Visualizing Time-Dependent Atomic Data in OpenGL

By: Tyler Spears
Mentor: Dr. Bijaya Karki

Computer Visualization and OpenGL

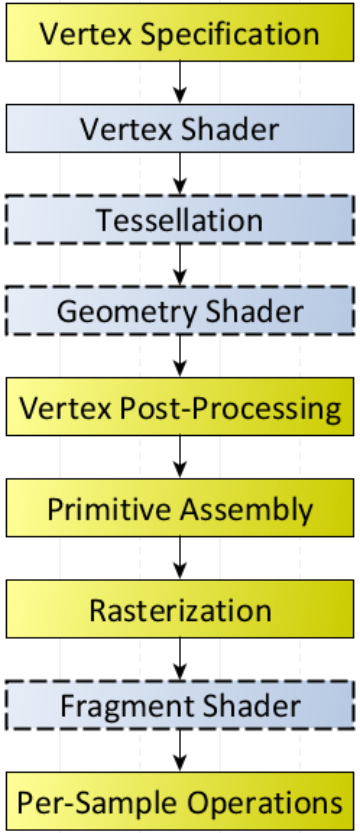
1

Scientific Visualization

- Scientific visualization is the field of representing scientific data in an intuitive and effective manner.
- Visualization is the medium between hard data and a human’s intuitive understanding.
- A vital component to the scientific process, it facilitates understanding to the scientific community and the general public.

OpenGL

- OpenGL (Open Graphics Library) is a platform-independent specification used to create 2D and 3D scenes and images.
- A very popular choice for visualization, especially when efficiency and flexibility are needed.
- Usually implemented in C++
- Also specifies GLSL (GL Shading Language), a C-like language used to program custom shaders.



OpenGL Rendering Pipeline

Rendering 2

Atomic Data

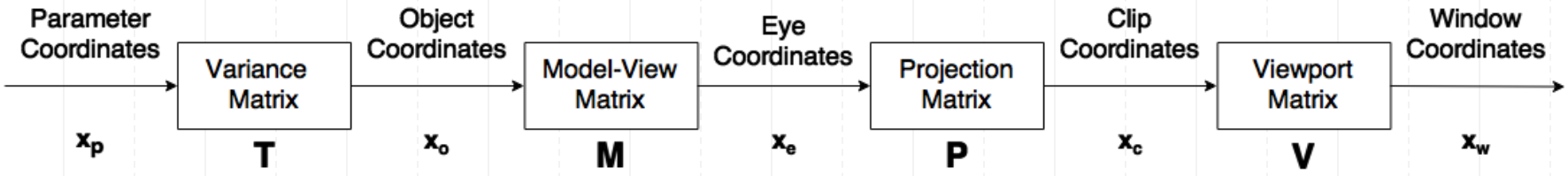
- Focus was on data representing atomic properties taken from simulations.
- Resulted in large amounts of data detailing magnetization values, distance traveled, velocity, etc.
- Atoms best represented as simple spheres with a single color, as the exact location of each sub-atomic particle is difficult (or even impossible) to determine.

Methods

- This project attempted to render atomic data in two ways:
  - Rendering 2D “pseudo-spheres” using point pixels in 3D space
  - Rendering true 3D spheres in 3D space using freeGLUT (GL Utility Tooklit)

Rendering Using 2D Point Pixels

- Through the use of custom shaders, point pixels are rendered as 2D “pseudo-spheres”
  - Coefficients of quadratic equation of the sphere are encoded into a matrix, forming the parameter space
  - With the use of the Variance, Model, View, Projection, and Viewport matrices, a bounding box is calculated to determine the dimensions of the point sprite



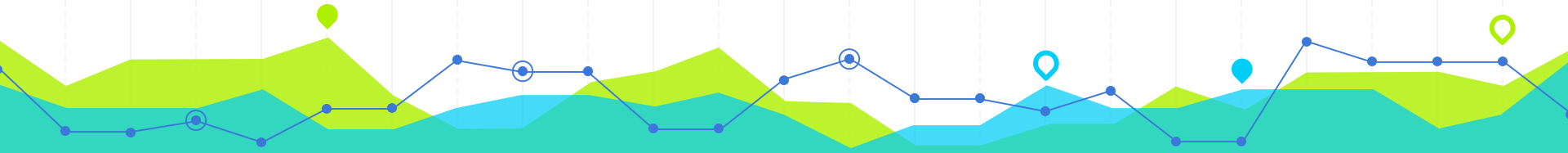
Sequence of coordinate spaces in OpenGL, with an added Parameter Coordinate Space for simplified matrix calculations

Rendering Using 2D Point Pixels

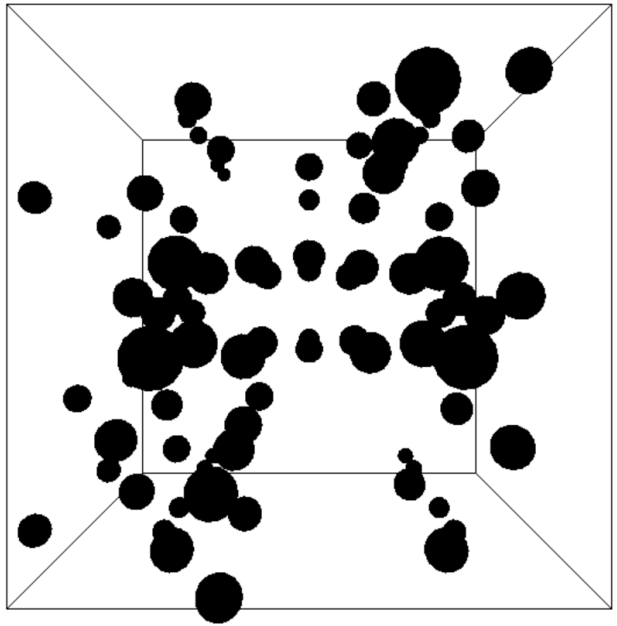
- A ray-surface intersection quadratic equation is then solved to determine which fragments are culled.
- Using Phong shading, ambient, diffused, and specular components light the surface of the “pseudo-spheres”
- Unfortunately, due to several compilation and runtime errors in the shader code, this method was not fully implemented in the project.
- So, the second method was used to render the data.

Rendering Using 3D Spheres

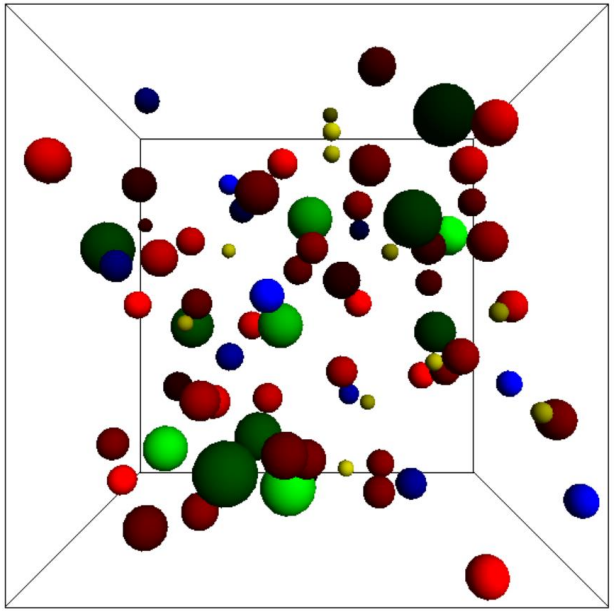
- 3D spheres were rendered using freeGLUT
- These spheres were placed into a scene according to their position in 3D space, a lighting model was implemented, and their color encoded certain scalar values.
  - In this project, atomic distance data (distance traveled from the origin), although any scalar value could easily be implemented.



Conclusion 3



All atoms in their initial position



Atoms in the middle of their simulation. Brightness indicates longer traversals

Results

- The program rendered the atomic data in an aesthetically-pleasing and efficient manner.
- The program can easily be used to render any kind of scalar atomic data.
- Future work includes adding shadows to the particles, and porting the rendering method over to using 2D point sprite “pseudo-spheres”.

References

- "Rendering Pipeline Overview." *Opengl.org*. N.p., 10 May 2015. Web. 29 July 2015.
- Sigg, Christian, et. al. “GPU-Based Ray-Casting of Quadratic Surfaces”. *Eurographics Symposium on Point-Based Graphics, Boston, Massachusetts, July 29-30, 2006.* Ed. M. Botsch, B. Chen. 2006.

Acknowledgements

Thank you to the the CCT coordinators for organizing this project, the DMC for hosting this REU, and to Dr. Karki and his lab for mentoring throughout this project, as well as the NSF for funding this research experience.

Questions?

CREDITS

- Presentation template by SlidesCarnival
- Photographs by Unsplash