CSCI 580 - Project
Shadow Mapping

Sijie Bu
Jessi Bustos
Shao Ling Tan

Shadow Mapping

- A natural extension to the z-buffering hidden surface removal
  - If a pixel is in shadow, then the light would not see it
- Need to be able to render 2 images:
  - One from the the perspective of the light source
  - One from the perspective of the viewer (camera)

Algorithm Description

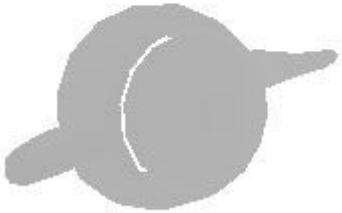
- Create additional renderers for each non-ambient light
  - Set each additional renderer with the view frustum of the given light
  - Render the shadow buffer images as usual
- In the main renderer
  - When computing the color of each visible pixel, do it for each light
    - Project the current pixel to light space
    - Compare the projected z-value with the shadow map z-value
    - If projected z-value is greater than the stored one, then it’s in shadow

Starting Point

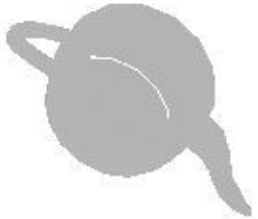
- Extended Gz Library from our homework 6 to allow for shadow mapping capabilities.
- Modified the base code to now include one main renderer and shadow map renderer(s).
    - Details on next slide

Algorithm Implementation (1)

- Modified the GzLight struct to contain a pointer to the corresponding shadow map renderer
- Modified the Application code to initialize the shadow map renderers, and push transformation matrices onto these renderers
- Added a “shadow map” rendering mode to the renderers, which skips color computation, and visualizes the shadow map with the z-value information



Shadow map #1



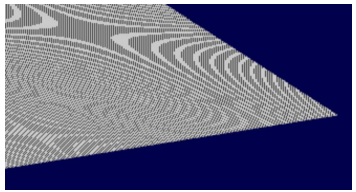
Shadow map #2

Algorithm Implementation (2)

- Shadow lookup and rendering in the rasterization process
- The Ximage of the main renderer is inverted to allow screen-space-to-image-space projections
- Each visible pixel is evaluated for shadow against every light
  - Projected first to image space, then to light space
  - Compare the projected z-value with the shadow map z-value
    - Use nearest-neighbor interpolation
  - Decrease the ambient and diffuse contributions of current light if the projected z-value is greater than the shadow map z-value

Artifacts

- Moiré-like pattern: shadow acne
  - Self-shadowing errors and
  - High frequency in nature
- Low-pass filtering with a bias value:
  - zPixel - zShadowMap > bias
- Note:
  - Bias too small: no effect
  - Bias too large: Peter Panning or no shadows at all



Bias: 8.8×10^6
(Peter Panning)

Bias: 8×10^7
(No shadows visible)

Different Shadow Bias

0 | 5 | 500 | 5×10^4 | 5×10^5

Thank You

Bibliography

1. L. Williams, “Casting curved shadows on curved surfaces,” 1978.
2. “Project 7 - Shadow Mapping,” The University of Utah, [Online]. Available: https://graphics.cs.utah.edu/courses/cs6610/spring2020/?prj=7. [Accessed 24 4 2021].
3. “Tutorial 16 : Shadow mapping,” [Online]. Available: http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/. [Accessed 24 4 2021].
4. "new __gluInvertMatrix() function (Mesa bug 6748)," 27 08 2007. [Online]. Available: https://gitlab.freedesktop.org/eric/mesa/commit/3069f3 484186df09b671c44efdb221f50e5a6a88#57de6276c46a97f5e8629ed0c426cdf64d6a4103. [Accessed 24 04 2021]
5. "Warp: Apply projective or affine transformation to an image," The MathWorks, Inc., [Online]. Available: https://www.mathworks.com/help/vision/ref/warp.html;jsessionid=096ebcfe483c693c8d89c12c3495#bun4186-1_sep_mw_e47bf23e-68c7-4ef1-bf1b-18ae877bd90c. [Accessed 26 04 2021].
6. T. Annen, T. Mertens, H.-P. Seidel, E. Flerackerts and J. Kautz, "Exponential shadow maps," GI '08: Proceedings of Graphics Interface 2008, pp. 155-161, 2008.
7. Microsoft, "Common Techniques to Improve Shadow Depth Maps," Microsoft, [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/dxtecharts/common-techniques-to-improve-shadow-depth-maps?redirectedfrom=MSDN. [Accessed 27 04 2021].
8. "Filtering an Image," [Online]. Available: https://northstar- www.dartmouth.edu/doc/idl/html_6.2/Filtering_an_I magehvr.html. [Accessed 26 04 2021].