```
Activating project at `D:\Code\Projects\AccelerateRT`
```

```
ingredients (generic function with 1 method)
```

```
ACC = Main.AccelerateRT.jl.AccelerateRT
```

```
BVH = Main.AccelerateRT.jl.AccelerateRT.BVH
```

```julia
using LinearAlgebra
```

```julia
using ProgressLogging
```

```julia
using Plots
```

```julia
using Logging
```

```julia
using Random
```

```julia
using Statistics
```

```julia
using Printf
```

```
"Number of threads: 6"
```

Structures and Constants

```julia
mutable struct Camera
    pos::ACC.Vector3{Float32}       # position
    center::ACC.Vector3{Float32}    # center to look at
    fov::Float32                    # field of view in degrees
    res::ACC.Vector2{UInt32}        # rendered frame resolution
end
```

```julia
mutable struct Ray
    origin::ACC.Vector3{Float32}    # original position
    dir::ACC.Vector3{Float32}       # direction
    dist::Float32                   # distance to nearest hit
end
```

```
models =  ["teapot", "bunny", "dragon", "sponza"]
```

```
bvhTypes =  ["middle", "median", "sah", "sahm"]
```

Utility Functions

```
intersect (generic function with 1 method)
```

```
intersect! (generic function with 1 method)
```

```
loadData (generic function with 1 method)
```

```
rayTrace (generic function with 1 method)
```

```
visualize (generic function with 2 methods)
```

```
visualizeAll (generic function with 1 method)
```

```
sample1 (generic function with 3 methods)
```

```
sample2 (generic function with 3 methods)
```

```
sample3 (generic function with 3 methods)
```

```
sample4 (generic function with 3 methods)
```

```
visualizeSphere (generic function with 1 method)
```

```
collectInfo (generic function with 1 method)
```

```
loadData (generic function with 2 methods)
```

```
visualizeTrend (generic function with 1 method)
```

```
visualizeComparison (generic function with 1 method)
```

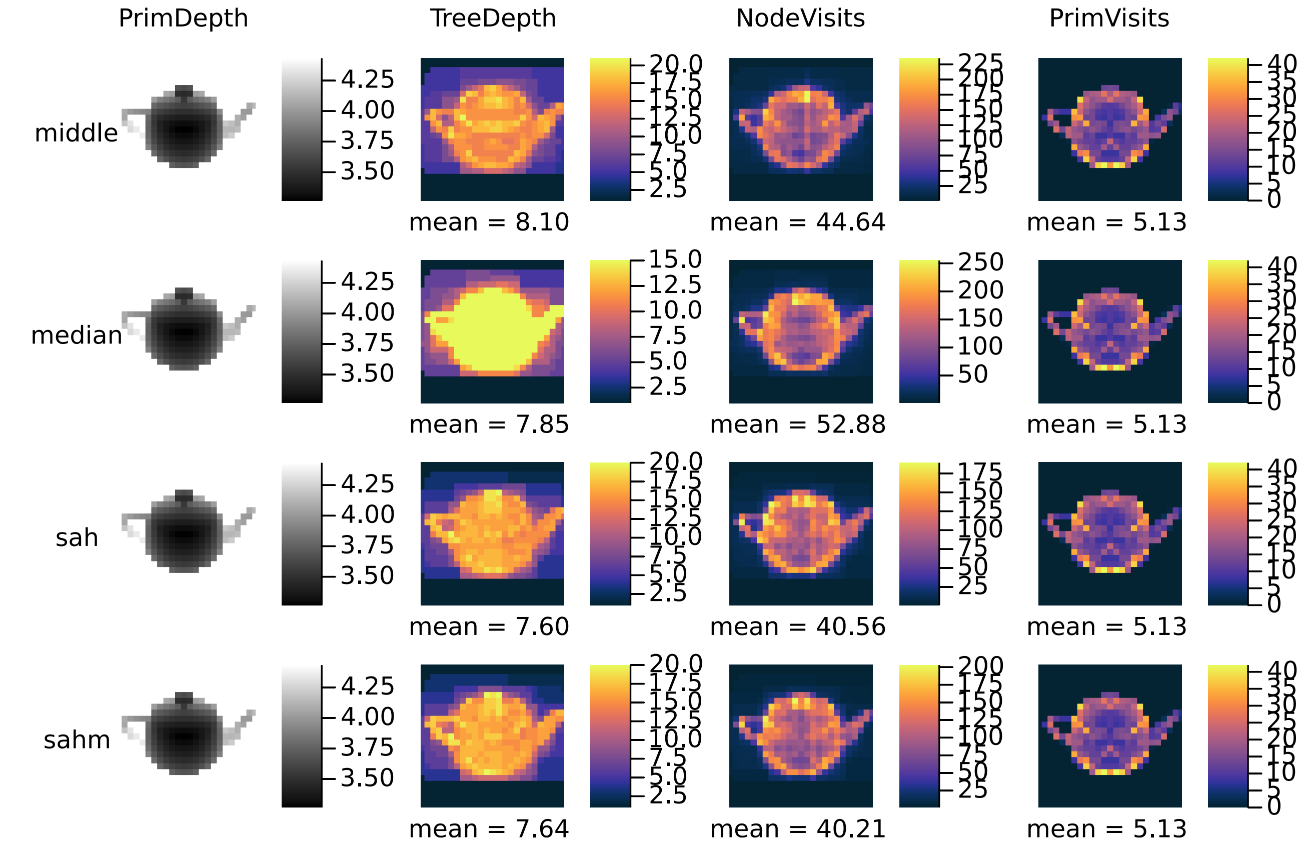
Comparison

Compare different BVH's on teapot model

```
camera =  Camera(
              pos =  StaticArrays.MVector{3, Float32}: [0.0, 3.0, 3.0]
              center =  StaticArrays.MVector{3, Float32}: [0.0, 0.0, 0.0]
              fov = 45.0f0
              res =  StaticArrays.MVector{2, UInt32}: [0x00000019, 0x00000019]
          )
```

```
camera = Camera(
    ACC.Vector3{Float32}(0, 3, 3),
    ACC.Vector3{Float32}(0, 0, 0),
    45.0f0,
    ACC.Vector2{UInt32}(25, 25)
)
```

```
teapotvis =
```



1 1 1 100%

Sampling

Test unit sphere sampling for experiment

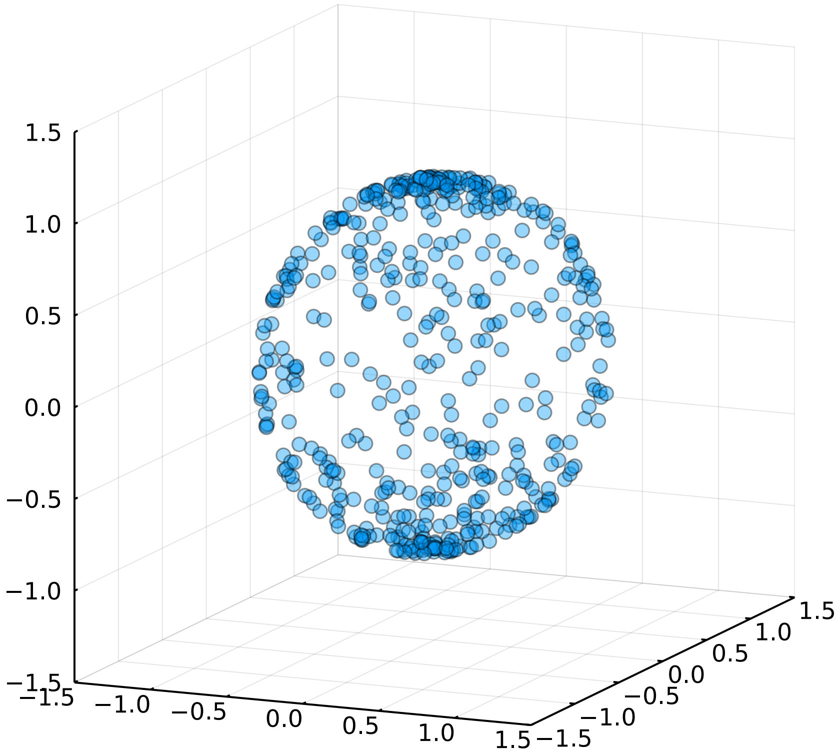
`https://mathworld.wolfram.com/SpherePointPicking.html` for reference

```
randseed = 123
```

`sample1` implements the following algorithm:

$$
\begin{aligned}
\theta &= \mathrm{Uniform}(0, 2\pi), \\
\phi &= \mathrm{Uniform}(0, \pi), \\
x &= \sin\phi \cdot \cos\theta, \\
y &= \sin\phi \cdot \sin\theta, \\
z &= \cos\phi.
\end{aligned}
$$

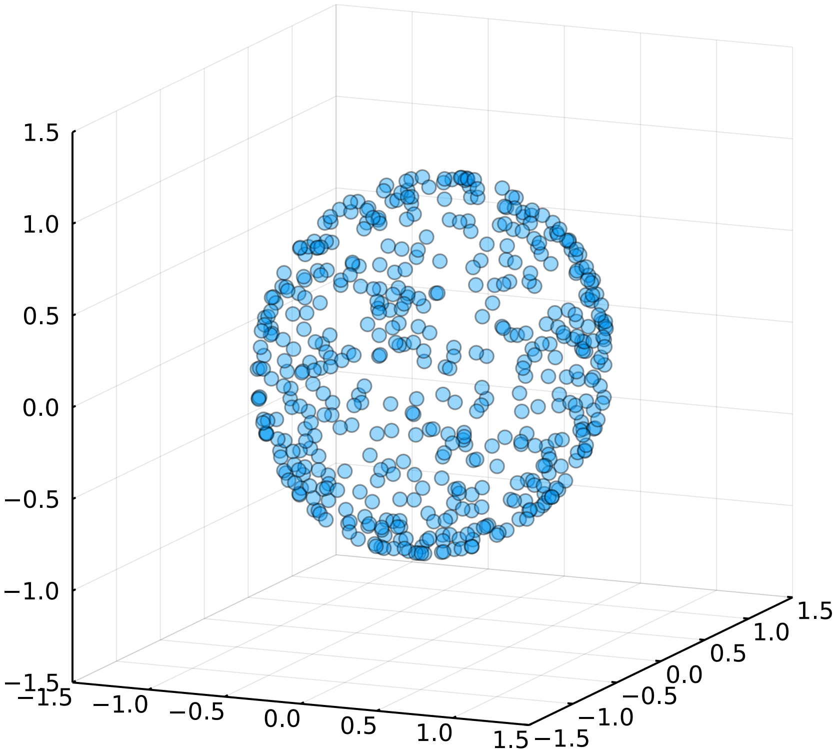
```
sample1vis =
```



`sample2` implements the following algorithm by Marsaglia (1972):

$$\begin{aligned} x_1, x_2 &= \text{Uniform}(-1, 1), \\ \text{Reject If} \quad & x_1^2 + x_2^2 \geq 1, \\ x &= 2x_1\sqrt{1 - x_1^2 - x_2^2}, \\ y &= 2x_2\sqrt{1 - x_1^2 - x_2^2}, \\ z &= 1 - 2(x_1^2 + x_2^2). \end{aligned}$$

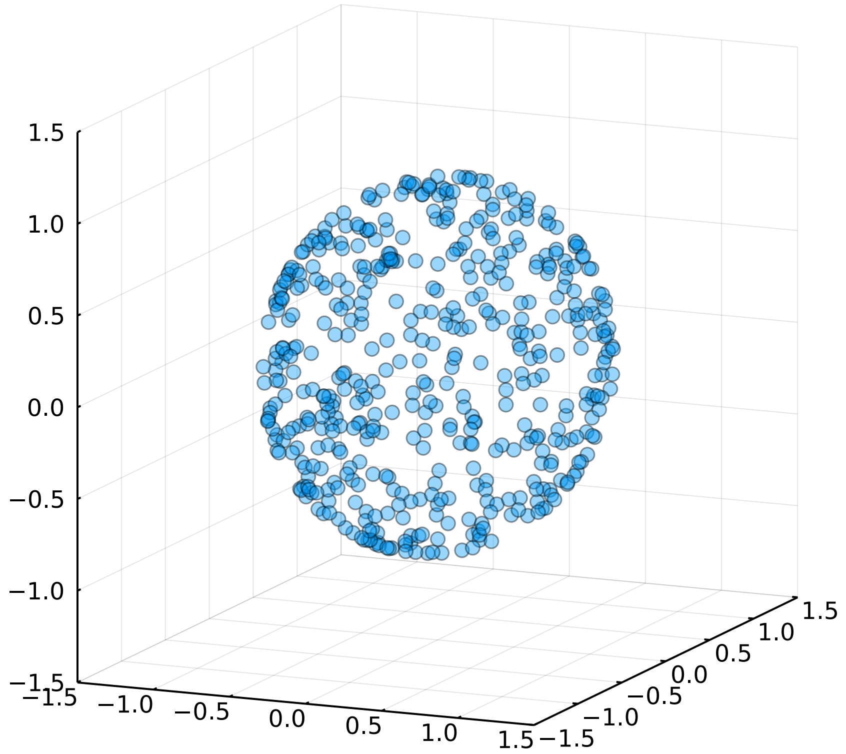
```
sample2vis =
```



`sample3` implements algorithm by Cook (1957):

$$
\begin{aligned}
x_0, x_1, x_2, x_3 &= \text{Uniform}(-1, 1), \\
\text{Reject If} \quad & x_0^2 + x_1^2 + x_2^2 + x_3^2 \geq 1, \\
x &= \frac{2(x_1 x_3 + x_0 x_2)}{x_0^2 + x_1^2 + x_2^2 + x_3^2}, \\
y &= \frac{2(x_2 x_3 - x_0 x_1)}{x_0^2 + x_1^2 + x_2^2 + x_3^2}, \\
z &= \frac{x_0^2 + x_3^2 - x_1^2 - x_2^2}{x_0^2 + x_1^2 + x_2^2 + x_3^2}.
\end{aligned}
$$

```
sample3vis =
```

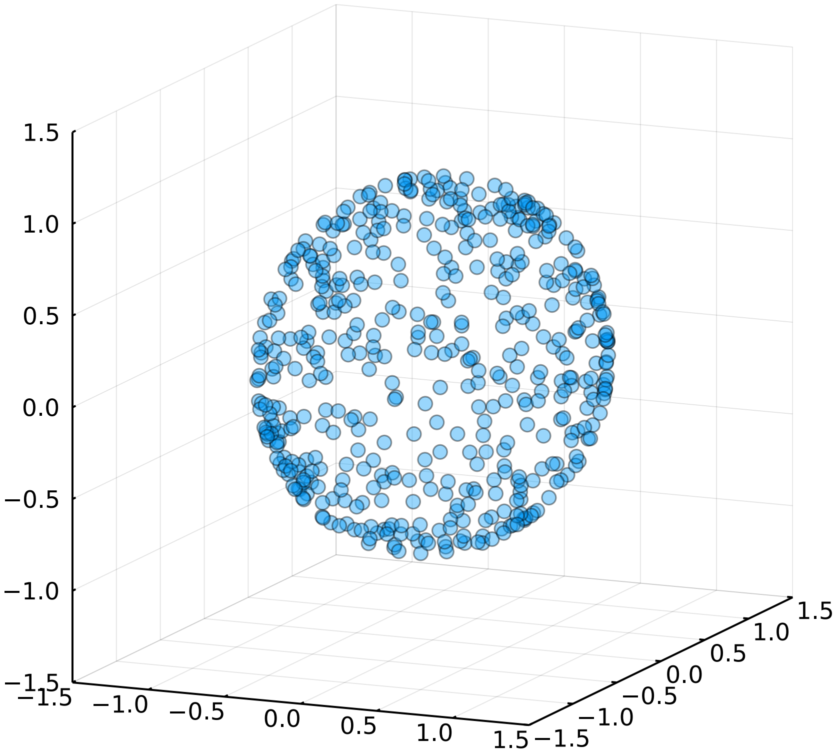


`sample4` implements a simple algorithm with Gaussian distribution:

$$x, y, z = \text{Normal}(),$$

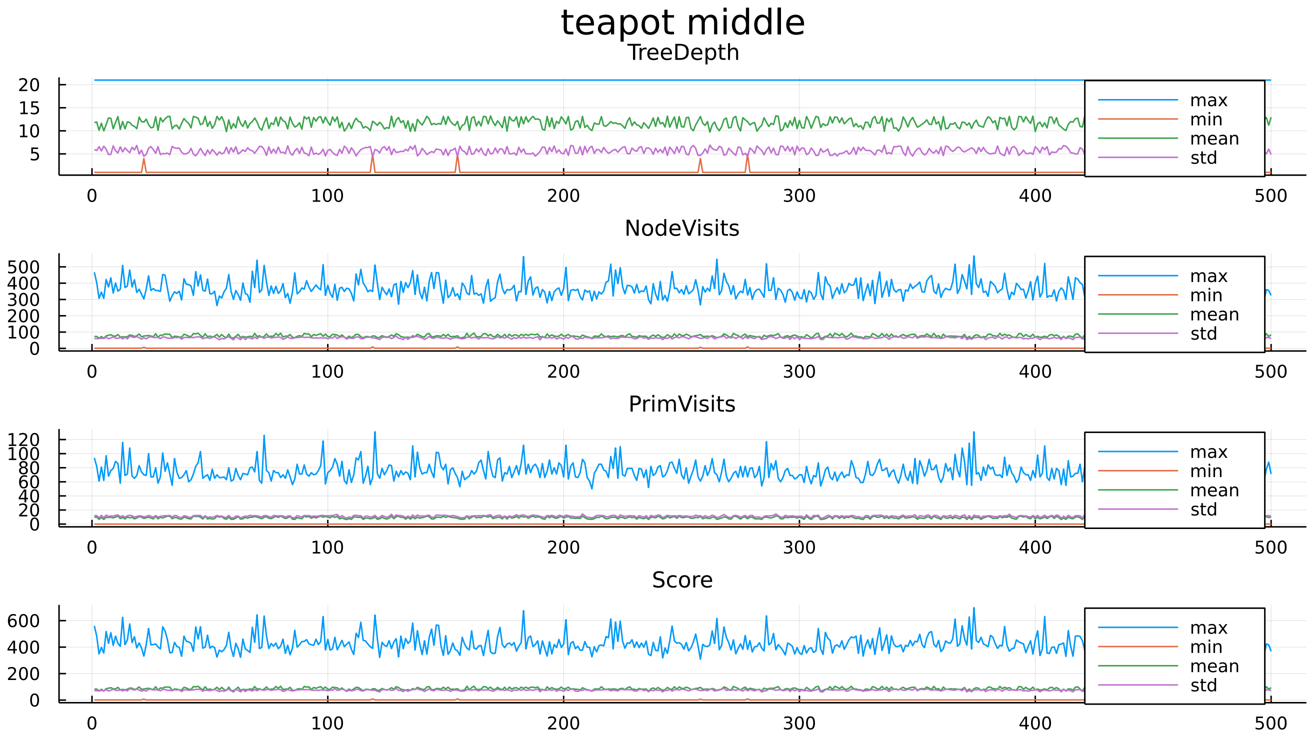
$$\vec{v} = \frac{1}{\sqrt{x^2 + y^2 + z^2}} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

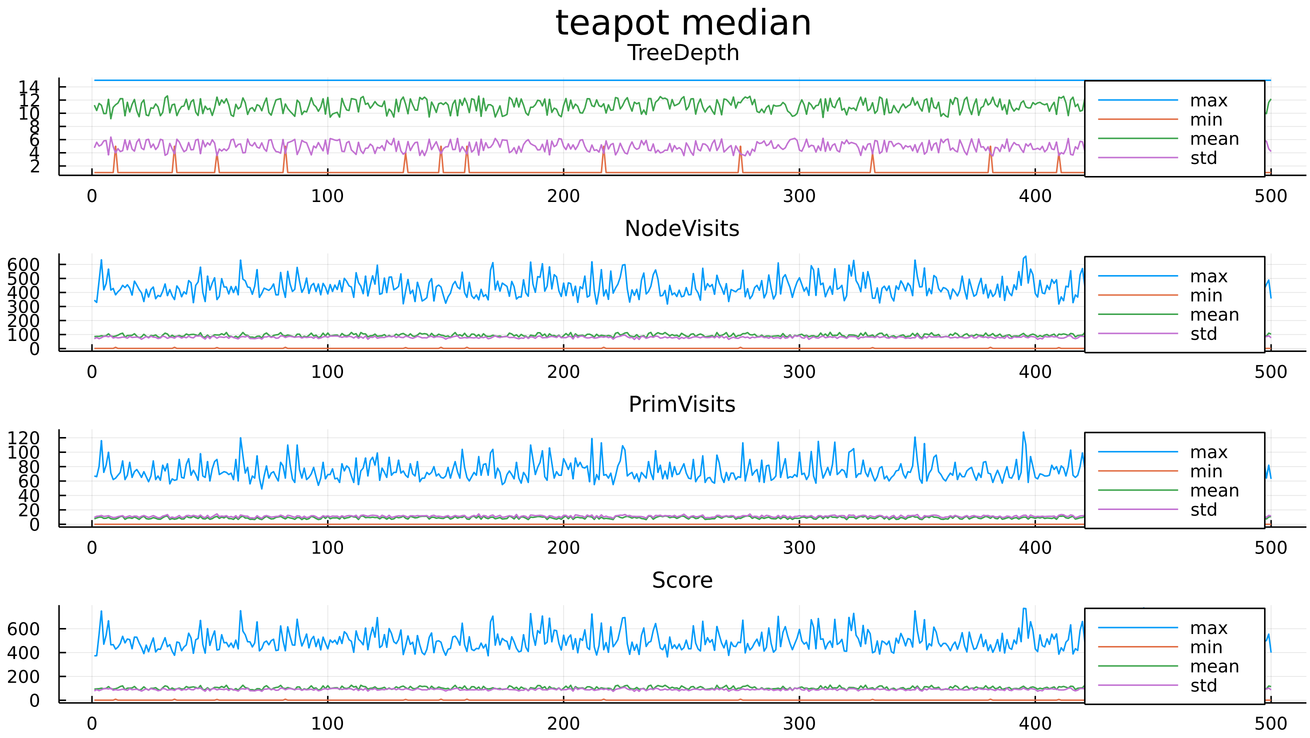
```
sample4vis =
```

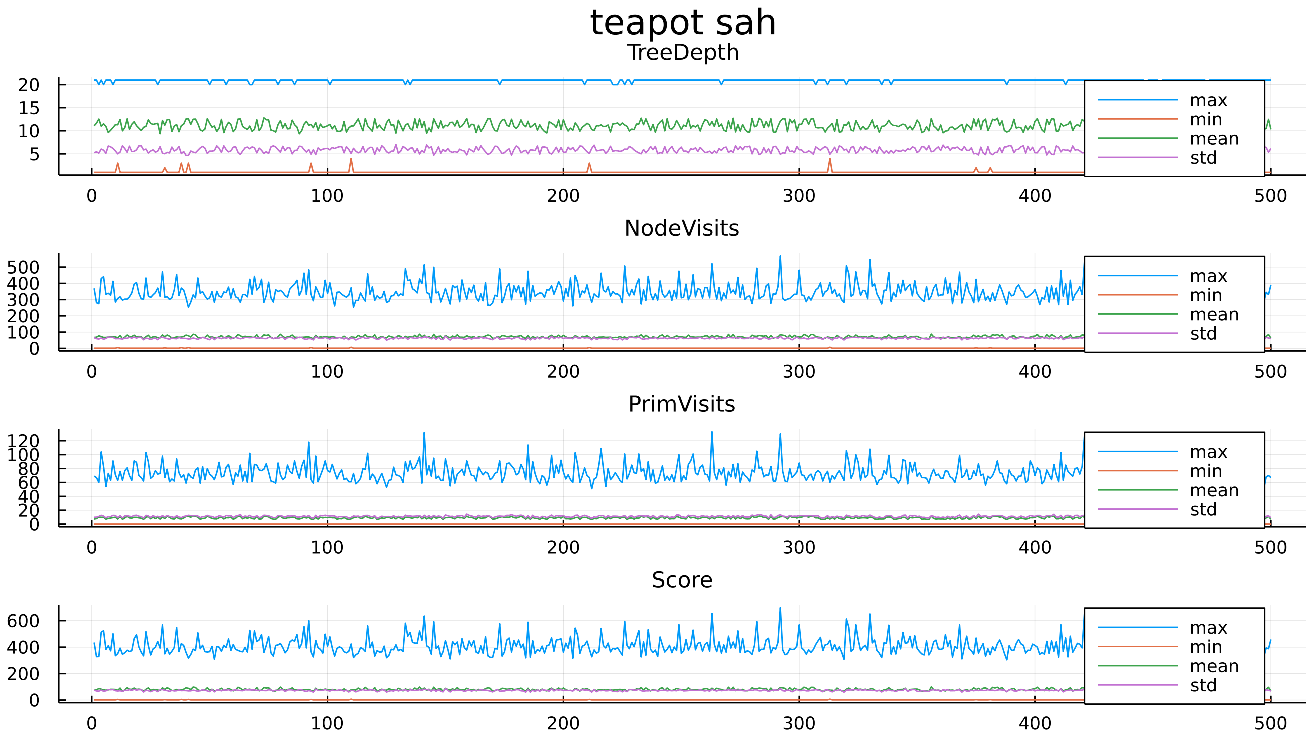


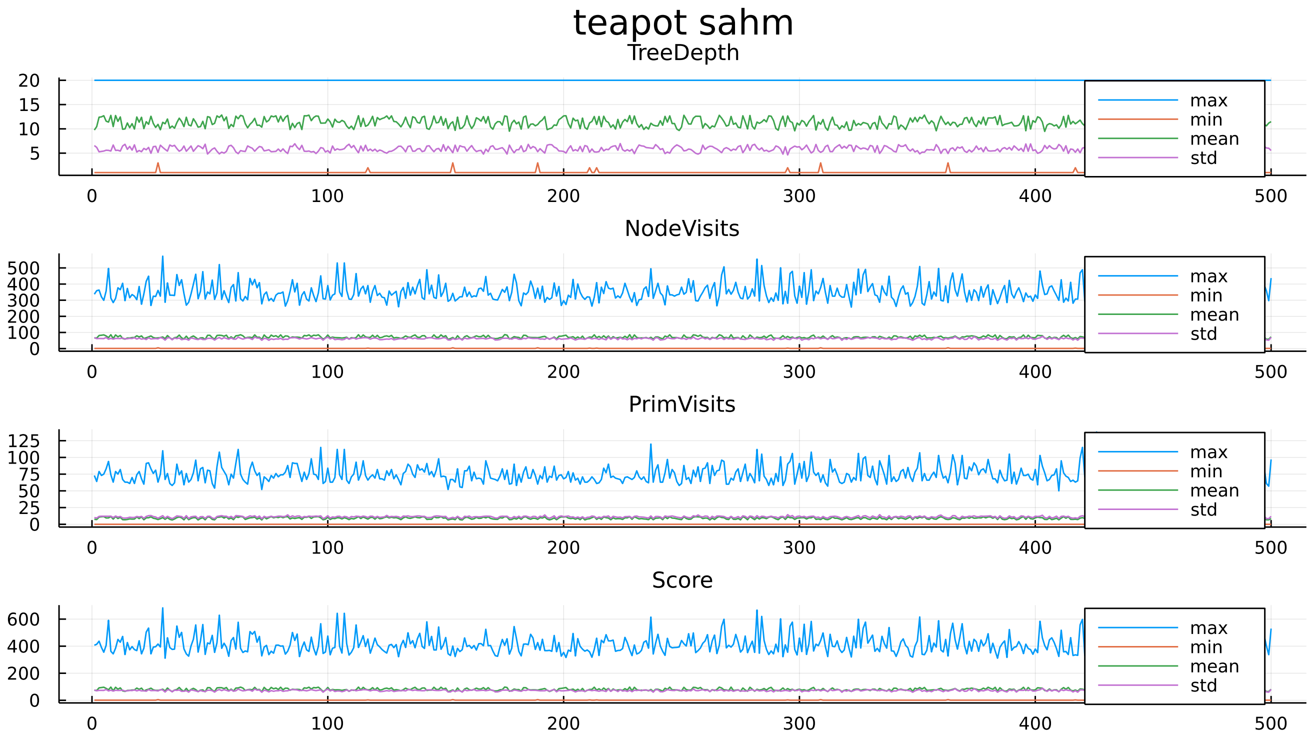
Performance Metric

Visualization of 500 samples



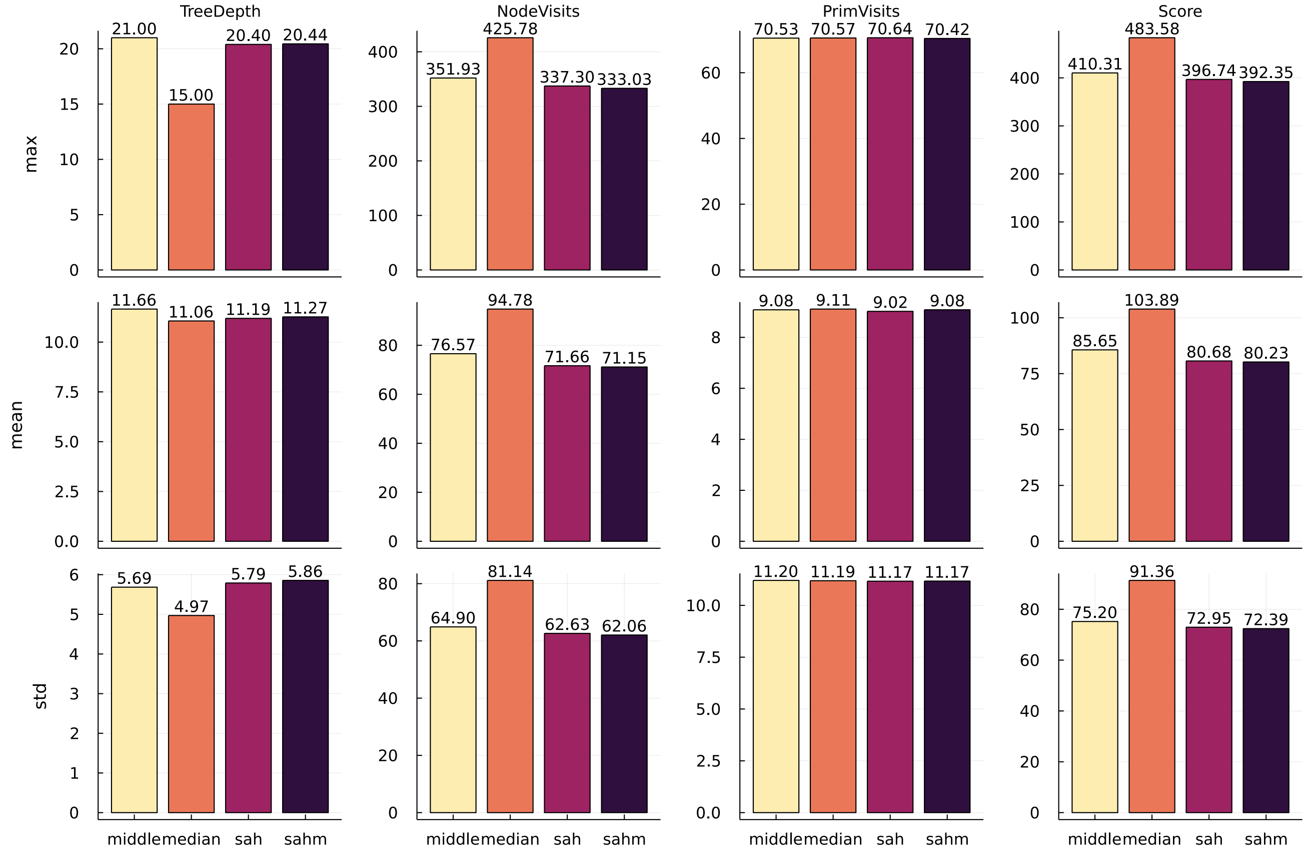

teapot median
TreeDepth
14
12
10
8
6
4
2
0
100
200
300
400
500
max
min
mean
std
NodeVisits
600
500
400
300
200
100
0
0
100
200
300
400
500
max
min
mean
std
PrimVisits
120
100
80
60
40
20
0
0
100
200
300
400
500
max
min
mean
std
Score
600
400
200
0
0
100
200
300
400
500
max
min
mean
std


teapot sah
TreeDepth
20
15
10
5
0
100
200
300
400
500
max
min
mean
std
NodeVisits
500
400
300
200
100
0
0
100
200
300
400
500
max
min
mean
std
PrimVisits
120
100
80
60
40
20
0
0
100
200
300
400
500
max
min
mean
std
Score
600
400
200
0
0
100
200
300
400
500
max
min
mean
std

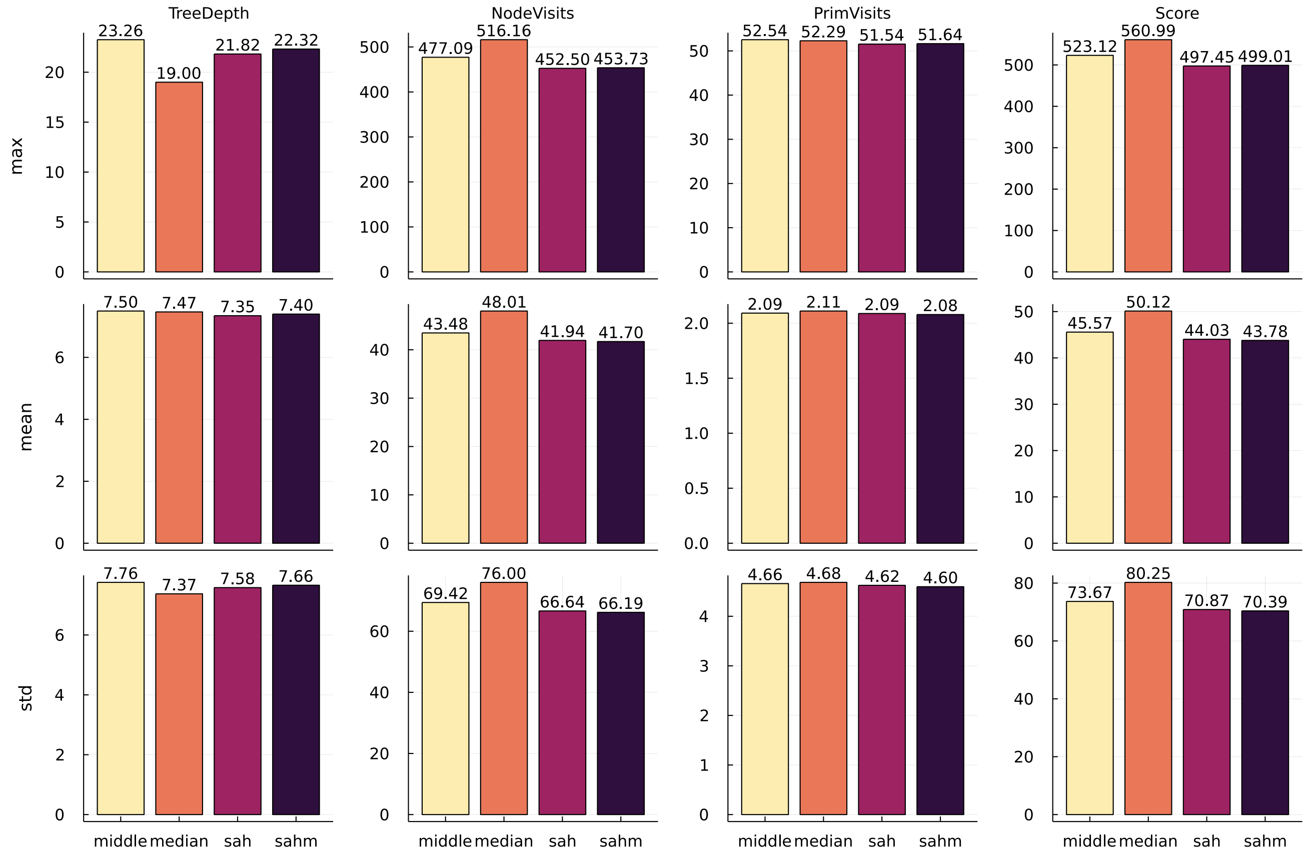


Visualization of 1000 samples with resolution 50x50

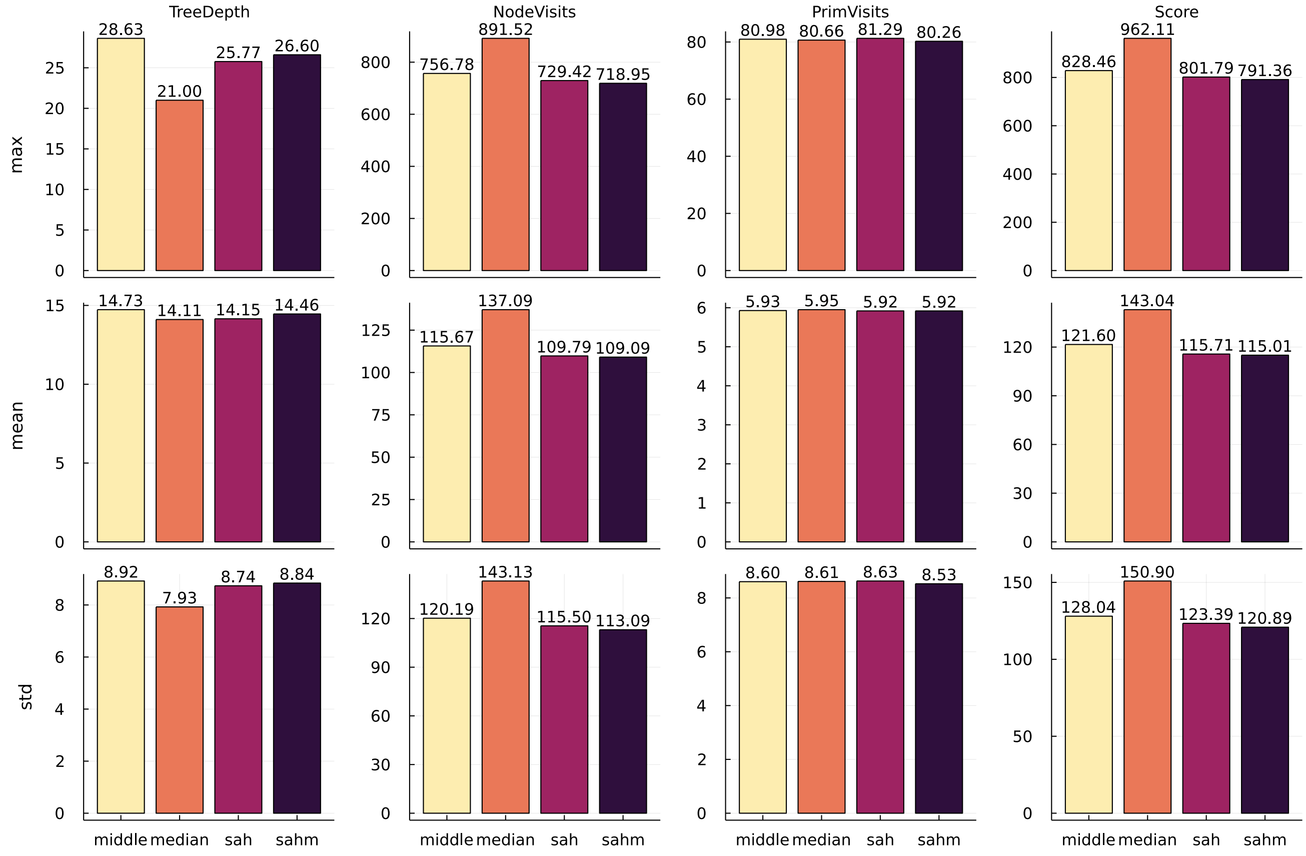
```
compareVisTeapot =
```



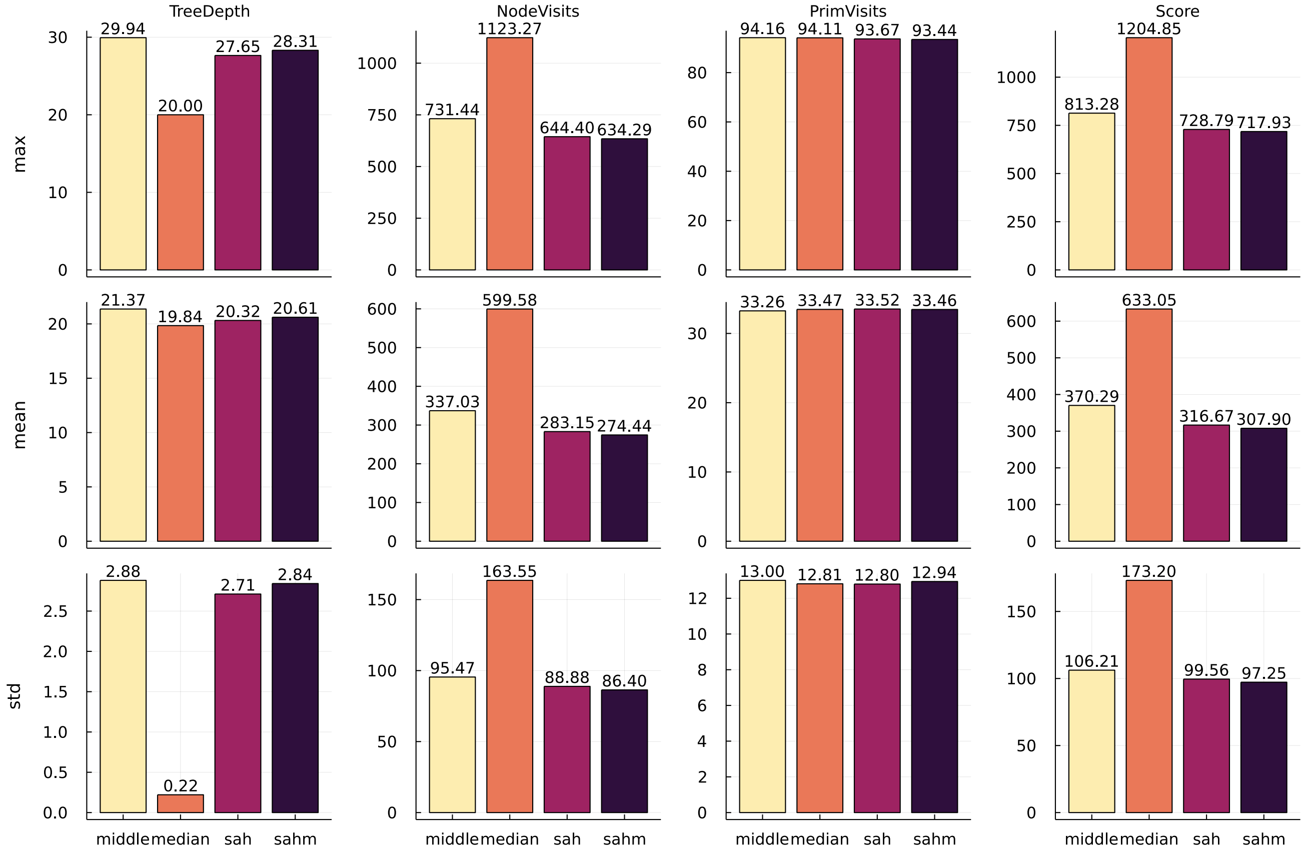
```
compareVisBunny =
```



```
compareVisDragon =
```



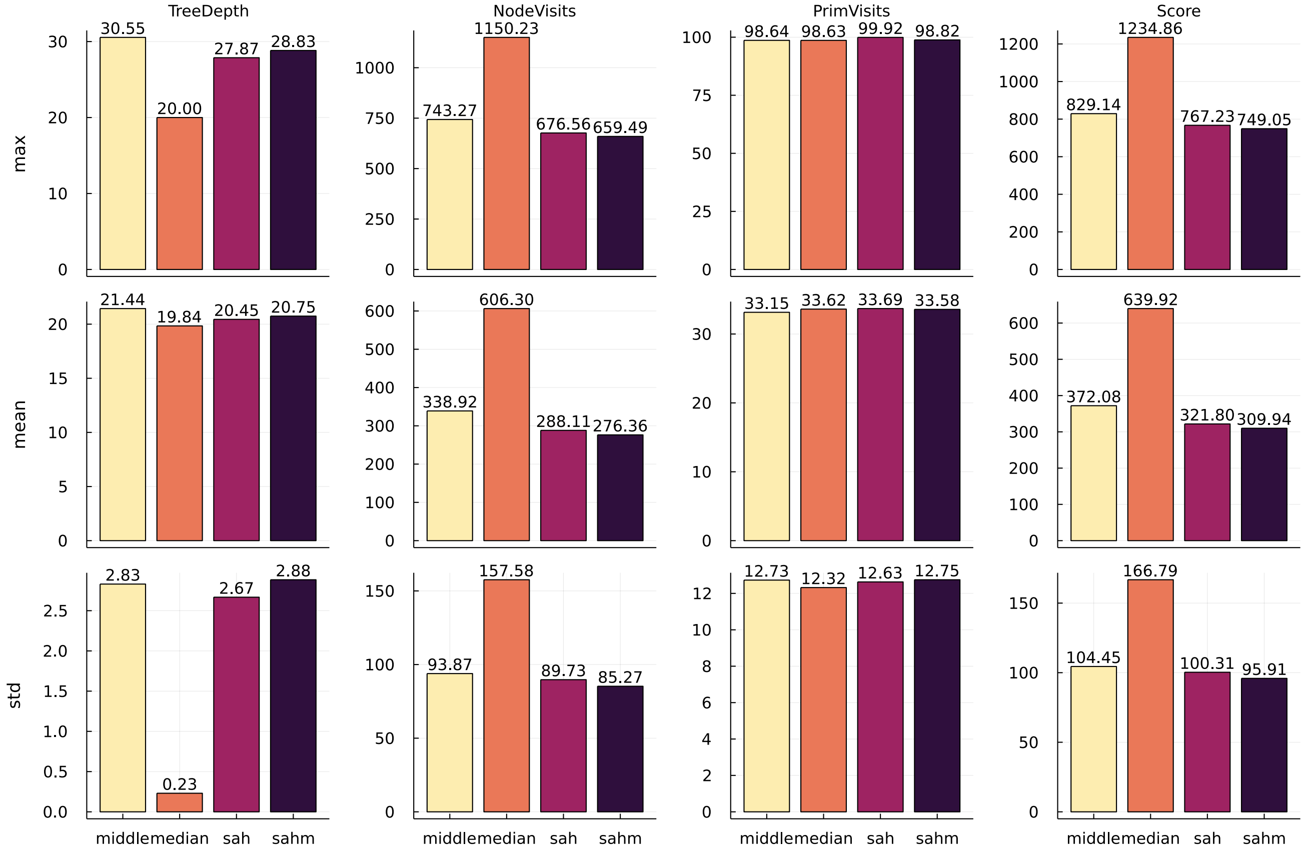
```
compareVisSponza =
```



Visualization of 500 samples with resolution 100x100

Resolution does not affect results much!

```
compareVisSponza100x100 =
```


TreeDepth
NodeVisits
PrimVisits
Score
max
mean
std
middle
median
sah
sahm