Relatório Projeto Final

Universidade Estadual de Campinas

Gabriel Massaki Wakano Bezerra
E
Emely Pujólli da Silva

CAMPINAS
2016

Sumário

Resumo 3
Introdução 3
Desenvolvimento 3
Conclusão 3

Referências **3**

Resumo

Neste relatório, tratamos do projeto final da disciplina de Computação Gráfica I ministrada no primeiro semestre de 2016. Para o projeto, foi requisitado a construção de uma imagem realística utilizando as ferramentas aprendidas em sala e a Interface de Programação de Aplicativos (API) gráfica OpenGL [3].

Introdução

A necessidade de criar uma imagem no computador compatível com a realidade, impulsiona a pesquisa em computação gráfica. A síntese de uma imagem é o processo de criação de imagens por computador utilizando a descrição dos elementos que compõem a cena tridimensional desejada. Para isso, levamos em conta três elementos: objetos, fontes de luz e câmera. Os objetos são os elementos que desejamos visualizar. Os objetos são descritos através de atributos geométricos e ópticos. A geometria descrevendo a forma do objeto e o atributo óptico, o material que o compõem. Já as fontes de luz, são os elementos que emitem luz. Os modelos de iluminação e tonalização são responsáveis pela definição da cor de cada ponto dos objetos visíveis de uma determinada imagem gerada. Os modelos de iluminação, determinam a cor de um ponto na cena levando em conta as propriedades ópticas da superfície do objeto, as características da luz incidente e a posição do observador. Já os modelos de tonalização determinam onde e como estes modelos de iluminação devem ser aplicados [1]. Por fim, a câmera é o elemento responsável pela projeção da representação tridimensional em um plano imagem bidimensional [1].

É conhecido que o princípio de programação é a programação orientada a eventos. Este é o modelo de programaçãoo utilizado para implementaçãoo de aplicativos gráficos interativos. Para construir a imagem utilizaremos a biblioteca de *widgets* FREEGLUT (*open source implementation of GLUT - OpenGL Utility Toolkit*) para a implementação de interfaces de aplicativos gráficos 3D [4]. Além de ser independente do sistema de janelas, o GLUT provê um *widget* de tela capaz de se comunicar a interface de programação de aplicativos OpenGL e utiliza o modelo de callback para registrar os tratadores de eventos. Também utilizamos a biblioteca GLEW (*OpenGL Extension Wrangler Library*), uma multi-plataforma de fonte aberta C/C++ de extensão de carregamento. GLEW fornece mecanismos eficientes de tempo de execução para determinar quais extensões de OpenGL são suportadas na plataforma desejada. OpenGL é uma API em constante evolução e aprimoramento. Novas versões das especificações OpenGL são regularmente liberada pelo Khronos Group, cada um dos quais se estende a API para suportar vários novos recursos.

Para imagem realística, decidimos construir um terreno desnivelado com algumas plantas e flores.

Desenvolvimento

Para gerar a imagem desejada, foi realizada a triangularização de todo um terreno, para tal, foi criada uma grade de quadrados que foi gerada de maneira recursiva até que os quadrados se tornassem muito pequenos, mais detalhes da implementação estão disponíveis no arquivo disponibilizado juntamente ao relatório. Para construir as plantas foi utilizado o *L-System* [2] e todos os modelos geométricos usados no trabalho foram gerados manualmente, ou seja, as coordenadas dos vértices dos modelos foram definidas após os desenhos dos mesmos terem sido feitos em papel quadriculado. A posição dos vértices dos objetos segue uma função pseudo-aleatória, ou seja, dado execuções distintas, esperasse que tanto o terreno gerado, quanto as posições das plantas e suas formas sejam diferentes.

Foi calculado os vetores normais de todos os modelos geométricos para o *shading*. Utilizamos o modelo de iluminação de Phong, que pode ser observado na Figura 1, onde está mais escura a parte inferior do terreno.

É possível mover a imagem utilizando o mouse para poder visualizar melhor a iluminação e as profundidades do terreno.

O arquivo contendo todas as partes do código estará disponível online no site da disciplina (`http://www.dca.fee.unicamp.br/cursos/IA725/1s2016/`).

Conclusão

Para compilar o programa, no ambiente linux, basta executar o *Makefile* tendo o OpenGl instalado. Podemos visualizar a imagem final na Figura 1.

Ainda, durante a apresentação do projeto, para verificação de que os vetores normais à superfície estavam em diferentes direções, indicamos as normais com cores, obtendo assim a imagem exposta na Figura 2. Vale ressaltar que devido a aleatoriedade, cada vez que executamos o programa, as normais possuirão valores diferentes.

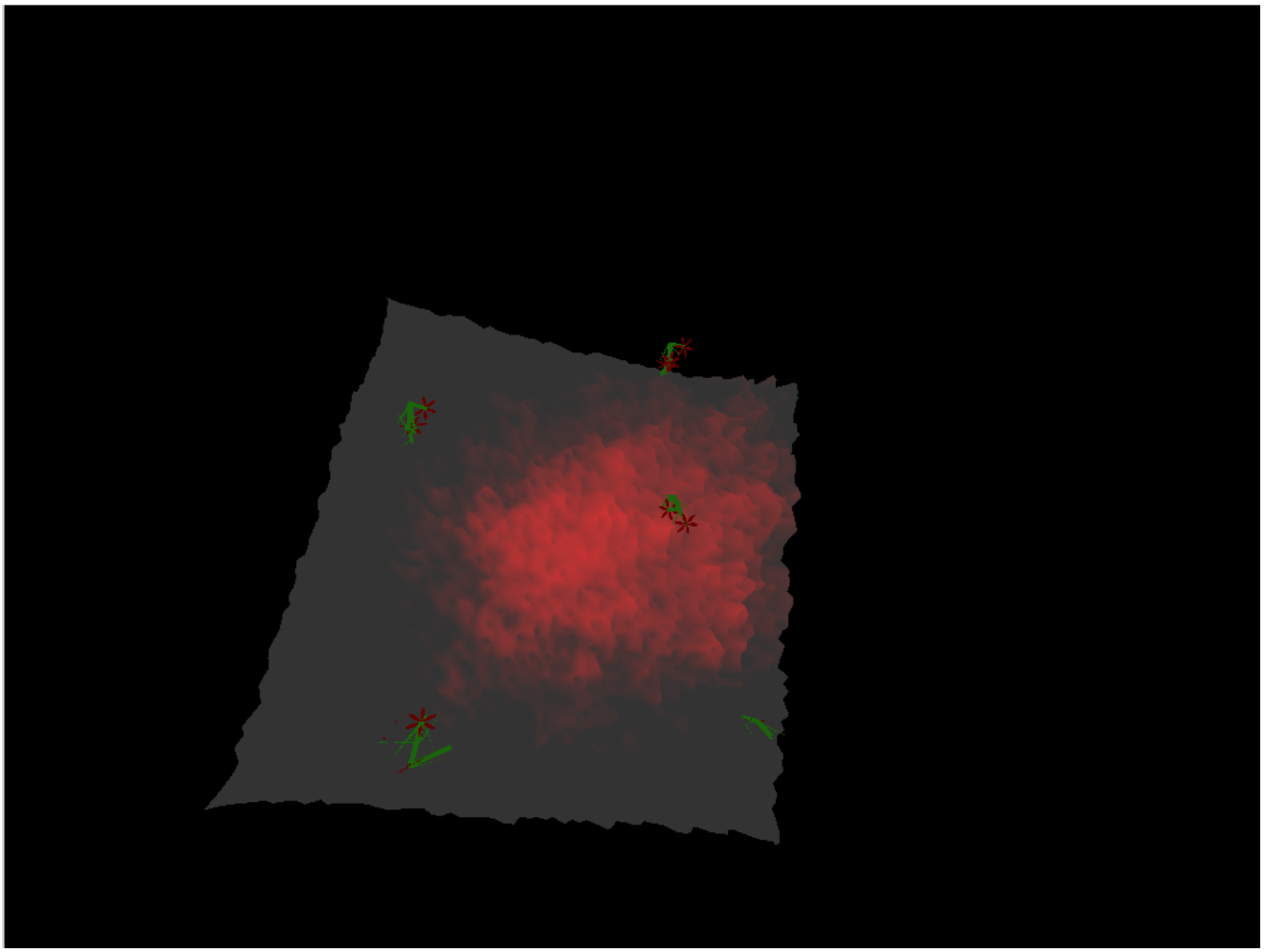

Figura 1: Imagem final

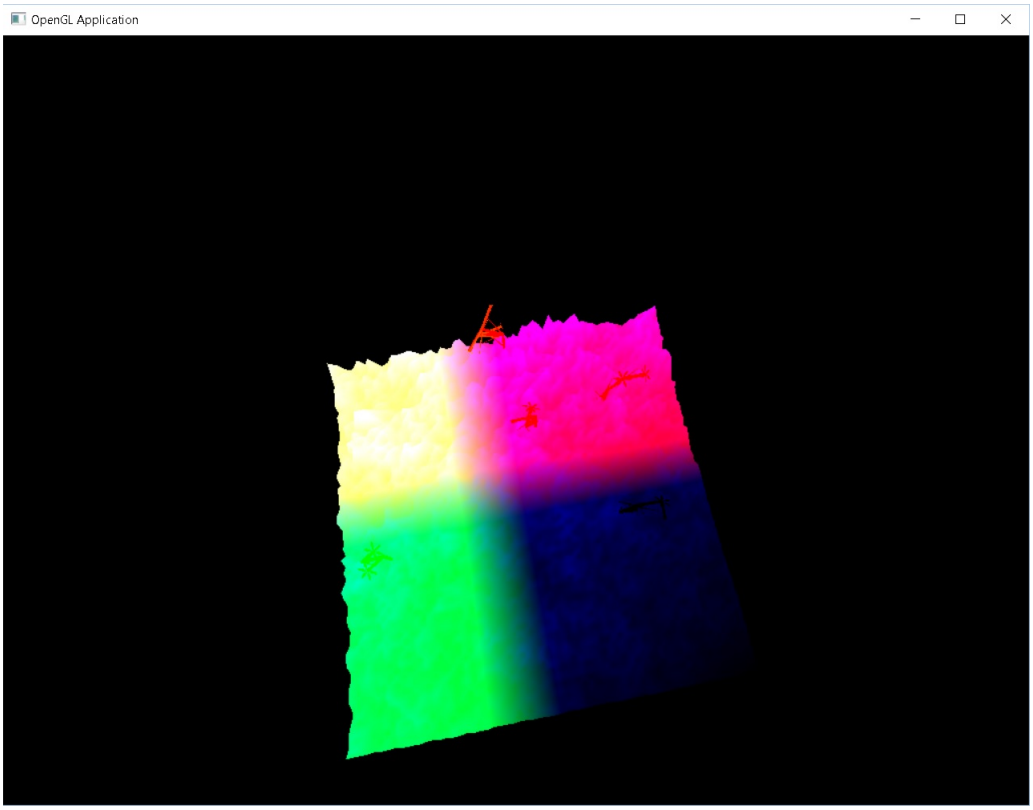

Figura 2: Imagem com normais coloridas

Referências

[1] Paulo Augusto Dal Fabbro. *Imagens ilustrativas de técnicas de sÃntese de imagem por computador*. `http://www.dca.fee.unicamp.br/~martino/iniciacao/pauloau/iniciacao.html`. 1998.

[2] *L-System*. `https://en.wikipedia.org/wiki/L-system`. Accessed: 2016-07-10.

[3] Dave Shreiner et al. *OpenGL Programming Guide: The Offcial Guide to Learning OpenGL (red book)*. Addison-Wesley Pub Co, 2005.

[4] Shin-Ting Wu. *Sistemas de informações Gráficas*. 2009.