Proposta de Dissertação de Mestrado

Aplicação de Tags em Objetos de Modelos Massivos com Visualização em Tempo Real

Renato Deris Prado

agosto 2012

Introdução

- Softwares de gerência de projetos de engenharia, setor de óleo & gás...

  Objetivos:

  Em adição a possuírem bancos de dados com informações do projeto, devem ser interativos.

  Fornecer recursos para visualização 3D de seus modelos com um nível suficiente de realismo.

- Modelos CAD...

  CAD é a abreviação para Computer Aided Design.

  São modelos de engenharia para visualização científica (prédios, veículos, entre outros).

  Podem ser muito complexos como de uma refinaria de petróleo.

- Tags!

  Cada objeto em um modelo CAD possui um nome.

  Uma tag é o nome de um objeto, exibido em sua superfície.

Figura1. Caixa com versão simples de tag. [Prado et al]

Introdução

- Motivação

Em estruturas reais existem rótulos fixados nas superfícies dos objetos (ver figura 2).

Isso facilita a identificação deles e ajuda as pessoas a se orientarem quando trabalhando no campo.

Tags facilitarão a identificação dos objetos e orientação dos usuários na aplicação virtual.

O Aveva Review faz o uso de tags mas detalhes da implementação não são conhecidos (ver figura 3)

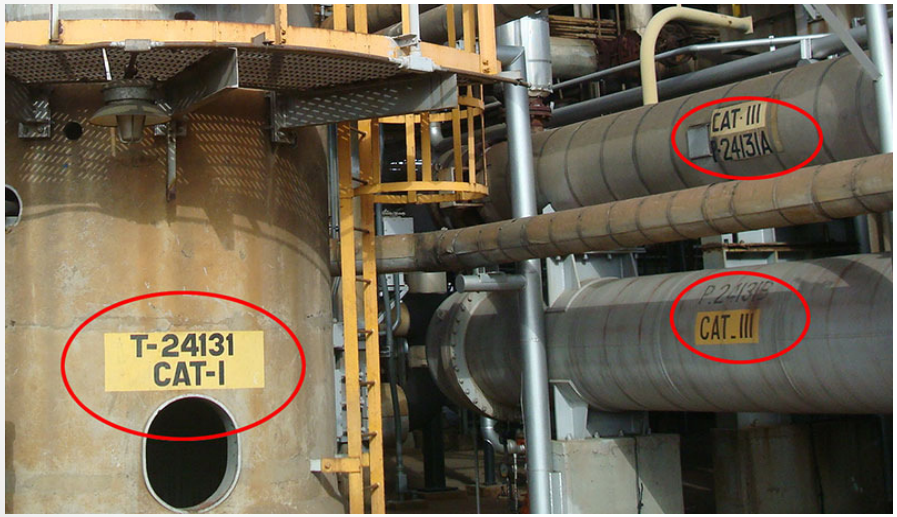

Figura 2. Foto de uma refinaria da Petrobras mostrando objetos com rótulos.

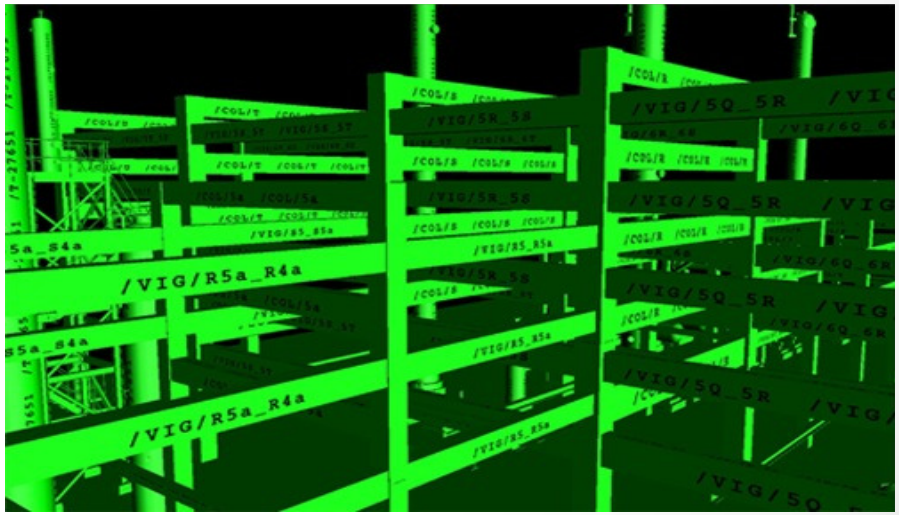

Figura 3. Screenshot do software Aveva Review, de um modelo com tags.

Problemas

- Gerência de memória RAM e VRAM

  Os modelos CAD podem ter milhões de objetos geométricos.

  Cada objeto possui um nome diferente.

  Se cada objeto possuir uma textura diferente a quantidade de memória gasta será inviável.

  A memória poderá esgotar e paginação poderá ocorrer (problema de desempenho).

  É preciso haver um gerenciamento de memória!

- Desempenho

  Um número muito grande de objetos pode estar visível na cena ao mesmo tempo.

  "Texturizar" todos eles irá gerar muitas trocas de contexto de textura por quadro.

  Muitas trocas de contexto de textura prejudicarão o desempenho da aplicação [Prado et al.]

  Exibir modelos CAD por si só já é um problema para o desempenho.

  As tags não podem causar uma queda de FPS grande (as taxas devem permanecer "iterativas").

Problemas

- Posicionamento das tags

  Os objetos possuem formas e tamanhos diferentes, existem pirâmides, cones, caixas, entre outros.

  Como mapear o texto sobre essas diferentes superfícies e obter um bom efeito visual?

  De qualquer ângulo que um objeto estiver sendo visualizado é preciso que sua tag esteja aparente.

  É interessante que a orientação das tags esteja sempre correta quando um modelo é carregado.

- Aliasing

  O usuário pode se aproximar ou se afastar de um objeto. Isso pode causar *aliasing* nas tags.

  Esses problemas são conhecidos como "texture magnification" e "texture minification".

  Os artigos encontrados falam mais sobre magnificação de texto [Green][Lefebvre et al.].

  Em geral usam uma textura de baixa resolução e técnicas para reduzir o *aliasing* na magnificação.

  Quando o problema de "texture minification" poderia ser mais grave, no caso do visualizador 3D...

  os objetos seriam muito pequenos e não precisariam necessariamente de tags.

Estado da Arte

- Aveva Review [Aveva Review 2010]

  Exibe tags em cilindros e caixas apenas.

  Limita número de objetos que recebem tags por quadro.

  Implementação não conhecida.

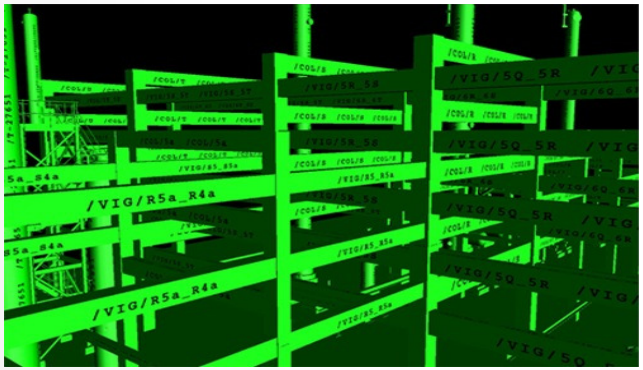

Figura 3. Screenshot do software Aveva Review

- Autotag [Prado et al.]

  Cada objeto possui uma textura (cilindros e caixas apenas).

  Não há gerenciamento de memória. É preciso haver algum.

  Limita objetos que recebem tags por quadro. Escolhe um número arbitrário de objetos mais próximos da câmera. Seria melhor objetos maiores na tela.

Figura 4. Objetos com tags. [Prado et al]

Estado da Arte

- Texture Sprites [Lefebvre et al.]

  Pequenas texturas (*sprites*) que juntas podem formar uma maior.

  Utiliza estrutura similar a *octree* que pode ser mapeada em uma textura 3D em GPU.

  Octree guarda informações de cada *sprite* como posição, tamanho e id da textura.

  Não fala sobre modelos massivos, mostram um objeto de cada vez.

  Posicionamento dos *sprites* não é automático.

Figura 6. "Bunnys" com sprites [Lefebvre et al.]

- Real-Time Texture-Mapped Vector Glyphs [Qin et al.]

  Texto é representado de forma vetorial para reduzir aliasing quando magnificado.

  Utiliza estratégia similar a de Texture Sprites para posicionar palavras sobre objetos 3D.

  Atlas com letras é amostrado diretamente em GPU

  Mesmas questões de "Texture Sprites"

Figura7. "Bandeira com texto [Qin et al.]

Estado da Arte

- Text Scaffolds [Cipriano et al.]

  Posicionamento de texto em superfícies de curvatura acentuada e até com buracos

  Utiliza andaimes para guardar as letras

  Posiciona as letras flutuando sobre os objetos

  Mesmas questões de Texture Sprites.

  Z-fightning.

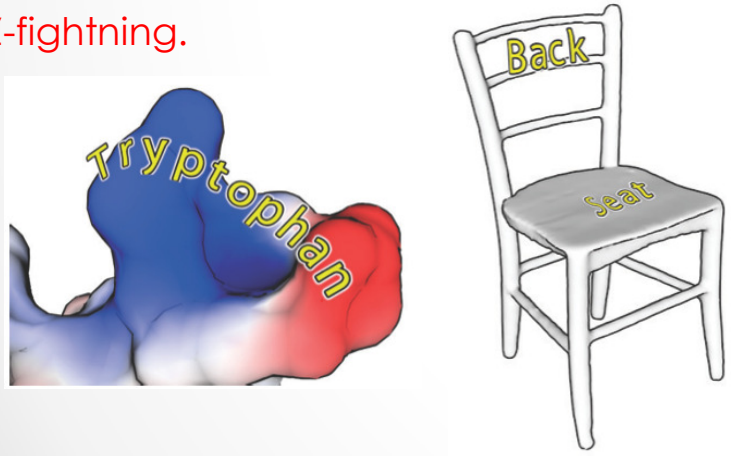

Figura 8. Objetos com *scaffolds* [Cipriano et al.]

- Improved Alpha-Tested Magnification for Textures and Special Effects [Green]

  Renderiza texto com pouco aliasing

  Constrói imagem de baixa resolução a partir de imagem de alta resolução.

  Utiliza o teste de alfa para renderizar a textura.

(a) High resolution input (b) 64x64 Distance field

Figura 9. Imagem de alta resolução e campo de distância [Green]

Estado da Arte

- Sparse Virtual Textures [Barret]

  Armazena textura virtual no HD dividida em "páginas" de mesmo tamanho.

  Solução "*out-of-core*".

  Somente as páginas necessárias pela API gráfica são transferidas para memória.

  Para mapear as páginas virtuais em páginas físicas é construída uma tabela de páginas.

  Resolve o problema de de texturas que não cabem em memória.

  Como organizar a textura virtual com os nomes dos objetos?

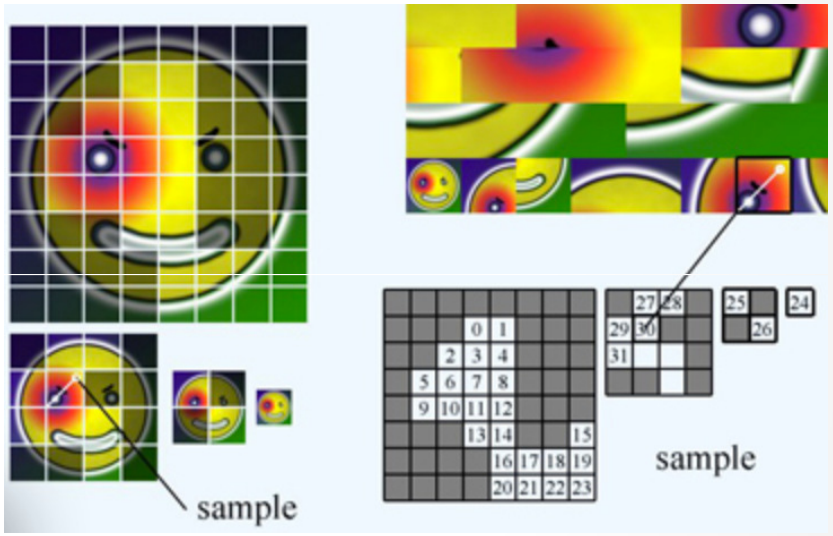

Figura 10. Na esquerda texturas virtuais, na direita acima a textura física e abaixo as tabelas de páginas [Barret]

Proposta

- Escolha do que atacar

  Não há tempo hábil para resolver todos os problemas da melhor forma.

  Os problemas são: gerenciamento de memória, posicionamento de tags, aliasing.

  O objetivo é chegar em uma solução viável de ser utilizada em visualizadores 3D de modelos CAD.

  O gerenciamento de memória pode ser mais simples do que o de Sparse Virtual Textures [Barret].

  O posicionamento de tags em objetos de diferentes tipos será o foco principal.

  O problema de aliasing poderá ser resolvido pela técnica utilizada por [Green].

- Gerenciamento de memória

  Existirá um limite de objetos que exibirão tags por quadro. Objetos escolhidos serão os maiores na tela.

  Será mantido em RAM e em VRAM somente o que for necessário a cada quadro.

  Nomes deverão ser reconstruídos conforme o necessário, em tempo real.

  Espera-se que com isso um número razoável de objetos possa ter tags por quadro...

  Que o desempenho não seja muito afetado...

  E que o usuário possa identificar vários objetos simultaneamente.

Proposta

- Posicionamento das tags

  Posicionar automaticamente tags sobre os objetos de diferentes formas e tamanhos

  - Coordenadas de textura?

  - Abordagem como as de [Lefebvre et al] e [Qin et al], com texturas codificadas em GPU?

  Chegar a uma forma de exibir as tags que permita a identificação do objeto quando ele é visualizado por diversos ângulos.

  - Repetir os nomes nas superfícies dos objetos?.

  Quando um modelo for carregado as tags não devem estar de cabeça para baixo.

- Aliasing

  A técnica de Green parece ser a mais eficaz.

  Porém é algo que pode afetar o desempenho ainda mais. Será viável?

Proposta

- Objetivo final

  Possuir uma solução viável para ajudar os usuários de uma aplicação 3D que visualiza modelos CAD massivos em tempo real.

  Apresentar melhorias em relação ao software Aveva Review e ao artigo Autotag [Prado et al.].

  Apresentar possíveis melhorias ou técnicas novas em relação aos artigos mencionados e a outros possíveis trabalhos que poderão ser encontrados.

O que já foi feito?

- Engine gráfica

  Aplicação capaz de carregar modelos CAD, do tipo RVM (figura 11).

  E gerar cenas para testes (figura 12).

  Utiliza técnica de instanciação de primitivas para gastar menos memória

  Algoritmo para desenhar objetos em ordem de tamanho na tela em construção.

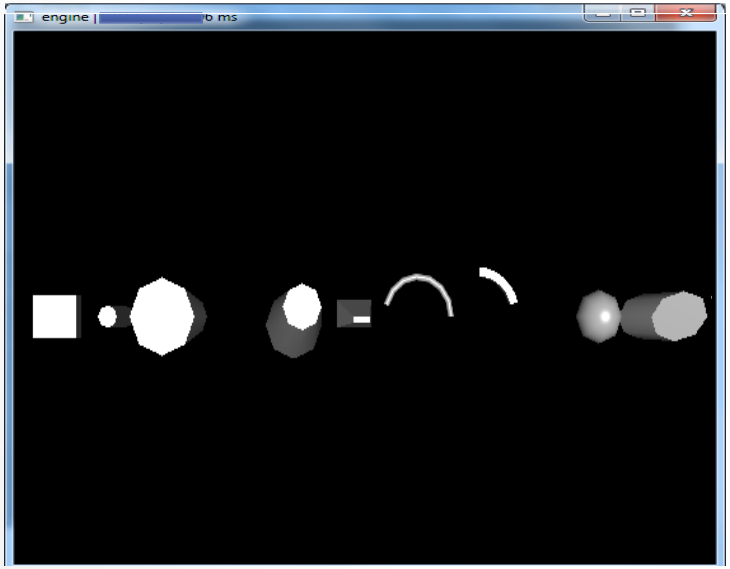

Figura 11. Engine renderizando modelo CAD de teste com todas as primitivas de um RVM.

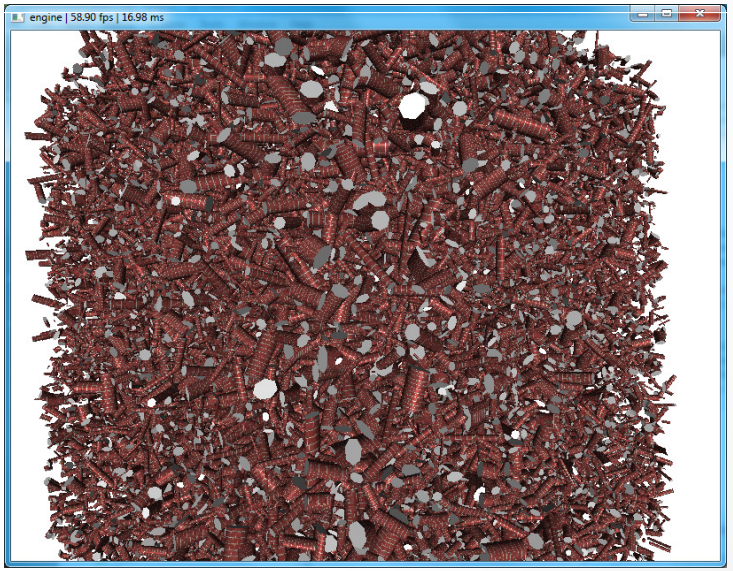

Figura 12. Engine renderizando cilindros com textura.

O que já foi feito?

- Testes base de performance

  (pentium core 2 duo, 2gb RAM, 9600gt)

Teste 1 – Cilindros sem textura

Cilindros	FPS
10000	282
60000	56
100000	34
200000	17

Teste 3 – Cilindros com textura

Cilindros	FPS
10000	236
60000	49
100000	30
200000	16

Teste 2 – Código de textura adicionado

Cilindros	FPS
10000	282
60000	54
100000	33
200000	17

Teste 4 – Cilindros com textura, sem a gerência do GLState.

Cilindros	FPS
10000	201
60000	42
100000	26
200000	13

Tabelas 1 a 4 . Testes de desempenho. A tabela 4 mostra a queda de desempenho quando a API gráfica tenta trocar de contexto de Textura toda vez mesmo a textura sendo sempre a mesma.

- E de memória

  (pentium core 2 duo, 2gb RAM, 9600gt)

cilindros	VRAM total	RAM da aplicação
-	276	-
1	292	10
50	292	10
1000	292	10
10000	292	14
50000	292	22
100000	292	36
500000	292	93
1000000	292	119
2000000	292	215

Tabela 5 . Teste de consumo de memória mostrando que independente do número de objetos a memória de vídeo não aumenta devido a técnica de instanciação.

Referências

- AVEVAREVIEW 2010. Aveva Home Page. Disponível em: http://www.aveva.com/ Acesso em: abr 2010.
- PRADO, R. D., RAPOSO, A., SOARES, L. P. 2011. Autotag: Applying Tags to Virtual Objects in Real-Time Visualization Systems. 8º Workshop de Realidade Virtual e Aumentada - WRVA, 2011, Uberaba. Anais do WRVA'2011 - 8o Workshop de Realidade Virtual e Aumentada (CD-ROM). Porto Alegre: Editora da SBC, 2011. p. 1-6.
- LEFEBVRE, S., HORNUS, S., AND NEYRET, F. 2005. Texture Sprites: Texture Elements Splatted on Surfaces. In ACM Symposium on Interactive 3D Graphics.
- QIN, Z., MCCOOL, M. D., AND KAPLAN, C. S. 2006. Real-time texture-mapped vector glyphs. I3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games, ACM Press, New York, NY, USA, 125–132.
- CIPRIANO, G., GLEICHER, M. 2008. Text scaffolds for effective surface labeling. IEEE Trans. Visualization and Computer Graphics 14(6), 1675–1682 (2008)
- GREEN, C. 2007. Improved alpha-tested magnification for vector textures and special effects. In ACM SIGGRAPH 2007 courses, SECTION: Course 28: Advanced real-time rendering in 3D graphics and games, 9–18.

FIM