Real-Time High-Quality View-dependent Texture Mapping using Per-Pixel Visibility

Damien Porquet
Jean-Michel Dischler
Djamchid Ghazanfarpour

MSI Laboratory, Limoge University,
LSIIT, Strasbourg University,
France

Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Introduction 1/2

Domain : Real-time Rendering

Context : Real-time rendering of complex meshes

Problematic : Geometrical complexity (i.e. visual accuracy) versus computation time

Our approach : hybrid- image based & geometry rendering methods

The method :

Combine a simplified mesh and reference images of the original mesh to speed-up rendering time

Introduction 2/2

Related work :

View-dependent Texture Mapping [Debevec *et al.* 96,98]

► no blending of reference images

► no subdivisions to avoid blurring

Appearance Preserving Simplification [Cohen *et al.* 98]

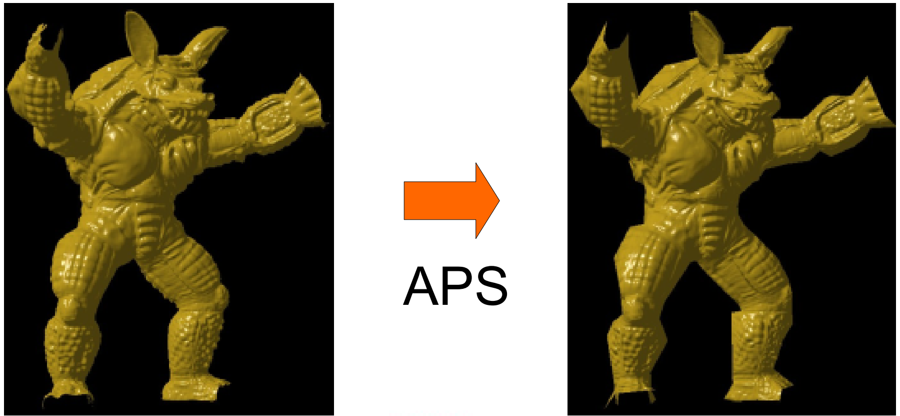
► no need to compute bump-map for each LOD

Relief Mapping [Oliveira *et al.* 00, Policarpo *et al.* 05]

► no parametrization

► relief not restricted to height field

VDTM



APS

Relief Mapping

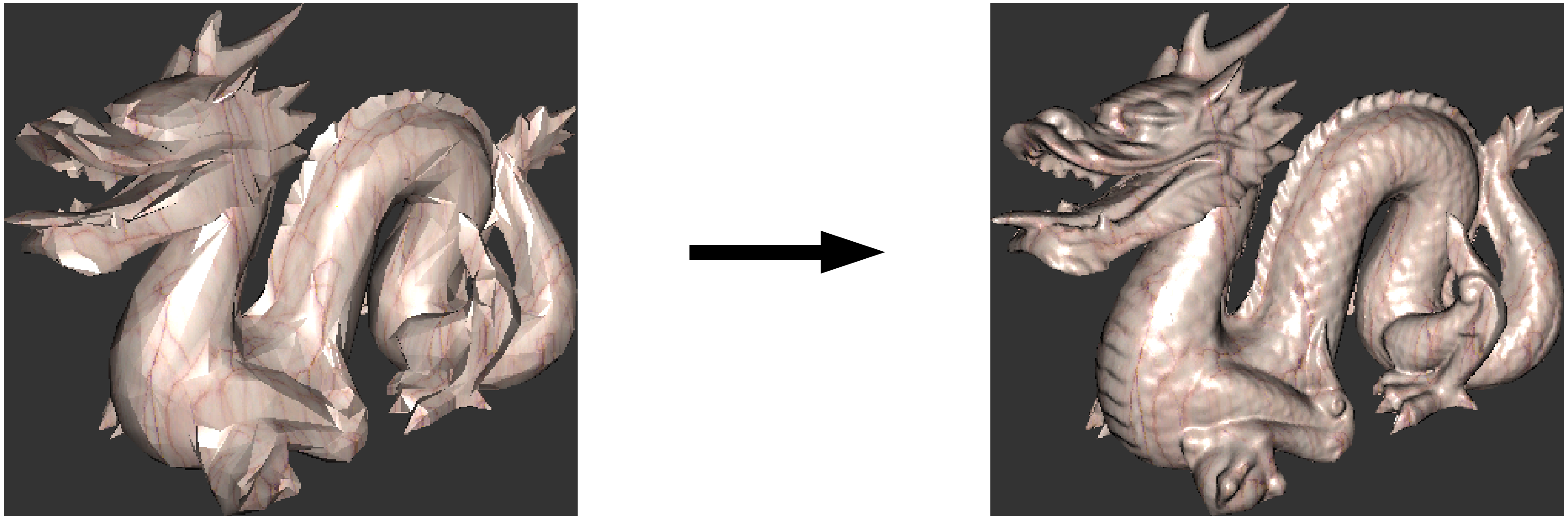
Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Overview of the method

The method consists in

- Replacing a complex mesh laid out in a scene with a geometrically simplified version of it ...
- ... and map it with full relief stored in reference images



Overview of the method

The method consists in

- Replacing a complex mesh laid out in a scene with a geometrically simplified version of it ...
- ... and map it with full relief stored in reference images

Precomputation step

  - Given a complex object, compute its low polygon count version
  - Grab viewpoints of the original mesh

Rendering step

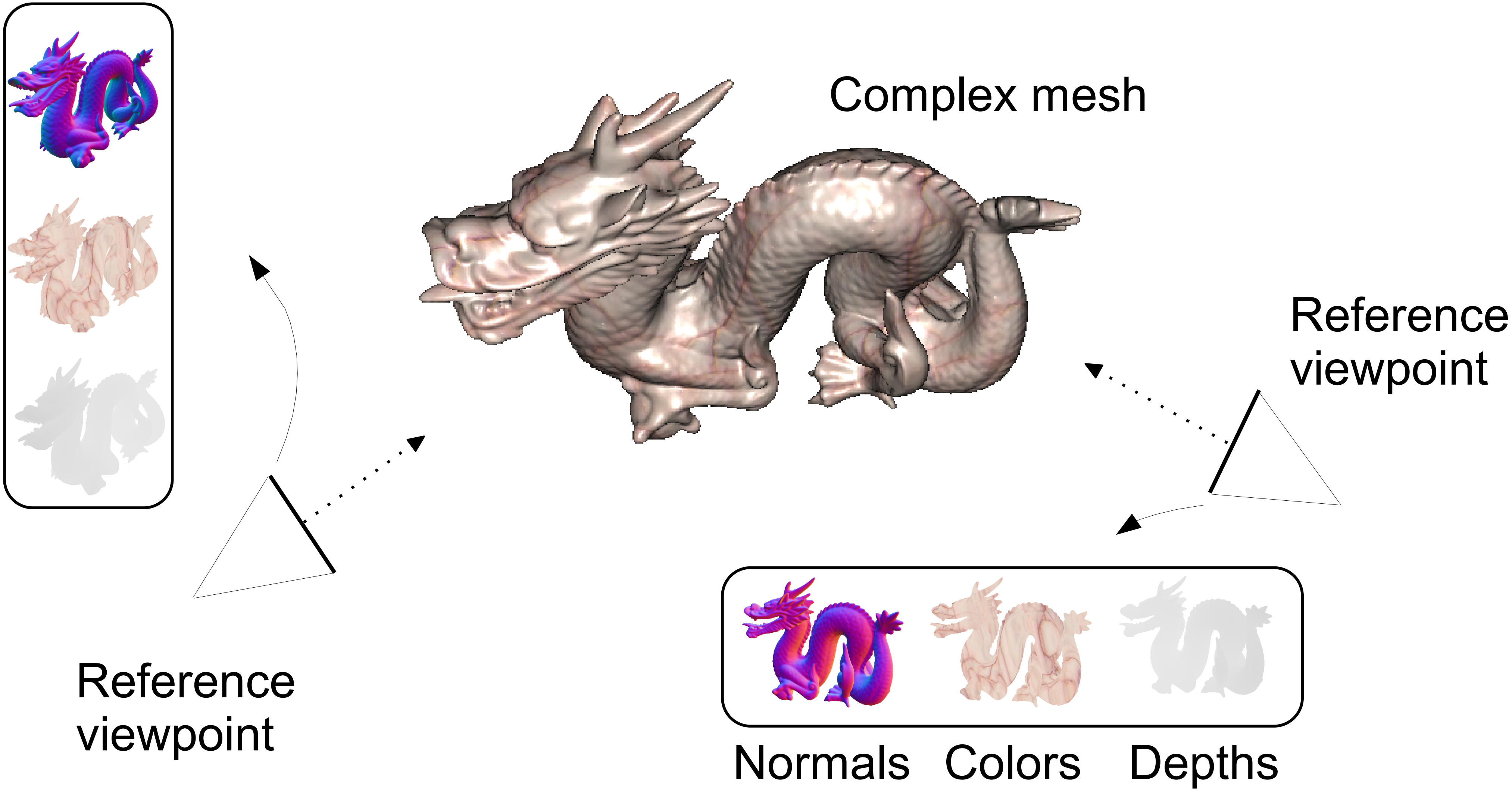
  - Select best viewpoints
  - Map lost relief onto the simplified surface

Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Precomputation step 1/2

Reference images acquisition



Complex mesh

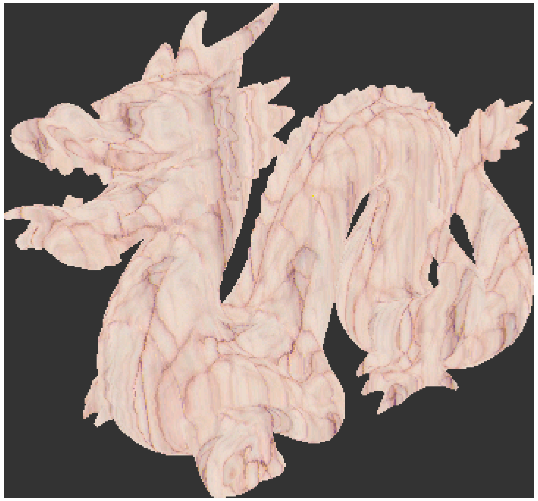
Reference viewpoint

Reference viewpoint

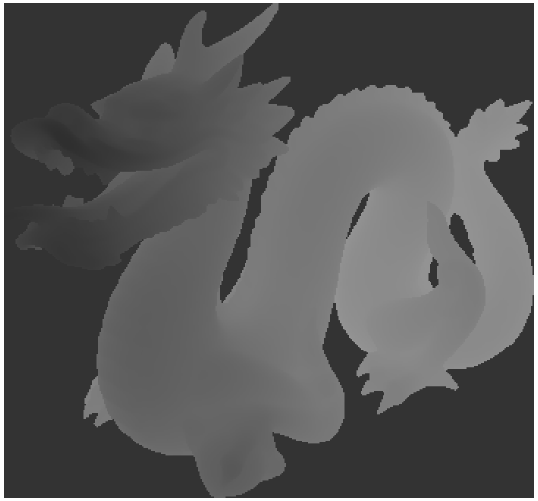
Normals Colors Depths

Precomputation step 2/2

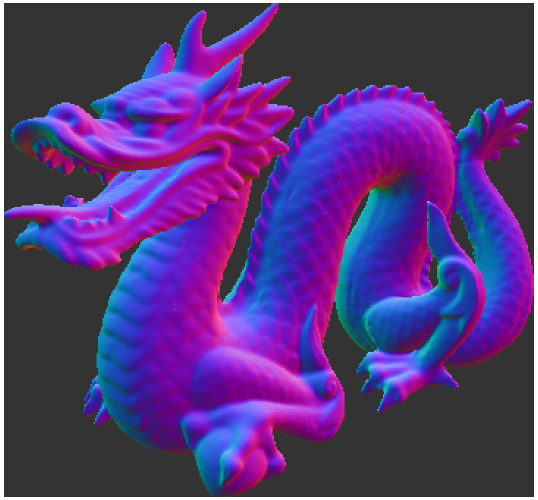
Reference viewpoint



Colors map
RGB 24 bits



Depths map
16 bits



Normals map
RGB 24
(scaled to [-1,1])

Texture resolution : 512x512
Data size without optimization : 2 Mb per viewpoint
Optimization : 1 Mb/view using indexed colors and normals textures

Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Real-time rendering 1/5

Two steps:

1. CPU : Selection of three best viewpoints

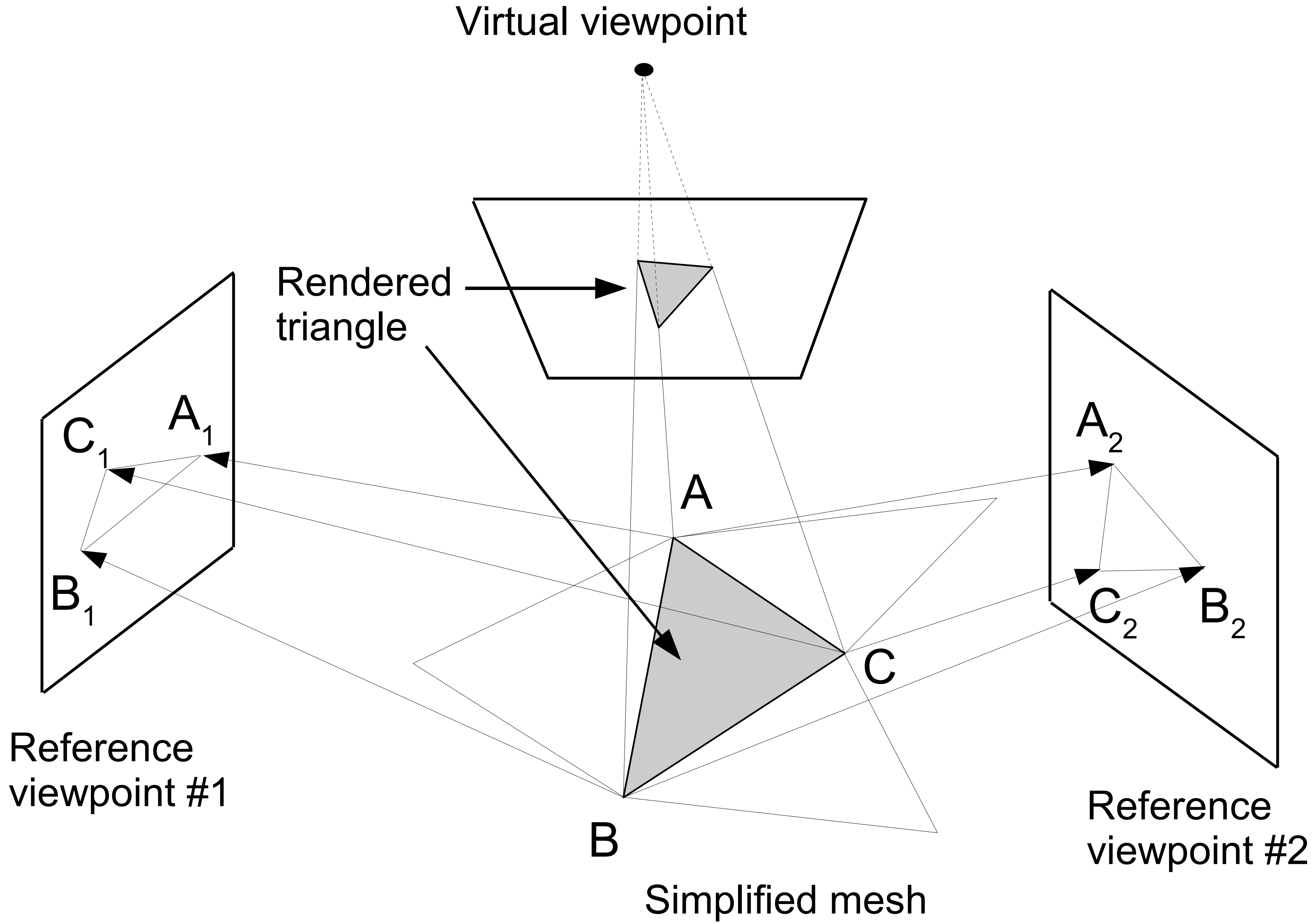
For a given virtual eye position we select the three closest viewpoints among the reference viewpoints set

=> We consider that these three viewpoints covers all of the object surface

2. GPU : Determination of the best viewpoint

We extract one pixel from each of the three selected viewpoints, and determine the best

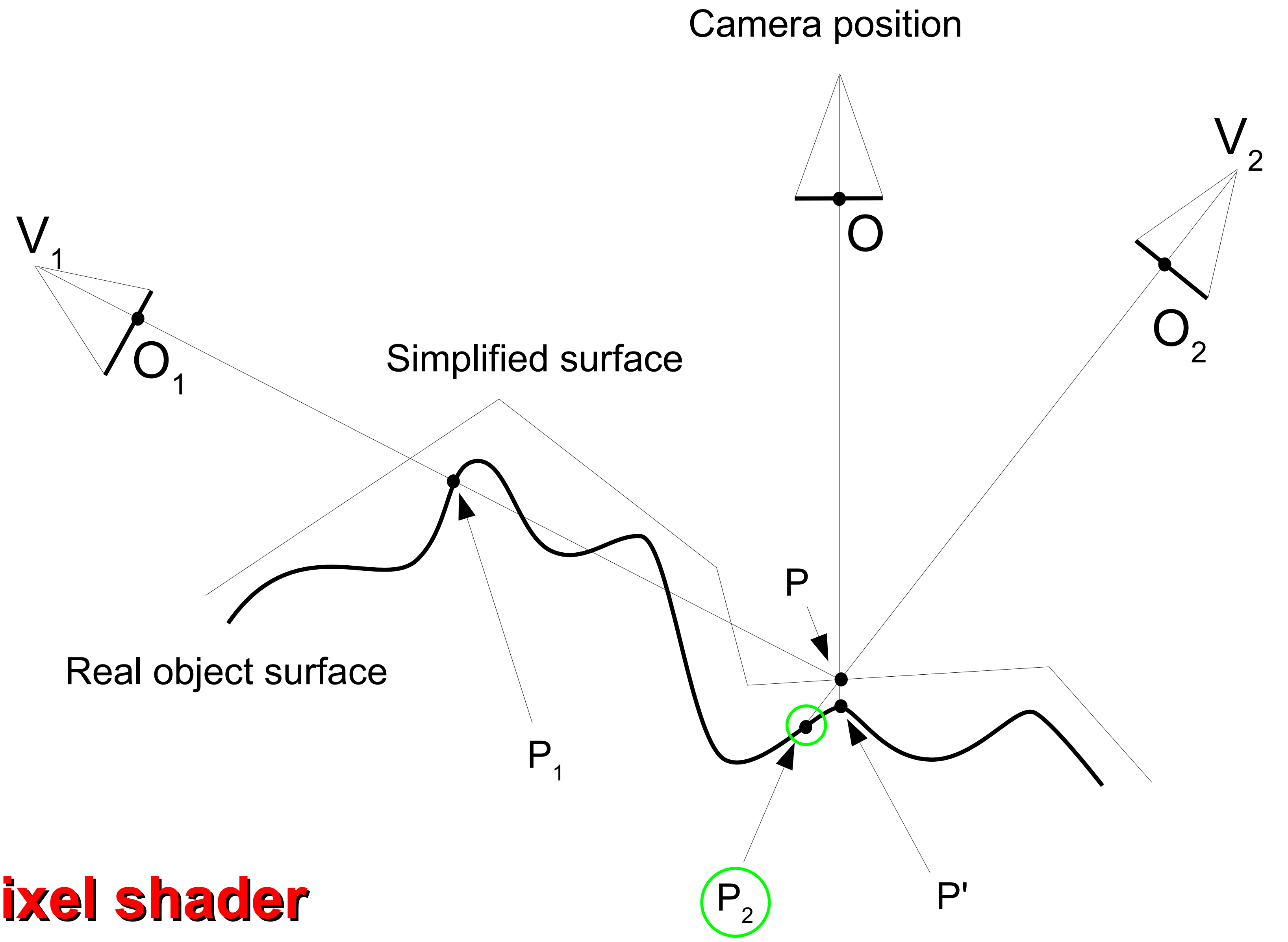
Real-time rendering 2/5



Vertex shader

Real-time rendering 3/5

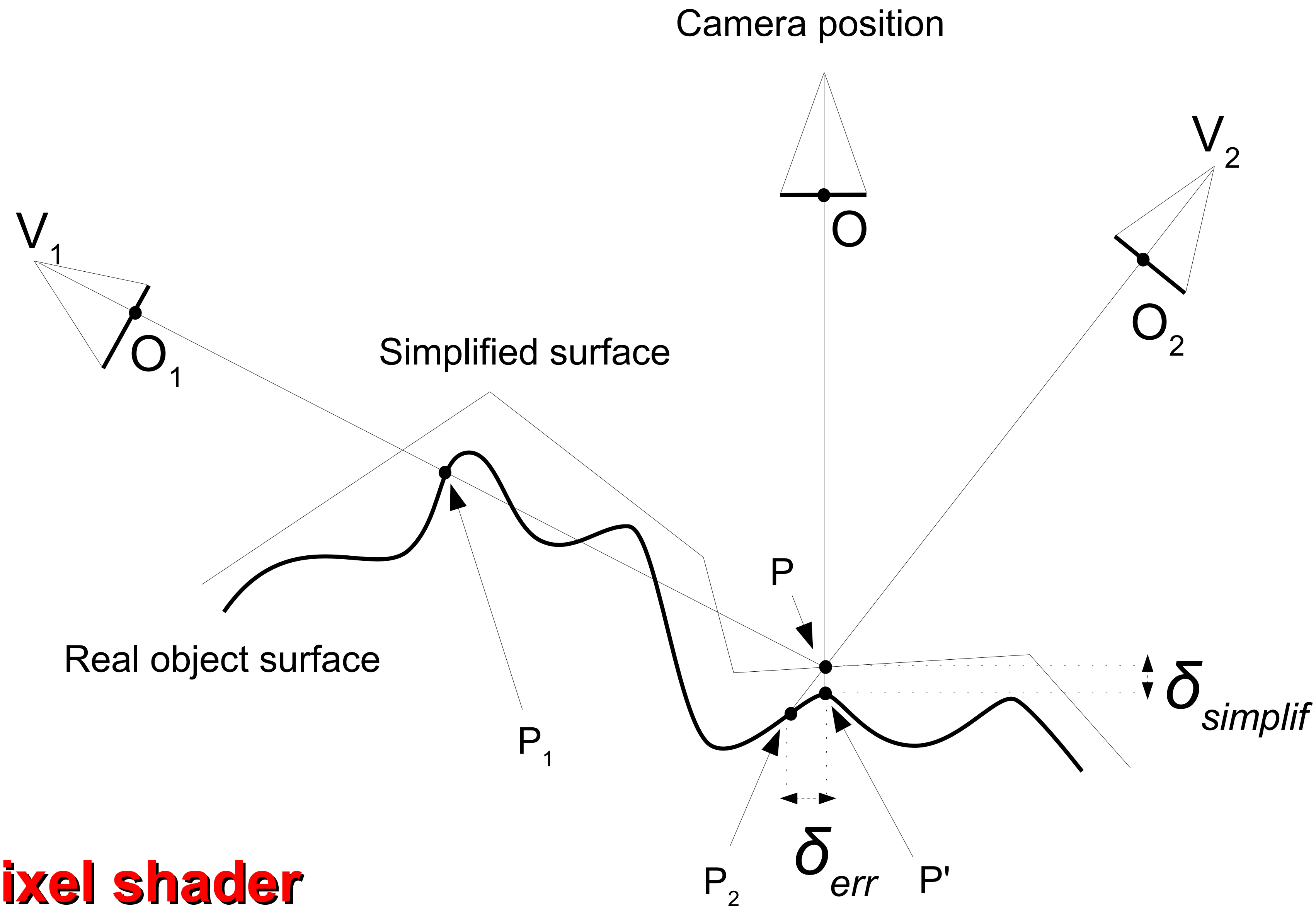
Per-pixel visibility determination



Pixel shader

Real-time rendering 4/5

Per-pixel visibility determination



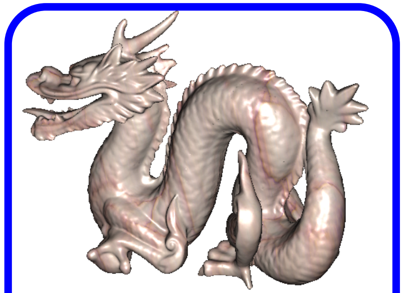
Pixel shader

Real-time rendering 5/5

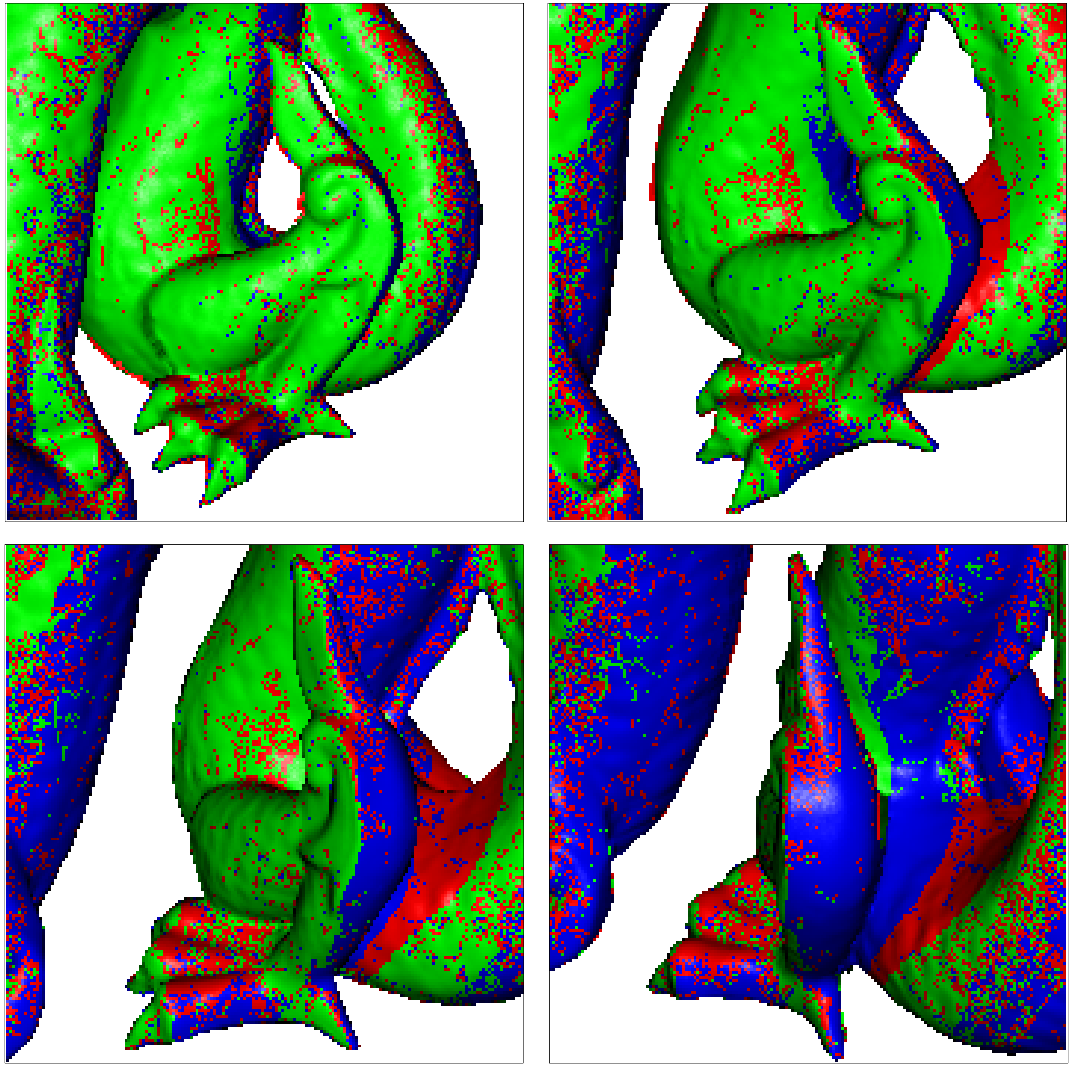
Visibility determination: results

viewpoint #1

Viewpoint #2



Viewpoint #3



Plan

1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Results

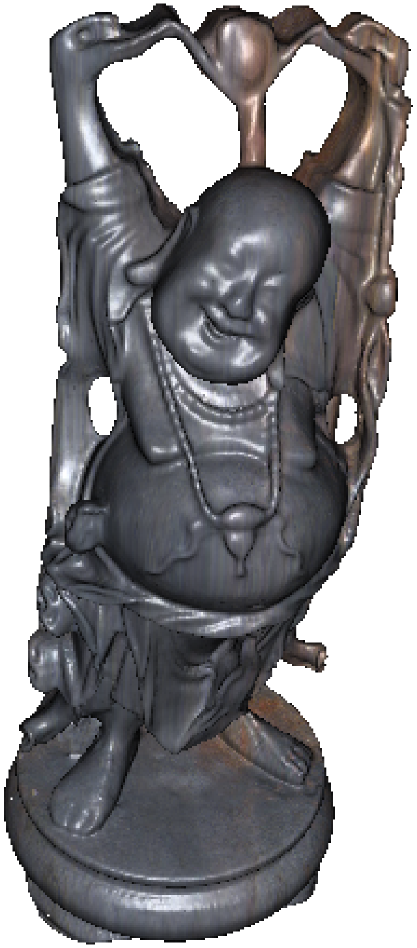
- Hardware
  - CPU: 1.8 GHz
  - Graphic card: NVIDIA GeForce FX6800GT

- Vertex & fragment programs are written in Cg.

- Textures: 512x512 RGB

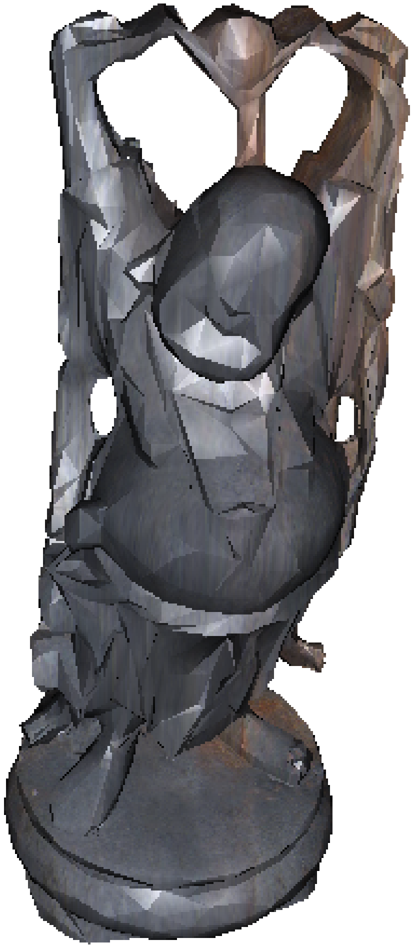
- Resolution: 512x512

Results

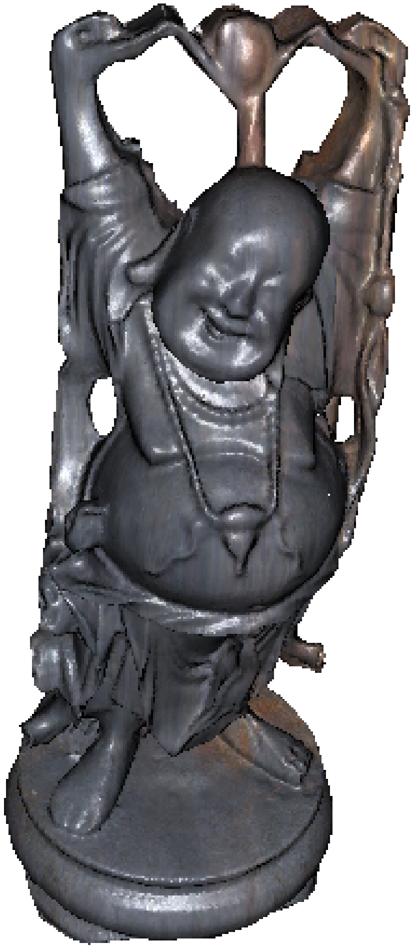


Original mesh
1.1 M triangles

30.2 FPS



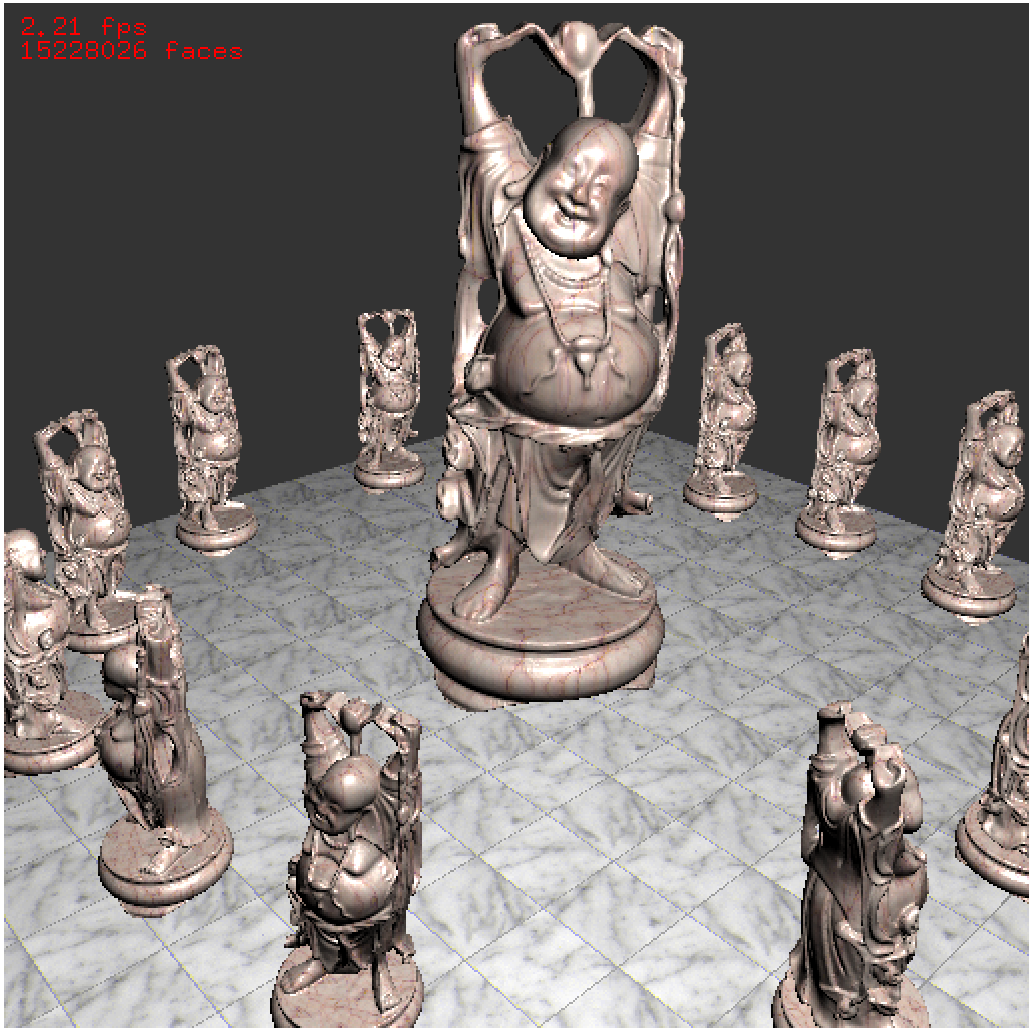
Simplified mesh
6 K triangles



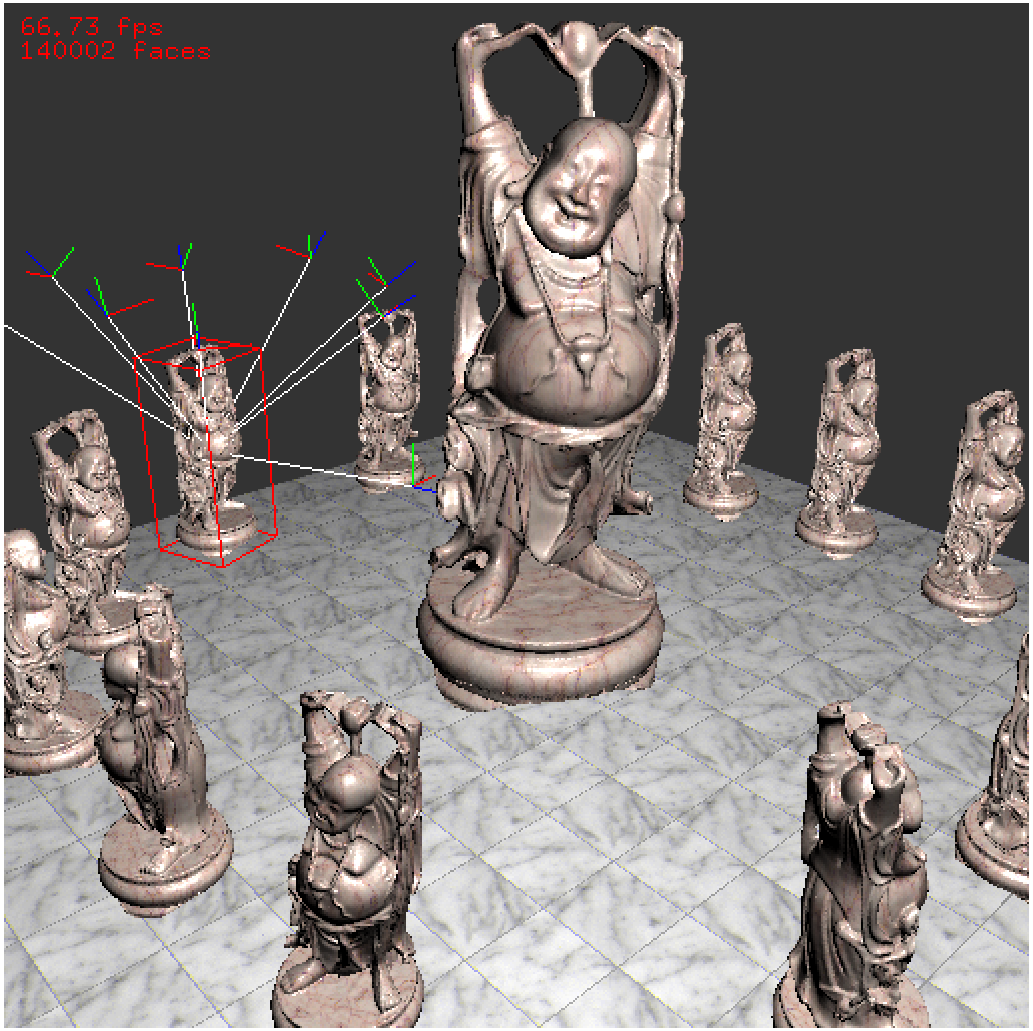
Simplified mesh
+ our method

606 FPS

Results



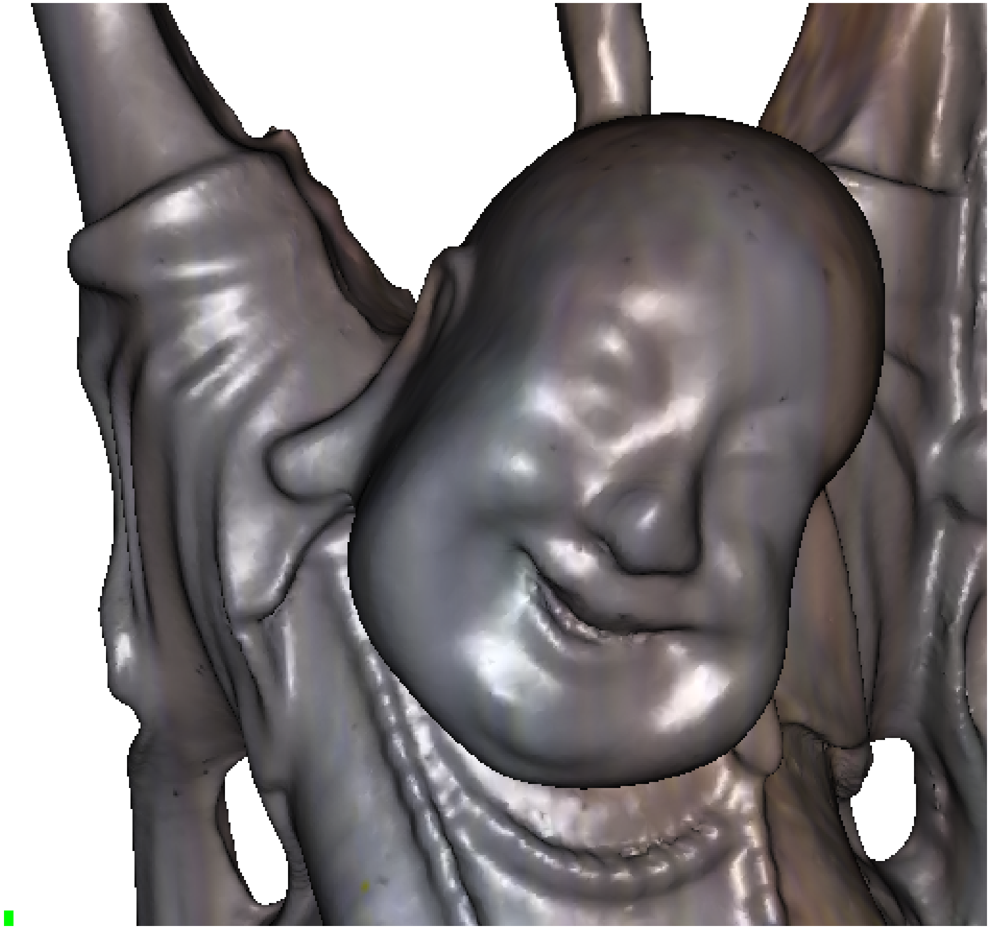
Original scene
15 M triangles
2.21 FPS



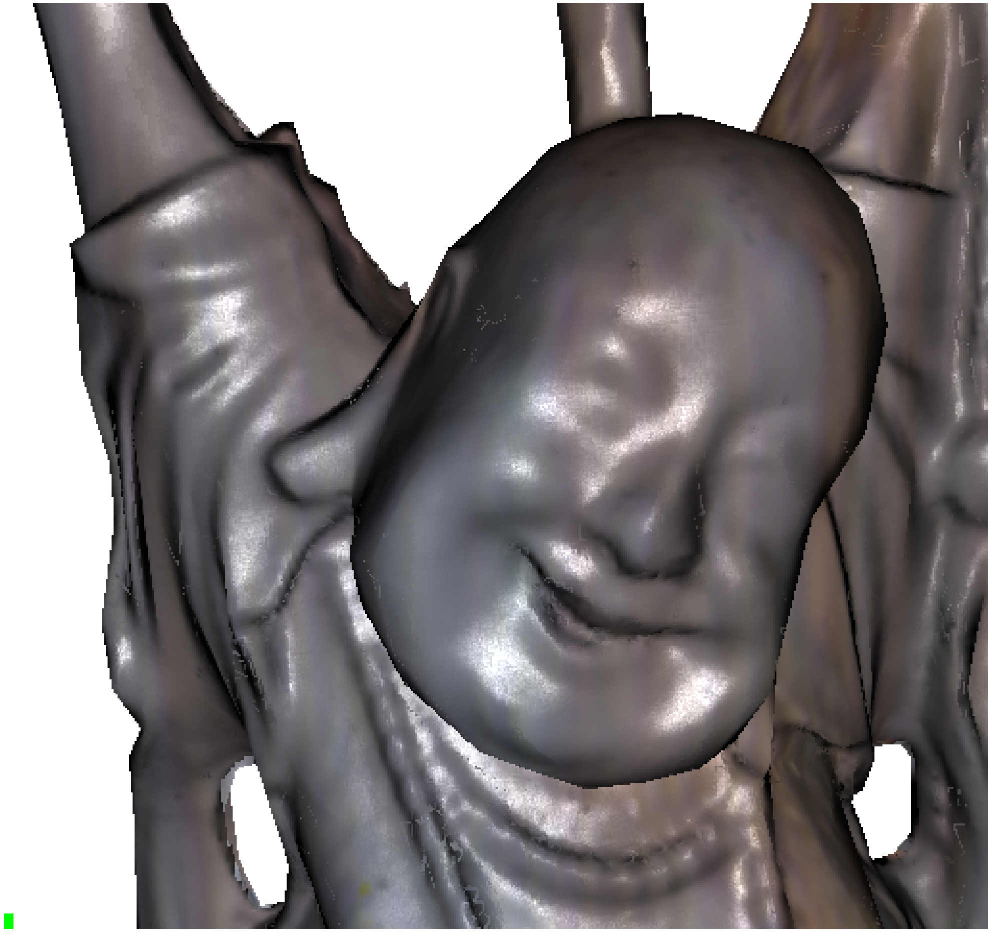
Reconstructed scene
140 K triangles
66.73 FPS

Results

Close view



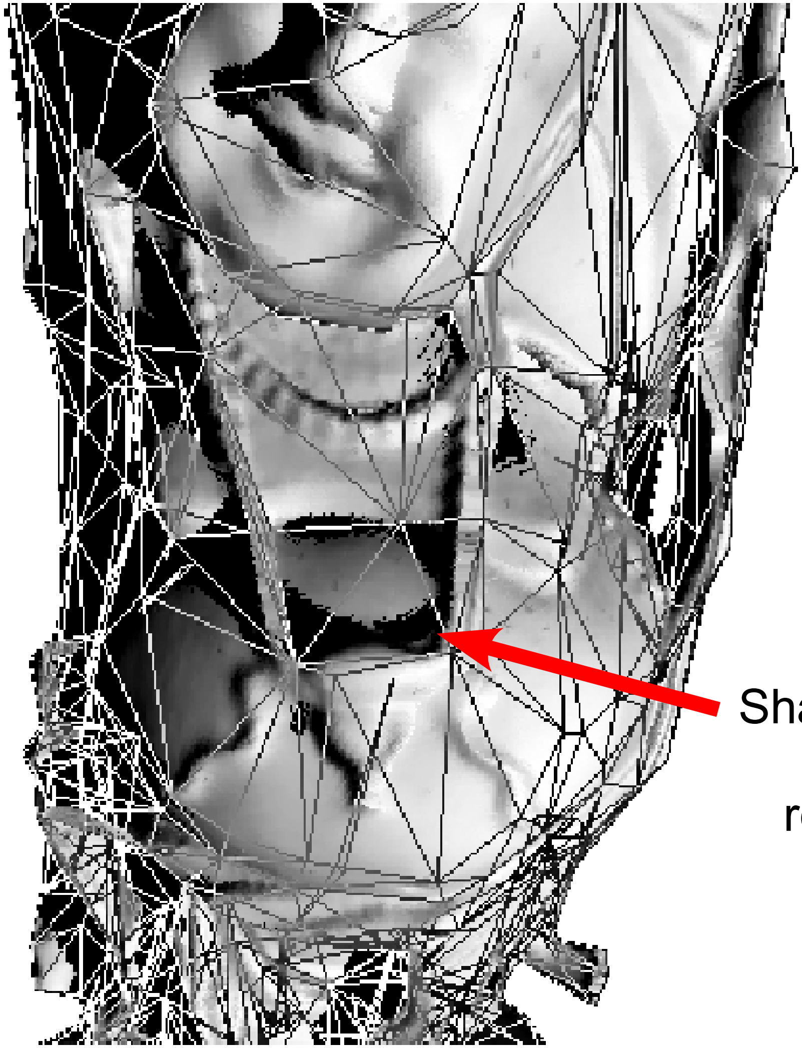
Original mesh



Simplified mesh

Results

Shadow projection



Shadow is “deformed” on flat triangles because we reconstruct original relief

Simplified mesh = 2000 faces

Results

Videos

Plan

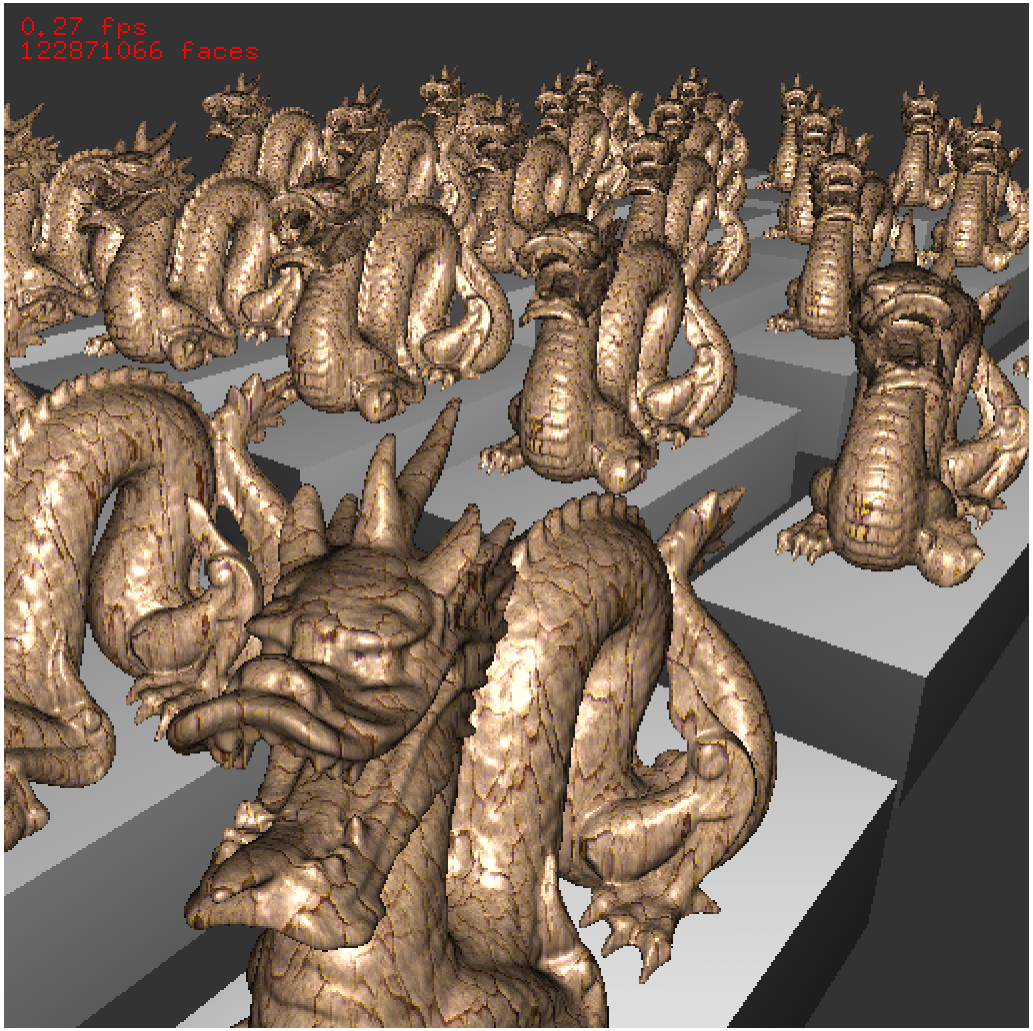
1. Introduction
2. Overview of the method
3. Precomputation step
4. Real-time rendering
5. Results
6. Conclusion & future work

Conclusion & future work

- This paper presented a technique for real-time rendering of complex 3D objects using view-dependent texture mapping approach
- We combine reference images of a mesh and geometrical simplification to speed-up rendering for a little lost of visual accuracy
- Mapping done in object space
  - No surface parametrization needed
- Future Work
  - Add details to object's silhouettes
  - Automatic selection of best viewpoints
  - Reduce relief deformation

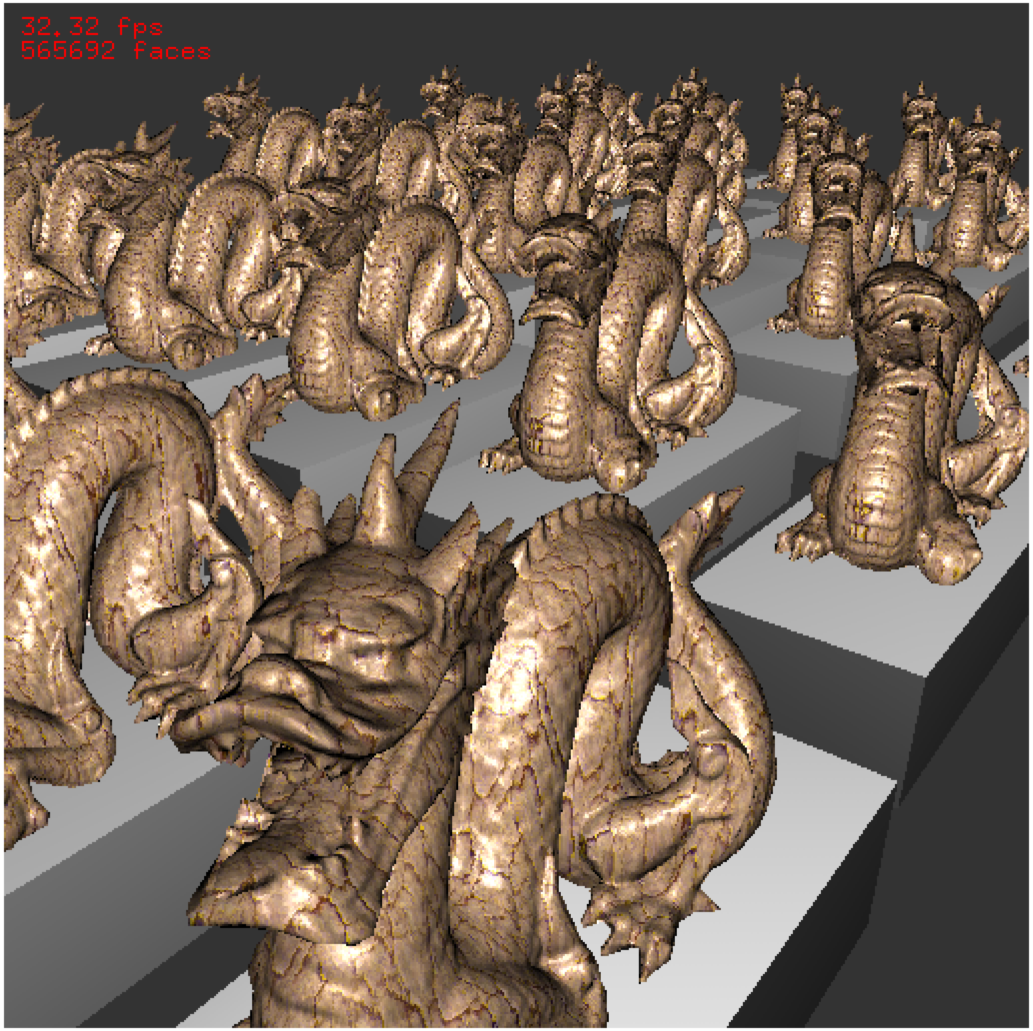
Questions ?

Results



```
0.27 fps
122871066 faces
```

Original scene
122 M triangles
0.3 FPS



```
32.32 fps
565692 faces
```

Reconstructed scene
565 K triangles
32 FPS