Pontifícia **U**niversidade **C**atólica
do Rio de Janeiro

Visualização de terrenos em GPU

Leonardo Martins { lmartins@inf.puc-rio.br}

Disciplina: Visualização de Modelos Massivos
Professor: Alberto Raposo

Sumário

- Introdução
- Objetivos
- Visão geral
- Hierarquia de malhas
  - Balanceamento
- Renderização
- Resultados
- Conclusões/trabalhos futuros
- Referências

Introdução

- Visualizar terrenos grandes de forma eficiente e com qualidade permanece sendo um grande desafio.
  - Imagens de satélite da ordem de bilhões de amostras estão disponíveis
  - A capacidade de processamento e armazenamento das máquinas atuais cresce na mesma proporção da capacidade de resolução dos *scanners* e *displays*
  - Terrenos estão presentes em um grande número de aplicações computacionais, tais como GIS, jogos e simuladores de vôo.
- Técnicas de visualização tais como LoD (Level of Detail) e *culling* ajudam a reduzir o número de primitivas e a aumentar a taxa de exibição de quadros.

Introdução

- Propriedades desejadas em algoritmos de visualização de terrenos (Bösch et al, 2009)
  - Suporte a LoD
  - Renderização de alta-perfomance
  - Exibição contínua
  - Recuperação rápida de dados
  - Armazenamento compacto
  - Acesso direto aos dados
  - Simplicidade
  - Pré-processamento rápido

Objetivo

- Implementar através da GPU um visualizador de terrenos
  - Taxas de exibição aceitáveis
  - Qualidade
  - Gerenciamento de memória
- "GPU-Friendly High-Quality Terrain Rendering"
  - Schneider and Westermann, 2006

Visão geral

- Pre-processamento
  - Divide-se o terreno em *tiles* e para cada um é gerado um conjunto discreto de LoDs através de uma hierarquia de malhas (quad-tree)
- Execução
  - Não é feita nenhuma re-triangulação de malha
  - Reduzida a necessidade de largura de banda (menos triângulos)

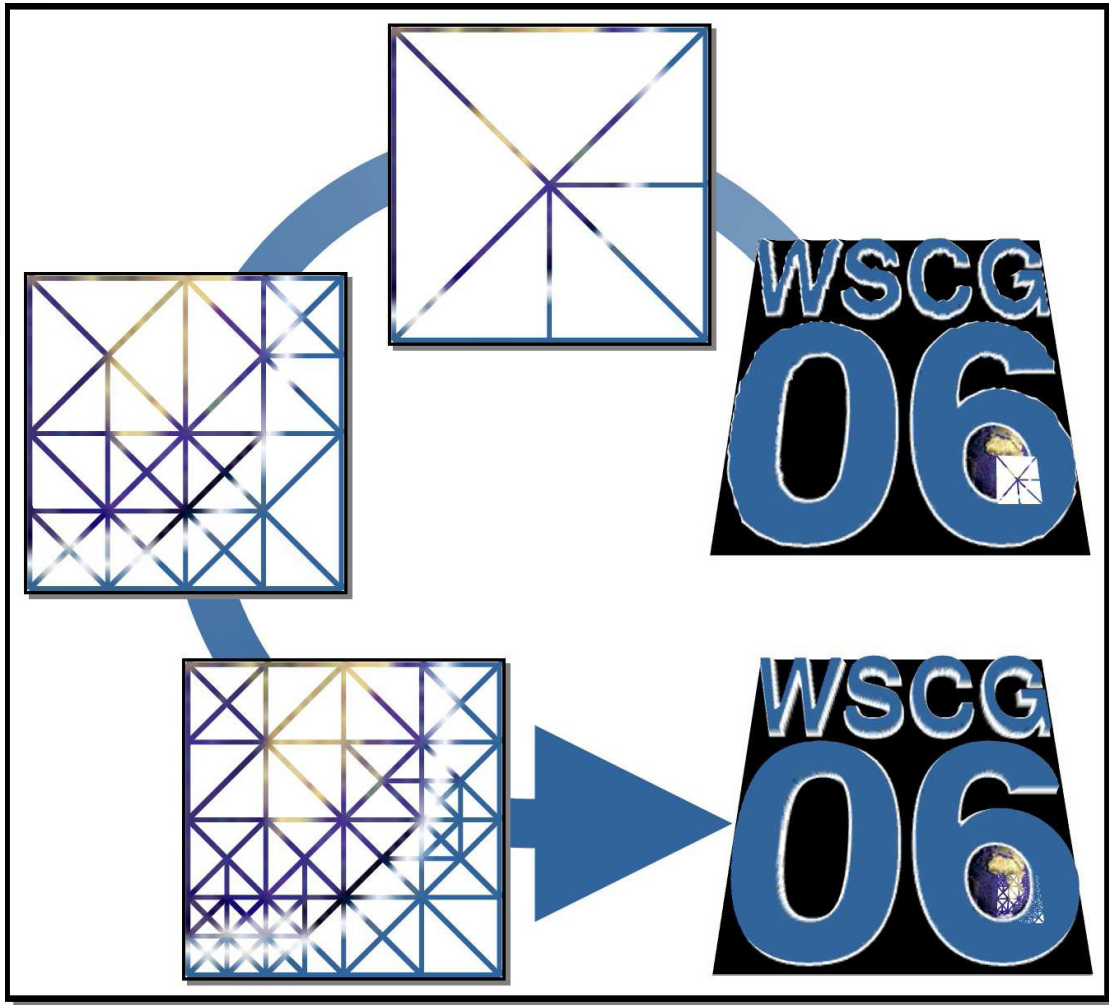
Hierarquia de malhas

- Um dado campo de alturas H: $N^2 \rightarrow Z$ pode ser aproximado por uma malha triangular sobre um domínio 2D.
- A superfície define uma reconstrução H' de H.
- A qualidade da reconstrução pode ser medida através de uma métrica de erro que se estendem por todo o domínio espacial δ: R x R → R
- A hierarquia é dita *aninhada* em relação à triangulação
  - *O triângulo no nível i está inteiramente contido no de nível i+1*
  - Dessa forma, tal hierarquia pode ser inteiramente gerada por uma quadtree

Hierarquia de malhas

- Para cada nó da árvore, verifica-se a necessidade de subdivisão através do seguinte critério
  - Se o desvio padrão das alturas dos pontos contidos no nó for maior do que um valor s_d pré-estabelecido, subdivide o nó
  - Caso contrário, trata-se de um nó folha

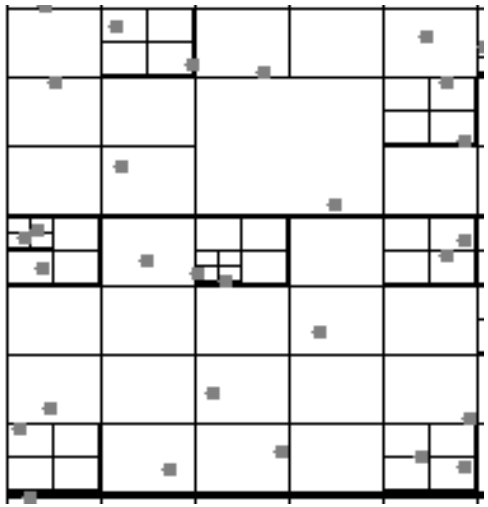
Hierarquia de malhas



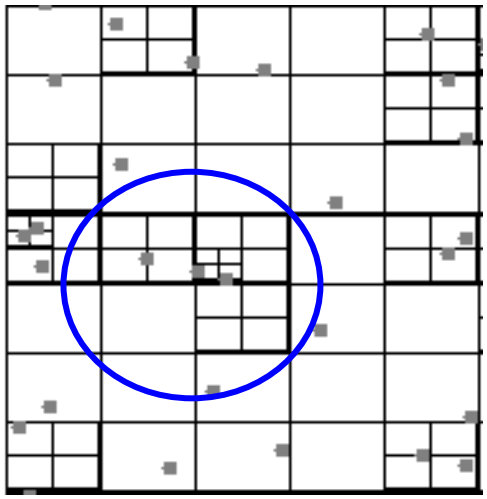
Balanceamento

Uma *quadtree* é dita balanceada se quaisquer dois quadrados vizinhos diferem no máximo de um fator dois; desta forma em uma *quadtree* balanceada, quaisquer duas folhas cujos quadrados são vizinhos, tem profundidades diferindo no máximo de 1.

Não balanceada



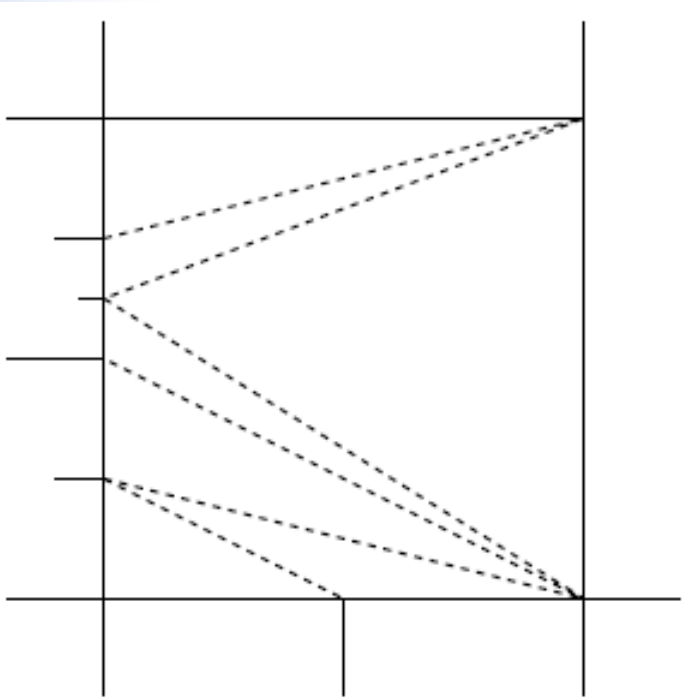
Balanceada



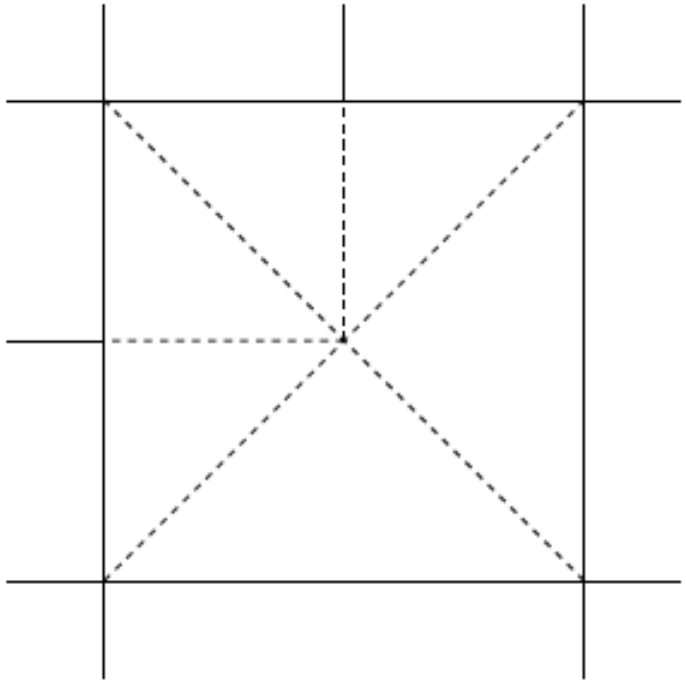
Balanceamento

Verificar se um quadrado s precisa ser dividido

1. Obter os vizinhos norte, sul, leste, oeste.
2. Dado um quadrado S verificamos seu vizinho norte por exemplo.
3. Verificamos se é folha
    3.1. Caso negativo, verificamos os filhos cujos lados sejam adjacentes ao lado do quadrado S e verifica se são folhas. Caso negativo, divide-se S e acrescentam os quatros filhos na lista para ser analisados.
4. Caso o vizinho norte não satisfaça as condições para divisão de S, tomam-se os demais vizinhos.



Malha obtida com quadtree não balanceada



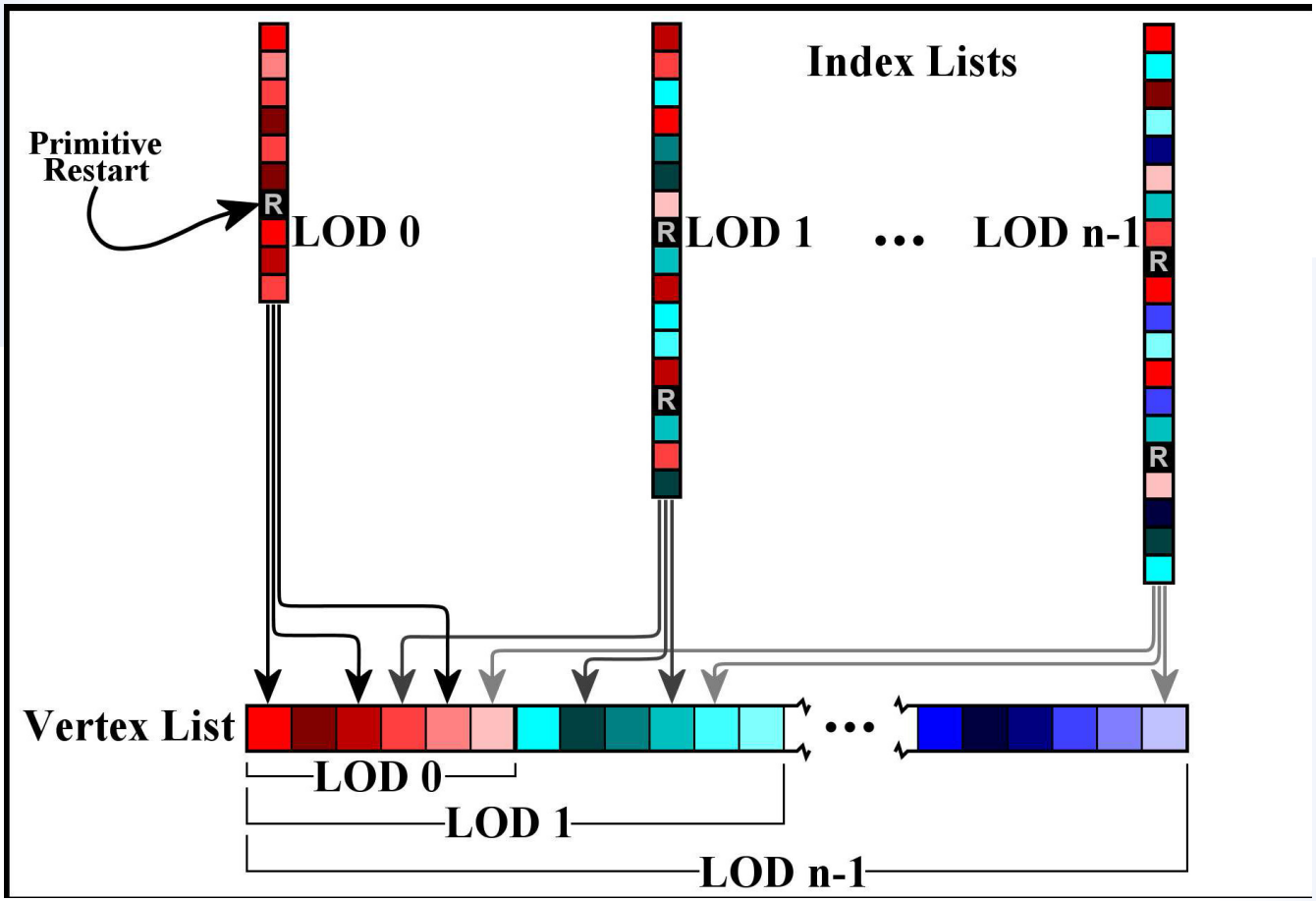
Malha obtida com quadtree balanceada

Geração de Malhas

Renderização

- A hierarquia de malhas permite que os tiles sejam enviados progressivamente a GPU
- Na GPU, rendering de tempo real em alta qualidade é realizado através de uma estrutura de dados apropriada
- Ao mesmo tempo, a CPU realiza o *frustum culling* e os cálculos de LoD para cada *tile*

Renderização



Renderização

- Para cada tile, calcula-se o bounding box para operações de *frustum culling*
- Para cada quadro, tiles visíveis são ordenados em ordem de profundidade para aproveitar o teste de profundidade anterior e evitar sobre-desenhos
- Para cada tile visível, o LoD apropriado é computado via CPU.

Renderização

- Para achar o LoD adequado para cada tile, utilizou-se uma métrica linear basada na distância da câmera até o centro do tile

$$LoD = \frac{(LevelMaximo - 1) * (dis\tan cia * dis\tan ciaMaxima)}{dis\tan ciaMaxima}$$

Implementação

- Implementação da malha hierárquica através de uma quadtree balanceada
  - Dado um nível da hierarquia, retorna a malha correspondente
- Divisão do dado em tiles
  - Cada tile representado por uma quadtree
  - Formato binário para armazenamento em disco
- Implementação em GPU
  - Linguagem Cg
- NVIDIA GeForce 9600 GT

Implementação

- Para evitar descontinuidades entre os tiles, usou-se a informação de altura fornecida pelo dado completo, de maneira que pontos coincidentes da quadtree tenham sempre o mesmo valor de z.

Resultados – G. Canyon, AZ (16k x 16k)

	4x4	5x5	6x6	7x7
tempo	1min 16 s	1 min 14 s	1 min 15 s	1 min 35 s
FPS	5	9	7	7

Resultados – G. Canyon, AZ (16k x 16k)

Terminal | lod - File Browser | trab_tcg_tiles_cg - Fil... | [Gmail: Email do Goo... | HierarchicalLODRend... | Terminal

HierarchicalLODRender.cpp (~/lod/lod_ok4/trab_tcg_tiles_cg) - gedit

File Edit View Search Tools Docum

New Open Save Print... Und

HierarchicalLODRender.cpp

```
        _vertices = new float[ve
        _normals = new float[ver

    }
    _nVertices = vertices.size()

    if (indexes.size()/3 > _nIn
    {
        if (_indexes) delete []
        _indexes = new unsigned
    }
    _nIndexes = indexes.size()/

    for (int i = 0; i< vertices
    {
        _vertices[i] = vertices
    }

    for (int i = 0; i< indexes.s
    {
        _indexes[i] = indexes[i];
    }
    computeNormals( _vertices, _indexes, _nVertices, _nIndexes, _normals );
}



void HierarchicalLODRender::render( FrustumG* frustum, bool wireFrame  )
{

    int nTiles = 1;

    if (wireFrame)
       glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);

    float fConstColor[] = { 0.5f, 0.5f, 0.5f, 1.0f };
    int level;

    glEnable( GL_DEPTH_TEST );

        glEnableClientState(GL_VERTEX_ARRAY);
    glEnableClientState(GL_NORMAL_ARRAY);
    glEnableClientState(GL_TEXTURE_COORD_ARRAY);

    glEnable(GL_LIGHTING);
    glDisable(GL_BLEND);
    glEnable(GL_TEXTURE_1D);
    glEnable(GL_COLOR_MATERIAL);
    glColorMaterial(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE);
```

Terminal

File Edit View Terminal Help

```
Tile 7 salvo
Tile 8 salvo
Tile 9 salvo
Tile 10 salvo
Tile 11 salvo
Tile 12 salvo
Tile 13 salvo
Tile 14 salvo
Tile 15 salvo
Tile 16 salvo
Tile 17 salvo
Tile 18 salvo
Tile 19 salvo
Tile 20 salvo
Tile 21 salvo
Tile 22 salvo
Tile 23 salvo
Tile 24 salvo
Tile 25 salvo
[itajuba:~/lod/lod_ok4/trab_tcg_tiles_cg] ./meshViewer teste.lod
[itajuba:~/lod/lod_ok4/trab_tcg_tiles_cg] ./meshViewer teste.lod
[itajuba:~/lod/lod_ok4/trab_tcg_tiles_cg] ./meshViewer teste.lod
[itajuba:~/lod/lod_ok4/trab_tcg_tiles_cg]
[itajuba:~/lod/lod_ok4/trab_tcg_tiles_cg]
```

Terminal

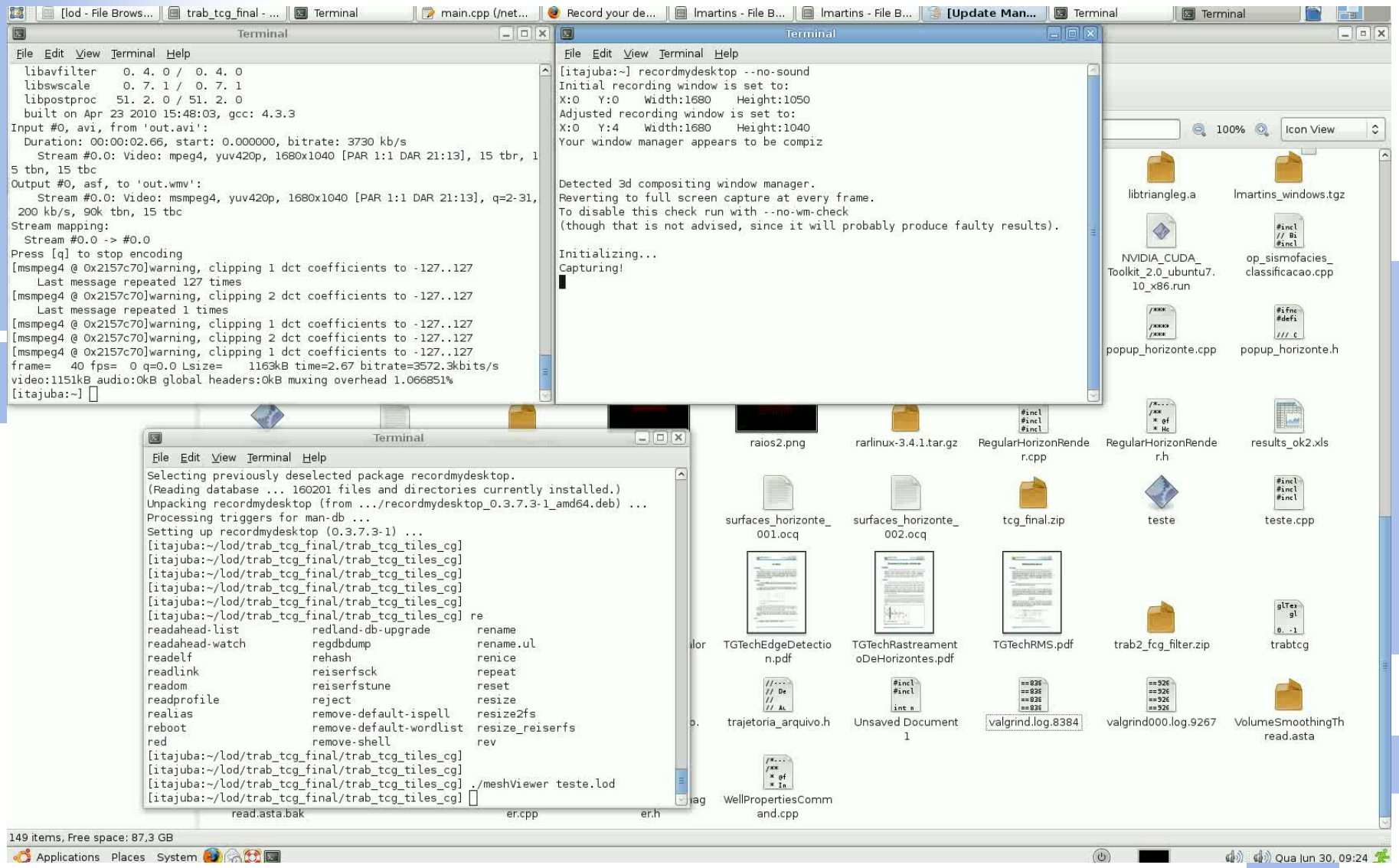
File Edit View Terminal Help

```
Written 165892015 bytes
(165892015 of which were video data and 0 audio data)

Cleanning up cache...
Done!!!
Goodbye!
[itajuba:~] rm out.ogv
rm: remove regular file `out.ogv'? y
[itajuba:~] recordmydesktop --no-sound
Initial recording window is set to:
X:0   Y:0    Width:1680    Height:1050
Adjusted recording window is set to:
X:0   Y:4    Width:1680    Height:1040
Your window manager appears to be compiz


Detected 3d compositing window manager.
Reverting to full screen capture at every frame.
To disable this check run with --no-wm-check
(though that is not advised, since it will probably produce faulty results).

Initializing...
Capturing!
```

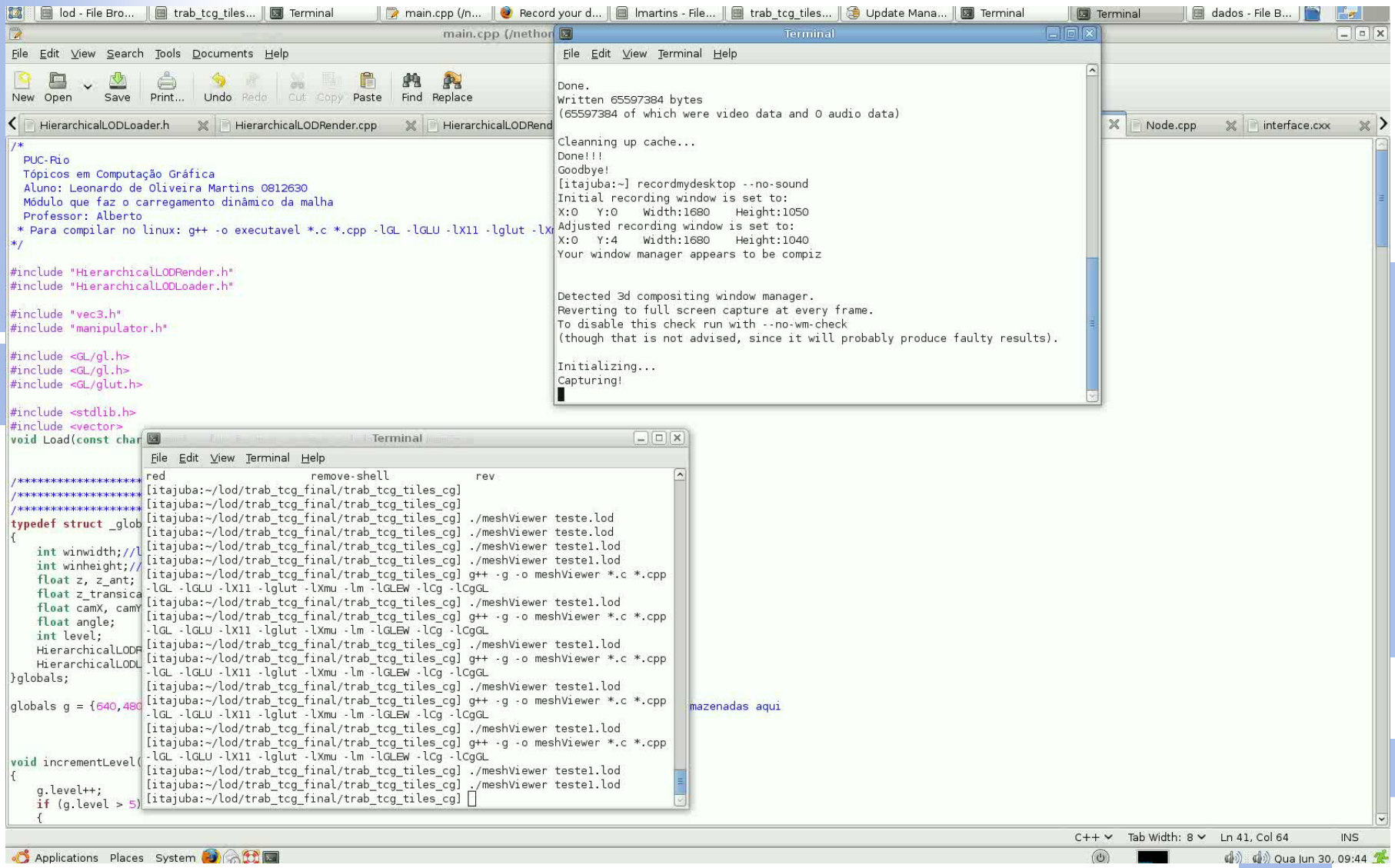
C++ Tab Width: 8 Ln 419, Col 76 INS

Applications Places System Qua Jul 7, 10:03

Resultados


[lod - File Brows...
trab_tcg_final - ...
Terminal
main.cpp (/net...
Record your de...
lmartins - File B...
lmartins - File B...
[Update Man...
Terminal
Terminal
File Edit View Terminal Help
libavfilter 0. 4. 0 / 0. 4. 0
libswscale 0. 7. 1 / 0. 7. 1
libpostproc 51. 2. 0 / 51. 2. 0
built on Apr 23 2010 15:48:03, gcc: 4.3.3
Input #0, avi, from 'out.avi':
Duration: 00:00:02.66, start: 0.000000, bitrate: 3730 kb/s
Stream #0.0: Video: mpeg4, yuv420p, 1680x1040 [PAR 1:1 DAR 21:13], 15 tbr, 1
5 tbn, 15 tbc
Output #0, asf, to 'out.wmv':
Stream #0.0: Video: msmpeg4, yuv420p, 1680x1040 [PAR 1:1 DAR 21:13], q=2-31,
200 kb/s, 90k tbn, 15 tbc
Stream mapping:
Stream #0.0 -> #0.0
Press [q] to stop encoding
[msmpeg4 @ 0x2157c70]warning, clipping 1 dct coefficients to -127..127
Last message repeated 127 times
[msmpeg4 @ 0x2157c70]warning, clipping 2 dct coefficients to -127..127
Last message repeated 1 times
[msmpeg4 @ 0x2157c70]warning, clipping 1 dct coefficients to -127..127
[msmpeg4 @ 0x2157c70]warning, clipping 2 dct coefficients to -127..127
[msmpeg4 @ 0x2157c70]warning, clipping 1 dct coefficients to -127..127
frame= 40 fps= 0 q=0.0 Lsize= 1163kB time=2.67 bitrate=3572.3kbits/s
video:1151kB audio:0kB global headers:0kB muxing overhead 1.066851%
[itajuba:~]
Terminal
File Edit View Terminal Help
[itajuba:~] recordmydesktop --no-sound
Initial recording window is set to:
X:0 Y:0 Width:1680 Height:1050
Adjusted recording window is set to:
X:0 Y:4 Width:1680 Height:1040
Your window manager appears to be compiz
Detected 3d compositing window manager.
Reverting to full screen capture at every frame.
To disable this check run with --no-wm-check
(though that is not advised, since it will probably produce faulty results).
Initializing...
Capturing!
Terminal
File Edit View Terminal Help
Selecting previously deselected package recordmydesktop.
(Reading database ... 160201 files and directories currently installed.)
Unpacking recordmydesktop (from .../recordmydesktop_0.3.7.3-1_amd64.deb) ...
Processing triggers for man-db ...
Setting up recordmydesktop (0.3.7.3-1) ...
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg] re
readahead-list redland-db-upgrade rename
readahead-watch regdbdump rename.ul
readelf rehash renice
readlink reiserfsck repeat
readom reiserfstune reset
readprofile reject resize
realias remove-default-ispell resize2fs
reboot remove-default-wordlist resize_reiserfs
red remove-shell rev
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg] ./meshViewer teste.lod
[itajuba:~/lod/trab_tcg_final/trab_tcg_tiles_cg]
100%
Icon View
libtriangleg.a
lmartins_windows.tgz
NVIDIA_CUDA_Toolkit_2.0_ubuntu7.10_x86.run
op_sismofacies_classificacao.cpp
popup_horizonte.cpp
popup_horizonte.h
raios2.png
rarlinux-3.4.1.tar.gz
RegularHorizonRender.cpp
RegularHorizonRender.h
results_ok2.xls
surfaces_horizonte_001.ocq
surfaces_horizonte_002.ocq
tcg_final.zip
teste
teste.cpp
TGTechEdgeDetection.pdf
TGTechRastreamentoDeHorizontes.pdf
TGTechRMS.pdf
trab2_fcg_filter.zip
trabtcg
trajetoria_arquivo.h
Unsaved Document 1
valgrind.log.8384
valgrind000.log.9267
VolumeSmoothingThread.asta
WellPropertiesCommand.cpp
read.asta.bak
er.cpp
er.h
149 items, Free space: 87,3 GB
Applications Places System
Qua Jun 30, 09:24

Resultados - Grand Canyon, AZ (4096 x 2048)



Conclusões

- A criação de uma hierarquia a partir dos pontos do mapa de altura permite um refinamento adaptativo que permite maior eficiência na visualização desse mapa
- Entretanto, diversas melhorias podem ser implementadas no presente trabalho de maneira a acelerar o processo de *rendering* de dados cada vez maiores, assim como aumentar a qualidade da visualização
  - Implementação do LoD contínuo
  - Gerenciamento de memória eficiente para reduzir o tempo de acesso a disco
  - Investigar outras métricas para determinar o LoD de um tile
  - Cálculo de normais e iluminação na GPU

Referências

- "GPU-Friendly High-Quality Terrain Rendering". Schneider and Westermann, 2006
- M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, *Computational Geometry: Algorithms and Applications*, Springer-Verlag, 1997.
- RASTeR: Simple and Efficient Terrain Rendering on the GPU. Bösch, Goswami and Pajarola, 2009.
- http://www.vterrain.org/LOD/Papers/
- http://lodbook.com/terrain/