



# PointAO

## Improved Ambient Occlusion for Point-based Visualization

Sebastian Eichelbaum<sup>1</sup>   Mario Hlawitschka<sup>2</sup>   Gerik Scheuermann<sup>1</sup>

<sup>1</sup> Image and Signal Processing Group, Leipzig University, Germany

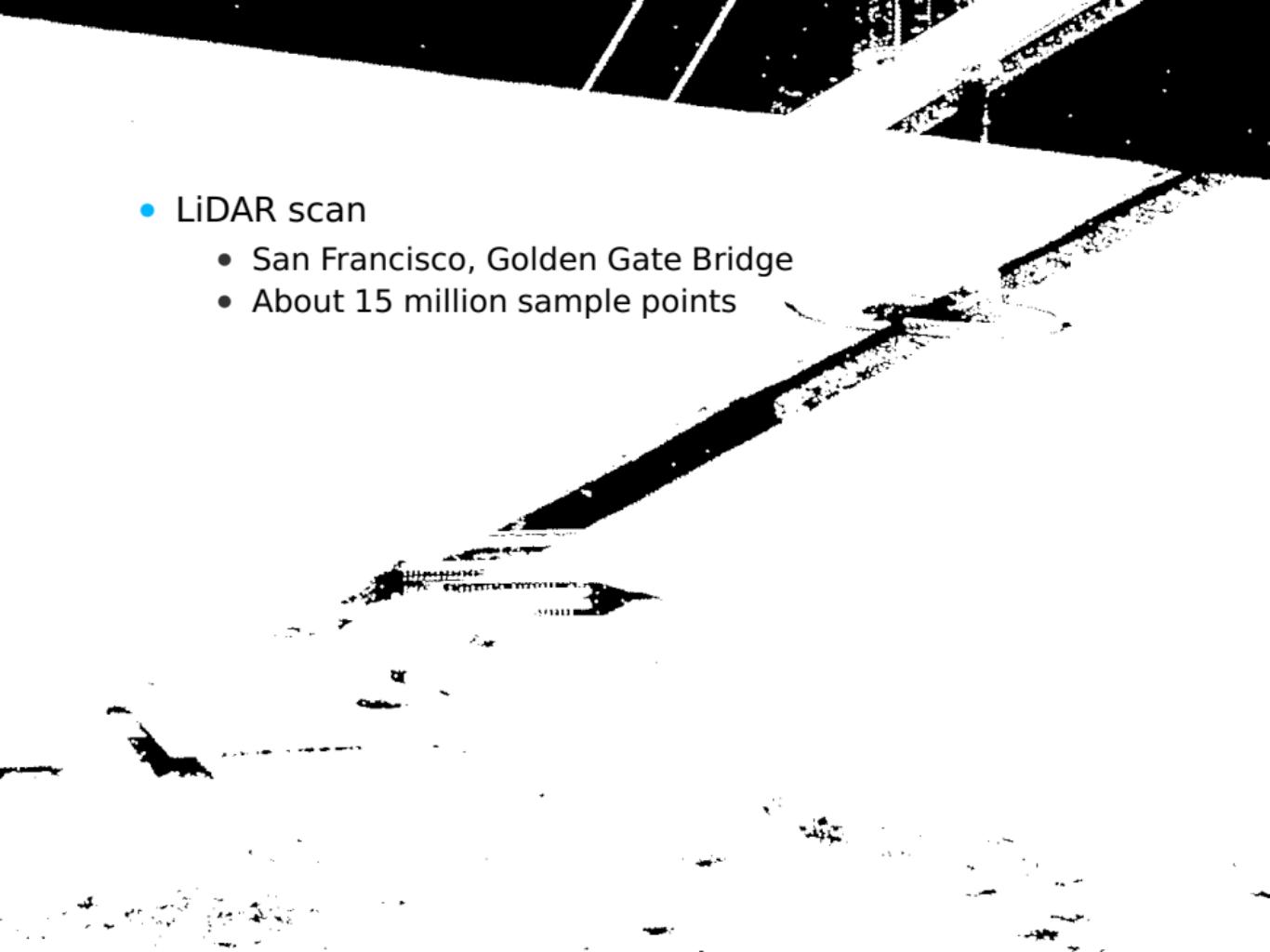
<sup>2</sup> Scientific Visualization Group, Leipzig University, Germany



# Why?

- Isn't plain point and glyph rendering sufficient for visualization?

- LiDAR scan
  - San Francisco, Golden Gate Bridge
  - About 15 million sample points



A grayscale depth map generated by a LiDAR sensor. The image shows a bridge spanning a body of water. A car is visible on the bridge, appearing as a dark silhouette against the lighter background. The water below is represented by a textured gray surface. The sky above is dark, suggesting it might be night or the sensor is operating in low-light conditions.

# Depth-Colormap

only available for LiDAR



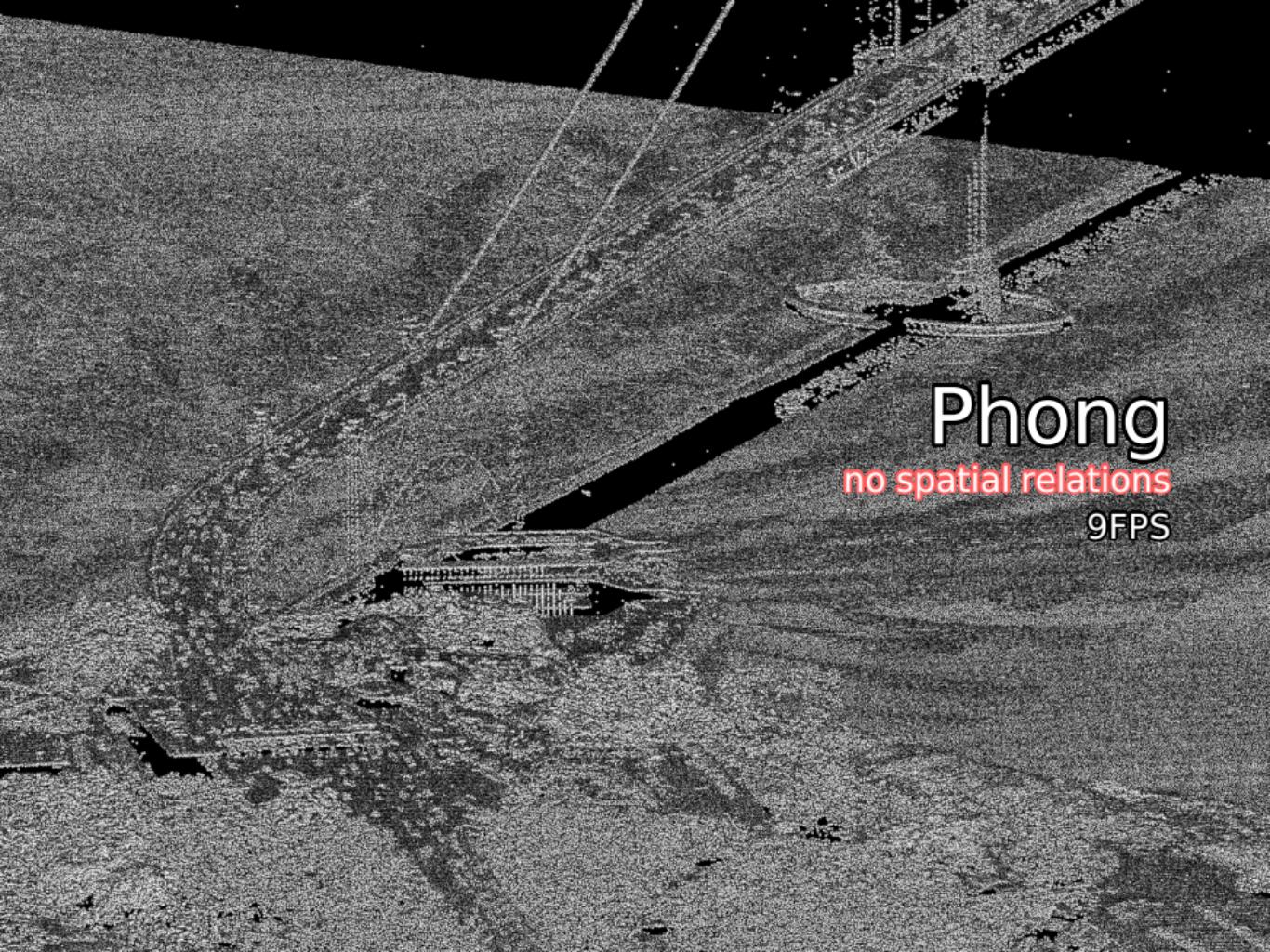
# Why? — The Problem

- Possible solution: **shape from shading**.
    - See Ramachandran et al.
    - Shading in computer graphics?
      - local illumination provides structure
      - global illumination provides relative, spatial information
- Let's try!

V. S. Ramachandran. Perception of shape from shading. *Nature*, 331:163–166, 1988.



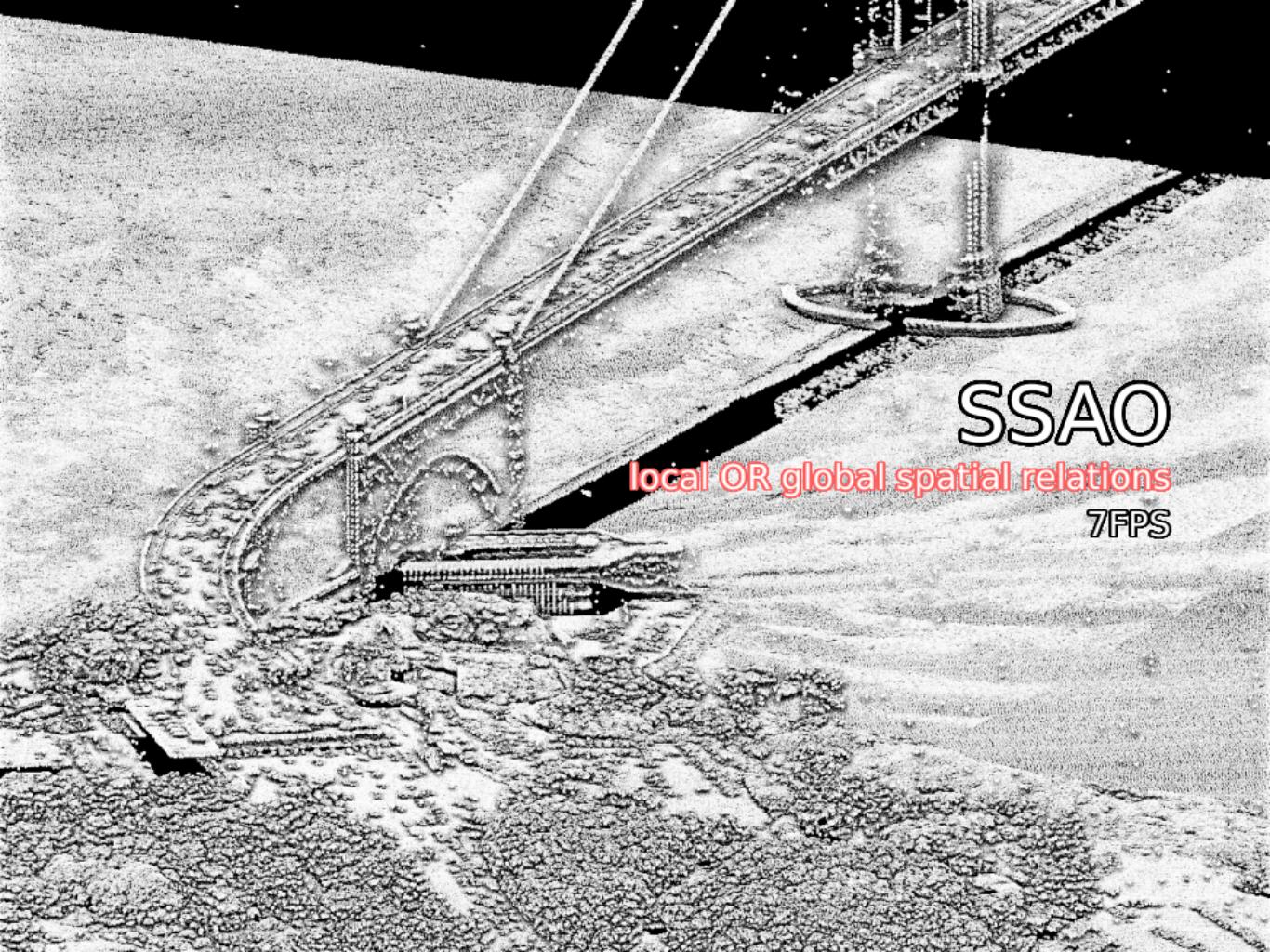
No Shading

A grayscale 3D rendering of a cityscape at night. The scene features a large bridge with multiple towers and cables stretching across the frame. Below the bridge, there's a body of water reflecting the lights. In the background, several buildings are visible, their windows glowing with light. The overall atmosphere is dark and moody, with the light from the structures providing the primary illumination.

Phong

no spatial relations

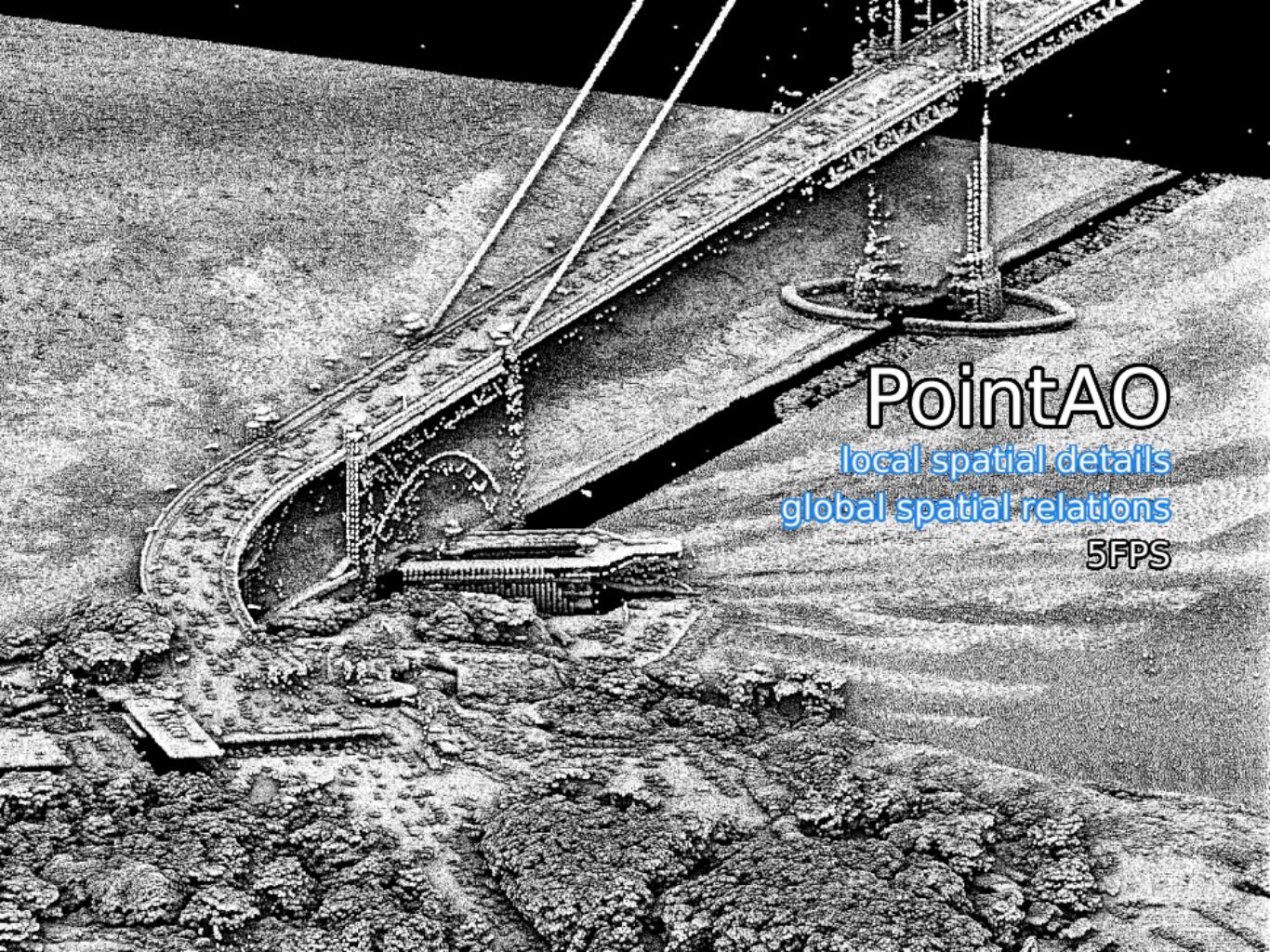
9FPS



**SSAO**

local OR global spatial relations

7FPS

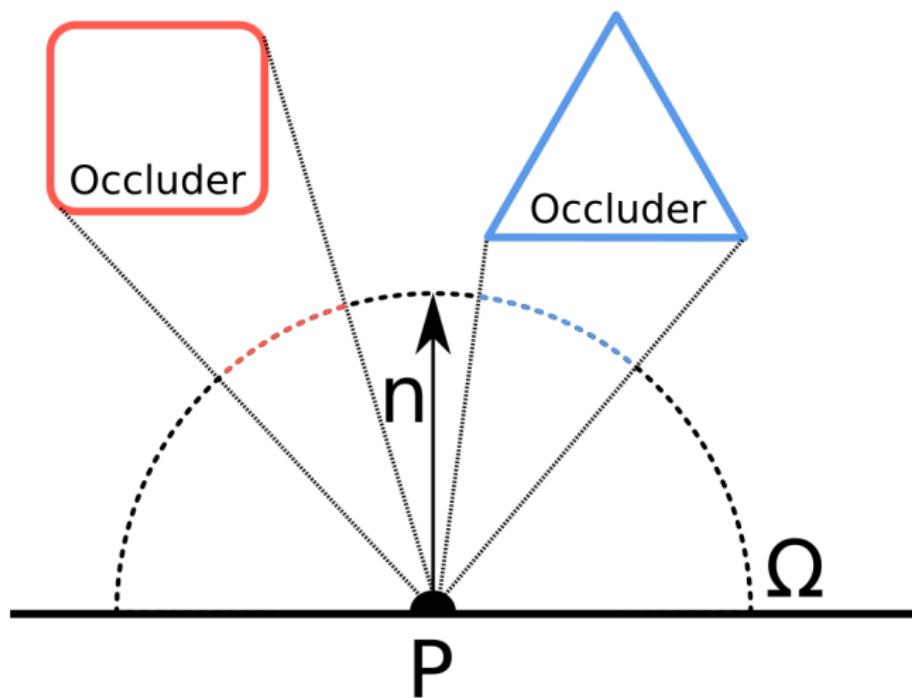


# PointAO

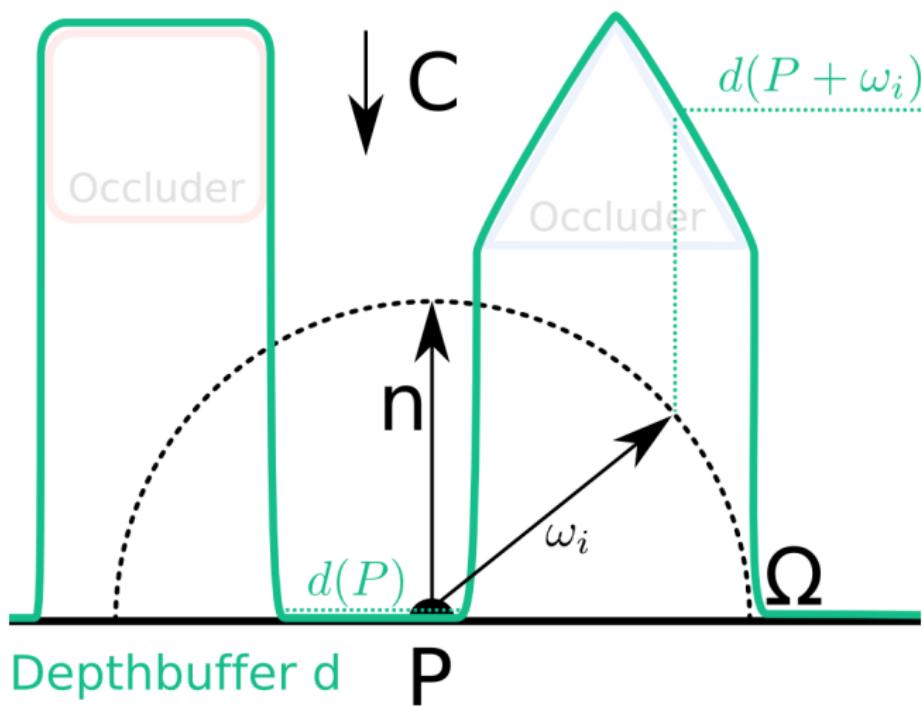
local spatial details  
global spatial relations

5FPS

# How? — Ambient Occlusion

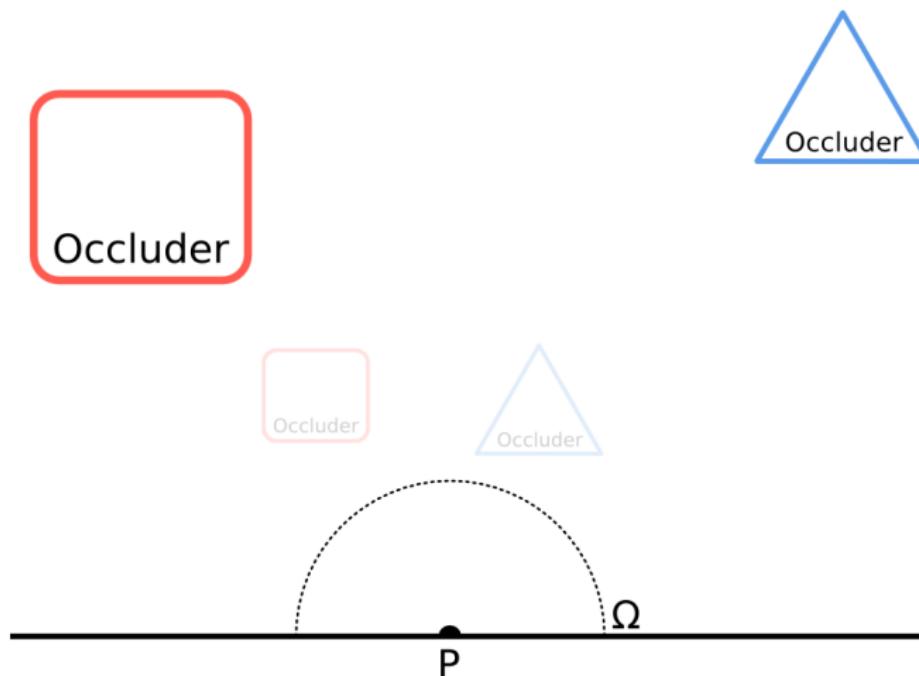


# How? — SSAO



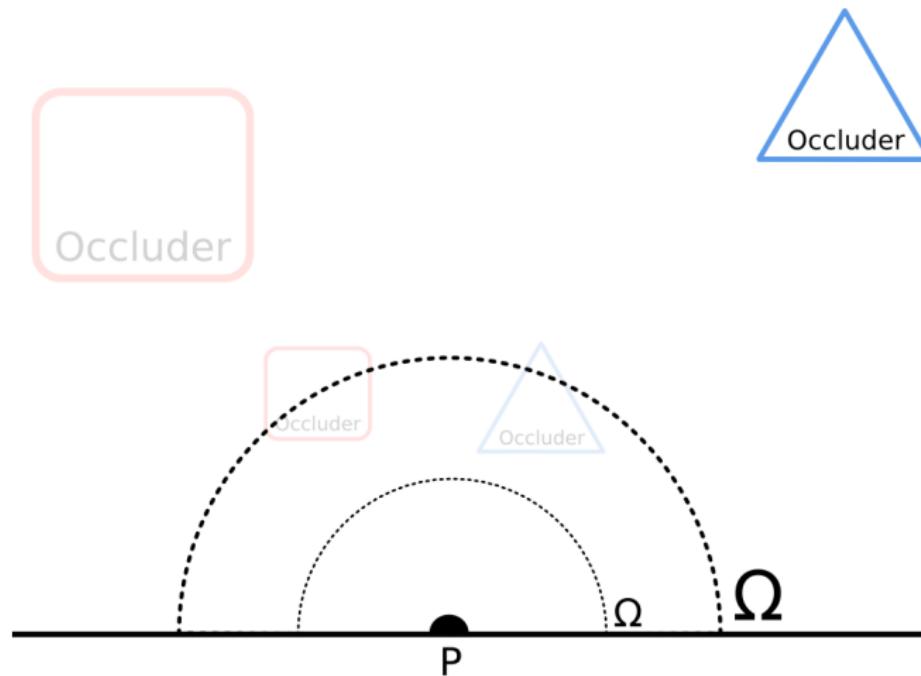


# How? — From SSAO to PointAO



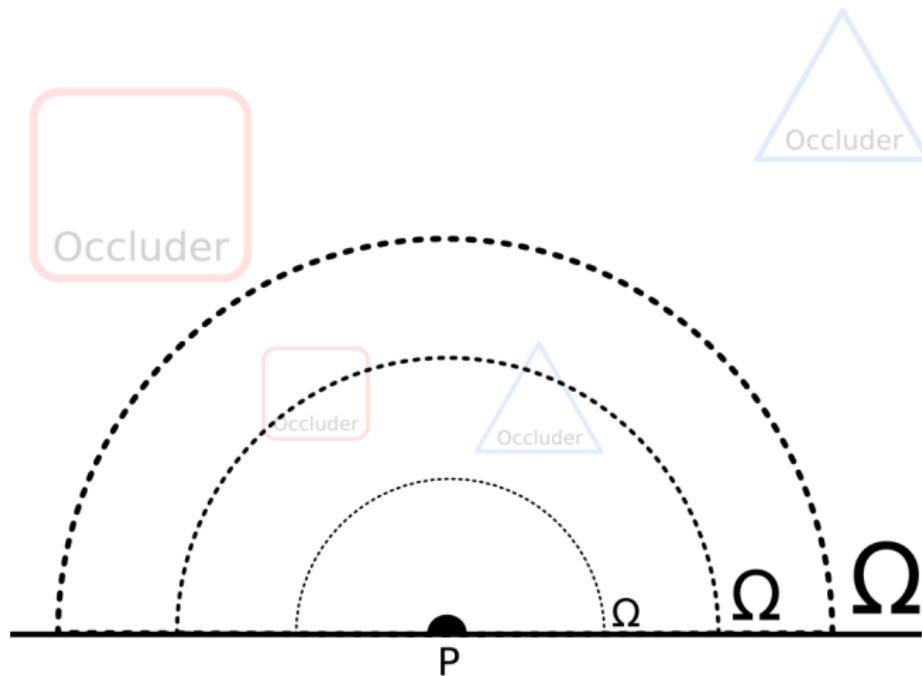


# How? — From SSAO to PointAO

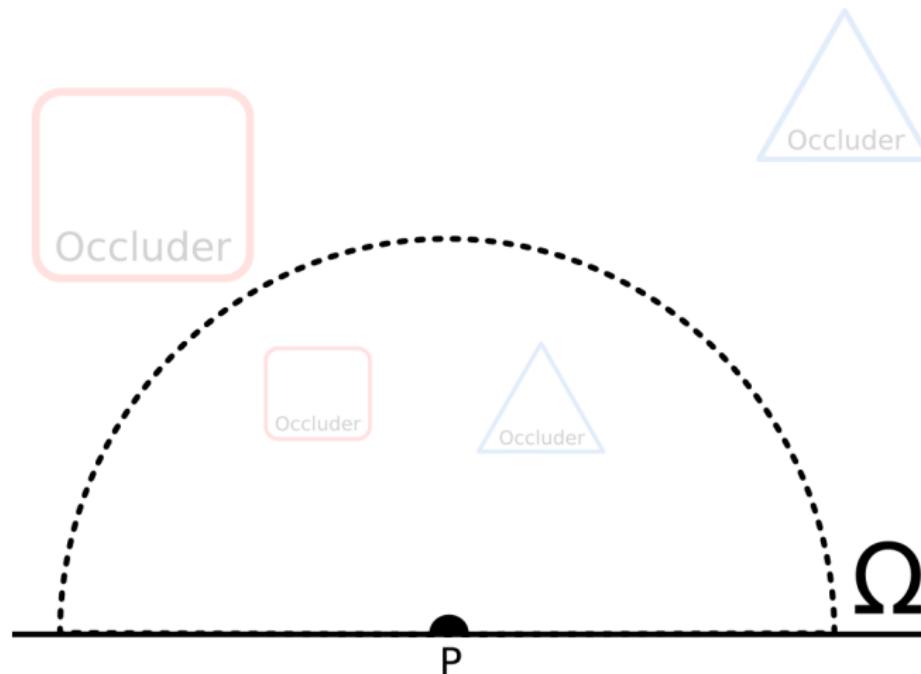




# How? — From SSAO to PointAO



# Why not a single, large hemisphere?

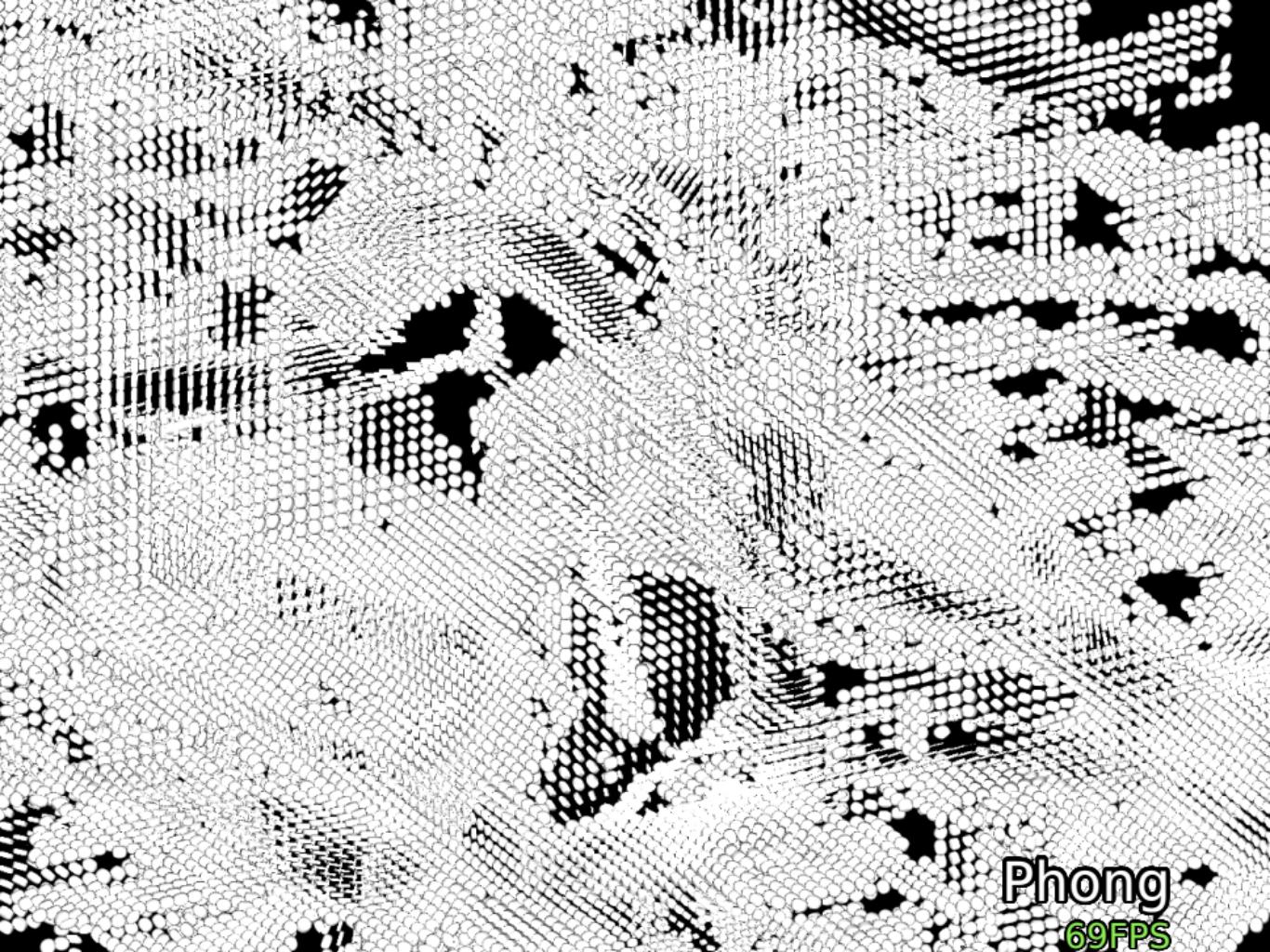




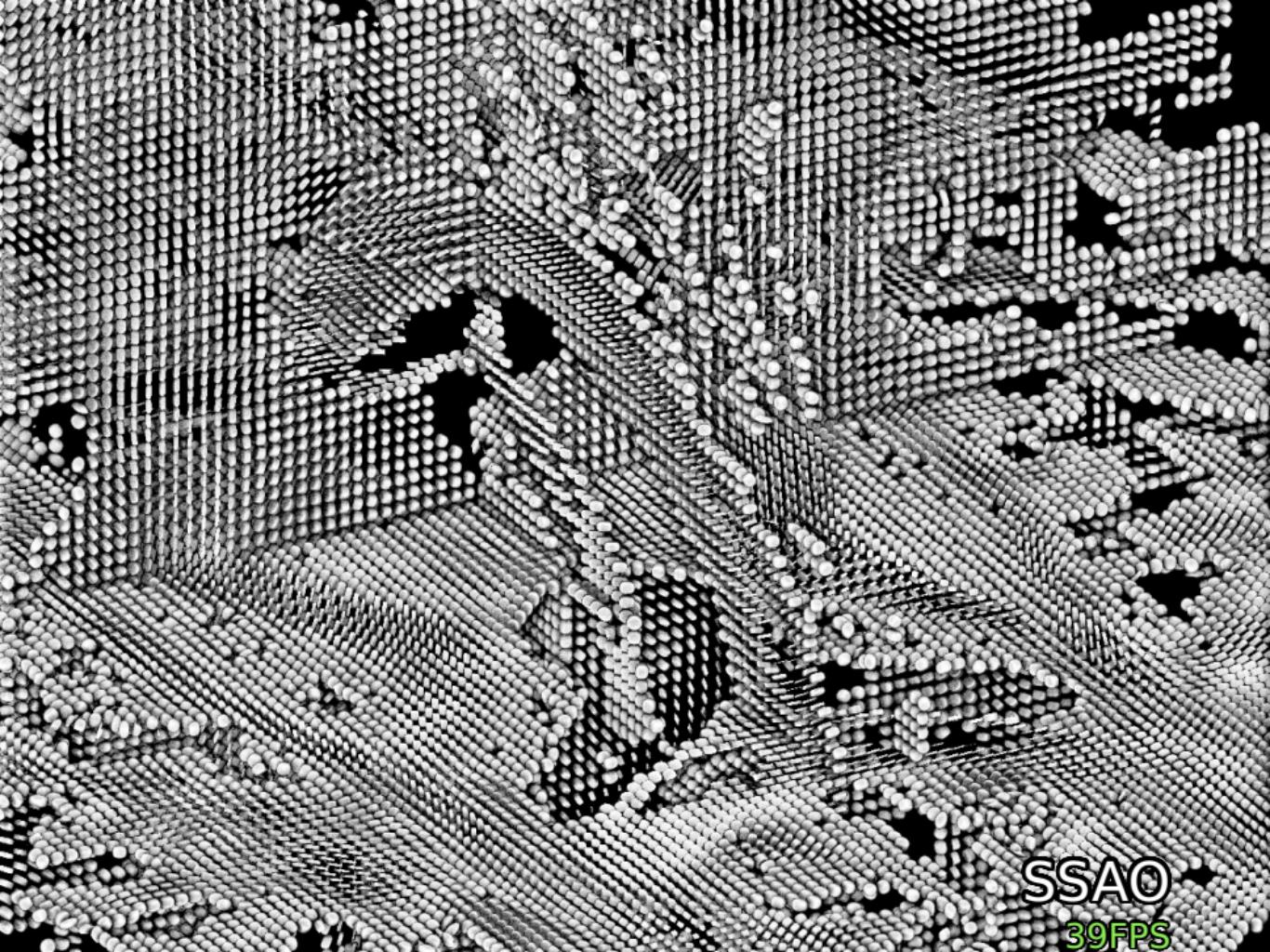
# How? — PointAO

- Similar to LineAO
- Modified:
  - Sampling Scheme
  - Radius Scaling Scheme
  - Weighting Function

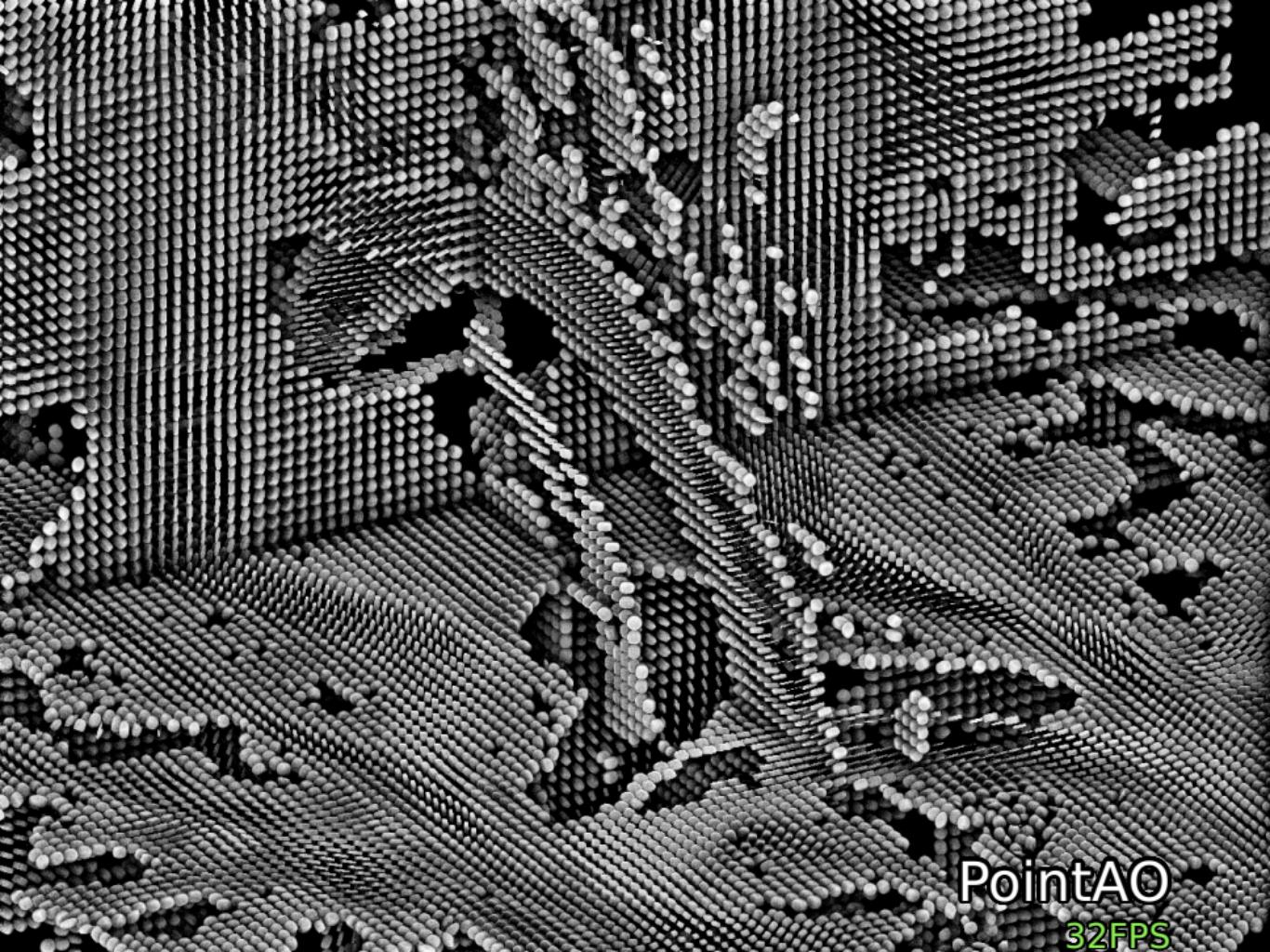
Eichelbaum et al.: LineAO – improved three-dimensional line rendering. IEEE Transactions on Visualization and Computer Graphics 19, 3 (2013), 433–445



Phong  
69FPS

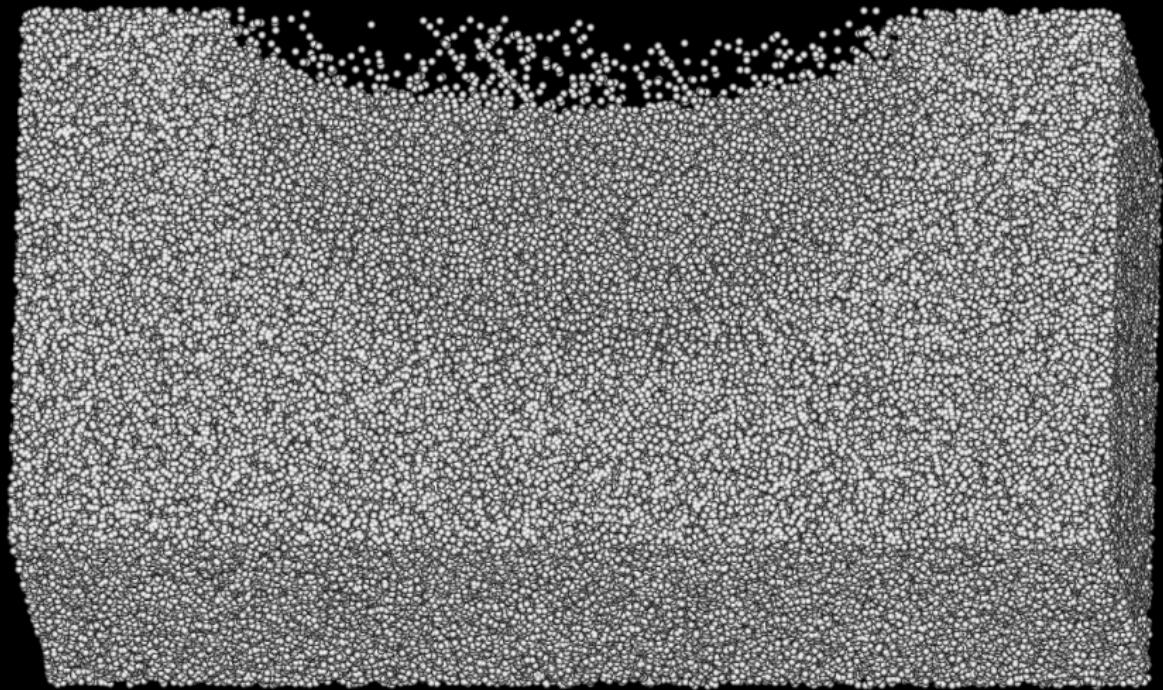


SSAO  
39FPS

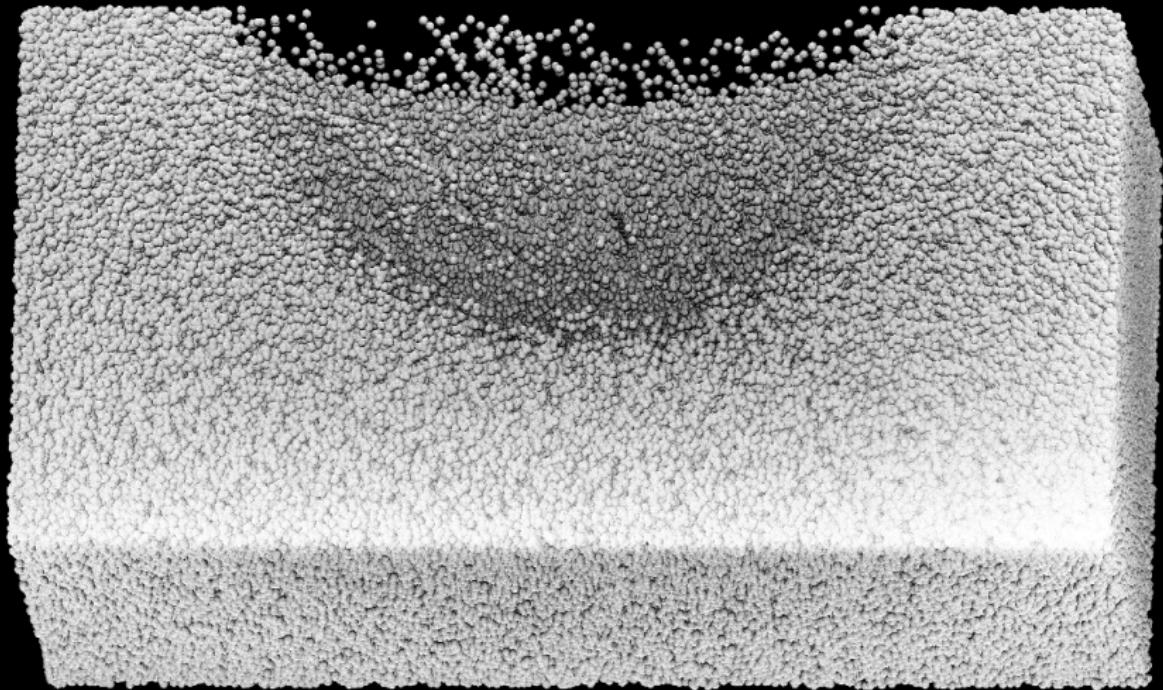


PointAO

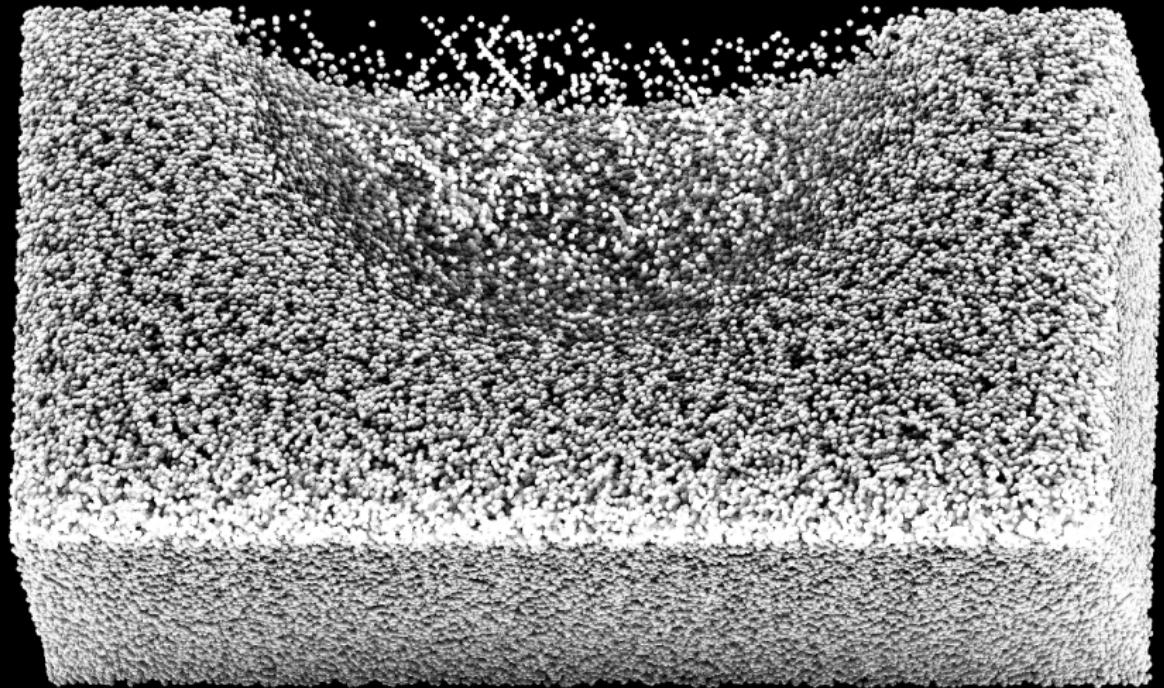
32FPS



Phong  
30FPS



SSAO  
19FPS



PointAO

12FPS



## Results — Features

- Greatly improved structural and spatial perception
- Simultaneous depiction of local and global structures
- Renders in real time without pre-computation
- Consistency under modification and interaction



## Results — Limitations

- Not suited for coarse data
- Does not work for two dimensional and quasi two dimensional data

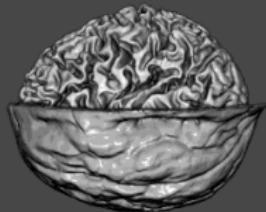


## Future Work

- Combination with other visualization paradigms



Thank You!  
Questions?



**OpenWalnut**  
Visualization in a Nutshell





# Details – Math

$$\text{PointAO}_{S_r, S_h, r_0}(P) =$$

$$\sum_{j=0}^{S_r-1} \text{AO}_{S_h, j}(P, \frac{1}{1-d(P)} * z(P) * (j^2 + j * r_0))$$

$$\text{AO}_{S, l}(P, r) =$$

$$\frac{1}{s} \sum_{i=1}^s [(1 - V(r\omega_i, P)) g_l(r\omega_i, P)]$$

$$V(\omega, P) = \begin{cases} 1 & \text{if } d(P) - d(P + \omega) < 0 \\ 0 & \text{else,} \end{cases}$$

$$g_l(\omega, P) = g_l^{\text{depth}}(\omega, P) \cdot g_l^{\text{light}}(\omega, P)$$

$$\Delta d_l(\omega, P) = d(P) - d(P + \omega) \in [-1, 1]$$

$$\delta(l) = \left(1 - \frac{l}{S_r}\right)^2 \in (0, 1]$$

$$h(x) = 3x^2 - 2x^3, \forall x \in [0, 1] : h(x) \in [0, 1]$$

$$g_l^{\text{depth}}(\omega, P) =$$

$$\begin{cases} 0, & \text{if } \Delta d(\omega, P) > \delta(l) \\ 1, & \text{if } \Delta d(\omega, P) < \delta_0 \\ 1 - h\left(\frac{\delta(l) - \delta_0}{\delta(l) - \delta_0}\right), & \text{else.} \end{cases}$$

$$g_l^{\text{light}}(\omega, P) = 1 - \langle \omega, n(P) \rangle$$

- Scale radius according to depth of pixel.
- Weight each occluder with  $g(r\omega_i, P)$
- Classify and weight according to distance and used hemisphere
- Incorporate local light to retain shape of rendered objects (especially useful for glyphs)



# LineAO vs. PointAO

- Sampling Scheme
    - No Gaussian pyramid
    - No reduction of samples
- more details



# LineAO vs. PointAO

- Radius Scaling
  - No linear scaling
  - Instead:  $\frac{1}{1-d_j(P)} * z(P) * (j^2 + j * r_0)$ 
    - include depth of pixel
    - more near pixels use smaller sampling radii for more detail
    - far away pixels use larger sampling radii for smooth shadows when behind others
    - = darken holes/cracks



# LineAO vs. PointAO

- Weighting Function
  - Ignore scattered light of far away occluders
  - Instead: use local BRDF to emphasize shape
    - Retains shape of glyphs