VGP353 – Week 2

- Agenda:
  - Shadow Textures
    - Improvements over planar projected shadows
    - Implementation details
    - Optimizations
- Assignments:
  - Assignment #1 due
  - Begin assignment #2

8-April-2008

Planar Projected Shadows

- As discussed previously, planar projected shadows have a number of faults

Planar Projected Shadows

- As discussed previously, planar projected shadows have a number of faults
  - No self-shadowing
  - Can only cast shadows on the ground plane

Planar Projected Shadows

- As discussed previously, planar projected shadows have a number of faults
  - No self-shadowing
  - Can only cast shadows on the ground plane
  - Can only cast shadows on a ***flat*** ground plane

Planar Projected Shadows

- As discussed previously, planar projected shadows have a number of faults
  - No self-shadowing
  - Can only cast shadows on the ground plane
  - Can only cast shadows on a ***flat*** ground plane
- Shadow textures fix *most* of these problems

Shadow Textures

- Algorithm outline:
  - Render shadow caster to a texture from the point of view of the light
    - Texture background is the color of the light
    - Object is rendered in black
  - Using *projective texturing* cast the shadow texture onto each shadow receiver
  - Use the sampled texture color as the light color

Shadow Textures

- Advantages?

Original image from *Battlefield 1942* © Copyright Digital Illusions CE 2002.

8-April-2008

Shadow Textures

- Advantages?
  - Can cast shadows on non-flat surfaces
  - Can cast shadows on multiple objects

Shadow Textures

- Advantages?
  - Can cast shadows on non-flat surfaces
  - Can cast shadows on multiple objects
- Disadvantages?

Shadow Textures

- Advantages?
  - Can cast shadows on non-flat surfaces
  - Can cast shadows on multiple objects
- Disadvantages?
  - No self-shadowing
    - Shadow *maps* will solve this problem...next week
  - Requires render-to-texture pass for *each* shadow caster for each light
  - Shadow receiver must sample multiple shadow textures

Original image from *Battlefield 1942* © Copyright Digital Illusions CE 2002.

8-April-2008

Shadow Texture Creation

- Setup modelview-projection (MVP) matrix to render from the light looking at the object
  - The eye-point is actually the light position
  - Set the FoV to just enclose the object
    - The object's bounding box is helpful here
- Render object as shadow
  - Clear the color buffer to the light's color
  - Render the object as solid black
    - Can “fake” soft shadows by using distance from light (eye) to determine color: closer to the light is darker, farther is lighter

Determining Receiver / Caster

- For each shadow texture, determine which objects are potential receivers
  - If the object is *completely* on the opposite side of the near plane from the light, it is a candidate

Projective Texturing

- Does what it says: projects a texture onto an object
- This is a *perspective* projection, so what is needed to make it “work”?

Projective Texturing

- Does what it says: projects a texture onto an object
- This is a *perspective* projection, so what is needed to make it “work”?
  - Divide by Z...just like perspective viewing projections
    - Uses the q texture coordinate

Projective Texturing

- Algorithm outline:
  - Use object-space vertex positions as initial texture coordinates
  - Transform object-space texture coordinate to projector-space
  - Apply perspective transformation
    - Same MVP matrix as is used to render to the texture
  - Scale and bias coordinates from [-1, 1] to [0, 1]
    - Unless one of the mirroring wrap modes is being used

Projective Texturing

- Uses different sampling functions in GLSL:
  - `texture[123]DProj` vs `texture[123]D`
  - Use these functions instead of doing the perspective divide by hand
  - Cubic textures not supported. Why?

Projective Texturing

- Uses different sampling functions in GLSL:
  - `texture[123]DProj` vs `texture[123]D`
  - Use these functions instead of doing the perspective divide by hand
  - Cubic textures not supported. Why?
    - The q component is already used as part of the texture lookup!

Projective Texturing

- What happens if the point is *behind* the projection point?

  Hint: What happens if an object is behind the eye?

Projective Texturing

- What happens if the point is *behind* the projection point?

  Hint: What happens if an object is behind the eye?

  - It gets a *negative* Z (or q) value
  - The projection then “flips” the position
    - Because it divides by a negative number

Projective Texturing

- What happens if the point is *behind* the projection point?

  Hint: What happens if an object is behind the eye?

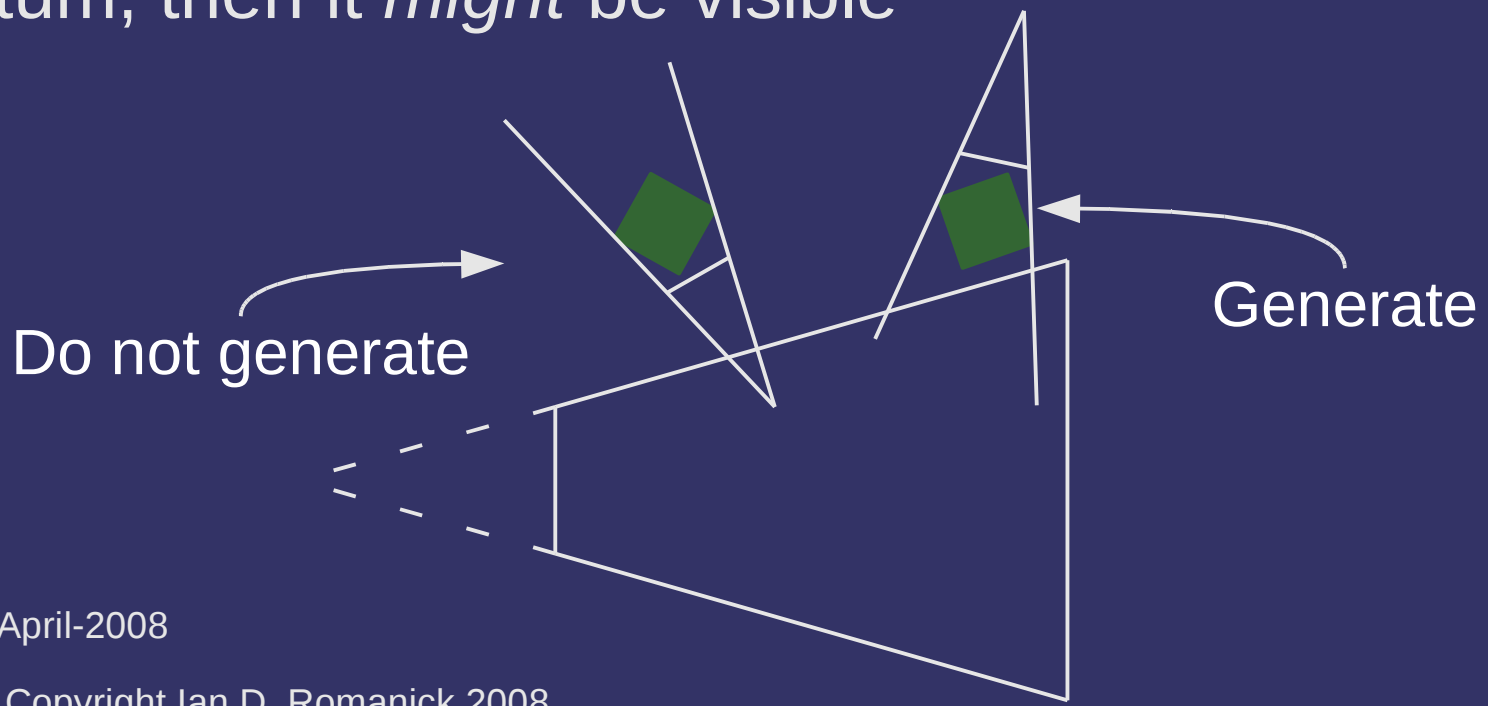
  - It gets a *negative* Z (or q) value
  - The projection then “flips” the position
    - Because it divides by a negative number
- Shadows are cast on objects on the opposite side of the light from the caster
  - Just a fact of projective texturing
  - Reject points with q less than near-plane distance

Optimizations

- Performance problems with shadow textures:
  - Lots of textures need to be generated *per frame*
  - Shadow receivers need to read lots of textures
- General speed-up techniques:
  - Regenerate a texture only if light or caster moved
  - Generate textures for shadows that might intersect view volume
  - Apply texture only to objects that might be shadowed
  - Composite multiple shadow textures together

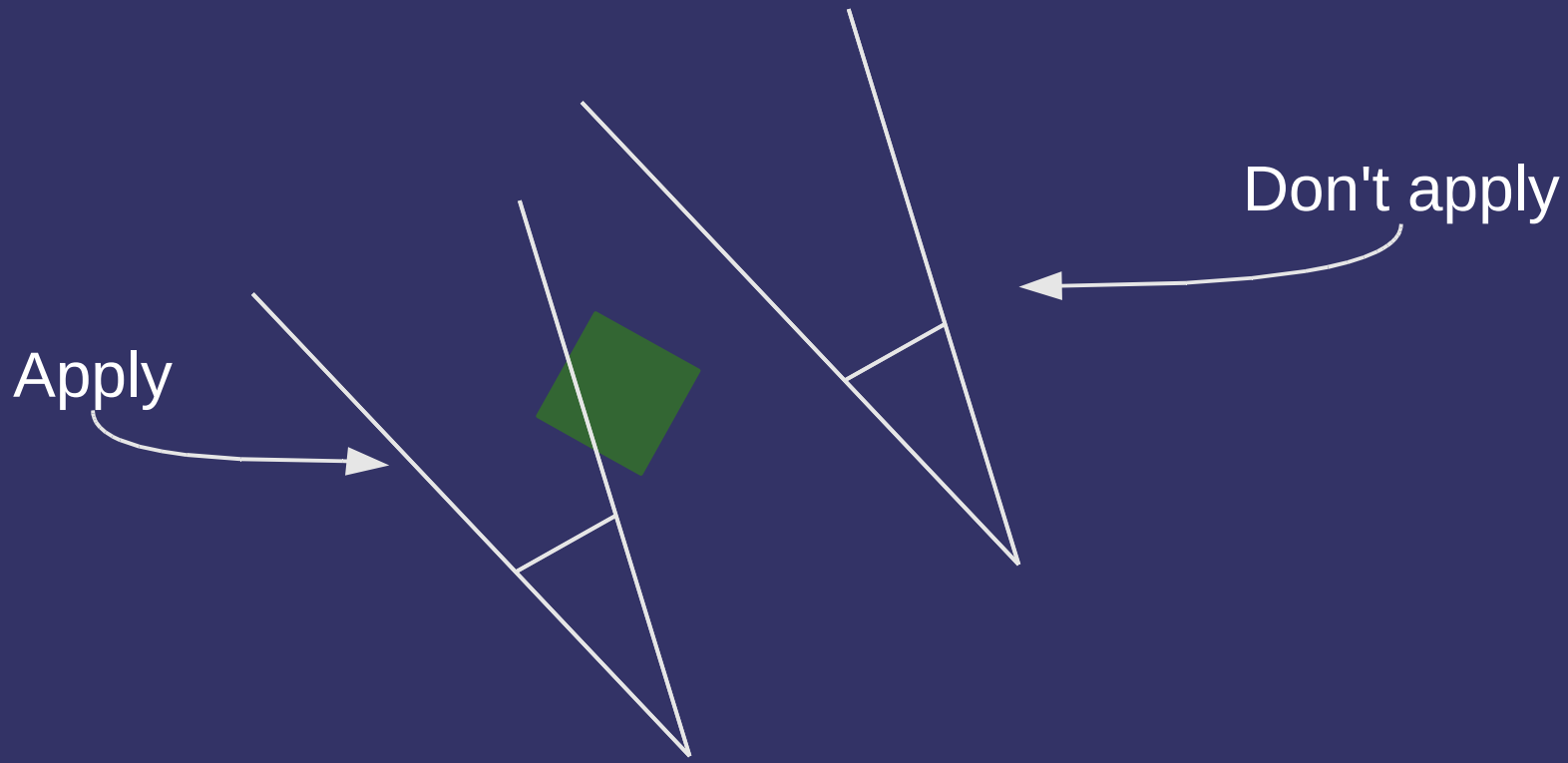
Optimizations

- Generate textures for shadows that might intersect view volume
  - Each shadow texture has an associated frustum
    - “View” frustum used to render the shadow texture
  - If the shadow's frustum intersects the view (eye) frustum, then it *might* be visible



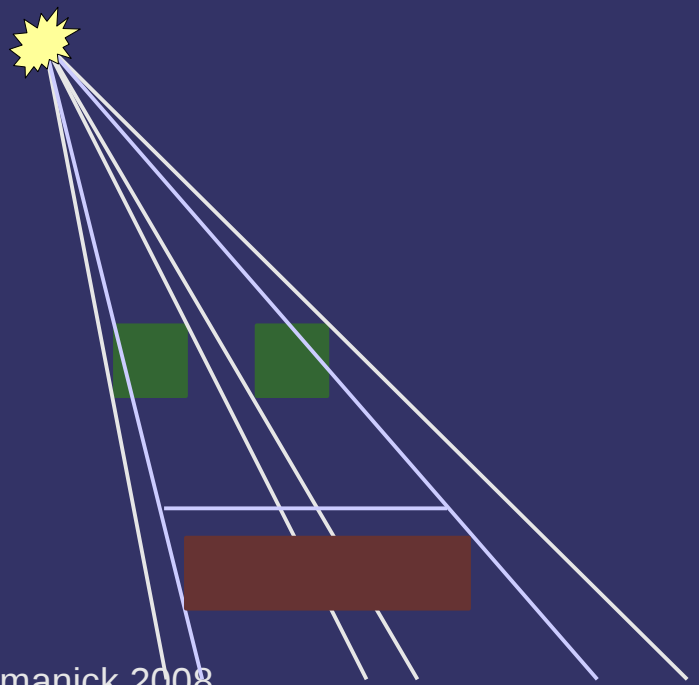
Optimizations

- Apply texture only to objects that might be shadowed
  - Any object that does not intersect the shadow's frustum is not a receiver



Optimizations

- Composite multiple shadow textures together
  - Many casters can affect all members of a group of receivers
  - Create a new shadow texture by compositing all potential casters shadow textures together
  - Project each shadow texture onto the near-plane



References

Bloom, Charles. *Projective Shadow Mapping* [article on-line]. June 30, 2000, accessed April 4, 2008; available from http://www.cbloom.com/3d/techdocs/shadowmap.txt; Internet.

Bloom, Charles, and Teschner, Phil. *Advanced Techniques in Shadow Mapping* [article on-line]. June 3, 2001, accessed April 4, 2008; available from http://www.cbloom.com/3d/techdocs/shadowmap_advanced.txt; Internet.

Next week...

- Quiz #1!
  - Shadow terminology
  - Projective planar shadows
  - Shadow textures
- Assignment #2 due
- Shadow maps
  - Similarities and improvements to shadow textures

Legal Statement

This work represents the view of the authors and does not necessarily represent the view of IBM or the Art Institute of Portland.

OpenGL is a trademark of Silicon Graphics, Inc. in the United States, other countries, or both.

Khronos and OpenGL ES are trademarks of the Khronos Group.

Other company, product, and service names may be trademarks or service marks of others.

8-April-2008