VGP351 – Week 10

- Agenda:
  - More anti-aliasing:
    - AA during primitive rendering
    - FSAA
      - Supersampling
      - Multisampling
    - Temporal AA
  - Discuss the final

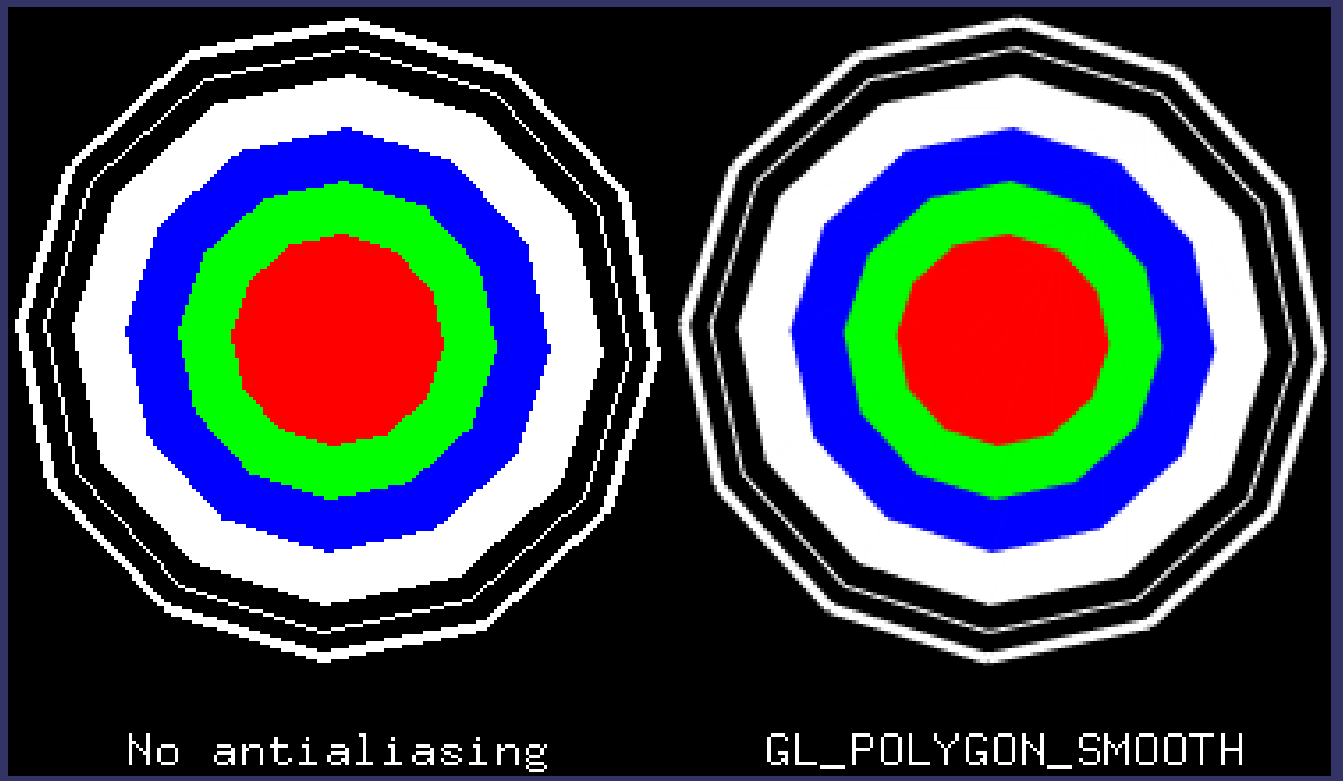
9-December-2009

Aliasing During Rendering

- 3D world is sampled at fixed locations
  - We call these locations pixels
  - The resolution is the sample rate
- If the world has higher frequency elements than the sample rate can support, we get aliasing
  - In other words, if there are details smaller than two pixels, there will be aliasing

Rasterization AA

- Edges pass partially through pixel locations
  - We can calculate which pixels the edge intersects, and give some color to each



Rasterization AA

- However...
  - Hardware doesn't optimize for this case anymore
    - So it either doesn't exist or is slow
  - Only really helps where edges meet
    - Does nothing for aliasing caused by the shader within the polygon
  - Quality depends on back-to-front rendering order
    - Just like transparency blending, and for the same reason

Full Screen Anti-Aliasing

- What to do?
  - If too few samples are the problem... get more samples

Supersampling

- Obvious answer:
  - Render at much higher resolution and down-sample
  - *Ideally* performance decreases linearly with the increase in samples
    - In reality, performance may be worse than that due to caches, etc.

Supersampling

- Supersampling executes the fragment pipeline for each sample
  - Adds memory bandwidth cost
  - Adds computation cost

Multisampling

- Increases the sample rate, just like supersampling
  - The same value is used for each subsample within a pixel

Multisampling

- Multisampling executes the fragment pipeline once per pixel
  - Adds memory bandwidth cost
  - Keeps the same computation cost

Multisampling

- Sample buffers are a property of the drawable
  - Must be requested when the drawable is created

```
SDL_GL_SetAttribute(SDL_GL_MULTISAMPLEBUFFERS, 1);
SDL_GL_SetAttribute(SDL_GL_MULTISAMPLESAMPLES, 2);
```

- Multisample rasterization is separately enabled:

```
glEnable(GL_MULTISAMPLE);
```

Multisampling

- Can also be used with alpha test
  - A special mode will cause the fragment alpha value to modify the coverage mask

```
glEnable(GL_SAMPLE_ALPHA_TO_COVERAGE);
```

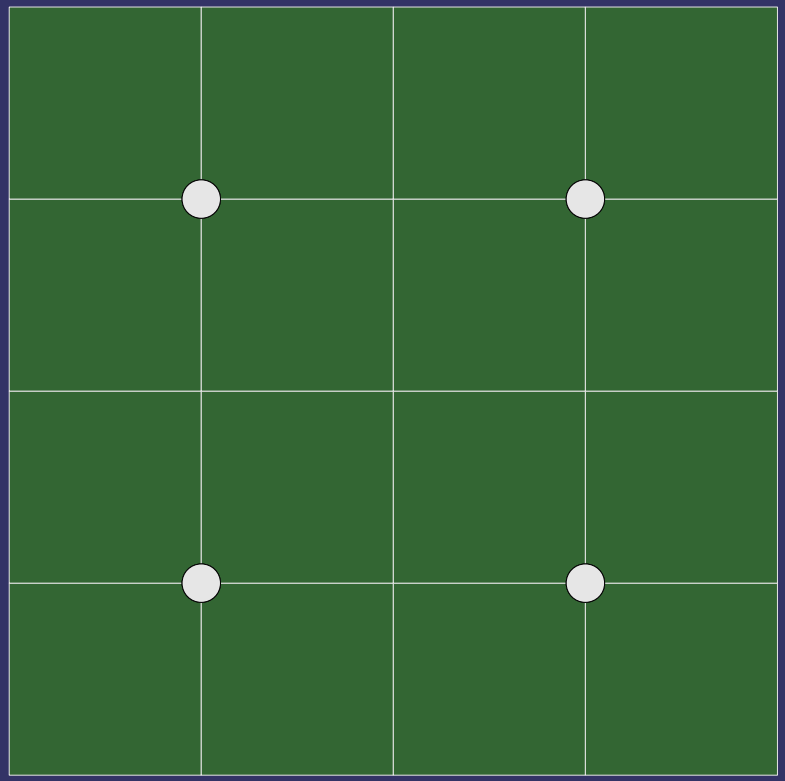
  - This can eliminate the need to alpha blend with alpha test
    - Yay! No sorting!

Multisample Resolve

- When a pixel is accessed, the samples are “resolved”
  - Accessing the pixel could be:
    - Reading via `glCopyTexImage` or `glReadPixels`
    - Displaying via `SwapBuffers`
  - Resolving involves filtering the samples together in some manner to generate a single color value

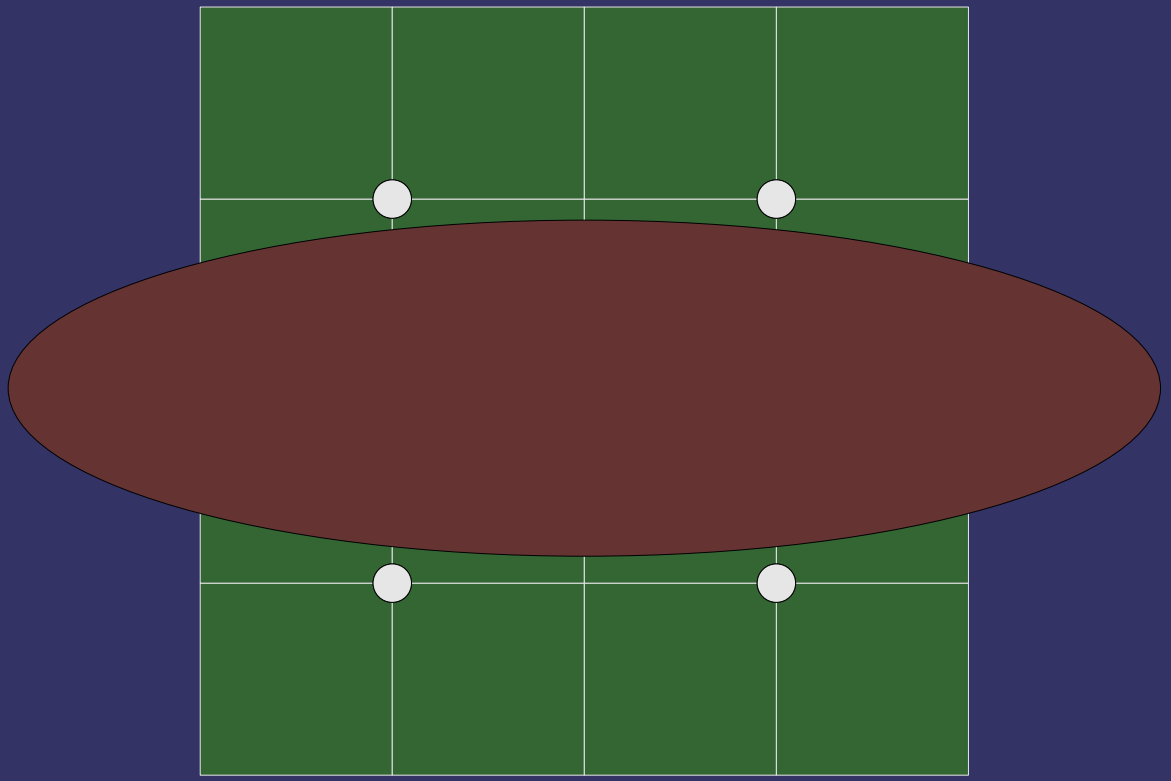
Sample Patterns

- Coverage can be sampled in many ways



Sample Patterns

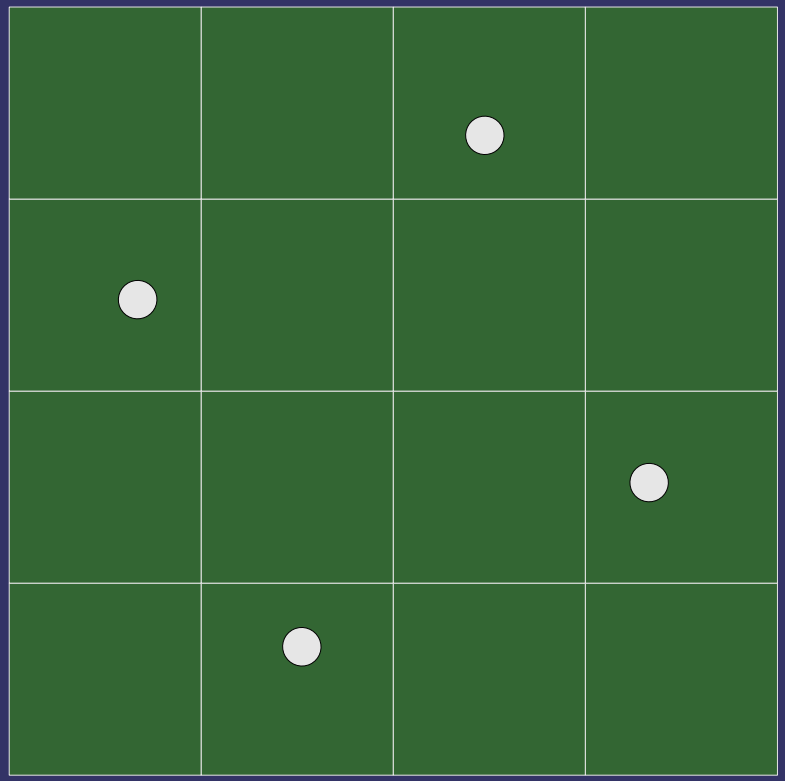
- Coverage can be sampled in many ways



Sample Patterns

- Coverage can be sampled in many ways

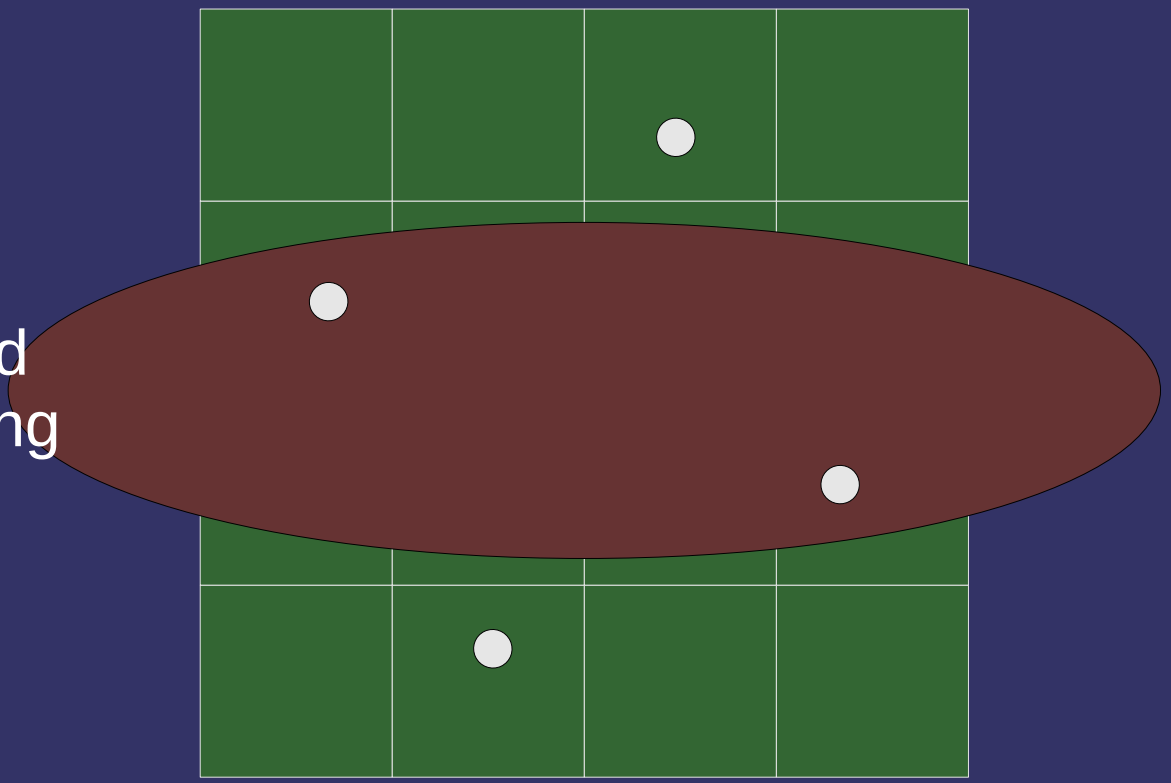
Rotated Grid Supersampling



Sample Patterns

- Coverage can be sampled in many ways

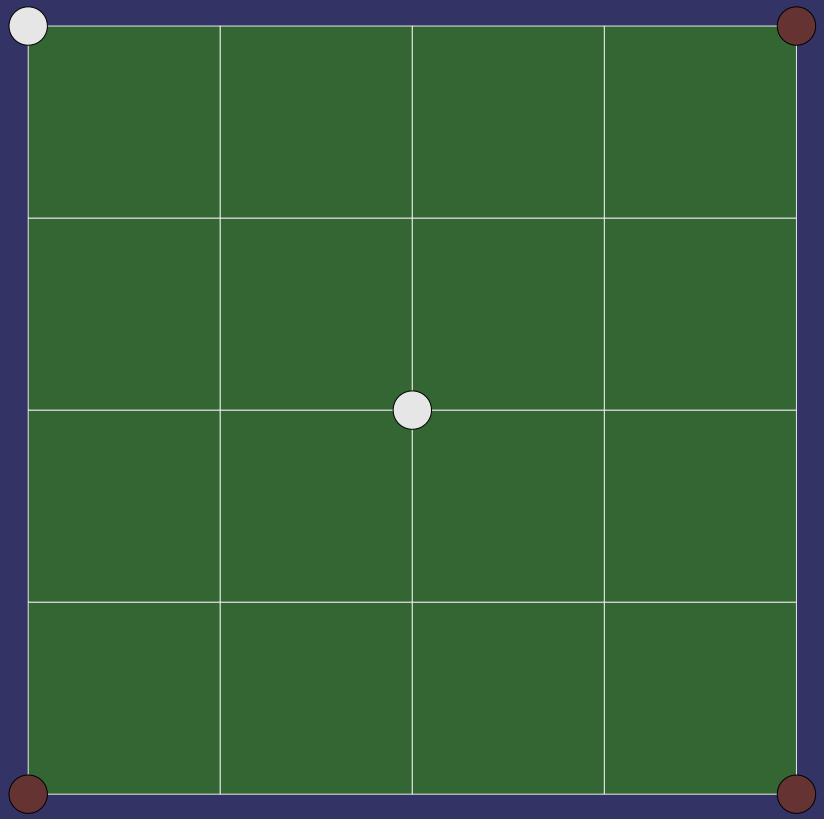
Rotated Grid Supersampling



Sample Patterns

- Coverage can be sampled in many ways

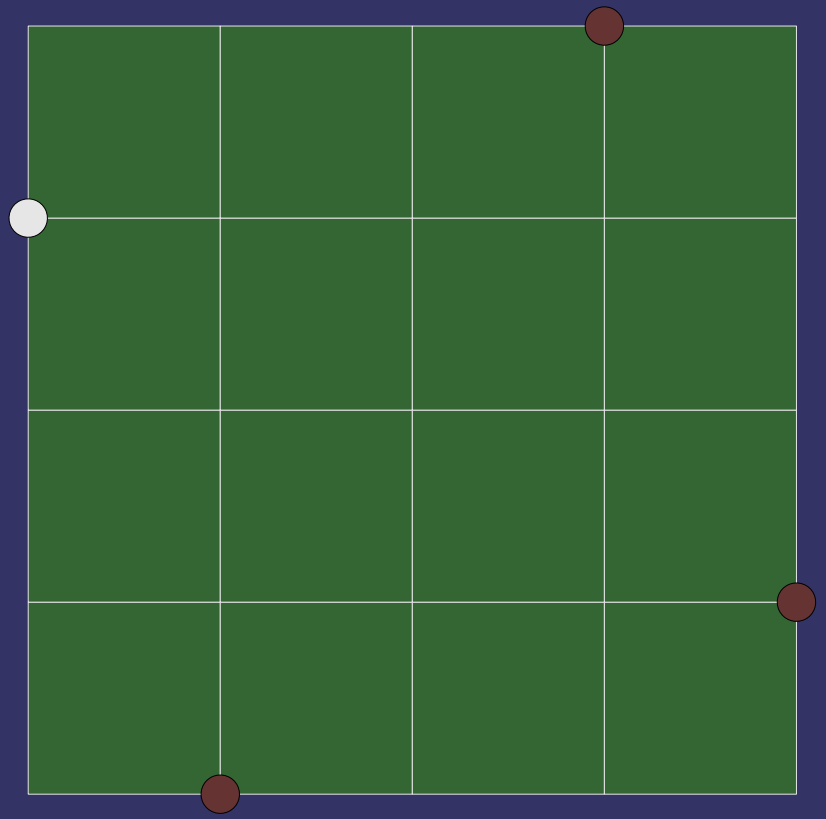
Quincunx



Sample Patterns

- Coverage can be sampled in many ways

FLIPQUAD



References

http://developer.nvidia.com/object/gdc_ogl_multisample.html

http://graphics.stanford.edu/courses/cs248-07/schedule.php

- Look at the lecture notes from October 11th
- Kurt Akeley is one of the original designers of OpenGL

http://en.wikipedia.org/wiki/Alpha_to_coverage

Temporal Aliasing

- Caused by the same problem as any aliasing:
  - Not enough samples *through time*
  - Have infer the missing information between rendered frames
  - Even if the brain infers correctly, the images seem unnatural

Temporal Aliasing

- Examples:

  http://www.youtube.com/watch?v=W1UbXriii_Y

  http://www.youtube.com/watch?v=cWGn6_EH2gM

  http://www.youtube.com/watch?v=4wW0txXoan8

Temporal Aliasing

- Film movie cameras generate motion blur...
  - Shutter is nearly a semi-circle that spins
    - For a little less than half of 1/24th of a second, the film is exposed to light
  - When the film is not exposed, it advances to the next frame

Temporal Aliasing

- Film movie cameras generate motion blur...
  - Shutter is nearly a semi-circle that spins
    - For a little less than half of 1/24th of a second, the film is exposed to light
  - When the film is not exposed, it advances to the next frame
- We can “defeat” this
  - Use a smaller shutter
    - The movie Gladiator used ~45° during some fight scenes
  - The film is exposed for less time
  - Results in *less* realism

Temporal Aliasing

- Examples:

  http://www.youtube.com/watch?v=czQfPdPaK_8

Temporal Anti-Aliasing

- Barriers in real-time graphics:
  - Limited by the refresh rate of the display
    - Usually 60fps
  - Limited by how fast we can render

Temporal Anti-Aliasing

- Naïve approach:
  - Render multiple frames at different time steps
  - Blend the results

Temporal Anti-Aliasing

- Pros:
  - Easy to implement
  - Produces good results *with fine enough time steps*
- Cons:
  - Expensive!
  - Produces really bad results if the time steps are not close enough

Temporal Anti-Aliasing

- We can fake motion blur on individual objects
  - Stretch the object from its previous position to its current position

Temporal Anti-Aliasing

- Algorithm overview:
  - Render the object once normally
  - Render a second time with alpha blending:
    - In the vertex shader, transform each vertex by it's current and previous transformation matrix
      - The vector, $\mathbf{m}$, between the two is the motion vector
    - Compare $\mathbf{m}$ and $\mathbf{n}$
      - If $\mathbf{m}$ and $\mathbf{n}$ point the same direction, use the current frame transform
      - Otherwise, use the previous frame transform and decrease the alpha

Temporal Anti-Aliasing

- Pros:
  - Produces good results on individual objects
  - Inexpensive
- Cons:
  - May be very complex to add to some shaders
  - Really only works on individual objects
    - Doesn't help if the camera is moving quickly

References

Wloka, M. and Zeleznik, R. “Interactive Real-Time Motion Blur.” The Visual Computer 12 (1996): 283 – 295.
http://graphics.cs.brown.edu/research/pub/papers/viscom-motionblur.ps

Wloka, M. "Implementing Motion Blur & Depth of Field using DirectX 8," *Meltdown 2001*, July 2001.
http://www.microsoft.com/mscorp/corpevents/meltdown2001/ppt/Externals/NVidia_MotionBlurDepthOfField.ppt

Next week...

- All assignments due
- The final!
  - **Wednesday** at 5:30PM... do **NOT** be late!

Legal Statement

This work represents the view of the authors and does not necessarily represent the view of Intel 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.

9-December-2009