

Ray Tracing: O Mundo Através De Raios de Luz

XXXVI Jornada Giulio Massarani de Iniciação Científica, Tecnológica,
Artística e Cultural

Thiago Barroso Perrotta
Prof.^o Ricardo Marroquim

Universidade Federal do Rio de Janeiro

10 de outubro de 2014



Agenda

- 1 Ray-tracer
- 2 Extração de primitivas em nuvens de pontos
- 3 Na prática
 - Resultados
 - Conclusão
 - Referências

Ray-tracer

Conceituando

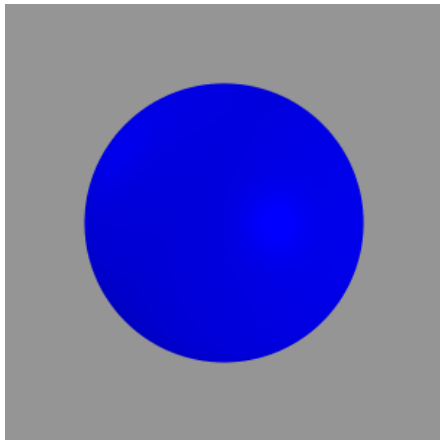
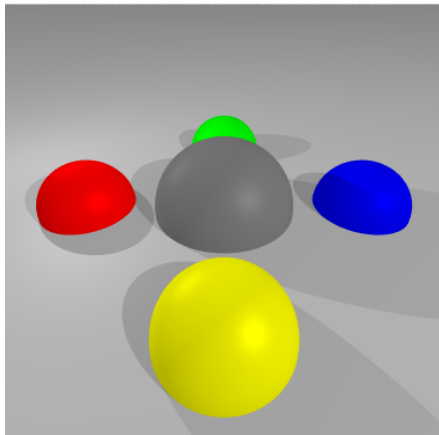
- Interação física da **luz** com objetos
- Modelo físico, com diversas **aproximações matemáticas**
- Renderização de imagens com alto grau de **realismo**

O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

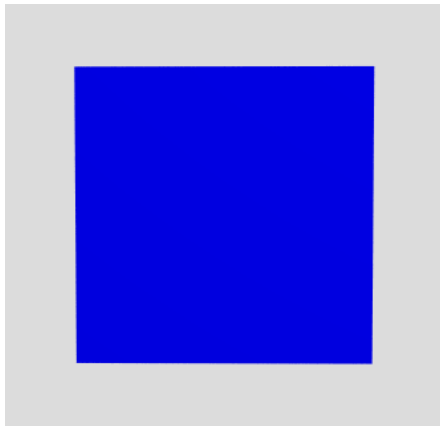
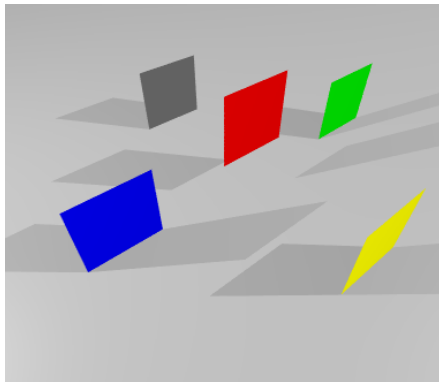
Objetos

Esferas



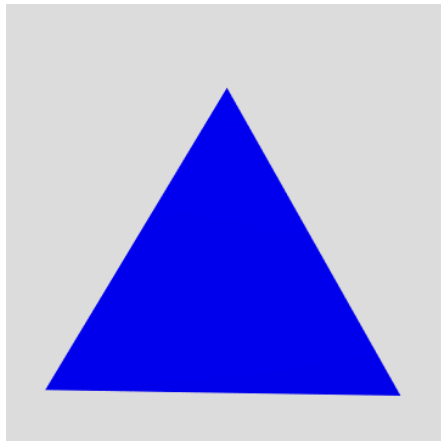
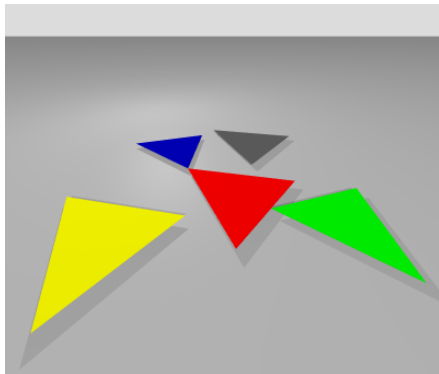
Objetos

Retângulos



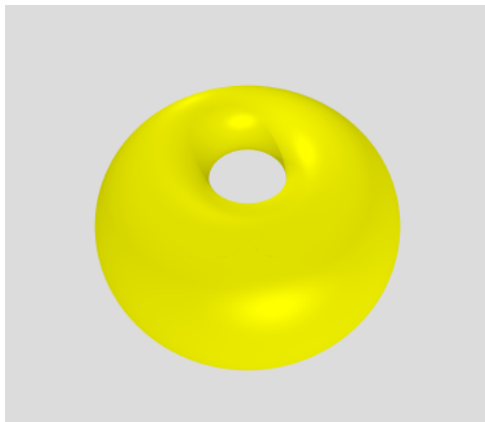
Objetos

Triângulos



Objetos

Toros



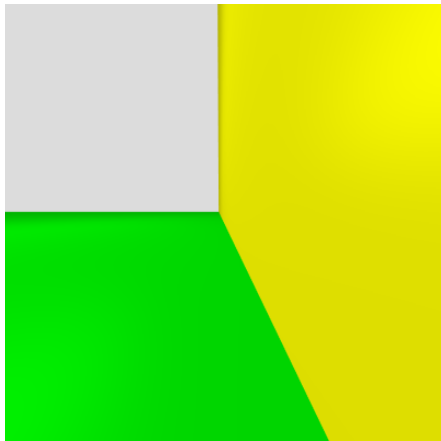
Objetos

Cilindros



Objetos

Planos



O algoritmo

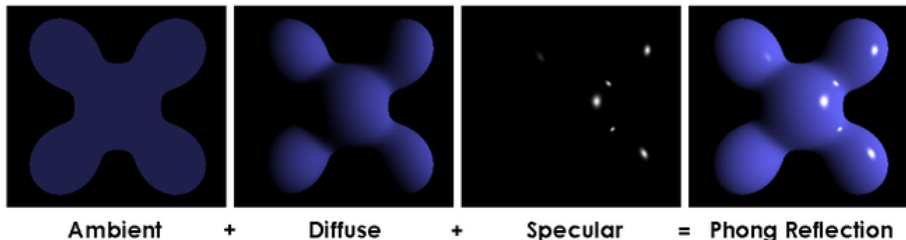
- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

Os materiais

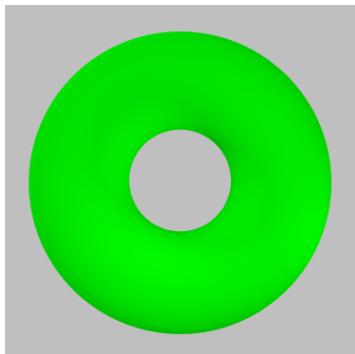
⇒ **Como** dado objeto deve **interagir** com a luz?

Tipos de Iluminação

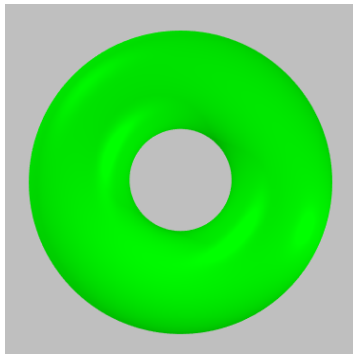
- Ambiente
- Difusa
- Especular



Os materiais



Matte: interação
ambiente + difusa



Phong: interação
ambiente + difusa + especular

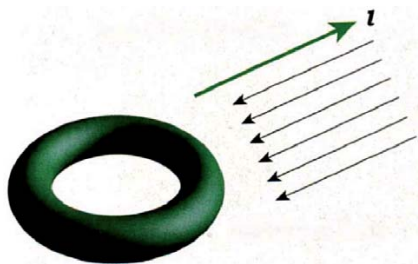
O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

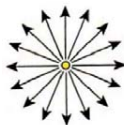
As fontes de luz

Tipos

- Direcionais
- Pontuais



Luz direcional



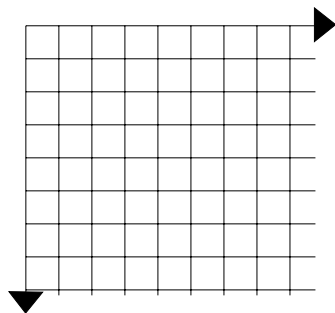
Luz pontual

O algoritmo

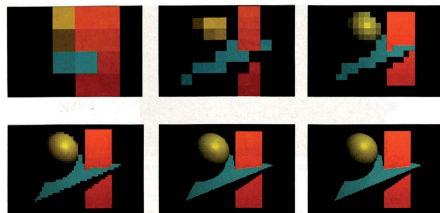
- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

O plano de visualização

- **Resolução** (número de *pixels*) \implies Ex.: 400×400
- Tamanho de cada *pixel*
- Câmera



Plano de Visualização



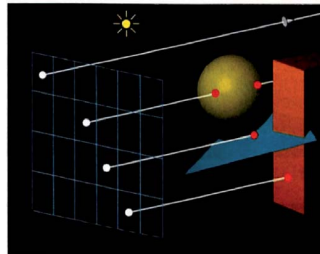
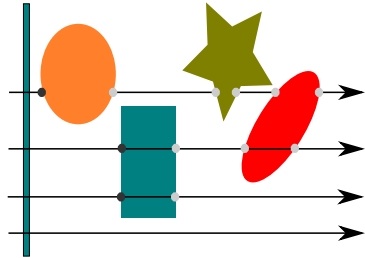
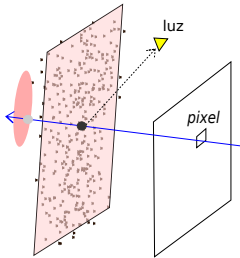
Várias resoluções distintas

O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

Interseção entre um raio e vários objetos

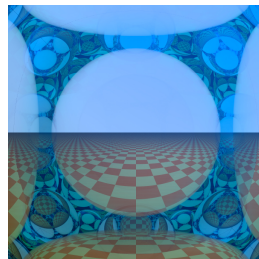
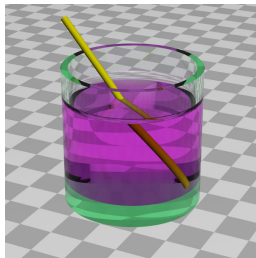
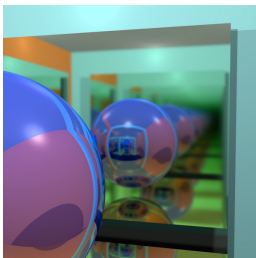
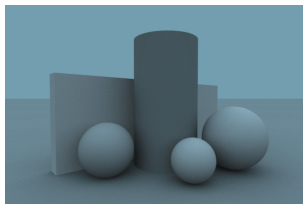
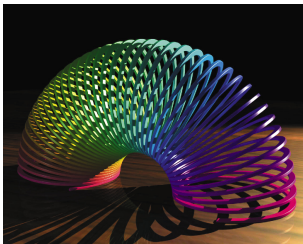
- Função HIT para cada objeto



O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina um plano de visualização
- Para cada *pixel*
 - Atire um raio do centro do *pixel* na direção dos objetos
 - Dentre os pontos atingidos, compute o mais próximo
 - Se o raio atingiu algum objeto
 - Use o material do mesmo e as luzes para computar a cor do *pixel*
 - Caso contrário
 - Ponha o *pixel* com a cor de fundo

Exemplos



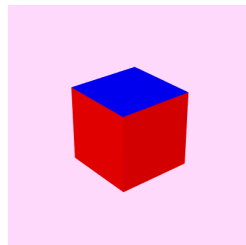
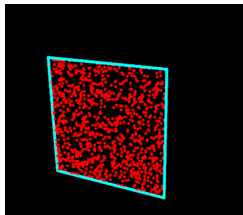
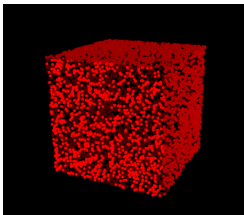
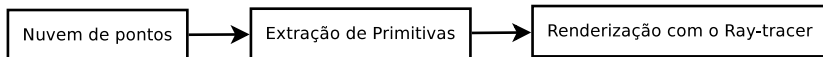
Agenda

- 1 Ray-tracer
- 2 Extração de primitivas em nuvens de pontos
- 3 Na prática
 - Resultados
 - Conclusão
 - Referências

Extração de primitivas em nuvens de pontos

Conceituando

- **Origens:** Modelos 3D escaneados
- **Motivações:** simplificação, compactação

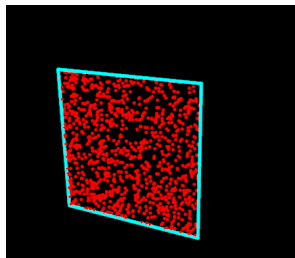
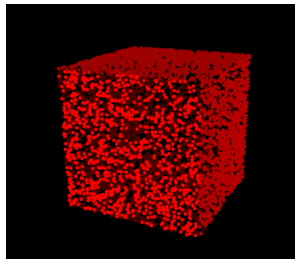


Primitivas: cones, cilindros, planos, esferas, toros.

Exatção de primitivas em nuvens de pontos

Algoritmo RANSAC

- Tome alguns pontos aleatoriamente
- Considere a primitiva que eles formam
- Verifique se ela é um bom candidato
- Repita esse procedimento até dado limite, selecionando as melhores primitivas



Agenda

- 1 Ray-tracer
- 2 Extração de primitivas em nuvens de pontos
- 3 Na prática
 - Resultados
 - Conclusão
 - Referências

Alguns resultados

Contexto

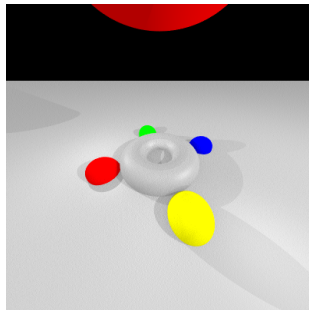
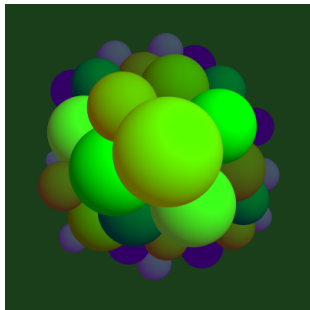
Linguagem de Programação	C++ (g++)
Gerenciamento de <i>Build</i>	CMake
<i>Kit</i> gráfico	Qt 5
<i>Framework</i> de testes	Google Test

Ambiente de testes

- Ubuntu 14.04 64-bit
- Intel Core i5-3317U @ 1.7GHz x 4
- 4GB de memória

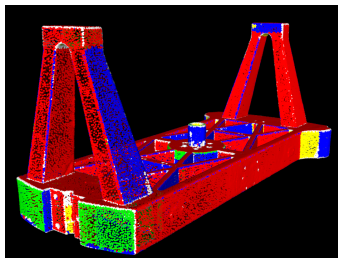
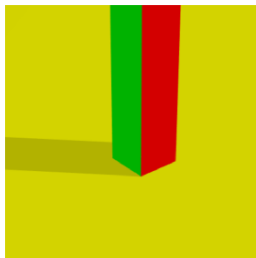
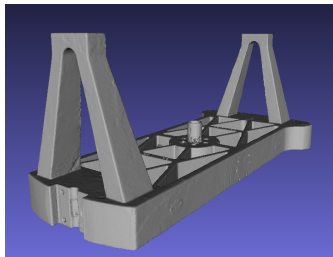
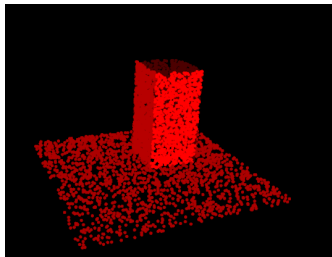
Alguns resultados

Ray-tracing



Alguns resultados



Extração de primitivas



Conclusão e Ideias Futuras

- Ray-tracer
 - Esquemas de aceleração
 - Mais recursos
 - Paralelização
- Extração de primitivas
 - Melhorar algoritmos
 - Testar com modelos mais complexos
 - Salvar estados intermediários

Referências

-  Suffern, Kevin Geoffrey, and Suffern, Kevin. Ray Tracing from the Ground up. AK Peters, 2007.
-  Schnabel, Ruwen, Roland Wahl, and Reinhard Klein. "Efficient RANSAC for Point-Cloud Shape Detection." Computer graphics forum. Vol. 26. No. 2. Blackwell Publishing Ltd, 2007.