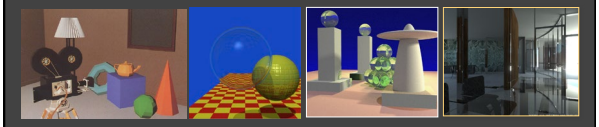
Computer Graphics II: Rendering

CSE 168[Spr 20],Lecture 14: Environment, Texture Maps
Ravi Ramamoorthi

http://viscomp.ucsd.edu/classes/cse168/sp20



To Do

- Start working on final projects (initial results and proposal due in < 2 weeks). Ask me if problems
- Adding HDR/Envmaps (this lecture) may be one component of the final project
- Will briefly also talk about texture mapping

Reflection Maps

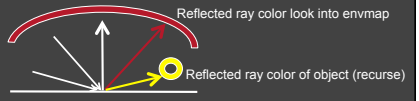
Blinn and Newell, 1976

Environment Maps

Miller and Hoffman, 1984

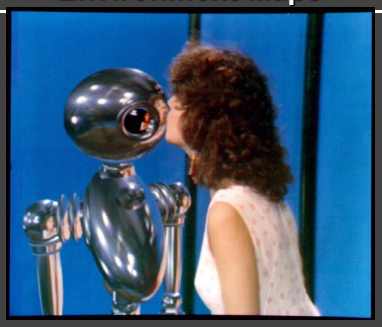
Using Environment for Reflection Map

- Simplest: Mirror reflections (refraction)
  - Start with a simple ray tracer
  - Reflected ray traced to environment (is emission/color)
  - Color += reflectivity * Color of reflected ray
  - Directly use envmap if miss geometry, otherwise recurse
  - (As opposed to zeroing reflections if miss geometry)



  - Easy to do in ray tracer. For path tracer, if reflected ray is sampled (BRDF has mirror component)

Environment Maps



Interface, Chou and Williams (ca. 1985)

Environment Maps

Cylindrical Panoramas

Cubical Environment Map

180 degree fisheye
Photo by R. Packo

Reflection Maps in the Movies

- From history, pauldebevec.com/ReflectionMapping
- First movie, Flight of the Navigator 1986

Environment Map Representations

- Simplest lat-long spherical coords (θ,φ)
  - Convert direction to spherical coords, direct lookup

- Cubemaps popular (6 faces of cube)
  - Take biggest (abs) of (x,y,z)
  - Divide/renorm by it to get coords
  - E.g. if +z, use x/z, y/z, z=+1
  - Cubemap coord to vec: normalize
  - Easy convert bet cube, latlong

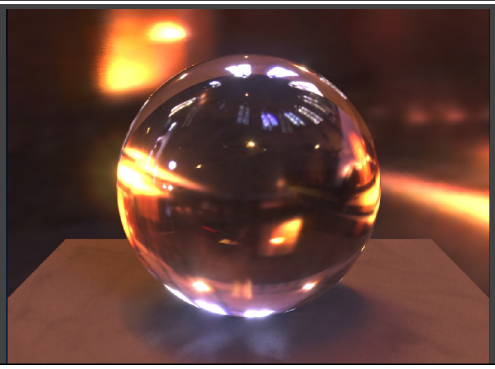
High Dynamic Range

- Ratio of brightest to darkest environment regions can be a million to 1. High Dynamic Range HDR
- Acquiring (floating point) HDR envmaps is good
- Tonemap as needed for display (large topic)
- Accurate HDR values needed for accuracy
  - When considering diffuse/specular BRDFs
  - Tonemap mirror reflections, viewing environment
  - Photograph a mirror ball with HDR or use many HDR envmaps found online
  - See Debevec 97, 98 for discussion of HDR
  - (HDR Imaging images from Wikipedia)

Environment Maps Generally

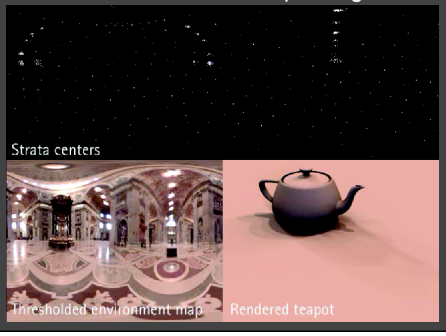
- Mirror reflections good but not general
- Can we render all effects with envmap?
- Simple idea, envmap on large sphere around scene
  - When path leaves scene, it hits envmap
  - Consider emission (radiance) from given envmap pixel
  - Significant noise/aliasing for high-frequency HDR envmaps (e.g. you may almost always miss the sun)
- Challenge is we effectively have millions of lights
  - Need to importance sample the environment map
  - Effectively extend next-event estimation to envmaps
  - Or identify bright lights (Debevec 98,99 asked undergraduates to trace this out manually!)

HDR Environment Illumination

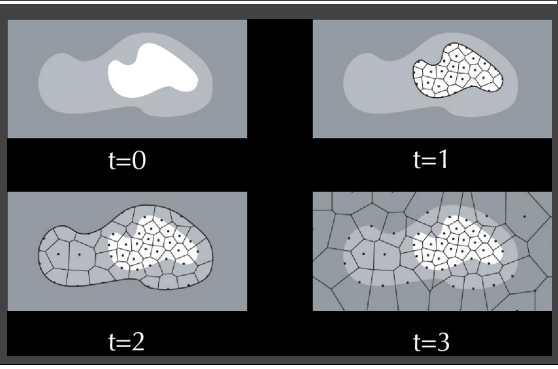


Structured Importance Sampling

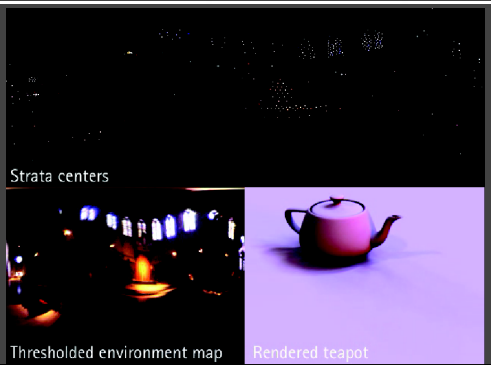
- Goal: Reduce environment to point lights



Hierarchical Stratification



Structured Importance Sampling



Lat-Long Importance Sampling

- Simple alternative (PBRT book)
- Multidimensional importance sampling θ,φ
  - Generate a numerical 1D CDF along φ integrating over all θ
  - For each φ generate a numerical CDF over θ
  - Essentially creates axis-aligned (lat-long) cells
  - Compatible with any sampling scheme (stratified)
  - I implemented this at Pixar (circa 2011)
  - Done properly, PDF (almost) cancels lighting (can work out on board). Many subtleties involved, MIS
- Other Simplifications
  - Integrate lighting in strata to create point lights
  - Jitter only for visibility (if at all)

Sampling General 2D Distributions

- Treat Lighting as general 2D distribution
  - Doing this for 1 color channel, take avg for probs

$$\iint L(\theta,\varphi)\sin\theta\, d\theta\, d\varphi = \iint L(u,v)\, du\, dv \quad u=\cos\theta = z, v=\varphi$$

- Normalize to convert to probability to sample from
  - Note that probability distribution also enables MIS

$$p(u,v) = \frac{L(u,v)}{|L|} \qquad |L| = \iint L(u,v)\, du\, dv$$

  - For direct lighting, illumination cancels out (careful re color)
    - Will bring down a term of L_c / L_{avg}

$$L_o(\omega_o) = \left\langle \frac{L(u,v)V(u,v)f(u,v;\omega_o)\max(0,n\cdot\omega_i(u,v))}{p(u,v)} \right\rangle = |L| \left\langle V(u,v)f(u,v;\omega_o)\max(0,n\cdot\omega_i(u,v)) \right\rangle$$

How to Sample 2D Distribution

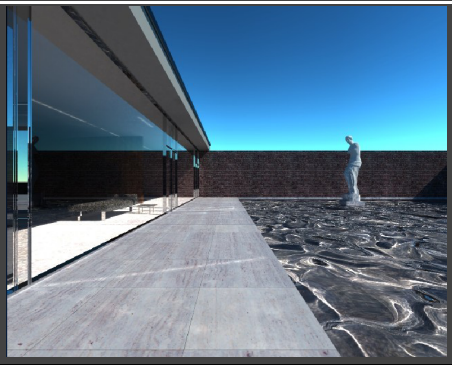
- Form (numerical) 1D CDFs $p(v) = \int p(u,v)\, du \qquad p(u\,|\,v) = \frac{p(u,v)}{p(v)}$
- Generate 2 random numbers in standard way
  - Use numerical 1D CDF inversion to find v, then u
  - Works with any sampling scheme (stratified etc.)
- Note that I've done everything in integrals, but you will need to discretely sum, dividing by resolution (and consider factors of Pi for environment maps)

$$|L| = \frac{4\pi}{n_u n_v} \sum_u \sum_v L(u,v) \qquad p(v) = \frac{2}{n_u} \sum_u p(u,v)$$

- Or look up SIS paper, code (Agarwal et al. 03)

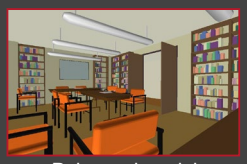
From UCB class many years ago

Mies House: Swimming Pool



Texture Mapping

- Important topic: nearly all objects textured
  - Wood grain, faces, bricks and so on
  - Adds visual detail to scenes
- Meant as a fun and practically useful lecture

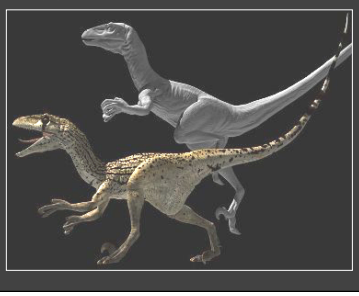


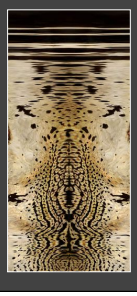
Polygonal model

With surface texture

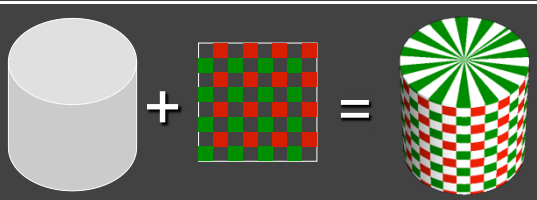
Adding Visual Detail

- Basic idea: use images instead of more polygons to represent fine scale color variation





Parameterization

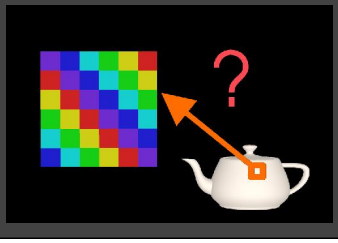


geometry image texture map

- Q: How do we decide *where* on the geometry each color from the image should go?

How to map object to texture?

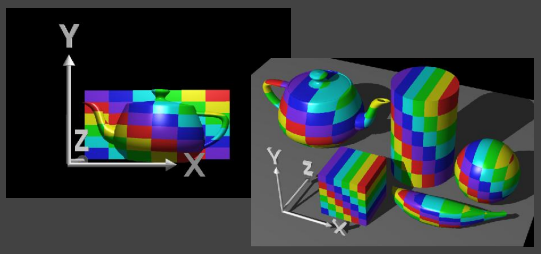
- To each vertex (x,y,z in object coordinates), must associate 2D texture coordinates (s,t)
- So texture fits "nicely" over object



Option: it's the artist's problem

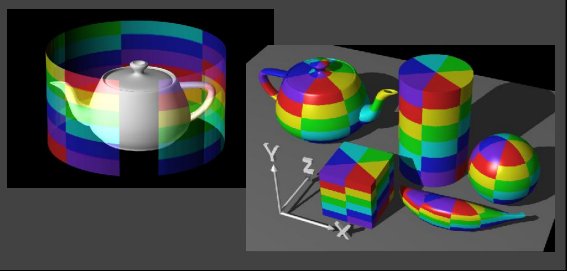
Planar mapping

- Like projections, drop z coord (s,t) = (x,y)
- Problems: what happens near z = 0?



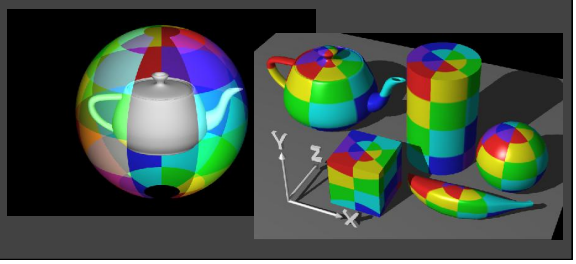
Cylindrical Mapping

- Cylinder: r, θ, z with (s,t) = (θ/(2π),z)
  - Note seams when wrapping around (θ = 0 or 2π)

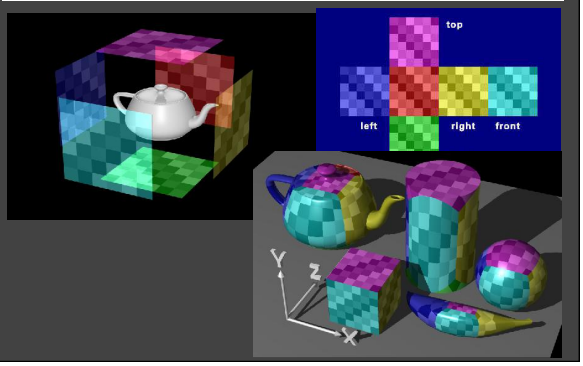


Spherical Mapping

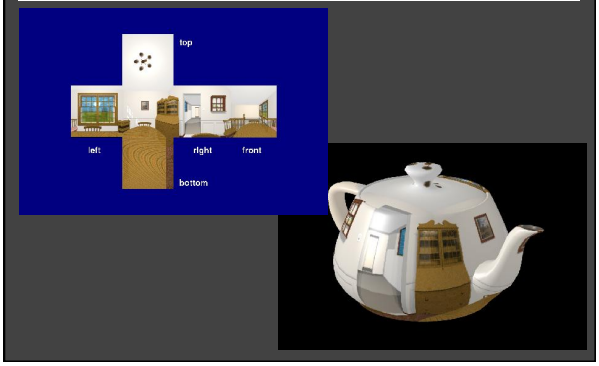
- Convert to spherical coordinates: use latitude/long.
  - Singularities at north and south poles



Cube Mapping



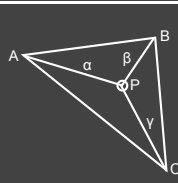
Cube Mapping



Interpolating Texture Coordinates

- Texture Coordinates at Vertices of Triangle
- How to compute coordinate at intersection?
- Use barycentric coordinates from in triangle test
- Same weights to combine texture coordinates
- Then use texture coordinates to look up texture
- Textures can also be procedural (use a formula)

Ray inside Triangle



$$P = \alpha A + \beta B + \gamma C$$
$$\alpha \geq 0, \beta \geq 0, \gamma \geq 0$$
$$\alpha + \beta + \gamma = 1$$

$$P - A = \beta(B - A) + \gamma(C - A)$$
$$0 \leq \beta \leq 1\ ,\ 0 \leq \gamma \leq 1$$
$$\beta + \gamma \leq 1$$

Texture Map Filtering

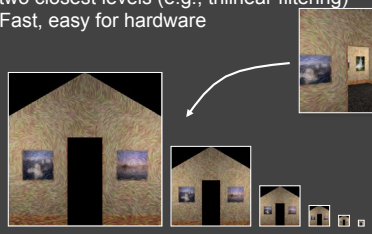
- Naive texture mapping aliases badly
- Look familiar?

```
int uval = (int) (u * denom + 0.5f);
int vval = (int) (v * denom + 0.5f);
int pix = texture.getPixel(uval, vval);
```

- Actually, each pixel maps to a region in texture
  - |PIX| < |TEX|
    - Easy: interpolate (bilinear) between texel values
  - |PIX| > |TEX|
    - Hard: average the contribution from multiple texels
  - |PIX| ~ |TEX|
    - Still need interpolation!

Mip Maps

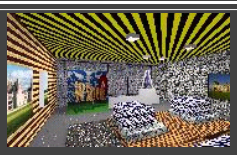
- Keep textures prefiltered at multiple resolutions
  - For each pixel, linearly interpolate between two closest levels (e.g., trilinear filtering)
  - Fast, easy for hardware



- Why "Mip" maps?

MIP-map Example

- No filtering:



AAAAAAAGH
MY EYES ARE BURNING

- MIP-map texturing:

Where are my glasses?

Texture Mapping Applications

- Modulation, light maps
- Bump mapping
- Displacement mapping
- Illumination or Environment Mapping
- Procedural texturing
- And many more

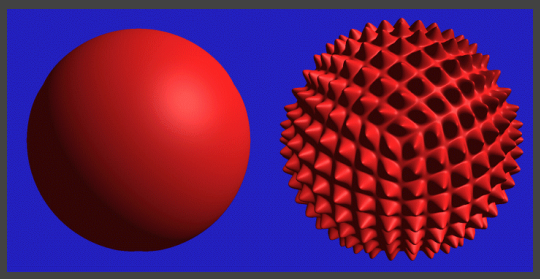
In physically-based rendering, texture doesn't give color directly, rather controls some attribute (like diffuse/specular BRDF coefficient, roughness etc.)

Bump Mapping

- Texture = change in surface normal!

Sphere w/ diffuse texture | *Swirly bump map* | *Sphere w/ diffuse texture and swirly bump map*

Displacement Mapping



Environment Maps

Images from *Illumination and Reflection Maps:*
Simulated Objects in Simulated and Real Environments
Gene Miller and C. Robert Hoffman
SIGGRAPH 1984 "Advanced Computer Graphics Animation" Course Notes

Solid textures

Texture values indexed by 3D location (x,y,z)

- Expensive storage, or
- Compute on the fly, e.g. Perlin noise →