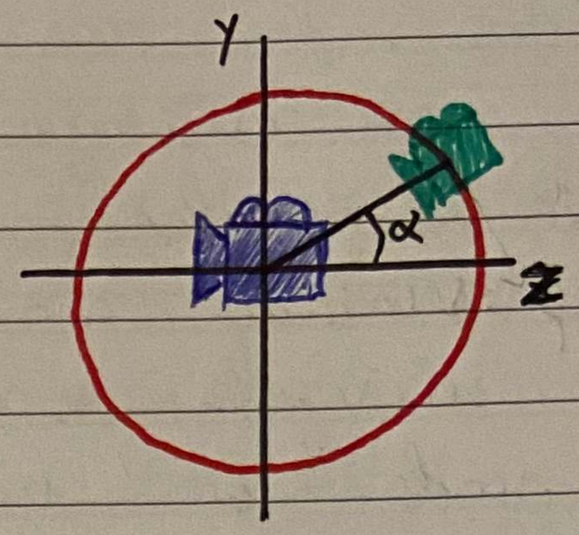
2020/2021

①

a)

gluLookAt (Cos(α), Sen(α), 0, 0, 0, 0, 0, 1, 0)



b)

glTranslate (Cos(α), Sen(α), 0)
glRotate (α, 0, 0, 1)

ou

glRotate (α, 0, 0, 1)
glTranslate (1, 0, 0)

②

gluLookAt (P_1, P_2, P_3, l_1, l_2, l_3, u_1, u_2, u_3)

$\overrightarrow{Din} = \frac{L - P}{\|L - P\|}$

$\overrightarrow{Right} = \overrightarrow{Din} \times \overrightarrow{Up}$

$\overrightarrow{RealUp} = \overrightarrow{Right} \times \overrightarrow{Din}$

$NewP = P + dh \cdot \overrightarrow{RIght} + dv \cdot \overrightarrow{Up}$

③

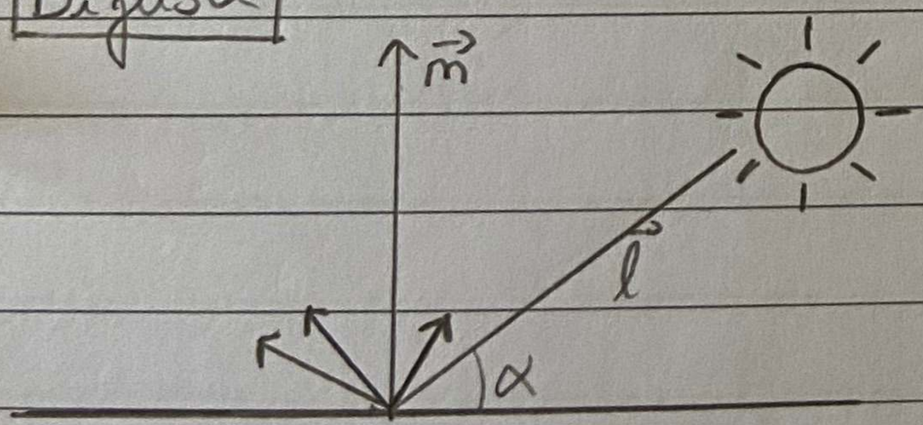
O modelo de iluminação de Gouraud promove a interpolação de cores, tal causa a dificuldade na apresentação da componente especular pois caso tenhamos um triângulo de grandes dimensões a componente especular deste, assumindo que apenas seria apresentada no centro, não irá aparecer pois irá ser utilizada uma interpolação de cores dos vértices onde se vai verificar que a componente especular não existe, assim a interpolação vai dar como resultado a falta de especularidade no objeto.

O modelo de Phong por outro lado utiliza a interpolação de normais dos vértices para cada pixel, desta forma iremos observar uma componente especular próxima daquilo que esperaríamos na realidade.

Em termos computacionais, em objetos "modernos", em que o número de triângulos é superior ao número de triângulos, é mais vantajoso usar o modelo de Gouraud.

④

Difusa



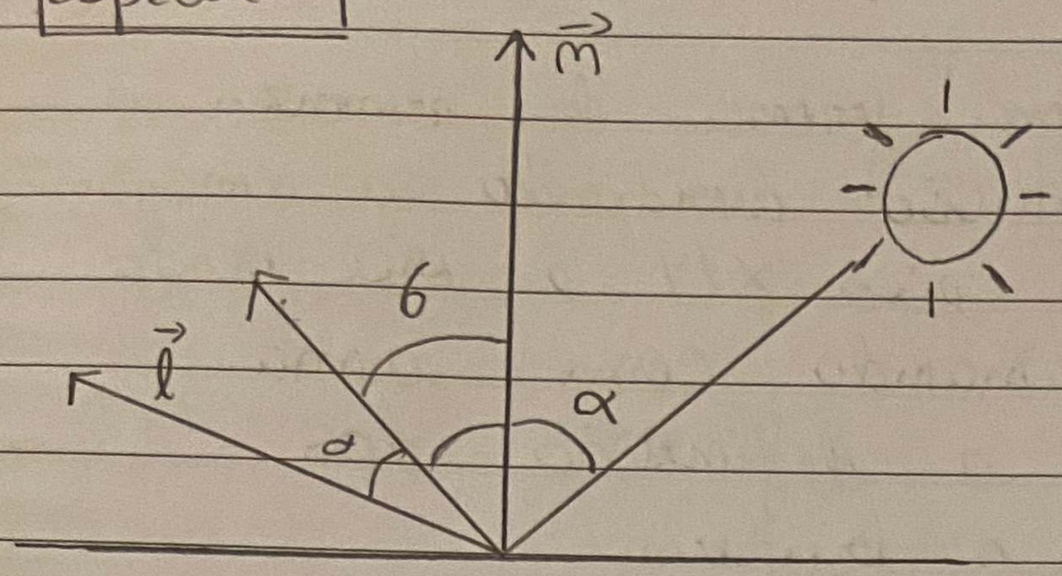
$$I = Id \times Kd \times \cos(\alpha)$$

Id: intensidade

Kd: cor do objeto

α: ângulo de incidência

Especulan



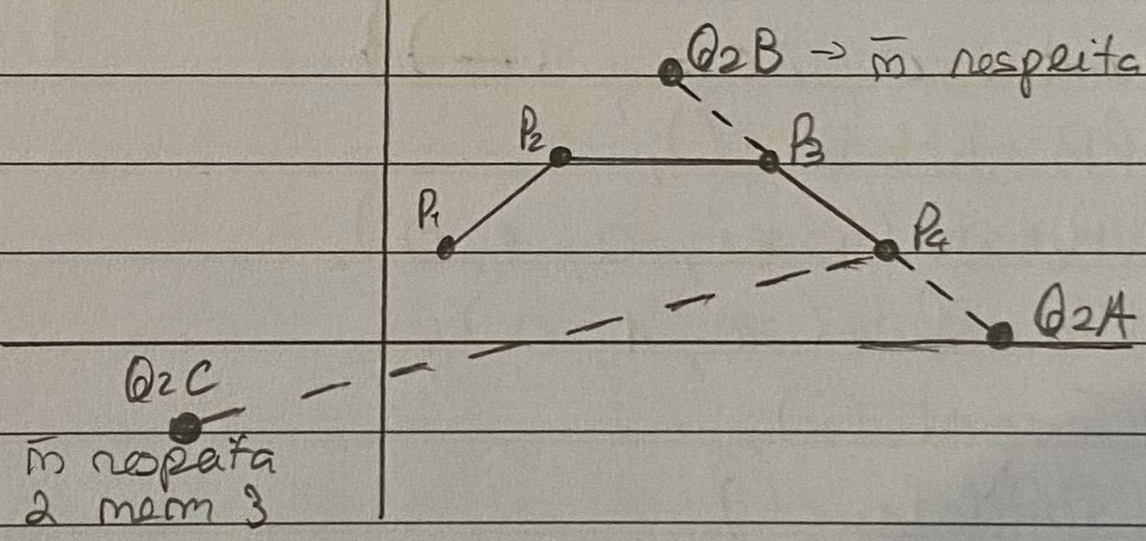
$$I = Is \times Rs \times \cos(6)^S$$

Is: intensidade
Ks: Cor Especular
6: ângulo de reflexão
S: tamanho da mancha brilhante

⑤

1- Q_1 tem de ser igual a P_4
2- $Q_1 - Q_2$ e $P_3 - P_4$ têm de fazer um segmento de reta contínuo
3- $Q_1 - Q_2$ e $P_3 - P_4$ têm de ser o mesmo trabalho

B: Opção B



⑥

```
glBegin(GL_QUADS);
    glTexCoord2f(1.5, 0); glVertex3f(-1.0f, -1.0f, 0.0f);
    glTexCoord2f(0.5, 0); glVertex3f(1.0f, -1.0f, 0.0f);
    glTexCoord2f(0.5, 1); glVertex3f(1.0f, 1.0f, 0.0f);
    glTexCoord2f(1.5, 1); glVertex3f(-1.0f, 1.0f, 0.0f);
glEnd();
```

(7)

A KD-trees surgem como uma forma de remover o grau de liberdade da orientação dividindo a imagem em planos perpendiculares ao eixo x/y e vice-versa na próxima iteração, terminando com uma arvore binária que facilita a definição dos objetos a renderizar com o frustum.

Critérios de Paragem:

- nº de poligonos chega a um limite
- profundidade da árvore for muito grande
- a celula ser demasiado pequena

(8)

```
void renderScene (void) {
int rc = 15;
(...)
for (int i=0; i<m; i++) {
    glPushMatrix();
    glRotate(45*i, 0, 1, 0);
    glTranslate(rc, 1, 0);
    glTeapot(2);
    glPopMatrix();
}
(...)
}
```