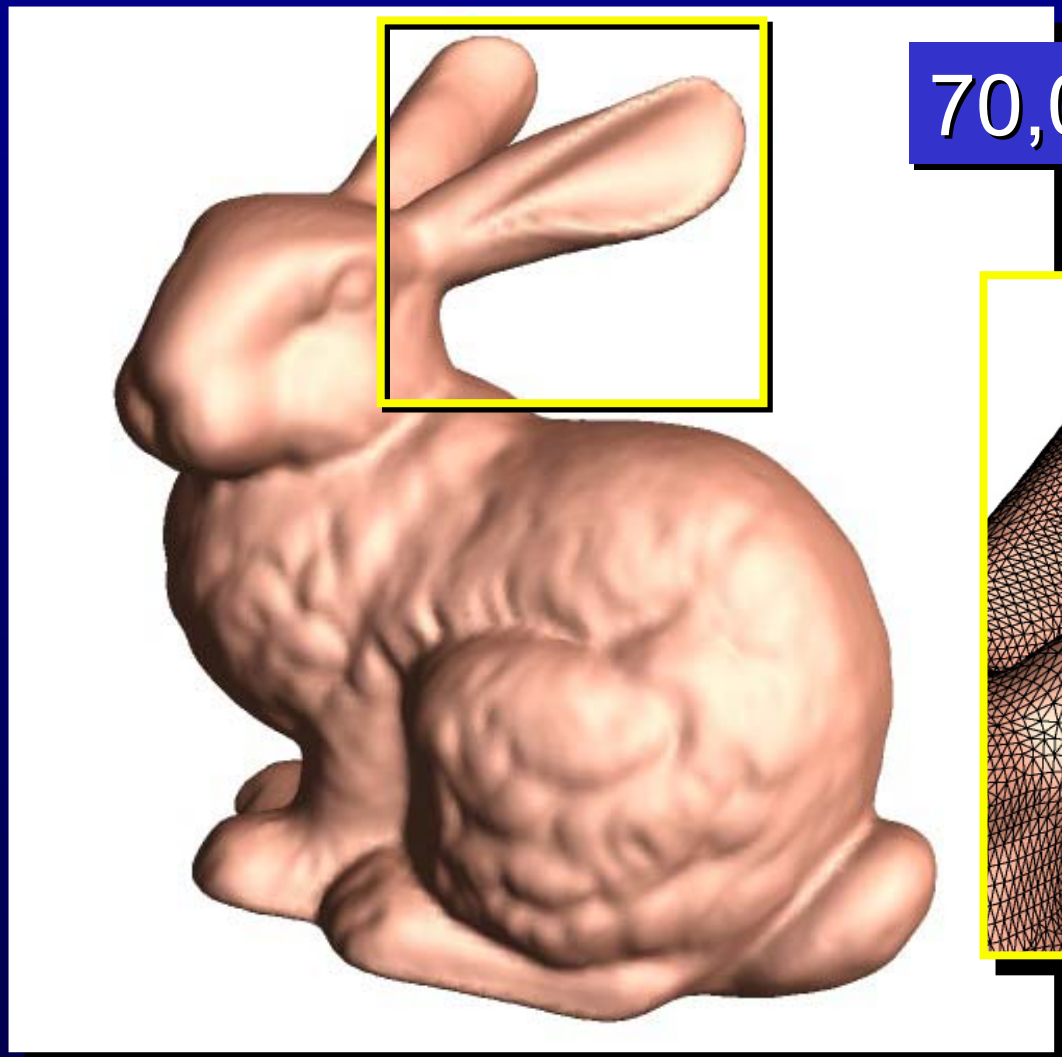
Irregular to Completely Regular Meshing in Computer Graphics

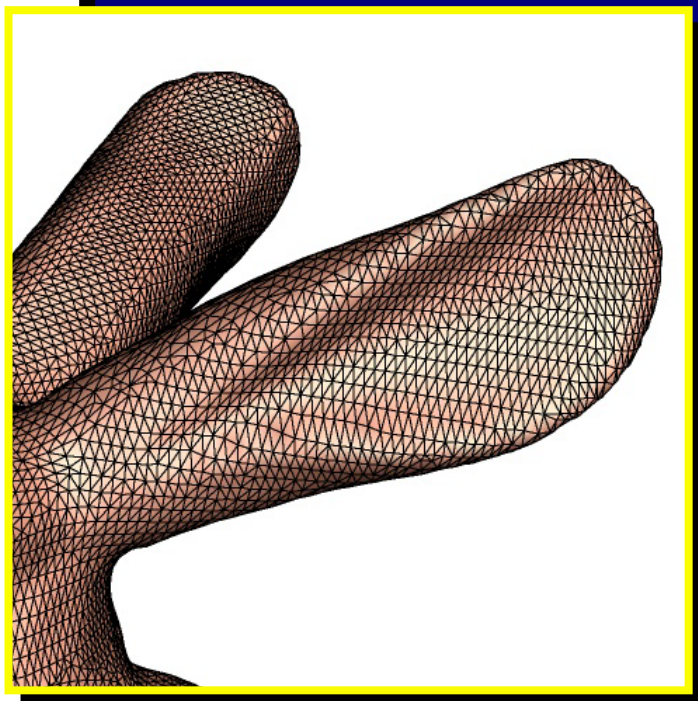
Hugues Hoppe
Microsoft Research

International Meshing Roundtable
2002/09/17

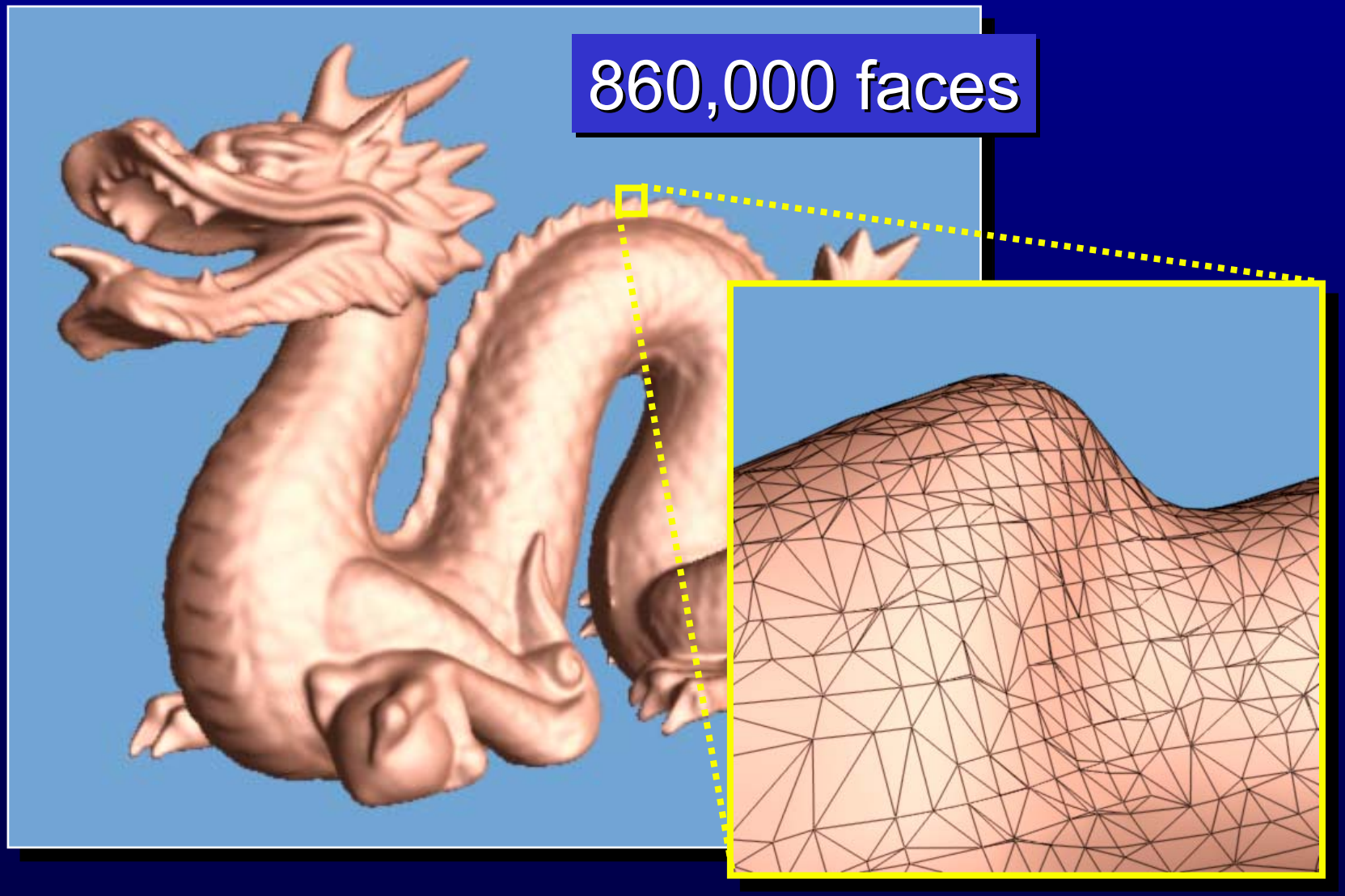
Complex meshes in graphics (1994)



70,000 faces



Complex meshes in graphics (1997)



Complex meshes in graphics (2000)

[Digital Michelangelo Project]

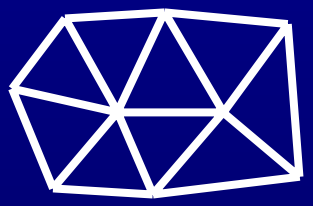
2,000,000,000 faces

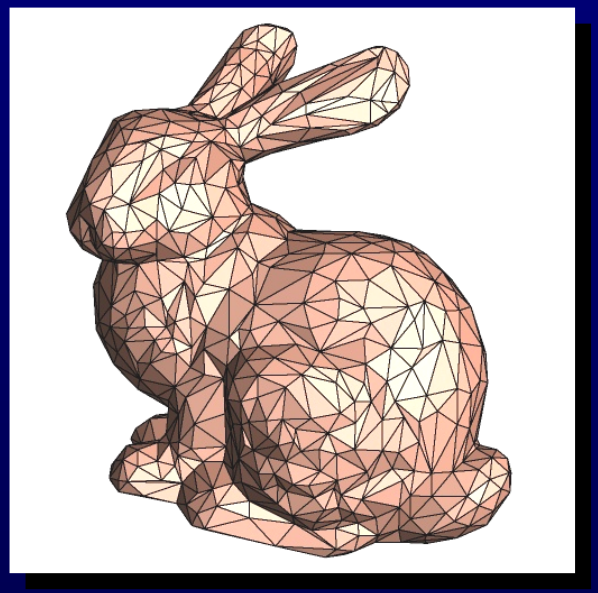
<u>Challenges:</u>

- rendering
- storage
- transmission
- scalability

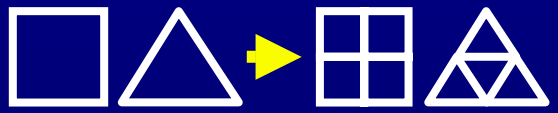
Multiresolution geometry

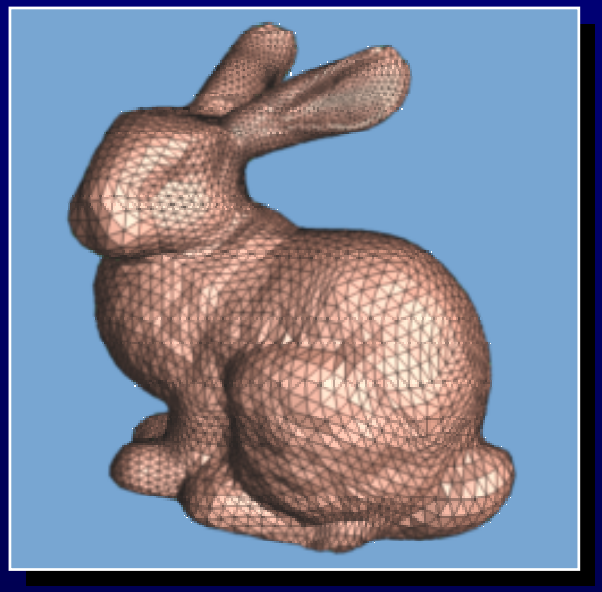
Irregular



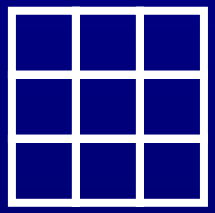


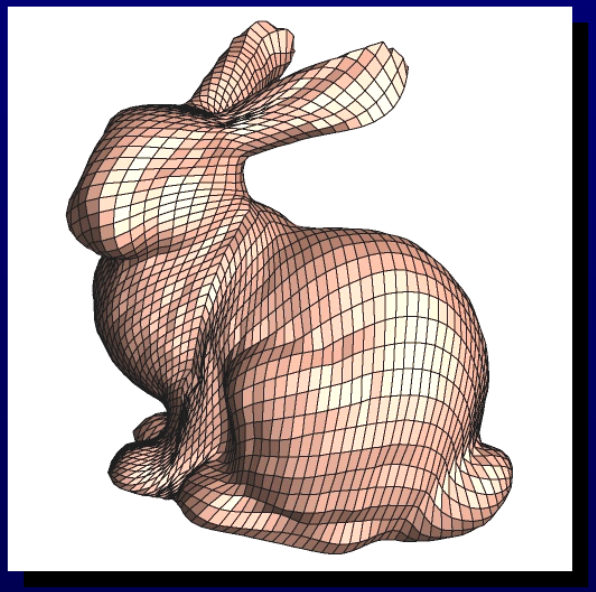
Semi-regular





Completely regular





Multiresolution geometry

- Irregular meshes
  - *Progressive meshes* [1996]
  - *View-dependent refinement* [1997]
  - *Texture-mapping PM* [2001]
- Semi-regular meshes
  - *Multiresolution analysis* [1995]
- Completely regular meshes
  - *Geometry images* [2002]

Goals in real-time rendering

#1 : Rendering speed

- 60-85 frames/second

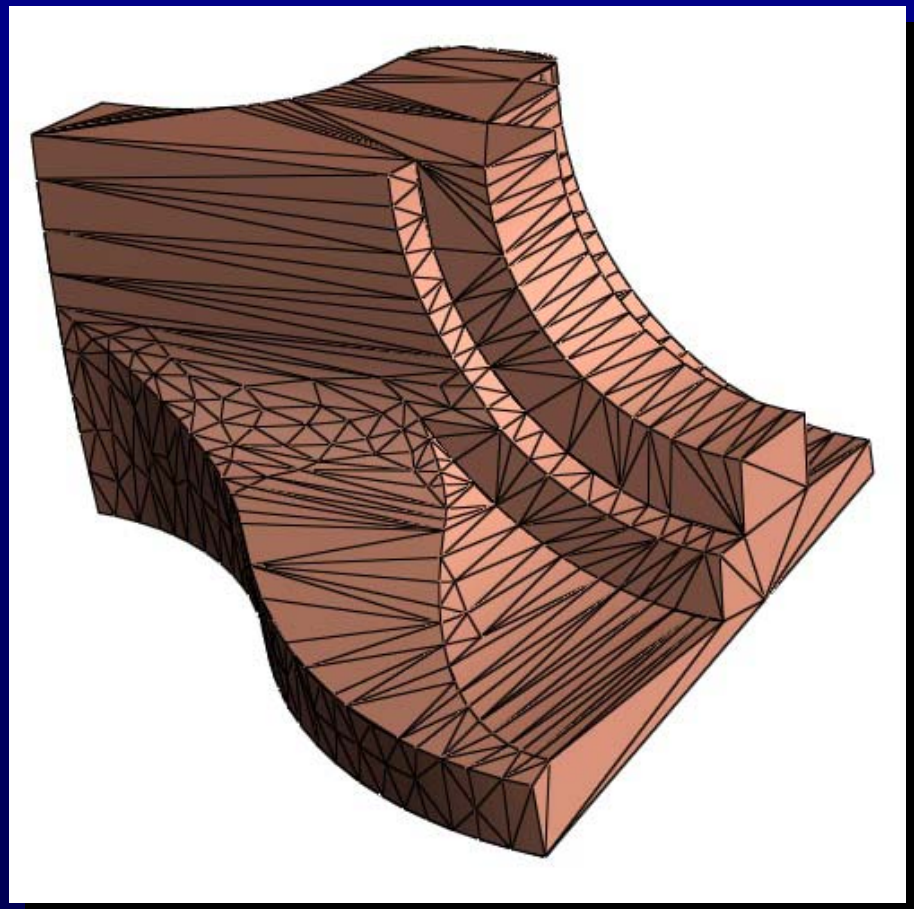
#2 : Rendering quality

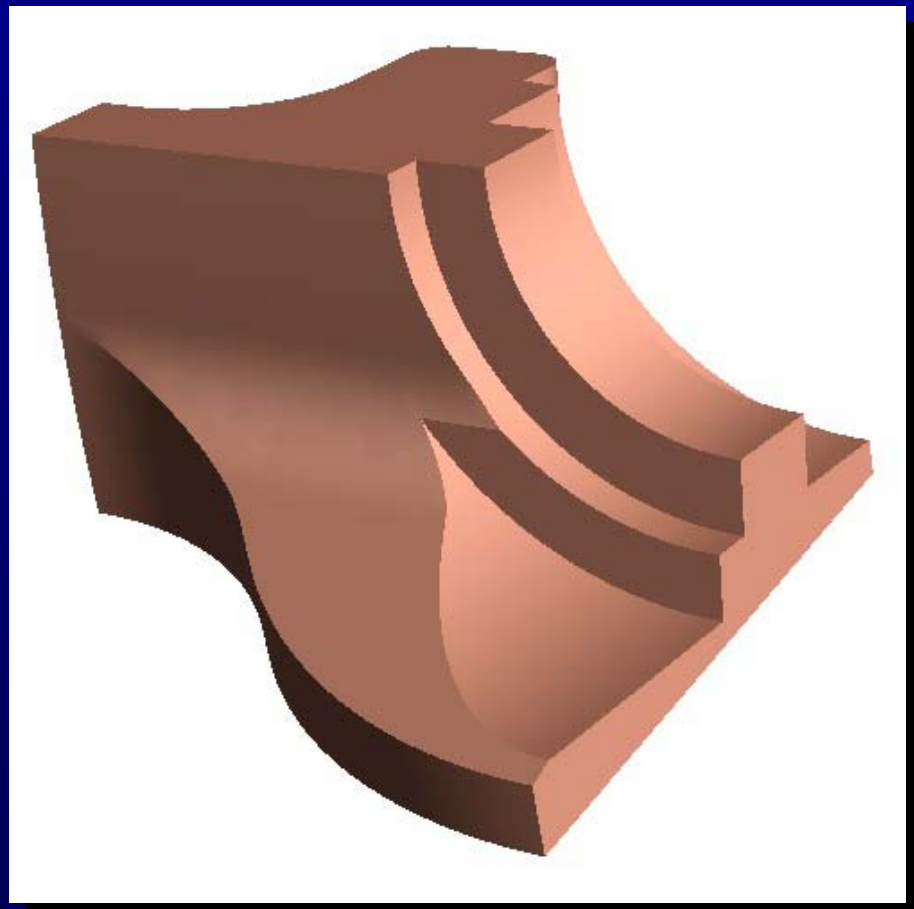
- ~~geometric~~ “visual” accuracy
- temporal continuity

Not a Goal:

- Mesh “quality”

Not a goal: mesh quality





13,000 faces → 1,000 faces

Irregular meshes

Vertex 1 x_1 y_1 z_1
Vertex 2 x_2 y_2 z_2
…

Face 2 1 3
Face 4 2 3
…

Rendering cost = vertex processing + rasterization

~ #vertices

~ constant

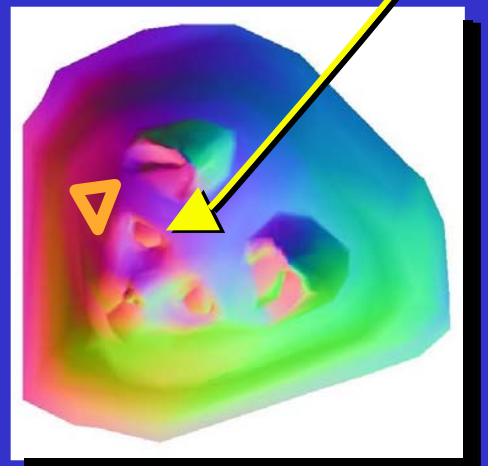
yuck

Texture mapping

"Visual" accuracy using coarse mesh

Vertex 1 x_1 y_1 z_1 s_1 t_1
Vertex 2 x_2 y_2 z_2 s_2 t_2
…

Face 2 1 3
Face 4 2 3
…



normal map

Goals in real-time rendering

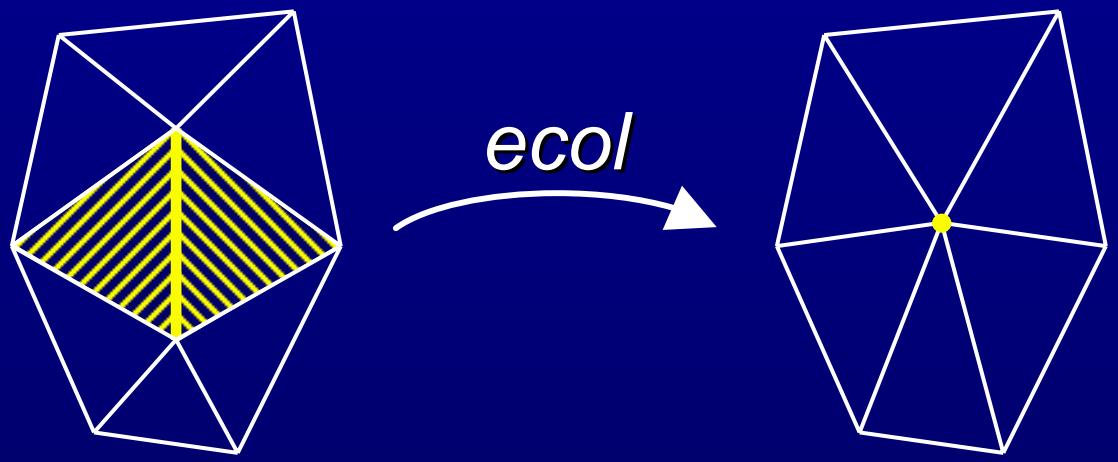
#1 : Rendering speed

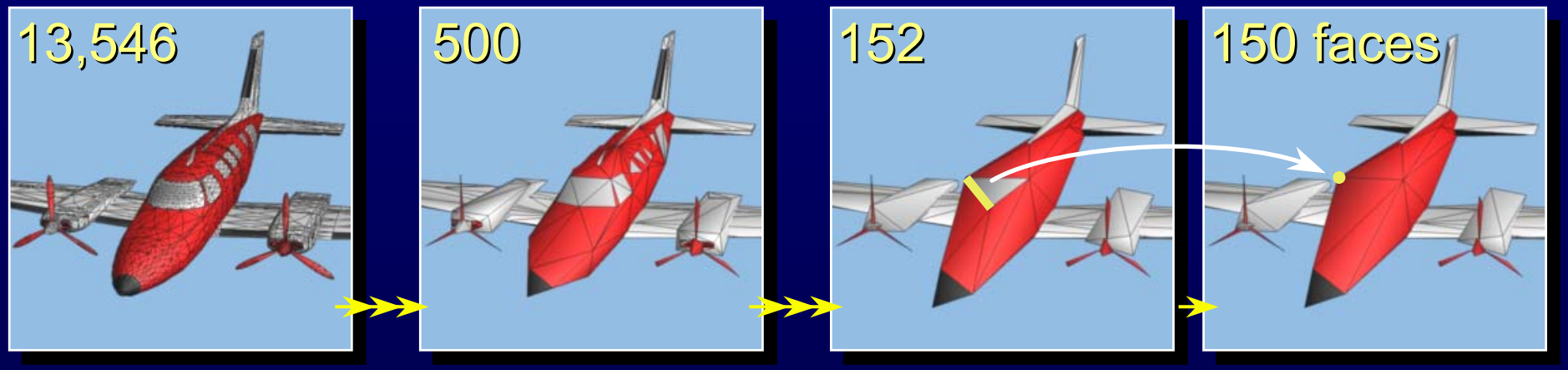
- Minimize #vertices →
  best accuracy using irregular meshes

#2 : Rendering quality

- Use texture mapping → parametrization

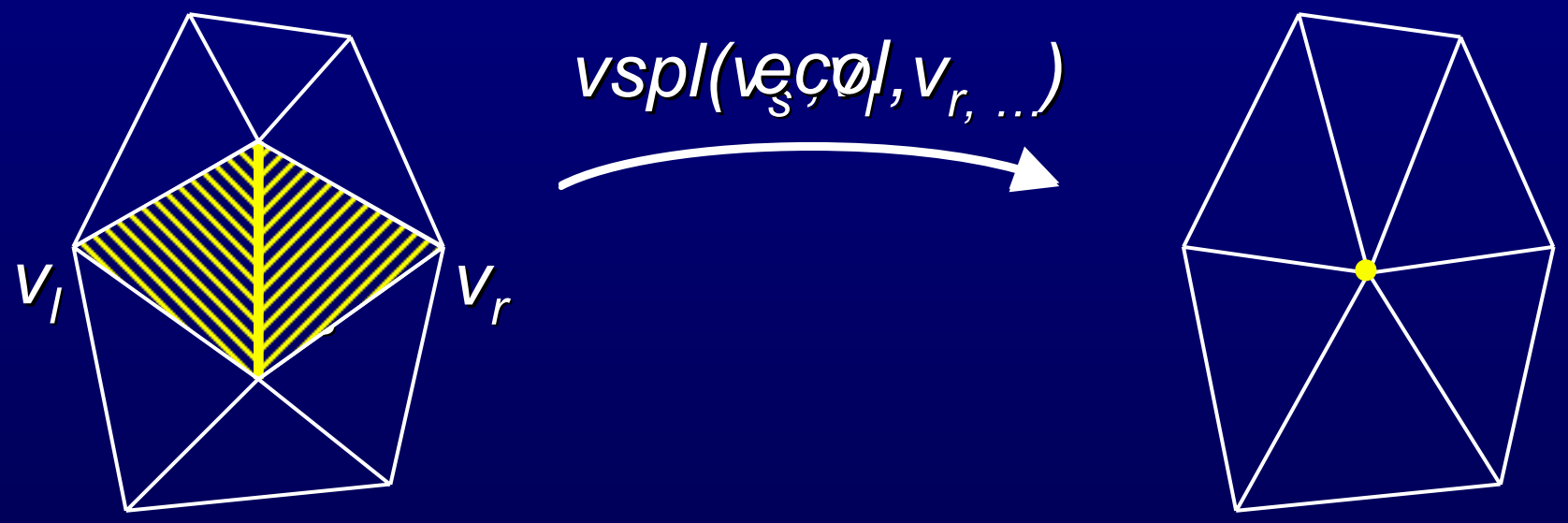
Simplification: Edge collapse



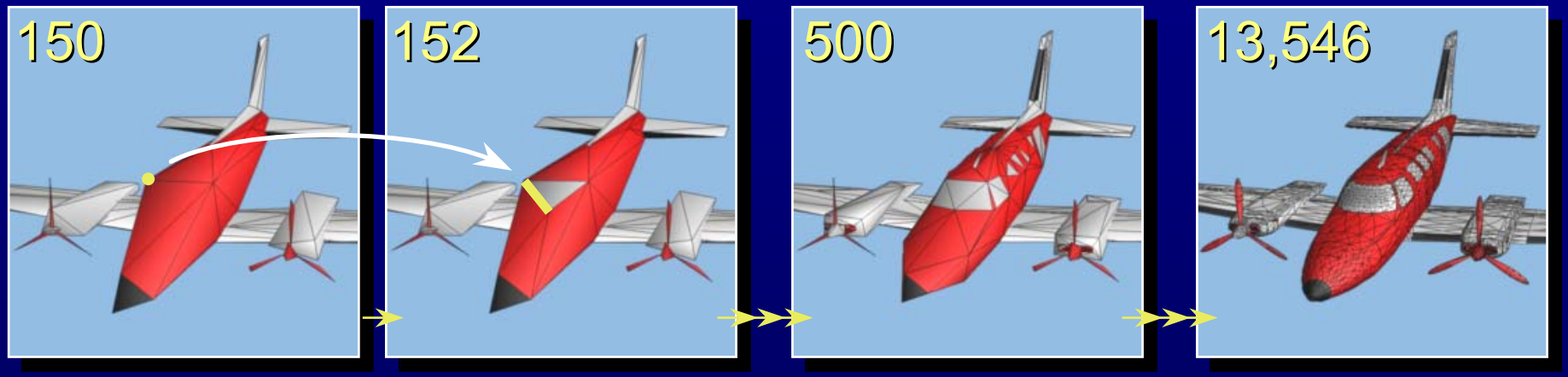


$$M^n \xrightarrow[ecol_{n-1}]{} \cdots M^{175} \cdots \xrightarrow[ecol_i]{} M^1 \xrightarrow[ecol_0]{} M^0$$

Invertible: vertex split transformation



Progressive mesh



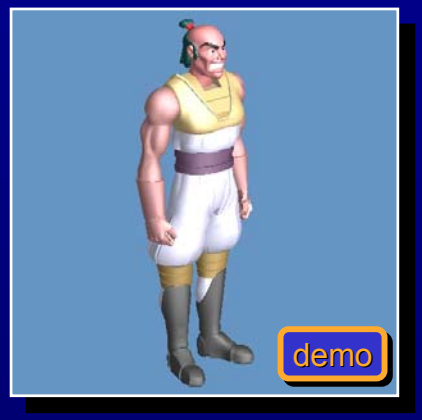
$M^0 \longrightarrow M^1 \longrightarrow \cdots M^{175} \cdots \longrightarrow M^n$

$vspl_0 \quad \ldots \quad vspl_i \quad \ldots \quad vspl_{n-1}$

progressive mesh (PM) representation

Applications

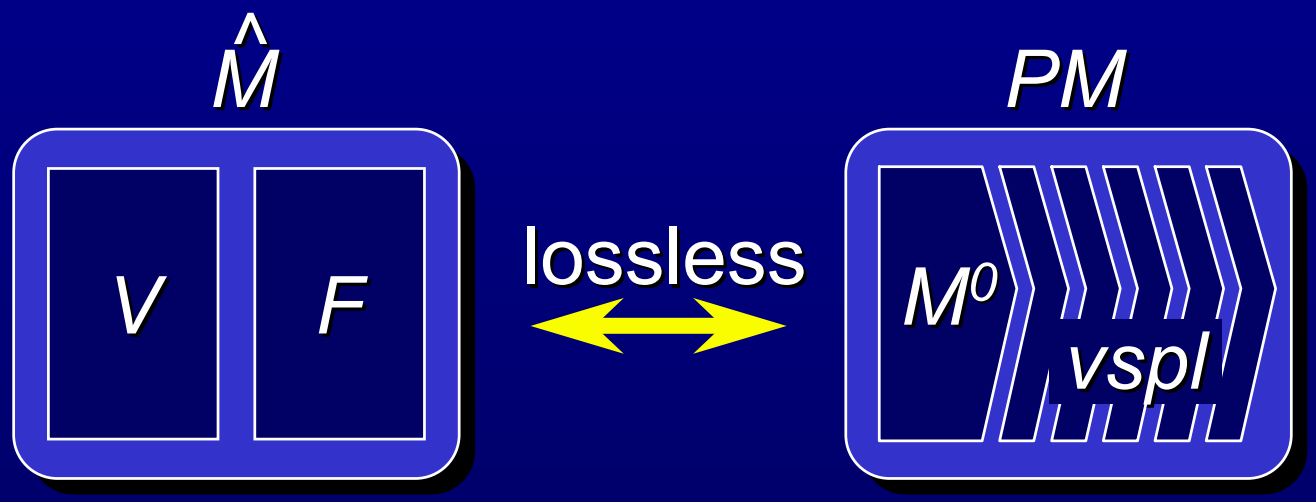
- Continuous LOD



- Geomorphs

- Progressive transmission

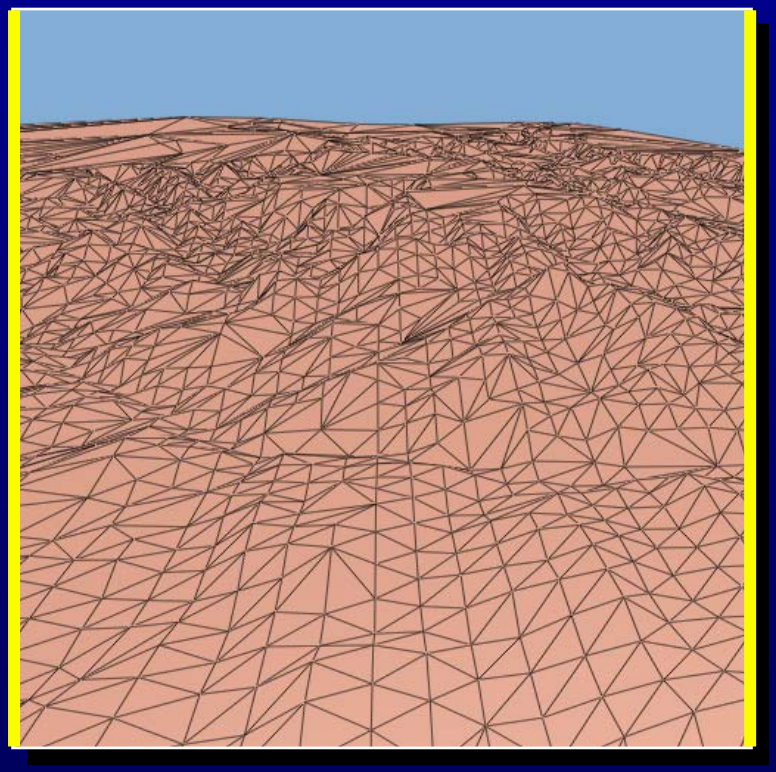
Progressive Mesh Summary



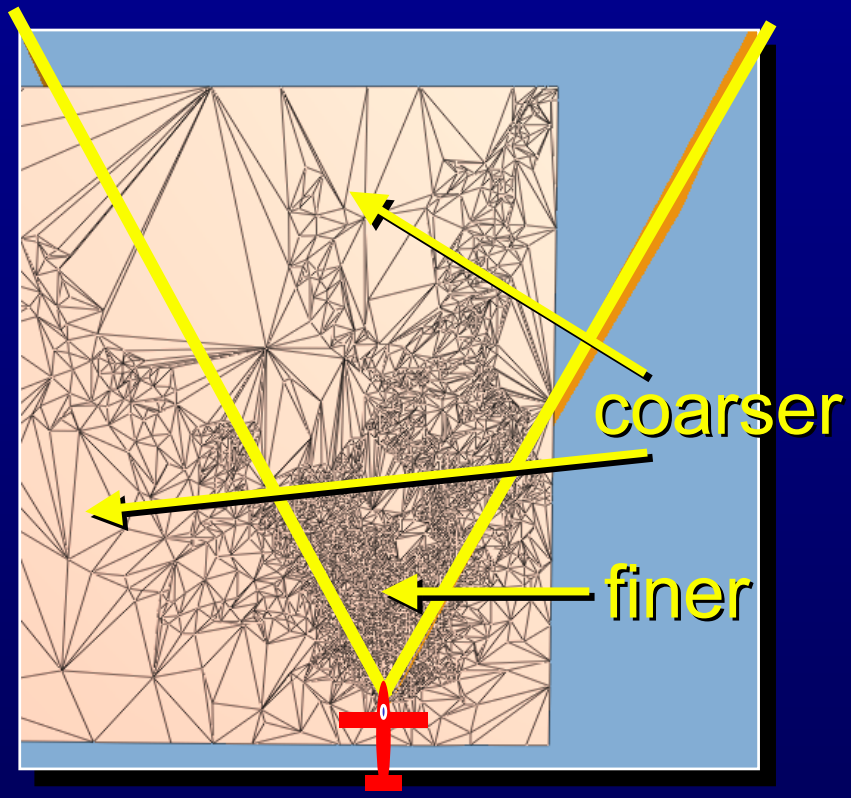
- single resolution

- continuous-resolution
- smooth LOD
- progressive
- space-efficient

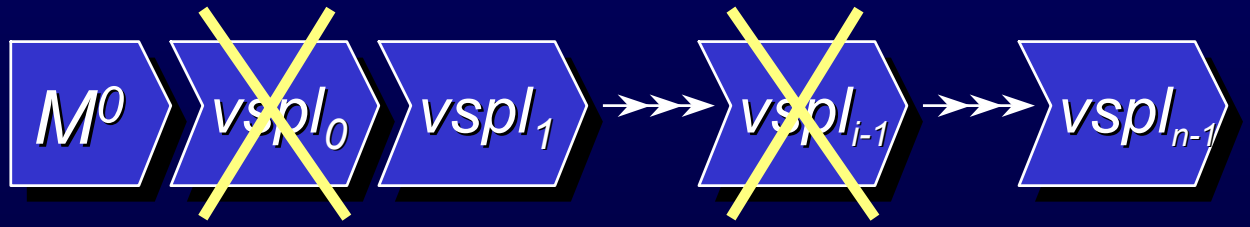
View-dependent refinement of PM's



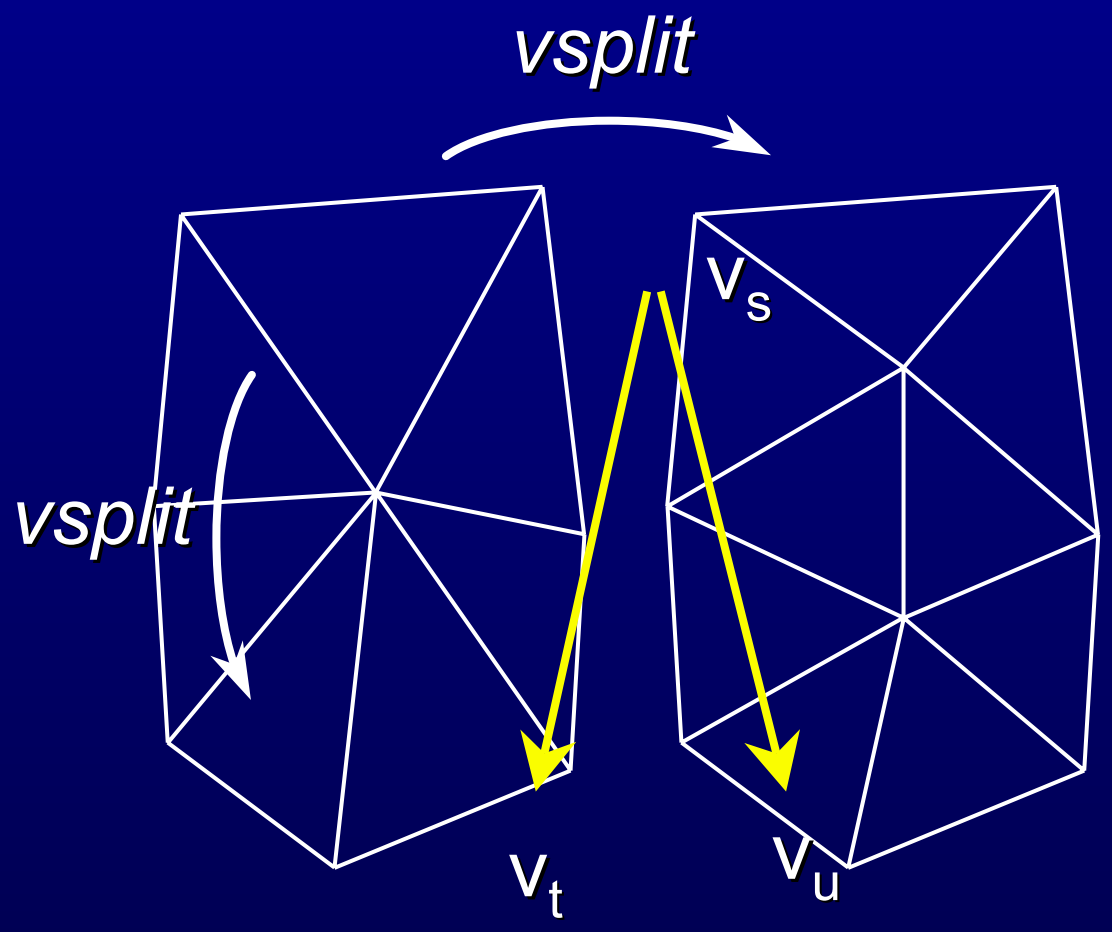
actual view



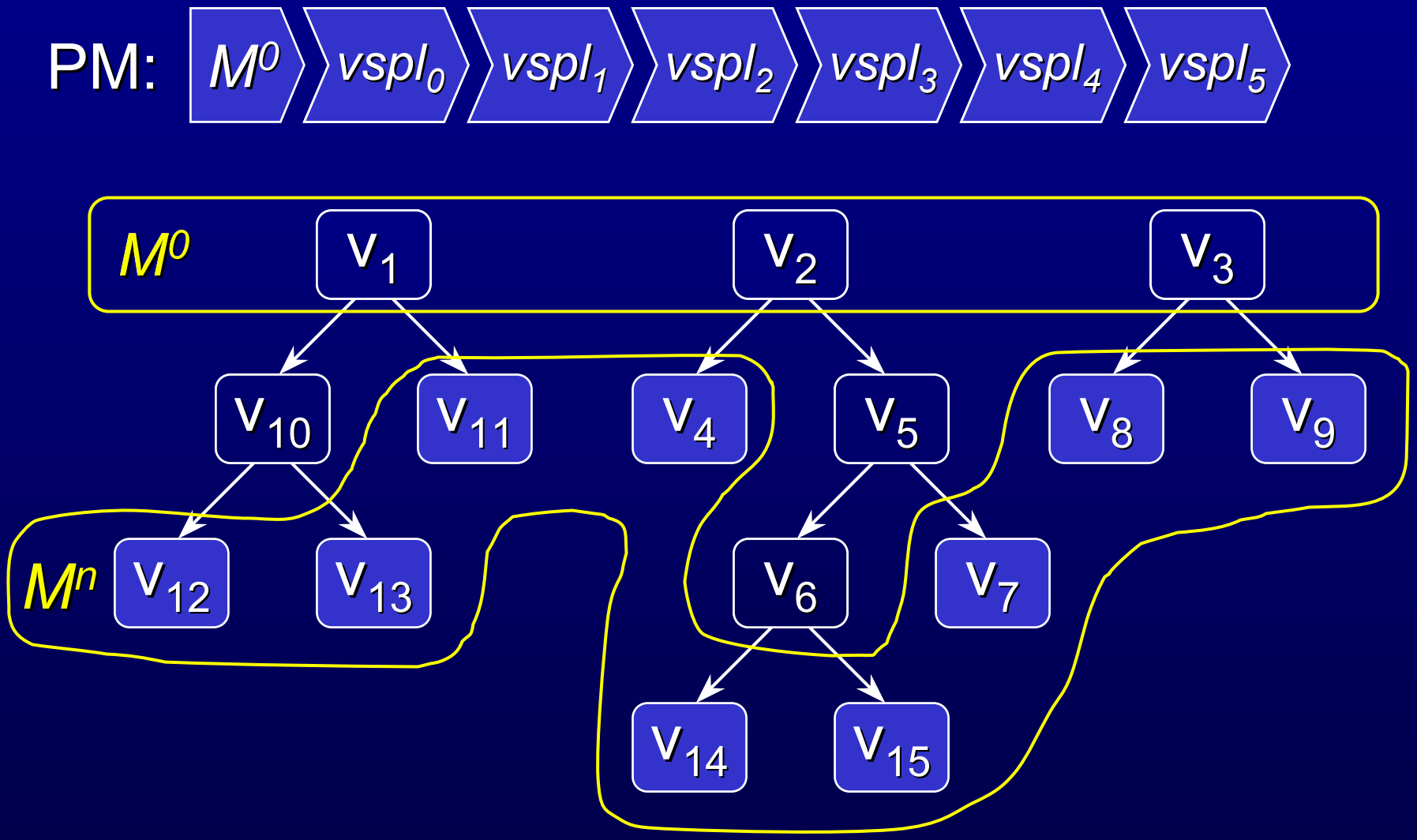
overhead view



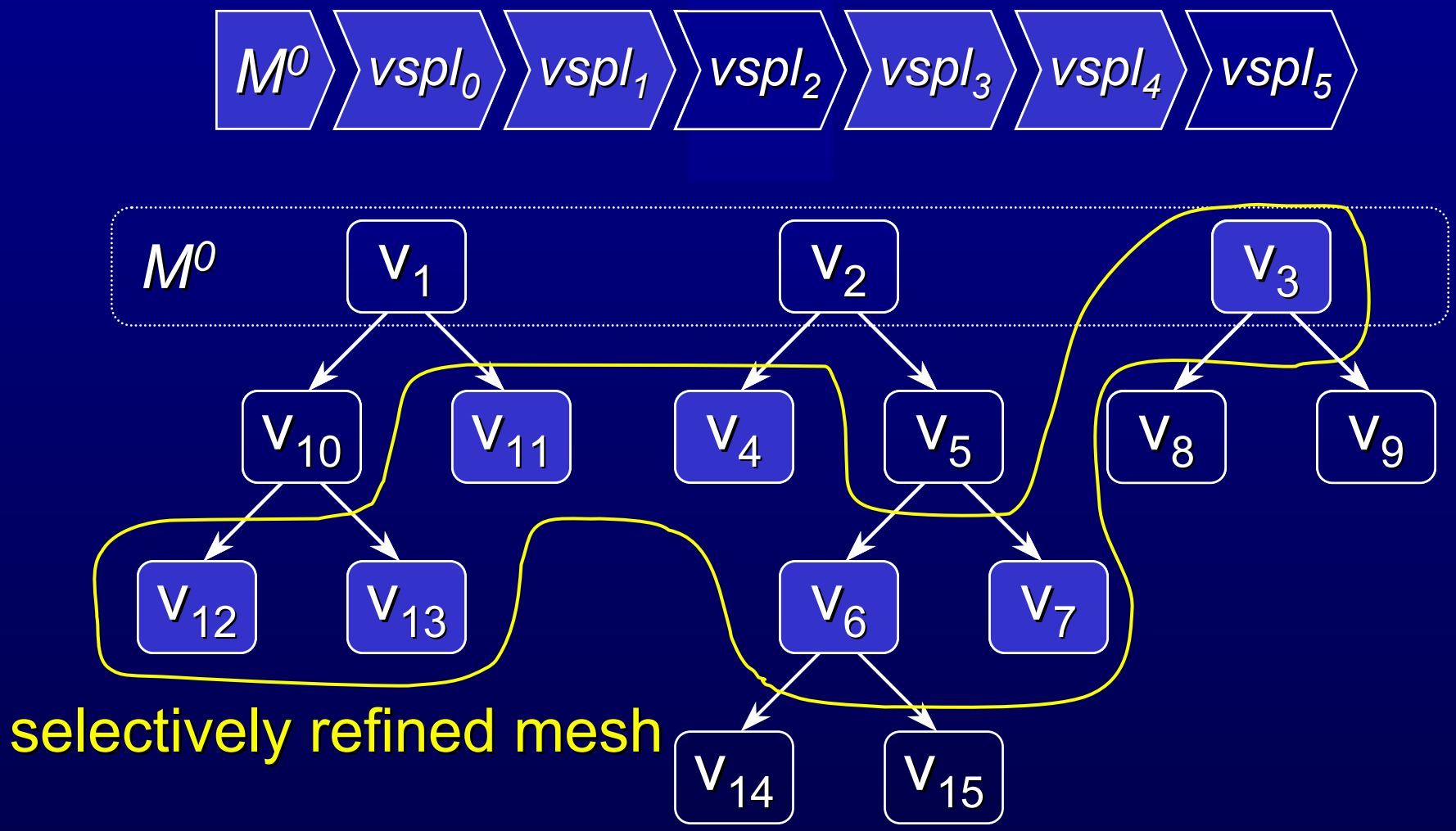
Parent-child vertex relations



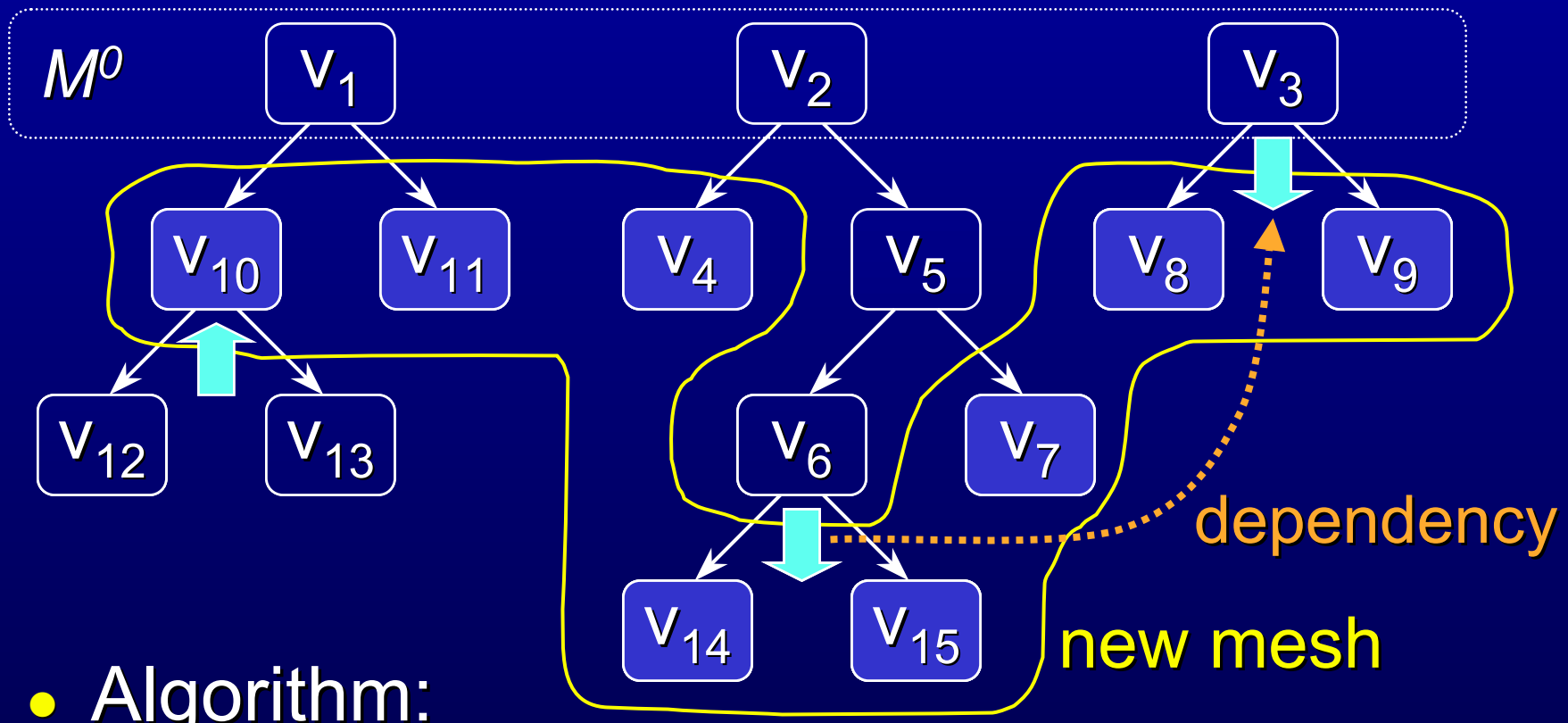
Vertex hierarchy



Selective refinement

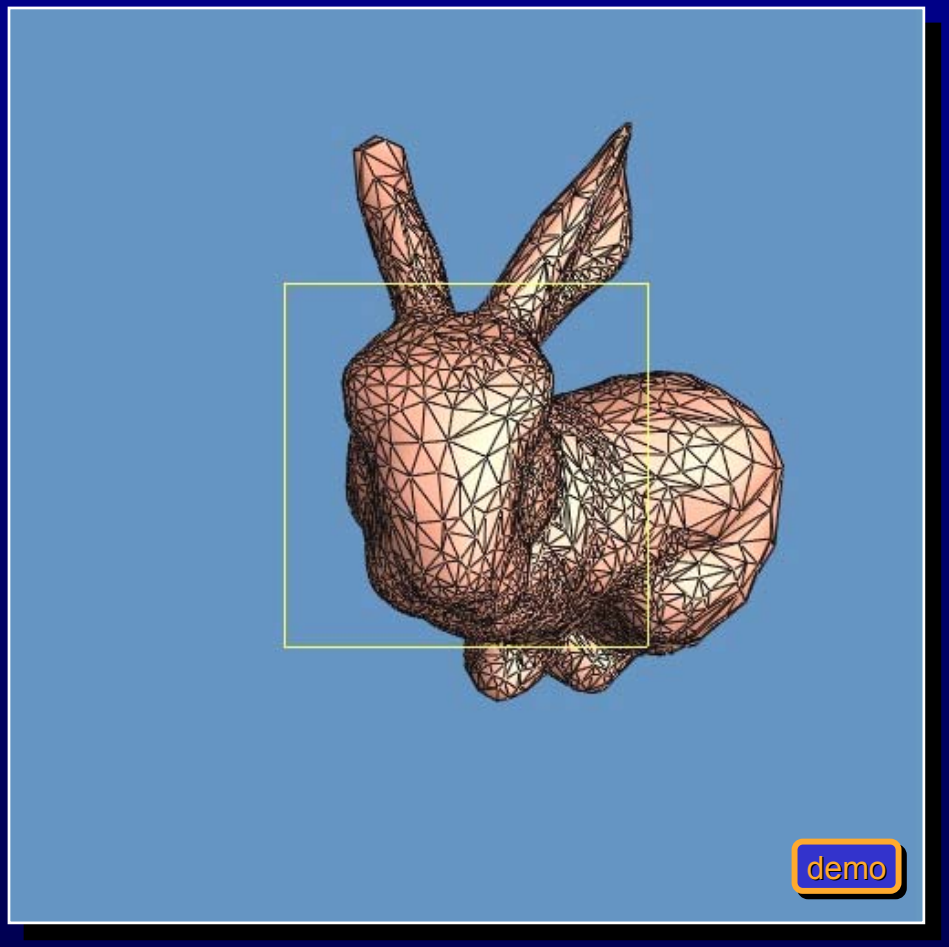


Runtime algorithm

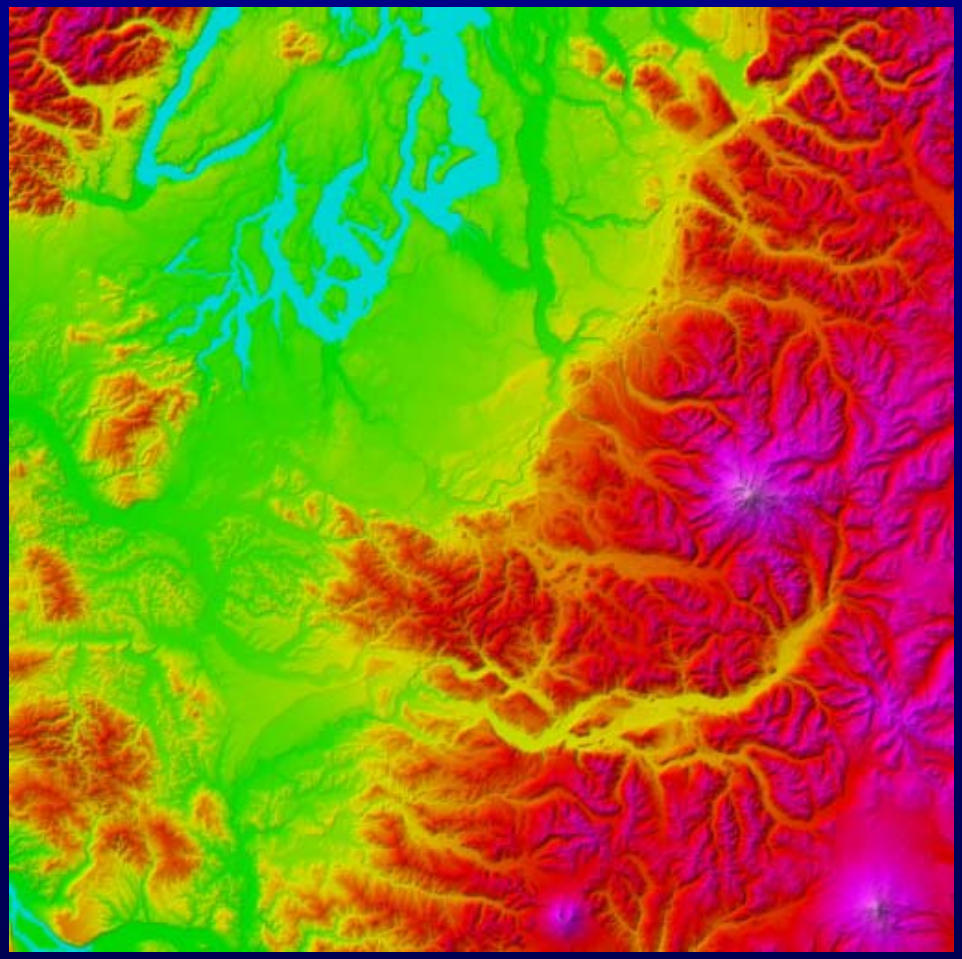


- Algorithm:
  - incremental
  - efficient
  - amortizable

DEMO: View-dependent LOD



Complex terrain model



Puget Sound data

16K x 16K vertices

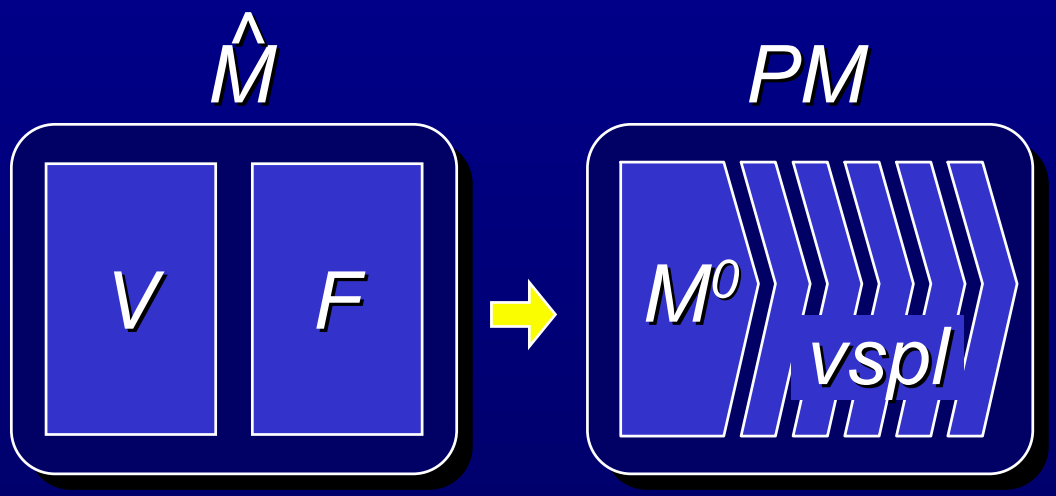
~537 million triangles

10m spacing, 0.1m resolution

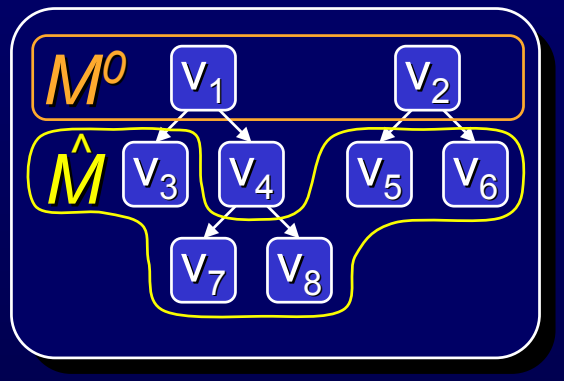
4m demo

simpler 10m demo

Selective Refinement Summary



- continuous-resolution
- smooth LOD
- space-efficient
- progressive

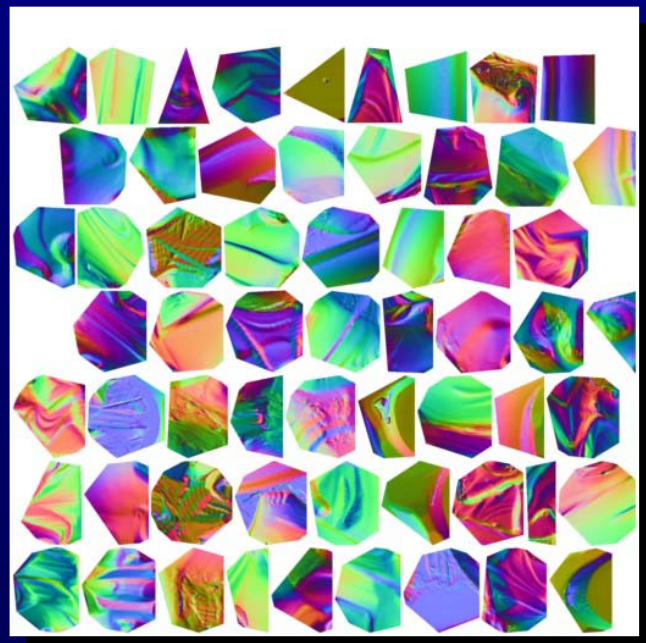


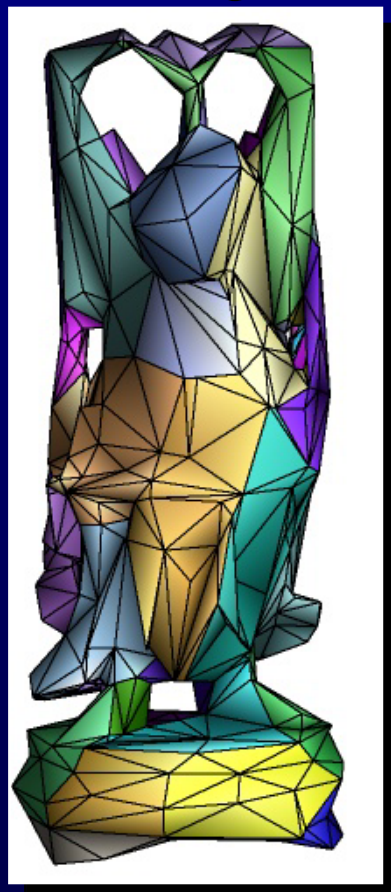
- view-dependent refinement
- real-time algorithm

Texture mapping progressive meshes [Sander et al 2001]

- Construct texture atlas valid for all $M^0 \ldots M^n$.

e.g. 1000 faces





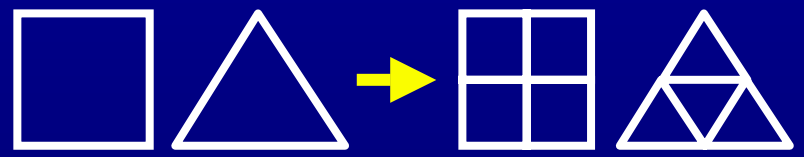
demo

pre-shaded demo

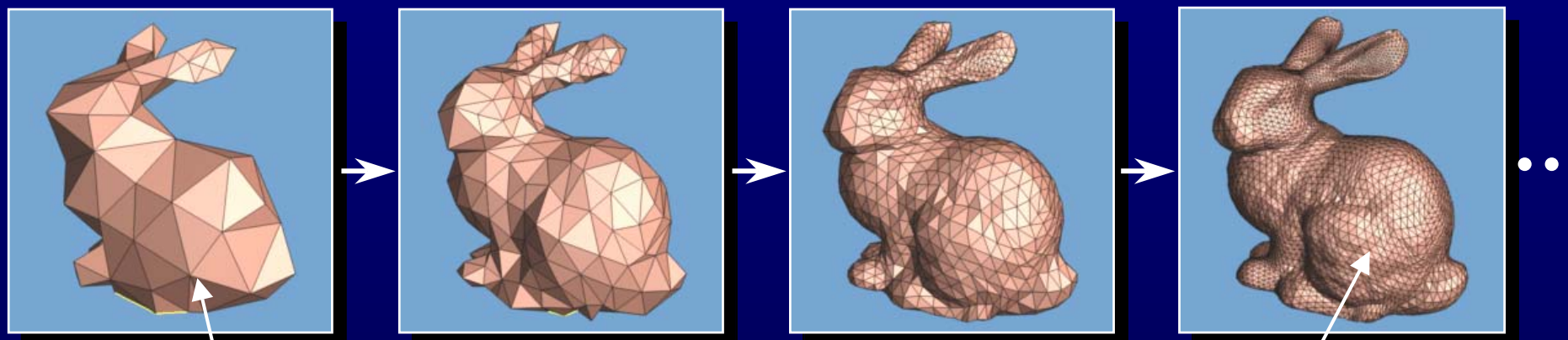
Multiresolution geometry

- Irregular meshes
  - *Progressive meshes* [1996]
  - *View-dependent refinement* [1997]
  - *Texture-mapping PM* [2001]
- Semi-regular meshes
  - *Multiresolution analysis* [1995]
- Completely regular meshes
  - *Geometry images* [2002]

Semi-regular representations



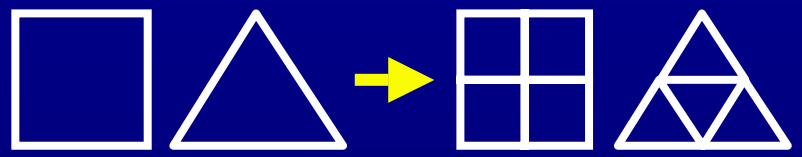
[Eck et al 1995]
[Lee et al 1998]
[Khodakovsky 2000]
[Guskov et al 2000]
[Lee et al 2000]
…



irregular base mesh

***semi**-regular*

Challenge: finding domain



[Eck et al 1995]
[Lee et al 1998]
[Khodakovsky 2000]
[Guskov et al 2000]
[Lee et al 2000]
…

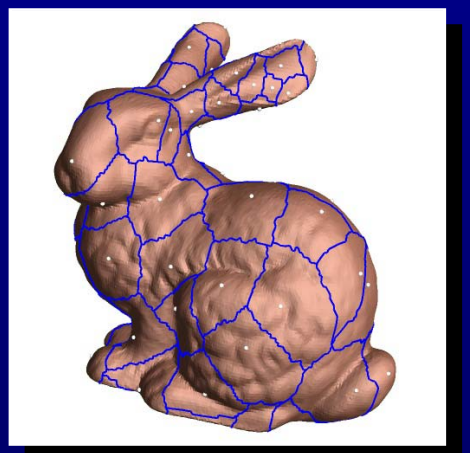
base domain

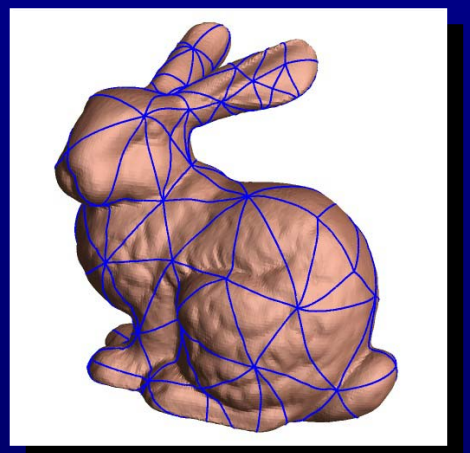
original surface

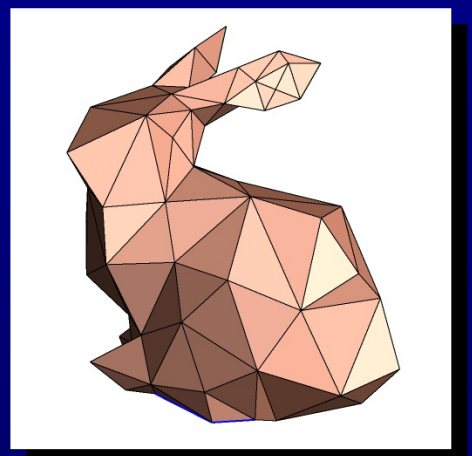
Techniques

- “Delaunay” partition + parametrization

[Eck et al. 1995]

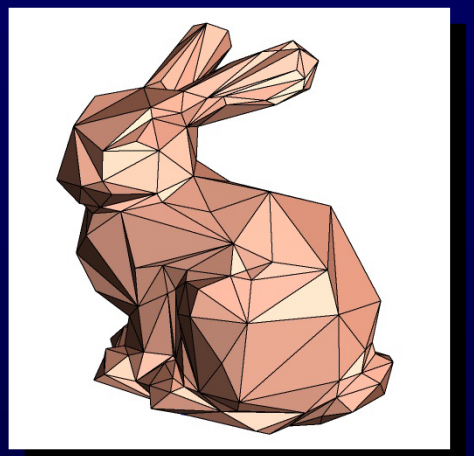


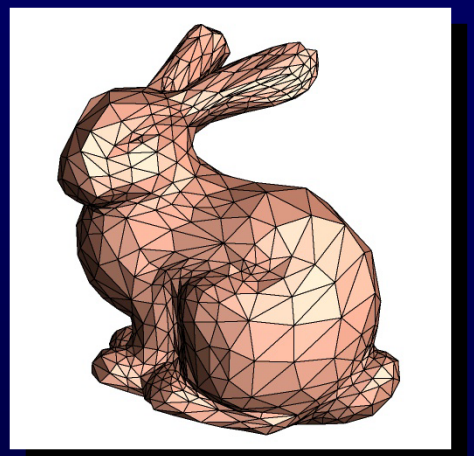


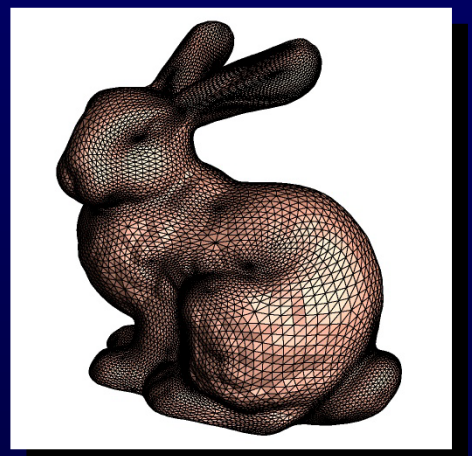


- Mesh simplification + …

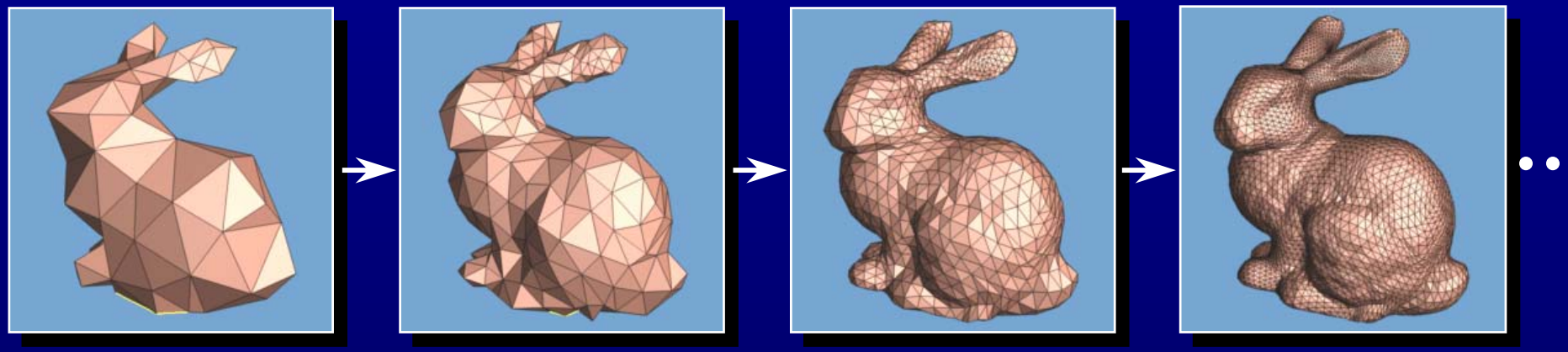
[Lee et al. 1998]
[Lee et al. 2000]
[Guskov et al. 2000]







Semi-regular: Applications



- View-dependent refinement [Lounsbery et al. 1994]
- Texture-mapping [Certain et al. 1995]
- Multiresolution editing [Zorin et al. 1997]
- Compression [Khodakovsky et al. 1999]
- …

Multiresolution geometry

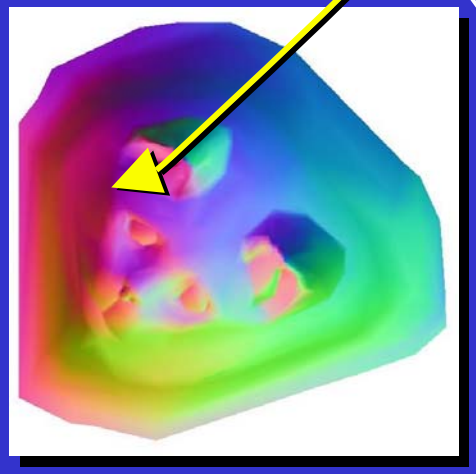
- Irregular meshes
  - *Progressive meshes* [1996]
  - *View-dependent refinement* [1997]
  - *Texture-mapping PM* [2001]
- Semi-regular meshes
  - *Multiresolution analysis* [1995]
- Completely regular meshes
  - *Geometry images* [2002]

Mesh rendering: complicated process

Vertex 1 x_1 y_1 z_1 s_1 t_1
Vertex 2 x_2 y_2 z_2 s_2 t_2
…

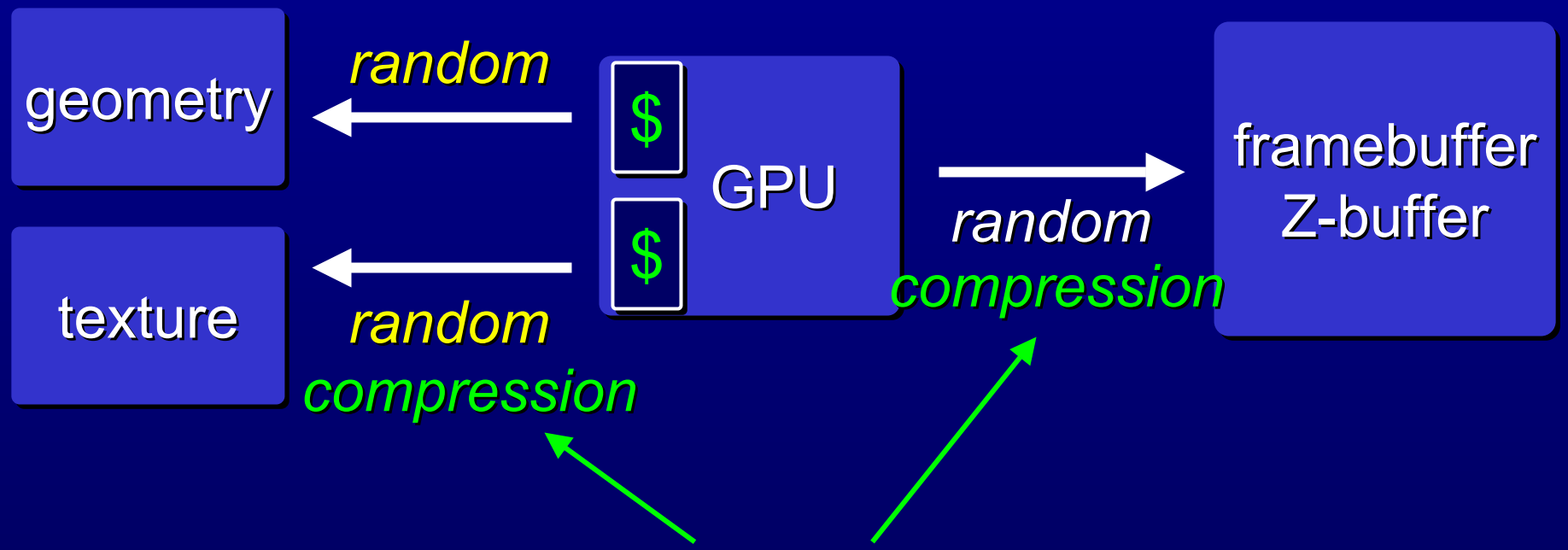
Face 2 1 3
Face 4 2 3
…

random access!



random access!

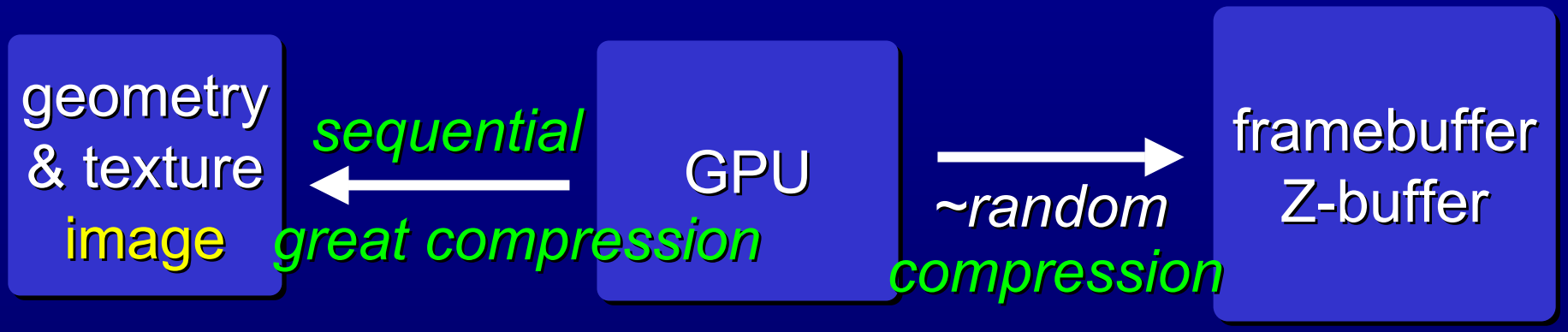
Current architecture



2D image compression

~40M Δ/sec

New architecture

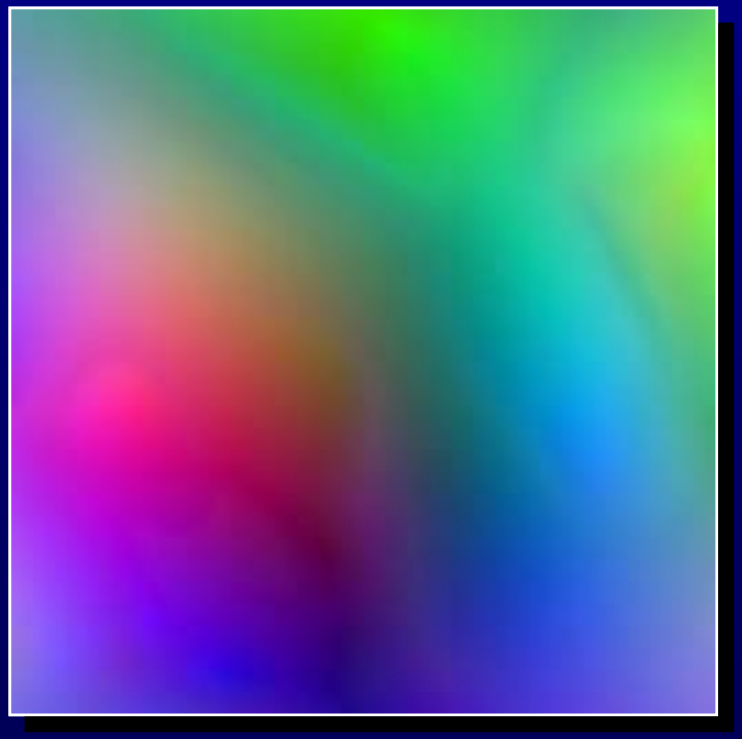


- Minimize ~~#vertices~~ bandwidth, through compression.
- Maximize sequential (non-random) access

Geometry Image

[Gu et al 2002]

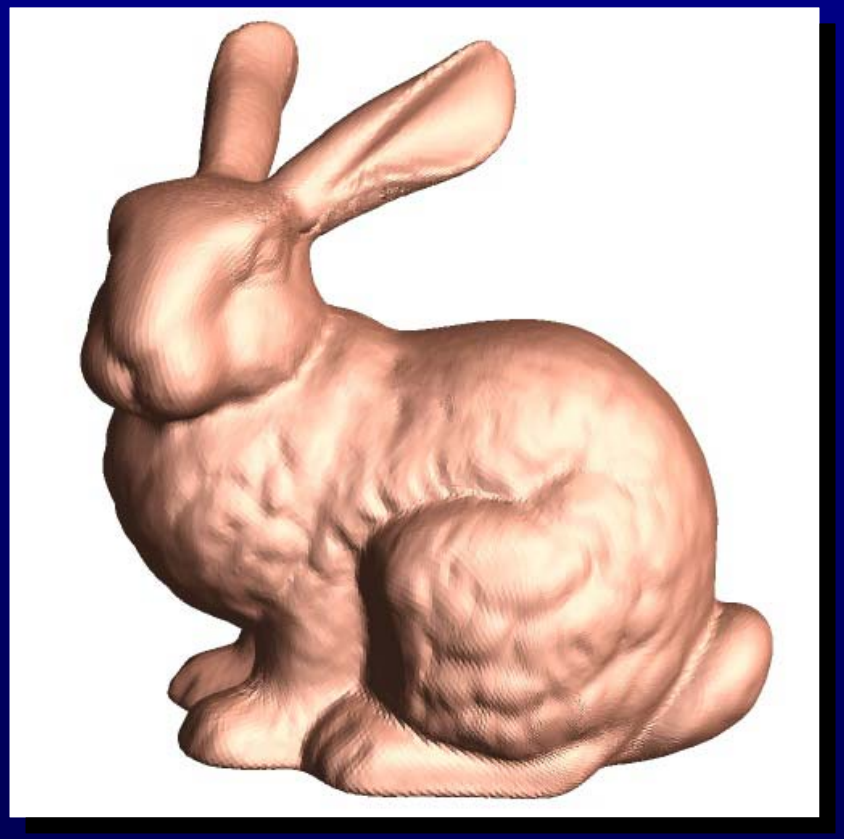
completely regular sampling



geometry image

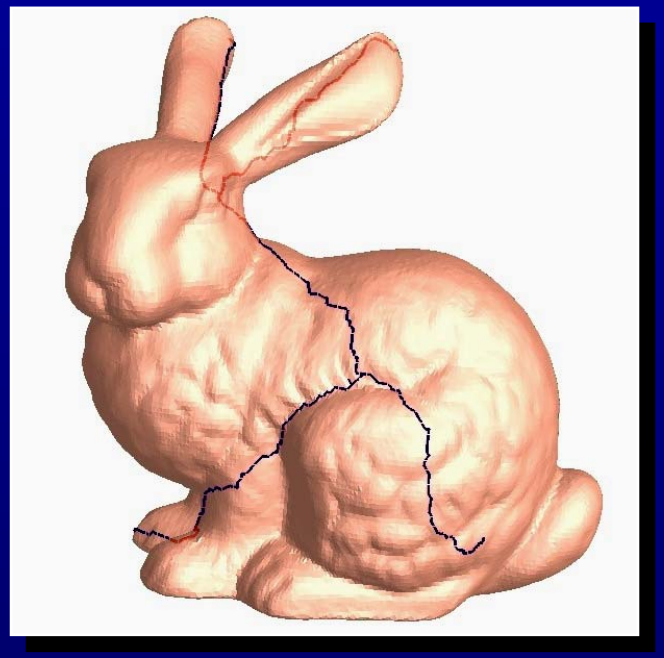
257 x 257; 12 bits/channel

3D geometry

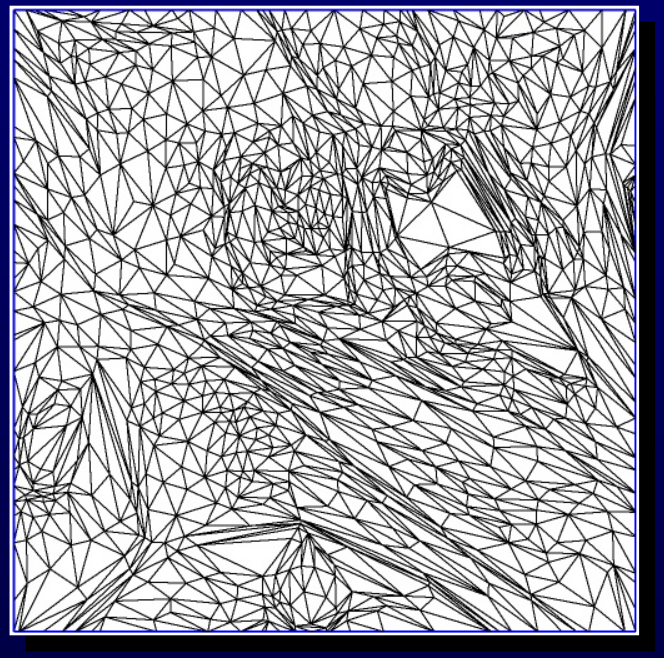


Basic idea

cut



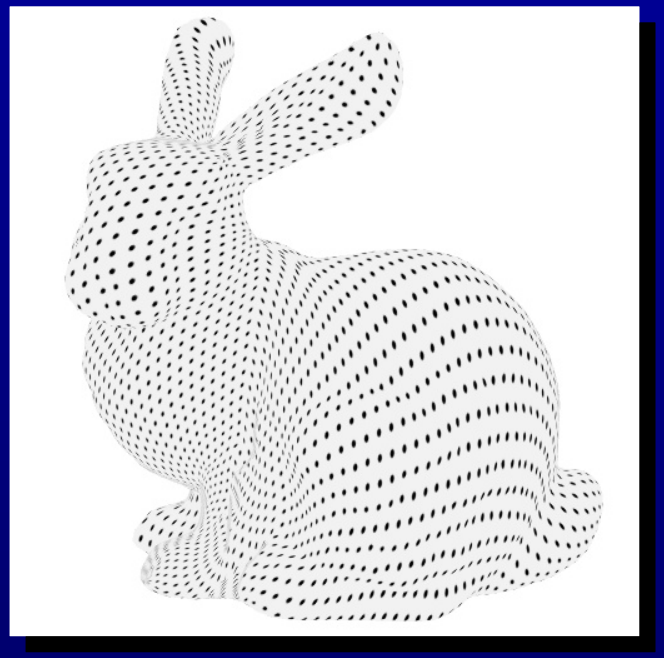
parametrize



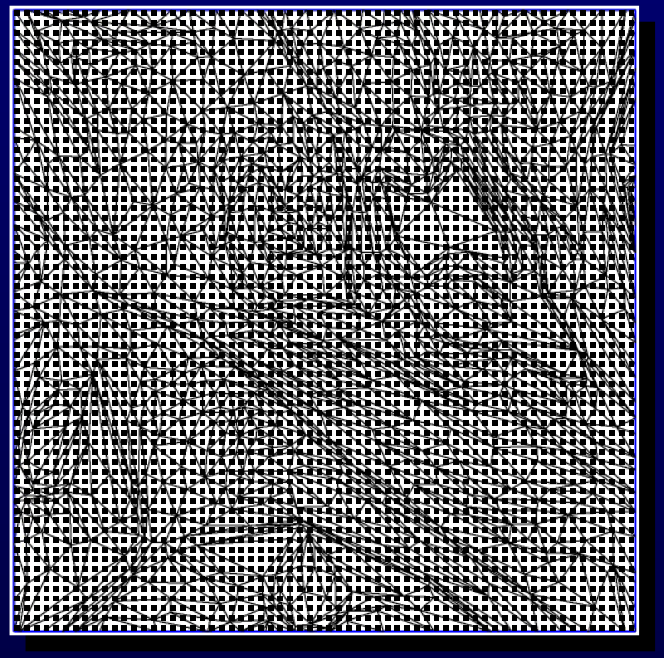
demo

Basic idea

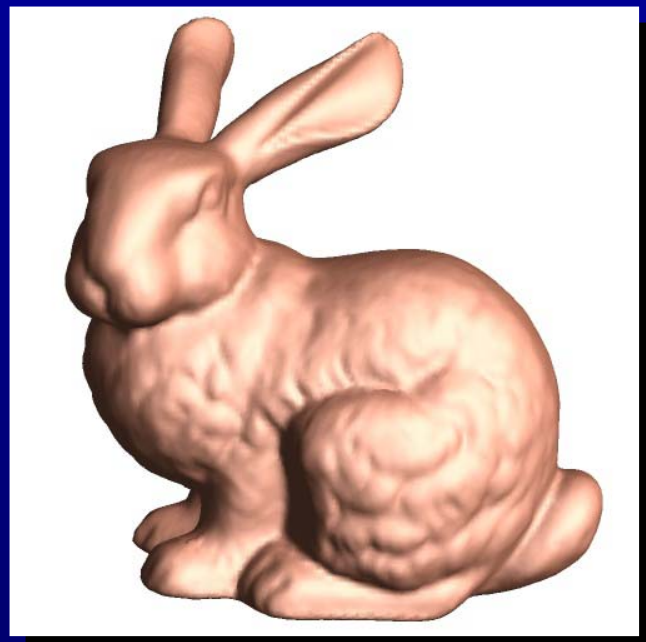
cut



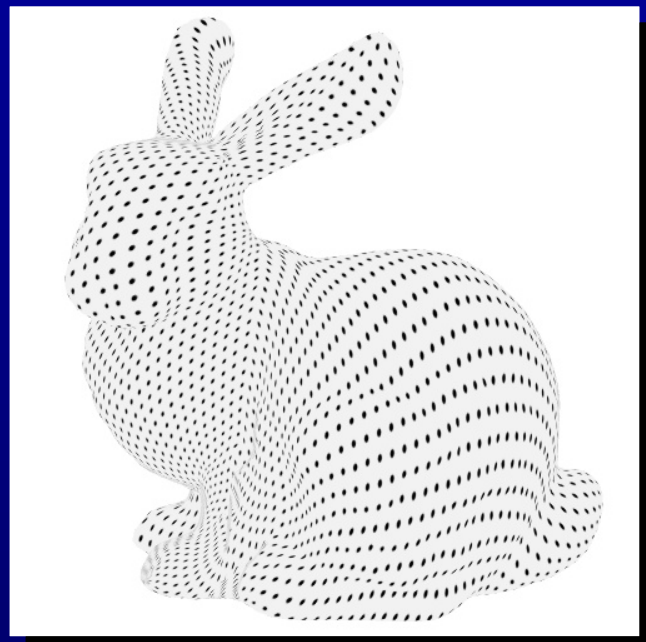
sample



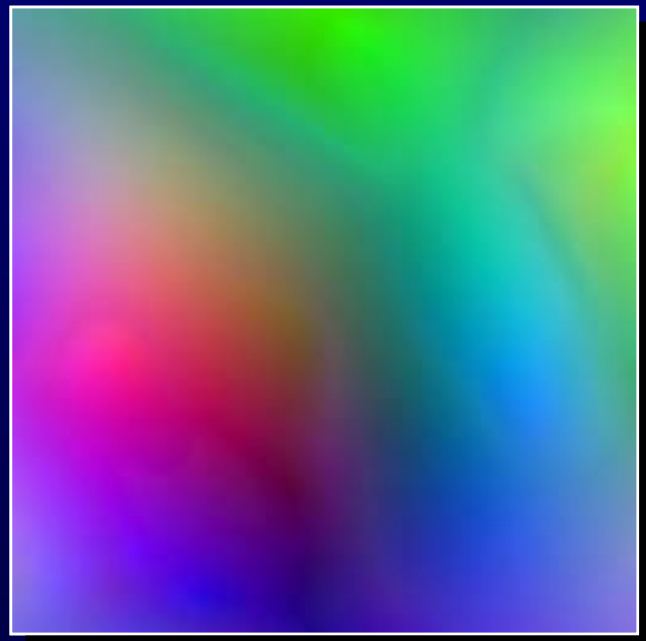
Basic idea



cut

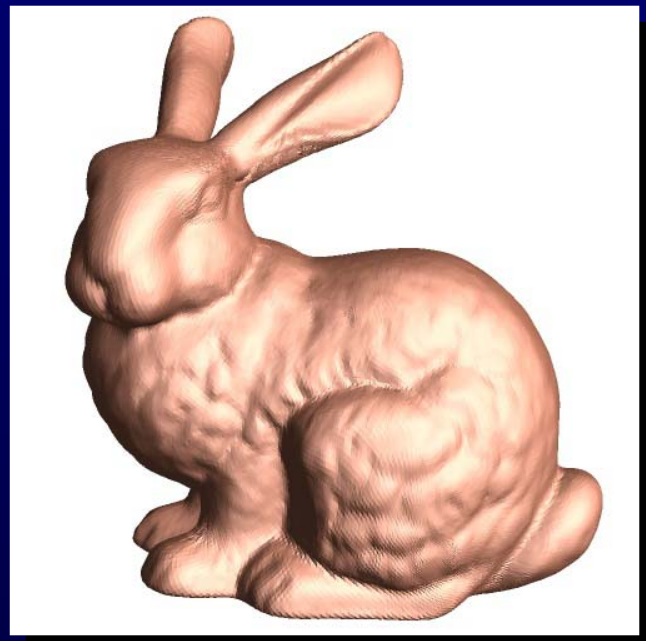


store

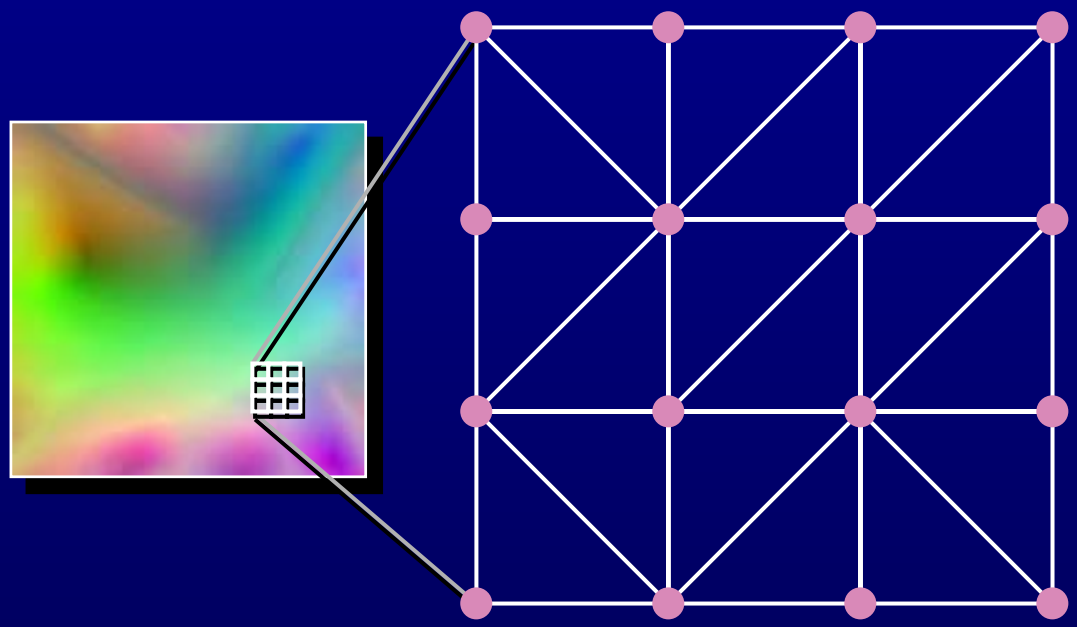


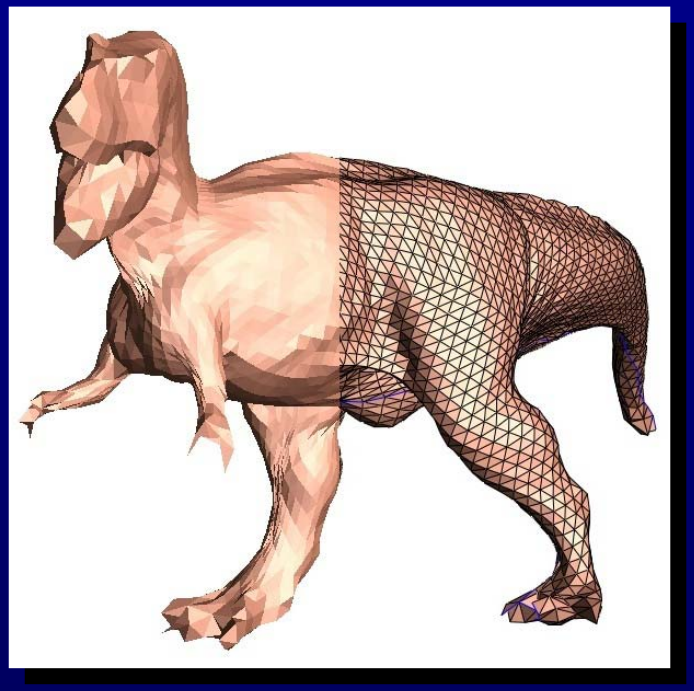
$[r,g,b] = [x,y,z]$

render



Rendering



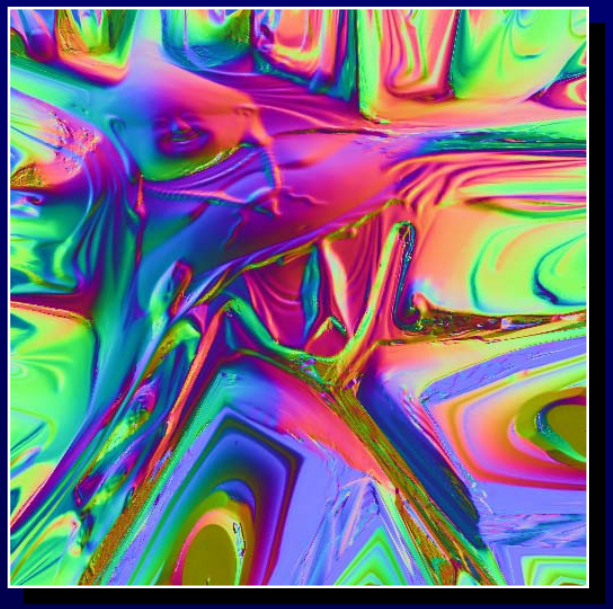


(65x65 geometry image)

demo

Rendering with attributes

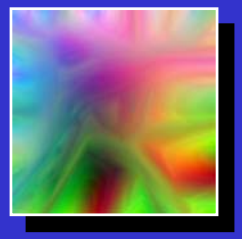
geometry image 257^2 x 12b/ch



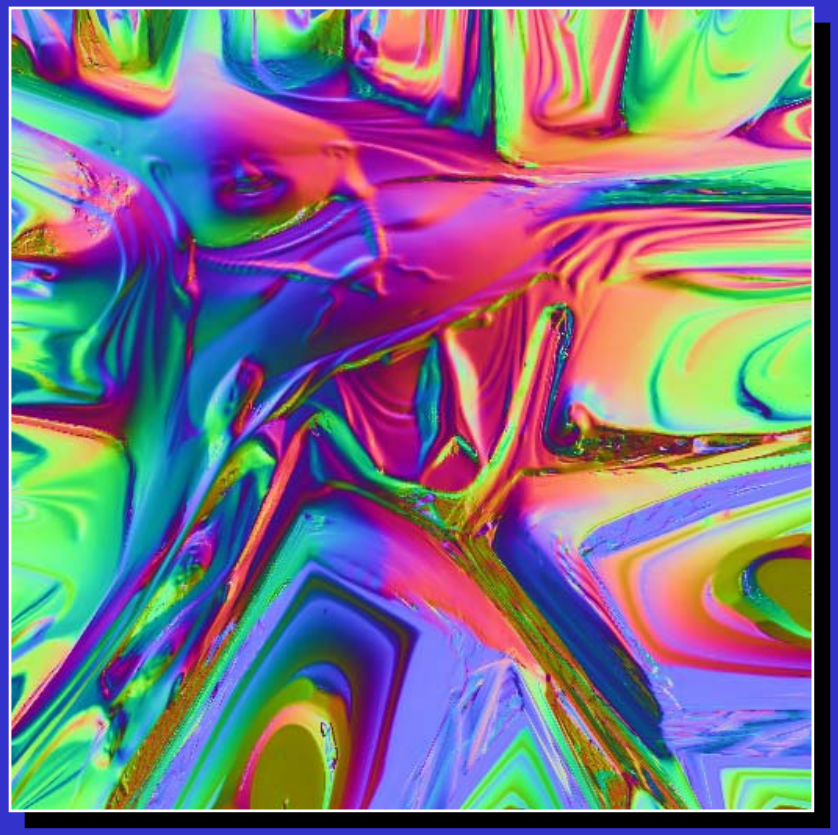
normal-map image 512^2 x 8b/ch

rendering

Normal-Mapped Demo



geometry image
129x129; 12b/ch

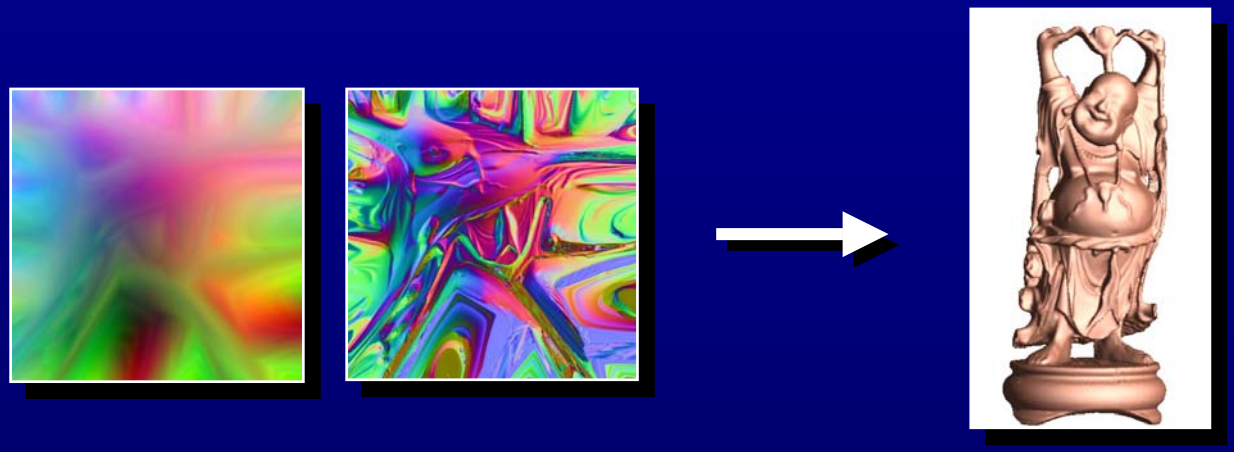


normal map
512x512; 8b/ch

demo

pre-shaded demo

Advantages for hardware rendering

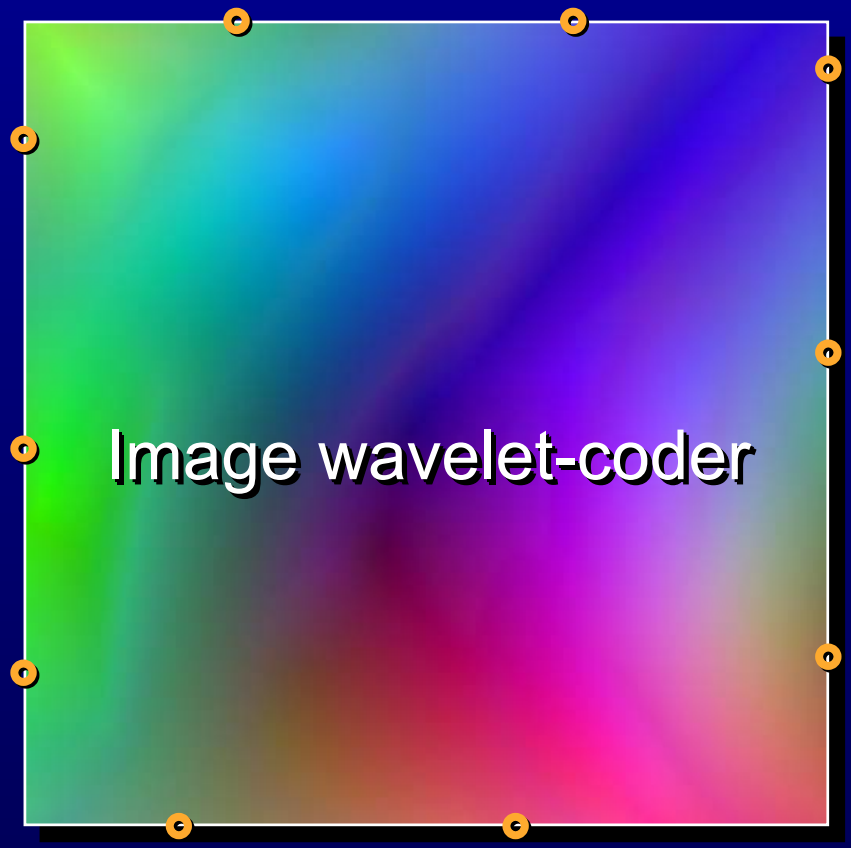


- Regular sampling → no vertex indices.
- Unified parametrization → no texture coordinates.

→ *Raster-scan* traversal of *source* data

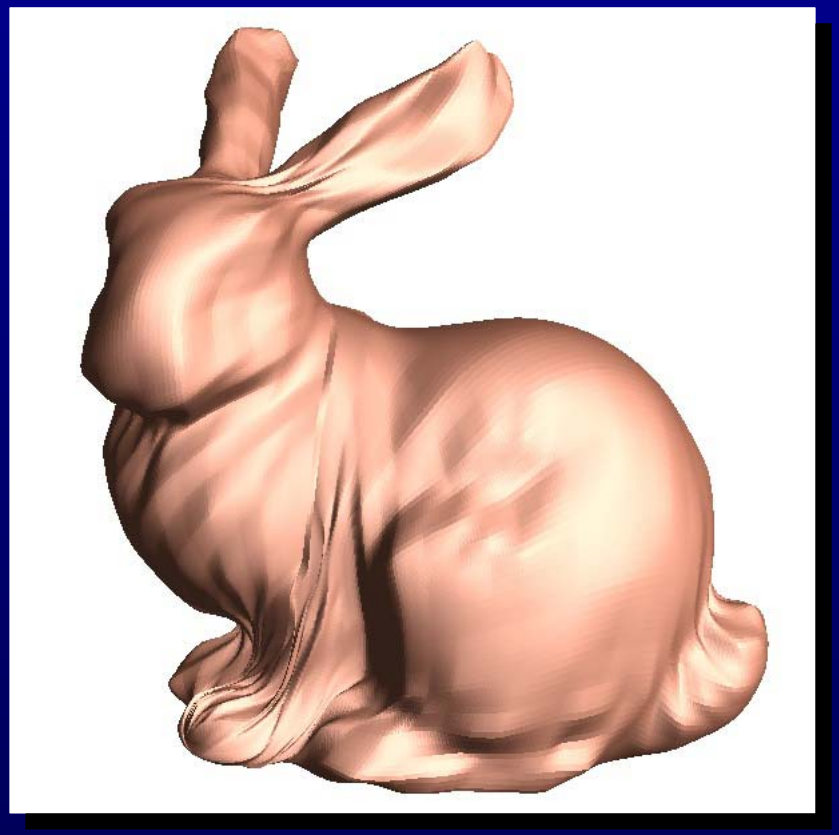
→ Run-time decompression?

Compression



295 KB → 1.5 KB

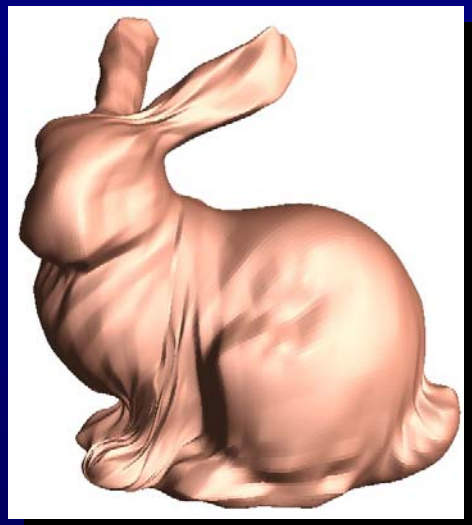
+ topological sideband (12 B)



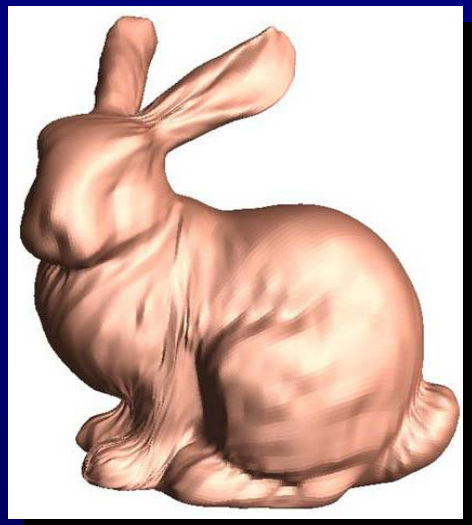
fused cut

Compression results

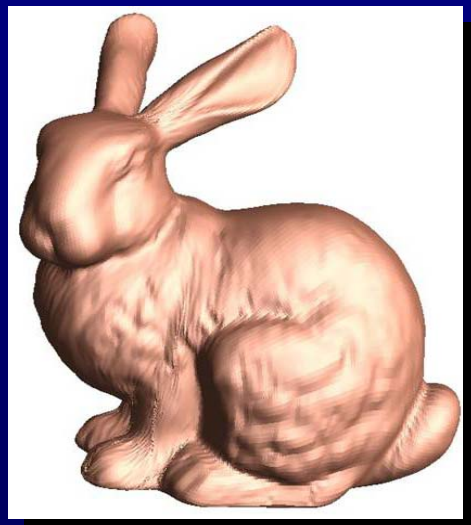
295 KB →



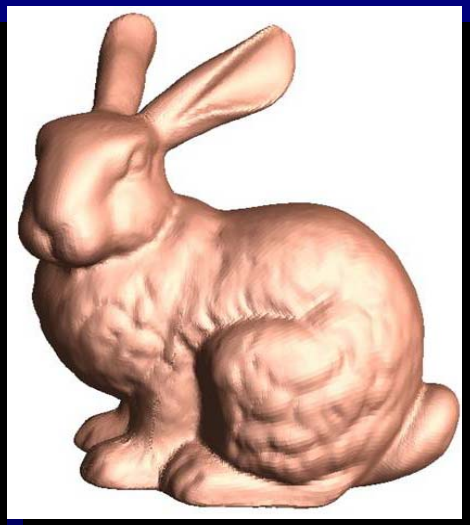
1.5 KB



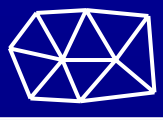
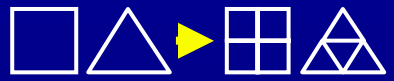
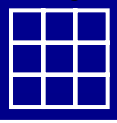
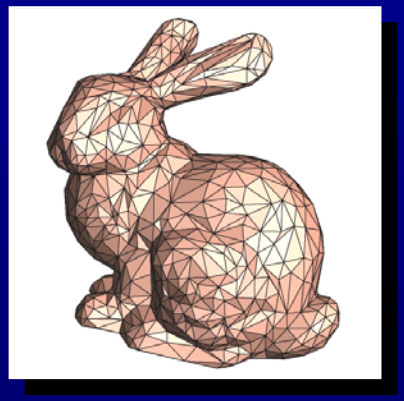
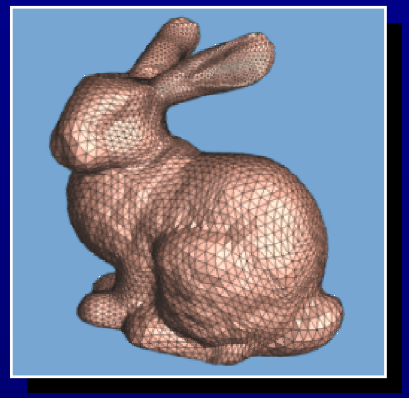
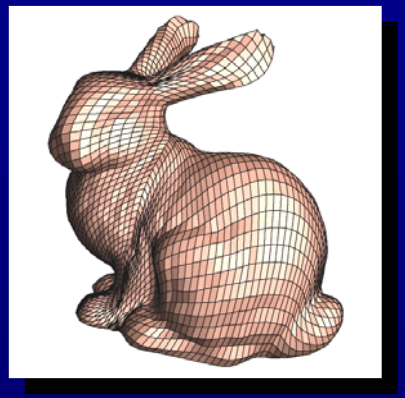
3 KB



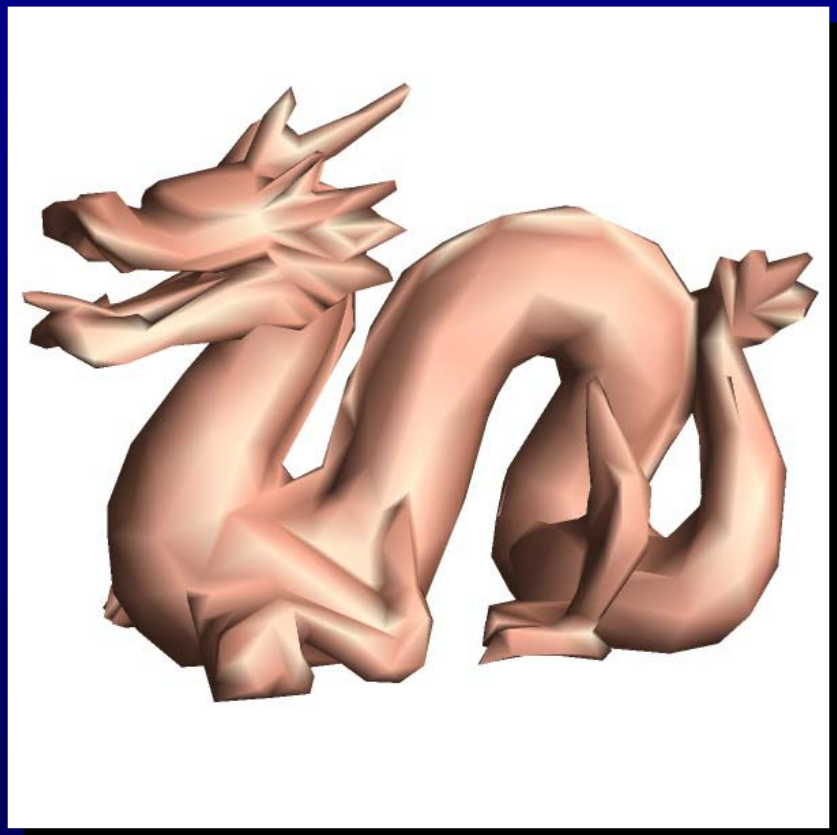
12 KB



49 KB

	Irregular	*Semi-regular*	*Completely regular*
			
			
Flexibility	*any* input mesh	subdivision connect.	uniform grid
Remeshing	unnecessary	required	required
Sharp features	yes	difficult	difficult
Neighborhood / multiresolution	irregular, cumbersome	mostly regular, but irregular vertices	regular, except at “cut”
Rendering	vertex & tex. caching	N-patches	simple raster scan
Compression	poor, delta-encoding	fancy wavelets, software	easy wavelets, hardware?
Element quality	good if desired	good if desired	trouble areas

Texture Mapping Demo

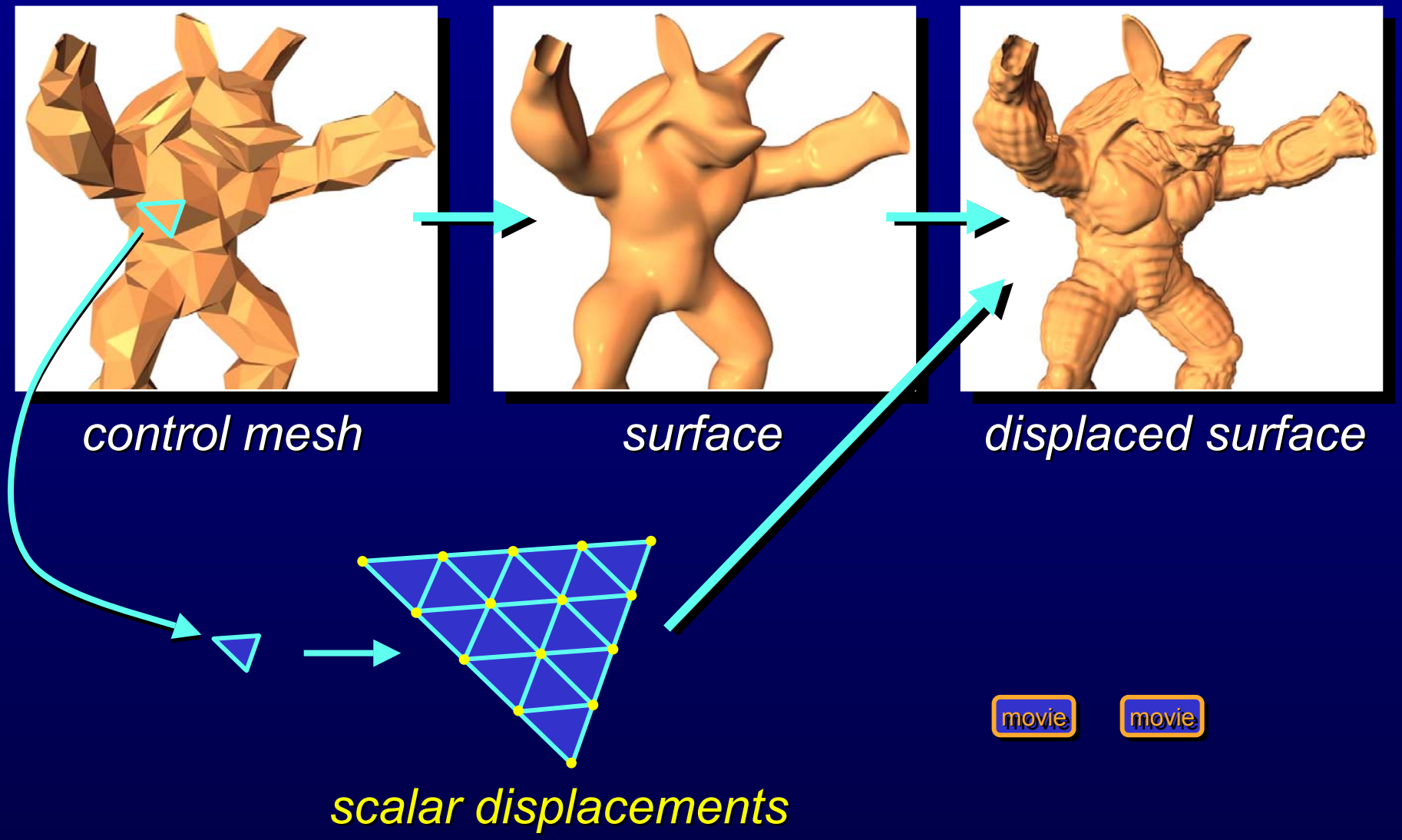


2,000 faces

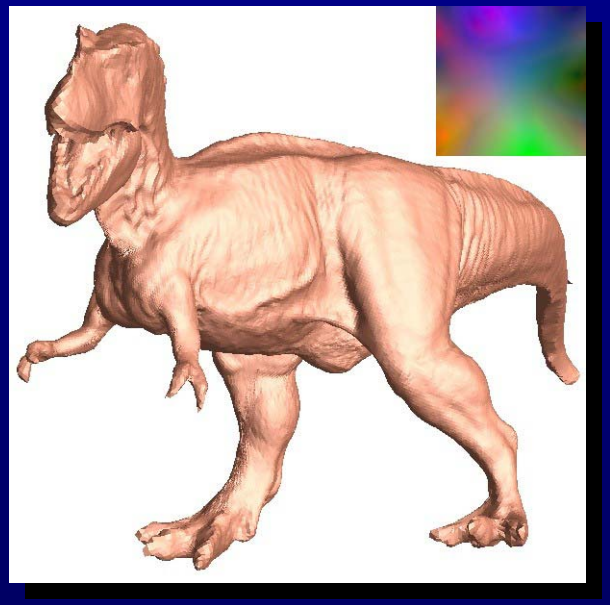
demo

Displaced subdivision surfaces

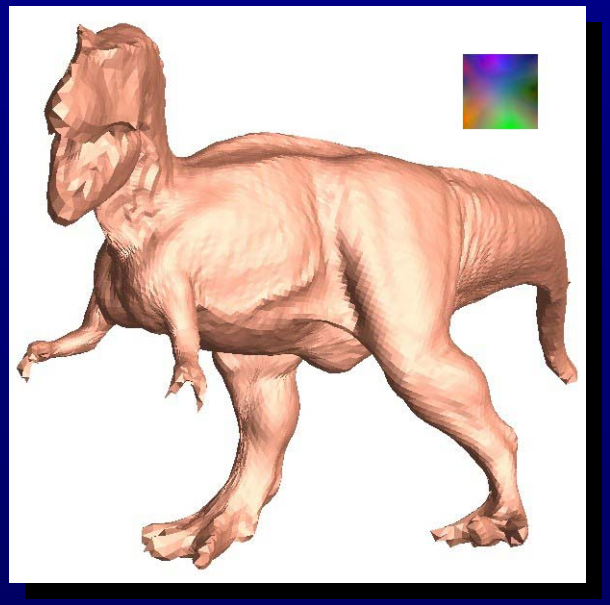
[Lee et al 2000]



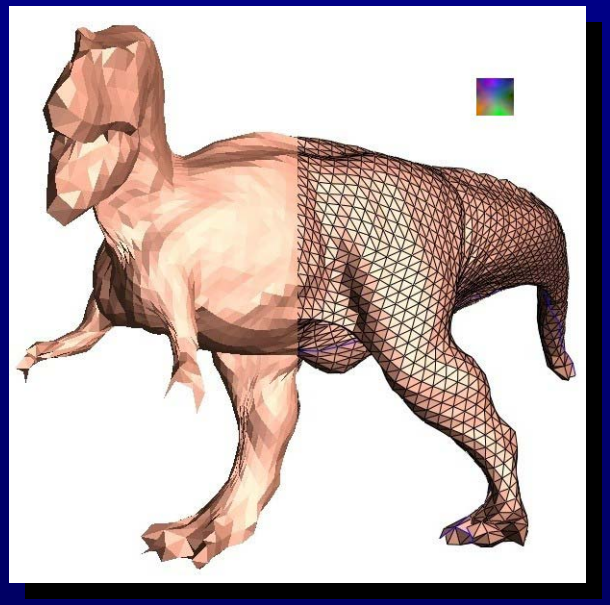
Mip-mapping



257x257

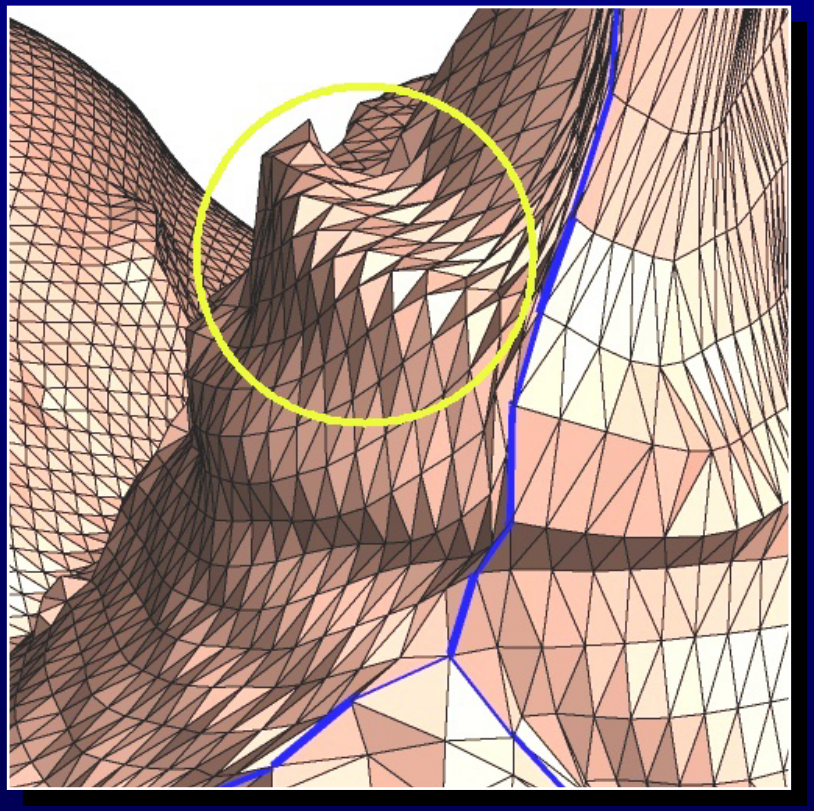


129x129

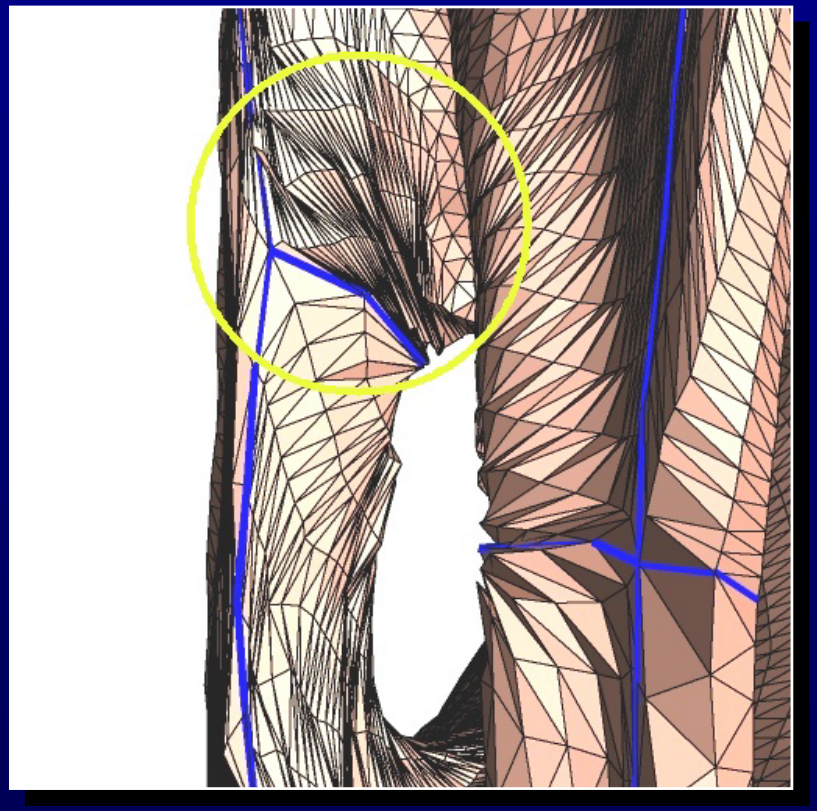


65x65

Some artifacts



aliasing



anisotropic sampling