) could enable transfer function 'detail maps' on terrain
  - *All-Frequency Shadows Using Non-linear Wavelet Lighting Approximation* (Ng, Ramamoorthi & Hanrahan SIGGRAPH03) promises more accurate shadows

This is an area worth following for anyone interested in real-time realistic outdoor lighting. Some of these papers are being presented at this conference in the "Precomputed Radiance Transfer" session.

Time-Based Approaches

These techniques let us independently control sun position, sky color, etc.

- Some applications just need to light the terrain with a predetermined sequence of lighting changing over a time period
  - Can use 1D RGB Polynomial Texture Maps
  - Or Video-based lighting (Mitchell, Game Programming Gems 3)

Explain a bit about both techniques.

Terrain Lighting

Further refinements are possible to these techniques

- For example, detail maps can be applied to add fine-grained detail to color, normals, even horizon maps; *Pixel Shader Optimizations for Terrain Rendering* (Mitchell, Graphics Programming Methods)
- Techniques can be combined
  - E.g. polynomial texture maps for sunlight, precomputed transfer functions for skylight

Outdoor Object Lighting

For our purposes, anything in the scene that isn't terrain is an object

- Static objects (buildings)
- Dynamic objects (characters)
- The light environment is the same for terrain and objects
  - But techniques differ

Techniques tend to be very different—objects are usually more dynamic and lack the regular structure of terrain. On the other hand, we can usually get away with coarser approximations on dynamic objects.

Object Lighting

Sunlight on outdoor objects can be handled by commonly-used techniques

- The skylight however is an interesting case
  - Rich lighting environment
  - And mostly constant throughout the scene

Object Lighting

- We can take advantage of the fact that the skylight environment is the same throughout the scene by using environment maps for sky lighting on objects
  - Best for shiny & smooth surfaces
  - Pre-filtered environment maps can be used for glossy surfaces; *A Unified Approach to Prefiltered Environment Maps* (Kautz, Vázquez, Heidrich, Siedel EGWR00)

Object Lighting

For sky light on diffuse objects:

- *An Efficient Representation for Irradiance Environment Maps* (Ramamoorthi & Hanrahan SIGGRAPH01)
- Uses same spherical harmonics representation of incident light as [Sloan02]
- Local illumination using only the normal
- Can handle deforming / skinned objects
- Sun, other lights handled “for free” by adding them into the SH lighting coefficients

This has the advantage of matching the terrain lighting, while being inexpensive (there is a simple closed form for the irradiance as a function of the world-space normal).

Terrain-Object Transfer

We have seen ways to handle terrain and objects separately

- However, they can cast shadows on each other, etc.
- For sunlight, existing methods (like depth maps or stencil shadows) can be used

Terrain-Object Transfer

Skylight poses some interesting problems for terrain-object transfer

- Terrain shadowing object
  - Could be solved by sampling SH ‘occlusion’ coefficients over the terrain and using those (interpolated) to modulate the skylight coefficients
- Object shadowing terrain
  - An ‘occlusion blob’ can be precomputed in SH coefficients as a texture
  - Applied to modulate SH lighting under objects to produce a rough ‘contact shadow’

Terrain Tools and Data

- Terragen (Planetside Software)
- 3DEM (Visualization Software LLC)
- Terrain Data from the University of Washington

Acknowledgements

Arcot Preetham and Kenny Mitchell for considerable assistance programming the terrain demo

Tom Malzbender for PTM slide image and terrain PTM datasets

Peter-Pike Sloan for PRT slide image