Bachelor-Thesis
Ray-Tracing Point Clouds

Christoph Wiesmeier

18. November 2011

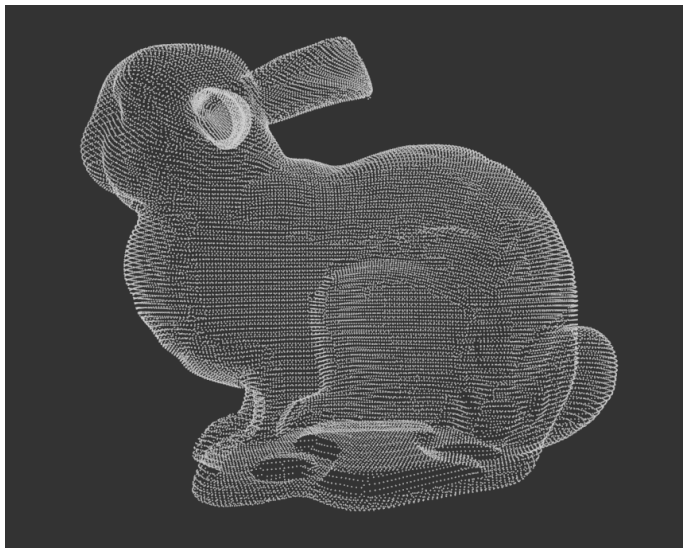
Overview

Challenges when rendering points

- Points have no surface
  - Rasterisation $\rightarrow$ Which shape has an infinitesimal point?
  - Ray-Tracing $\rightarrow$ How to intersect a ray?
- Points have no orientation
  - How to compute the shading?

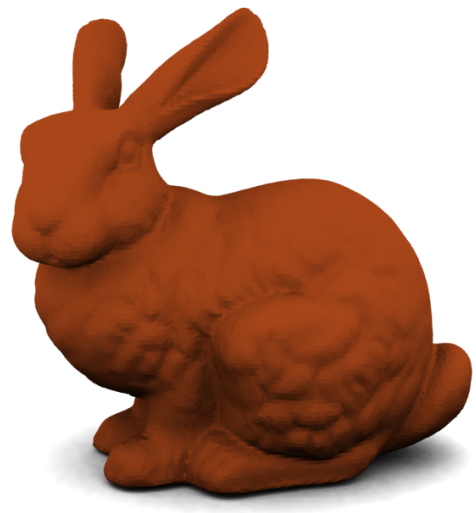
Rendering approaches

- OpenGL
  - First attempt



Rendering approaches

- OpenGL
  - First attempt
- Triangle reconstruction
  - Standard renderer



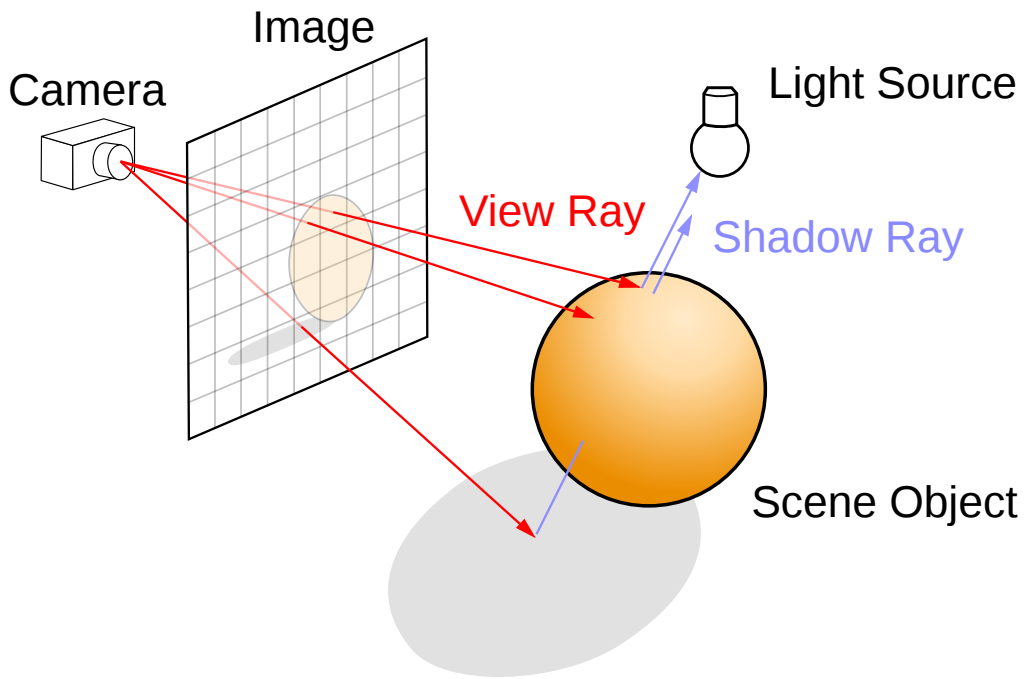
[0]Image taken from en.wikipedia.org

Rendering approaches

- OpenGL
  - First attempt
- Triangle reconstruction
  - Standard renderer
- Ray Tracing
  - Fast on huge scenes
  - Full lighting calculation

Recall on Ray-Tracing

- Start one ray per pixel
- Intersect the scene
- Start new rays
  - Shadows
  - Reflection
  - Refraction
- Combine results



[0]Image taken from en.wikipedia.org

OpenGl Ray-Tracer

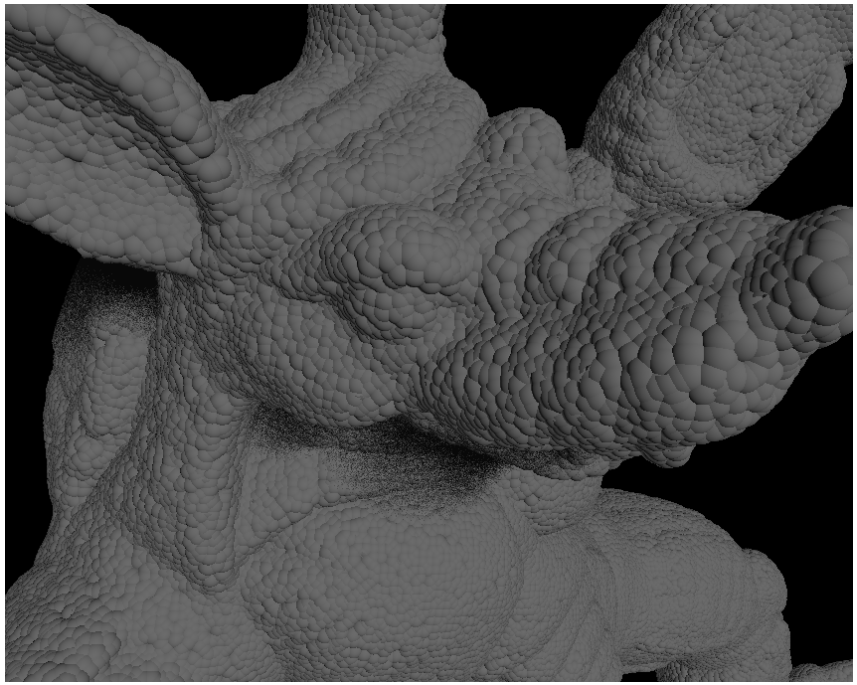
- Scene handling with OpenSG
- Support for Triangles
- Light computation based on normals
- Bounding-Volume Hierarchy
- GPU accelerated

Ray-Point intersection (Spheres)

- Simple
- No normals required
- Point Size?

Ray-Point intersection (Spheres)

- Simple
- No normals required
- Point Size?
- Problems:
  - Rippled surface
  - Object grow



Ray Point intersection (Disc)

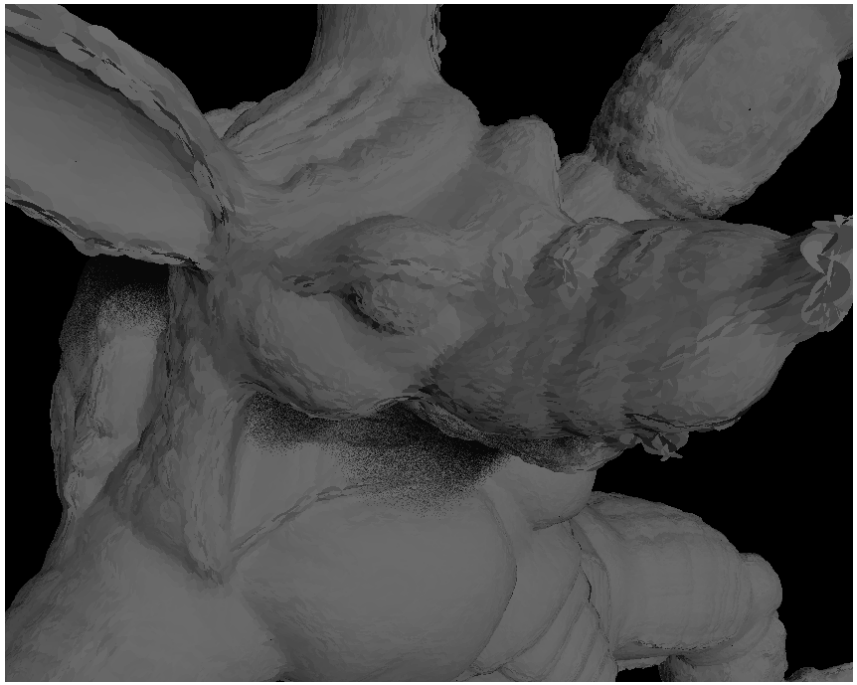
- Accurate position
- Flat surface
- Disc orientation ?
- Disc size ?
- Problem self shadowing

Ray Point intersection (Disc)

- Accurate position
- Flat surface
- Disc orientation ?
- Disc size ?
- Problem self shadowing

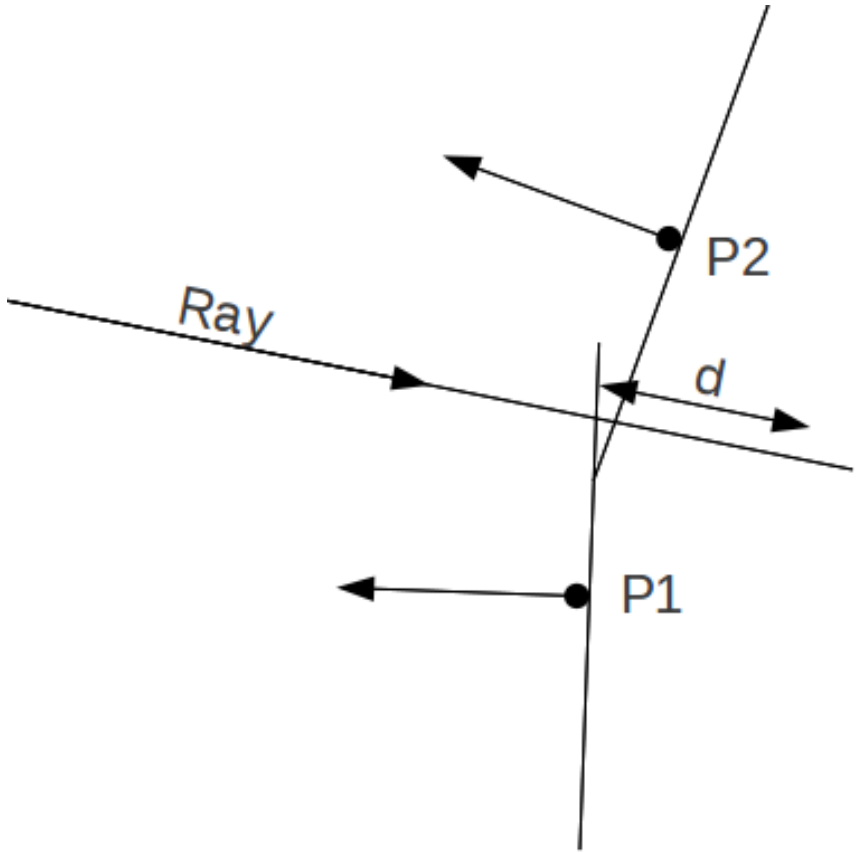
Ray Point intersection (Disc)

- Accurate position
- Flat surface
- Disc orientation ?
- Disc size ?
- Problem self shadowing



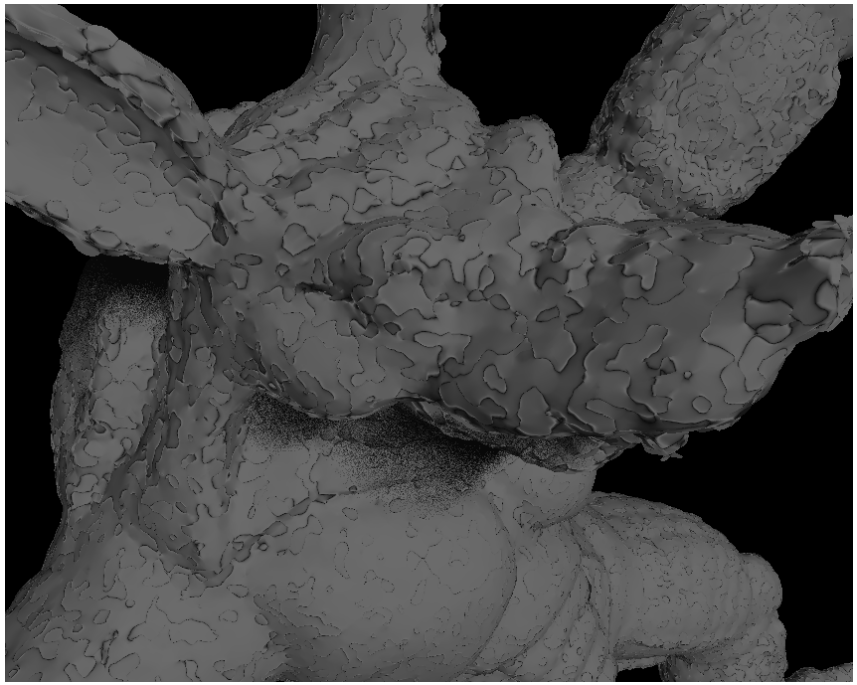
Splats (Shading)

- Use multiple intersection
- 16 nearest points
- Using disc intersection
- Local surface representation
- Using the nearest disc as intersection point



Splats

- Normal selected as weighted average
- $n = \sum_i^N n_i * (1 - \frac{d_i}{r_i})$
- Creates smooth surface
- Requires aligned normals

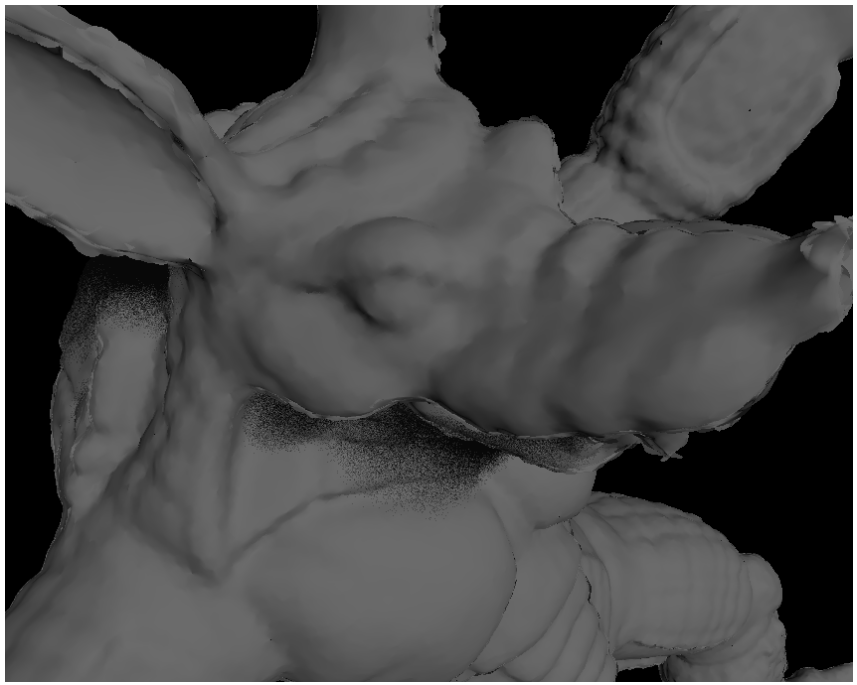


Splats (aligned)

- Alignment towards the ray origin
- Flip normal if $n \bullet ray.d \geq 0$
- Weighted average as before

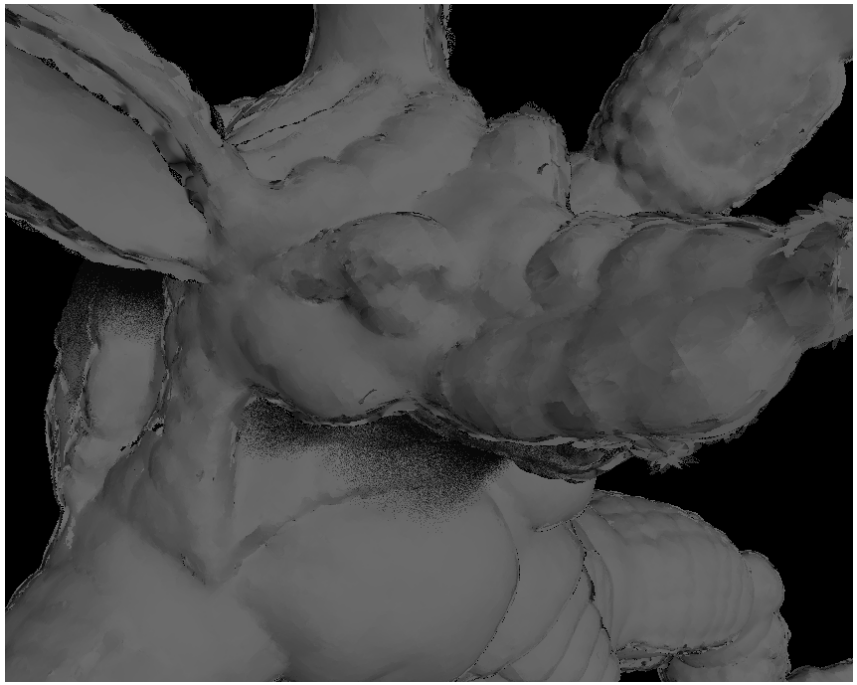
Splats (aligned)

- Alignment towards the ray origin
- Flip normal if $n \bullet ray.d \geq 0$
- Weighted average as before
- Artifacts on edges

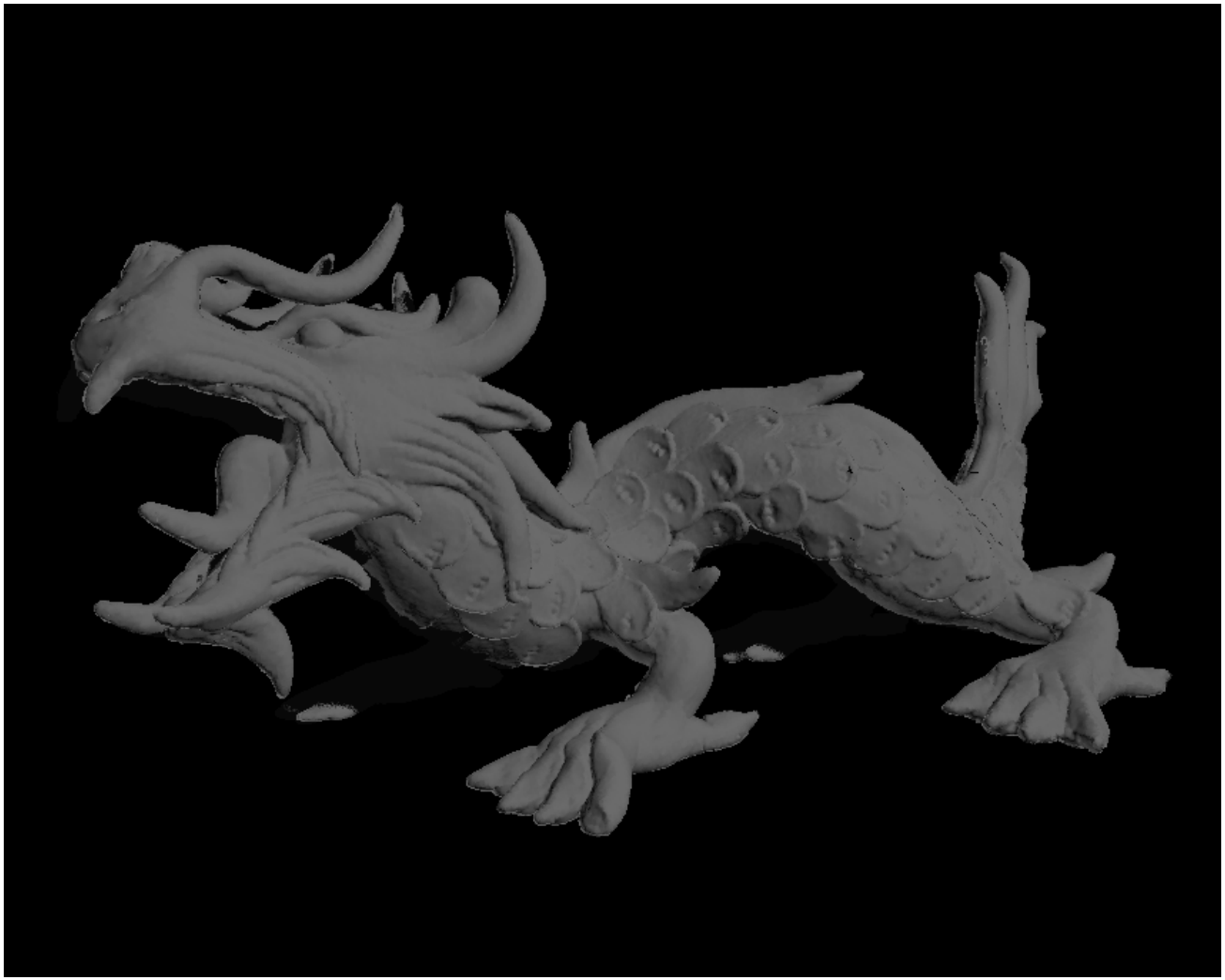


Splats (runtime normal)

- Calculate the normal using intersected points
- Normal is required for intersection test
- Requires more points than weighted average
- Increased render times

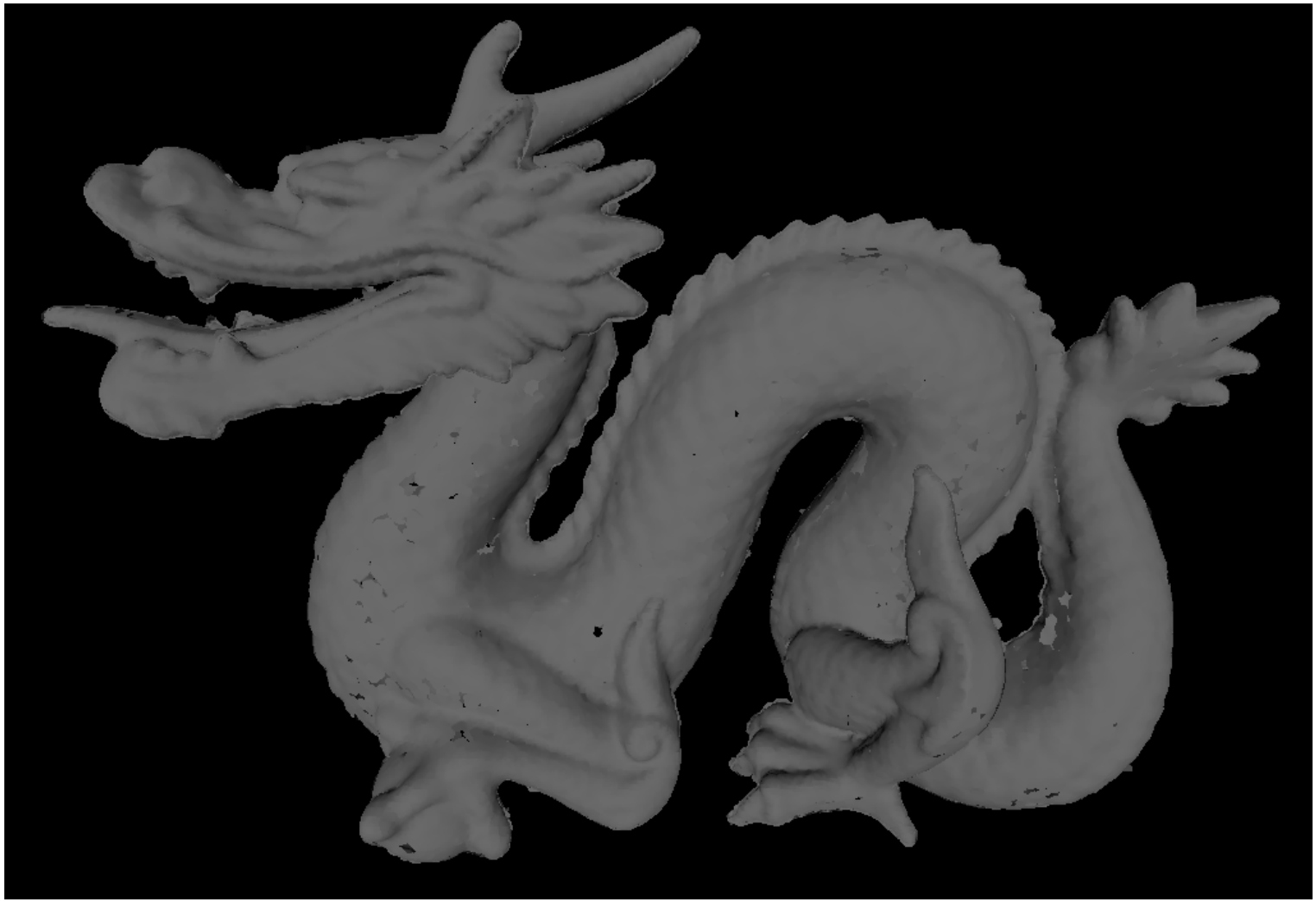


Rendering Results



1.5M points, splats aligned

Rendering Results

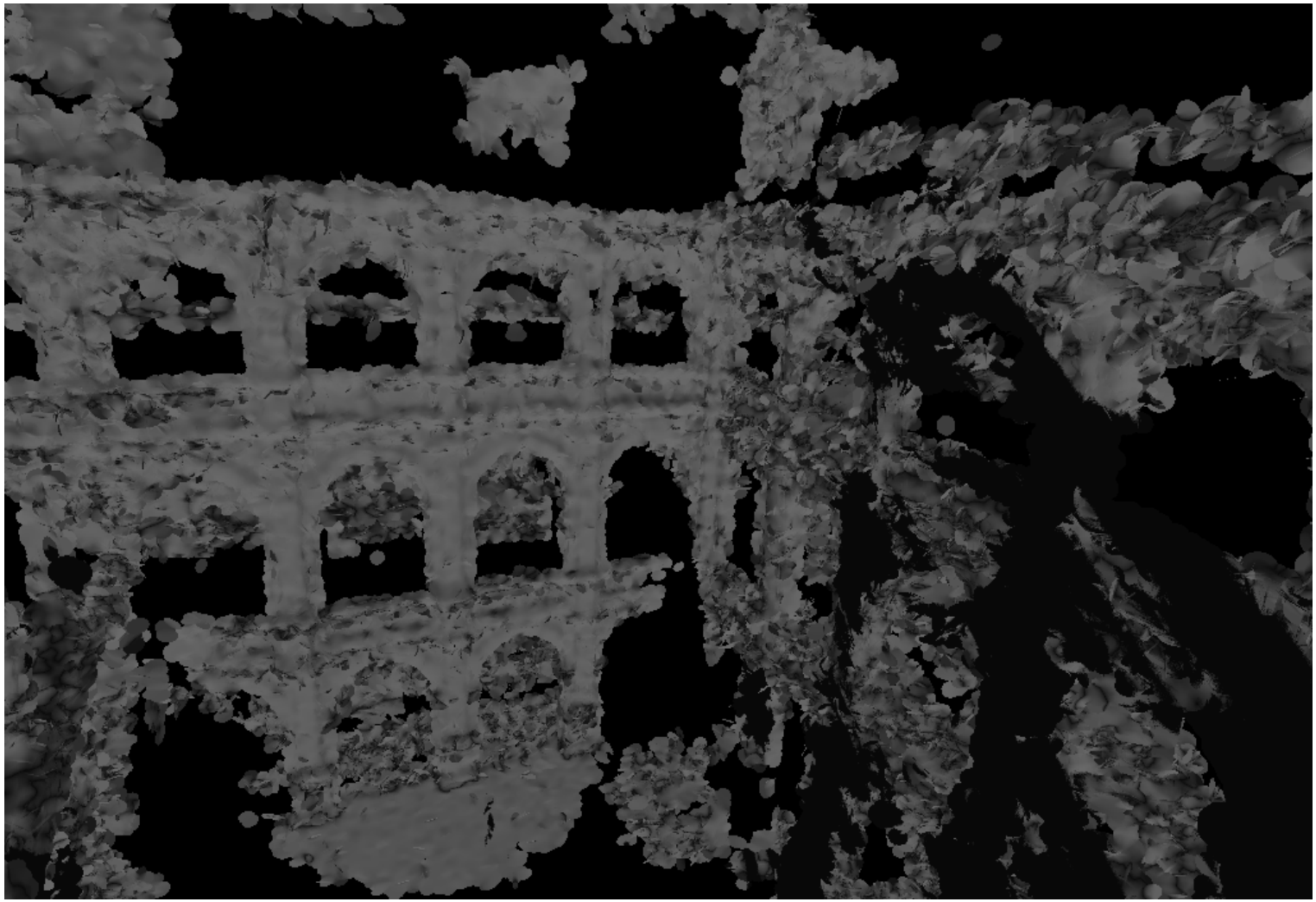


407k points, splats aligned

Rendering Results

1M points, splats aligned

Rendering Results



265k points, splats aligned

Performance

Dataset NrPoints	Bunny 36k		Armadillo 173k		Pisa 1M	
	trace ms	render ms	trace ms	render ms	trace ms	render ms
Sphere	9.6	46.9	5.9	19.7	16.5	48.7
Disk	7.4	38	9.2	25.3	26.3	119
Multi mean	9.2	38	12.4	28	38.9	135
Multi aligned	9.3	38.5	12.3	28	39	135
Multi runtime	39	102	15	32	85	183

Conclusion / Further work

- Good rendering results
- Interactive rendering
- Limited number of Points
- Does not work on noisy datasets
- Requires equal distributed points
- Problems on sharp edges
- Runtime normal estimation requires normals

End of Presentation

Questions?