Shaders - Básico

Abordagem de "*shaders*" em *OpenGL 3.2* e *GLSL 1.50*

Filipi Nascimento Silva

Aula para a disciplina de Computação Gráfica da pós graduação do Instituto de Ciências Matemáticas e de Computação

Tempo de aula: 1 hora e 30 minutos.

1.Objetivos:

- Apresentar ao aluno os conceitos básicos de "Shaders": "Fragment", "Vertex" e "Geometry Shader".
- Introduzir e motivar o aluno ao uso de "pipeline" programável para o desenvolvimento de aplicações gráficas.
- Fornecer condições para que o aluno possa iniciar o desenvolvimento de aplicações gráficas usando "shaders" em *OpenGL 3.2*.

2.Resultados:

- O aluno deve entender os principais usos de cada tipo de "shader" assim como suas participações correspondentes no "pipeline" gráfico.
- O aluno também deverá preferenciar o uso de "pipeline" programável em seus futuros projetos.
- O aluno deverá estar apto a compreender e desenvolver programas "shaders" simples escritos em *GLSL* para *OpenGL 3.2*.
- Ao término da aula o aluno deve estar motivado para buscar mais informações sobre o assunto, principalmente sobre o uso de "shaders" em sua plataforma/linguagem preferida de desenvolvimento.

3.Metodologia:

- O conteúdo é apresentado por meio de 51 slides.
- Detalhes extras e dúvidas são discutidas verbalmente e no quadro negro.
- O método de programação de “shaders” em *GLSL* é apresentado através de exemplos comentados seguidos pela demonstração do resultado gráfico final.

4.Ementa:

- **História e Introdução:** Contexto histórico do termo “shaders”, evolução de shaders nas *GPUs*, principais linguagens de programação de “shaders”: *DirectX*, *CG*, *GLSL*.

- **Pipeline Fixo:** Principais características do “pipeline” gráfico fixo e do “pipeline” programável. Principais vantagens do “pipeline” programável sobre o fixo.

- “**Shaders” em *OpenGL*:** Evolução dos “shaders” em *OpenGL*. Comparação entre o core “profile” e o “compatibility profile” do *OpenGL 3.2*. Requisitos computacionais para se desenvolver com *OpenGL 3.2*.

- **Linguagem *GLSL*:** Principais características da linguagem *GLSL*. Diversos exemplos de diferentes técnicas implementadas em *GLSL*: Iluminação, iluminação phong, texturização e modo “wireframe” implementado por “geometry shader”.

- **Uso de GLSL em OpenGL:** Apresentação do código *OpenGL* necessário para preparar e fornecer dados a um programa *GLSL*.

5.Bibliografia:

- **Exemplos em OpenGL 3.2 e 4.0 incluindo shaders**
  http://nopper.tv/opengl.html

- **Especificação OpenGL 3.2 Core Profile**
  http://www.opengl.org/registry/doc/glspec32.core.20090803.pdf

- **OpenGL Wiki**
  http://www.opengl.org/wiki/

- **Curso de Shaders da Oregon University**
  http://web.engr.oregonstate.edu/~mjb/cs519/

- **Tutoriais Lighthouse 3D**
  http://www.lighthouse3d.com/tutorials/

- David Wolff, **OpenGL 4.0 Shading Language Cookbook**
  http://www.packtpub.com/opengl-4-0-shading-language-cookbook/book?utm_source=rk&utm_campaign=opengl4-abr1&utm_medium=0811

- **Migrating to OpenGL 3.2 Core Profile**
  http://athile.net/library/blog/?p=582

- **Pacote de exemplos de shaders diversos**
  http://www.g-truc.net/project-0026.html

- **WebGL sandboxes**
  http://mrdoob.com/139/GLSL_Sandbox

- **Shader Editor escrito em WebGL**
  http://www.kickjs.org/example/shader_editor/shader_editor.html