INF 2063 – Tópicos em CG III

Visualização de Modelos Massivos

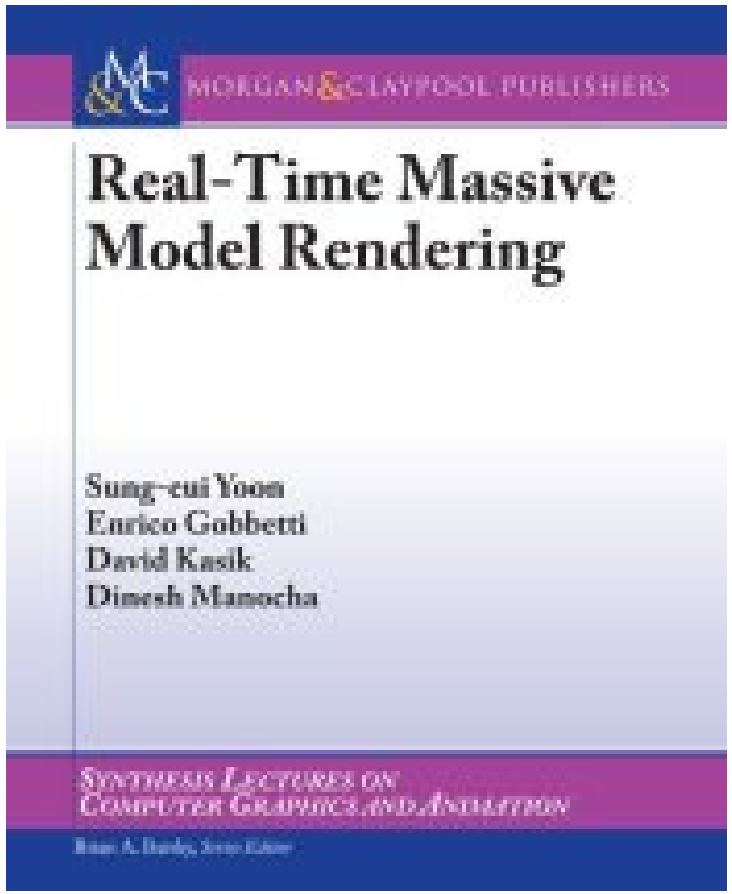
Prof. Alberto Raposo
Tecgraf / DI / PUC-Rio

Aula 04

Representações Alternativas

Alberto Raposo - 2010

Livro



1. Introdução √
2. Visibilidade √
3. Simplificação e Níveis de Detalhe √
4. Representações Alternativas (hoje)
5. Cache-Coherent Data Management

Tecgraf PUC-RIO VR GROUP

Outra referência

- Gustavo Nunes Wagner. "*Visualização Interativa de Modelos Massivos de Engenharia na Indústria de Petróleo com o Algoritmo de Voxels Distantes*". Dissertação de Mestrado, Depto. de Informática, PUC-Rio. Orientadores: Marcelo Gattass e Alberto Raposo. Abril 2007.
  - http://www.tecgraf.puc-rio.br/~abraposo/pubs/alunos/dissertacao_GustavoWagner.zip

Malhas de triângulos

- Técnicas de simplificação e LOD visam reduzir complexidade da renderização, computando e explorando representações “filtradas” dos detalhes dos objetos
  - Em malhas de triângulos: simplificação das malhas, organização dos detalhes em estruturas multi-resolução, etc
- Redução da complexidade de renderização também pode ser obtida com representações diferentes de malhas de triângulos.

Outras Representações

- 3 abordagens
  - Higher order representations
    - Ex., Modelos CAD podem ser bem representados por primitivas de maior ordem: cúbicas, quádricas, etc.
    - Vantagens: reduz uso de memória e gera superfícies “smooth” mesmo com visão bem próxima
  - Sample-based representations
    - Em representações complexas, polígono pode representar menos que um pixel na imagem final
      - Representações tipo voxels, pontos, etc, podem tirar vantagem disso
  - Image-Based representations

Higher Order Representations

Alberto Raposo - 2010

Modelos Paramétricos

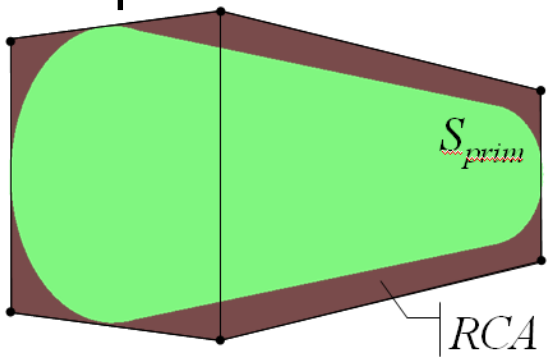
- Modelos CAD, em sua maioria, são originalmente definidos por representações de maior ordem: representações paramétricas de curvas, superfícies ou volumes (Bézier, Splines, etc), definições canônicas.
- Normalmente, essas primitivas são pré-computadas a partir dessa definição geométrica de maior ordem.
  - Converte-se para malhas de triângulos (ou alguma forma intermediária)
    - Tradicionalmente, rendering direto das primitivas de maior ordem era muito lento e impraticável mesmo em modelos pequenos.

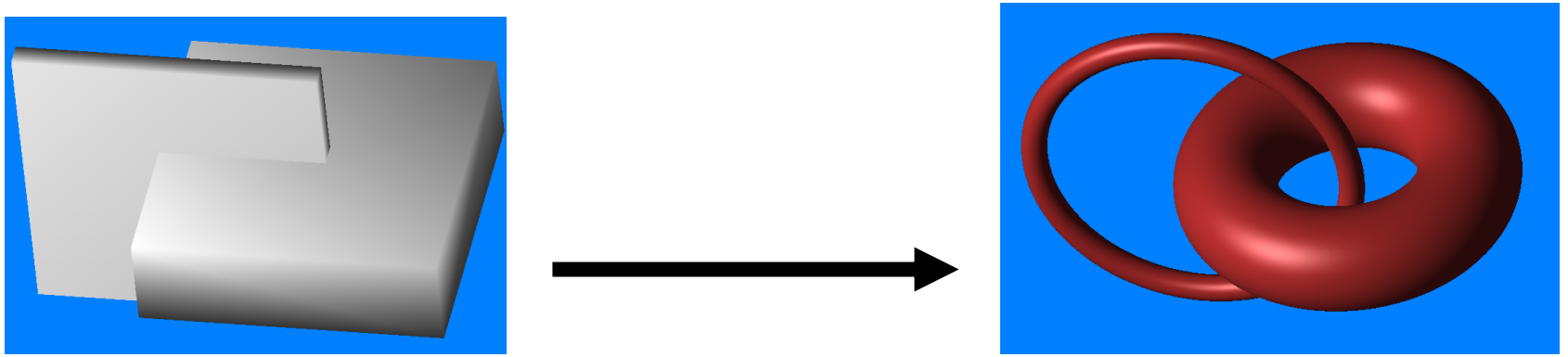
Em GPU

- Uso de GPU para renderizar diretamente as superfícies de maior ordem tem começado a obter sucesso
- O objetivo é renderizar superfícies implícitas na GPU
  - O mais rápido possível
  - Da maneira mais precisa possível

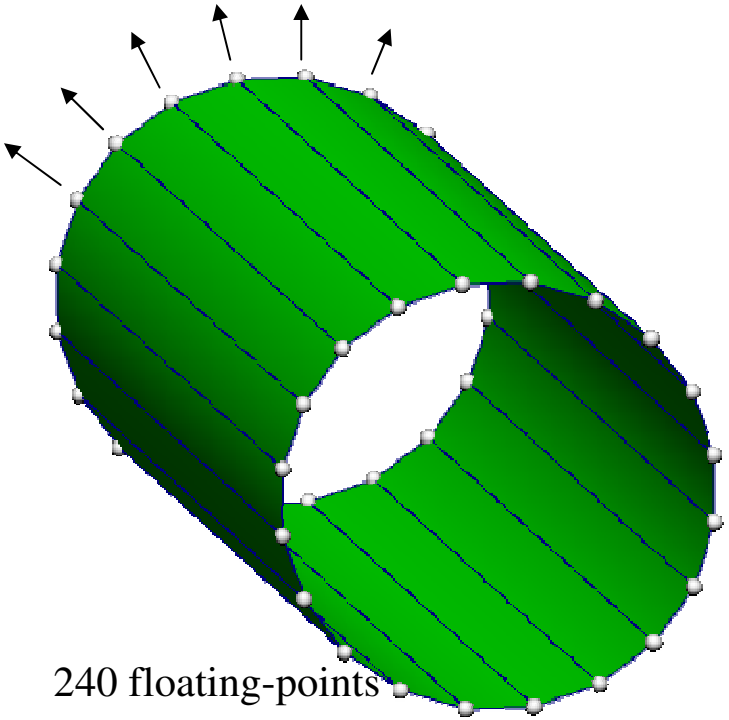
Ideia Básica

- Define-se uma área de raycasting (RCA – Raycasting Area), que engloba a primitiva a ser renderizada

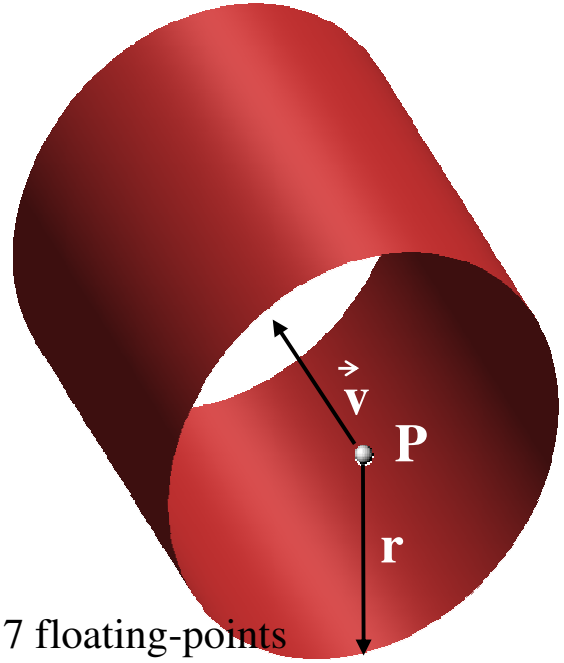




Ganho de Memória



240 floating-points



7 floating-points

Total floating points: n*6

n is the number of vertices

6 ➔ 3 for position + 3 for normal

Total floating points: 7

3 for position of *P* + 1 for *r*

+ 3 for *v* direction

GPU-based Higher Order Representations

- "Real-time GPU rendering of piecewise algebraic surfaces". C. Loop, J. Blinn. SIGGRAPH 2006, pp. 664-670.
  - Renderiza em GPU superfícies obtidas a partir da sua representação polinomial (Tetraedros de Bézier)

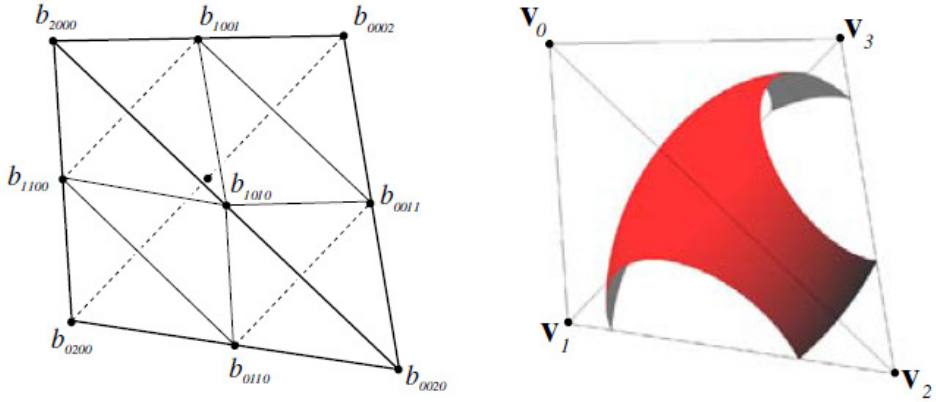


Figure 2: A Quadratic Bézier tetrahedron. On the left, the layout of Bézier weights within a tetrahedron. On the right, vertex labeling together with an algebraic surface restricted to the tetrahedron.

Tecgraf PUC-RIO VR GROUP

Exemplos do artigo anterior

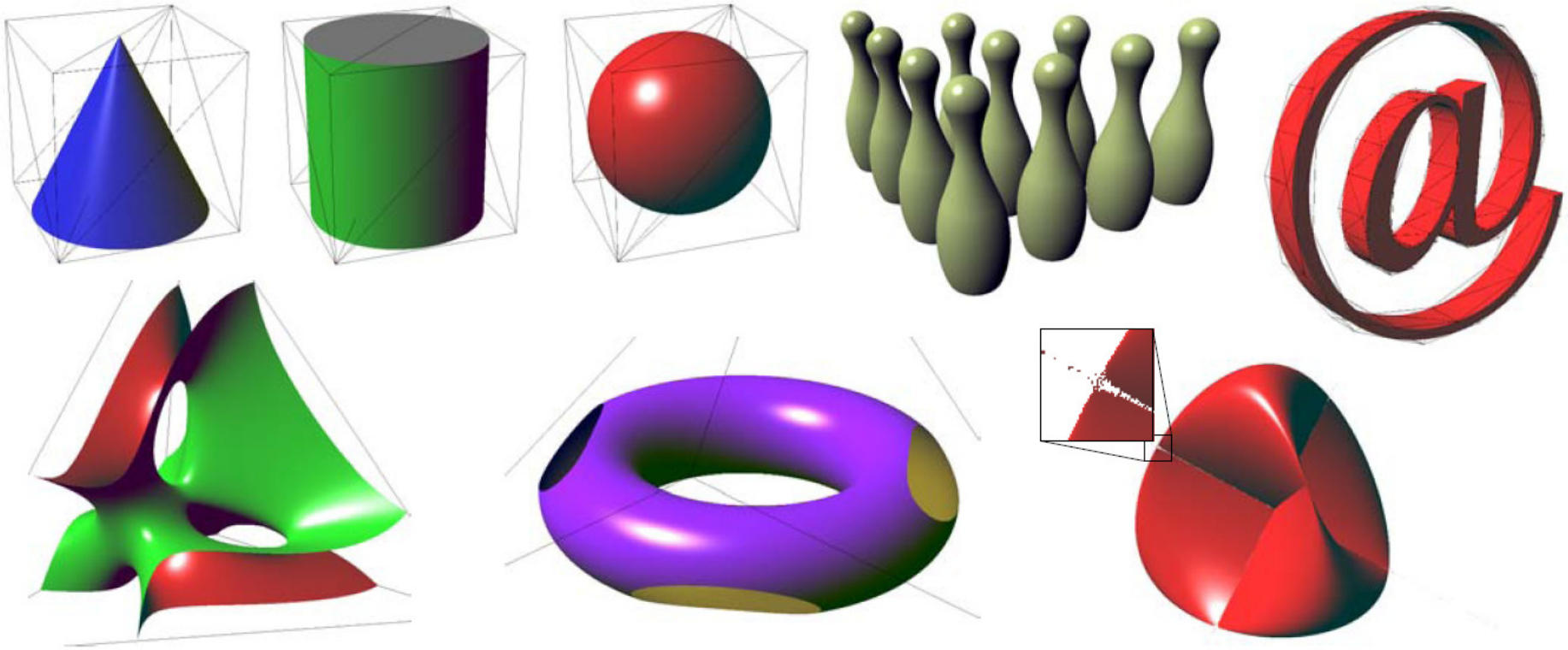


Figure 5: Some sample piecewise algebraic surfaces composed of Bézier tetrahedra and rendered using our technique.

Vídeo:
http://delivery.acm.org/10.1145/1150000/1141939/p664-loop.mov?key1=1141939&key2=0735092721&coll=GUIDE&dl=GUIDE&CFID=86951917&CFTOKEN=49912175

Limitações da abordagem de Loop & Blinn

- Baseada exclusivamente em tetraedro de Bézier
- Resolve equações analiticamente
- Tem gargalos de desempenho
  - Muitos fragmentos descartados quando o tetraedro de Bézier não se encaixa bem na superfície
- Perde precisão em superfícies muito próximas

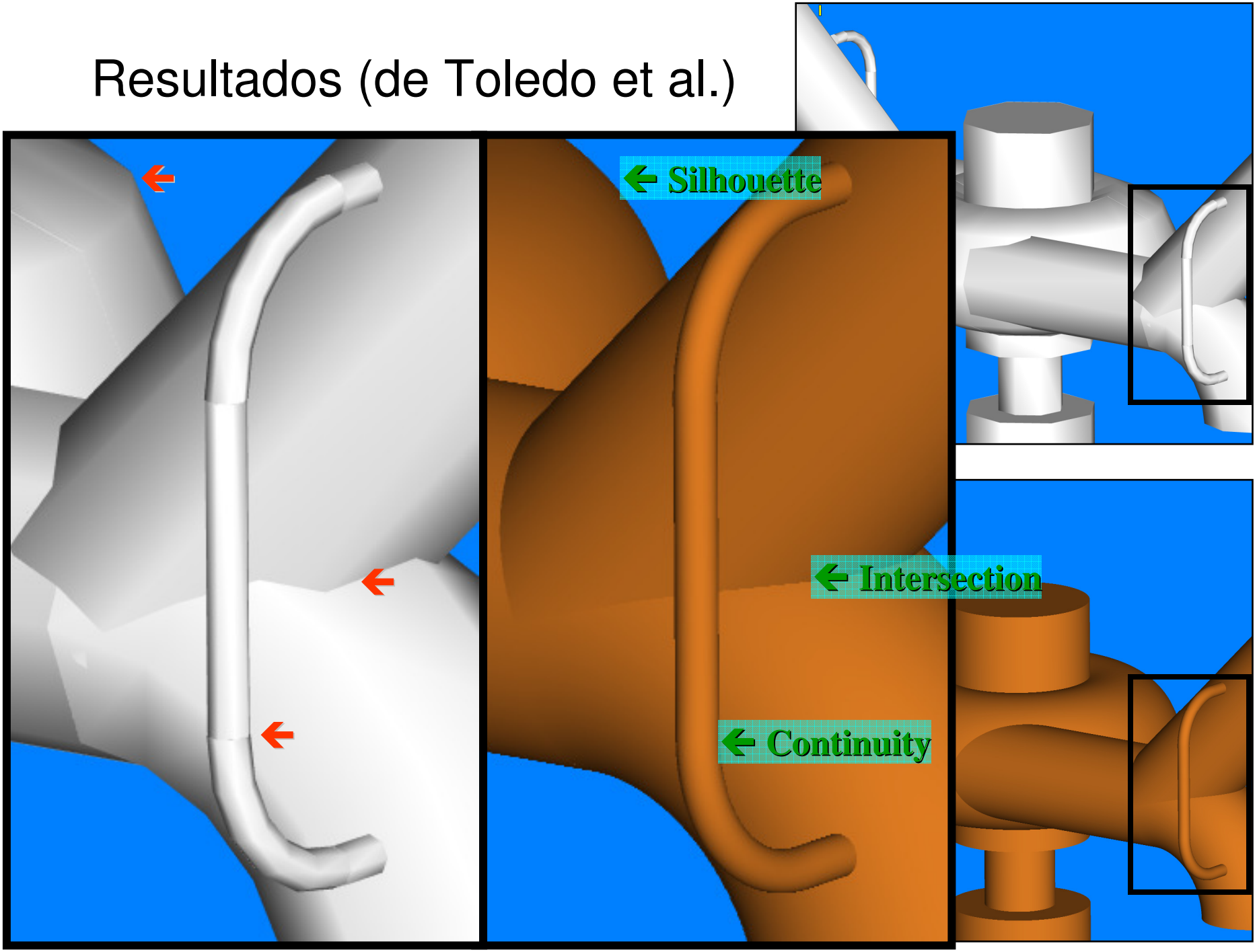
GPU-based Higher Order Representations

- “Interactive Visualization of Massive Data using Programmable Graphics Cards.” Rodrigo de Toledo. (Adivisors: Jean-Claude Paul and Bruno Levy) *Ph.D. Thesis, Université Henri Poincaré, October 2007*.
- “Iterative Methods for Visualization of Implicit Surfaces on GPU.” Rodrigo de Toledo, Bruno Levy, Jean-Claude Paul. *ISVC, International Symposium on Visual Computing, 2007*.

Abordagem de “de Toledo et al.”

- Usa método iterativo ao invés do analítico de Loop & Blinn
- Não se limita a tetraedros de Bézier
  - Cúbicas e Torus / Torus slices

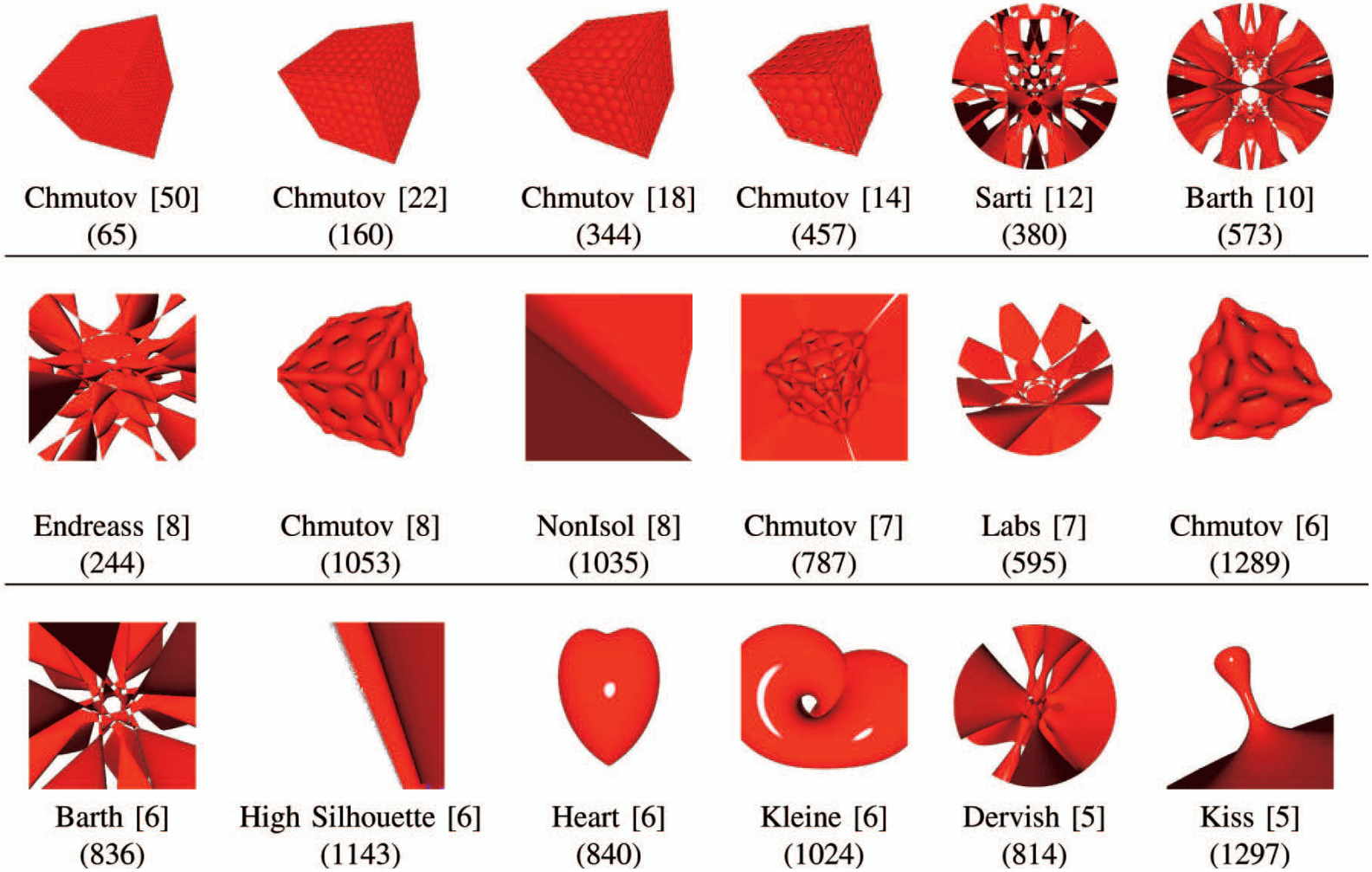
Resultados (de Toledo et al.)



GPU-based Higher Order Representations

- Abordagem por Ray Tracing
  - "Real-Time Ray Tracing of Implicit Surfaces on the GPU" J. M. Singh, P.J. Narayanan. IEEE Transactions on Visualization and Graphics, 16(2): 261-272. March-April 2010.
    - Usam um "adaptive marching points" algorithm para renderizar com taxas interativas superfícies complexas de ordens superiores

Exemplos do artigo anterior



Vídeo: 05_GPUprimitives_indianos2010.avi

Higher Order vs Triangles

- Uso direto das primitivas
  - Melhor aparência quando se aproxima (zoom)
  - Menores datasets (menos consumo de memória)
  - Renderização ainda mais lenta que a de malhas
  - Não resolve por si só os problemas dos modelos massivos, pois modelos complexos podem conter número elevado de primitivas

Outras Representações

- 3 abordagens
  - Higher order representations
    - Ex., Modelos CAD podem ser bem representados por primitivas de maior ordem: cúbicas, quádricas, etc.
    - Vantagens: reduz uso de memória e gera superfícies "smooth" mesmo com visão bem próxima
  - Sample-based representations
    - Em representações complexas, polígono pode representar menos que um pixel na imagem final
      - Representações tipo voxels, pontos, etc, podem tirar vantagem disso
  - Image-Based representations

Sample-based Representations

Alberto Raposo - 2010

Representações baseadas em amostras

- Estão no extremo oposto às representações de maior ordem: exploram métodos discretos para a representação de modelos através de conjuntos de amostras
- Exemplos:
  - Point-based rendering
  - Representações volumétricas
  - Sample-based LOD

Point-based Rendering

- Uma representação de geometria baseada em pontos pode ser considerada uma amostragem discreta de uma superfície
  - Simples
  - LOD “imediato”
  - Como reconstruir as imagens contínuas (i.e., sem buracos) a partir das amostras?

Point-Based Rendering & Massive Models

- “Layered point clouds: a simple and efficient multiresolution structure for distributing and rendering gigantic point-sampled models”. E. Gobbetti, F. Marton. Computers & Graphics 28: 815–826, 2004.

Layered Point Clouds

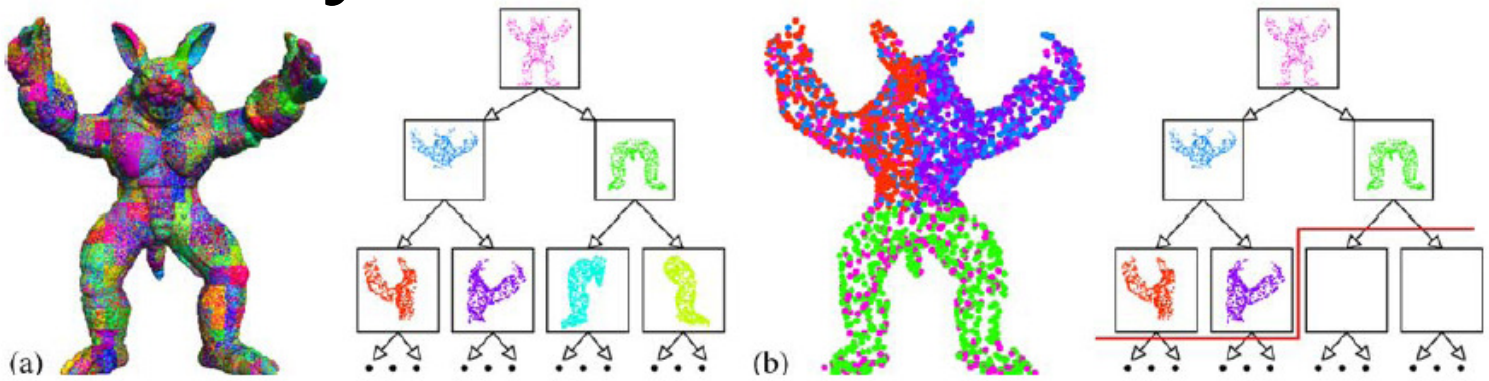


Fig. 1. Layered point cloud structure. The multiresolution model has exactly the same points of the input model, but grouped into constant size chunks and organized in a level of detail representation. Variable resolution representations of the models are obtained by defining a cut of the hierarchy and merging all nodes above the cut. (a) Full structure, and (b) adaptive resolution.

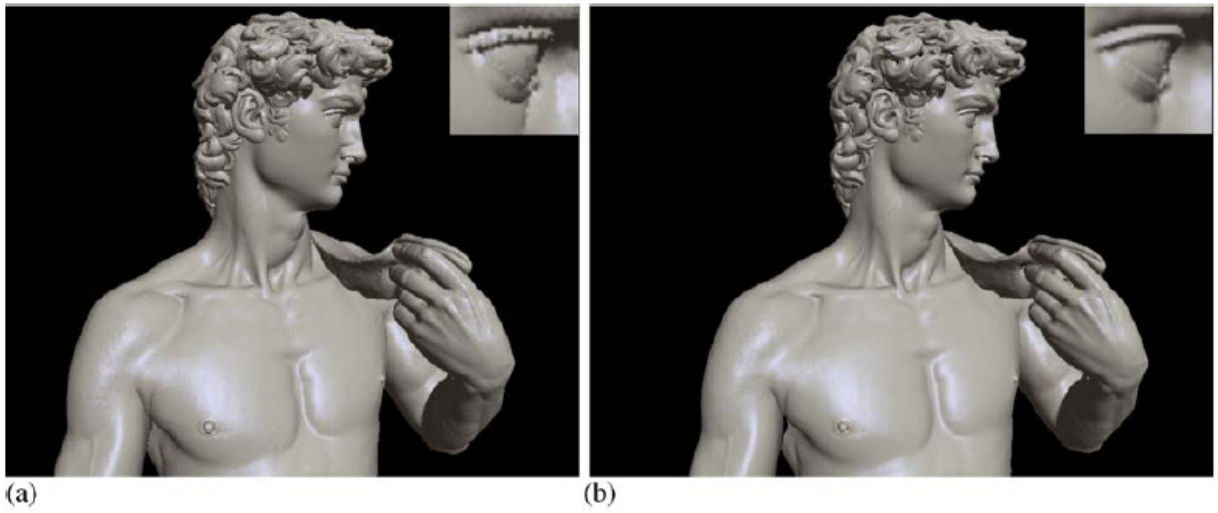


Fig. 8. Rendering quality. Renderer tolerances configured to generate approximately the same number of primitives. Inset images show a detail with a 4× magnification. Even though the quality of the point based version does not match that of the triangle based one, it appears sufficient for most interactive display applications. (a) Layered point clouds (1345 K & splats), and (b) adaptive tetrapuzzles (1421 K triangles).

Representações baseadas em amostras

- Estão no extremo oposto às representações de maior ordem: exploram métodos discretos para a representação de modelos através de conjuntos de amostras
- Exemplos:
  - Point-based rendering
  - Representações volumétricas
  - Sample-based LOD

Representações Volumétricas

- Usadas não apenas para renderização, mas também para primitivas para a geração de LODs
- “Far voxels: a multiresolution framework for interactive rendering of huge complex 3D models on commodity graphics platforms”. E. Gobbetti, F. Marton. ACM Transactions on Graphics, 24(3): 878-885, Jul. 2005.

Far Voxels: Ideia

- Hierarquia de LODs, onde as folhas representam a geometria original (triângulos) e cada nó pai contem uma representação simplificada de seus filhos (voxels)

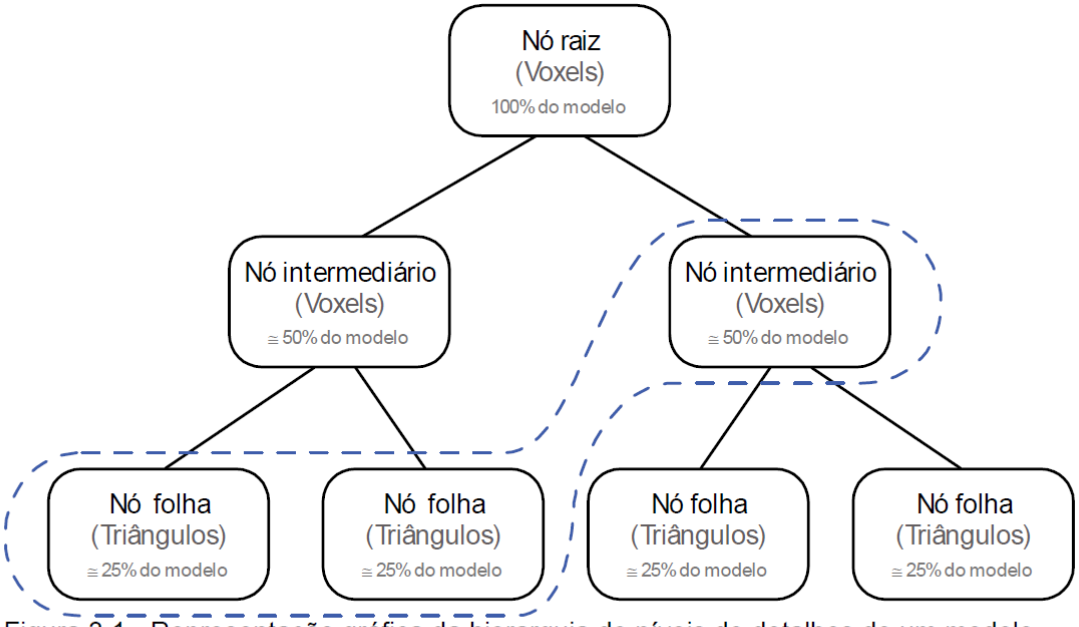


Figura 3.1 - Representação gráfica da hierarquia de níveis de detalhes de um modelo. A linha pontilhada mostra uma possível configuração de nós escolhidos para gerar uma visualização do modelo

No trabalho do Gustavo usamos octree

Tecgraf PUC-RIO VR GROUP

Usando voxels distantes para representar a geometria (1)

- Ideia básica:
  - Da mesma forma que uma imagem é usada como um painel (*billboard*) numa cena, uma grade de voxels pode ser usada para melhor representar um modelo geométrico 3D que pode ser visto de diversas direções.
- Dada uma caixa envolvente do objeto a ser representado, os voxels são organizados em uma grade regular alinhada com a caixa.

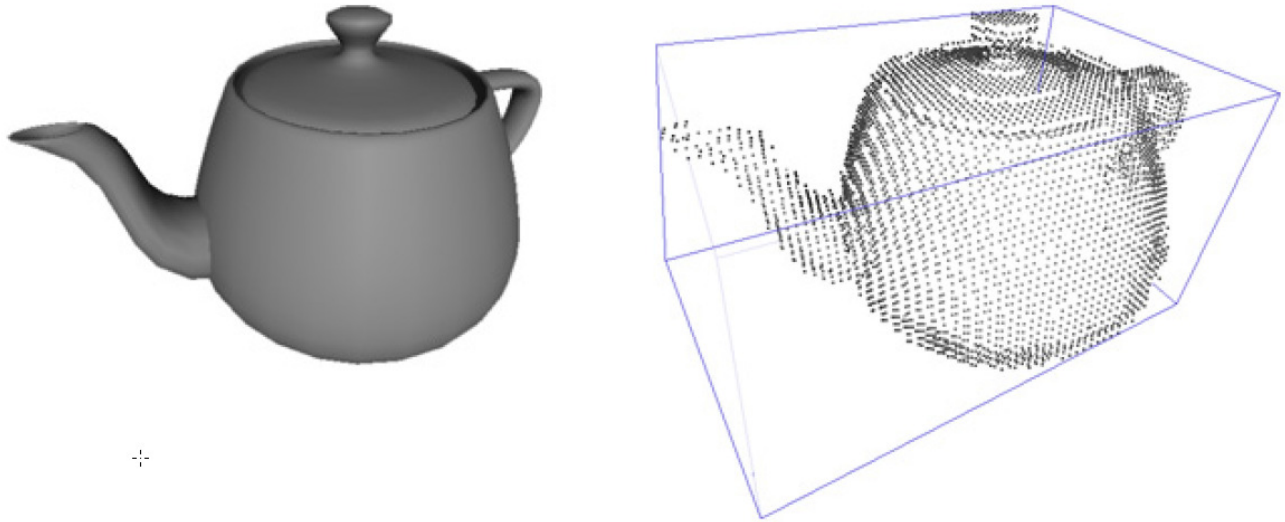
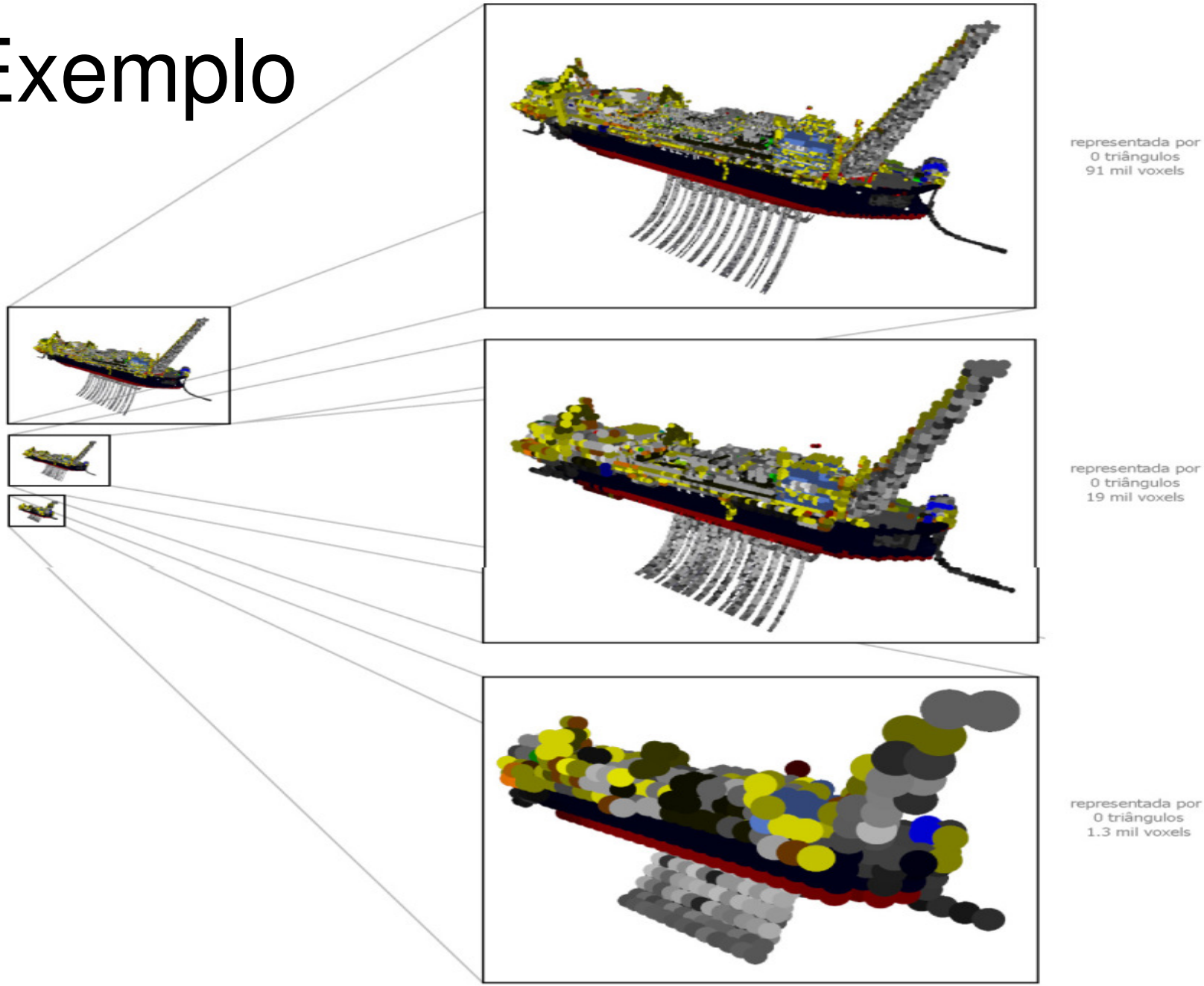


Figura 3.2 - Caixa envolvente e representação de voxel. Os voxels foram representados como pequenos pontos para notarmos sua organização.

Usando voxels distantes para representar a geometria (2)

- Dentro de cada voxel podem existir zero, um ou mais triângulos representando um ou vários materiais
  - Quando este voxel se projeta em um ou poucos pixels o valor da radiância capturada pela câmera sintética do OpenGL seja a mesma deste conjunto de triângulos.
  - Para isto as propriedades ópticas do voxel precisam ser escolhidas de forma correta.
  - Para facilitar esta escolha, idealmente um voxel deve se projetar em um ou poucos pixels.
- Balanço necessário:
  - Quanto menor o voxel maior a resolução necessária e o custo de renderização da representação volumétrica.
  - Quanto maior o voxel, maior é a distância a partir da qual os voxels formam uma representação adequada.

Exemplo


representada por
0 triângulos
91 mil voxels
representada por
0 triângulos
19 mil voxels
representada por
0 triângulos
1.3 mil voxels

Usando voxels distantes para representar a geometria (3)

- Como direção de visualização não é fixa, para obter uma representação fiel ao modelo original quando visto de longe, cada voxel pode assumir uma dentre diversas representações possíveis.
- É necessário que cada voxel possa assumir cores e normais diferentes de acordo com a direção de onde é visualizado e a configuração das luzes da cena.

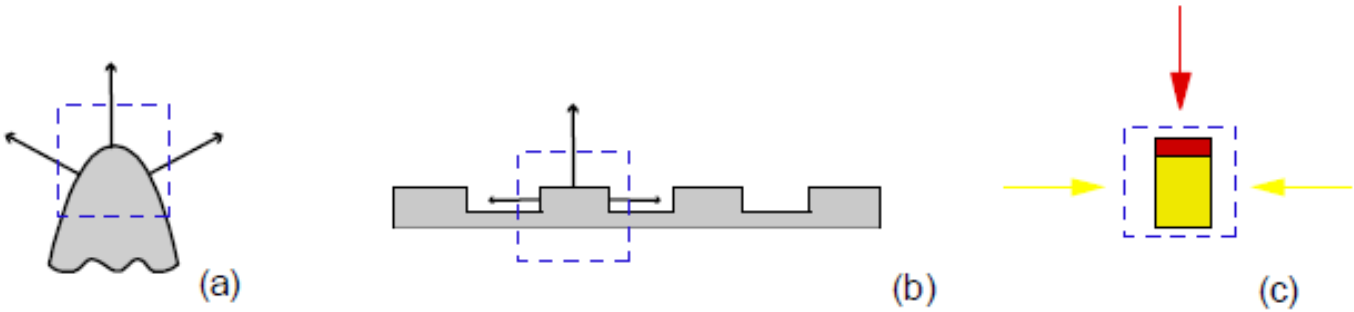


Figura 3.5 - Exemplos de voxels com atributos que variam com a direção de visualização: (a) e (b) ilustram normais variando de acordo com a posição de observação; (c) ilustra a variação de cor.

Usando voxels distantes para representar a geometria (4)

- Voxel armazena propriedades independentes de:
  - cor, propriedades especulares e normal para cada direção associada às 3 principais direções de visualização (±x, ±y, ±z)
- Quando a câmera está alinhada exatamente com um dos eixos, a cor final resultante será o resultado do cálculo de iluminação usando apenas os parâmetros associados àquela direção.
  - Em direções intermediárias, usamos interpolação.

Pré-processamento: geração dos voxels (1)

- Divisão da cena em octree adaptada

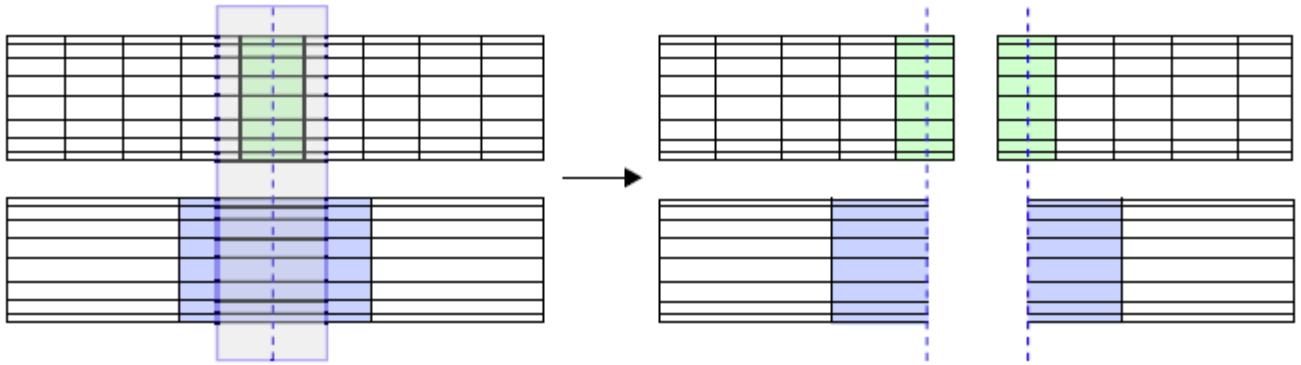


Figura 3.8 - Limite de tolerância (representado pelo retângulo semi-transparente) que determina quando as faces deverão ser duplicadas, caso do objeto de cima ou cortadas, caso do objeto de baixo.

- Nós são divididos recursivamente até que o nó resultante tenha um número máximo de faces pré-definido

Pré-processamento: geração dos voxels (2)

- Gerada a subdivisão hierárquica da cena, temos que preencher todos os nós intermediários com representações simplificadas da região que cada um deles ocupa.
  - A aparência de cada voxel é gerada a partir de um algoritmo de traçado de raios que coleta amostras da cor dos materiais das superfícies visíveis na geometria do modelo.
    - Raios são traçados a partir de uma grande quantidade de possíveis direções de visualização (1 milhão de raios por grupo de voxels).
      - Todas as superfícies atingidas por raios são consideradas visíveis e irão contribuir para a cor final do voxel correspondente ao ponto em que houve a interseção.
      - Superfícies que não forem atingidas poderão ser ignoradas.

Pré-processamento: geração dos voxels (3)

- Algoritmo de voxelização
  - Calcular grade 3D de voxels **V** que represente com a maior fidelidade possível a geometria da região correspondente no modelo.
  - Raios são traçados a partir de uma superfície **S**, afastada da grade **V** de uma distância definida do seguinte modo:
    - Dado que o tamanho de um voxel em unidades do modelo seja **t**, e assumindo que essa grade de voxel será usada para a visualização apenas quando seus voxels tiverem no máximo um pixel de tamanho quando forem projetados na tela.

Pré-processamento: geração dos voxels (4)

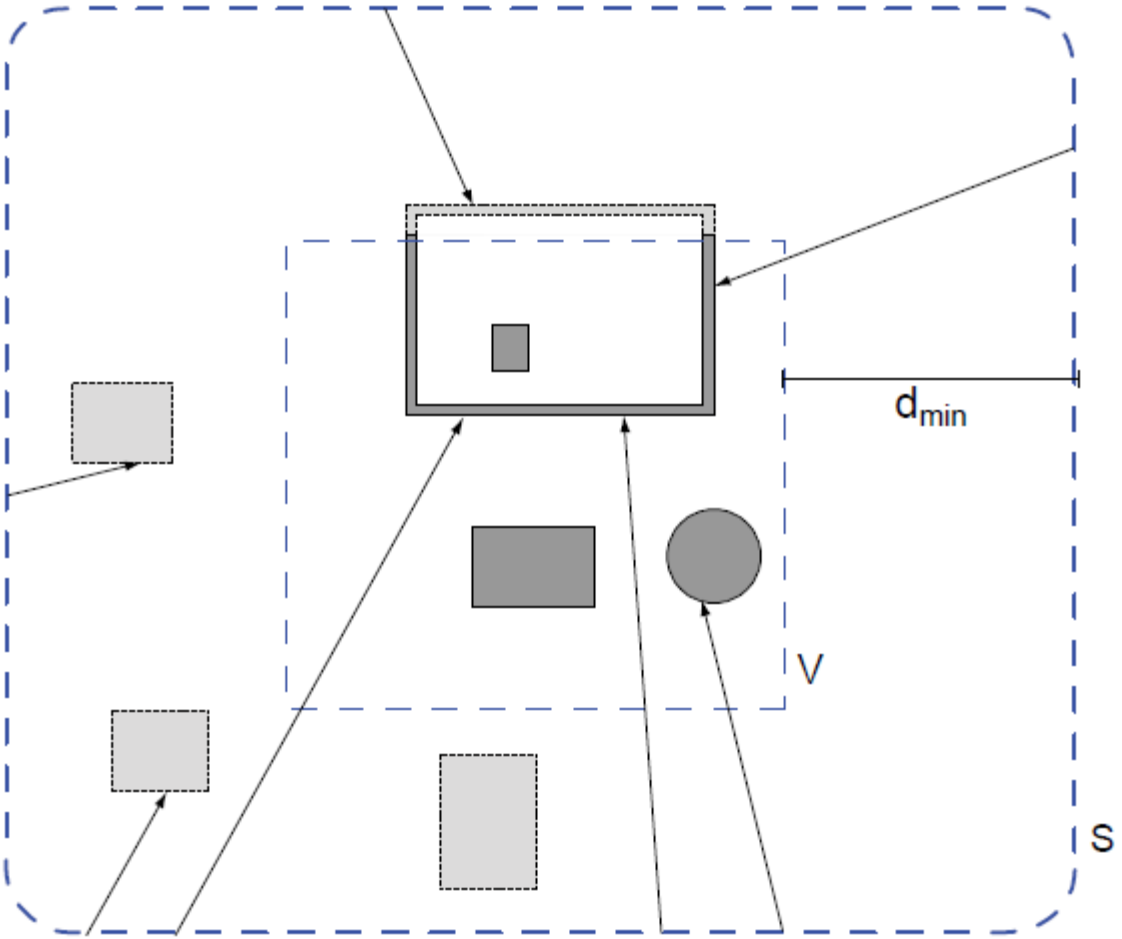


Figura 3.10 – Exemplo de configuração de um traçado de raios típico.

Pré-processamento: geração dos voxels (5)

- Terminado o traçado de raios
  - Caso não haja interseção com um determinado voxel, podemos assumir que este não contém nenhuma geometria ou que qualquer geometria contida nele nunca é visível quando a câmera está fora da superfície envolvente **S**.
    - Não precisa ser representado durante a visualização.
  - Caso haja apenas 1 interseção com o voxel
    - Cor difusa e normal da intersecção atribuídas ao voxel
  - Caso haja mais de 1 interseção
    - Voxel simples, se normais forem próximas
    - Voxel mais complexo, com três direções de normais e cores

Visualização dos Far Voxels

- Para cada nó percorrido da hierarquia da cena, o visualizador deve decidir se a resolução do nível de detalhe existente nele é apropriada para ser usada naquele momento ou se é necessário continuar descendo na hierarquia para escolher nós mais detalhados. .
  - Decisão é baseada num fator definido pelo usuário, que indica o tamanho aceitável que um voxel pode ter na tela.
    - Melhor visualização é obtida quando esse fator é 1: cada voxel deve corresponder a aproximadamente um pixel na tela.
- Visualizador também faz visibility culling convencional

Problemas dos Far Voxels

- Pré-processamento demorado
- Problemas com transparência e alias, especialmente para objetos muito finos

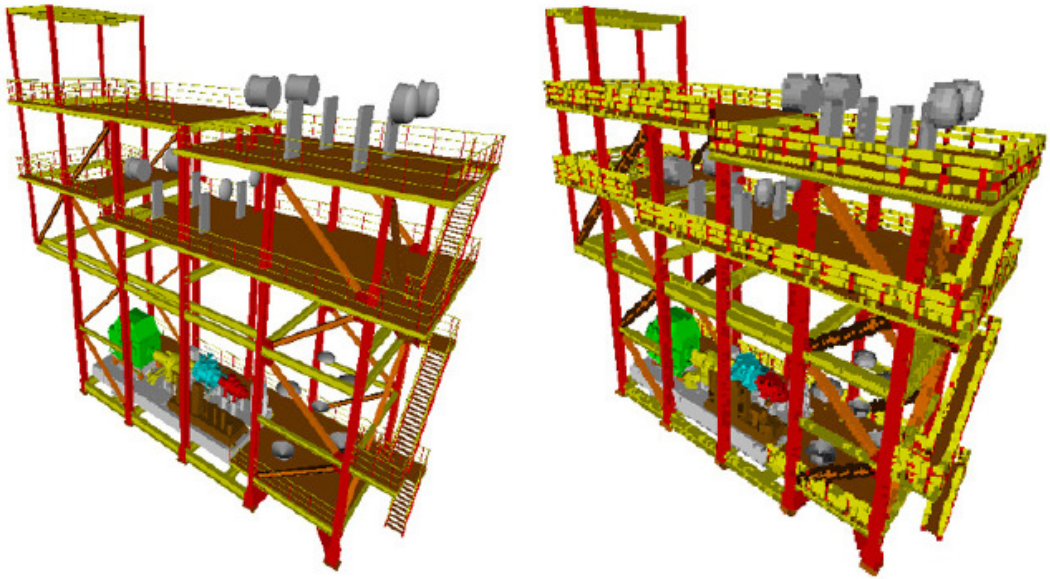


Figura 3.11 – Falhas visuais geradas por objetos muito finos sendo representados por voxels. Na figura da esquerda, representada totalmente por geometria, podemos ver que os corrimões são bem mais finos que o tamanho escolhido para os voxels, na figura da direita. Nessa figura, os voxels foram ampliados para ilustrar a falha.

Raposo - 2010

Tecgraf PUC-RIO VR GROUP

Far Voxels com Filtro Anti-Serrilhamento (1)

- Contribuição do trabalho do Gustavo
  - Além de atribuirmos cores diferentes para o voxel, podemos atribuir transparências diferentes para cada uma das seis principais direções de visualização
  - Para determinar grau de transparência do voxel
    - Usar a informação sobre quantos raios passaram através de cada voxel sem atingir nenhuma geometria.
      - Quando um voxel é transpassado por muitos raios traçados a partir de uma determinada direção, podemos assumir com segurança que a geometria contida naquele voxel ocupa apenas uma pequena área do voxel, quando este é visto a partir daquele ângulo.
      - Fator de transparência proporcional à razão entre a quantidade de raios que transpassou o voxel e a quantidade de raios que atingiu alguma superfície dentro do voxel.

Far Voxels com Filtro Anti-Serrilhamento (2)

- Problema que precisa ser tratado

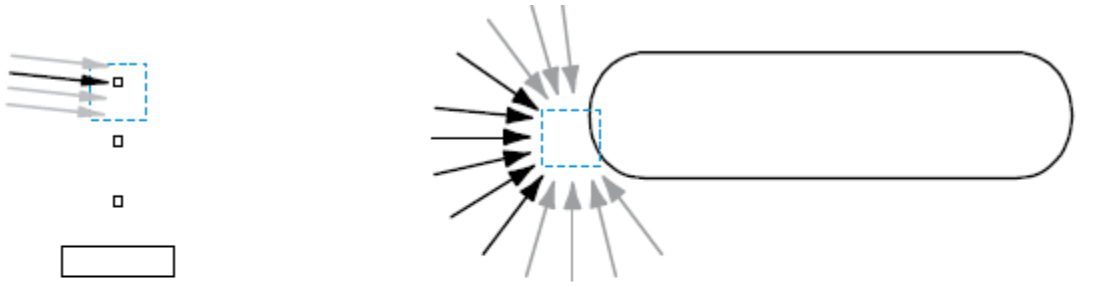


Figura 3.16 – Casos comuns de voxels transparentes encontrados durante o traçado de raios. No caso (a), o voxel poderá ser representado por um único fator de transparência, já que sua visibilidade é semelhante a partir de todas as direções. No caso (b), aplicar transparência nas direções marcadas em preto permitiria ver através do objeto.

Resultados Gustavo (1)

- P-50

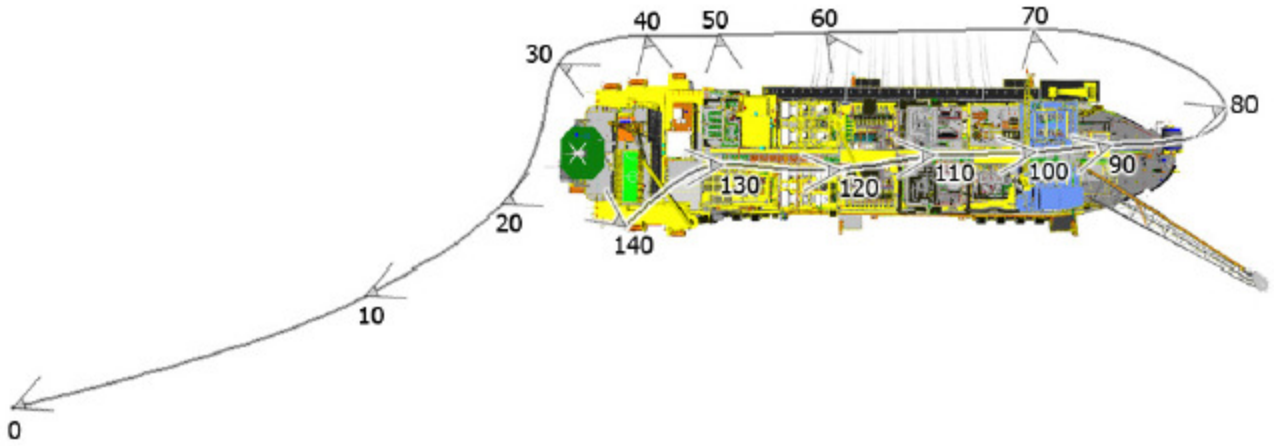
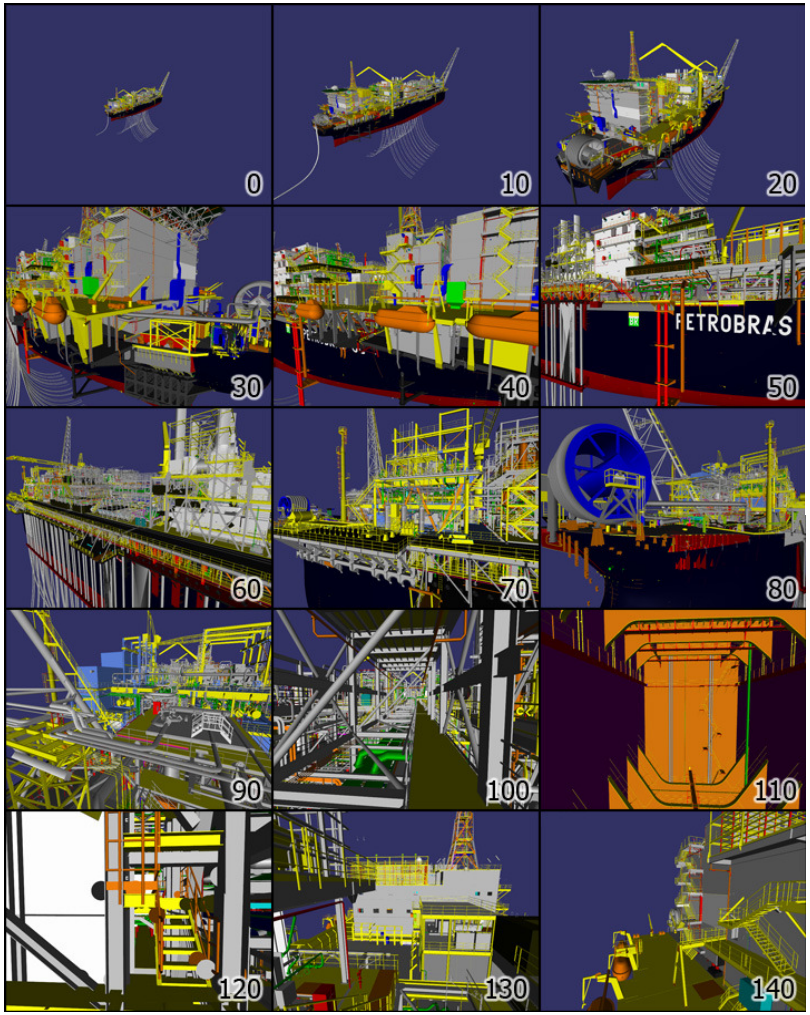
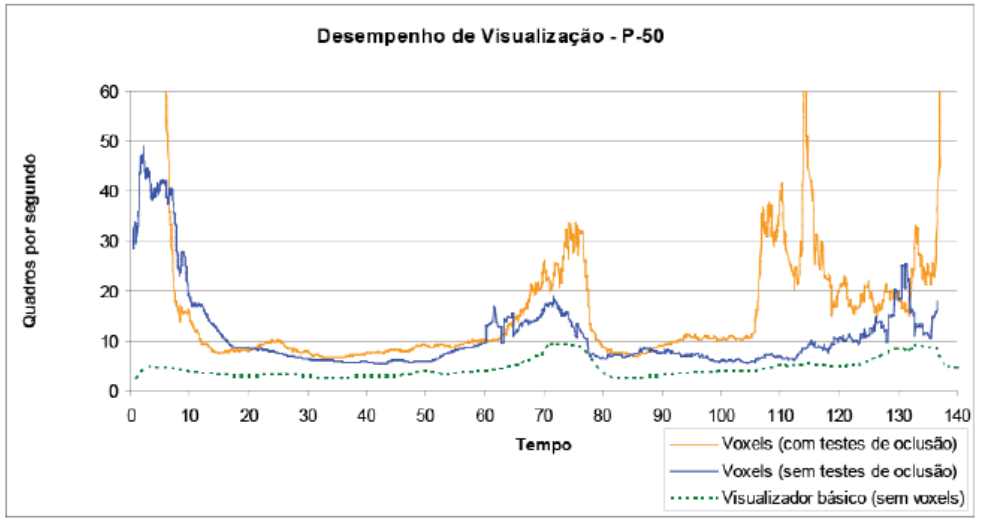
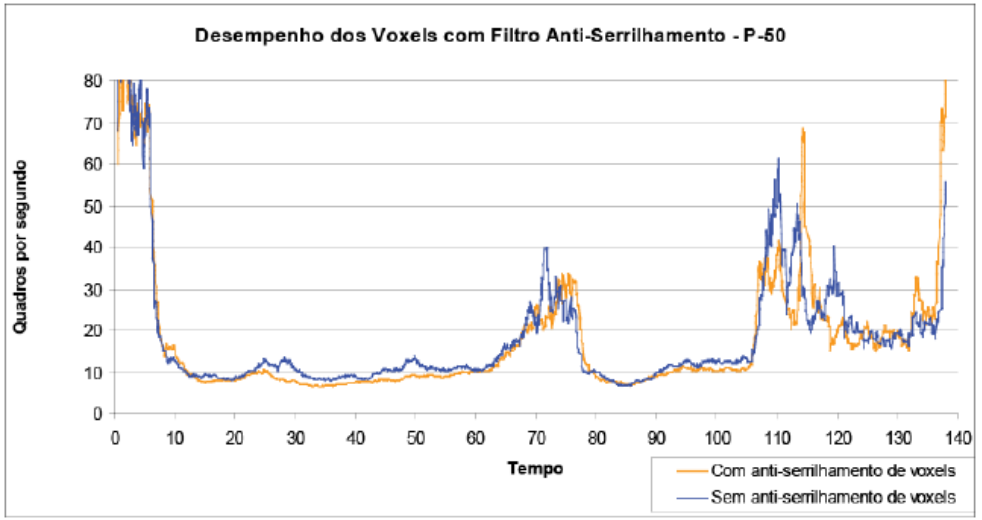


Figura 5.14 - Caminho percorrido durante o teste de desempenho na P-50. Os números indicam onde a câmera está em cada instante da animação

Resultados Gustavo (1)







Representações baseadas em amostras

- Estão no extremo oposto às representações de maior ordem: exploram métodos discretos para a representação de modelos através de conjuntos de amostras
- Exemplos:
  - Point-based rendering
  - Representações volumétricas
  - Sample-based LOD

Sample-based LOD for Ray Tracing (1)

- Soluções baseadas em traçado de raios têm três grandes vantagens sobre outros métodos de renderização:
  - Visibilidade de superfícies pode ser determinada em tempo logarítmico sobre o tamanho da cena, desde que o modelo seja estruturado de forma adequada.
  - Traçado de raios é altamente paralelizável, bastando que cada pixel a ser renderizado seja atribuído a um processador diferente.
  - Cálculo de sombras e reflexões pode ser feito com custo não muito maior que o da própria renderização.

Sample-based LOD for Ray Tracing (2)

- Problema: reduzir tamanho do “working set” (conjunto de dados que precisam estar alocados na memória principal para permitir a renderização de um quadro)
- Solução proposta: R-LODs
  - “R-LODs: fast LOD-based ray tracing of massive models”. S. Yoon, C. Lauterbach, D. Manocha. The Visual Computer 22(9-11): 772-784, 2006.

R-LODs

- Planos que contêm atributos de material, como as cores da geometria substituída.
- Usados para substituir grupos de triângulos inteiros nos nós internos da KD-Tree usada para o Ray Tracing
- Ganhos de 2 a 20 vezes com pouca perda de qualidade visual

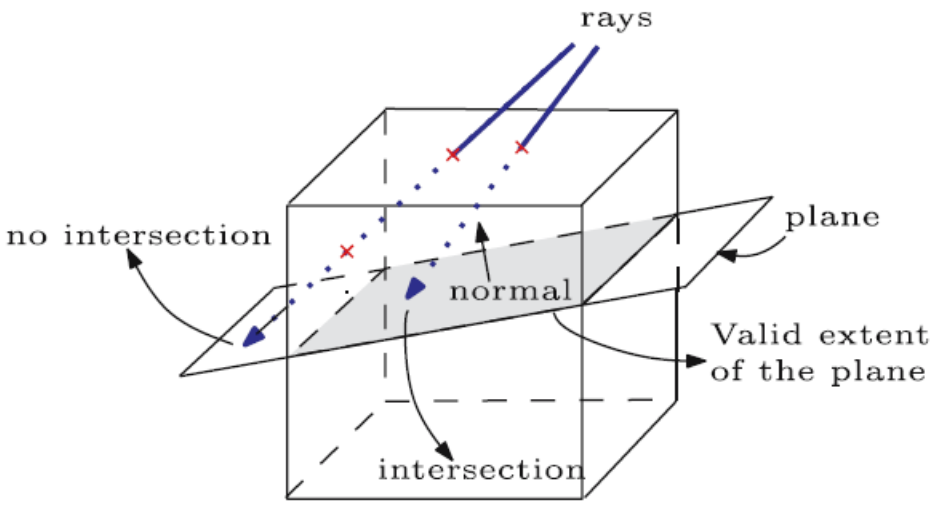


Fig. 3. LOD representation. A R-LOD consists of a plane with material attributes. It serves as a drastic simplification of triangle primitives contained in the bounding box of the subtree of a kd-tree node. Its extent is implicitly given by its containing kd-node. The plane representation makes the intersection between a ray and a R-LOD very efficient and results in a compact representation

R-LODs - Resultado

Fig. 9. Forest model. We render the forest model consisting of 32 million triangles with shadow rays using PoE $= 0$ and PoE $= 4$. The image resolutions are 512×512 without anti-aliasing to highlight image quality differences. We are able to render the model given the viewpoint at the 1.6 frames per second and achieve 5 times improvement by using R-LODs

Outras Representações

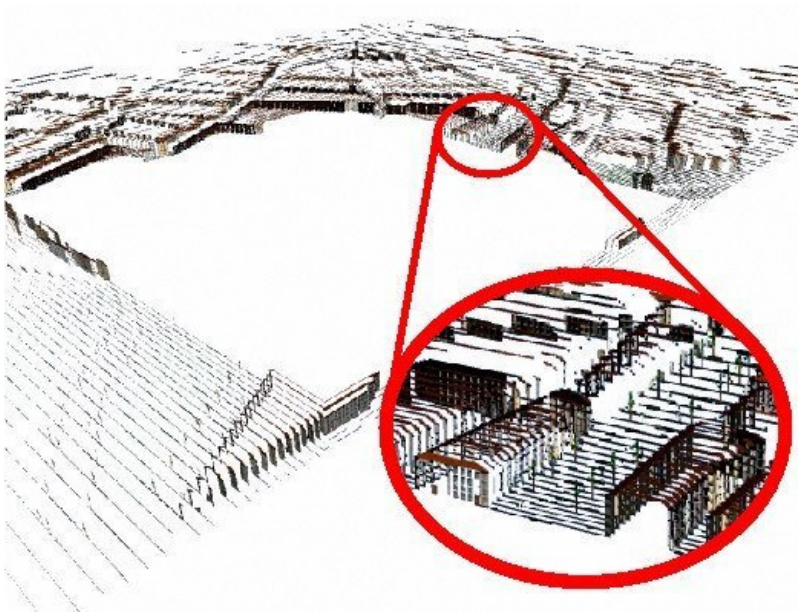
- 3 abordagens
  - Higher order representations
    - Ex., Modelos CAD podem ser bem representados por primitivas de maior ordem: cúbicas, quádricas, etc.
    - Vantagens: reduz uso de memória e gera superfícies “smooth” mesmo com visão bem próxima
  - Sample-based representations
    - Em representações complexas, polígono pode representar menos que um pixel na imagem final
      - Representações tipo voxels, pontos, etc, podem tirar vantagem disso
  - Image-Based representations

Image-Based Representations

Alberto Raposo - 2010

Impostores (1)

- Objetos distantes deixam de ser malhas poligonais para serem substituídos por uma representação baseada em imagem



Impostores (2)

- Billboard: polígono plano texturizado cuja orientação sempre se volta para o viewer.
  - Bom para substituir objetos cilíndricos, tipo árvores
- Texturas de portais: em caso de cenas com portais, se observador não está muito perto de um portal, e/ou se este portal não é muito grande, pode-se colocar textura na abertura

Impostores: Problemas

- Para todos os exemplos anteriores, a textura provê ponto de vista correto apenas do local onde ela foi amostrada
  - Gera artefatos quando o usuário se move
- Soluções
  - Textured depth meshes
    - Texturas são triangularizadas e um valor de profundidade é associado a cada vértice
  - Layered Depth Images
    - Guarda todas as interseções do raio de visão com a cena para cada pixel

BlockMaps (1)

- "Ray-Casted BlockMaps for Large Urban Models Visualization". P. Cignoni, M. Di Benedetto, F. Ganovelli, E. Gobbetti, et al. Computer Graphics forum (Proc. Eurographics 2007), 26(3): 405-413.

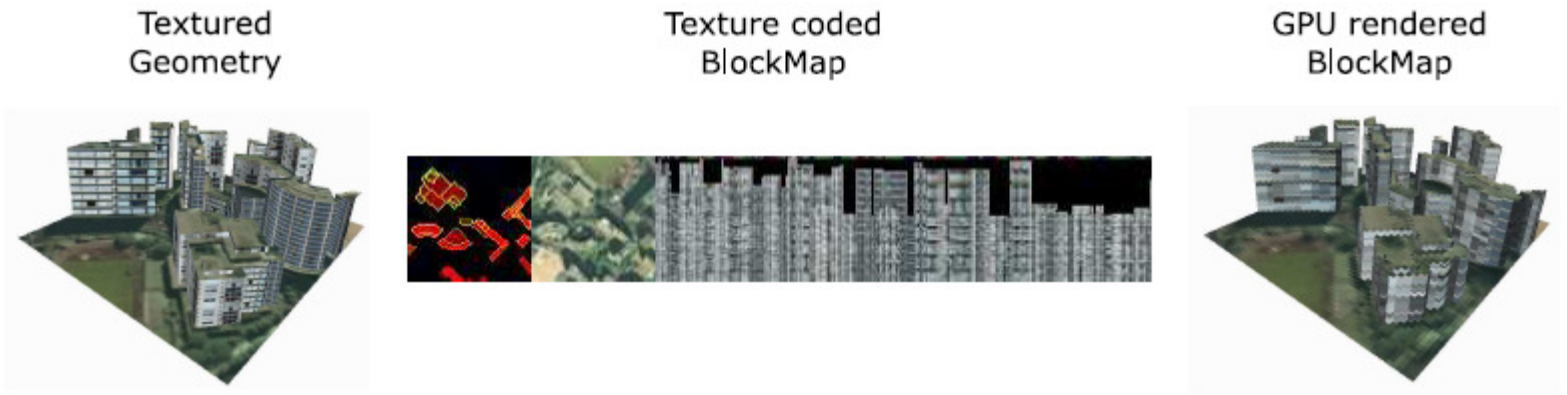


Figure 1: *The BlockMaps idea: replace far away groups of buildings with a concise texture-based representation, called* ***BlockMap****, which is efficiently rendered by a GPU ray caster.*

Tecgraf PUC-RIO VR GROUP

BlockMaps (2)

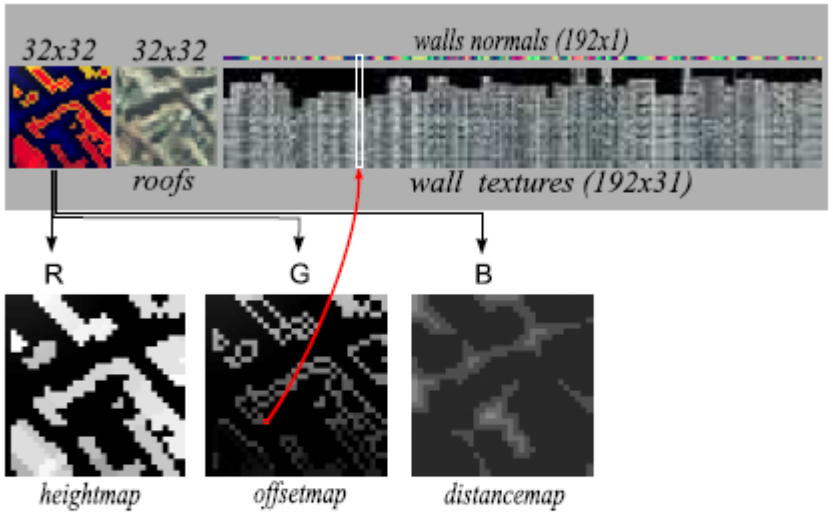


Figure 2: *In the upper part a the 256 × 32 texture coding a 32:192 BlockMap is shown. The bottom part shows the three channels of the first quad, named* geometry, *of the BlockMap. The red arrow illustrate the texture 2 fetches executed if a vertical wall is hit.*

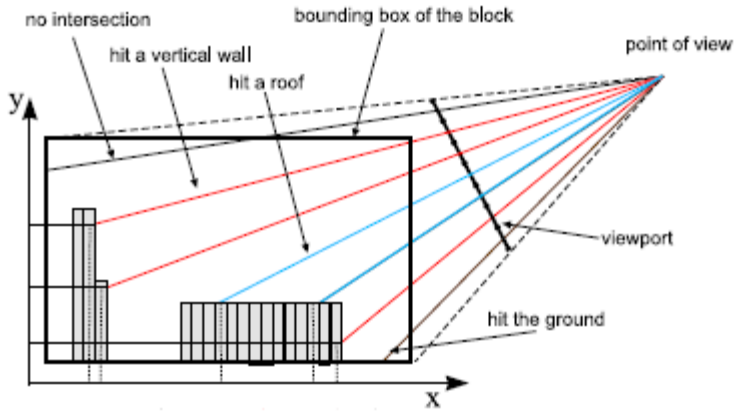


Figure 3: *A scheme of the ray casting process. For each ray hitting a surface, the BlockMap provides access to normal and color to use for rendering.*

BlockMaps (3)

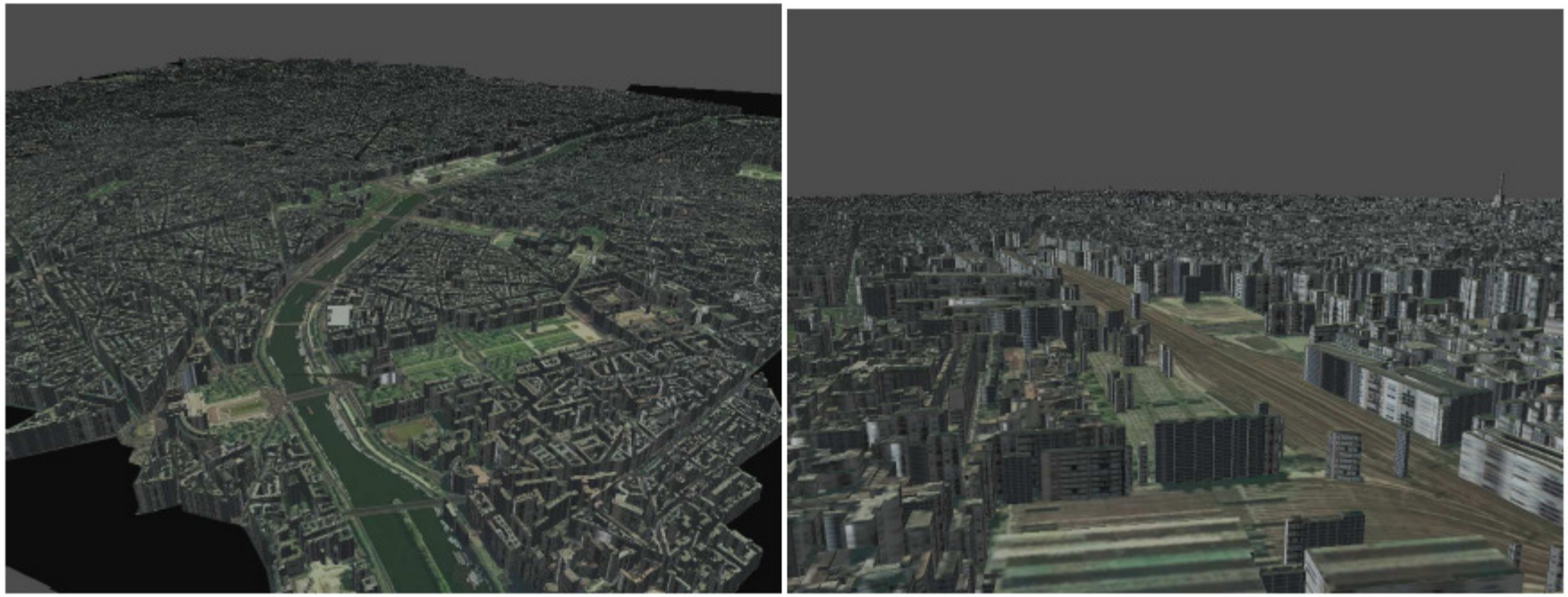


Figure 8: *Two snapshots from the flight over Paris video. The far away buildings are efficiently rendered using the BlockMaps approach.*

Tecgraf PUC-RIO VR GROUP

Conclusão

- As diferentes soluções desenvolvidas têm vantagens e desvantagens, e não há uma representação que seja simplesmente melhor em todos os aspectos (custos de memória, desempenho, implementação, etc.)
  - A solução mais apropriada é dependente da cena e da aplicação em questão

Conclusão

- Malhas de triângulos provêem solução “razoável”para ampla gama de situações
  - São muitas vezes a única forma de representação do sistema
- Representações por pontos e voxels também podem ser consideradas genéricas o suficiente para serem única forma de representação
  - Mas são geralmente usadas em níveis de detalhes intermediários, tendo os polígonos como nível de representação mais refinado
- Representações de maior ordem ainda são “promessas”