Computação Visual — 2020/2021

Propostas para o 1º Trabalho — WebGL

***** Entrega intermédia: 30 de novembro de 2020 – 1 página: objetivos, organização e duas imagens *****

***** Data limite para a entrega: 14 de dezembro de 2020 *****

***** Apresentações: 21 de dezembro de 2020 *****

Cada **grupo de dois alunos** poderá selecionar um trabalho de entre as várias propostas seguintes e **comunicar a sua escolha por email** ao docente da sua turma prática.

É encorajada, em **alternativa**, a apresentação de **propostas de trabalho autónomas** de desenvolvimento de **aplicações 3D** que devem demonstrar a **modelação** de objetos simples (pelo menos 3 **modelos distintos**), a utilização do modelo de **iluminação** e a aplicação de texturas num cenário à vossa escolha, com **interatividade / controlo** por parte do utilizador (**teclado e rato**).

Além do código, é pedida a entrega de um **relatório sucinto (**6 a 8 páginas no formato da revista do DETI**),** descrevendo os aspetos principais quer da aplicação desenvolvida quer do trabalho efectuado.

Será também necessário fazer uma **apresentação do trabalho** e uma demonstração da aplicação desenvolvida.

O trabalho entregue deverá ser original. Além dos ficheiros distribuídos nas aulas práticas, poderão ser utilizados (pontualmente) algum código ou bibliotecas existentes, desde que esse facto seja claramente referenciado no código e no relatório.

Trabalhos que utilizem grande volume de código não original, sem o referenciar explicitamente, serão anulados.

O trabalho deve ser desenvolvido em **WebGL**, biblioteca utilizada nas aulas práticas. A utilização de ferramentas de mais alto nível (three.js, por exemplo) não é permitida.

No caso de trabalhos em grupo, o **relatório** deve **identificar a contribuição**, e **estimar o esforço relativo** (em percentagem), de cada um dos membros do grupo para o trabalho global.

Propostas de trabalhos:

1. **Gerador de terreno 3D:** gerador de terreno 3D a partir de imagens de satélite e de texturas que codificam a elevação do terreno correspondente. Estes modelos podem ser gerados offline utilizando CPU ou gerados dinamicamente em GLSL por deformação dos vértices de uma geometria base. Os dados para este projeto poderão ser acedidos a partir da plataforma

Mapbox que disponibiliza de forma gratuita acesso à API (https://docs.mapbox.com/api/).

2. **Visualizador de nuvens de pontos usando árvores octais (octrees):** em ferramentas de visualização 3D é comum o uso de nuvens de pontos para representar cenas 3D. Os dados são tipicamente recolhidos de sensores laser ou camaras de profundidade, e são usados em Engenharia e Robótica. O objetivo é desenvolver um sistema para carregamento progressivo de nuvens de pontos de grande dimensão, usando árvores octais.

3. **Visualizador 3D com ferramentas de medição:** desenvolvimento de uma ferramenta de visualização 3D para malhas triangulares com suporte a ferramentas de medição. A aplicação deverá ser capaz de carregar modelos .obj a partir de ficheiro; visualizar e navegar sobre os modelos e permitir obter algumas mediadas: distâncias entre pontos, amplitudes de ângulos, áreas de superfícies.

4. **Edição interativa de modelos 3D:** aplicação que permita ler um modelo 3D (nuvem de pontos ou malha de triângulos) num formato pré-estabelecido, alterar / remover pontos ou triângulos e guardar o resultado em ficheiro. Devem ser implementadas operações adicionais (p.ex., triangulação de uma nuvem de pontos) e permitida a seleção / edição de múltiplos pontos ou triângulos.

5. **Fractais 3D:** gerar e visualizar diferentes fractais 3D representados como malhas de triângulos. Deve ser possível seleccionar o tipo de fractal e especificar o nível de subdivisão.

6. **Visualização de Superfícies:** visualizar superfícies (p.ex., quádricas) representadas como malhas de triângulos. Deve ser possível estabelecer a equação / os parâmetros que definem cada tipo de superfície usando menus e a consola. Ou então visualizar superfícies cuja representação é lida a partir de ficheiro — parte do trabalho consiste em construir vários ficheiros representando diferentes tipos de superfícies.

7. ***Shaders*:** aplicação que, com base em modelos simples, ilustre os efeitos que é possível obter com os *shaders*, explorando **funcionalidades não abordadas nas aulas**. Deve ser possível ler diferentes modelos 3D e especificar os tipos de efeitos que se pretende visualizar.

8. Demonstração do funcionamento do **Mapeamento de Texturas** em WebGL, sendo possível utilizar texturas de diferentes tipos, bem como alterar parâmetros do mapeamento, **explorando funcionalidades não abordadas nas aulas**. Um possível cenário consiste na apresentação de um mesmo modelo em dois *viewports* distintos, mas sendo mapeadas texturas distintas, de modo a permitir a comparação visual dos resultados obtidos.

9. Aplicação para representação do **volume de visualização** (*view volume)*, modificação das suas características (p.ex., posição do observador) e visualização da resultante imagem final. O programa deve criar uma cena em 3D com objetos simples. E mostrar ao utilizador duas janelas: uma com a vista 2D resultante do volume de visualização definido e, outra, apresentando o volume de visualização na cena 3D. Alterações quer da posição do observador, quer das características de visualização (p.ex., tipo de projecção), devem reflectir-se nas duas janelas.

10. **Simulador de um Labirinto 3D**: é carregado e construído um labirinto (p.ex., definido como uma matriz 2D ou em formato micro-rato) e são controlados os movimentos (livres) da câmara / do utilizador, sendo detetadas as colisões com as paredes do labirinto.

11. **Animação simples** que exemplifique de uma forma dinâmica o funcionamento do algoritmo de **Ray-Tracing**. Deve ser criada uma cena simples, com pelo menos duas esferas, um plano (chão) e duas fontes de luz configuráveis e de cores diferentes, e mostrado o plano da imagem e a posição do observador (um cone). O utilizador activa a animação sendo mostrados os percursos dos raios primários e secundários.

12. **Animador de Cenas 3D**: a aplicação deve ler de ficheiro e visualizar objetos distintos e permitir configurar a sua posição e orientação ao longo do tempo, bem como a duração de cada movimento, de modo a gerar uma animação – por exemplo, de um sistema planetário simplificado.

13. **Jogo das Damas:** desenvolver uma aplicação 3D intuitiva que permita jogar ao jogo das damas.

14. **O Problema das Damas ("The N-Queens Problem"):** desenvolver uma aplicação 3D que ilustre o funcionamento do algoritmo recursivo, com *backtracking*, para este problema tradicional de colocação de damas no tabuleiro de xadrez, sem que se ataquem mutuamente.

15. **O Percurso do Cavalo ("The Knight's Tour"):** desenvolver uma aplicação 3D que ilustre o funcionamento do algoritmo recursivo, com *backtracking*, para este problema – começando em qualquer casa de um tabuleiro de xadrez, percorrer as restantes casas, sem visitar duas vezes uma qualquer casa, e respeitando as regras de movimento do cavalo do xadrez.

J. Madeira / P. Dias
NOV 2019