J3dge: An Educational Game Engine

Minghui Liu

May 12, 2017

Abstract

J3dge is an educational 3D game engine written in the Java programming language. J3dge is created due to the lack of good learning materials on game engines, and is built to be a learning material for students interested in the topic. The goal of J3dge is to help students learn the internals of a game engine by studying its source code. While a lot of other open source game engines exist, they are not ideal for beginners to learn because of their complexity and code readability issue. J3dge is designed to be simple and readable: It contains only a core set of features and its source code is easy to read and self-explanatory. The Java programming language was chosen to be J3dge's implementation language because of its expressiveness and its popularity among computer science students. J3dge supports multiple windows, keyboard and mouse input, mesh loading, texture, material, realistic lighting, resource management and various other features essential to a game engine. A game demo was created to demonstrate the features of J3dge and to prove that it can be used for game production. The source code is organized logically into 30 classes controlled to stay below 4000 lines and can be read through quickly by any students with some graphics programming experience. J3dge is open source and free to everyone.

1 Introduction

A game engine is a software framework designed to create games. Game engines help developers reused code that are common to multiple games of the same genre and let developers focus on creating unique game resources, which typically include textures, materials, sound clips and animations.

2 Purpose

Many students are interested in learning about the implementation of game engines. A good way of learning is to read the source code. However there is a lack of good learning materials on the topic. While there are many open source game engines for production use, they are not good materials for the following reasons.

- In order to produce state of the art games and compete with competitors, production games engines usually pack a lot of features. Therefore they are very complex and hard to start with.
- Most production game engines are written in C++, a language most students are not familiar with and has a very steep learning curve
- Source code of production engines are highly optimized for performance and cryptic

The purpose of this project is to create a game engine that students can read and use to learn about the implementation of game engines. We want J3dge to be both a working game engine useful for creating games and also be a learning material for students.

3 Design Goal

J3dge is designed to fill the blank of an educational game engine. The two goals of J3dge is therefore:

- simplicity
- readability

To keep J3dge simple, only a set of most essential features were implemented. To keep the code readable, the Java programming language was chosen to be the implementation language of J3dge because of it expressive, popularity among college computer science most students and object oriented programming paradigm. Additionaly, readable code was favored over performant code in the development of J3dge to avoid code that need external documentation. The goal is for students to know what a function is doing by reading the source code and comments only.

4 Result

We achieved the goal of building a readable and working game engine. J3dge has just under 4,000 lines of code, organized into 30 classes. Each class contains only a few hundred lines of code and is very easy to go through. J3dge is also useful for creating games, which we demonstrated by building a game demo with it.

5 Technologies

J3dge is create using Java, OpenGL, GLFW and LWJGL.

6 System Design

J3dge classes can be divided into six major modules. The Window management module contains a single class, Window, which manages the creation and destruction of windows and their properties. The Math module contains classes for vector, matrix and Quaternion calculations. It also records of time and game constants. The graphics system, also known as the rendering system, contains all classes related to graphics and rendering. The graphics system depends heavily on the math module. The resource manager module contains classes that model resources and a loader that helps loading them. The Physics module contains collision detection code. Finally, the game logic module contains the game interface and other classes related to game play. Figure 1 shows the design diagram of J3dge.

7 Features

J3dge has a lot of features. Here is an incomplete list of features:

- **Window management**
  J3dge supports creation and destruction of multiple windows.
- **Player Input**
  Players need to interact with games. J3dge support keyboard, mouse click and scroll wheel input in way of callback functions.
- **Model loading**
  J3dge supports Wavefront mode files, aka obj files. You can create models in modeling softwares like Blender and import it into your game.
- **Texture and material**
  J3dge supports BMP and PNG format images as textures. Materials are defined using a data structure.
- **Lighting**
  J3dge uses Phong lighting model and supports three different types of light sources, directional light, point light, and spot light.
- **Level generation**
  J3dge supports level generation using Bitmaps. Developers can draw level floor plans on a bitmap image and J3dge will generate the map for them.

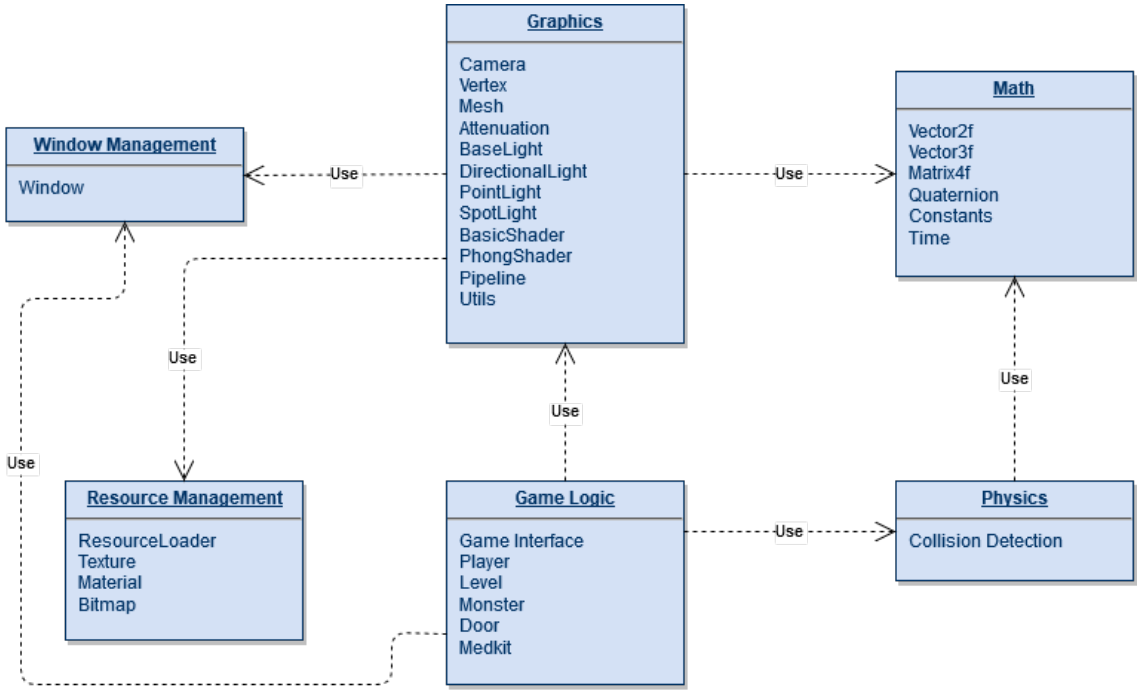

Figure 1: System Design Diagram of J3dge.

- **Game Resource management**
  J3dge provides ResourceLoader class with multiple methods for loading and parsing texture and mesh model files.

8 Game Demos

- **Rotating Triangle**
  Rotating triangle shows how to draw basic geometries on screen and how to use transformation methods in the math classes.

- **Textured Cube**
  TexturedCube demo shows how to use texture in your game.

- **Camera Demo**
  CameraDemo shows how to setup a camera and how to use keyboard, mouse to change camera position and view angle. Also shows how to change FOV and zoom using scroll wheel.

- **Phong Lighting Demo**
  PhongLighting shows directional light source, the phong lighting model and textures. You can decompose light and see all three component and how they add together to produce realistic lighting.

- **Light Source Demo**
  LightSource demo shows point light and spot light sources.

- **Wolfenstein 3D Clone Demo**
  This demo is a remake of the original Wolfenstein3D, the first 3D FPS game, made by idSoft. It demonstrates custom object creation, level loading, enemy generation and collision detection.

9 Timeline

Date	Progress
Fall Semester	
9/30/2016	Window and input
10/07/2016	Vectors, matrices and other core mathematical data structures and their operations
10/14/16	Rendering mechanics and mesh rendering
10/21/16	GLSL and basic shader
10/28/16	Uniforms, translation, rotation and scaling
11/04/16	Vertex reuse and OBJ model loading
11/11/16	Perspective projection
11/18/16	Quaternion rotation and Camera
11/25/16	Texture
12/05/16	Ambient lighting
12/07/16	Directional lighting
Spring Semester	
2/1/17	Specular Reflection
2/8/17	Point Lighting
2/16/17	Spot light
2/23/17	Tweak and optimize lighting
3/2/17	Organize engine architecture
3/16/17	Resource loading
3/23/17	Resource reuse and auto update
3/30/17	Level generation and level loading
4/6/17	Collision, Hit detection
4/20	Demos

10 Conclusion

In this project I build an Educational game engine for students. J3dge is open source and free to anyone. Everyone is encourage to try j3dge and use it for game developement and learning purposes.

11 Final Note

It was a great experience. Build something working and layer on top of it. Make demos along the way. Small progress, accumulate them and you get something larger.