WebNPR

Alan Lee, Jordan Hochman, Maya Diaz Huizar

Introductions

Alan Lee
Master's in Computer Graphics and Game Technology
LinkedIn: /in/soohyun-alan-lee/
GitHub handle: /Alan7996
Personal website: alannos.com

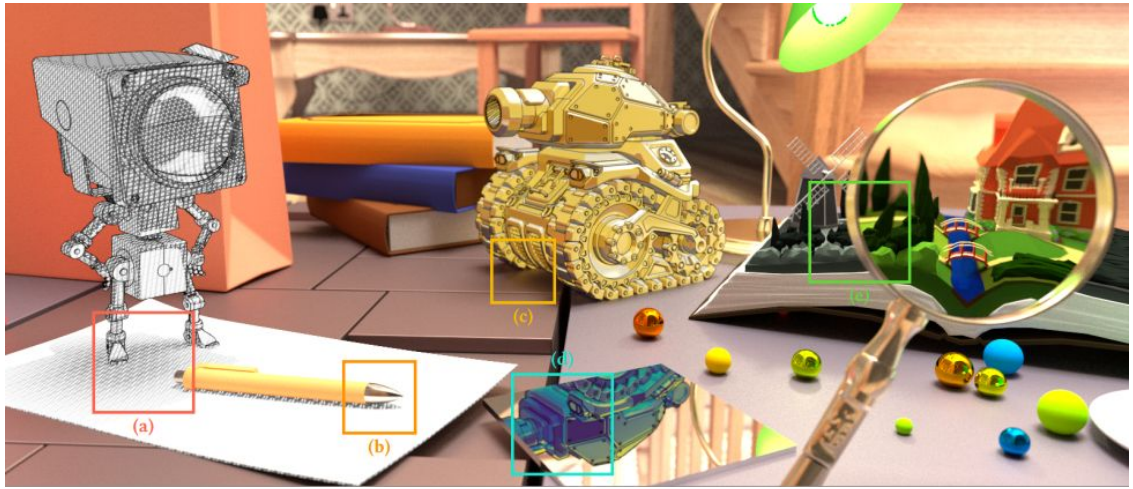
Jordan Hochman
Master's in Computer Science
LinkedIn: /in/jhochman24/
GitHub handle: /JHawk0224
Personal website: jordanh.xyz

Maya Diaz Huizar
Master's in Computer Science
LinkedIn: /in/maya-diaz-huizar/
GitHub handle: /Aorus1

WebGPU Pathtracer + NPR, Why?

Goal: Explore WebGPU as a platform for artists' exploration of stylization and simulation

- WebGPU-based pathtracer
- Stylized Rendering as a Function of Expectation (SIGGRAPH 2024)
- Cloth Simulation

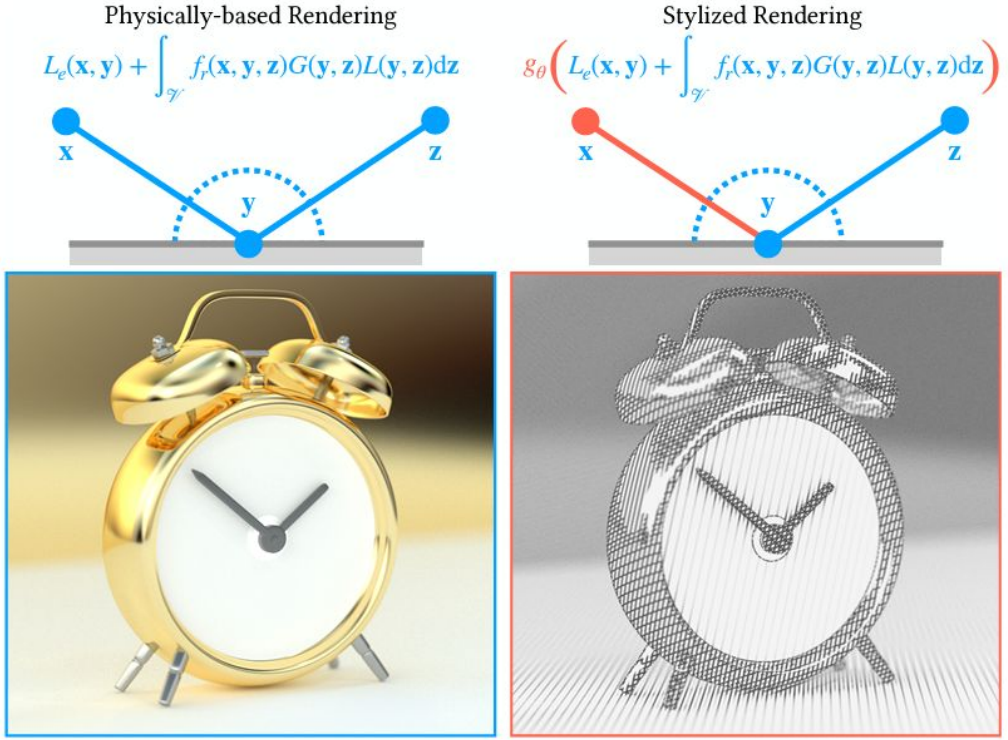


NPR how?

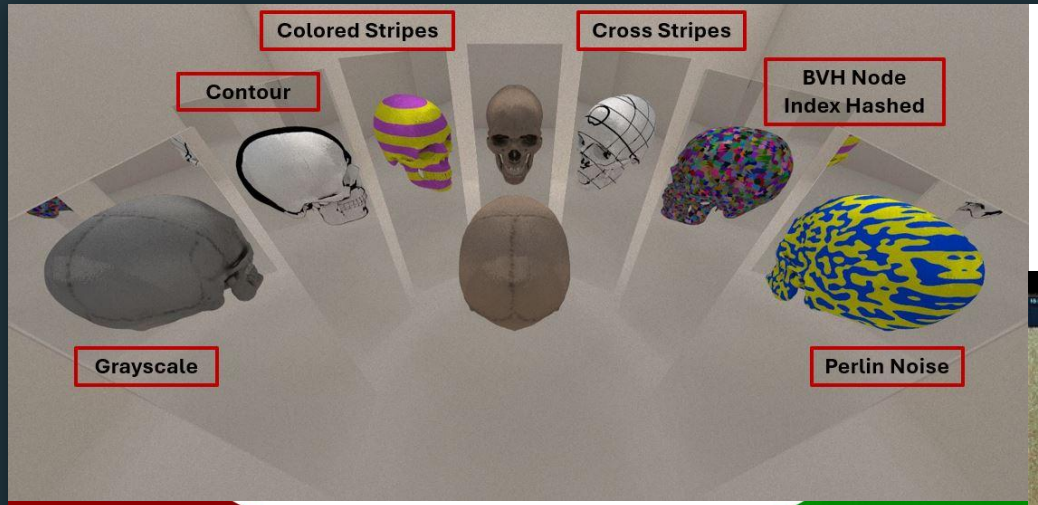
1. On each material shade kernel call…
2. If the current parameters qualify for ANY stylization…
    a. Generate additional rays to estimate incoming radiance
    b. Apply stylization function as expectation on these queries

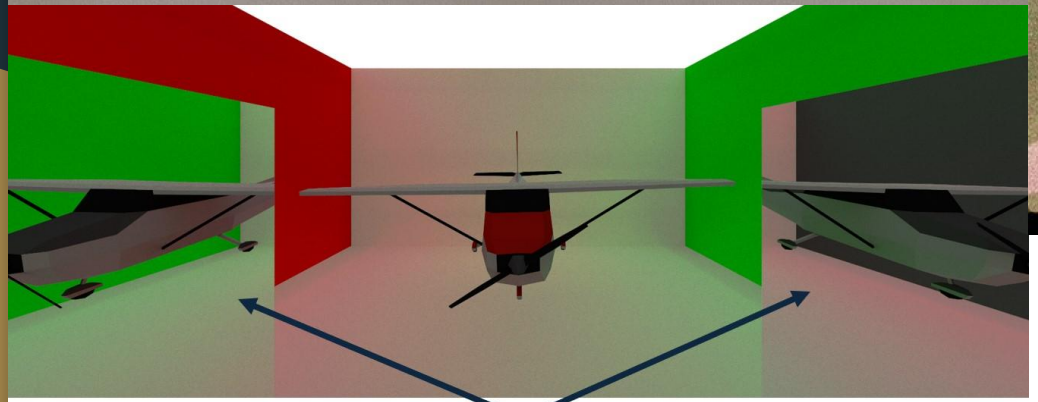
“StyleContext” captures

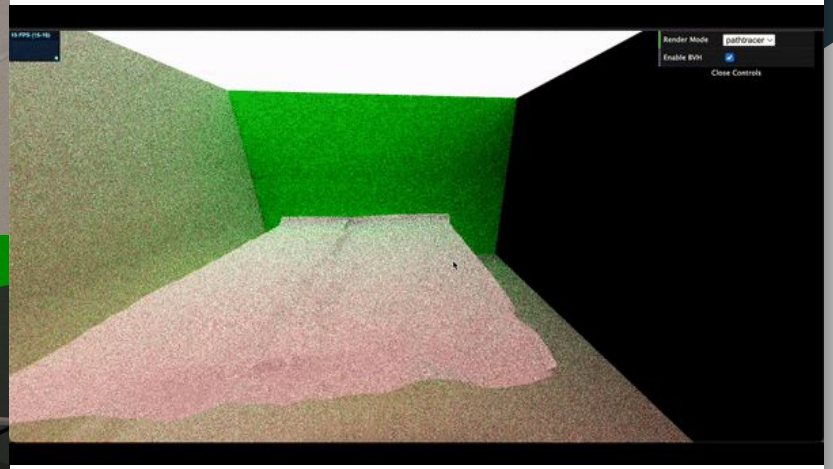
- ObjectId, MaterialId, Path Prefix, World position, Surface normal, Incoming ray direction, BVH node index, etc.

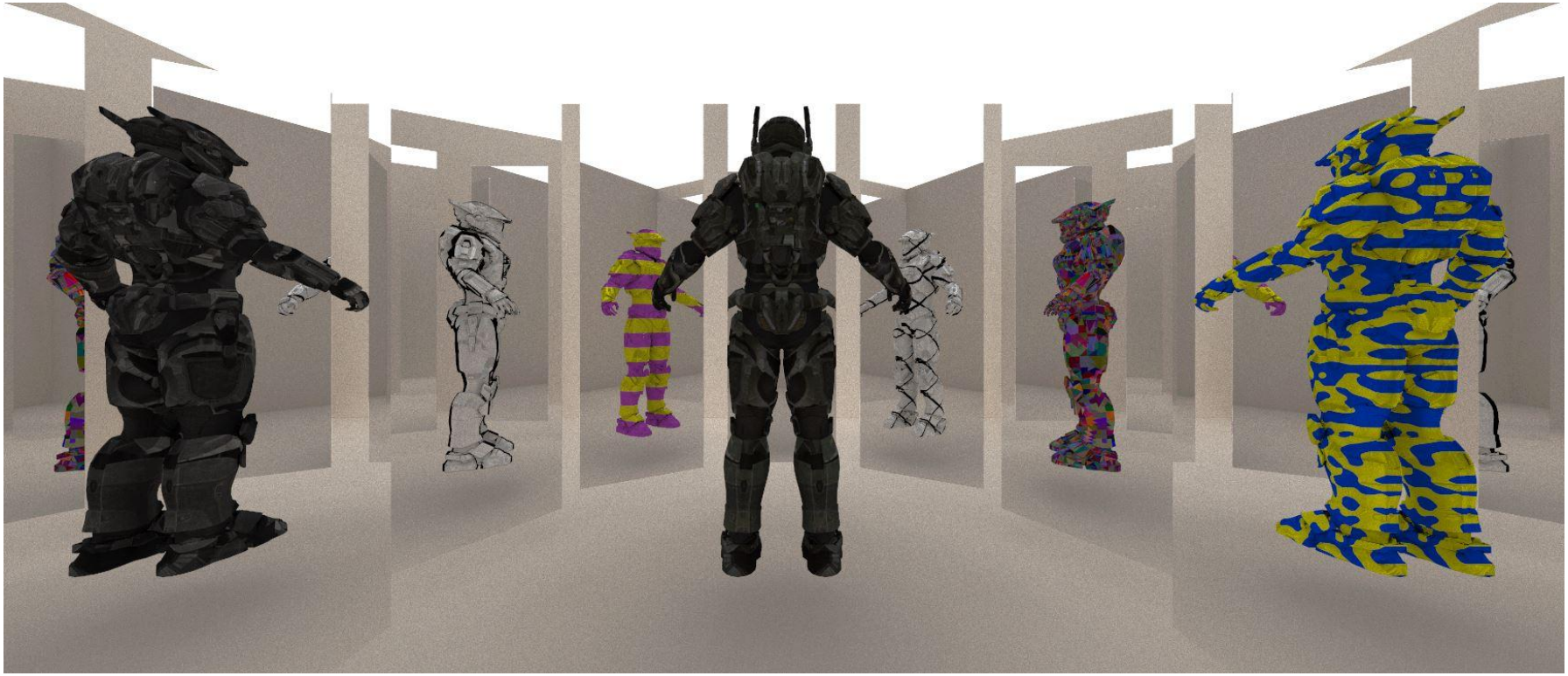


Results


Colored Stripes
Cross Stripes
Contour
BVH Node Index Hashed
Grayscale
Perlin Noise

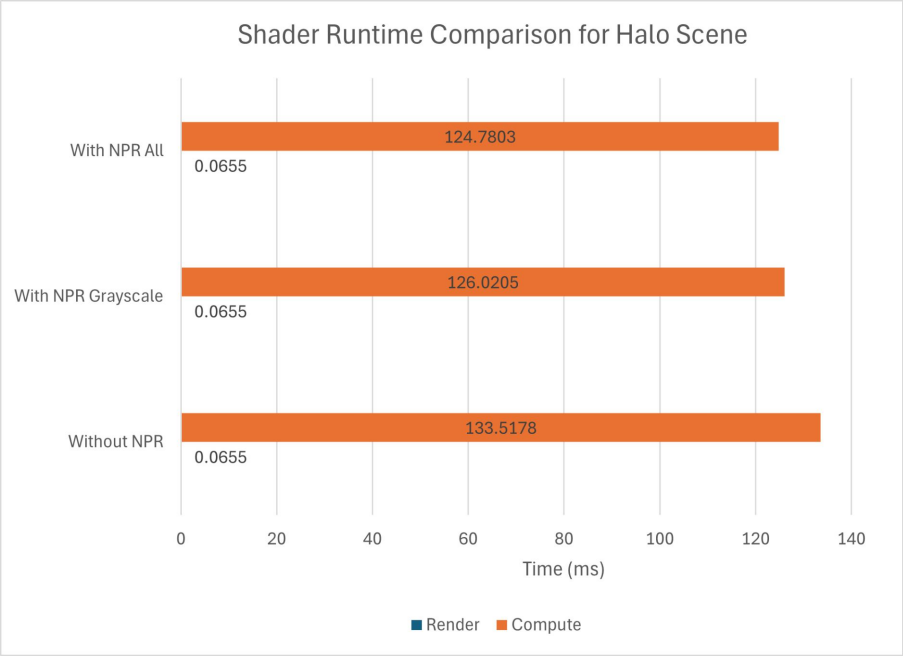

Only plane grayscaled
Reflections still colored
Entire scene grayscaled

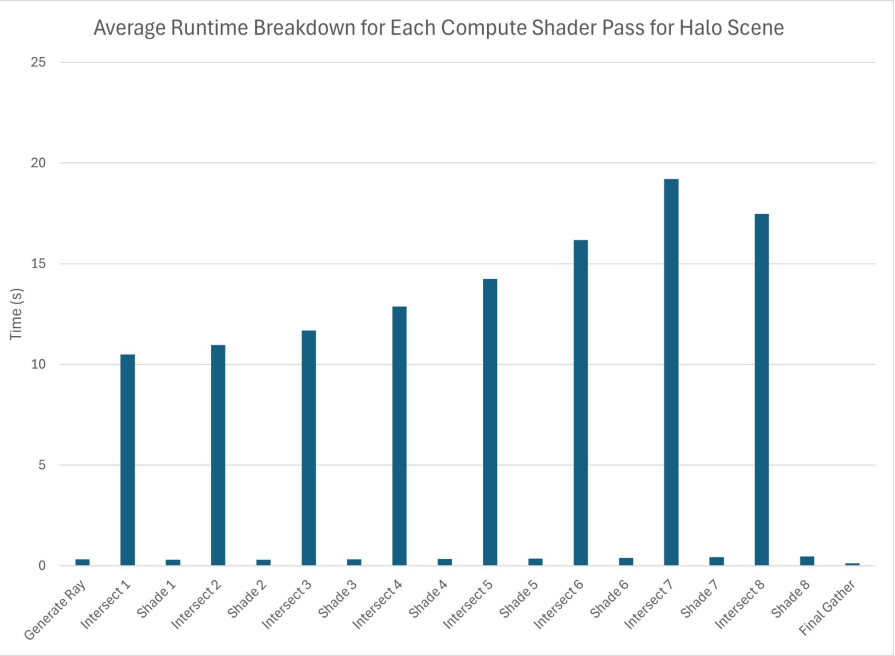

Render Mode
pathtracer
Enable BVH
Close Controls



Hero’s Room

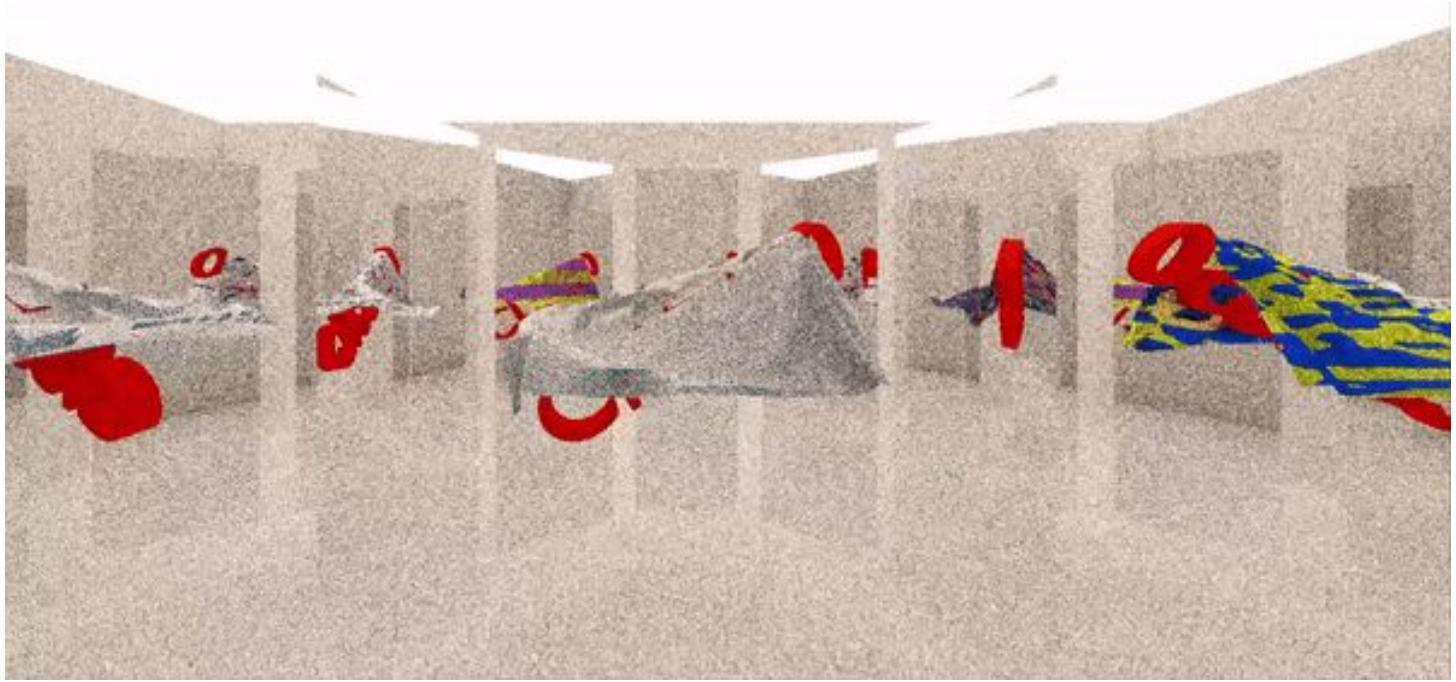
Performance





- Surprisingly, gained performance WITH NPR
- Very tiny material shade compute pass runtime explains minimal slow down, but why faster?

Thank you!



Alan Lee
alanlee.shl@gmail.com

Jordan Hochman
jhawk111@icloud.com

Maya Diaz Huizar
mayajhuizar@gmail.com

Credits

- Stylized Rendering as a Function of Expectation (2024) (http://cv.rexwe.st/pdf/srfoe.pdf)
- Progressive Simulation for Cloth Quasistatics (2023) (https://pcs-sim.github.io/pcs-main.pdf)
- Progressive Dynamics for Cloth and Shell Animation (2024) (https://pcs-sim.github.io/pd/progressive-dynamics-main.pdf)

- Vite (https://vitejs.dev/)
- loaders.gl (https://loaders.gl/)
- dat.GUI (https://github.com/dataarts/dat.gui)
- stats.js (https://github.com/mrdoob/stats.js)
- wgpu-matrix (https://github.com/greggman/wgpu-matrix)
- wgpu-basecode (https://github.com/CIS5650-Fall-2024/Project4-WebGPU-Forward-Plus-and-Clustered-Deferred)

- Halo Spartan Model (https://skfb.ly/6QVvM)
- Suzanne (https://github.com/KhronosGroup/glTF-Sample-Models/tree/main/2.0/Suzanne)
- Person (https://poly.pizza/m/qbDLeTtb8K)
- Skull (https://sketchfab.com/3d-models/skull-downloadable-1a9db900738d44298b0bc59f68123393)
- Small Airplane (https://poly.pizza/m/7cvx6ex-xfL)

Credits

From Rex West et al., Stylized Rendering as a Function of Expectation (2024)

Credits

Algorithm 1: Tree sampling for stylized rendering

```
for pixel ∈ image do
    ray = camera.sampleRay(pixel)
    x = ray.origin
    y = scene.intersects(ray)
    sample = W_p(x, y)G(x, y)stylize(x, y)/p(x, y)
    image[pixel].accumulate(sample)
Input: x: the previous vertex, y: the current vertex, θ: the stylization
       parameters (e.g. path prefix, vertex depth, position, time, etc.)
Output: L_s: the stylized radiance
Function stylize (x, y):
    style = determineStyle(θ, y)
    sampleCount = style.requiredSamples(θ, y)
    samples = []
    for i ∈ {0, ..., sampleCount − 1} do
        if isTerminal(y) then
            L_o = L_e(x, y)
        else
            ray = sampleNextRay(x, y)
            z_i = scene.intersects(ray)
            L_i = stylize(y, z_i)
            L_o = L_e(x, y) + f_r(x, y, z_i)G(y, z_i)L_i/p(z_i|x, y)
        samples.add(L_o)
    L_s = style.apply(θ, samples)
    return L_s
```

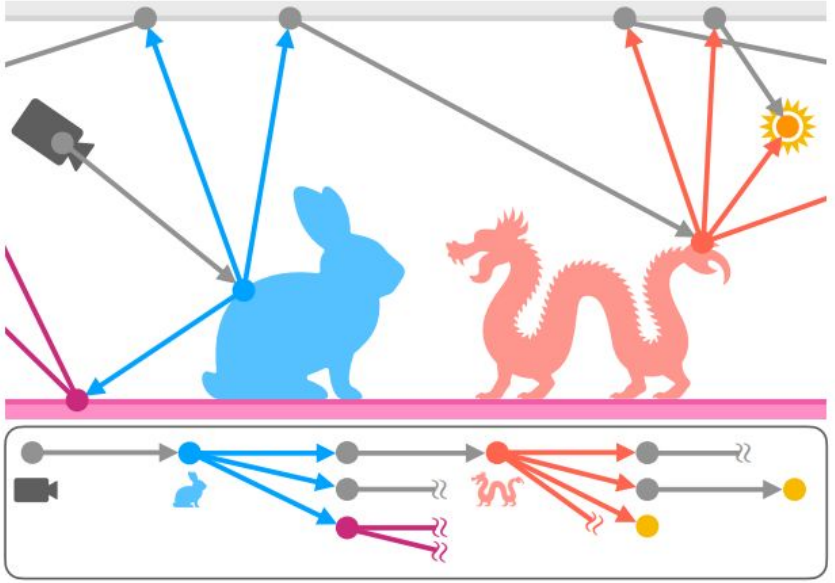
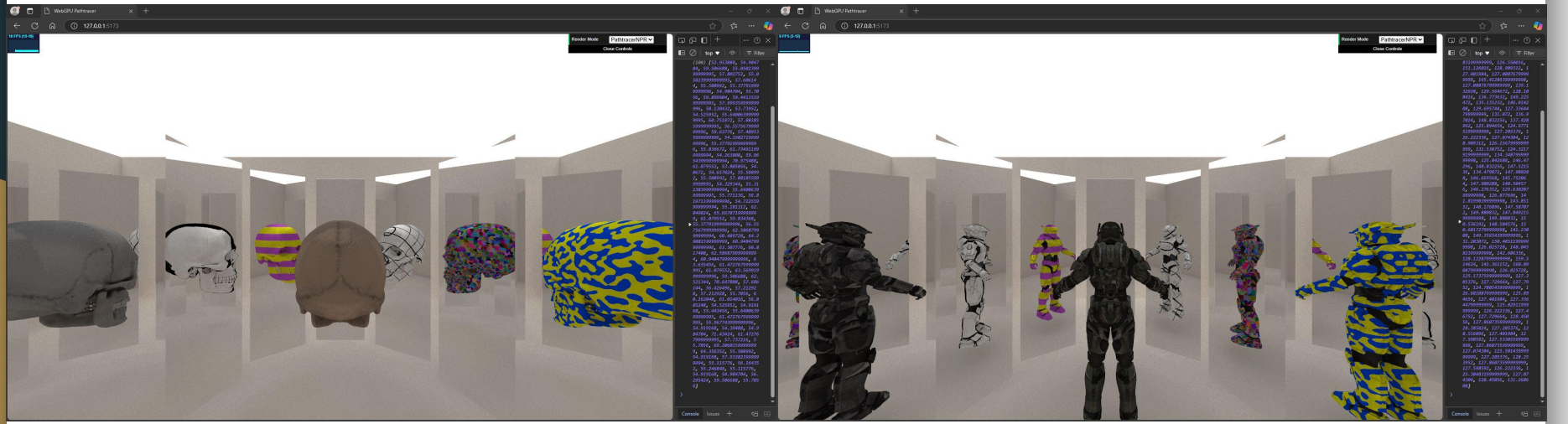


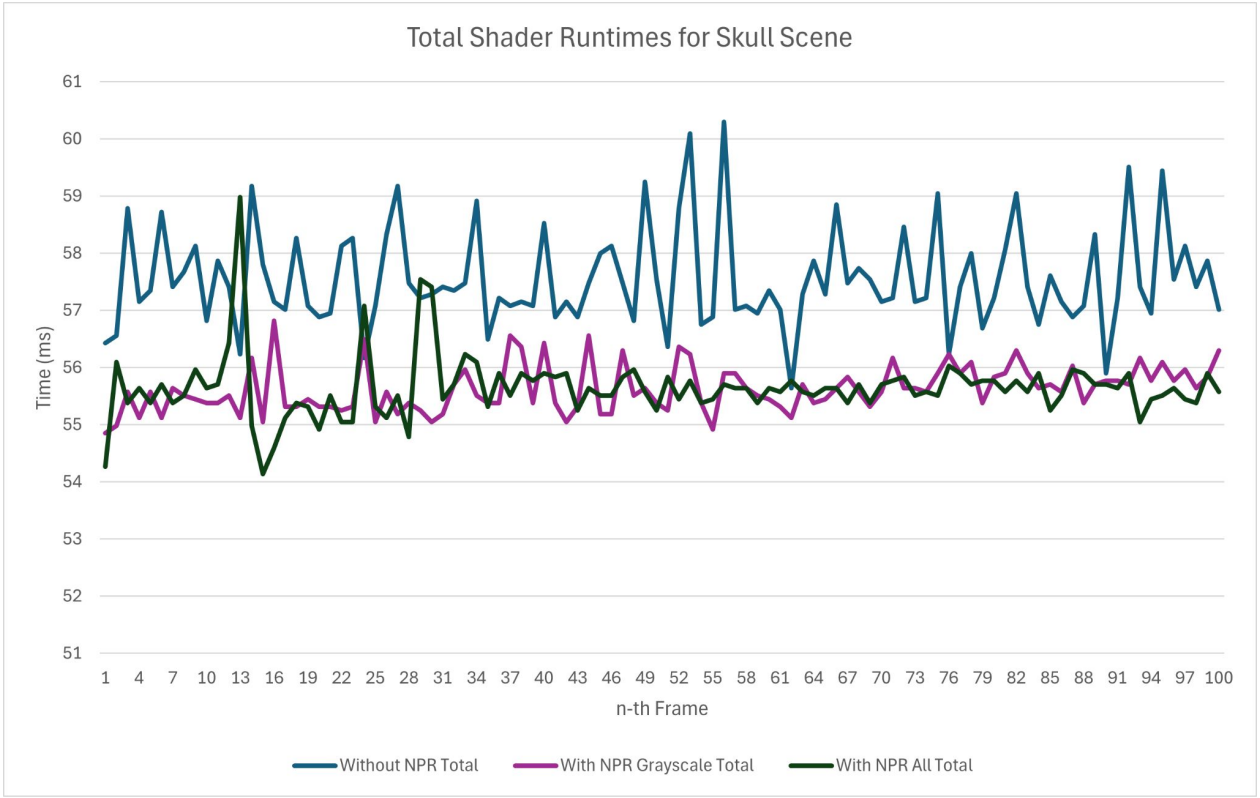
Fig. 6. Evaluating a group-unbiased estimator at a surface point may require more than one integrand sample (e.g. the blue vertex on the bunny requires 3 samples). Starting from the camera and recursively expanding outwards, we observe that satisfying the sample requirements of each next gu-estimator forms a tree (bottom), similar to that of distribution ray tracing [Cook et al. 1984].

From Rex West et al., Stylized Rendering as a Function of Expectation (2024)

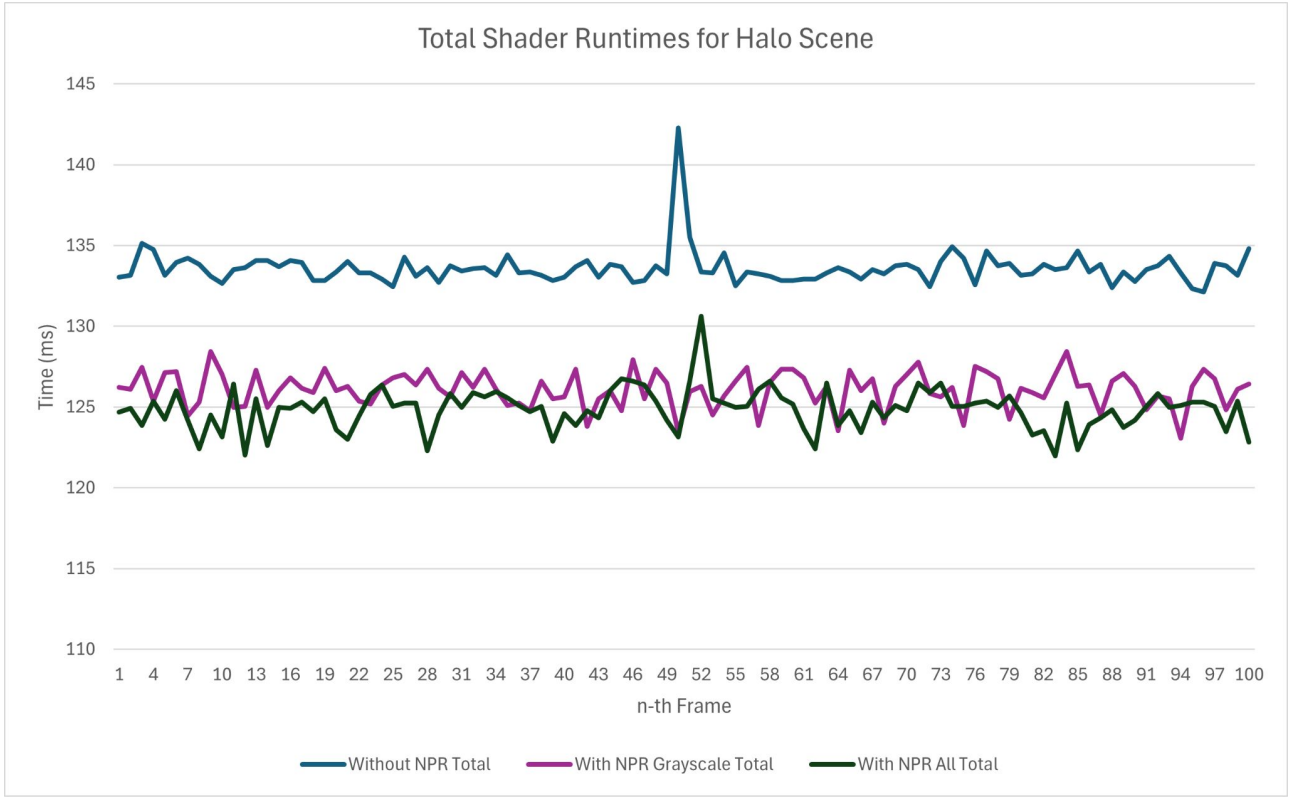
Appendix - Test Scene Setup



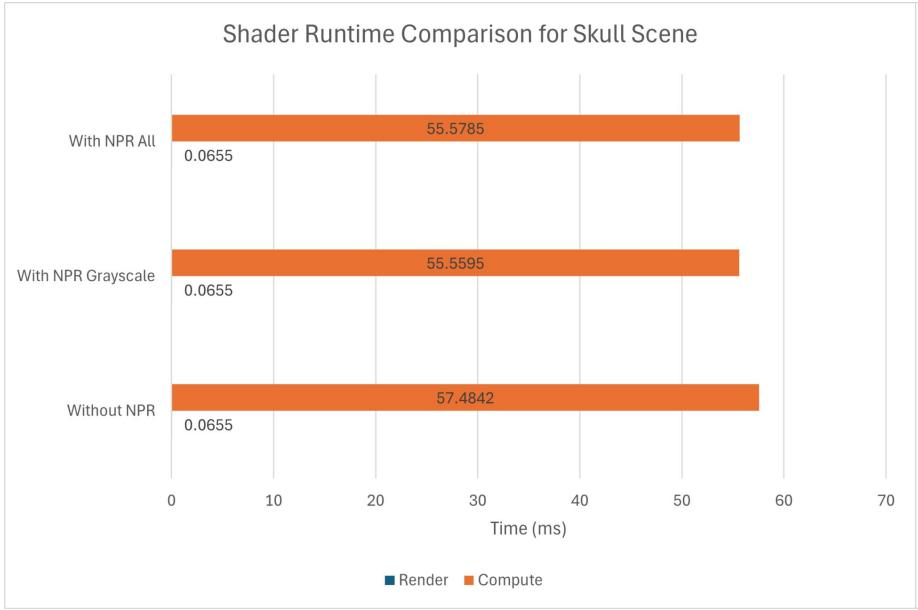
Appendix - Total Shader Runtimes for Skull

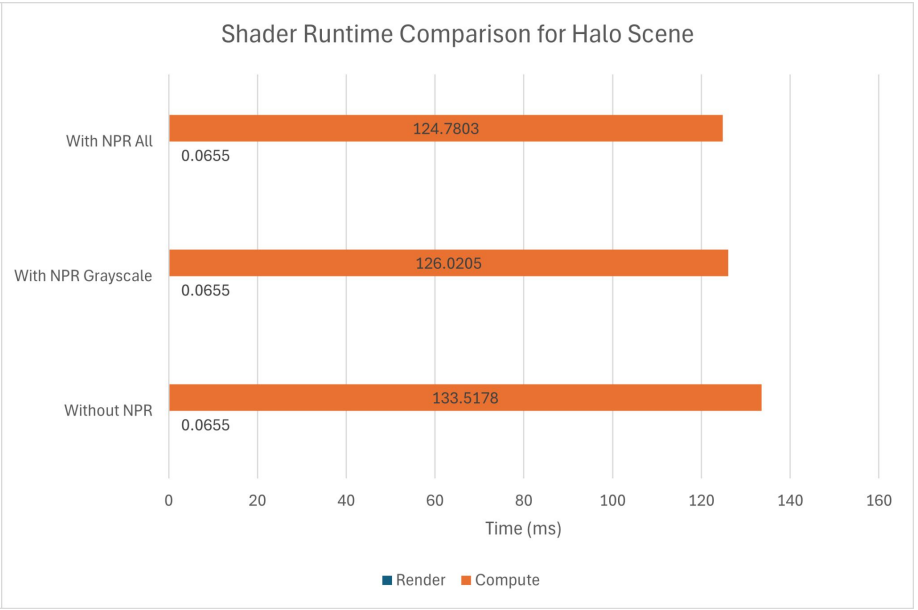


Appendix - Total Shader Runtimes for Halo

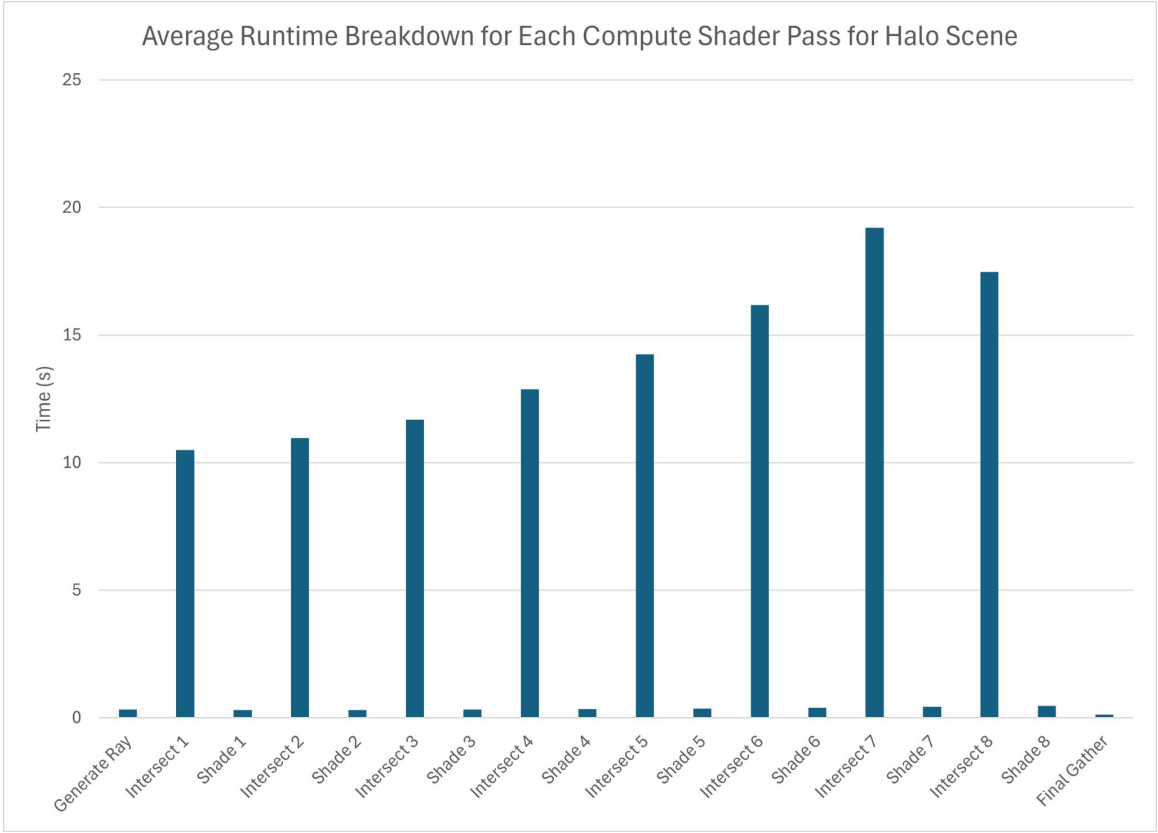


Appendix - Shader Runtime Comparisons

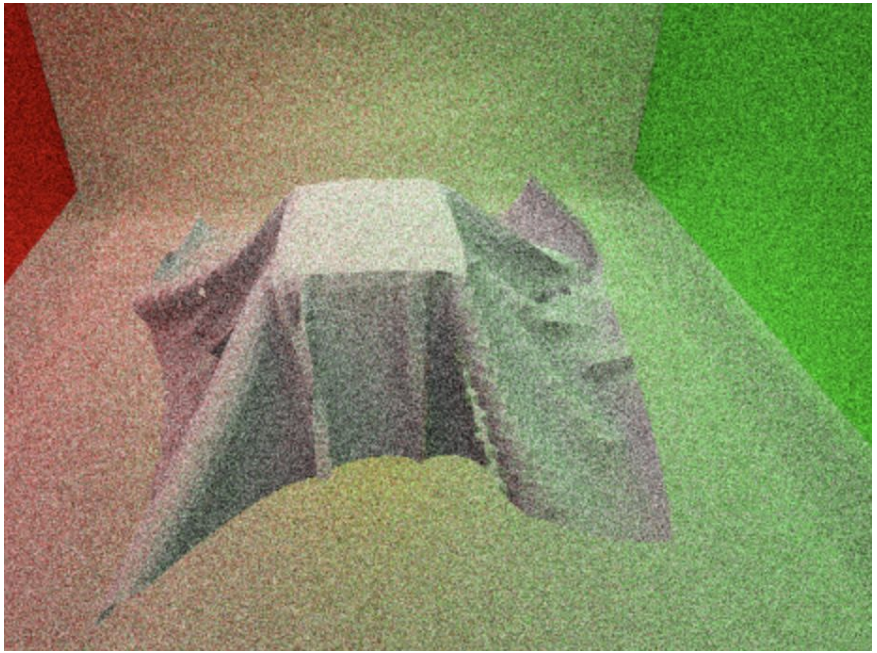




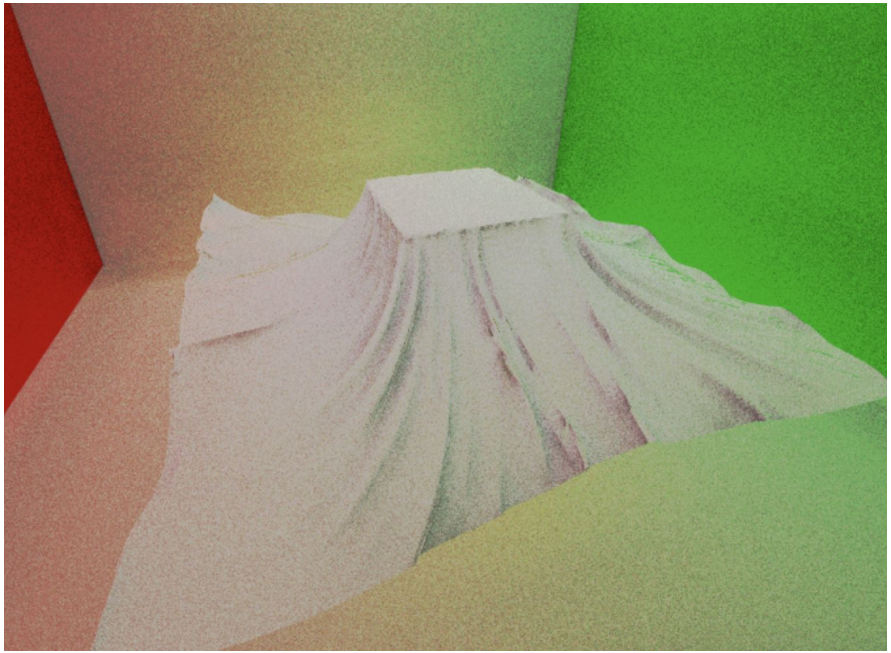
Appendix - Compute Shader Pass Runtime Breakdown



Appendix - Cloth Sim Granularities



50 subdivisions per axis



500 subdivisions per axis