

Iluminação

Implementação do modelo de iluminação de Phong em WebGL (sombreamentos de Phong e de Gouraud)



Fig 1 - Cow (sombreamento de Phong)

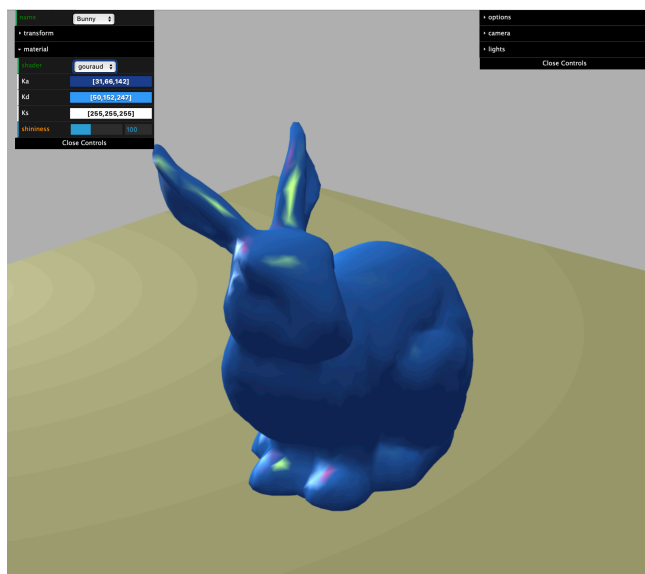


Fig. 2 - Bunny (sombreamento de Gouraud)

Objetivo

O objetivo do trabalho é o de implementar o modelo de iluminação de Phong, avaliado:

- a cada vértice (Gouraud shading)
- a cada fragmento (Phong shading).

A interface gráfica deverá utilizar a biblioteca [dat.gui](#). A cena é composta por 1 único objeto que pode ser modificado (transformações geométricas e material) e até mesmo ver alterada a sua geometria, substituindo-a por outra. No canto superior esquerdo estará a interface que permite manipular o objeto. Nessa interface poderá também optar por escolher iluminação por vértice ou por fragmento.

A Fig. 3 mostra um exemplo possível para a organização da componente da interface ligada ao objeto mostrado.

O utilizador poderá configurar algumas opções de visualização (back face culling, z-buffer e visualização da posição/orientação das luzes). Por outro lado, a aplicação também deverá permitir a modificação dos parâmetros da projeção perspetiva (fov,



Fig. 3 - Interface (objeto)

near e *far*), bem como o posicionamento e orientação da câmara. Deverá começar a realização do trabalho partindo da solução do exercício prático 27, o qual permite manipular a câmara. Se optar por não o fazer, espera-se que implemente na sua solução mecanismos idênticos.

A aplicação deverá também suportar a utilização e configuração de 3 luzes, tendo associadas as seguintes funcionalidades:

- mudar entre pontual ou direcional
- mudar as suas características (intensidades)
- mudar a sua posição e orientação.

A interface para o controlo da câmara, luzes e opções de visualização deverá estar no canto superior direito.

Funcionamento Geral

Controlo da câmara

A posição e orientação da câmara serão controladas pelo rato (ver exercício 27 das aulas práticas), mas mantendo *at* fixo na origem. O valor de *fovy* é controlado pelo gesto de scroll (*mouse scroll wheel* ou gesto no *trackpad*). Os valores de *near* e de *far* são controlados diretamente na interface gráfica.

Nota: Os valores de *eye*, *at* e *up*, embora devam estar presentes no interface, não precisam ser manipuláveis pelo utilizador através dos respetivos controladores `dat.gui`.

Opções de visualização

O utilizador deverá poder escolher se deseja aplicar ou não as seguintes técnicas de remoção de superfícies ocultas:

- backface culling
- z-buffer

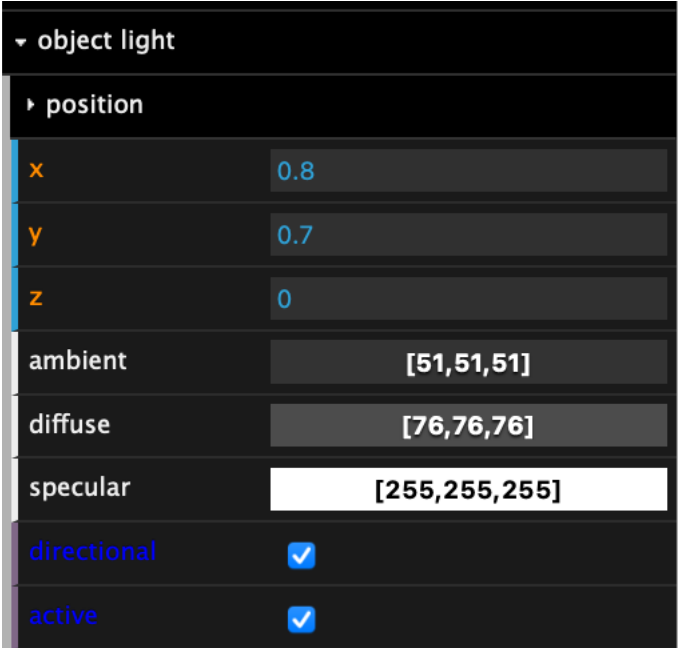
O utilizador também poderá ligar e desligar a visualização das posições das luzes através do desenho de pequenas esferas, de cor sólida (não sujeitas à iluminação) semelhante à cor da respetiva luz (componente difusa).

Especificação das luzes

Cada uma das três luzes deverá expor uma interface idêntica à ilustrada na figura XX. Cada luz deverá ser definida num referencial distinto, a saber:

- 1 luz no referencial do mundo
- 1 luz no referencial do objeto (sofrerá as mesmas transformações que forem aplicadas ao objeto)
- 1 luz no referencial da câmara (sofrerá as mesmas transformações que forem aplicadas à câmara)

Cada luz poderá estar ativada ou desativada num dado momento e o utilizador pode optar por tornar a luz direcional ou pontual. No caso das luzes direcionais é o valor de **L**



object light	
position	
x	0.8
y	0.7
z	0
ambient	[51,51,51]
diffuse	[76,76,76]
specular	[255,255,255]
directional	<input checked="" type="checkbox"/>
active	<input checked="" type="checkbox"/>

Fig. 4 - Interface (luz)

(modelo de Phong) que deverá ser mostrado nos campos aninhados em **position**. Cada luz possui uma intensidade específica para usar nos termos de reflexão ambiente (**Ia**), difusa (**Id**) e especular (**Is**). A interface mostrará valores entre 0 e 255 para cada componente de **Ia**, **Id** e **Is**, devendo os *shaders* receber valores normalizados no intervalo [0,1] para cada componente R,G,B das intensidades.

Quando a opção de visualização das fontes de luz está ligada, as esferas deverão ser desenhadas na localização dada pelas coordenadas mostradas no campo **position** (interpretadas no referencial da respetiva luz) da respetiva fonte de luz, quer para as luzes pontuais, quer para as luzes direcionais.

Não está prevista a mudança do referencial em que cada luz está definida. Isso é determinado pelo programador mas a interface deverá mostrar ao utilizador, para cada luz, o referencial em que ela está definida.

Manipulação do objeto

A interface para editar um objeto deverá ser equivalente à mostrada na Fig. 3. As propriedades estão agrupadas em dois grupos:

- transformações
- material

Na transformação, o utilizador deverá ser capaz de alterar os valores x e z da translação (**position**), a rotação (**rotation**) em torno do eixo y, e os três fatores de escala (**scale**). Os restantes campos deverão estar bloqueados.

No material, o utilizador pode ajustar os valores de **Ka**, **Kd**, **Ks** e shininess (**n**), utilizados pelo modelo de iluminação de Phong. Estes valores, embora sejam apresentados no intervalo [0,255], para cada componente R, G e B, deverão ser convertidos, antes de serem passados aos *shaders*, para a gama [0,1]. O utilizador poderá ainda optar por usar Phong shading ou Gouraud shading para a iluminação do objeto.

Nota: A manipulação de qualquer dos valores expostos na interface gráfica deverá ter efeitos imediatos na visualização do objeto.

Detalhes técnicos

Disposição da interface

O painel que permite controlar as propriedades dos objetos deve situar-se no canto superior esquerdo. Essa colocação deverá ser feita através do ficheiro `style.css`:

```
#object-gui {  
  position: absolute;  
  left: 20px;  
  top: 0px;  
}
```

No exemplo, o objecto com o id “object-gui” é posicionado de forma absoluta no top da janela e com uma margem de 20 pixels à sua esquerda.

Quando o painel é criado no ficheiro `app.js`, deverá ser-lhe então atribuído o mesmo id:

```
const objectGUI = new dat.GUI();  
objectGUI.domElement.id = "object-gui";
```

Informação relativa às luzes e material

O programa deverá usar as seguintes declarações no(s) shader(s) apropriado(s):

```
const int MAX_LIGHTS = 8;
```

```
struct LightInfo {  
  vec4 pos;  
  vec3 Ia;  
  vec3 Id;  
  vec3 Is;  
};
```

```
struct MaterialInfo {  
  vec3 Ka;  
  vec3 Kd;  
  vec3 Ks;
```

```
    float shininess;
};

uniform int u_n_lights;

uniform LightInfo u_light[MAX_LIGHTS]; // The array of lights present in the scene
uniform MaterialInfo u_material; // The material of the object being drawn
```

As luzes de tipo direcional deverão distinguir-se das luzes pontuais por possuírem a coordenada *w* de *pos*, com o valor 0, em vez de 1. O `uniform int u_n_lights` deverá receber o número de luzes ativas em cada momento. Ou seja, a aplicação javascript não deverá enviar aos *shaders* a informação das luzes desativadas.

O material do objeto é representado pela estrutura `MaterialInfo`, a qual tem um campo de nome *shininess* que corresponde ao expoente *n* (no termo de reflexão especular) no modelo de iluminação de Phong.

A iluminação deverá ser calculada pelos *shaders* usando o referencial da câmara. As coordenadas relativas às posições das luzes serão sempre fornecidas pela aplicação javascript já no referencial da câmara.

Código fonte inicial

Recomenda-se a utilização do código fonte da resolução do exercício 27 (ver repositório) como ponto de partida para a realização do projeto.

Avaliação

Estão reservados 2 valores da cotação do projeto para quem use uma representação para o grafo de cena (json carregado dinamicamente ou objeto declarado no código), desde que o formato usado para o grafo contemple a definição das próprias luzes e câmara.