EUROGRAPHICS 2023 / K. Myszkowski and M. Nießner
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 42 (2023), Number 2

Supplemental material for Stochastic Subsets for BVH Construction

L. Tessari[1] and A. Dittebrand[†1,2] M. J. Doyle[1] C. Benthin[1]

[1]Intel Corporation
[2]Karlsruhe Institute of Technology

In the supplemental we provide additional information for the analysis of our method, both on the CPU implementation based on PBRT and our optimized GPU version. Comparison of BVH levels, Variance, Morton window and Uniformity tests have been run on our GPU version, the others on the CPU version. The scenes on the CPU version are different from the ones used on the GPU. We picked a selection from the publicly available pbrt scenes repository (`https://pbrt.org/scenes-v3`)

1. PBRT

All trees are build up to 1 primitive per leaf. The black line in the plots represents the behavior of a standard top-down binned SAH builder.

1.1. Scenes

For our experiments we took publicly available scene from the pbrt repository, selecting low complexity, architectural and complex models. The scenes are the following: Killeroos, Dragon (f8-4a), Crown, Ecosys, Villa, Barcelona-Pavilion.

1.2. Comparison between different measures

We tested different measures for (importance) sampling our distribution, and their impact on quality at different subset sizes (see Figure 1). The random number generator used gives us a stratified sequence for all cases. The first measure is uniform (blue), which makes sure that the selection is stratified spatially, but does not take into account any primitive information besides its centroid position. Area (orange), as its name suggests, also weights the selection towards bigger triangles. Diagonal (blue), instead, prefers primitives that spawn bigger bounding boxes. With the exception of the killeroo scene, uniform perform worst at lower subset sizes, and slowly catches up with increasing sizes as it's more likely to select (bigger) primitives that have a stronger impact on the topology. Area and Diagonal perform similar. We decided to use the diagonal as it better express not only the size of the primitive, but also its orientation: the same triangle, oriented differently, has the same area but it will spawn different AABBs and thus directly impact the SAH heuristic.

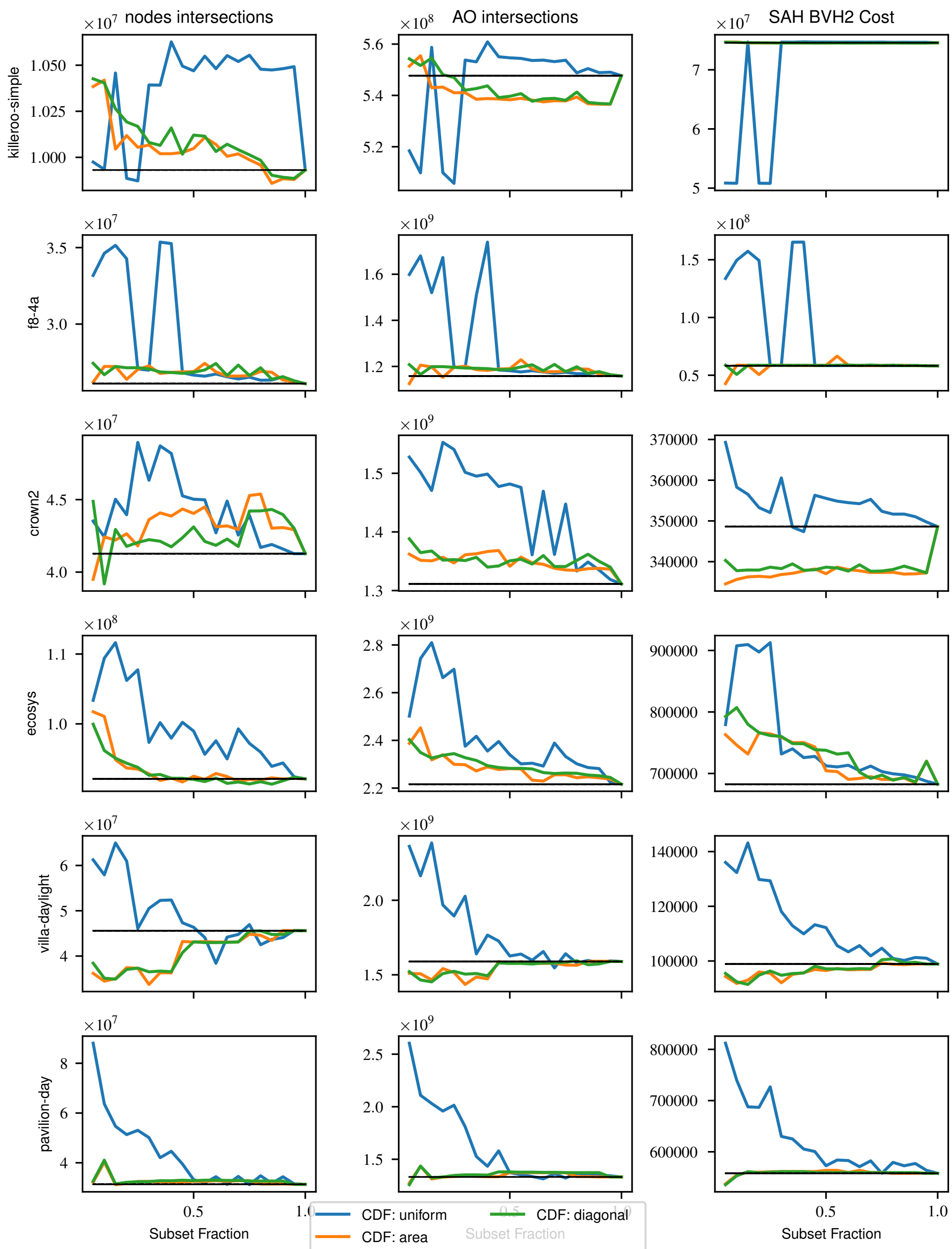

Figure 1: *Influence over the measure at different subset fractions: the lack of importance sampling of the uniform distribution creates huge gaps in bvh quality and thus performance, in particular at lower subset sizes.*

1.3. Clamping impact comparison

For efficiency reasons, we introduced a clamping condition. In this section we will analyze its impact on the overall selection quality under random, equidistant and Sobol RNGs. As a reminder, a progressive RNG like random or Sobol will be able to extract exactly $|M| = m$ primitives for the subset, while equidistant might select the same multiple times and thus have a lower final count. We are using Diagonal as our sampling distribution.

1.3.1. No Clamping

We can see that importance sampling with a stratified sequence, like equidistant and Sobol, can have a huge impact on the final quality of the selection (Figure **??**). A simple uniform selection misses the variance reduction given by the other sequences, thus clustering primitives closer together instead of spreading equally into the whole space.

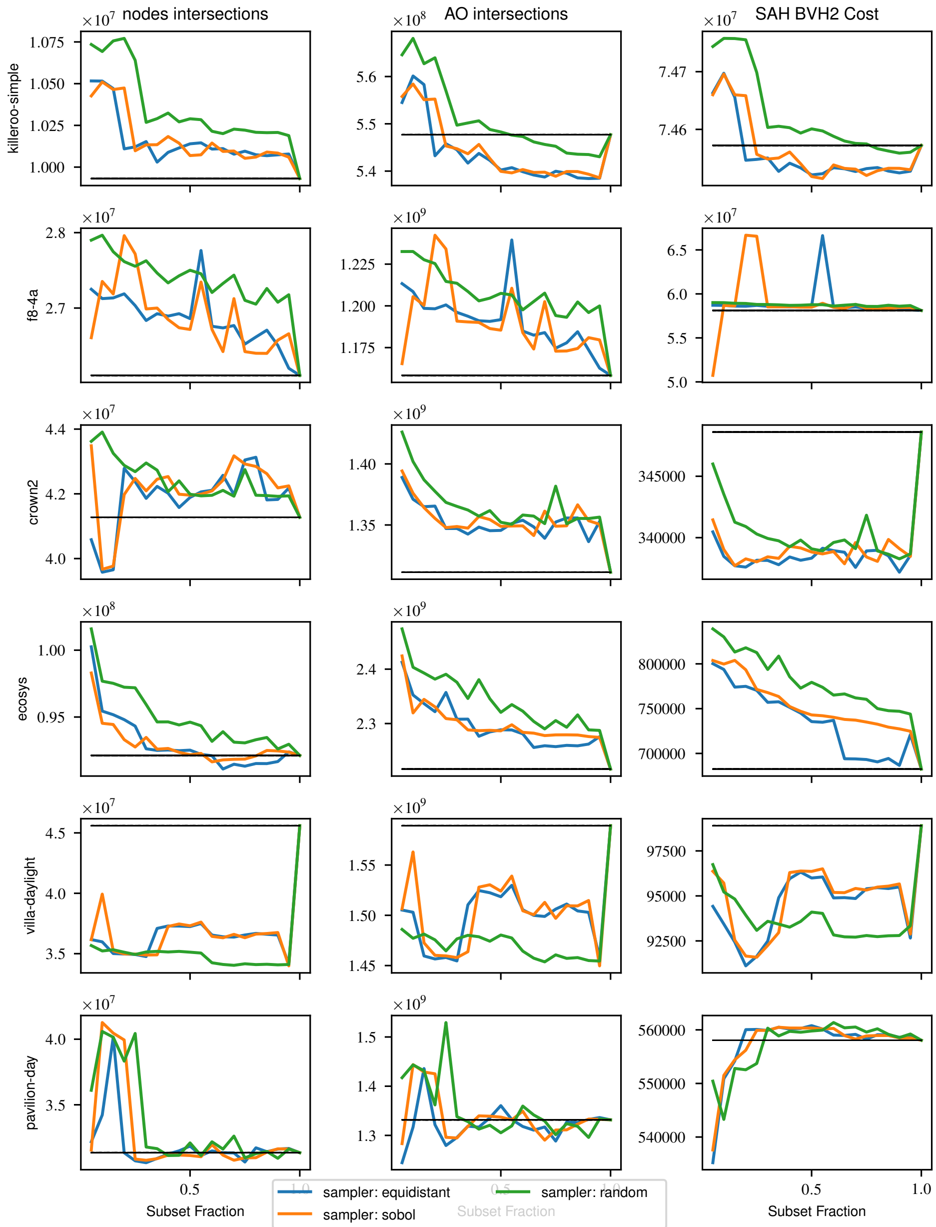

Figure 2: *Sampling without clamping: as expected sequences that stratify perform best.*

1.3.2. With Clamping

Our clamping lowers the probability of selecting the same primitive multiple times, assuming stratified random sequences with a maximum weight of exactly $\frac{|M|}{|N|}$. In practice, we obtain excellent conservative bounds slightly bigger than the theoretical optimum. Given these premises, it is not surprising that the uniform random sequence, which does not possess any stratification property, performs notably worse (see Figure **??**). Compare this to the previous experiment, where the random numbers possessed stratification properties against a non-optimal uniform distribution for the selection.

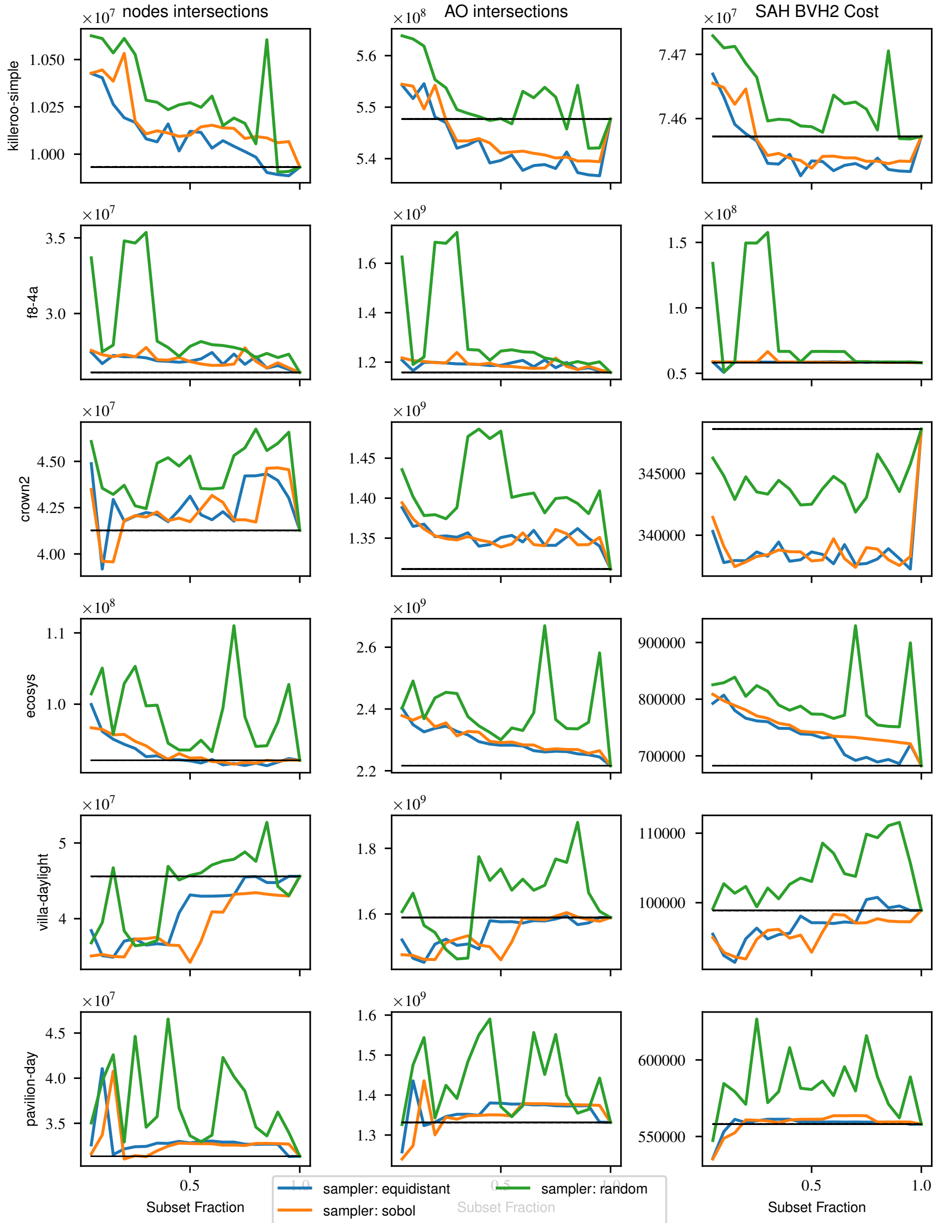

Figure 3: *Sampling with clamping: interestingly random has the worst performance, even compared to the version without clamping. This is to be expected, since now it cannot rely on selecting big triangles with large weights as before and stratification is thus required.*

1.3.3. With and without Clamping

In Figure **??** we can see the full picture and relative scaling when clamping is applied or not. Uniform random sequences perform worst, especially with clamping, as it cannot rely on massive weights to select the most important primitives with higher chance. Clamping does not influence the other sequences in any meaningful way, which proves the viability of this algorithmic optimization.

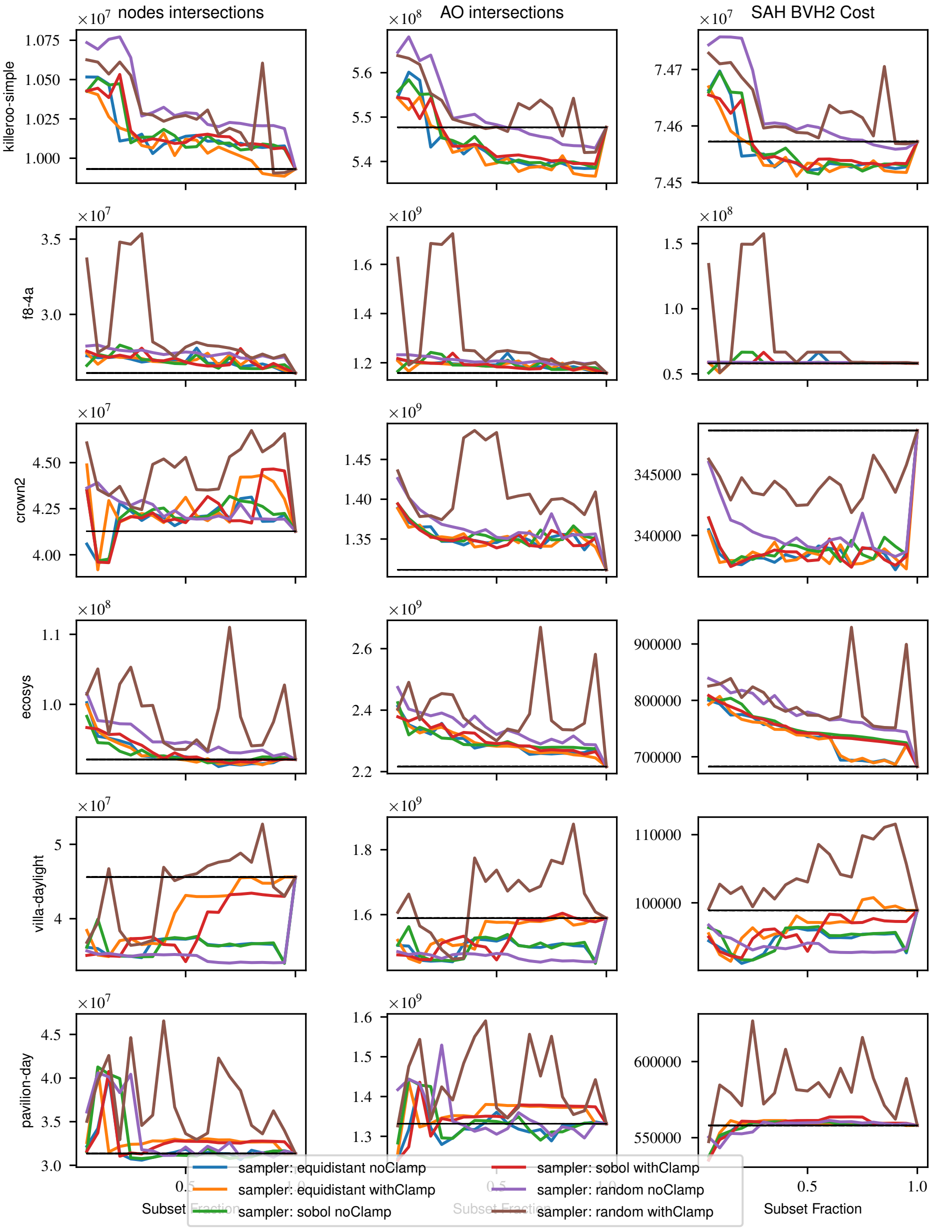

Figure 4: *Sampling with and without clamping, full comparison of the previous two plots.*

2. GPU implementation

Our optimized GPU implementation, based on oneAPI DPC++, is tested on an Intel ARC A770 using HW traversal. The testing system consists of an Intel Core i5 9600K CPU clocked at 3.70GHz with 16GB of DDR4 RAM running Ubuntu 20.04 LTS Linux OS on an NVMe SSD. We render all images at a resolution of 1024×1024 using primary rays at 1spp and ambient occlusion with 64 indirect rays.

2.1. Scenes

For our experiments we took publicly available scenes and converted in our format, selecting low complexity, architectural and complex models. The scenes are the following: Crytek Sponza, Bistro, Crown, Hairball, Rungholt, San Miguel.

2.2. Sampling variance

To assess the stability of our stochastic approach, we run the builder at increasing subset sizes. For each of these steps, we used different seeds and evaluated its variance in Figure **??** and Figure **??**. Interestingly increased subset sizes do not always correlate with a better SAH cost.

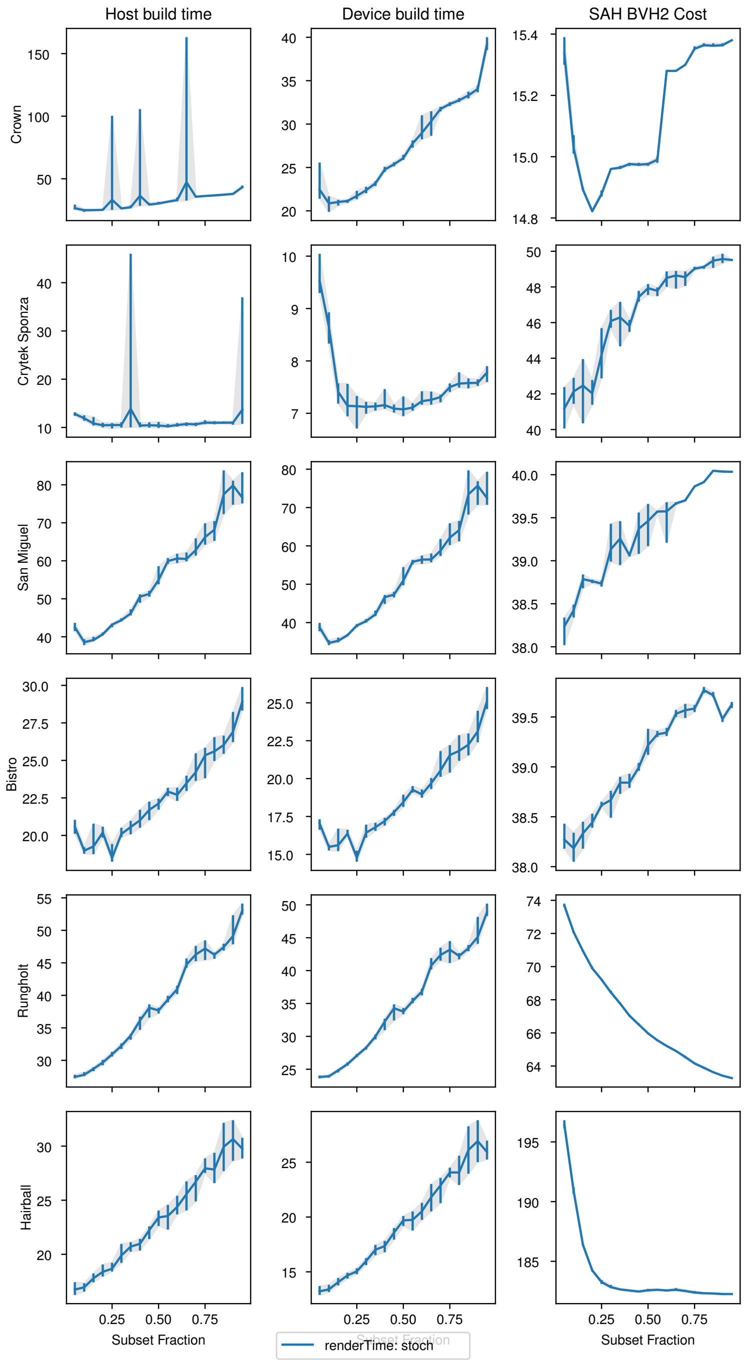

Figure 5: *Sampling variance on the construction times.*

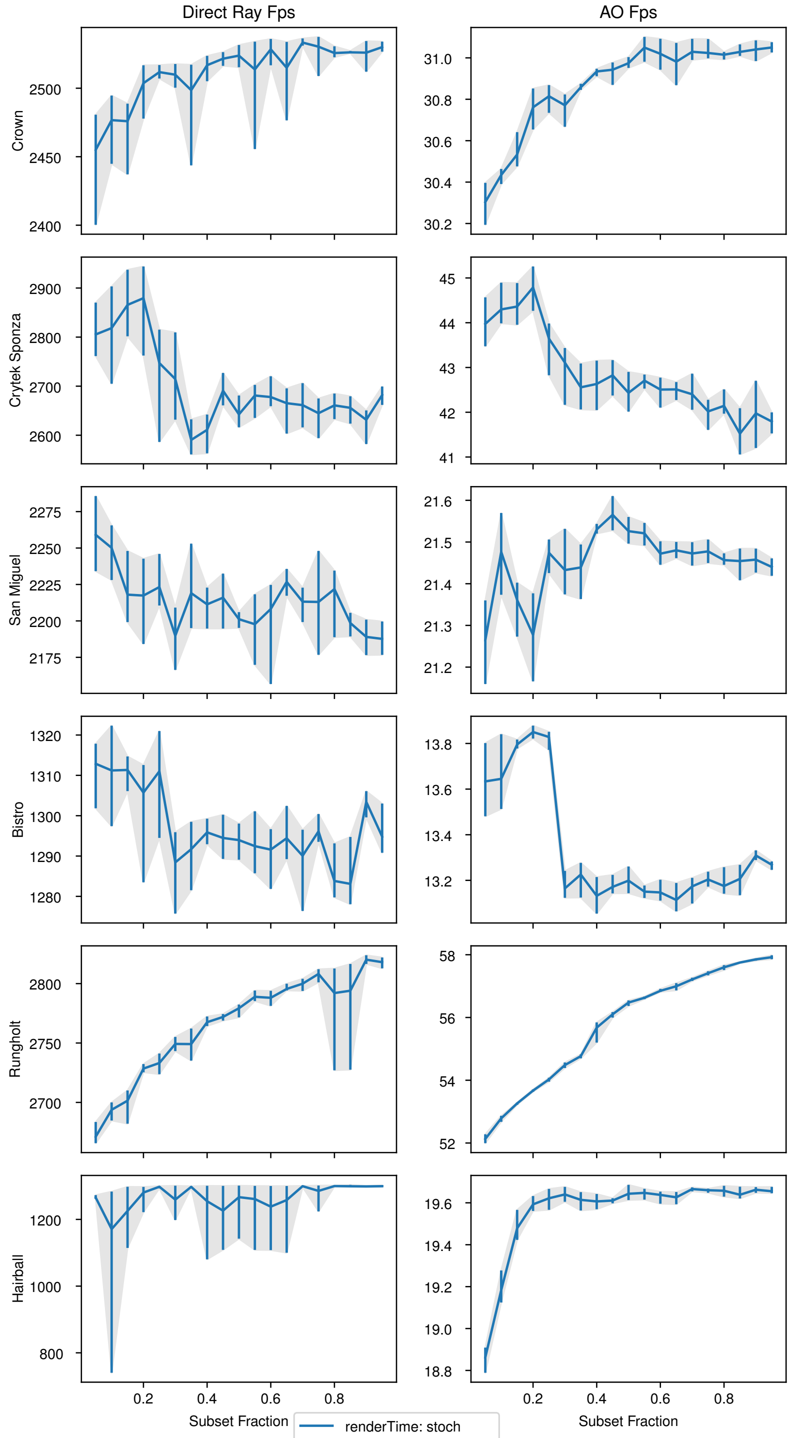

Figure 6: *Sampling variance on performance.*

2.3. Clustering

In the same spirit as HLBVH, we used a fixed amount of bits per axis to cluster primitives given their Morton code, and then build top and bottom with a binned BVH. The behaviour is quite inconsistent due to its inability to dynamically adapt to the geometry distribution or its increased computational cost when only a few clusters are created (Figure **??**). Another aspect to consider is the amount of clusters that can be used, significantly less than either the stochastic or the binned BVH constructor.

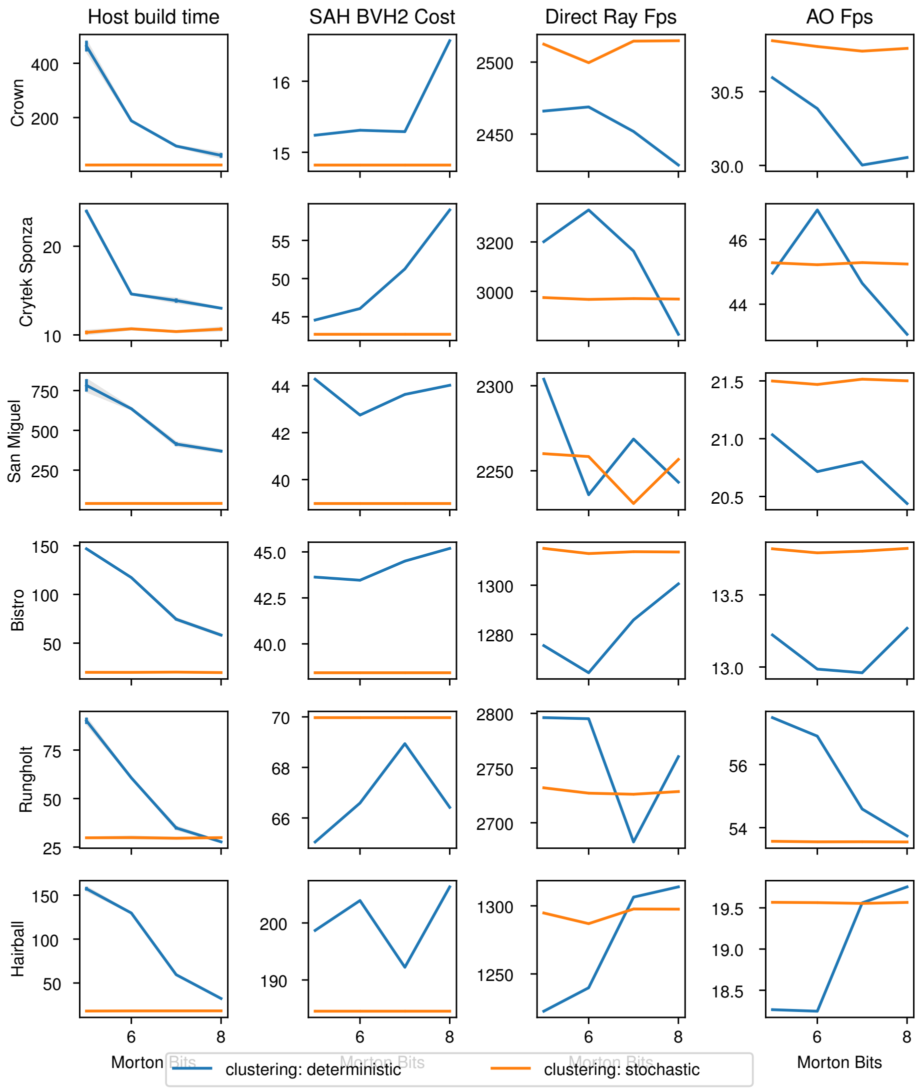

Figure 7: *Impact of different amount of morton bits when deterministically clustering the geometry.*

2.4. Morton window

We have tested how the size of the search window impacts both build cost and performance. Usually increased build cost do not correlate with higher fps (Figure **??**).

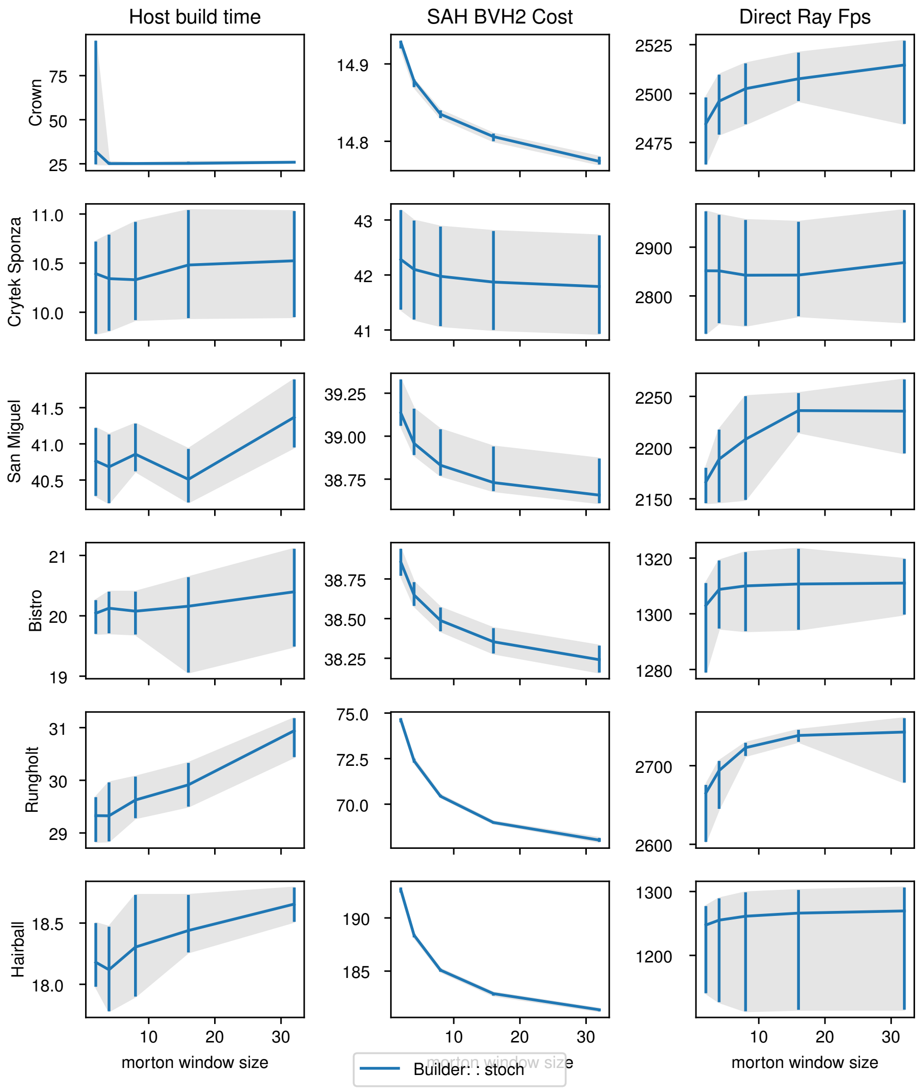

Figure 8: *Impact of the morton window size: the increased search radius is usually not worth the added cost in build time.*

2.5. Uniformity

Increasing uniformity in the clamping increases SAH cost and helps distributing work, as clusters have a more even number of primitives each (Figure **??**)..

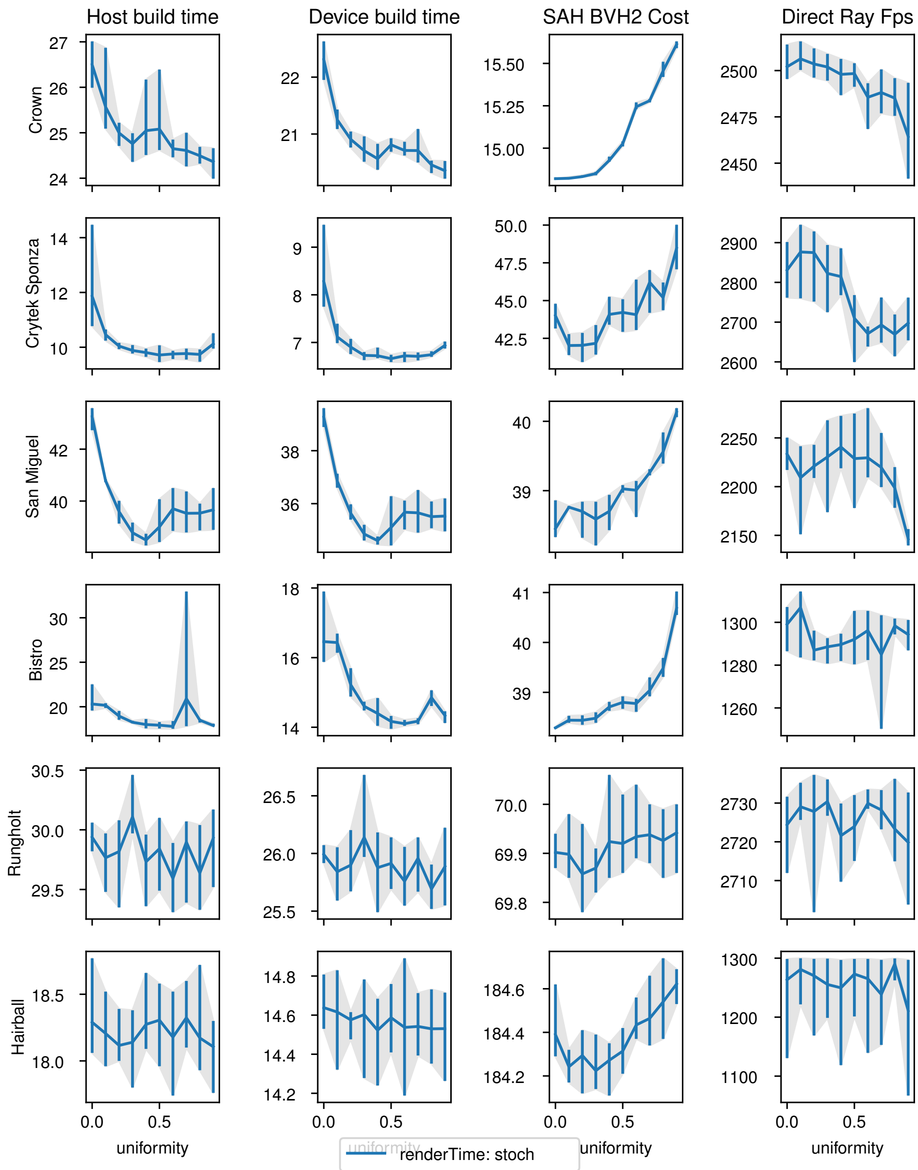

Figure 9: *Impact of adding different amount of uniformity: except for rungholt, the SAH quality decreases, in particular with fractions > 0.5.*