DIEGO DANTAS ALMEIDA - 6ª LISTA DE EXERCÍCIOS

1 - Dado um conjunto de objetos (com sua Geometria e Parametria) e uma imagem na tela (estrutura de dados na memória - matriz de pixels com origem no canto inferior esquerdo). O problema da visibilidade é: Como pintar na imagem os objetos certos?

2 - PINTOR

ordene os objetos pela z (de trás para frente)
itere sobre objetos
    itere em y
        intere em x
            escreva o pixel

Z-BUFFER

Inicialize o Z-Buffer (itere em x, y; Zbuff[x,y] = Infinito;)
itere sobre os objetos
    itere em y
        itere em x
            se z(x,y) < Zbuff[x,y]
                Zbuff[x,y] = z(x,y)
                escreva o pixel da imagem

RAY-CASTING

itere em y
    itere em x
        Trace o raio do olho através do pixel(x,y) em direção à cena
        Intersecte todas as superfícies, encontre a primeira que o raio toca
        Pinte o pixel de acordo

RAY-TRACING

SIGA CADA FÓTON DA FONTE enquanto:
  Se energia é totalmente absorvida sai do loop.
  Senão se sair do Frustrum sai do loop.
  Senão se tocar o plano imagem:
    Escreva o pixel na imagem

3- PINTOR - .

Vantagens - Simplicidade. Rápido se os objetos forem pré-classificados. Trabalha com transparências.

Desvantagens - Implementação difícil e lenta se for necessário classificar e dividir os objetos. Difícil de ordenar para objetos não poligonais. Em geral não muito adequado para renderização 3D, pela dificuldade de "sorting" e "spliting".

Z-BUFFER.

Vantagens - Implementação de nível moderado, com velocidade alta (exceto quando a complexidade da profundidade é muito alta). Fácil de mixar com polígonos, esferas e outras primitivas geométricas.

Desvantagens - Velocidade dependente da complexidade. Não faz transparência. Necessita de uma boa resolução do Z-buffer para não gerar artefatos.

RAY-CASTING

Vantagens - Implementação fácil e de excelente generalidade,

podendo realizar GSC (Geometria Sólida Construtiva), transparência e sombras.
Desvantagens - Lento se tiver muitos objetos (difícil de executar rapidamente).

4 - Dependendo do tamanho e formato do objeto, usaria o Z-Buffer, pois apesar de ser somente um objeto, ele equivale a muitos pequenos objetos triangulares. Dessa forma, pode haver sobreposição destes pequenos triângulos e necessidade de ordená-los, o que tornaria difícil a implementação do Algoritmo pintor. No caso do Ray Casting se o objeto tiver muitos triângulos pode aumentar muito a complexidade do algoritmo e torná-lo mais lento.

5 - O princípio básico do Forward Ray Tracing é seguir cada fóton da fonte de Luz até que aconteça uma das seguintes coisas:
- Toda sua energia seja absorvida (depois de muitas reflexões);
- Ele saia do universo conhecido (Frustum);
- Ele bata no plano imagem e sua contribuição seja adicionada ao respectivo pixel.

6 - O Forward Ray Tracing acompanha todos os raios de luz que saem da fonte e que são refletidos nos objetos, já o Backward Ray Tracing faz o caminho contrário acompanhando o raio da imagem até o objeto ou fonte. → continua

O principal problema do primeiro método é que apenas uma pequena parte dos raios alcançam a imagem, sendo extremamente lento calcular o percurso de todos os raios de uma fonte de luz. O Backward Ray Tracing resolveu o problema, pois calcula apenas os raios que contribuem para a formação da imagem.

7- A ideia básica do Ray Tracing recursivo é, assim como o Ray Tracing Convencional, acompanhar o raio até que alcance o objeto, a fonte ou saia do universo conhecido. No entanto, se alcançar um objeto, determina-se os raios refletidos e refratados, percorre-se esses raios para detectar se são provenientes de uma outra reflexão ou refração, calcula-se novamente os raios refletidos e refratados, percorre-se novamente e assim por diante de forma recursiva até a fonte de luz. Então são calculadas as contribuições de cada reflexão/refração para o Pixel. Também são calculados os "raios de sombra" no ponto da superfície. Tipos de raios:

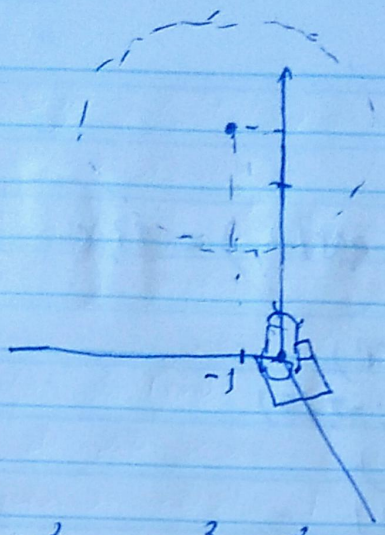
Eye rays: Originam-se no olho

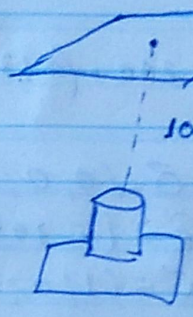
Shadow rays: Do ponto na superfície na direção da luz

Reflection rays: do ponto na superfície na direção refletida

Transmission rays: Do ponto na superfície na direção refratada.

8 -





1. $(x+1)^2 + y^2 + (z-4)^2 = 2^2$ Como é o pixel central da imagem, e a câmera está na origem apontando para o eixo z. O ponto de interseção estará em cima do eixo z.

$(0+1)^2 + 0^2 + (z-4)^2 = 2^2 \Rightarrow 1 + z^2 - 8z + 16 = 4$

$z^2 - 8z + 13 = 0$

$z' = 4 - \sqrt{3} \cong 2,268$ (mais próximo)

$z'' = 4 + \sqrt{3} \cong 5,732$

Logo será o ponto $(0; 0; 2,268)$

2- Normal $\vec{N} = (0;0;2,27) - (-1;0;4) = (1;0;-1,73) = \vec{N}$

$\vec{L} = (1,1,1) - (0;0;2,27) = (1;1;-1,27)$

$\vec{R} = 2N(N \cdot L) - L = 6,3942N - L = (6,3942; 0; -11,06) - L$

$N \cdot L = 1 + 0 + 2,197 = 3,197$

$(6,3942; 0; -11,06) - (1;1;-1,27) = (5,3742; -1; -9,79) = \vec{R}$

$\vec{O} = (0;0;-2,268)$ $\cos\varphi = \dfrac{\vec{R} \cdot \vec{O}}{|\vec{R}||\vec{O}|} = \dfrac{-22,27}{11,22 \cdot 2,268} = 0,872$

$\cos\varphi = 0,872$

$\cos\theta = 0,442$ $\cos\theta = \dfrac{\vec{N} \cdot \vec{L}}{|N| \cdot |L|} = \dfrac{3,197}{1,998 \cdot 3,6127} = 0,442$

$I_{d+a} = k_a I_a + f_{att} I_{light} (k_d \cdot \cos\theta + k_s ((\cos\varphi)^{n_{shiny}}))$

$I_{d+a} = 0,5 \cdot 100 + 200 (k_d \cdot 0,442 + k_s \cdot (0,872))$

$0,736 k_d = 2,18 k_s$

$k_d = 2,96 k_s$

$\dfrac{0,442 k_d}{0,442 k_d + 0,872 k_s} = 0,6$ $\dfrac{0,872 k_s}{0,442 k_d + 0,872 k_s} = 0,4$

$K_d = 0,276$

$K_n = 0,1$

$I_{d+a} = 50 + 200 \cdot (0,1308 + 0,0872) = 93,6$

3 - Vetores $\vec{O} = (0,0; -2,268)$

$\vec{R} = (5,394; -1; -9,19)$

$\vec{\mu} = (1; 0; -1,73)$

$\vec{\Gamma} = (1; 1; -1,27)$