IA727 – Computação Gráfica II – Segundo Semestre de 2016
Profa. Wu Shin-Ting

Aulas: 1 e 2

Objetivos:

- Fluxo de renderização
- Implementação deste fluxo: OpenGL
- Sistema de janelas
- Estrutura de um aplicativo de OpenGL (códigos para CPU e para GPU)
- Códigos executáveis a partir dos códigos-fonte em C/C++

Exercícios de Revisão de IA725:

1) O que você entende por *rendering* (renderização)? E o que você entende por um *pipeline* (uma linha de produção ou um fluxo)?

2) Desenhe em diagrama de blocos os estágios de um fluxo de renderização.

3) OpenGL/DirectX é uma API (*Application Programming Interface*) das modernas unidades de processamento gráfico programáveis (GPUs). Sendo OpenGL (*Open Graphics Library*) uma biblioteca gráfica aberta, nós a adotaremos nesta disciplina. Faça um paralelo entre o fluxo de renderização descrito no capítulo 2 de *Real Time Rendering* com a arquitetura de OpenGL apresentada no capítulo 1 de *Red Book*/capítulo 3 de *Real Time Rendering*.

4) Dependendo do sistema operacional, as extensões da API OpenGL não são reconhecidas. Por exemplo, Windows 7 só reconhece API OpenGL 1.1. GLEW (*OpenGL Extension Wrangler Library*) é uma biblioteca que suporta a busca e a carga de todas as extensões disponíveis na GPU em tempo de execução. Ela é multiplataforma. É necessário o uso da biblioteca GLEW n ambiente Linux?

5) O que você entende por estágios de funções fixas (*fixed-function stage*), de funções programáveis (*programmable functions* ou *shaders*) e de funções altamente configuráveis (*configurable functions*) no contexto da arquitetura de OpenGL? Identifique-os no fluxo de renderização.

6) O paralelismo suportado pela GPU (*Graphics Processing Unit*) é SIMD (*Single Instruction, Multiple Data*) e as operações em ponto-fluatuante são operações nativas. Por quê estas duas características torna GPU extremamente favorável para processamento gráfico em comparação com a CPU (*Central Processing Unit*)?

7) A saída de um fluxo de renderização é uma imagem digital 2D. Para exibí-la numa tela do computador, é necessário alocar os recursos gerenciados por um sistema de janelas instalado, como regiões de *pixels* na tela do monitor. **OpenGL não gerencia os recursos dos periféricos de entrada e de saída gráficos**. Mark J. Kilgard desenvolveu uma biblioteca minimalista, conhecida como GLUT, não só para fazer tal alocação como também para programar as rotinas

de serviços de alguns eventos de entrada gerenciados pelo sistema de janelas. Quais são as funções disponíveis na GLUT através das quais permite-se inicializar e manipular a área de exibição de uma imagem?

8) GLSL (*OpenGL Shading Language*) é uma linguagem criada pela OpenGL ARB (*Architecture Review Board*) para modificar os estágios programáveis de OpenGL. Antes de iniciar a "linha de produção" de imagens na placa de vídeo (GPU), os códigos dos estágios programáveis precisam ser compilados, ligados e carregados na GPU, como também todos os dados ("insumos") precisam ser transferidos para GPU. Quais são as funções utilizadas no *setup* do programa detalhado aqui?

9) Como é a estrutura básica de um código de renderização desenvolvido em cima da OpenGL? Como as instruções executadas em CPU e as em GPU se complementam para gerar uma imagem? Justifique as suas respostas com base no programa do item anterior.

10) Instale o ambiente de desenvolvimento básico no seu computador pessoal. Verifique se está operando corretamente.