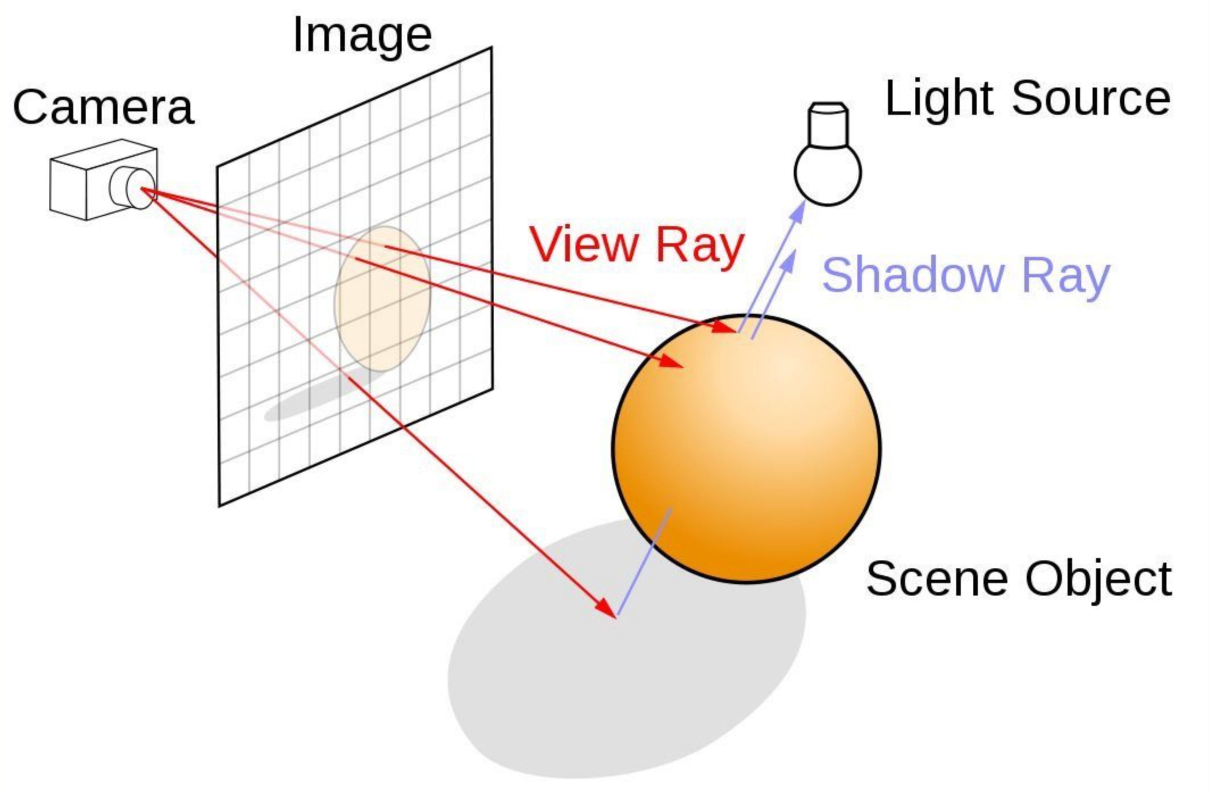
IG3DA - Final presentation

Comparison of BVH construction methods

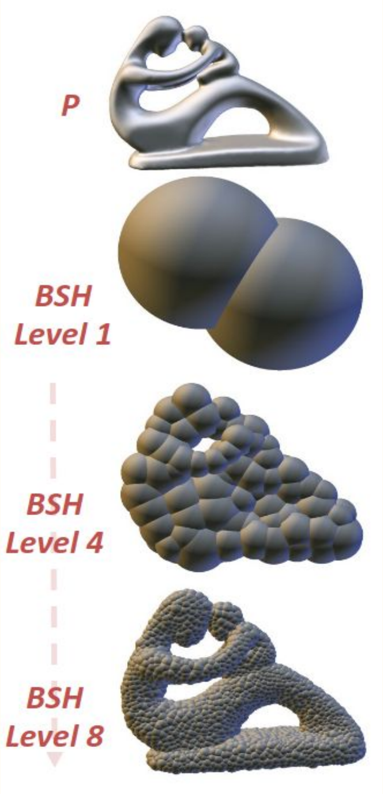
Kedadry Yannis

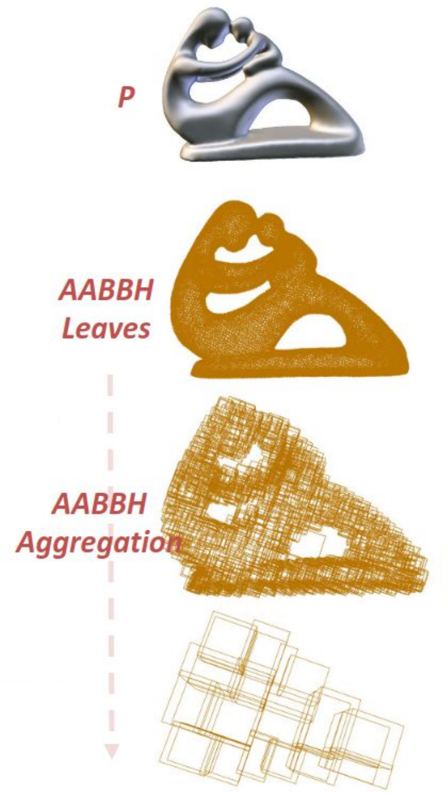
Quick recap on RayTracing



BVH Construction

Top-Down

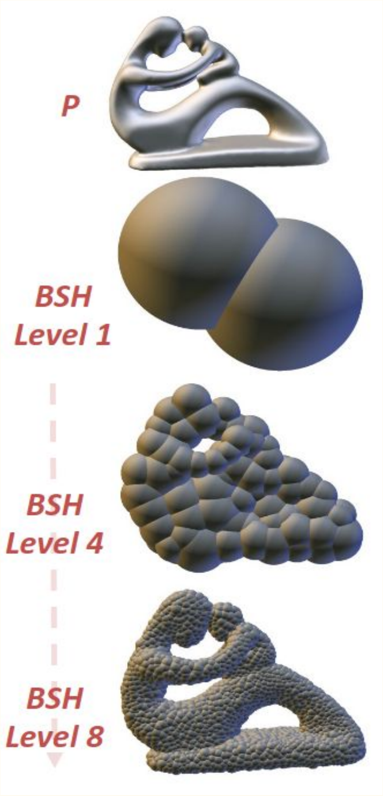




Bottom-Up

Top Down approach:

Top-Down



P
AABBH
Leaves
AABBH
Aggregation

Bottom-Up

Surface Area Heuristic:

$$\text{Total SAH of a BVH} = C_i \sum_{n \in I} \frac{A(n)}{A(root)} + C_l \sum_{n \in L} \frac{A(n)}{A(root)} + C_t \sum_{n \in L} \frac{A(n)}{A(root)} N(n)$$

A - Surface Area of a node’s bounding box

I - Internal nodes

L - Leaf node

N - Number of triangles in a node

$\boldsymbol{C_i}$ - Cost of traversing an internal node

$\boldsymbol{C_l}$ - Cost of traversing a leaf node

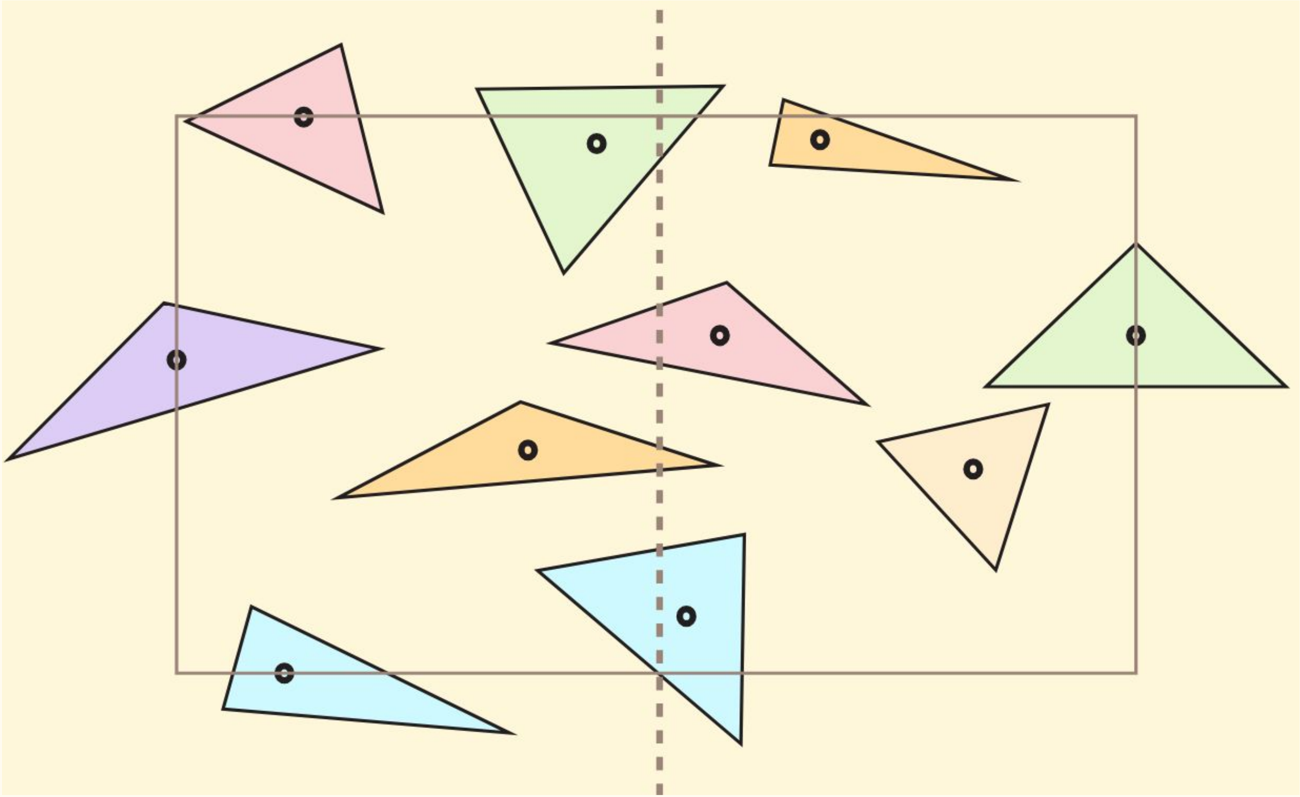
$\boldsymbol{C_t}$ - Cost of intersecting a triangle

To optimize a top-down BVH:

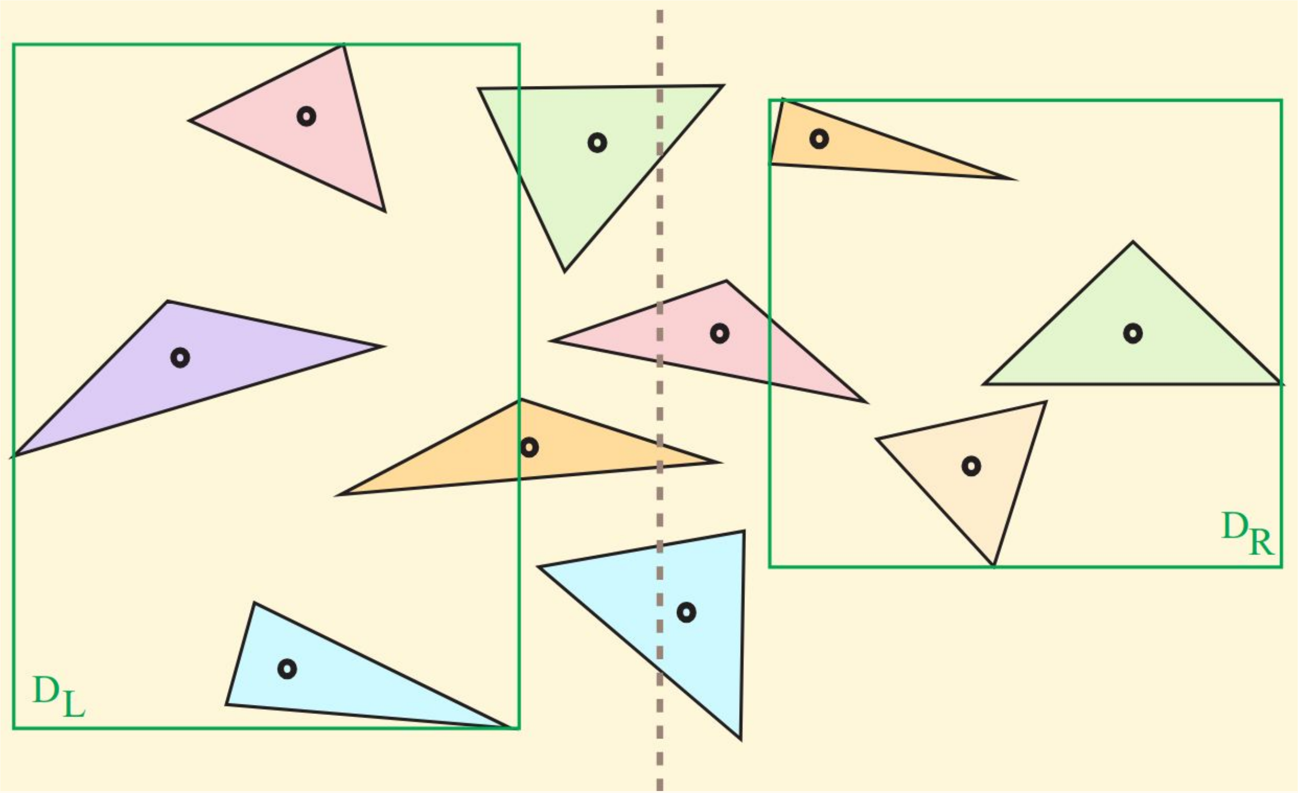
$$E_r = C_i A(n) + C_l (A(n_l) N(n_l)) + A(n_r) N(n_r)),$$

$$E_t = C_t A(n) N(n),$$

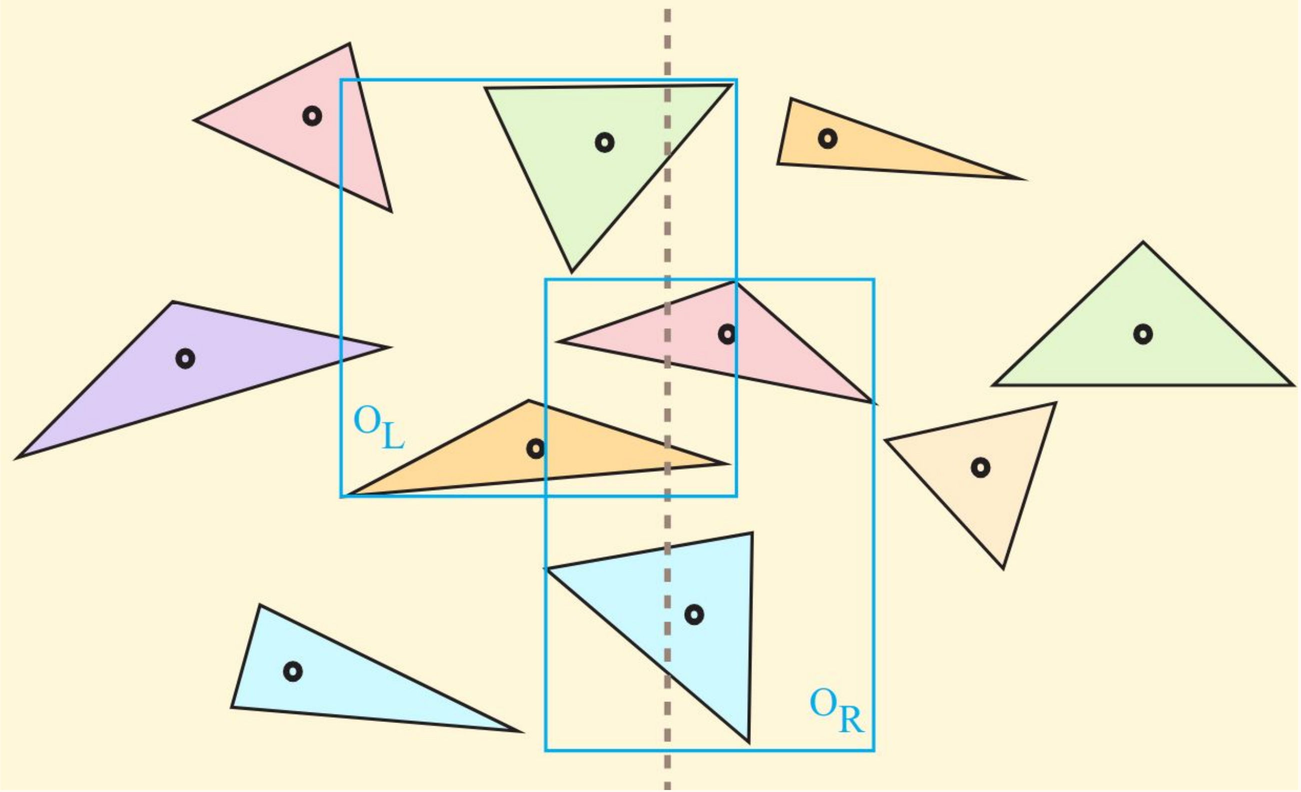
SAH Guided Mid-Point Split Partitioning :



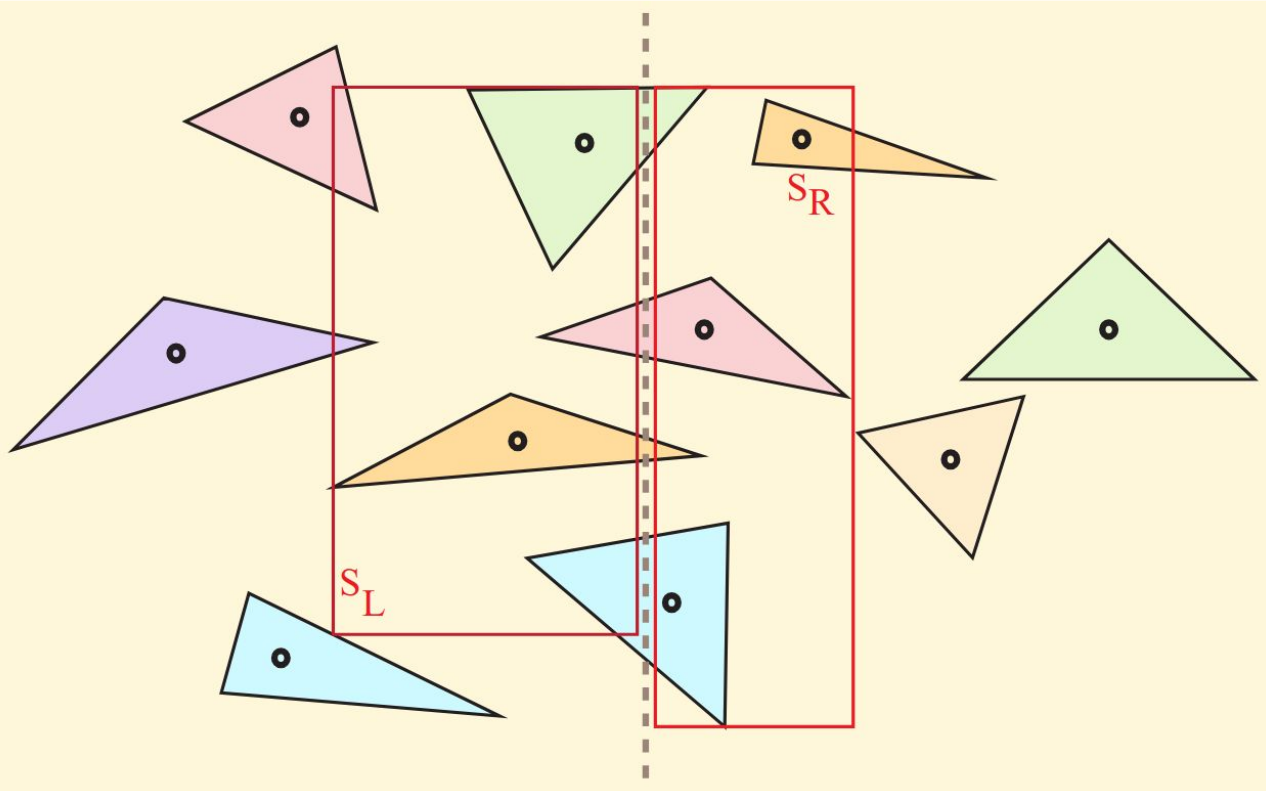
SAH Guided Mid-Point Split Partitioning :



SAH Guided Mid-Point Split Partitioning :



SAH Guided Mid-Point Split Partitioning :



SAH Guided Mid-Point Split Partitioning :

SAH cost of keeping the overlap sets

$$C_O = A(D_L \cup O_L)|D_L \cup O_L| + A(D_R \cup O_R)|D_R \cup O_R|$$

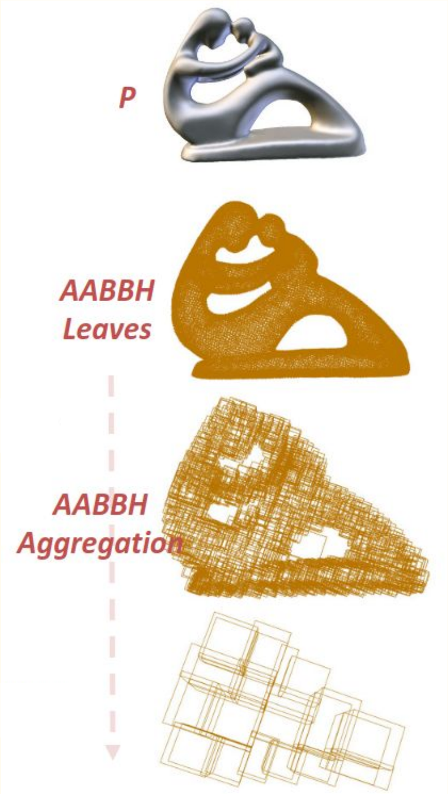
SAH cost of using the split sets

$$C_S = A(D_L \cup S_L)|D_L \cup S_L| + A(D_R \cup S_R)|D_R \cup S_R|,$$

Bottom Up approach:

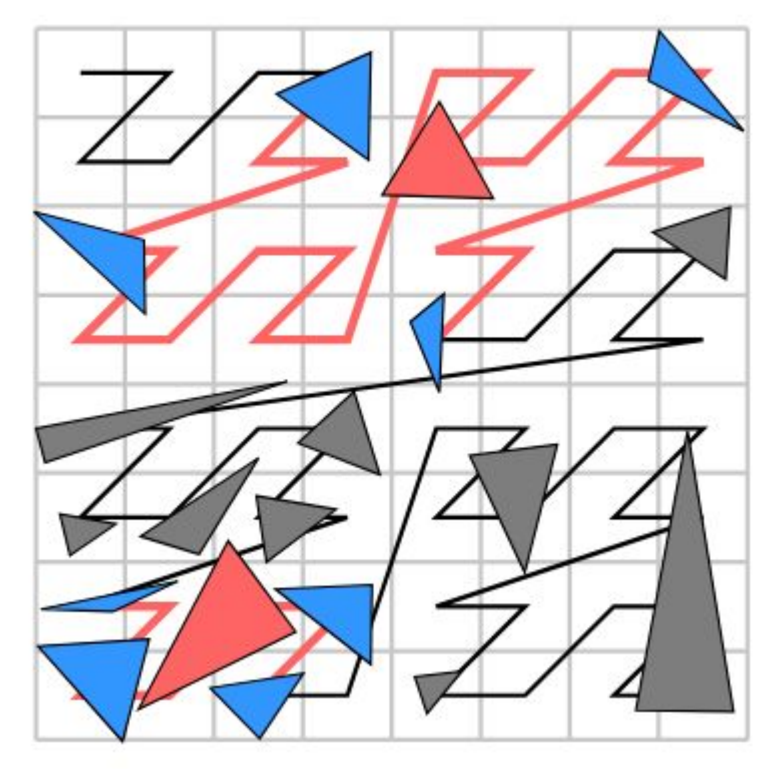
Top-Down

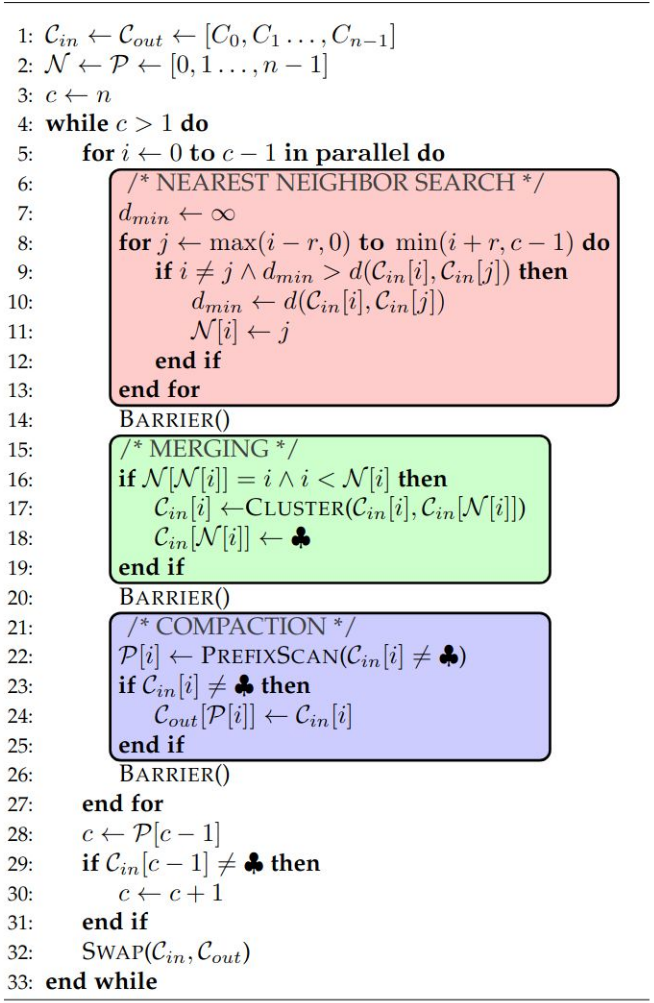
P
BSH Level 1
BSH Level 4
BSH Level 8



Bottom-Up

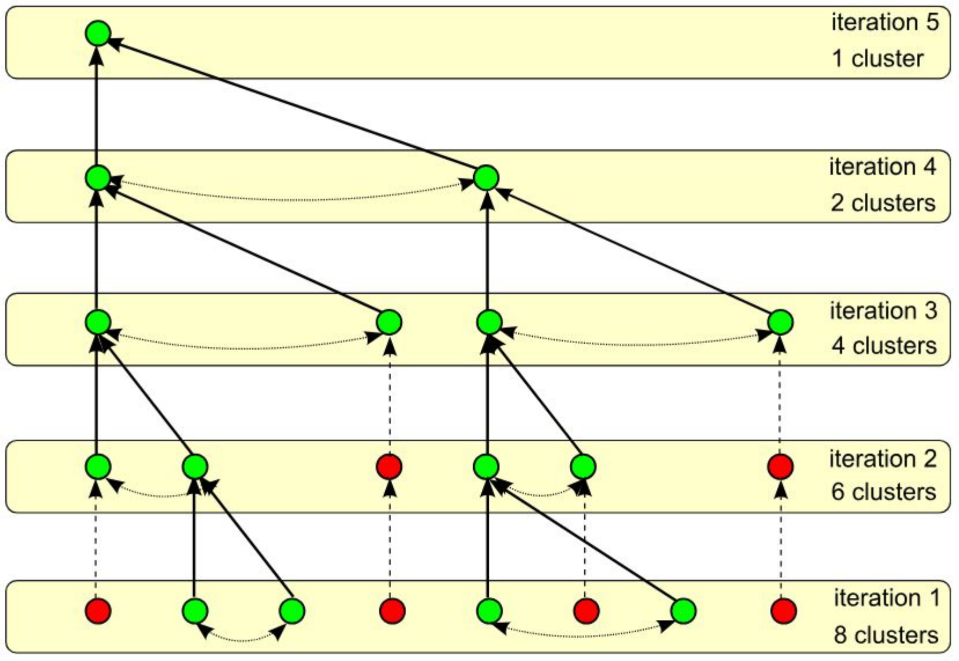
Ploc algorithm:

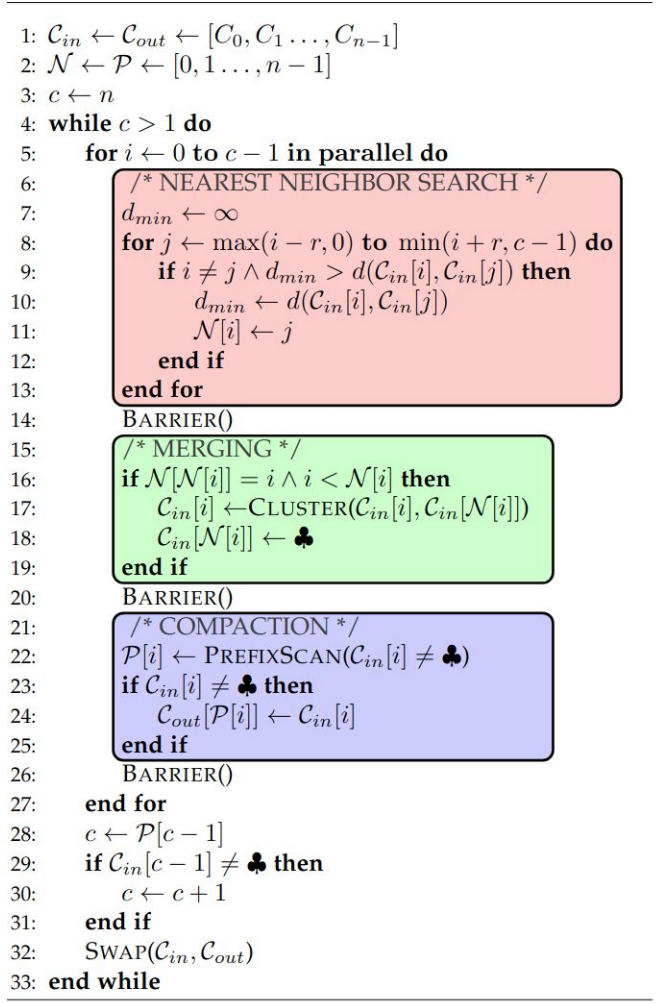




```
$\mathcal{C}_{in} \leftarrow \mathcal{C}_{out} \leftarrow [C_0, C_1 \ldots, C_{n-1}]$
$\mathcal{N} \leftarrow \mathcal{P} \leftarrow [0, 1 \ldots, n-1]$
$c \leftarrow n$
while $c > 1$ do
    for $i \leftarrow 0$ to $c - 1$ in parallel do
        /* NEAREST NEIGHBOR SEARCH */
        $d_{min} \leftarrow \infty$
        for $j \leftarrow \max(i - r, 0)$ to $\min(i + r, c - 1)$ do
            if $i \neq j \wedge d_{min} > d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$ then
                $d_{min} \leftarrow d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$
                $\mathcal{N}[i] \leftarrow j$
            end if
        end for
        BARRIER()
        /* MERGING */
        if $\mathcal{N}[\mathcal{N}[i]] = i \wedge i < \mathcal{N}[i]$ then
            $\mathcal{C}_{in}[i] \leftarrow$ CLUSTER($\mathcal{C}_{in}[i], \mathcal{C}_{in}[\mathcal{N}[i]]$)
            $\mathcal{C}_{in}[\mathcal{N}[i]] \leftarrow \clubsuit$
        end if
        BARRIER()
        /* COMPACTION */
        $\mathcal{P}[i] \leftarrow$ PREFIXSCAN($\mathcal{C}_{in}[i] \neq \clubsuit$)
        if $\mathcal{C}_{in}[i] \neq \clubsuit$ then
            $\mathcal{C}_{out}[\mathcal{P}[i]] \leftarrow \mathcal{C}_{in}[i]$
        end if
        BARRIER()
    end for
    $c \leftarrow \mathcal{P}[c - 1]$
    if $\mathcal{C}_{in}[c - 1] \neq \clubsuit$ then
        $c \leftarrow c + 1$
    end if
    SWAP($\mathcal{C}_{in}, \mathcal{C}_{out}$)
end while
```

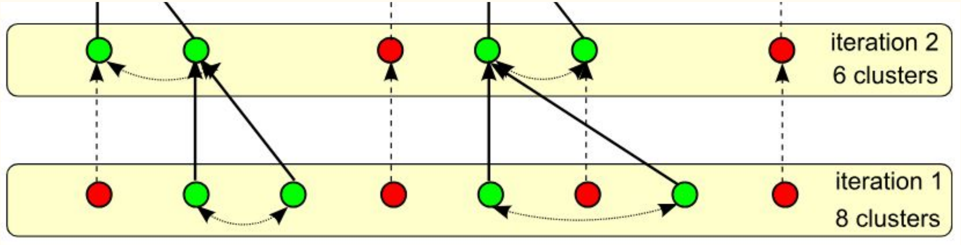
Ploc algorithm:



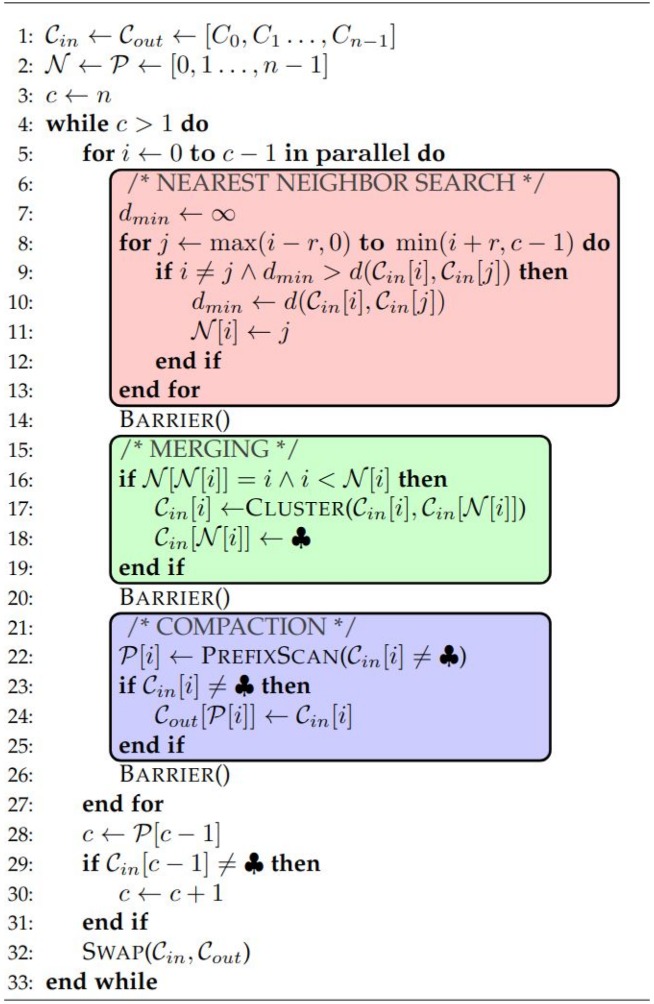


```
$\mathcal{C}_{in} \leftarrow \mathcal{C}_{out} \leftarrow [C_0, C_1 \ldots, C_{n-1}]$
$\mathcal{N} \leftarrow \mathcal{P} \leftarrow [0, 1 \ldots, n-1]$
$c \leftarrow n$
while $c > 1$ do
    for $i \leftarrow 0$ to $c - 1$ in parallel do
        /* NEAREST NEIGHBOR SEARCH */
        $d_{min} \leftarrow \infty$
        for $j \leftarrow \max(i - r, 0)$ to $\min(i + r, c - 1)$ do
            if $i \neq j \wedge d_{min} > d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$ then
                $d_{min} \leftarrow d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$
                $\mathcal{N}[i] \leftarrow j$
            end if
        end for
        BARRIER()
        /* MERGING */
        if $\mathcal{N}[\mathcal{N}[i]] = i \wedge i < \mathcal{N}[i]$ then
            $\mathcal{C}_{in}[i] \leftarrow$CLUSTER$(\mathcal{C}_{in}[i], \mathcal{C}_{in}[\mathcal{N}[i]])$
            $\mathcal{C}_{in}[\mathcal{N}[i]] \leftarrow \clubsuit$
        end if
        BARRIER()
        /* COMPACTION */
        $\mathcal{P}[i] \leftarrow$ PREFIXSCAN$(\mathcal{C}_{in}[i] \neq \clubsuit)$
        if $\mathcal{C}_{in}[i] \neq \clubsuit$ then
            $\mathcal{C}_{out}[\mathcal{P}[i]] \leftarrow \mathcal{C}_{in}[i]$
        end if
        BARRIER()
    end for
    $c \leftarrow \mathcal{P}[c - 1]$
    if $\mathcal{C}_{in}[c - 1] \neq \clubsuit$ then
        $c \leftarrow c + 1$
    end if
    SWAP$(\mathcal{C}_{in}, \mathcal{C}_{out})$
end while
```

Ploc algorithm:



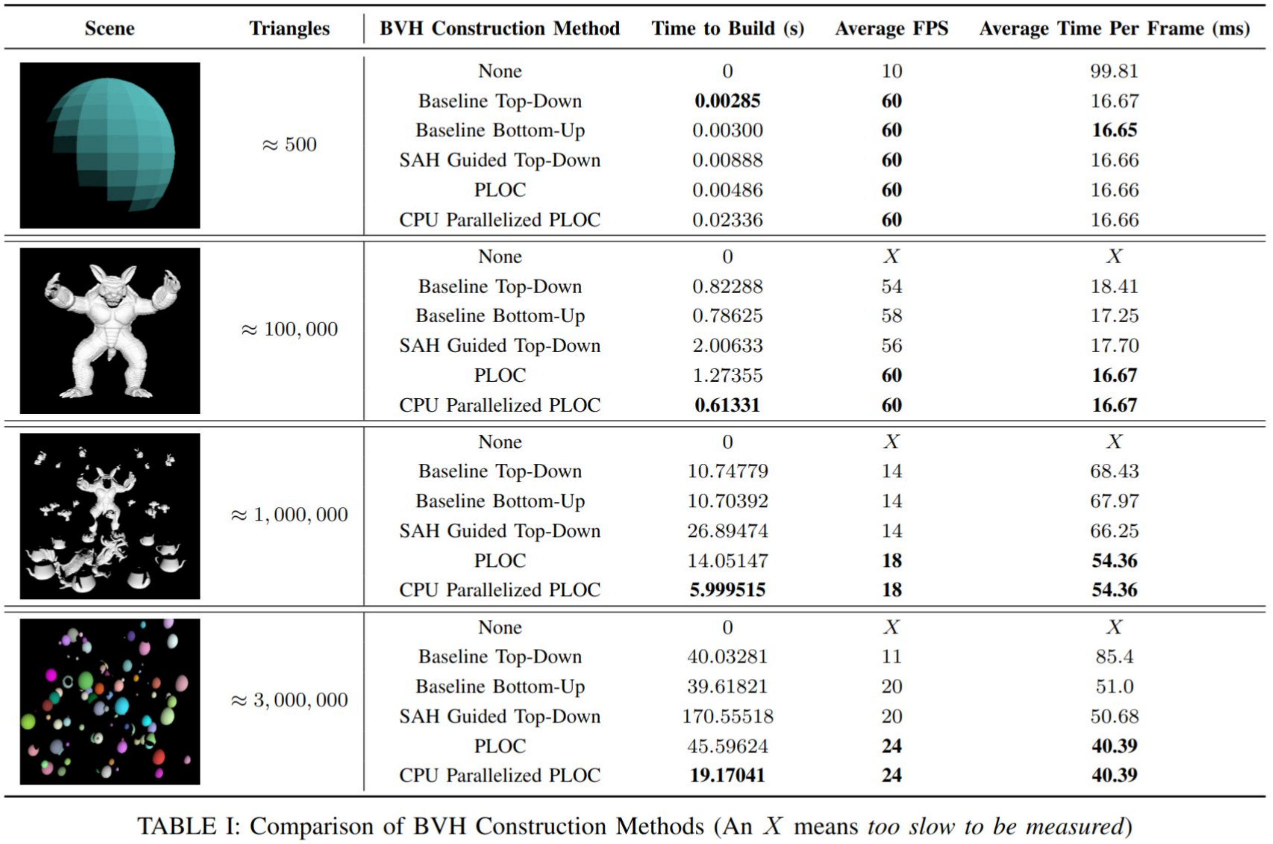
Cin	0	1	2	3	4	5	6	7
New$C$	0	1	0	0	1	0	0	0
P	0	0	1	1	1	2	2	2
Cin	0	8+0	♣	3	8+1	5	♣	7
Cin $\neq$♣	1	1	0	1	1	1	0	1
P	0	1	2	2	3	4	5	5
$Cout$	0	8	3	9	5	7	X	X



```
$\mathcal{C}_{in} \leftarrow \mathcal{C}_{out} \leftarrow [C_0, C_1 \ldots, C_{n-1}]$
$\mathcal{N} \leftarrow \mathcal{P} \leftarrow [0, 1 \ldots, n-1]$
$c \leftarrow n$
while $c > 1$ do
    for $i \leftarrow 0$ to $c - 1$ in parallel do
        /* NEAREST NEIGHBOR SEARCH */
        $d_{min} \leftarrow \infty$
        for $j \leftarrow \max(i - r, 0)$ to $\min(i + r, c - 1)$ do
            if $i \neq j \wedge d_{min} > d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$ then
                $d_{min} \leftarrow d(\mathcal{C}_{in}[i], \mathcal{C}_{in}[j])$
                $\mathcal{N}[i] \leftarrow j$
            end if
        end for
        BARRIER()
        /* MERGING */
        if $\mathcal{N}[\mathcal{N}[i]] = i \wedge i < \mathcal{N}[i]$ then
            $\mathcal{C}_{in}[i] \leftarrow$CLUSTER$(\mathcal{C}_{in}[i], \mathcal{C}_{in}[\mathcal{N}[i]])$
            $\mathcal{C}_{in}[\mathcal{N}[i]] \leftarrow \clubsuit$
        end if
        BARRIER()
        /* COMPACTION */
        $\mathcal{P}[i] \leftarrow$ PREFIXSCAN$(\mathcal{C}_{in}[i] \neq \clubsuit)$
        if $\mathcal{C}_{in}[i] \neq \clubsuit$ then
            $\mathcal{C}_{out}[\mathcal{P}[i]] \leftarrow \mathcal{C}_{in}[i]$
        end if
        BARRIER()
    end for
    $c \leftarrow \mathcal{P}[c - 1]$
    if $\mathcal{C}_{in}[c - 1] \neq \clubsuit$ then
        $c \leftarrow c + 1$
    end if
    SWAP$(\mathcal{C}_{in}, \mathcal{C}_{out})$
end while
```

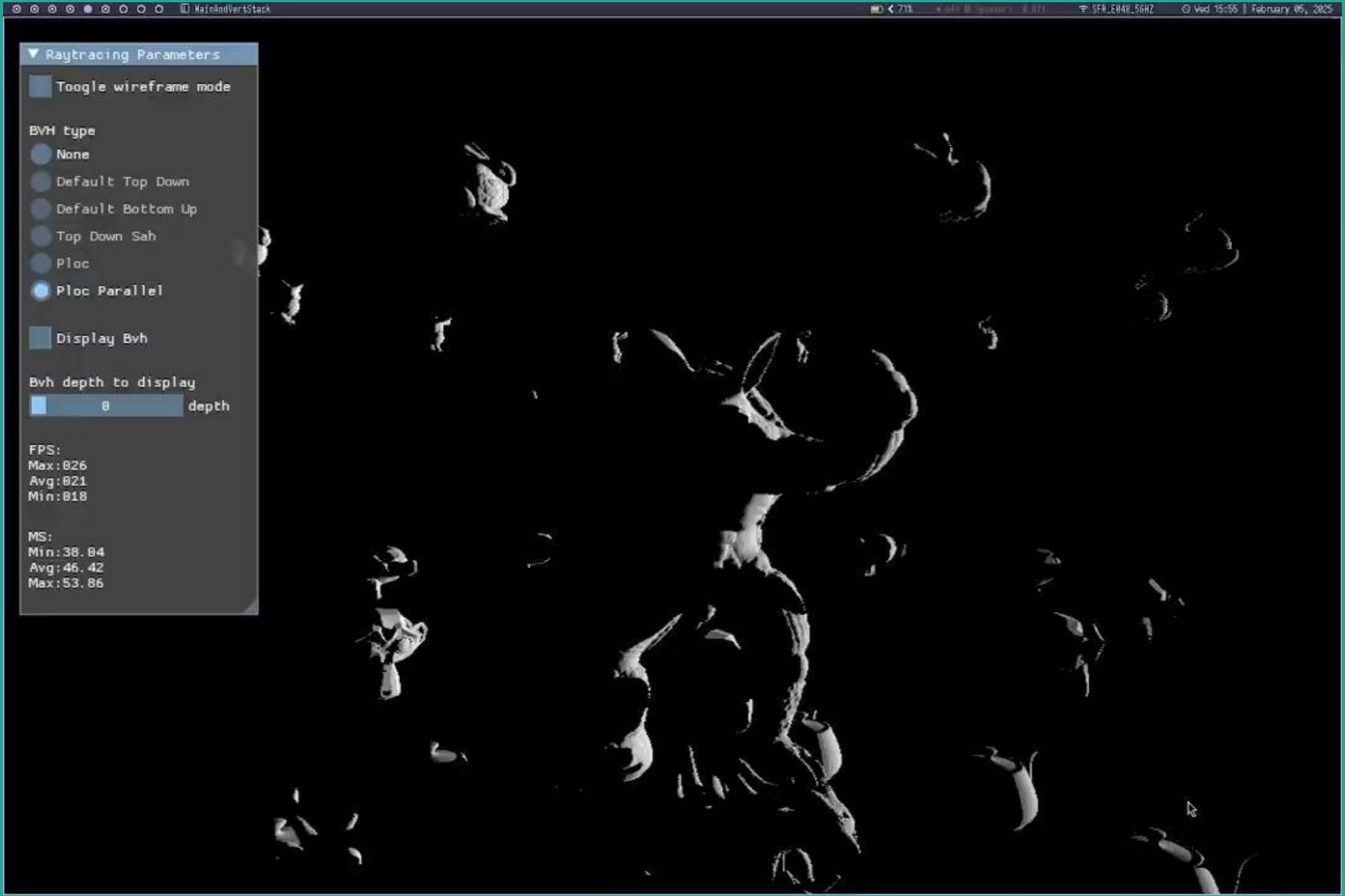
Implementation details:

Results:



Scene	Triangles	BVH Construction Method	Time to Build (s)	Average FPS	Average Time Per Frame (ms)
	≈ 500	None	0	10	99.81
		Baseline Top-Down	**0.00285**	**60**	16.67
		Baseline Bottom-Up	0.00300	**60**	**16.65**
		SAH Guided Top-Down	0.00888	**60**	16.66
		PLOC	0.00486	**60**	16.66
		CPU Parallelized PLOC	0.02336	**60**	16.66
	$\approx 100,000$	None	0	X	X
		Baseline Top-Down	0.82288	54	18.41
		Baseline Bottom-Up	0.78625	58	17.25
		SAH Guided Top-Down	2.00633	56	17.70
		PLOC	1.27355	**60**	**16.67**
		CPU Parallelized PLOC	**0.61331**	**60**	**16.67**
	$\approx 1,000,000$	None	0	X	X
		Baseline Top-Down	10.74779	14	68.43
		Baseline Bottom-Up	10.70392	14	67.97
		SAH Guided Top-Down	26.89474	14	66.25
		PLOC	14.05147	**18**	**54.36**
		CPU Parallelized PLOC	**5.999515**	**18**	**54.36**
	$\approx 3,000,000$	None	0	X	X
		Baseline Top-Down	40.03281	11	85.4
		Baseline Bottom-Up	39.61821	20	51.0
		SAH Guided Top-Down	170.55518	20	50.68
		PLOC	45.59624	**24**	**40.39**
		CPU Parallelized PLOC	**19.17041**	**24**	**40.39**

TABLE I: Comparison of BVH Construction Methods (An X means *too slow to be measured*)



▼ Raytracing Parameters

□ Toogle wireframe mode

BVH type

- ○ None
- ○ Default Top Down
- ○ Default Bottom Up
- ○ Top Down Sah
- ○ Ploc
- ● Ploc Parallel

□ Display Bvh

Bvh depth to display

0 depth

FPS:
Max:026
Avg:021
Min:018

MS:
Min:38.04
Avg:46.42
Max:53.86