

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 G. marroquim

Universidade Federal do Rio de Janeiro

10 de outubro de 2014



Agenda

- 1 Ray-tracer
 - O algoritmo
- 2 Extração de primitivas em nuvens de pontos
- 3 Resultados
 - Contexto
 - Imagens
 - Conclusão
 - Referências

Ray-tracer

Conceituando

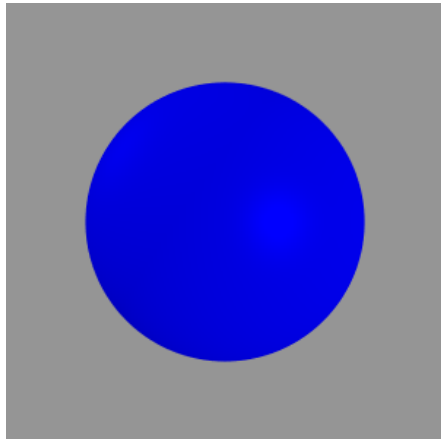
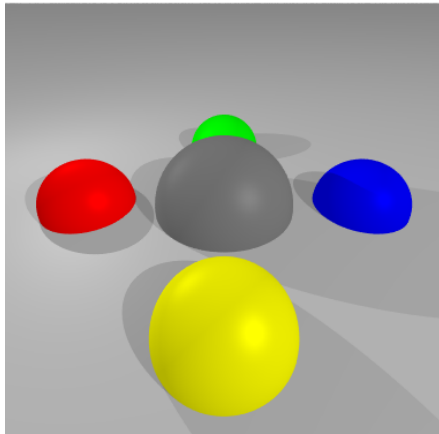
- Renderização de imagens
- Realismo
- % Mais sugestões? – incrementar

O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

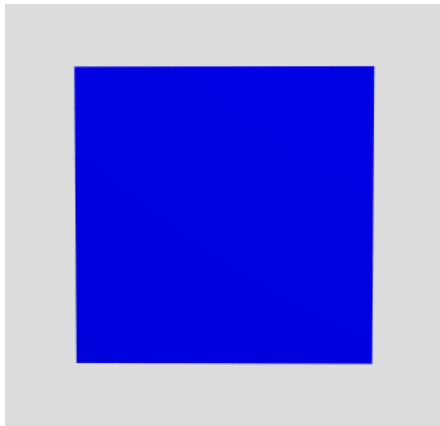
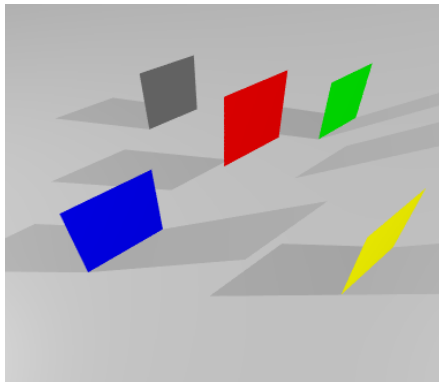
Objetos

Esferas



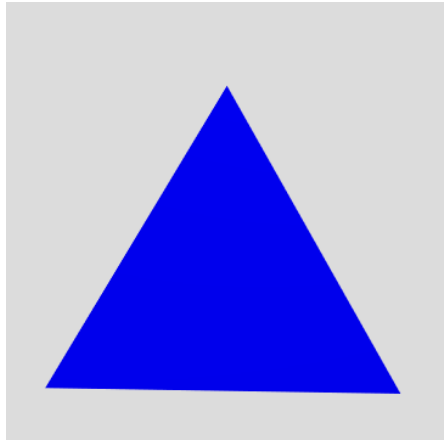
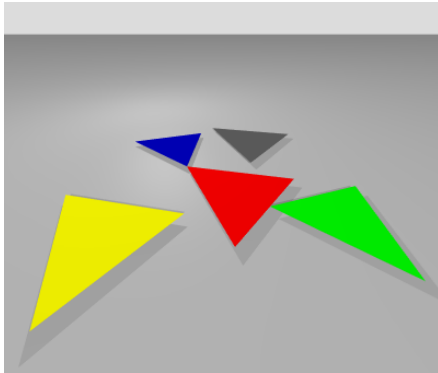
Objetos

Retângulos



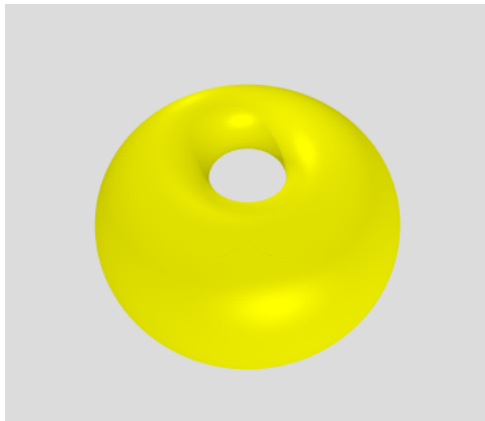
Objetos

Triângulos



Objetos

Toros



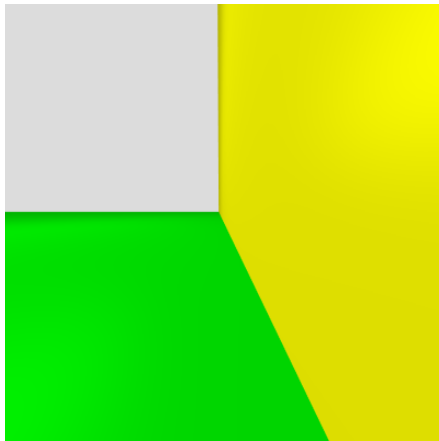
Objetos

Cilindros



Objetos

Planos



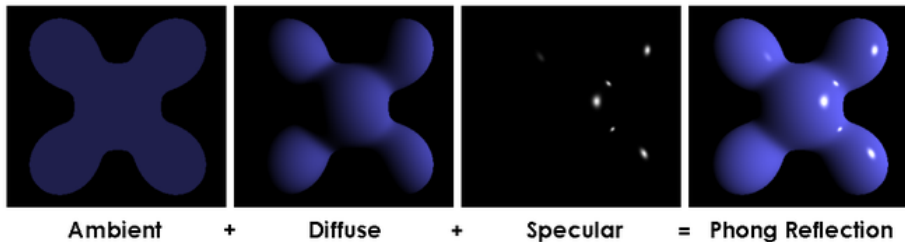
O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

Materials

Tipos de iluminação

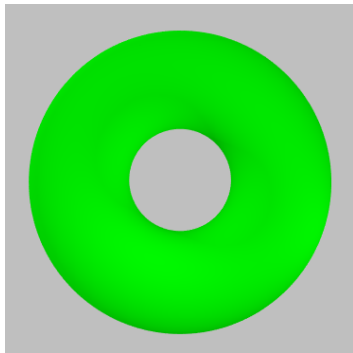
- Ambiente
- Difusa
- Especular



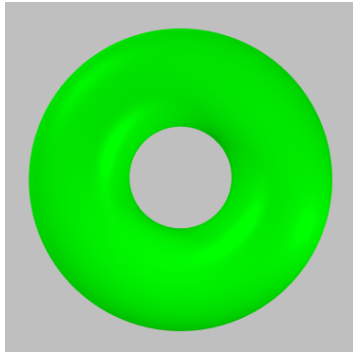
Materials

Tipos

- *Matte* = Luzes: Ambiente + Difusa
- *Phong* = Luzes: Ambiente + Difusa + Especular



Matte



Phong

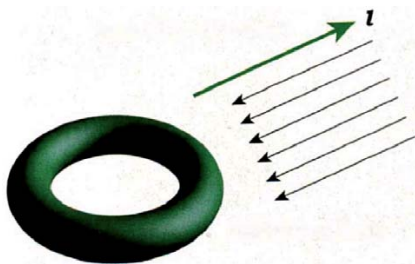
O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

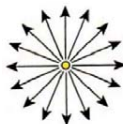
Fontes de luz

Tipos

- Direcional
- Pontual



Direcional



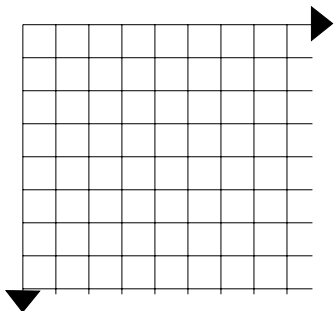
Pontual

O algoritmo

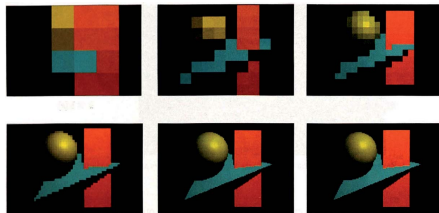
- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

Plano de visualização

- Número (horizontal + vertical) de *pixels* (ex.: 400×400)
- Tamanho de cada *pixel* \Rightarrow *zoom*



Plano de Visualização



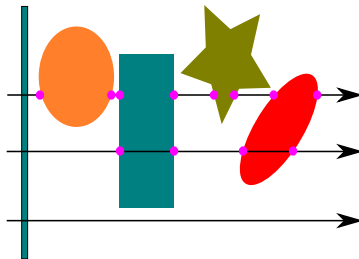
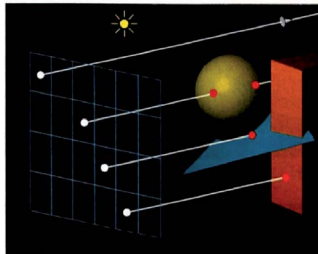
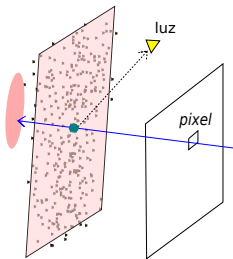
Diferentes resoluções

O algoritmo

- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

Interseção entre raio e objetos

- Função *Hit* para cada objeto



O algoritmo

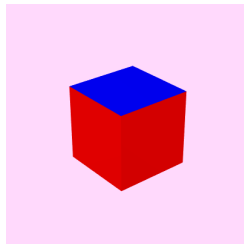
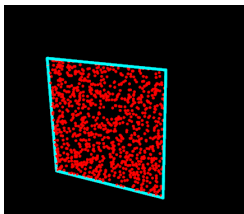
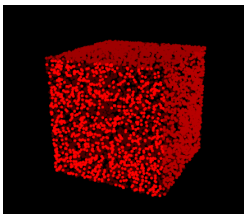
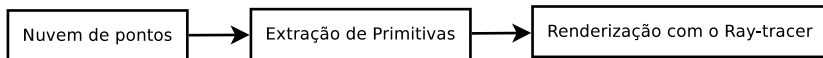
- Defina alguns objetos
- Especifique um material para cada objeto
- Defina algumas fontes de luz
- Defina uma janela cuja superfície seja coberta com pixels
- Para cada *pixel*
 - Atire um raio, a partir do centro do *pixel*, na direção dos objetos
 - Compute, dentre os pontos atingidos, o mais próximo
 - Se o raio atingir um objeto
 - Use o material do objeto e as fontes de luz para computar a cor do *pixel*
 - Senão
 - Ponha a cor do *pixel* como preta

Agenda

- 1 Ray-tracer
 - O algoritmo
- 2 Extração de primitivas em nuvens de pontos
- 3 Resultados
 - Contexto
 - Imagens
 - Conclusão
 - Referências

Exatção de primitivas em nuvens de pontos

Conceituando



● Primitivas

- cones
- cilindros
- planos
- esferas
- toros

Extração de primitivas em nuvens de pontos

Algoritmo RANSAC

- % explicar, esquema, ... \leftarrow Daniel
- % possivelmente pseudocódigo nessa parte

Agenda

- 1 Ray-tracer
 - O algoritmo
- 2 Extração de primitivas em nuvens de pontos
- 3 Resultados
 - Contexto
 - Imagens
 - Conclusão
 - Referências

Contexto

Linguagem de Programação	C++ (gcc), com O.O.
<i>Kit gráfico</i>	Qt 5
Gerenciamento de <i>build</i>	CMake
<i>Framework</i> de testes	Google Test

Ambiente

- Ubuntu 14.04 LTS 64-bit
- Intel Core i7 950 @ 3.07 GHz x 8 cores
- 16 GB de RAM

Alguns resultados

Ray-tracing

% Imagens

% Tabela

Alguns resultados

Extração de primitivas

% Imagens



% Tabela

O Futuro

Conclusão

- Ideias futuras
 - Ray-tracer
 - Esquemas de aceleração
 - Mais recursos
 - Paralelização
 - Extração de primitivas
 - Melhorar algoritmos (memória, performance)
 - Nuvens de pontos mais complexas
 - Serialização (salvar estados)

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.

TODO

- [] Introducao
- [] Extracao de primitivas (RANSAC, etc)
- [] Exemplos finais + benchmarking de ray-tracer
- [] Exemplos finais + benchmarking de primitivas