Development of a 3D Graphics Engine

Kevin Kassing – Computer Systems Lab 2006-07 – Period 5

Abstract

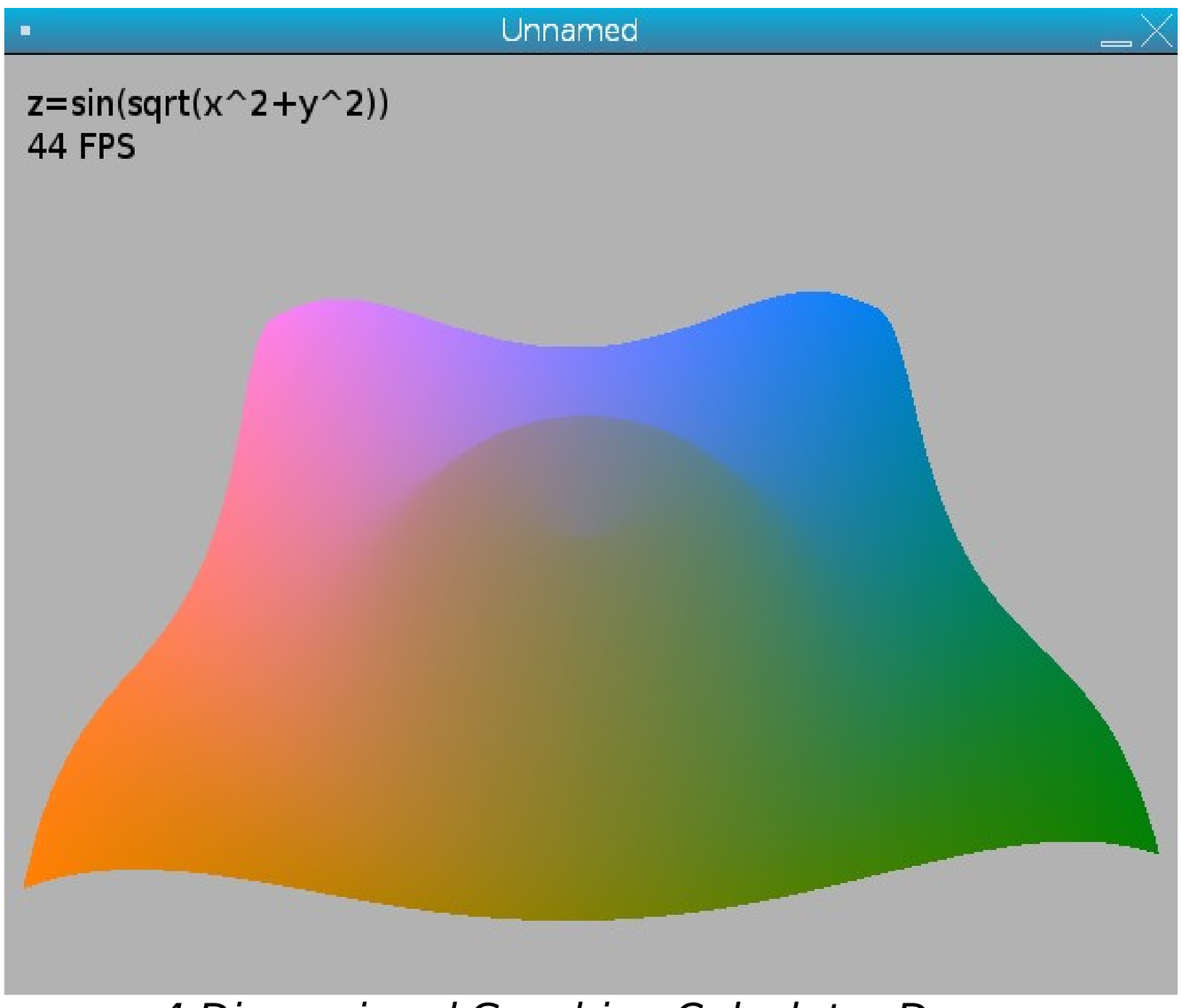
Visualization is an extremely valuable tool in problem solving, especially as problems become more complex. The goal of this project is to create an engine to facilitate three-dimensional visualizations of problems without requiring knowledge of OpenGL. Additionally, the engine should be able to perform at a high level by giving the developer a deeper level of control. The engine architecture is designed to let the developer have as much control over the IO and rendering processes as he or she wants. Game developers will also benefit from mesh modification functions and rendering optimizations.

Procedures and Methods

The engine is designed with speed as the primary goal, and platform independence as a secondary goal. The input routines may derive from different libraries, such as SDL or GLUT, but have a similar API.

The engine consists of a basic engine framework, and accessory methods useful for 3D graphics, such as collision detection methods, and conversions between different representations of rotations. There are texture loading methods and support for GPU shaders built into the engine. Abstracted from the main engine are building blocks not specific to 3D graphics, including vectors and quaternions.

The mesh functions are also exported to a separate library, and will be implemented in the engine via wrapper structures. By separating the mesh routines to an external library, the developer gets the choice of whether or not to use the main engine framework.



4 Dimensional Graphing Calculator Demo

Rendering a MD2 format mesh

Expected Results

I expect to be able to implement a data structure which can represent blended textures, shaders, and cube reflection maps. I also expect to be able to load several different mesh formats and convert them to a single generalized mesh format optimized for rendering and Level of Detail modification.