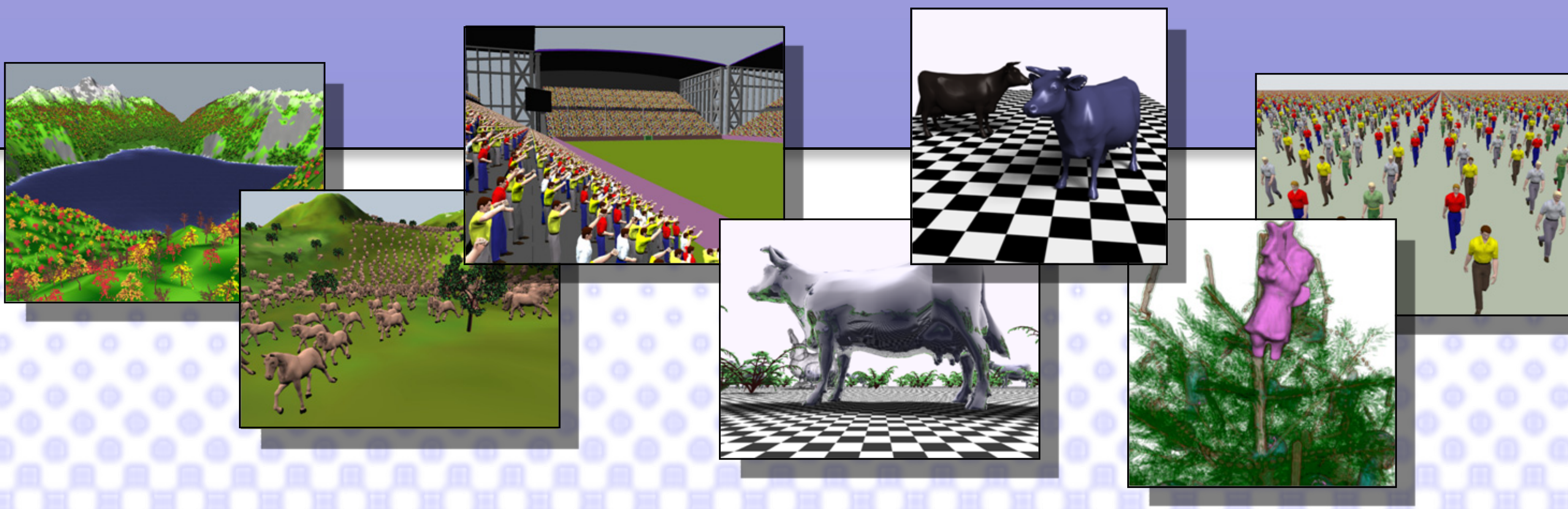
Point-Based Multi-Resolution Rendering



Overview

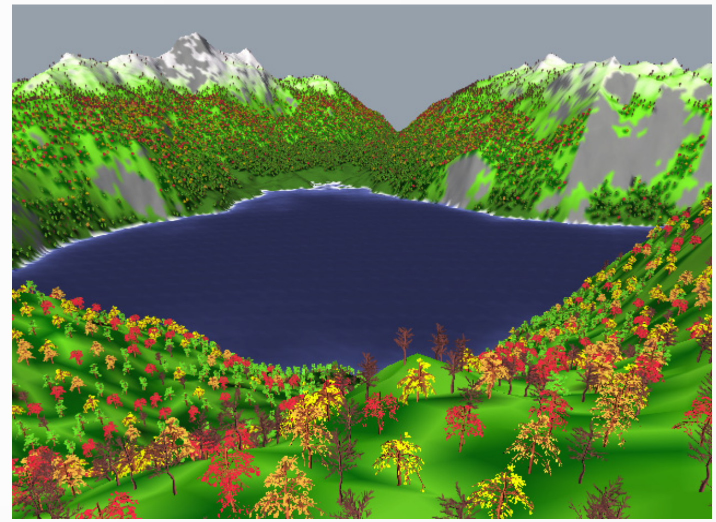
Overview:

- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions

Overview

Overview:

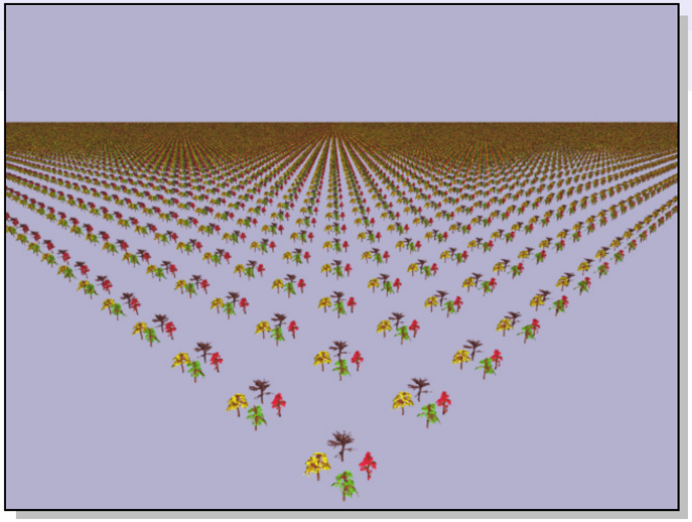
- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions



Motivation

10^6 triangles

10^8 triangles

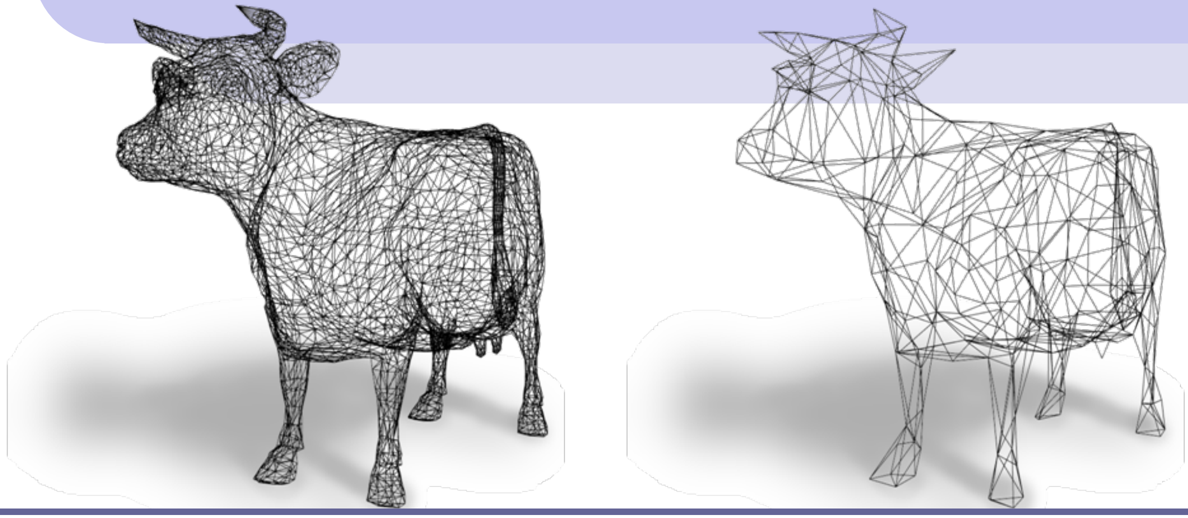


10^{14} triangles

Highly Complex Scenes

- Billions of primitives
- Interactive rendering (CAD, visualization, games...)
- No general solution

Mesh Simplification



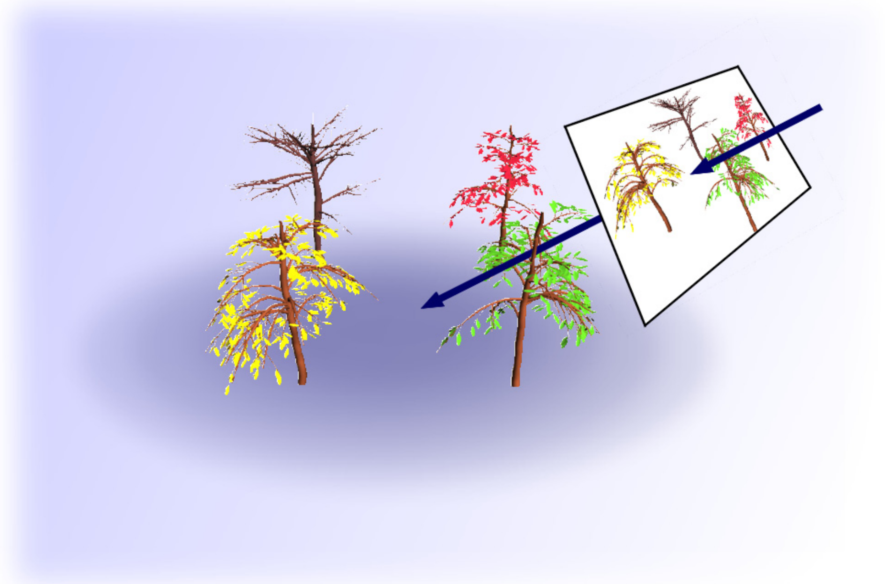
QSlim [Garland and Heckbert 97]

Related Work: Mesh Simplification

- [Schröder et al. 92], [Hoppe et al. 93], [Hoppe 96]
- Works well for smooth meshes
- Problems with irregular topology

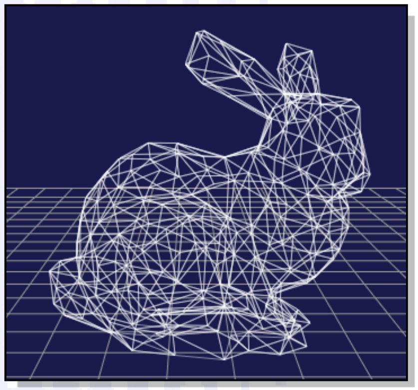
Image-Based Rendering

Related Work –
Image Based Rendering:



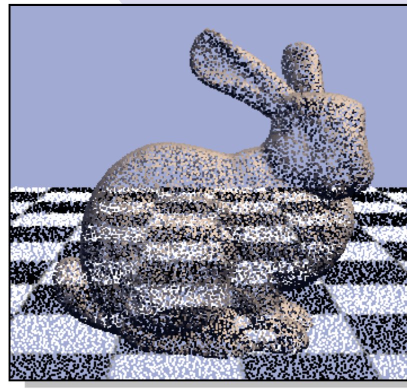
- [Regan and Pose 94],
  [Maciel and Shirley 95],
  [Gortler et al. 96],
  [Levoy and Hanrahan 96],
  [Shade 96], [Schauffler 98]
- Parallax problems vs. memory demands

Point-Based Approach...



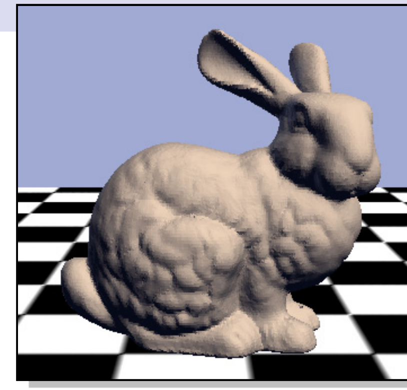
triangles

sample point selection



sample points

image recon-struction



bitmap

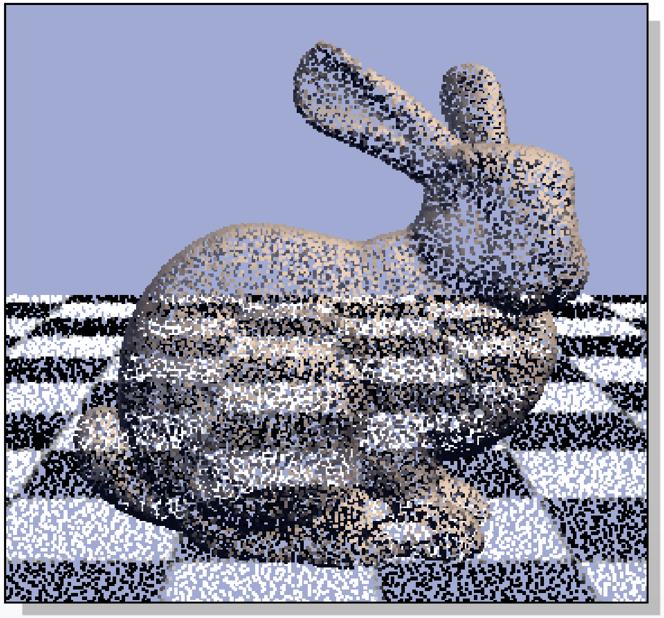
Point-Based Multi-Resolution Rendering:

- Select surface sample points
- Distributed uniformly in image
- Reconstruct image out of sample points

Point-Based Approach...

Advantages:

- Sample size independent of scene complexity
- No topological constraints
- Avoids parallax errors



Related Work

Some Earlier Work in Point-Based Rendering:

- *Particle Systems* [Reeves 83], [Reeves and Blau 85]
- *Surface Rendering* [Levoy and Whitted 85]
- *Volume Rendering* [Westover 90]
- *Voxel Space* [Novalogic 92]
- *Hierarchical Simplification* [Chamberlain et. al 96]
- *Image-Based Rendering* [Shade 98], [Lischinski and Rappoport 98]
- *Simplification* [Max 96, Grossman and Dally 98]

Recent Work

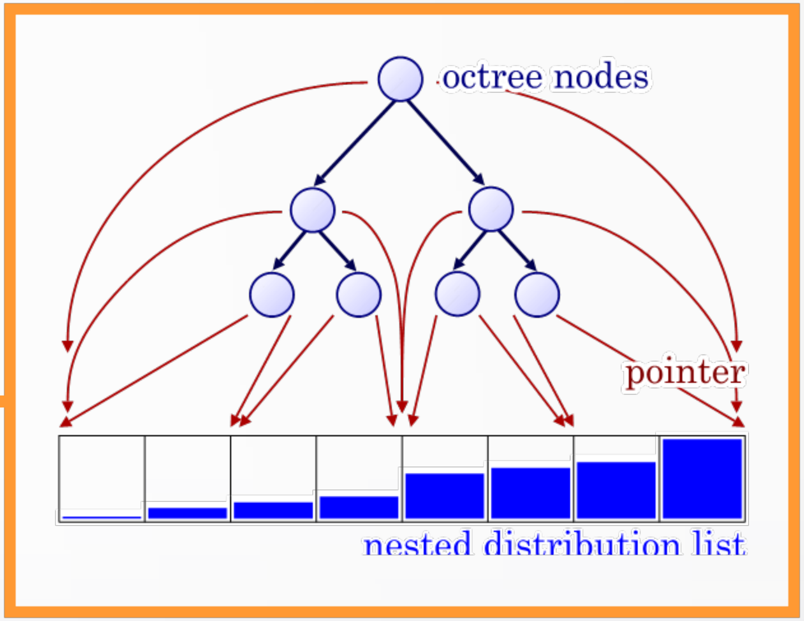
Some Recent Related Work

- *Surfels* [Pfister et al. 2000]
  QSplat [Rusinkiewicz and Levoy 2000]
- *Surface Splatting* [Zwicker 2001]
- *Modeling* [Pauly and Gross 2001, Zwicker et al. 2002, Pauly et al. 2003]
- *Raytracing* [Schaufler and Jensen 2000, Adamson and Alexa 2001]

Overview

Overview:

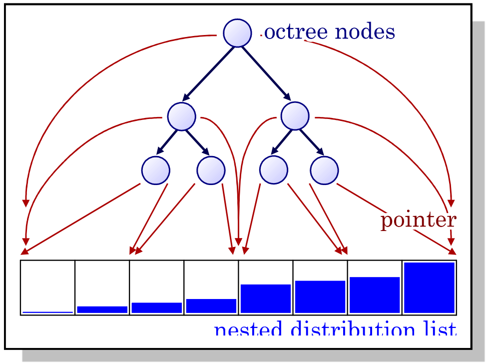
- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions



Data Structures

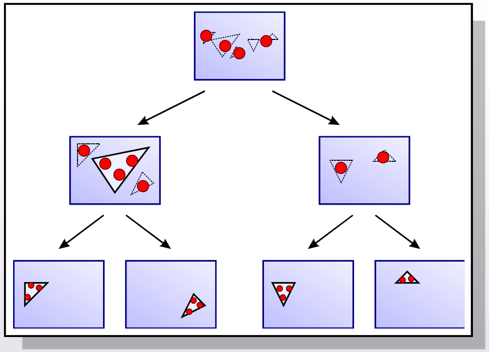
A Dynamic Sampling

- "Randomized z-Buffer"
- Developed in parallel to Surfels / QSplat (mid 1999 - 2000)
- Dynamic random sampling



B Static Sampling

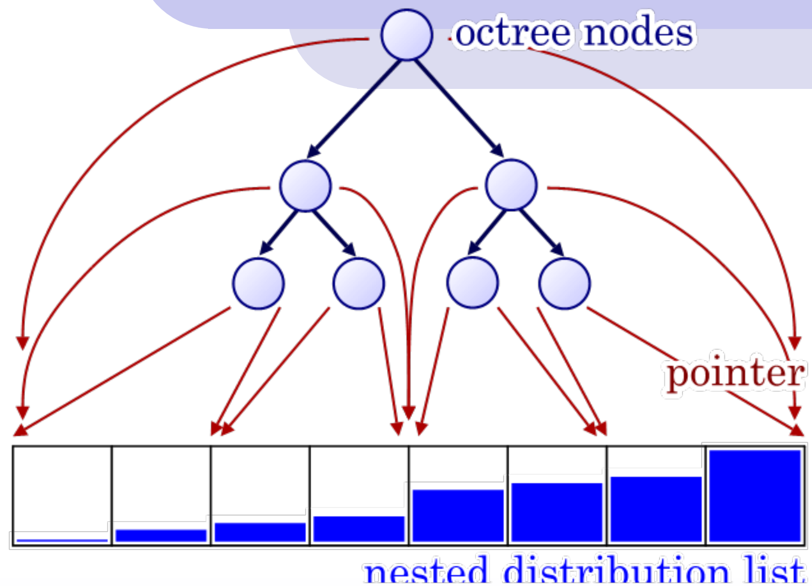
- Precomputed sample sets
- Based on Surfels & Rnd.-z-Buffer



A – Dynamic Sampling

Dynamic
Sampling

Data Structure



Dynamic Sampling Data Structure

- 1. Spatial octree: Fix sampling density for regions
- 2. Distribution lists: Random sampling in O(log *n*)
- Piecewise constant sampling density

First…

Sample
Set Selection

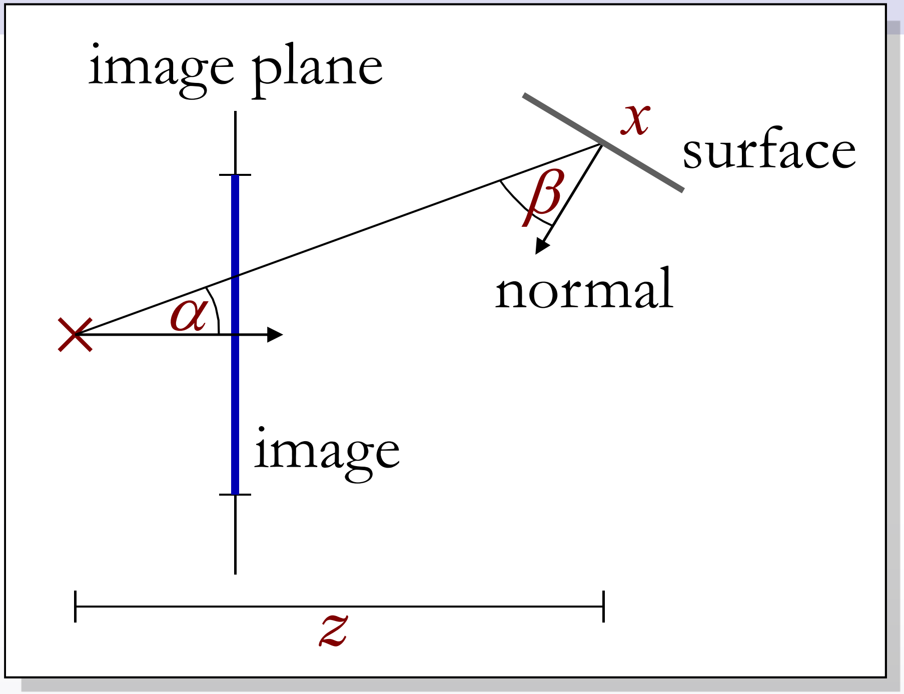
Perspective Mapping

depth factor

orientation factor

$$prj(x) = \frac{1}{z^2} \cdot \cos\beta \cdot \frac{1}{\cos\alpha}$$

distortion factor



Projection factor *prj*(*x*)

- area scale factor of perspective projection
- sampling density ~ *prj*, conservative approximation

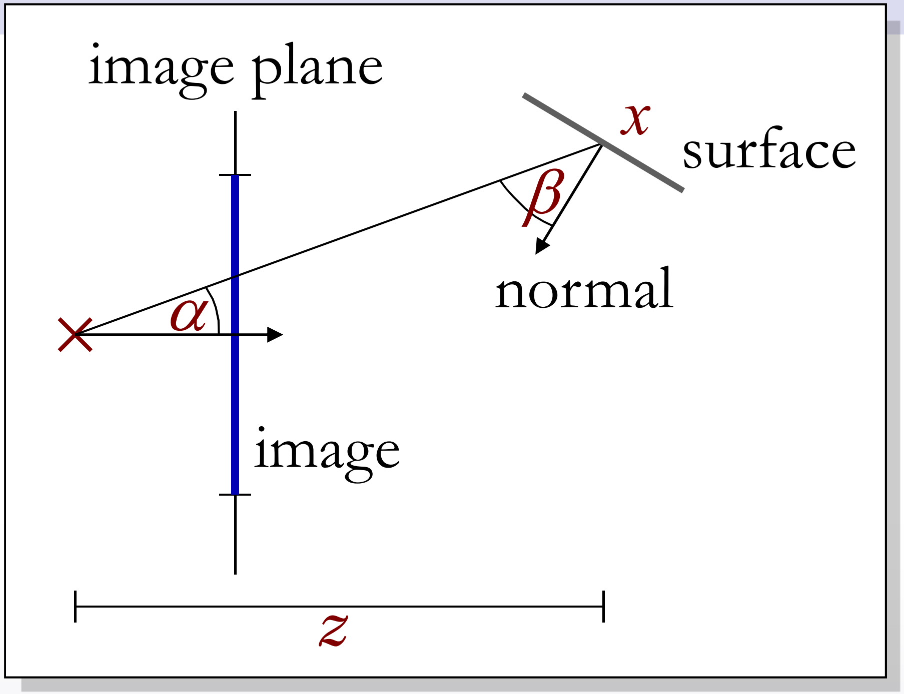
Perspective Mapping

depth factor

orientation factor

$$prj(x) = \frac{1}{z^2} \cdot \cos\beta \cdot \frac{1}{\cos\alpha}$$

distortion factor



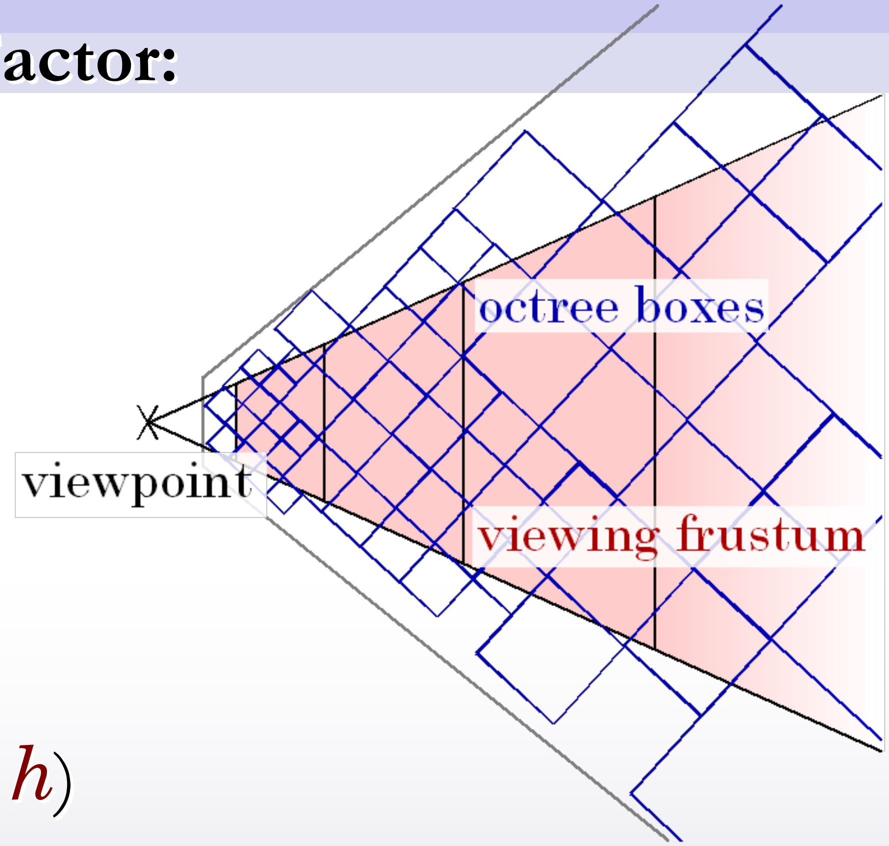
Projection factor *prj*(*x*)

- area scale factor of perspective projection
- sampling density ~ *prj*, conservative approximation

Depth Approximation

Approximating the Depth Factor:

- ε-approximation
- Octree traversal
- Subdivide if still $z_{max}{}^2/z_{min}{}^2 > 1 + \varepsilon$



Efficiency:

- Running time $O(\log \tau + h)$
- Relative depth range τ, octree height h

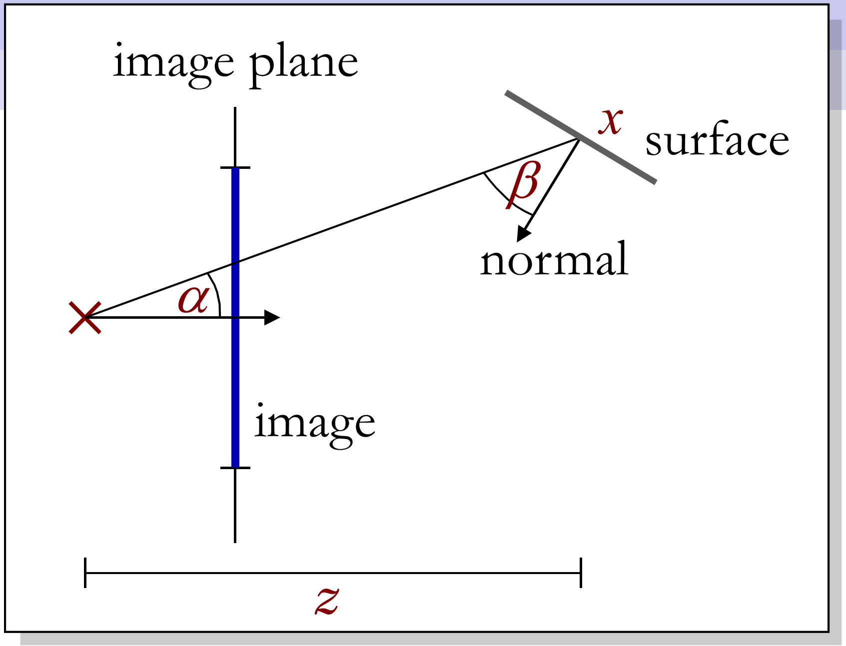
Orientation, Distortion

depth factor

orientation factor

$$prj(x) = \frac{1}{z^2} \cdot \cos\beta \cdot \frac{1}{\cos\alpha}$$

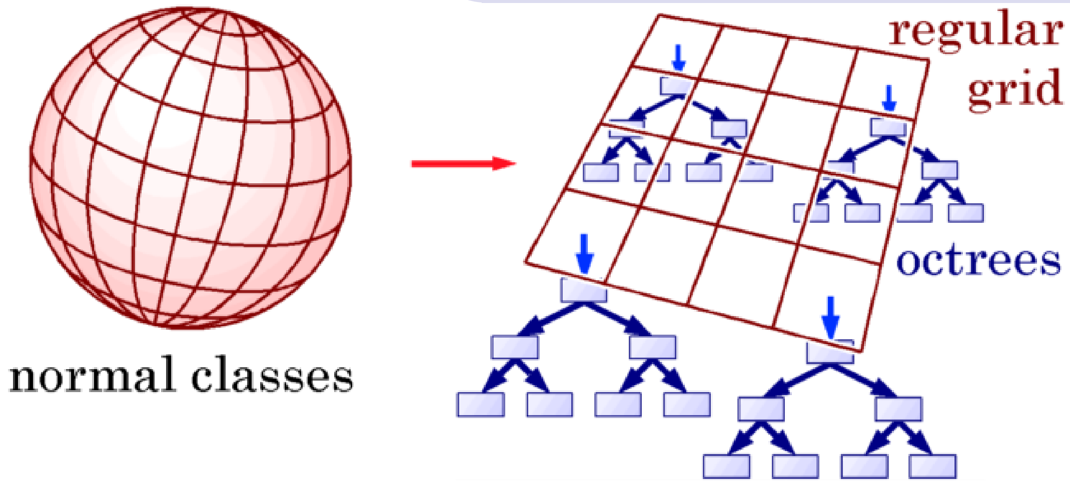
distortion factor



Other Factors:

- *Distortion factor:* No problem in practice
- *Orientation factor:* Two options...

Orientation Factor



Orientation Classes

- Adapt sampling density to orientation
- Minor speedup in practice
- No strict ε-approximation possible
- Ignore orientation $\Rightarrow$ *Average oversampling factor* $2\times$

Next…

Image Reconstruction

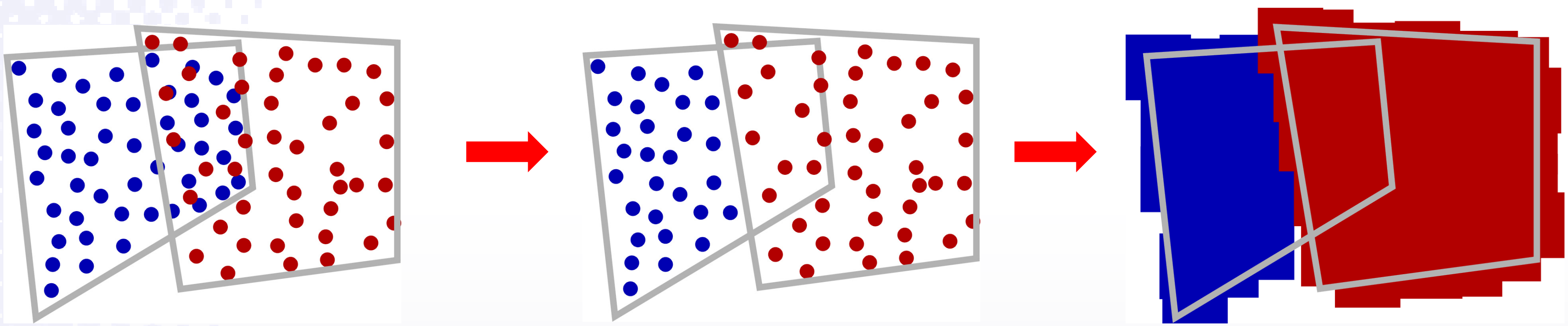
Reconstr. of Occlusion

Two problems:

Sample points

1. Reconstruction of occlusion

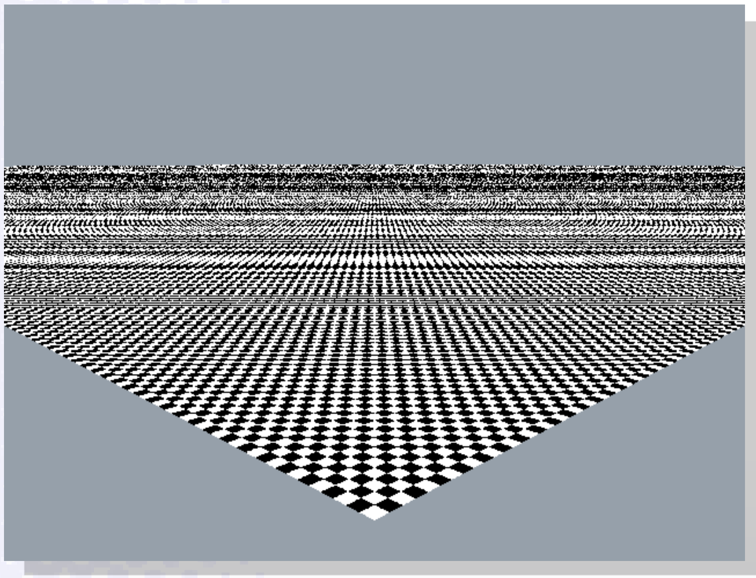
2. Filling



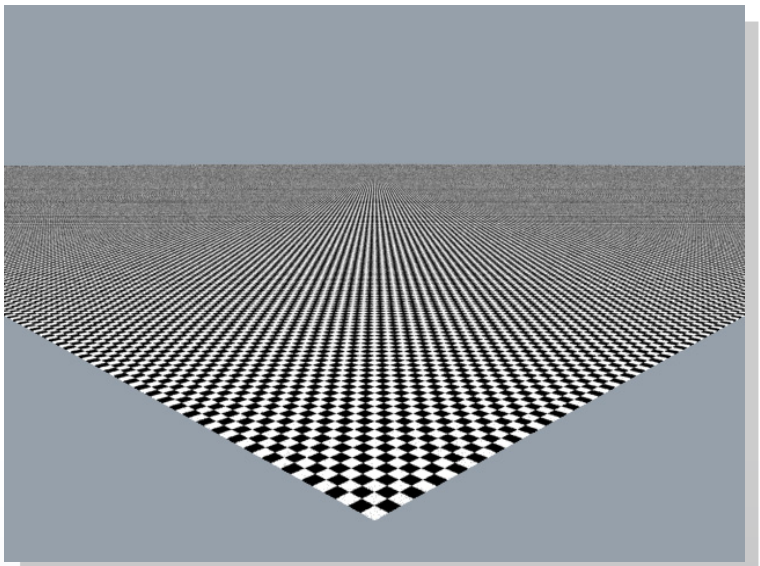
remove adjacent points with larger depth

scattered data interpolation

Interpolation Options



Per-pixel reconstruction



Gaussian reconstruction

Next...

Analysis

Correct Images

How many random sample points are necessary?

- Criterion: Save surface coverage
- $\boxed{k \leq a(\ln a + \ln f^{-1})}$ points necessary
- a – estimated projected area *in pixel* (incl. *overestimation* and *occlusion*)
- Average scenes: a linear in true projected area
- f = failure probability

Preprocessing

Preprocessing Costs:

- Preprocessing: O(n) memory, O($n \log n$) time.
- Dynamic updates (insert, delete triangles): O(h)
- In practice: 8 seconds for 90K triangles & instances

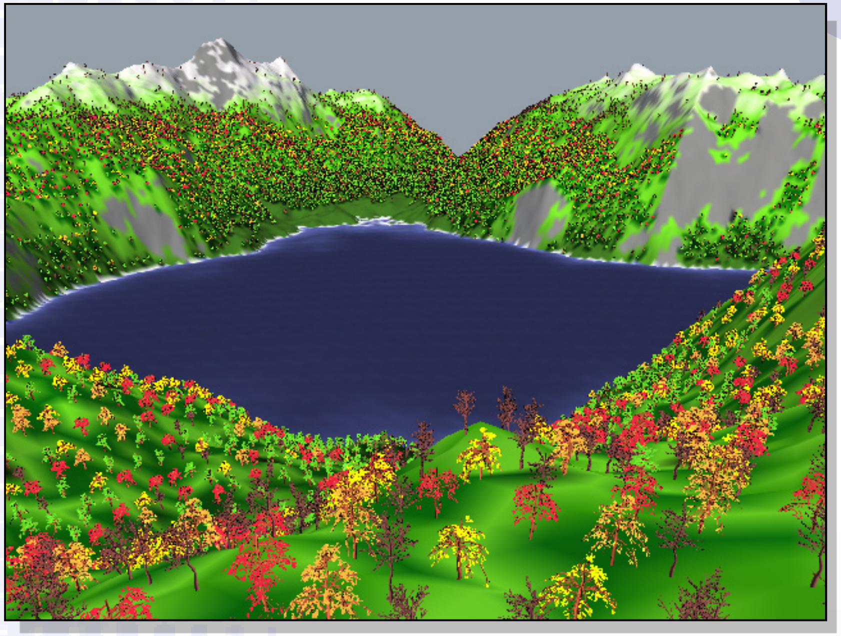
Efficiency

Rendering

- Running time $O(\log \tau + h + a \underbrace{\boxed{\log a}}_{\text{random sample sets}} \overbrace{\boxed{\log n}}^{\text{sampling}})$
- τ – relative depth range
- h – octree height
- a – estimated projected area (incl. *overestimation* and *occlusion*)

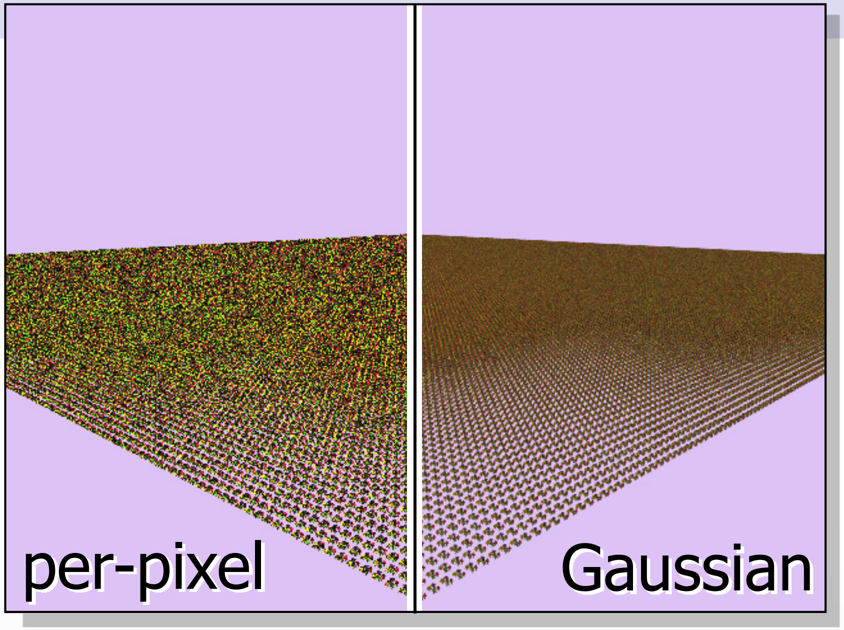
Applicable to highly complex scenes

Examples...



0.8 - 3.9 sec.

($4 \cdot 10^8$ triangles, 640×480 pixel)



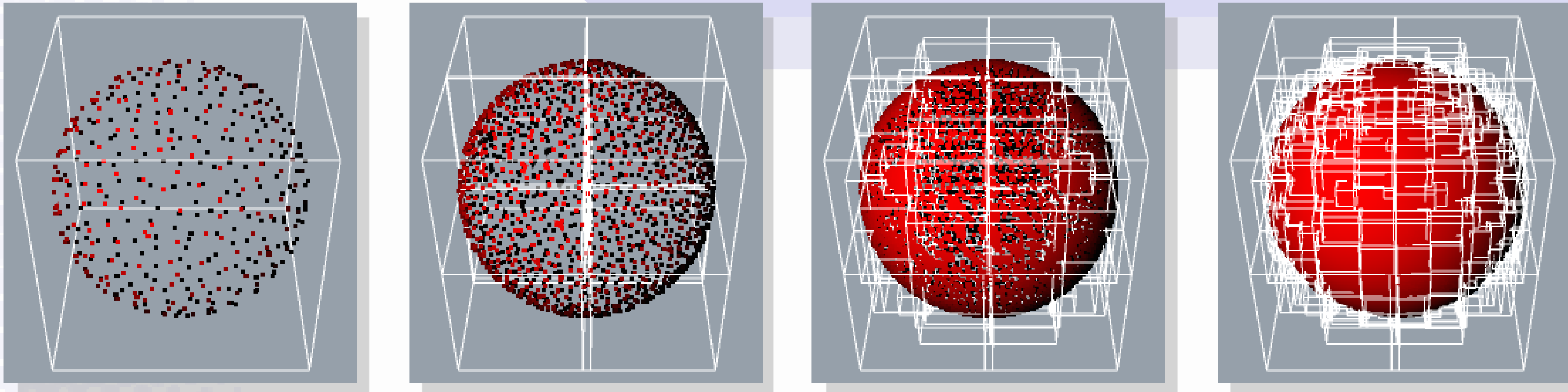
0.4 - 81 sec.

($6 \cdot 10^9$ triangles, 640×480 pixel)

Next...

Static Sampling

B – Static Sampling



Static Sampling: Precomputed Sample Sets (cf. Surfels)

- Octree Hierarchy
- Sample spacing: Fraction of box side length
- Store large triangles (say > 3 points) “as-is”

Sampling

Which sampling strategy is best?

- Candidates:
  - Random sampling
  - Jittered grid
  - Quantized grid
  - Neighborhood based sampling (≈ Poisson disc)
- Criterion: Oversampling
  - Analytical upper bounds
  - Average case (empirical)

Sampling

Which sampling strategy is best?

- Candidates:
  - Random sampling
  - Jittered grid
  - Quantized grid
  - Neighborhood based sampling (≈ Poisson disc)

flexible & efficient

- Criterion: Oversampling
  - Analytical upper bounds
  - Average case (empirical)

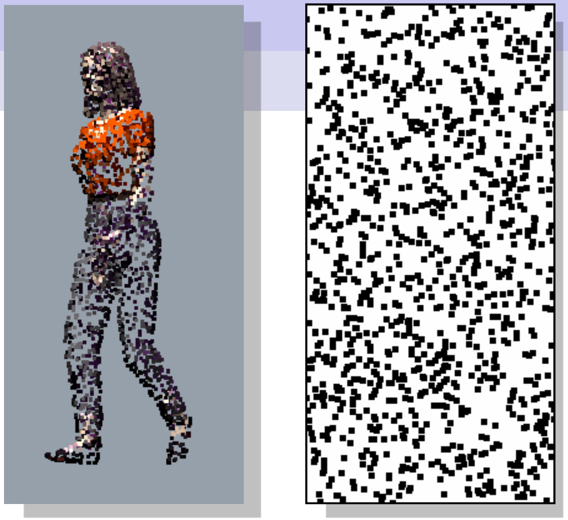
Sampling

Neighborhood-Based Point Removal:

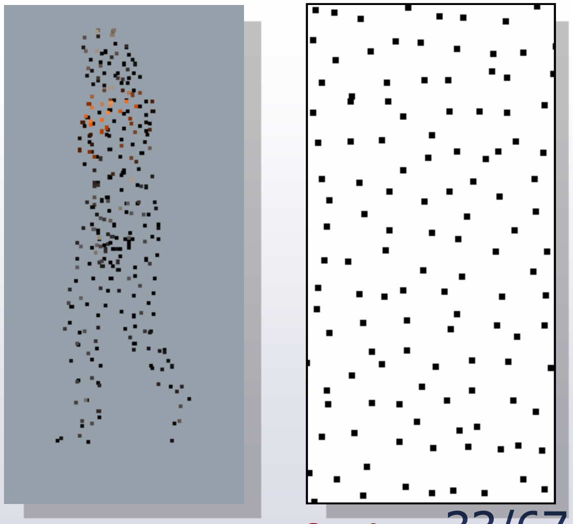
- Two step approach
- *Candidate set*, *stratification*

First Step: Candidate Set

- Random candidate set
- Uniformly distributed on triangle area
- O($\alpha \log \alpha + \log f^{-1}$) points
  (α = area / *sampling distance*^{2},
  f = failure probability)



candidate set

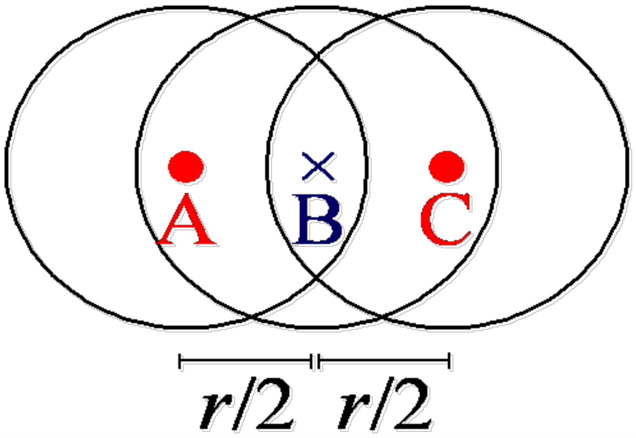


stratified

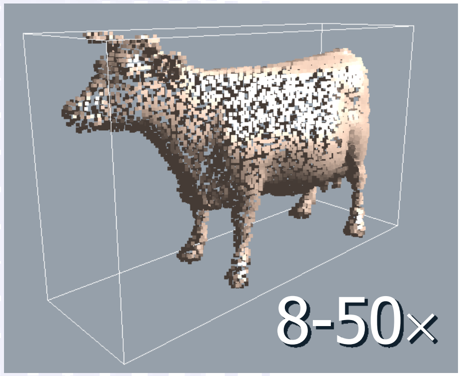
Stratification

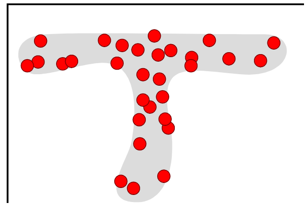
Second Step: **Stratification**

- Delete points that are still covered by other points
- Greedy strategy
- Typically $O(n \log^2 n)$ processing time, n points output

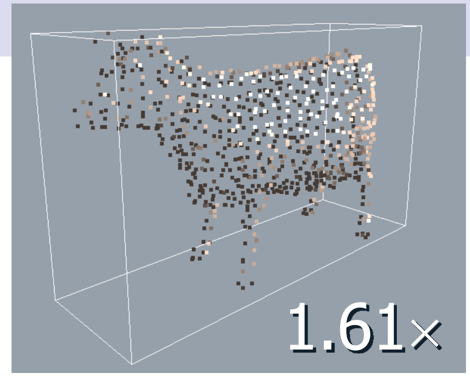


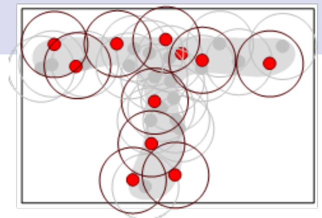
Oversampling





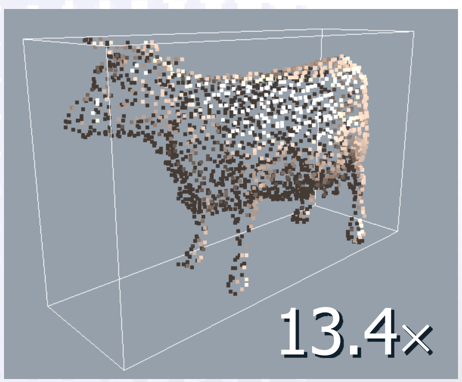
random

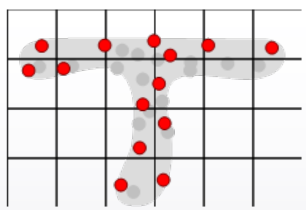




neighbors

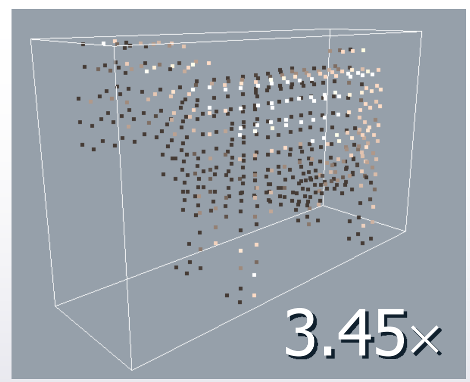
(upper bound: 3.6×)

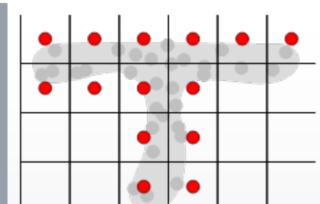




grid

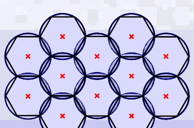
(upper bound: 28.3×)





quantized

(upper bound: 7.1×)



Optimum: 1.21×

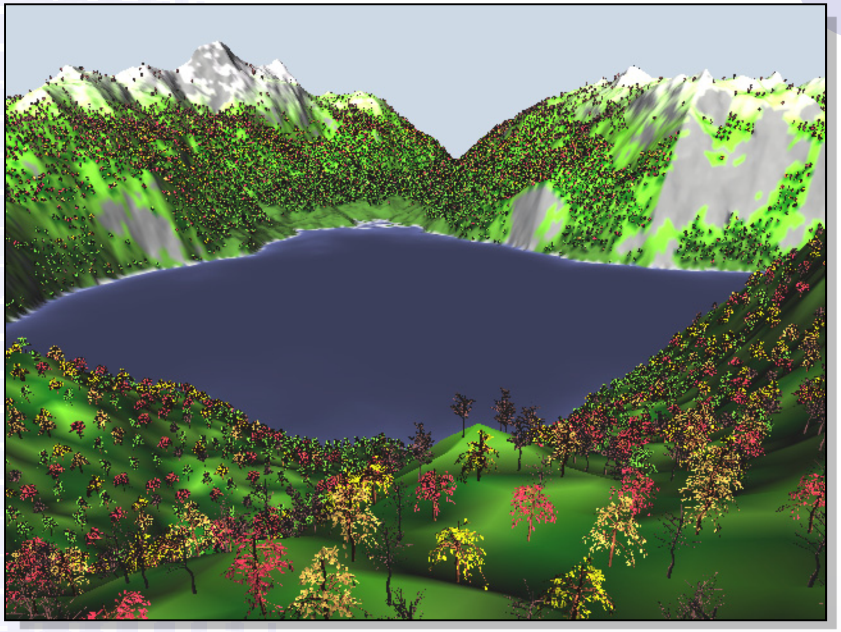
Efficiency

Static Sampling

- Overall rendering time O(log τ + h + a ~~log a log n~~)
- τ – Relative depth range
- h – octree height
- a – estimated projected area (incl. *overestimation* and *occlusion*)
- Preprocessing: O(n) memory, O(hn) time.

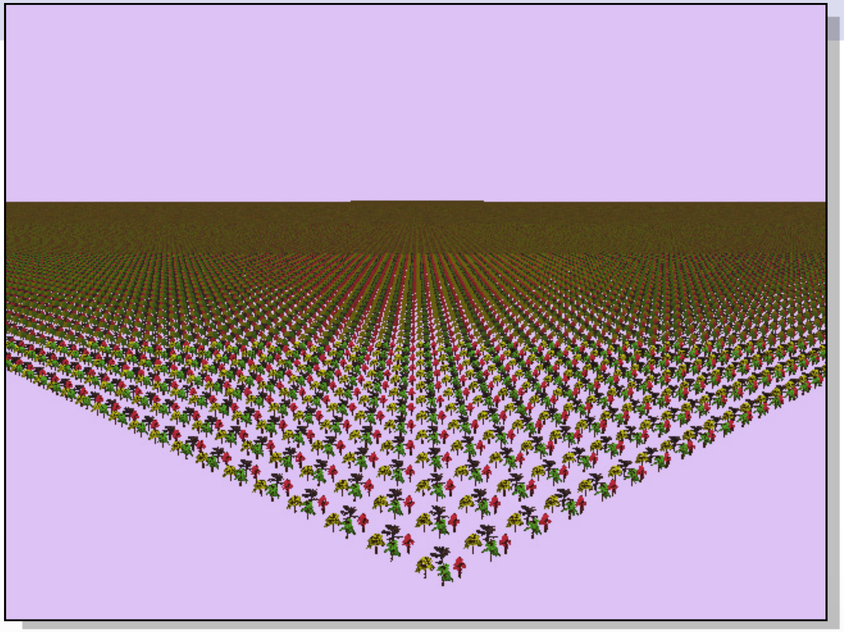
Allows real-time rendering

Rendering Performance



4 frames / sec.

($4 \cdot 10^8$ triangles,
640×480 pixel)



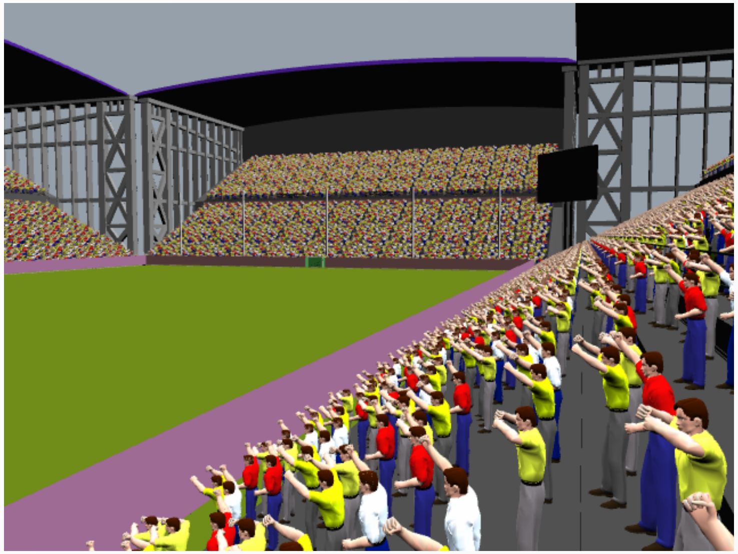
5-10 frames / sec.

(10^{15} triangles,
640×480 pixel)

Overview

Overview:

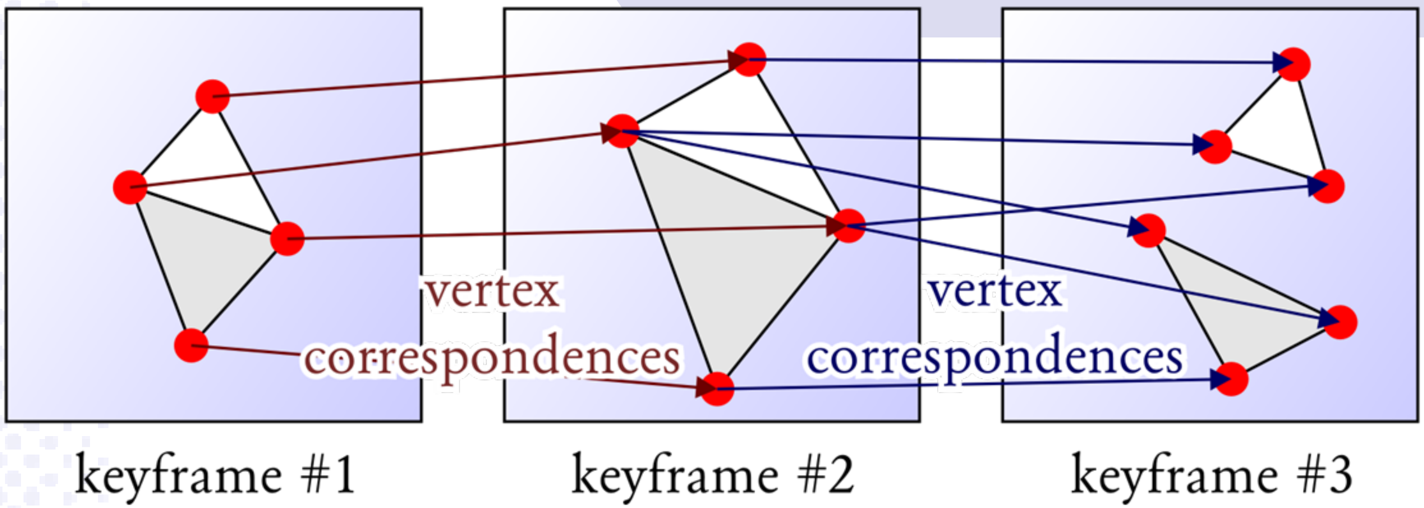
- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions



First…

Modeling Animations

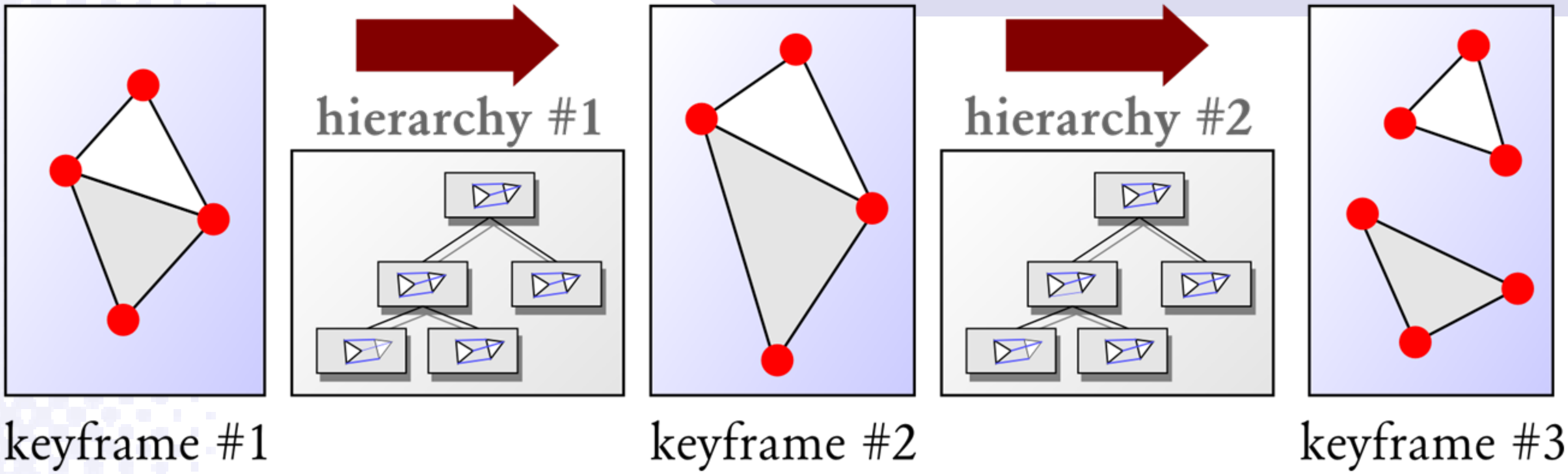
Keyframe Animations



Modeling of Animations:

- Keyframe animations
- Linear interpolation
- Arbitrary topology

Animation



Generalization to Keyframe Animations:

- Separate hierarchy between consecutive keyframes
- Hierarchy interpolation

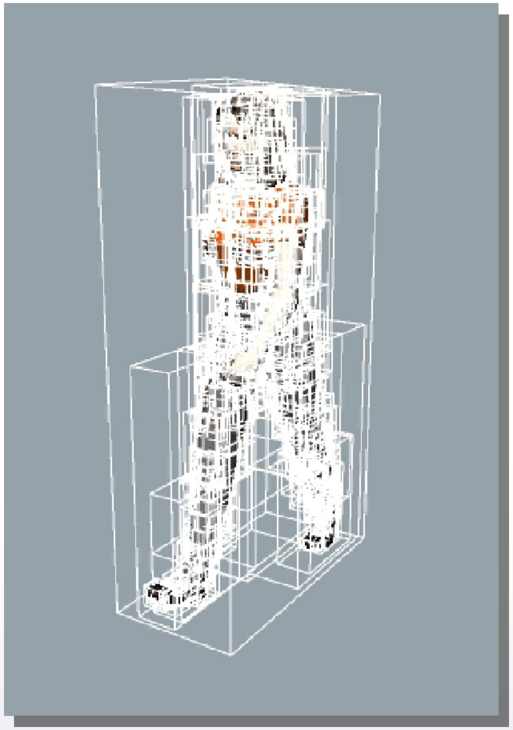
Next…

Interpolated Point Hierarchies

Hierarchy Interpolation

Interpolated Hierarchies:

- *Bounding boxes* can be interpolated linearly (upper bound)
- *Triangle vertices* are interpolated linearly (incl. attributes)
- *Point samples* are interpolated linearly (incl. attributes)



Animated Sampling

Problem: How to define sample sets?

- Grid-based sampling not applicable

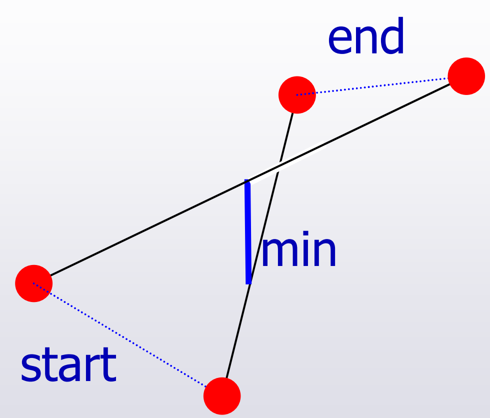
⇒ Neighborhood-based sampling

First Step: Random Sampling

- Use maximum area per triangle, *coverage still guaranteed*

Second Step: Stratification

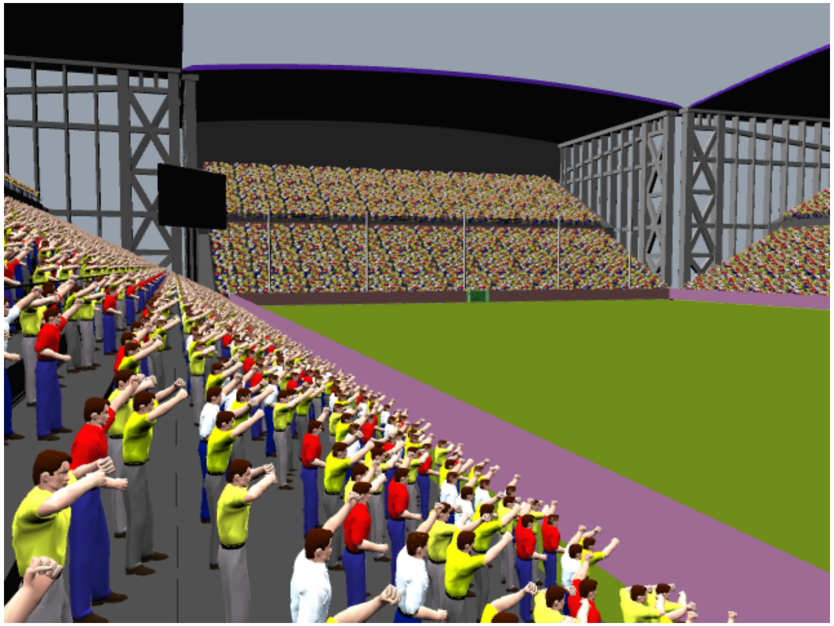
- Points cover each other if:
  - sufficiently small distance at start...
  - ...and end time.

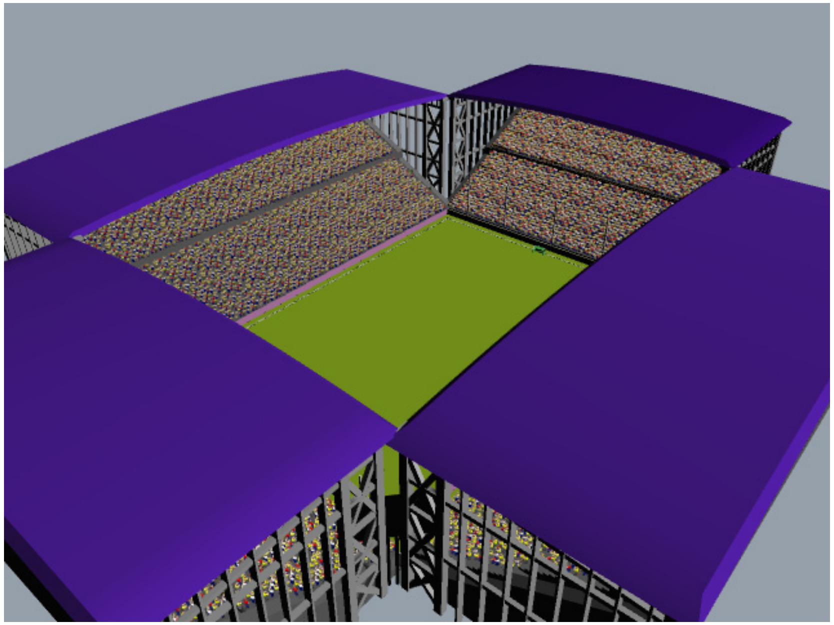


Next...

Application Examples

Instantiation



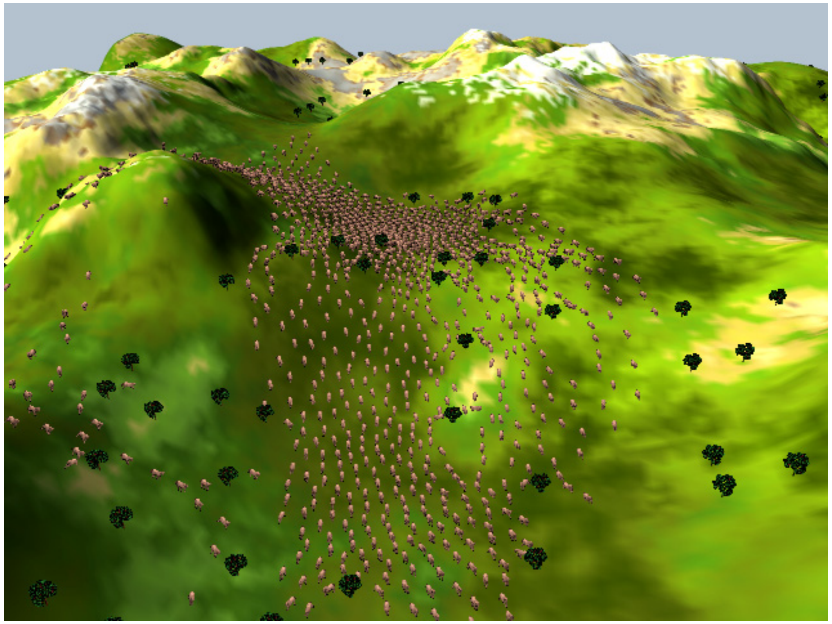


16.416 football fans à 6.400 triangles, 640×480 pixel

$\Rightarrow$ 105 million triangles

Rendering speed: 10-20 frames / sec

Simulation



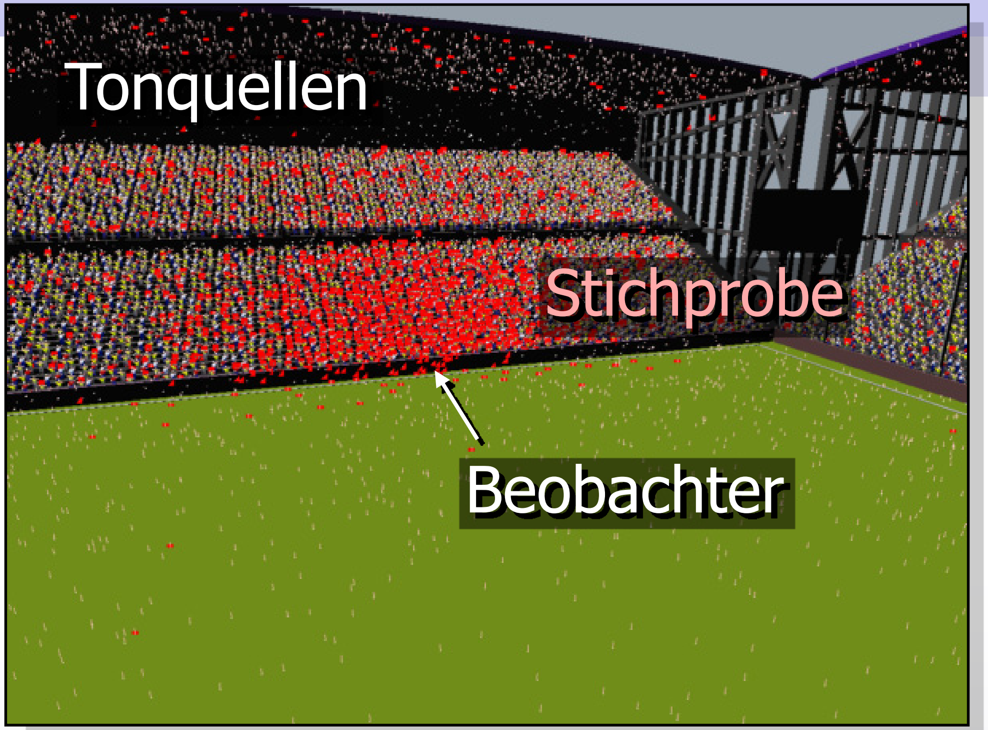
1.300 horses, 200 trees

⇒ 42 million triangles

Rendering speed: up to 8-10 frames / sec

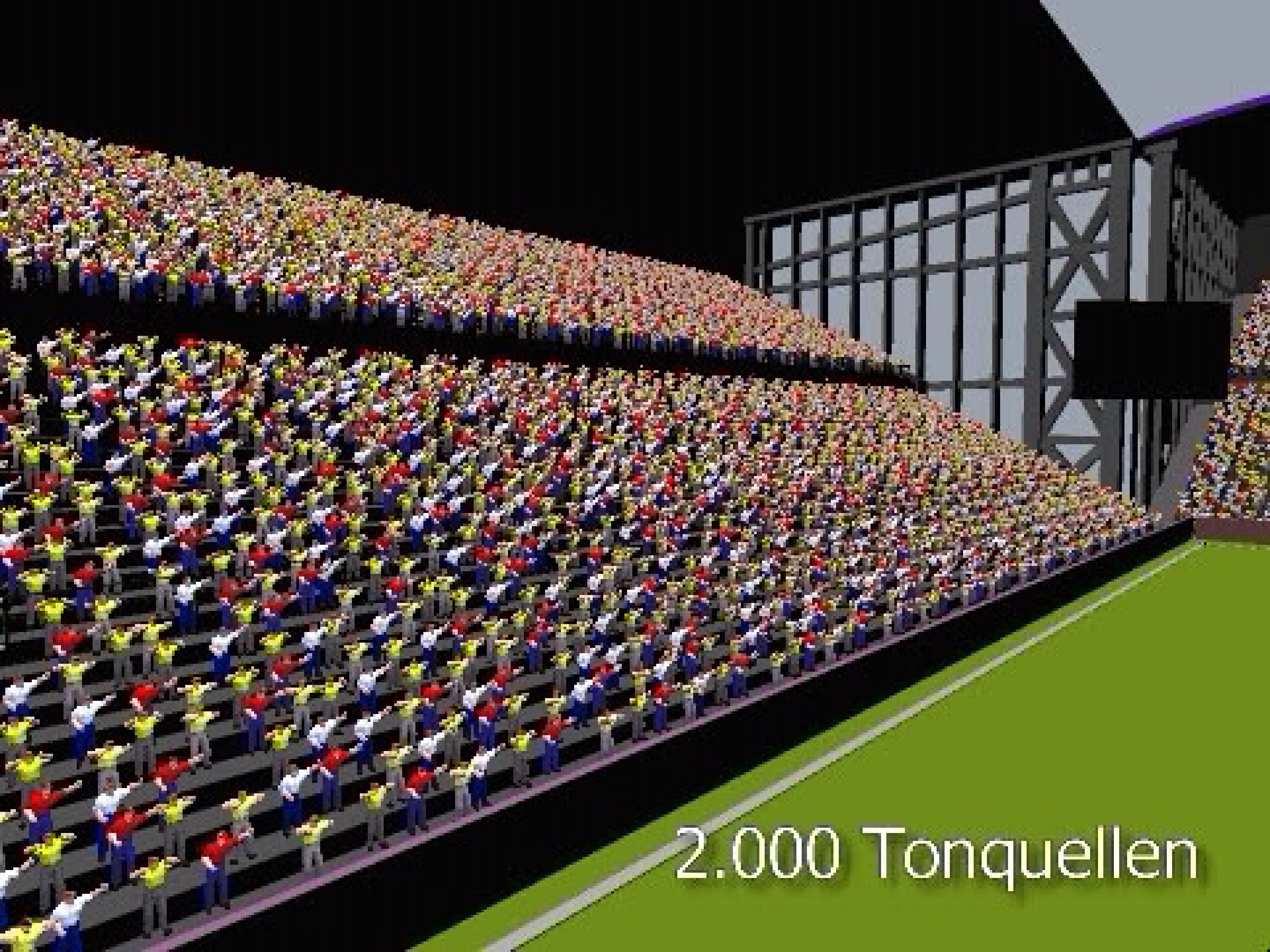
Sound Rendering

Beobachter



Extension: Point-Sampling Based Audio Synthesis

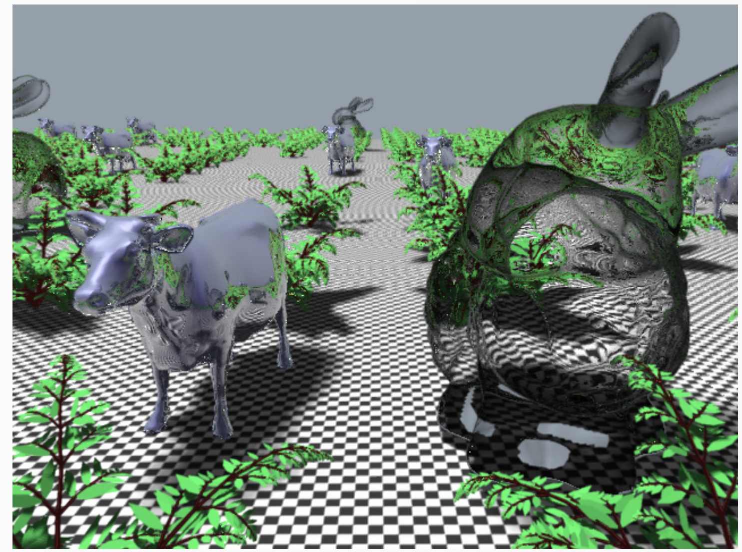
- Same principle – observer dependent sampling


2.000 Tonquellen

Overview

Overview:

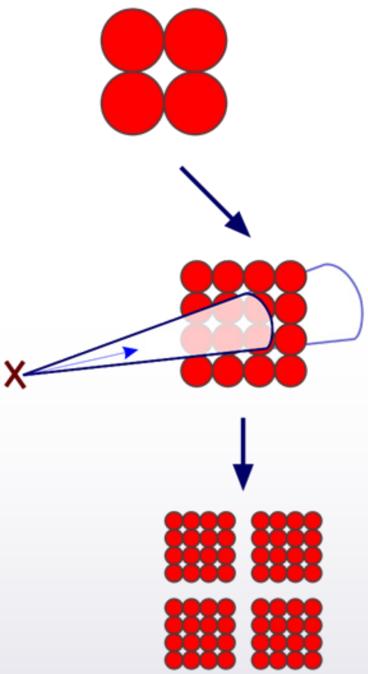
- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions



Multi-Resolution Raytracing

Point-Based Multi-Resolution Raytracing:

- Antialiasing, soft shadows, blurry reflections etc…
- *1 primary ray per Pixel*



Algorithm: [cf. Amanatides 84]

- Shoot extended ray cones
- Prefiltered (static) point hierarchy
- Select points that match ray footprint
- Local surface reconstruction

Extension of *Surface Splatting* [Zwicker et al. 2001]

First...

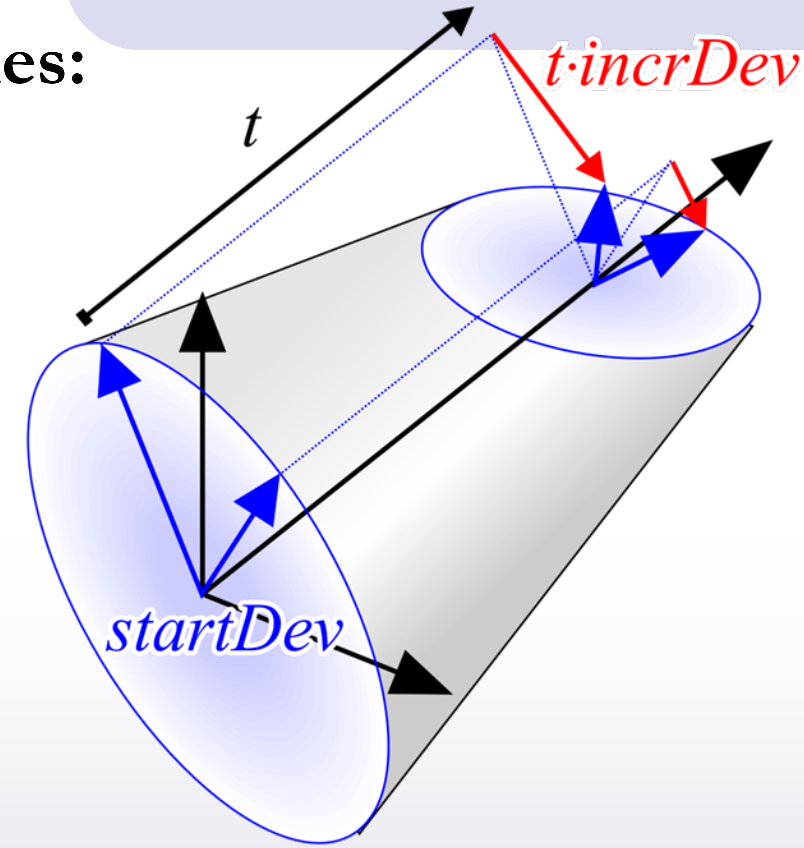
Ray Model

Extended Ray Volumes

Anisotropic Ray Volumes:

1st order approximation

(cf. [Igehy 99])



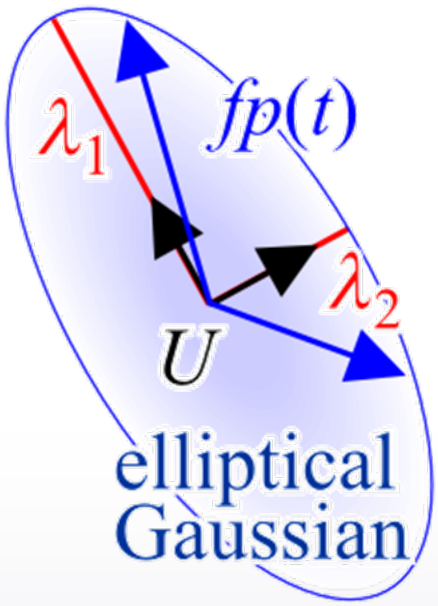
$$fp(t) = startDev + t \cdot incrDev$$

Extended Ray Volumes (2)

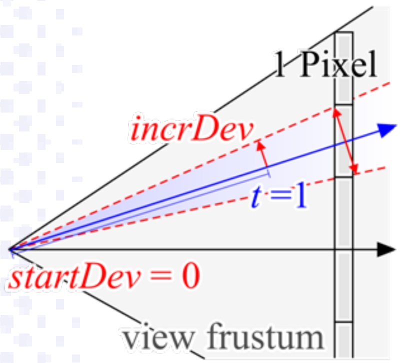
Gaussian Filter:

- $$weight(u,v) = e^{-(u,v)\cdot fp(t)^{-2}\cdot\begin{pmatrix}u\\v\end{pmatrix}}$$
- Eigenspace: $fp(t)^2 = U\begin{pmatrix}\lambda_1{}^2 & 0\\ 0 & \lambda_2{}^2\end{pmatrix}U^T$

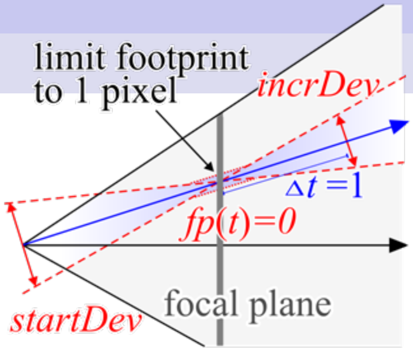
(U orthogonal)



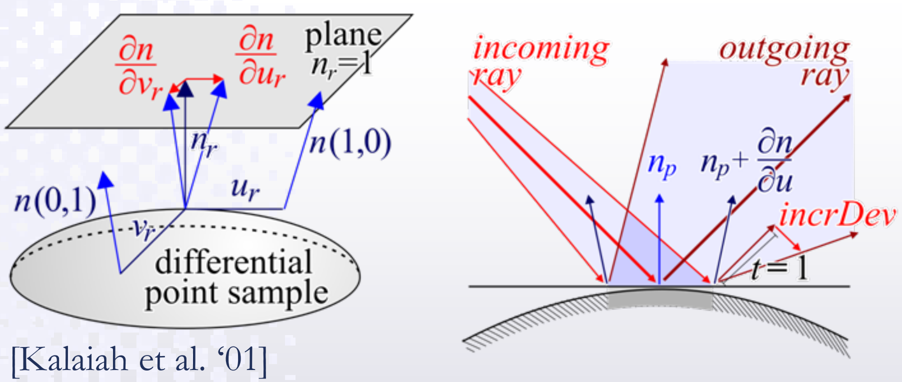
Surface Interaction



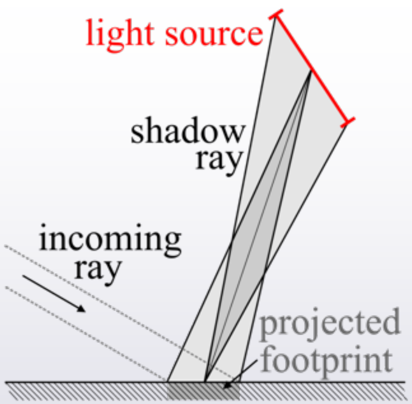
primary rays



depth-of-field



reflection



shadow rays

Next…

Intersection Tests

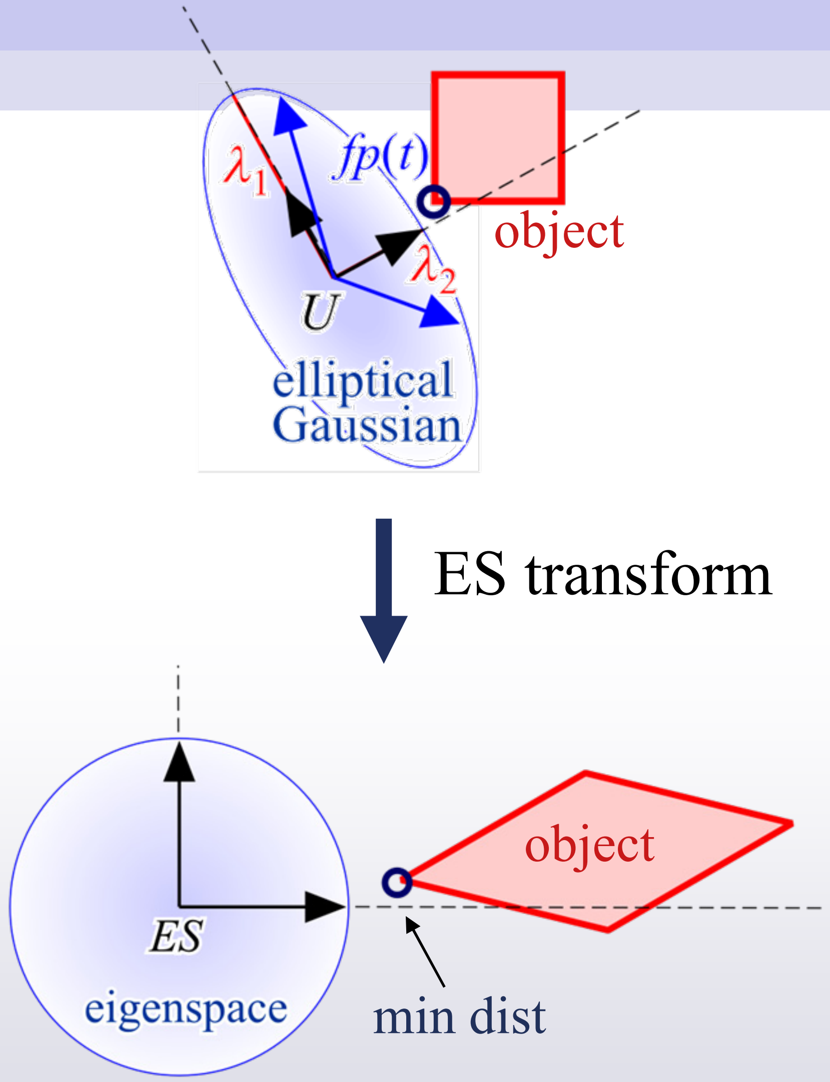
Intersection Tests

Intersection with:

- Bounding volumes, points, triangles

General Technique:

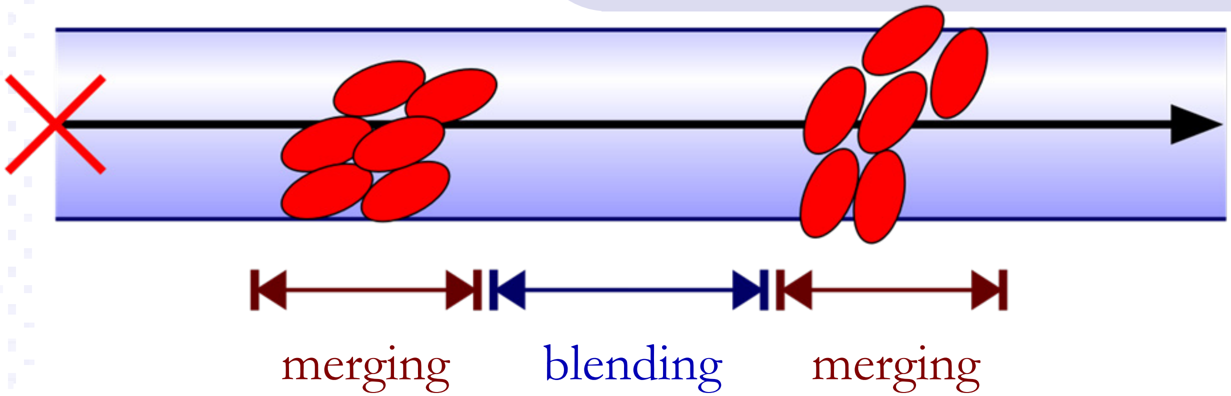
- Project vertices into ray coordinates (Eigensystem)
- Calculate distance
- Evaluate Gaussian filter



Next…

Compositing

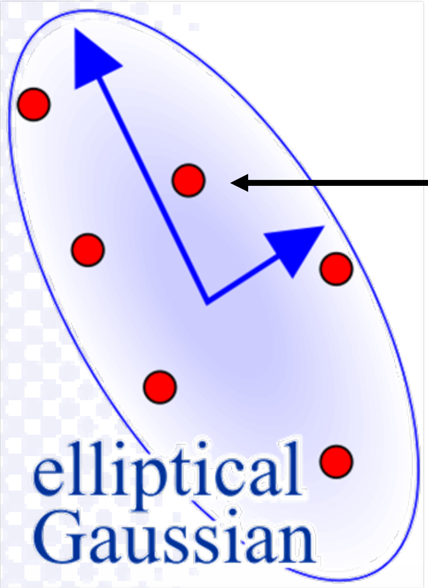
Compositing (2)



Ray Compositing: "*Ray A-Buffer*" (cf. [Zwicker et al. '01])

- Merge points with overlapping depth (same surface)
- Blending for different surfaces:
  - (a) Alpha blending (weight sum)
  - (b) Subpixel masks

Merging



point p_i:
attribute: $a_i(u,v)$
weight: $w_i(u,v)$

$$\text{reconstructed attribute} = \frac{\sum_{i=1}^{n} w_i \cdot a_i}{\sum_{i=1}^{n} w_i}$$

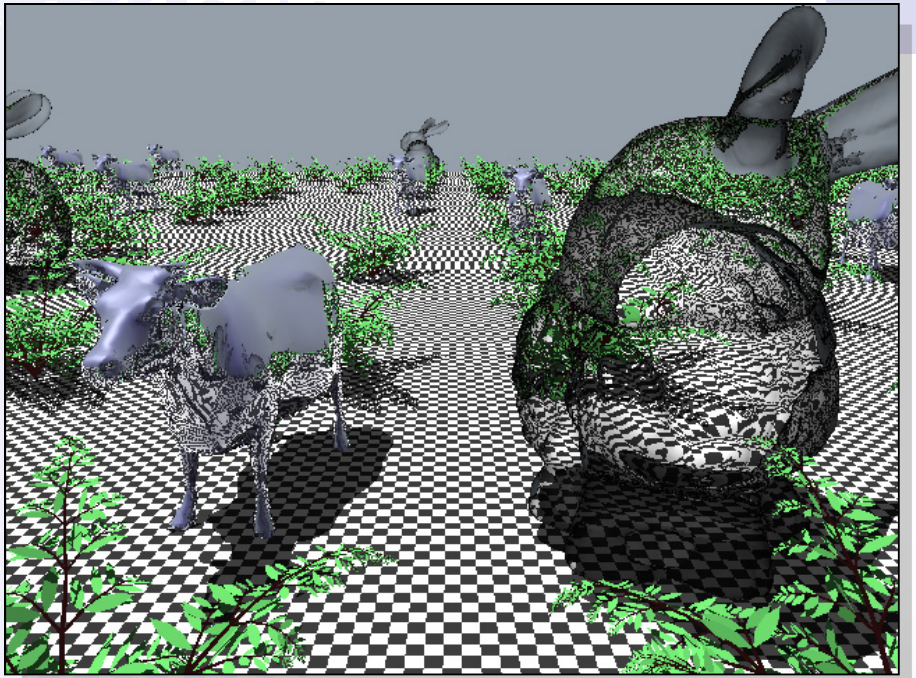
Fragment Merging:

- Weighted sum of point attributes
- Normalize by weight sum

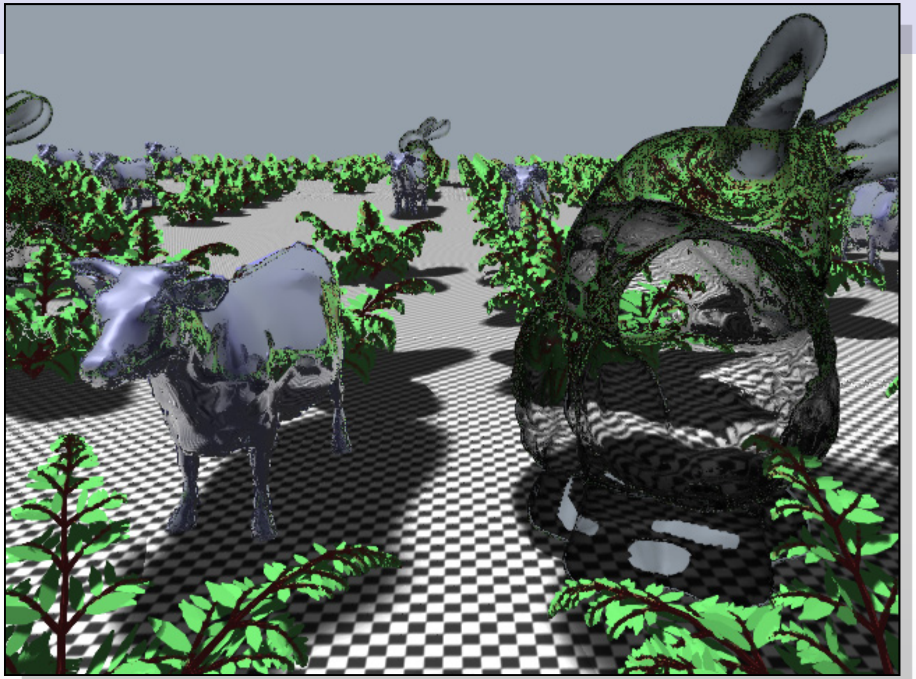
Next...

Results

Conventional Raytracing

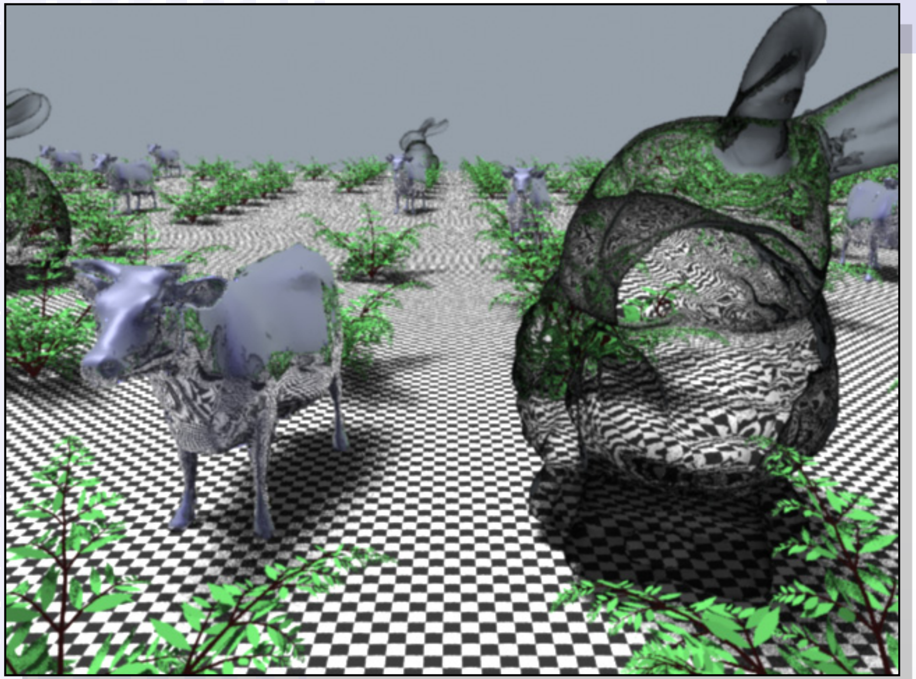


Conventional Raytracing:
215 sec

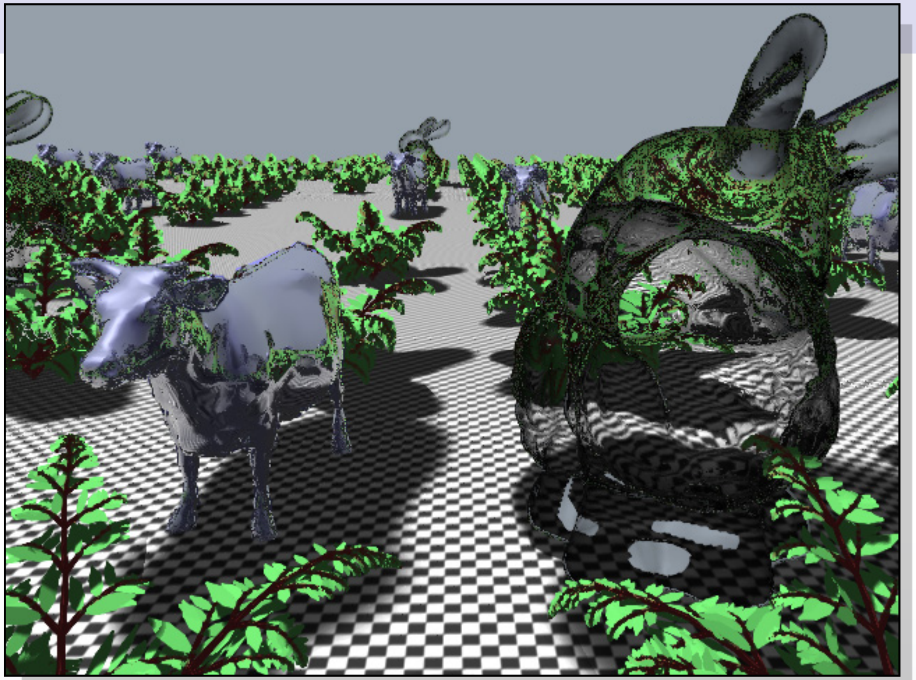


MR Raytracing:
1332 sec

Distributed Raytracing

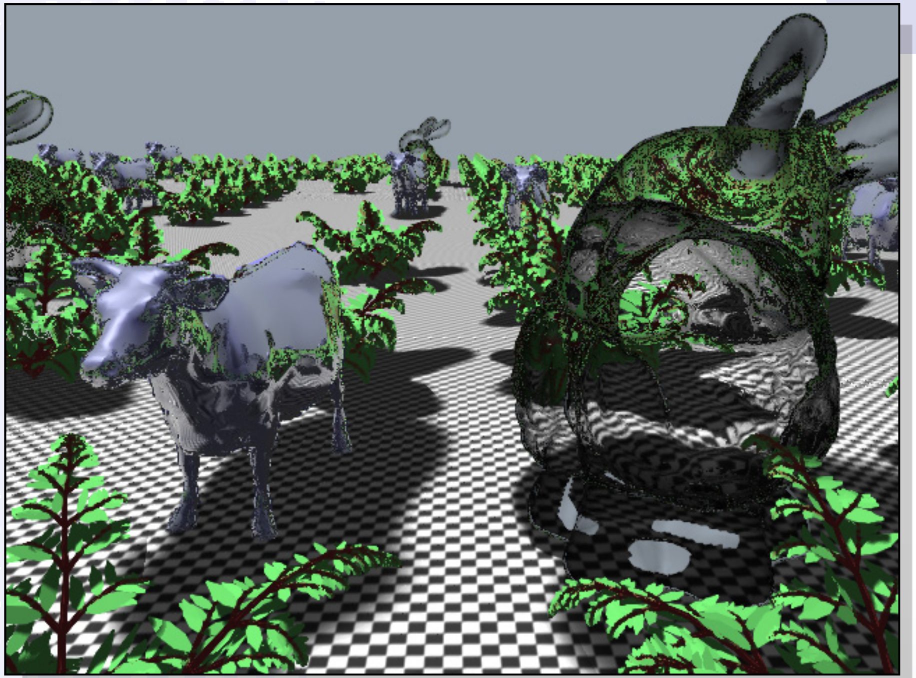


Distributed Raytracing:
1334 sec

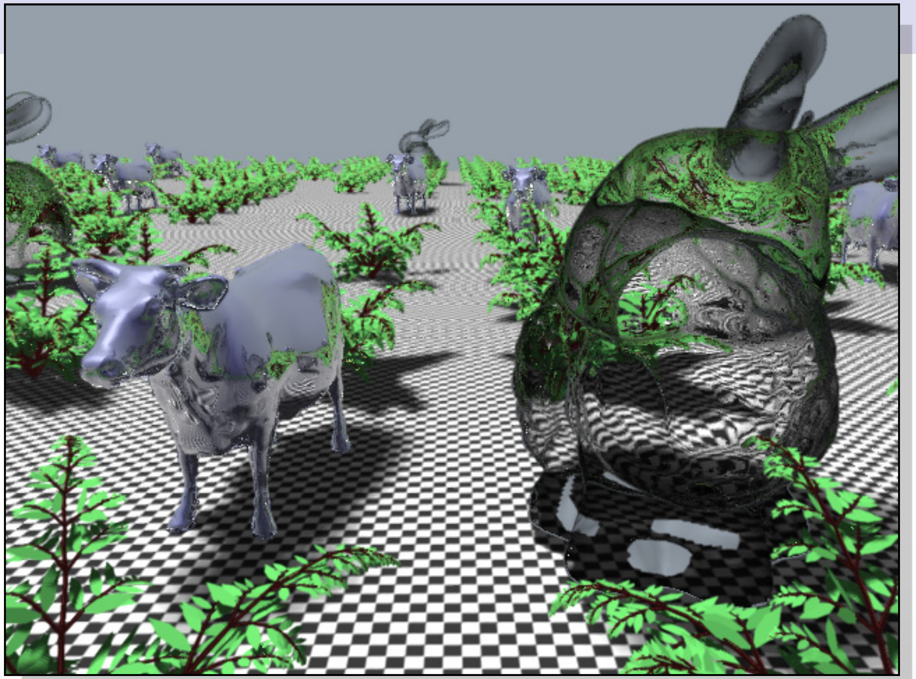


MR Raytracing:
1332 sec

Subpixel Masks

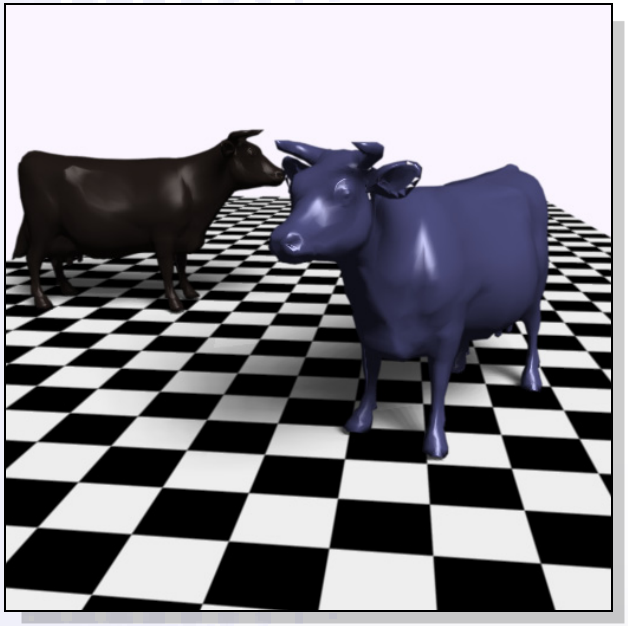


MR Raytracing, α-blending:
1332 sec

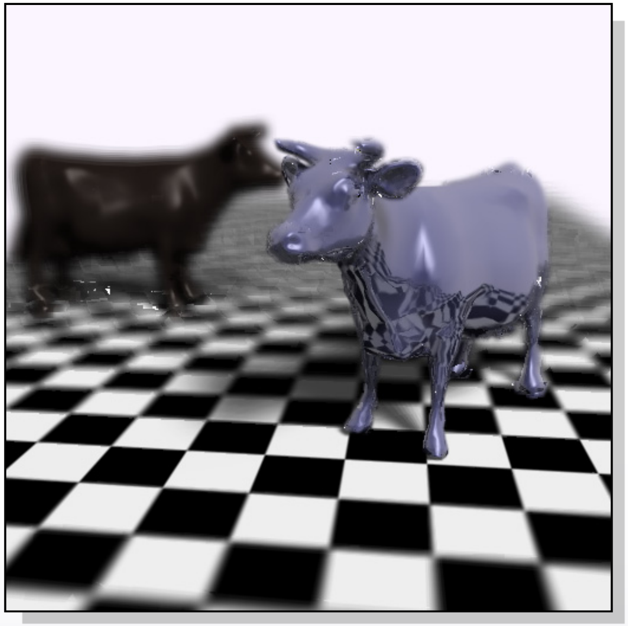


MR Raytracing, subpixel masks:
7244 sec

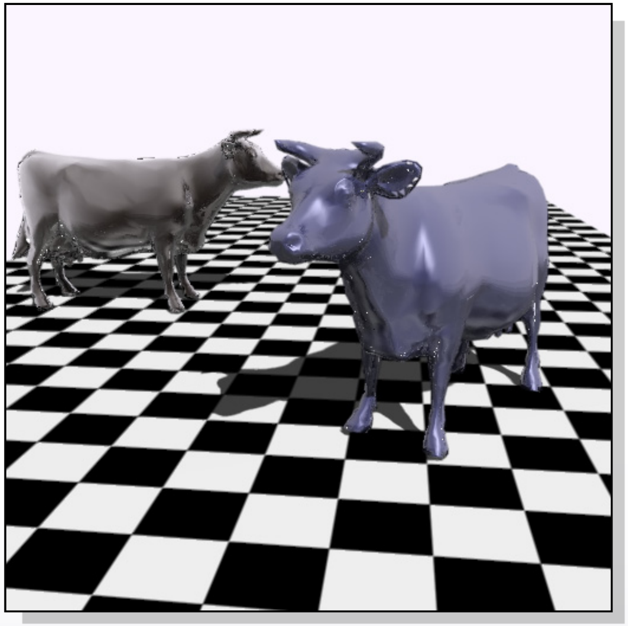
Special Effects



soft shadows



depth-of-field



blurry reflections

Overview

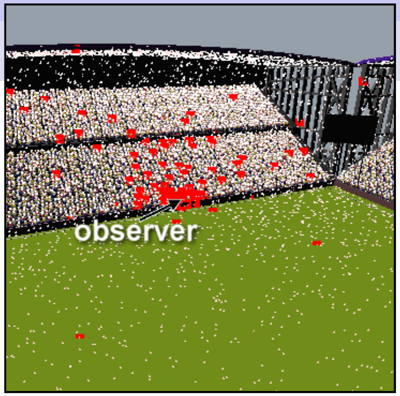
Overview:

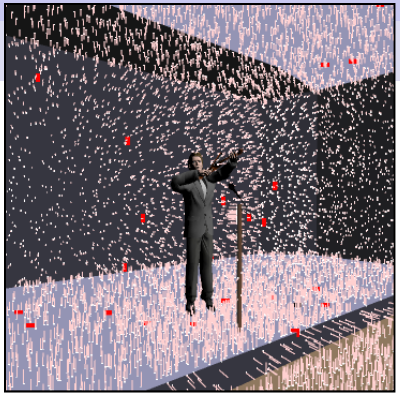
- Introduction
- Forward Mapping
- Animated Scenes
- Raytracing
- Extensions

Extensions...


Multi-Resolution Volume Rendering

(joint work with Stefan Guthe)





Sound Rendering

Volume Rendering: O(log n) rendering time for n^3 voxels
Visible Human (8GB) at 5-10 fps

Sound Rendering: Real-time auralization of scenes with a large numbers of sound sources