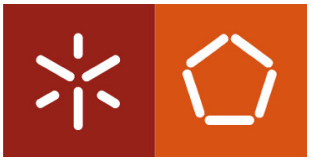
Universidade do Minho

Escola de Engenharia



Computação Gráfica

Licenciatura em Engenharia Informática

Normals and Texture Coordinates

Trabalho Prático - Fase 4 - Grupo 7

João Silva - [A89293]
Francisco Novo - [A89567]
João Vieira - [A93170]
Tiago Ribeiro - [A93203]

Junho, 2022

Conteúdo

Introdução

No âmbito da Unidade Curricular de Computação Gráfica, foi desenvolvida a quarta fase do trabalho prático proposto pelos docentes. O principal objetivo desta fase é que a aplicação gerador seja capaz de calcular as normais e coordenadas de textura para cada vértice das primitivas criadas. Com isto, o motor seria capaz de ler os ficheiros dos modelos e ativar as funcionalidades de iluminação e de texturização das primitivas.

Tal como nas fases anteriores, foi necessário recorrer à alteração do código desenvolvido previamente. A estrutura do ficheiro XML foi alterada, o que faz com que a leitura do ficheiro sofra alterações. Agora os modelos podem receber texturas e várias informações acerca da sua cor e é possível a definição de fontes de luz.

Este relatório encontra-se dividido em três partes: a primeira e a segunda que abordam as alterações efetuadas no gerador e no motor, respetivamente, e uma terceira e última que se refere ao trabalho realizado na *demo scene* do sistema solar.

Alterações no gerador

Nesta fase, a aplicação gerador tem de ser capaz de calcular as normais e coordenadas de textura de cada vértice de uma primitiva e realizar a escrita das mesmas num ficheiro.

Cálculo das normais

O cálculo das normais efetuado é realizado a partir da face em que o vértice está inserido. Estas serão necessárias para que seja possível representar a iluminação nas primitivas.

Plano

As normais de um plano são sempre as mesmas independentemente de onde o vértice esteja localizado no mesmo. Como o plano está voltado para o lado positivo do eixo dos yy's a normal de todos os vértices é $(0, 1, 0)$.

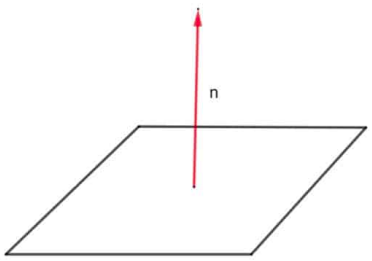


Figura 1: Normal de um plano

Cubo

As normais dos vértices de um cubo dependem da face onde o mesmo está inserido, sendo que para cada face a normal é sempre igual em todos os vértices.

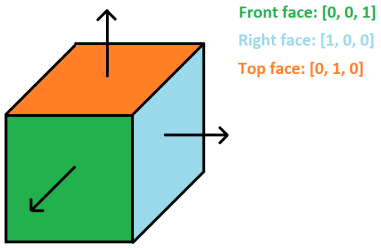


Figura 2: Normais num cubo

Esfera

As normais na esfera são calculadas a partir do vetor que vai do centro da esfera ao vértice em questão. Como a esfera está centrada na origem, nas coordenadas $(0, 0, 0)$, no cálculo de um vetor normal de um ponto P este irá ter as coordenadas iguais ao ponto, exceto que neste não irá ser efetuada a multiplicação pelo raio, permitindo assim que o vetor fique normalizado.

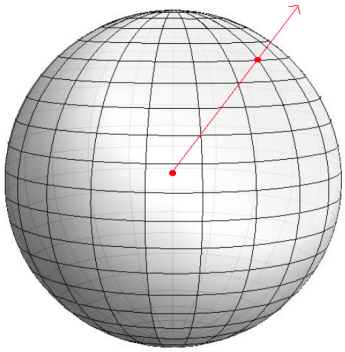


Figura 3: Normais numa esfera

Cone

As normais no cone estão separadas em duas partes: as da base e da face lateral. Na base as normais são iguais para todos os vértice, sendo estas $(0, -1, 0)$. Na superfície lateral as normais são dadas por $(sin(\alpha), cos(atan(H/R)), cos(\alpha))$.

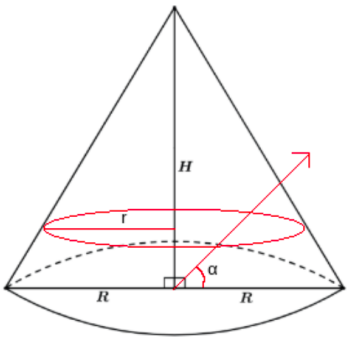


Figura 4: Normais num cone

Torus

As normais no torus são calculadas a partir do vetor que vai do centro da esfera da iteração atual até ao vértice que está a ser calculado. Estas podem ser obtidas por $(cos(\phi) * cos(\theta), cos(\phi) * sen(\theta), sen(\phi))$, sendo ϕ o ângulo da *slice* atual e θ o ângulo da *stack* atual.

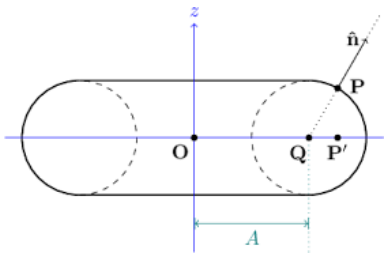


Figura 5: Normais num torus

Patch de Bezier

As normais no patch de Bezier são obtidas a partir do produto externo (cross product) de dois vetores: o vetor do alinhamento dos u's e o dos v's. Depois deste cálculo é necessário realizar a normalização do vetor resultado. Este processo é realizado para todos os pontos do patch.

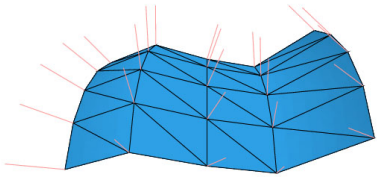


Figura 6: Normais num patch de Bezier

Cálculo das coordenadas de textura

O cálculo das coordenadas de textura é realizado para depois estas serem utilizadas na aplicação de texturas nas primitivas. As coordenadas de textura estão em 2D e a partir destas é possível saber que parte da textura utilizar num certo vértice da primitiva.

Plano

Para o cálculo das coordenadas de textura do plano foi divido 1 pelo número de divisões da primitiva, pois cada textura é 1 por 1, sabendo assim o tamanho de cada divisão na textura em questão. Assim sendo, o cálculo de todos os vértices na textura é idêntico ao cálculo de todos os vértices do plano original, mudando apenas o tamanho de cada aresta, consequentemente de cada divisão. Além disso, a textura é iniciada no ponto (0,0) e é incrementada em direção aos eixos positivos do X e do Y.

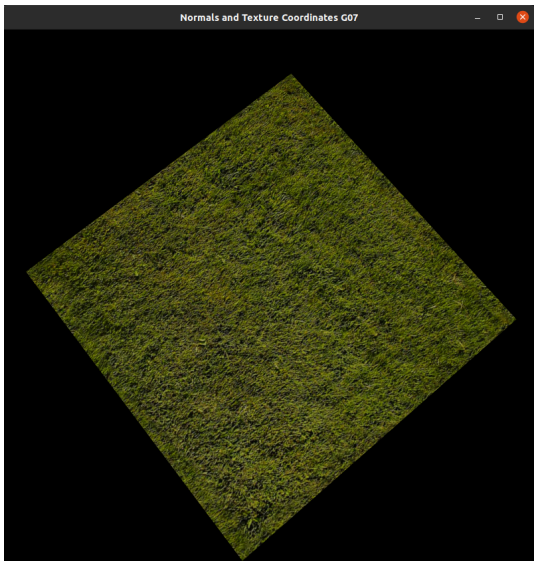


Figura 7: Textura aplicada num plano

Cubo

No cubo, os cálculos efetuados são semelhantes aos do plano, ou seja as coordenadas de textura são divididas de acordo com o número de divisões da primitiva. É feita a conversão, tal como no plano, para cada uma das faces do cubo. Primeiramente é feito o cálculo para as faces no eixo dos yy's, sendo seguido pelo cálculo nas faces no eixo dos xx's e por fim no eixo dos zz's.

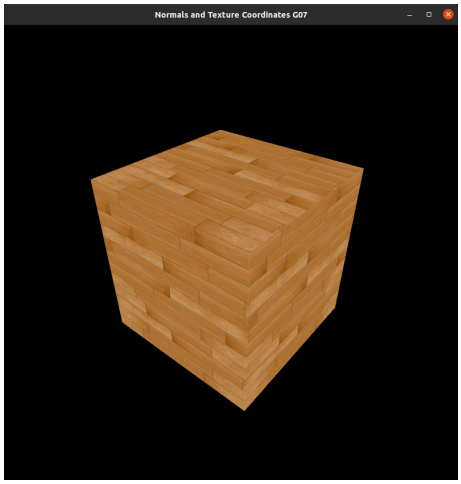


Figura 8: Textura aplicada num cubo

Esfera

As coordenadas de textura na esfera são divididas pelo número de slices e stacks da primitiva, permitindo assim mapear cada vértice para uma coordenada de textura. Ao iterar sobre as stacks a componente y das coordenadas de textura irá variar, enquanto que ao iterar sobre as slices é a componente x que se altera. Como a esfera está a ser desenhada em duas partes (parte superior e inferior) a componente y da coordenada de textura começa a 0.5 e vai aumentando ou diminuindo dependendo da parte que está a ser desenhada.

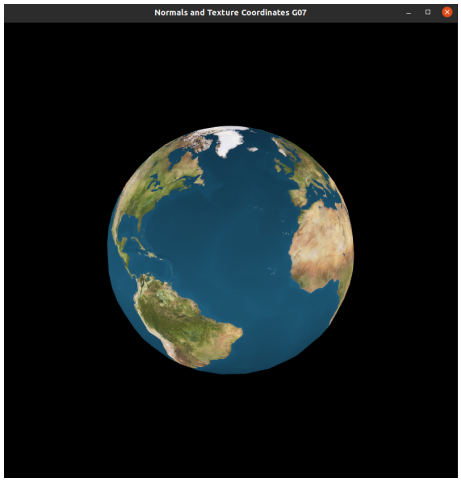


Figura 9: Textura aplicada numa esfera

Cone

O cálculo das coordenadas de textura no cone é dividido em duas parte: calcular as coordenadas de textura da base e as da superfície lateral.

Para a base são criados triângulos, sendo o número destes igual ao número de slices da primitiva. A base destes é igual à divisão de 1 pelo tamanho de cada slice, sendo a altura 1. Estes são colocados lado a lado nas coordenadas de textura.

Para a superfície lateral as coordenadas de textura são divididas pelo número de slices e stacks da primitiva e o processo para o cálculo das coordenadas de textura de cada vértice é semelhante ao processo realizado na esfera.

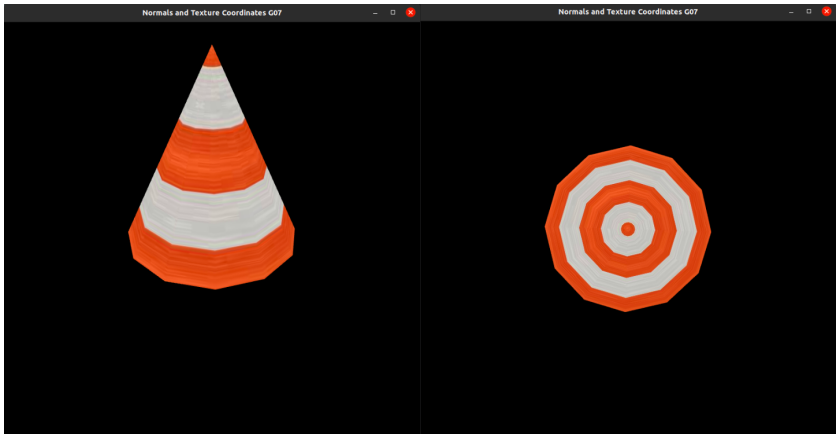


Figura 10: Textura aplicada num cone

Torus

Para calcular as coordenadas de textura no torus é feito um processo semelhante ao cálculo dos mesmos na esfera e na superfície lateral do cone em que as coordenadas de textura são divididas pelas slices e stacks e cada vértice é convertido de acordo com a slice e stack em que se encontra.

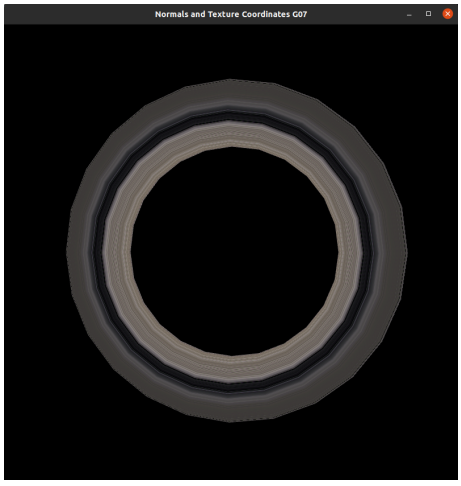


Figura 11: Textura aplicada num torus

Patch de Bezier

O cálculo das coordenadas de textura no patch de Bezier depende do nível de tecelagem do mesmo. É utilizada a grelha criada no cálculo dos pontos, sendo que esta já tem os pontos com as coordenadas no intervalo $[0, 1]$, logo a sua passagem para coordenadas de textura é fácil pois as divisões são iguais às da grelha de Bezier. Sendo assim, o tamanho das divisões é de $1/tecelagem$, sendo este multiplicado pelo número da iteração atual para obter as coordenadas de textura.

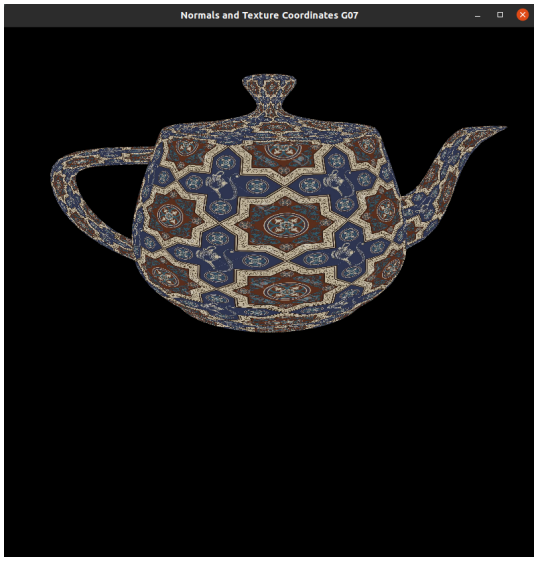


Figura 12: Textura aplicada e patches de Bezier

Alterações no motor

Relativamente às alterações feitas no motor foram adicionadas as funcionalidades de texturização e de iluminação. Foram habilitadas as opções disponibilizadas pelo OpenGL para permitir estes efeitos.

Nesta fase a aplicação motor tem de ser capaz de ler dois campos novos: o campo *lights* e *model* que sofreu alterações. Para guardar as informações relativas a cada modelo, foi criada um estrutura de dados nova chamada *model* que contém todos os parâmetros lidos do ficheiro XML e dos ficheiro 3d relativos ao modelo, sendo estes a textura a aplicar, as definições da cor, os vértices, as normais dos mesmos e as coordenadas de textura de cada um. Também contem dados relativamente às VBO's e dados que têm de ser guardados depois de carregadas as texturas para a GPU. Também foi criada uma estrutura de dados nova chamada *light* que tem como propósito armazenar informação das luzes criadas a partir do ficheiro XML. Foi criado um vetor global da estrutura *light*, sendo que cada elemento deste vetor representa uma luz que será representada no cenário.

Na leitura do XML por cada luz é criada uma estrutura *light* onde são armazenados os parâmetros recebidos do ficheiro XML e o ID correspondente à luz que é sempre incrementado à medida que são adicionadas novas luzes. Por cada modelo existente no ficheiro é criada uma estrutura *model* sendo esta devidamente preenchida, passando para a GPU todos os vetores e a textura, sendo este processo feito através da função *loadTexture*. Por fim a estrutura criada é colocada dentro do *group* global.

Na *renderScene* irá ser percorrido o vetor de luzes, que está definido globalmente, sendo representadas uma a uma. O modo de criação da luz depende do tipo desta, sendo que o tipo está indicado dentro da estrutura *light*. Depois de tratar das luzes ainda na *renderScene* é executada a função *drawFigures* onde são percorridas as estruturas *model* dentro do *group* e são desenhadas as suas primitivas com o auxílio da GPU.

Sistema solar

Para fazer a *demo scene* pedida nesta fase foi necessária a implementação de iluminação e texturas no nosso sistema solar. Deste modo, foi preciso modificar o ficheiro XML de modo a que o mesmo seja capaz de introduzir todas as texturas no sol, planetas, satélites e no cometa como também adicionar uma fonte de luz no sol.

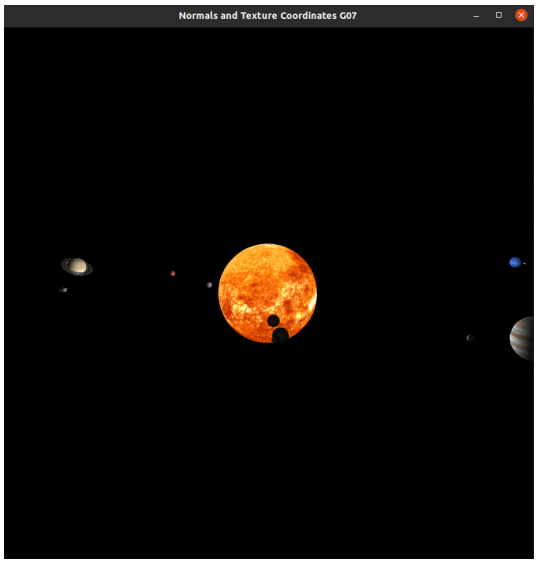


Figura 13: *Demo scene* do sistema solar

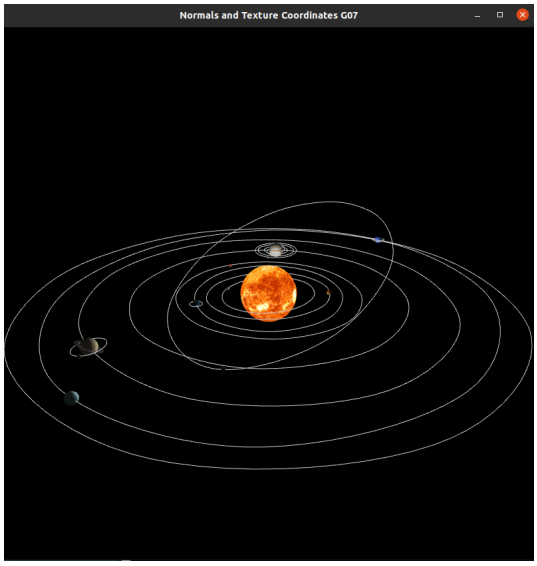


Figura 14: Sistema solar com as órbitas desenhadas

Conclusão

A quarta fase do trabalho prático permitiu aprofundar os conhecimentos adquiridos nas aulas relativamente a texturas e iluminação de primitivas.

As funcionalidades pedidas pelos docentes no enunciado foram maioritariamente cumpridas, faltando alguns aspectos que o grupo pretendia implementar porém não conseguiu alcançar, sendo estes a criação de uma câmara em terceira pessoa e uma *demo scene* mais detalhada.

Mesmo assim, o projeto é capaz de correr todos os testes requeridos pela equipa docente, e todas as suas funcionalidades estão bem planificadas e estruturadas, sendo que também foi desenvolvida uma *demo scene* que obedece ao pedido.