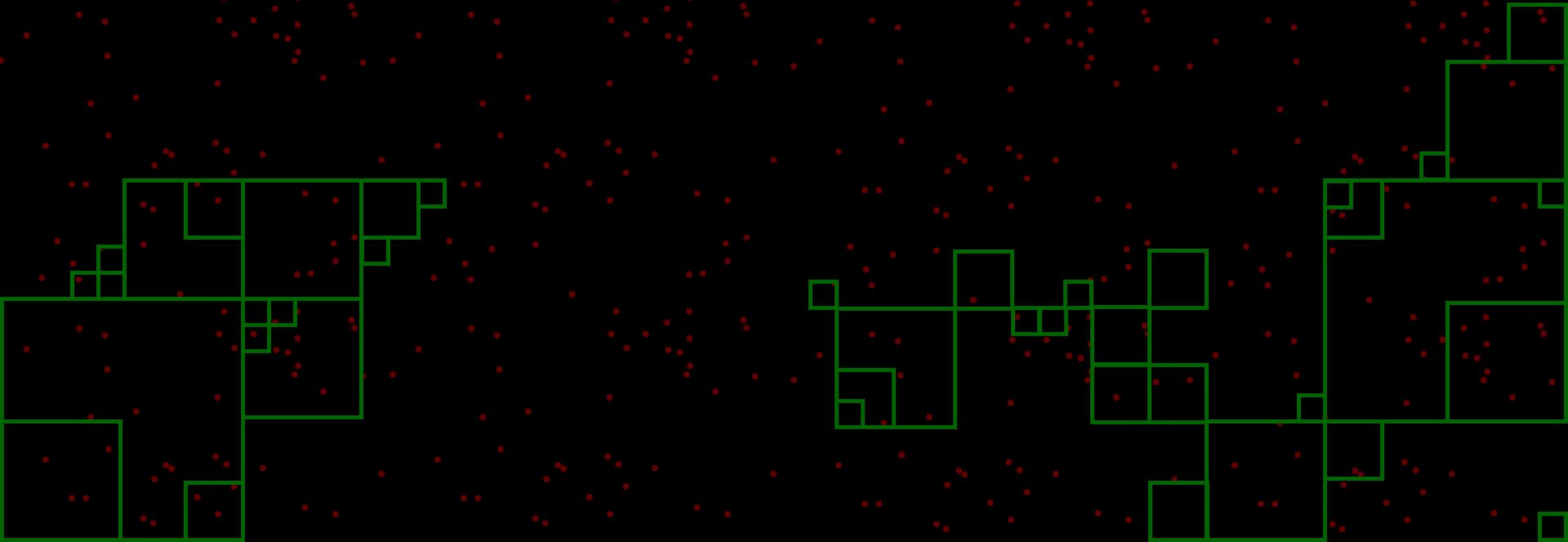


# Optimized Search Manager

Xavier Olivenza  
Research  
CITM 2016-17

# What is a search engine for?

-Search for an entity in an area/range





# Where I can use this?



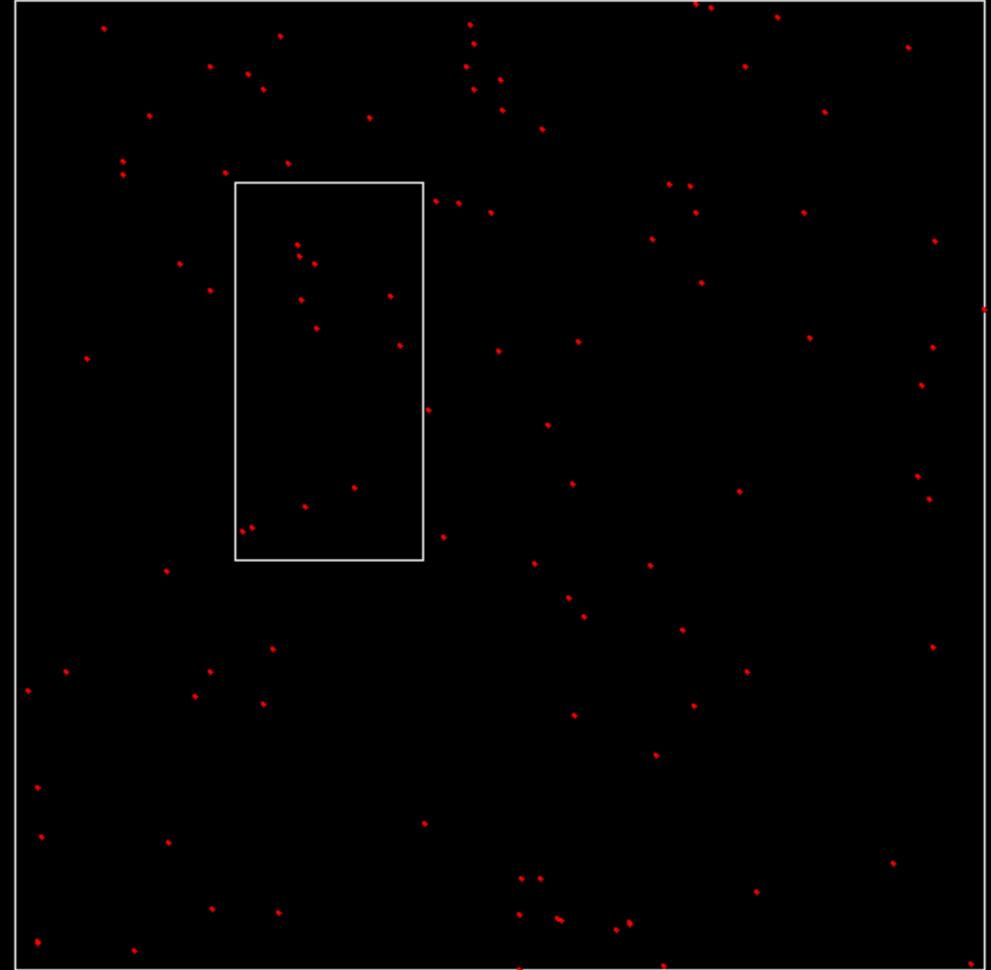


# Brute force

-You have to do as many checks as entities are evaluated

...

-What happens if we group and order the entities?



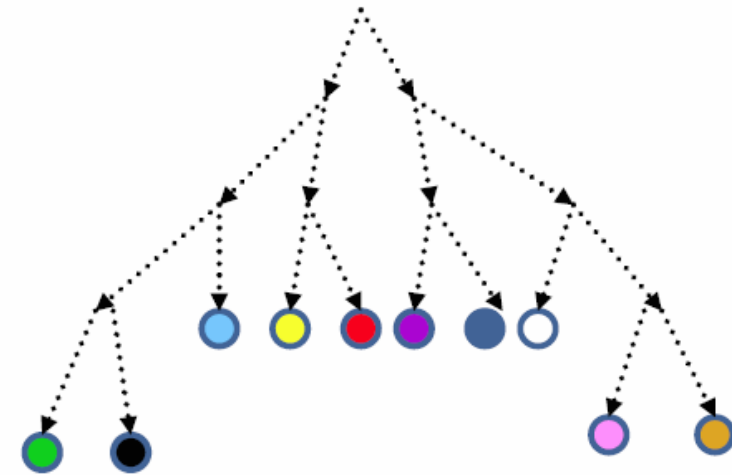
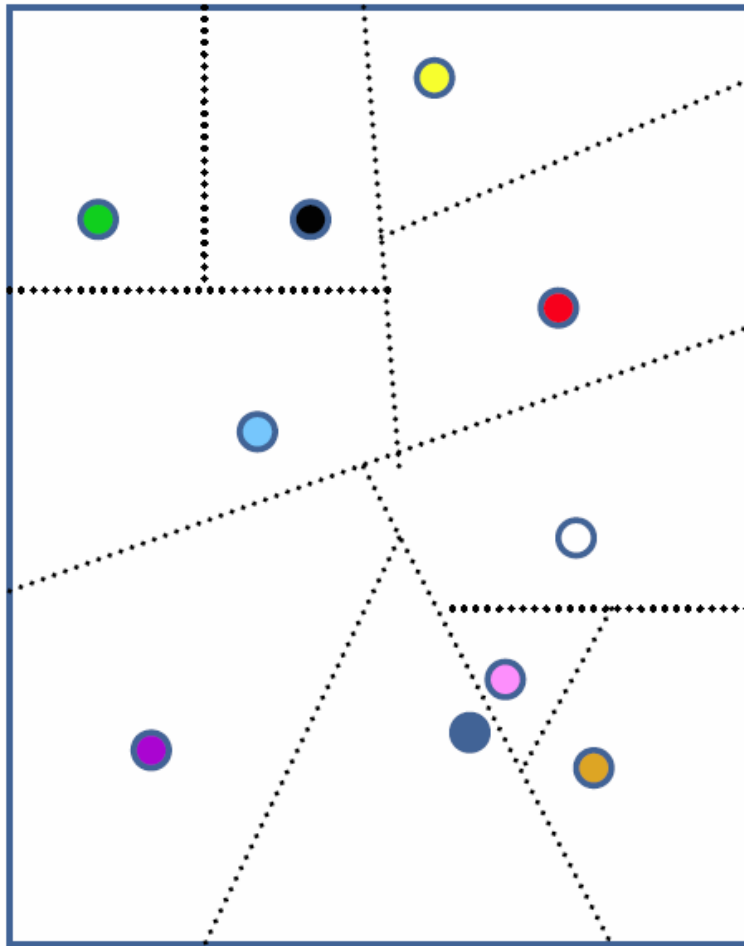
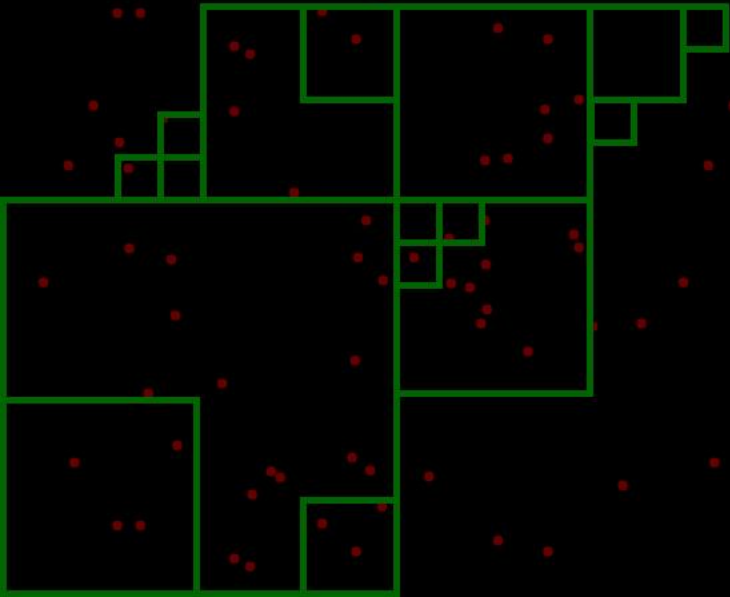
# Space Partitioning

· Dividing a space into two or more subsets which do not overlap

· Algorithms that tend to be hierarchical

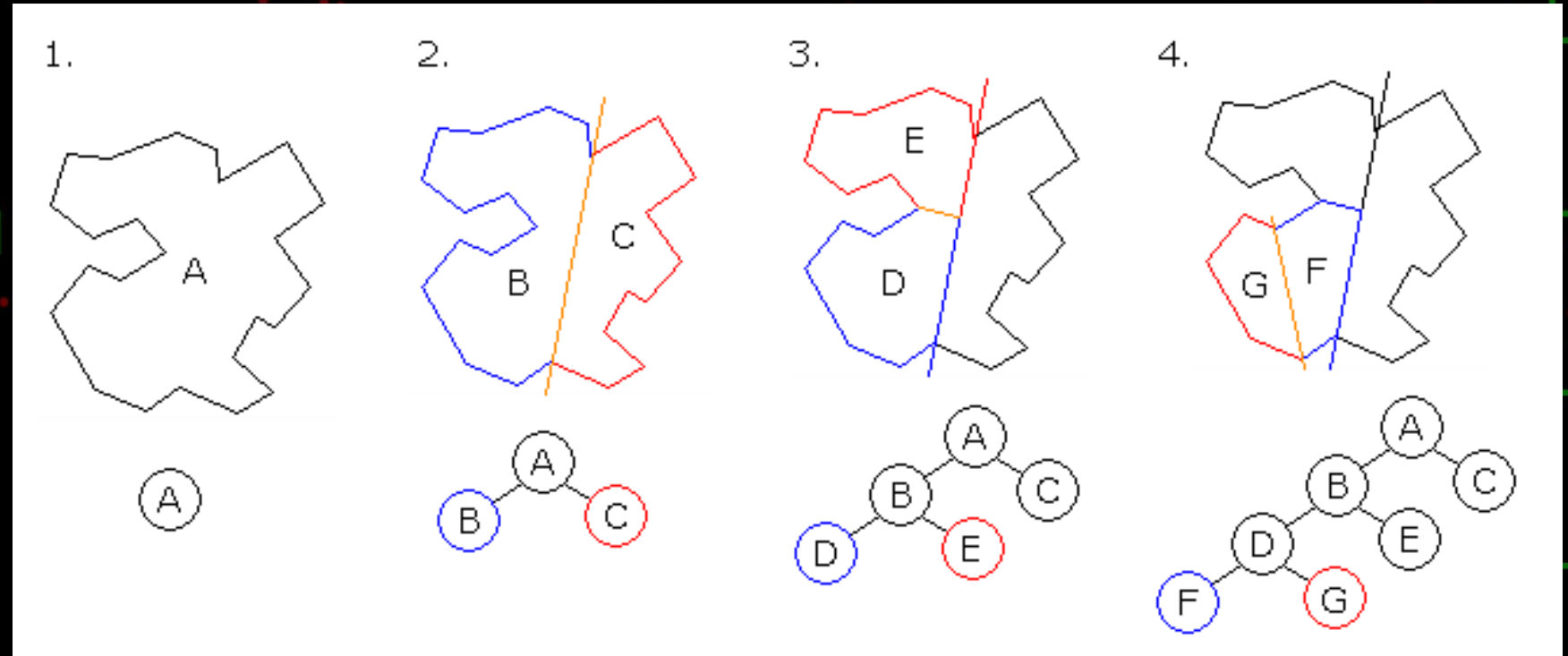
Most common partition data structures:

- BSP Trees
- Quadtrees
- Octrees
- K-dimensional trees
- R-Trees



# Binary Space Partitioning (BSP)

- Generalization
- Origin: Quickly draw polygonal 3d scenes
- Slow generation -> Pre-calculate



# BSP, Where is used?

Id-Teck 1 Doom



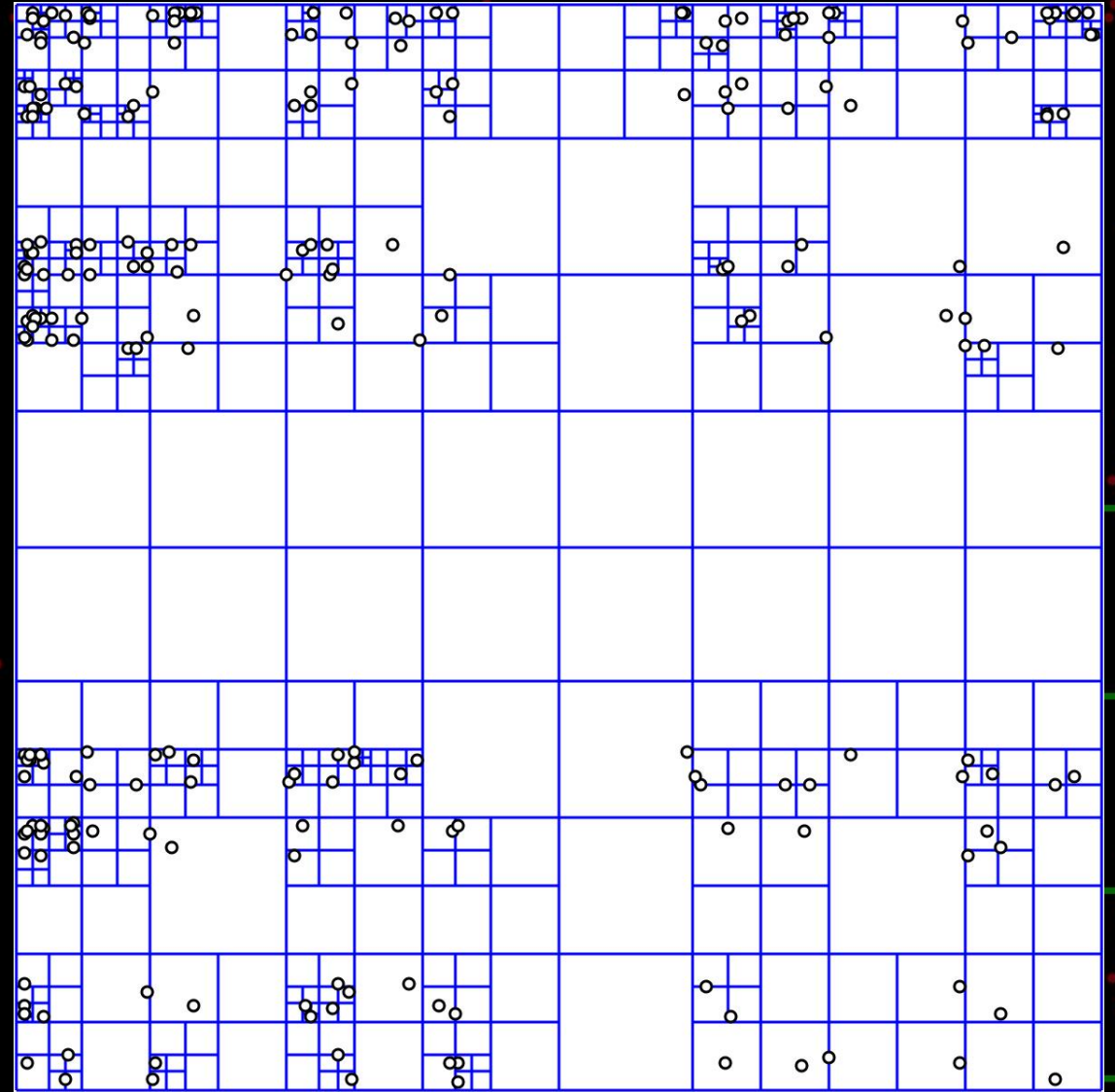
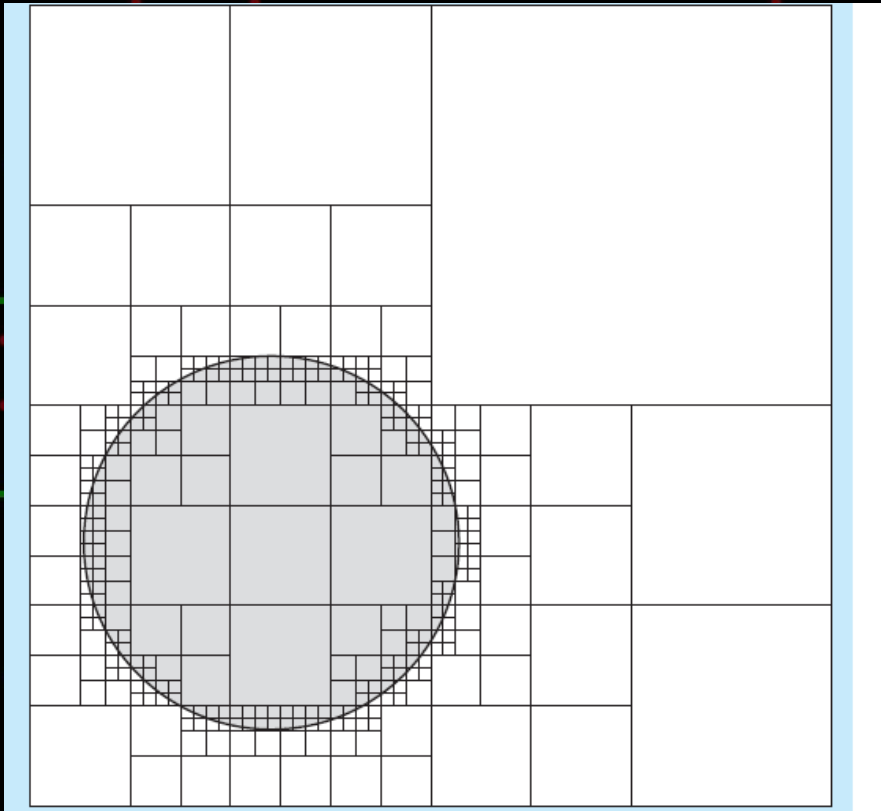
Quake Engine + descendants





# Quadtree

- Generally 2D
- When it reaches max node capacity -> split

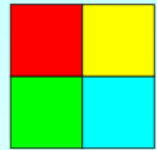




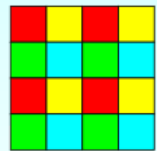
# Quadtree, Tree



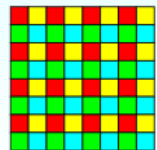
**Root**



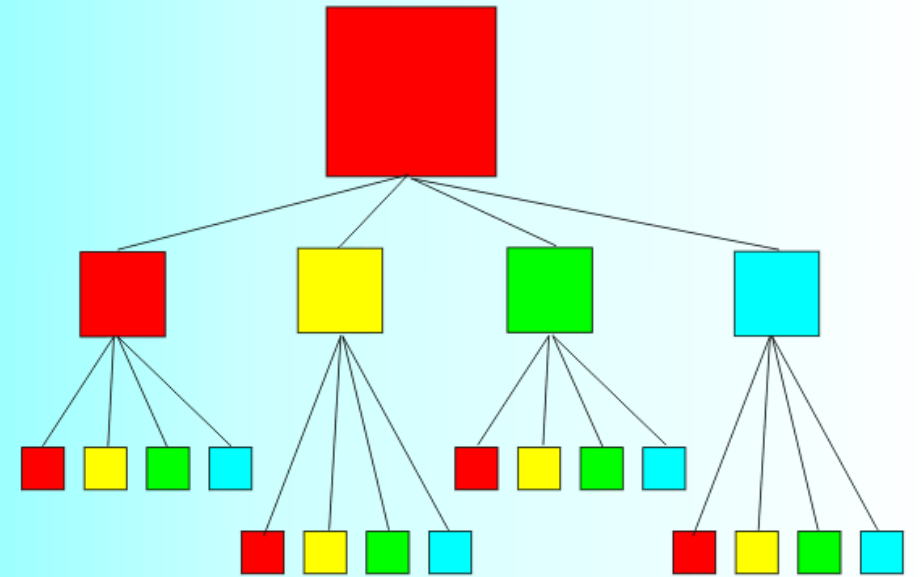
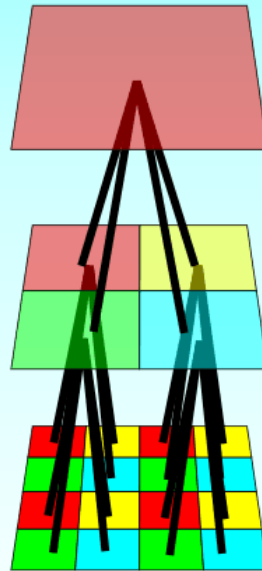
**First level**



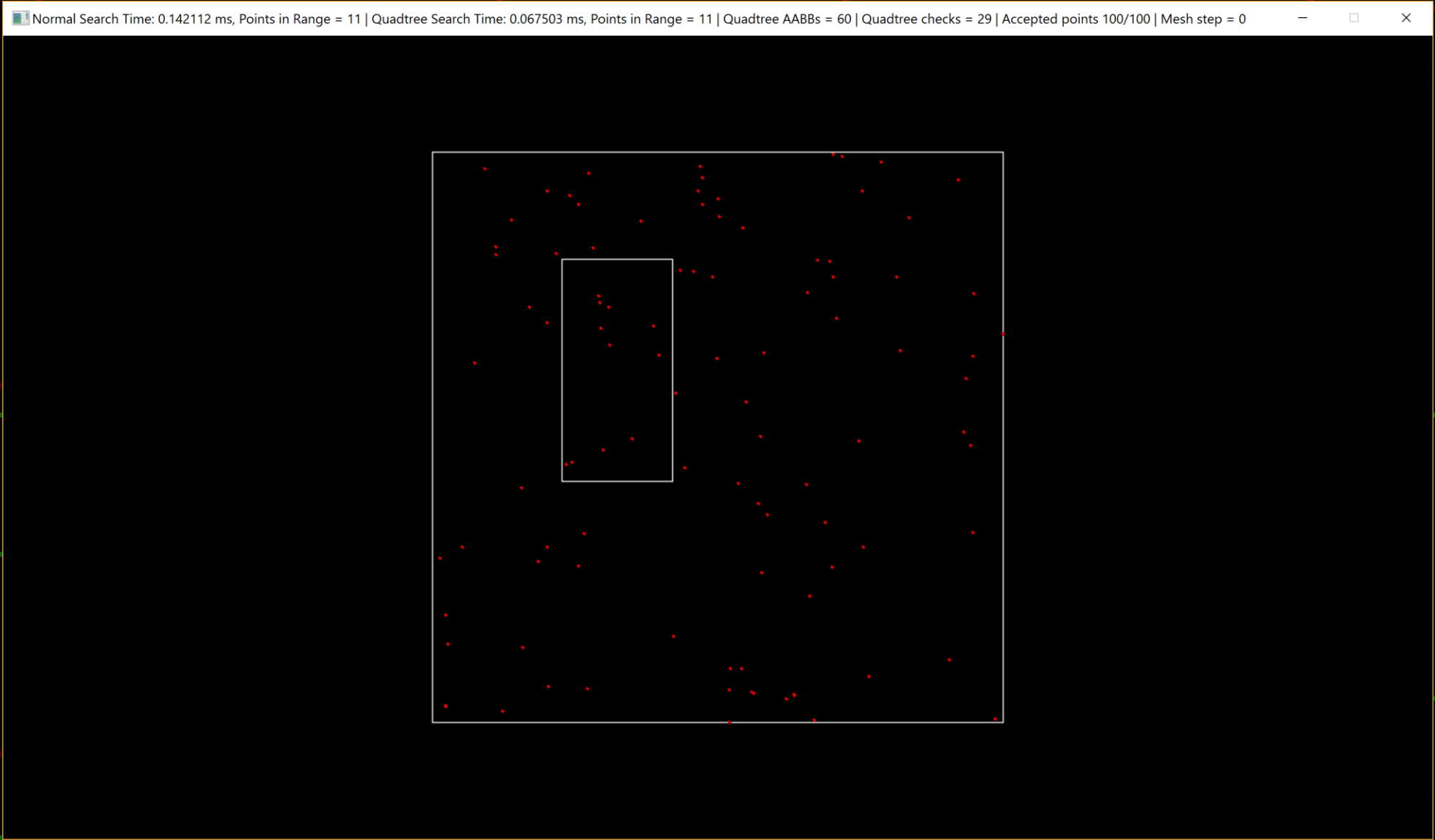
**Second**



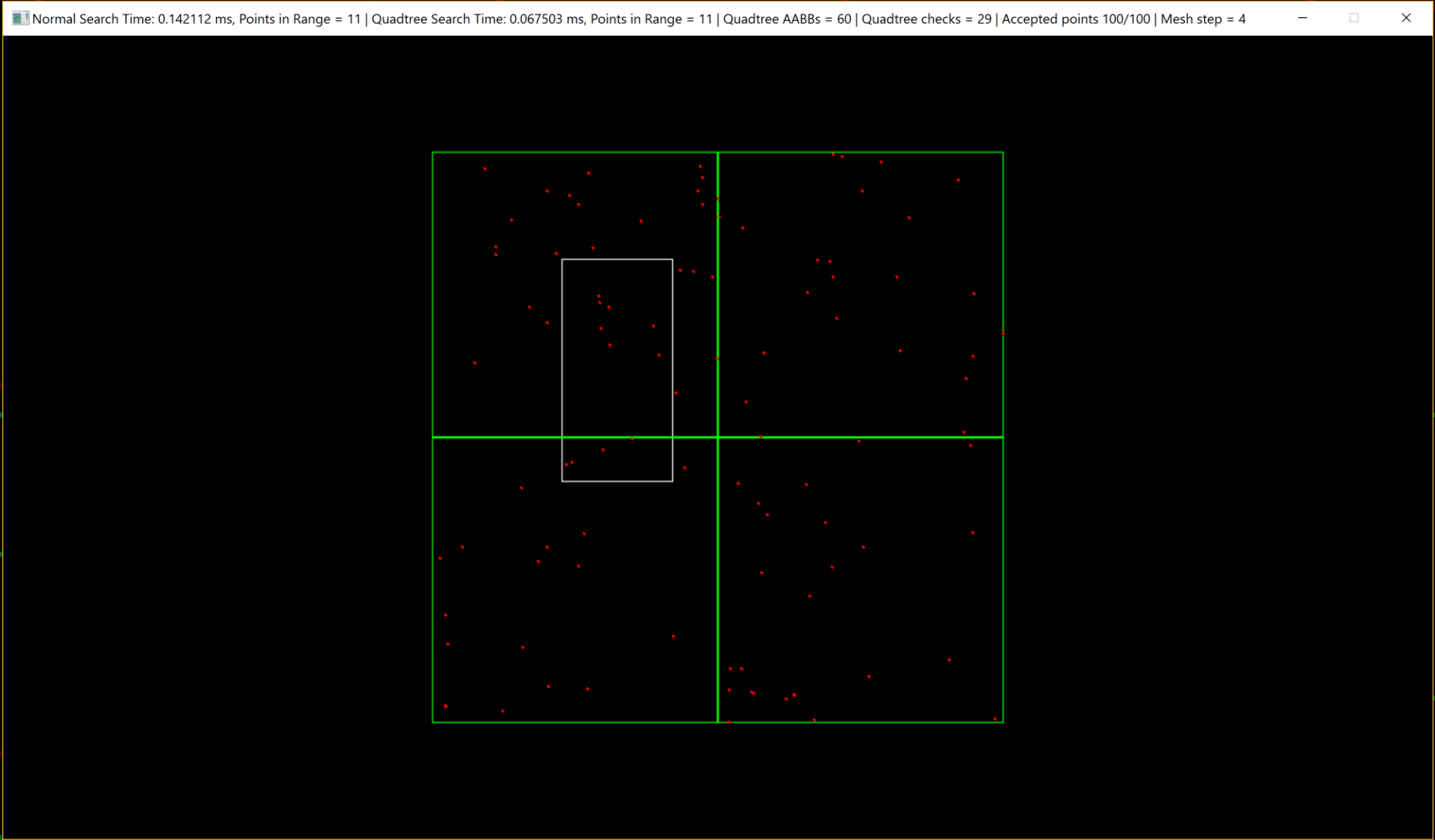
**Third**



# Quadtree, Tree



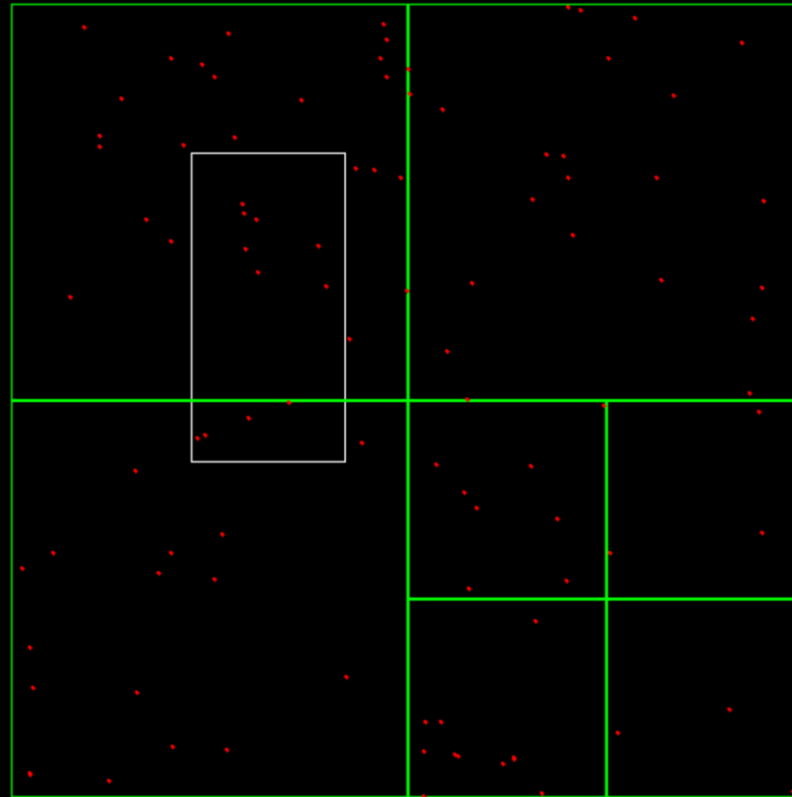
# Quadtree, Tree





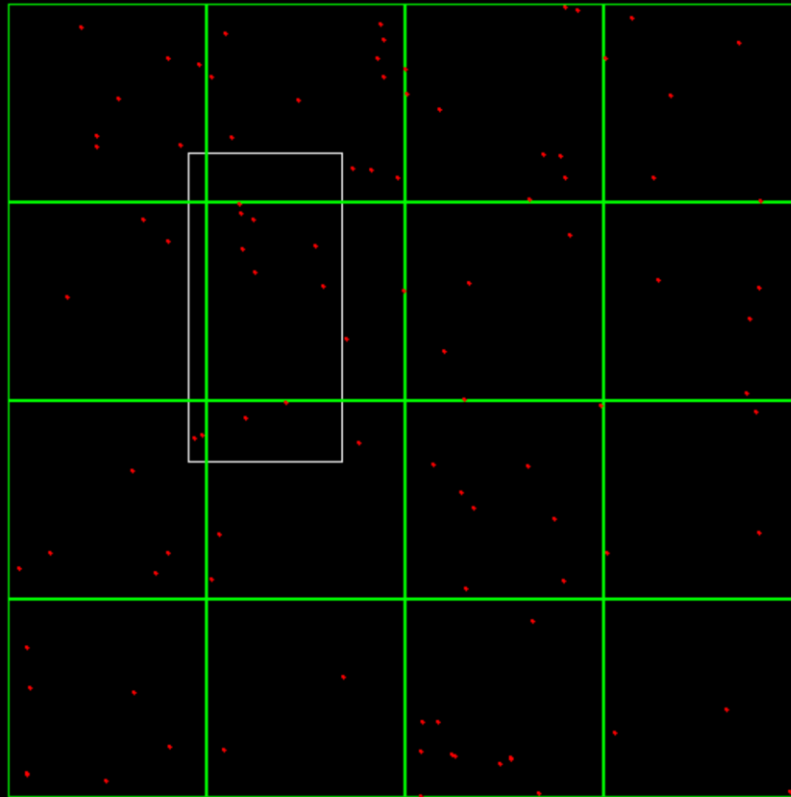
# Quadtree, Tree

Normal Search Time: 0.142112 ms, Points in Range = 11 | Quadtree Search Time: 0.067503 ms, Points in Range = 11 | Quadtree AABBs = 60 | Quadtree checks = 29 | Accepted points 100/100 | Mesh step = 8



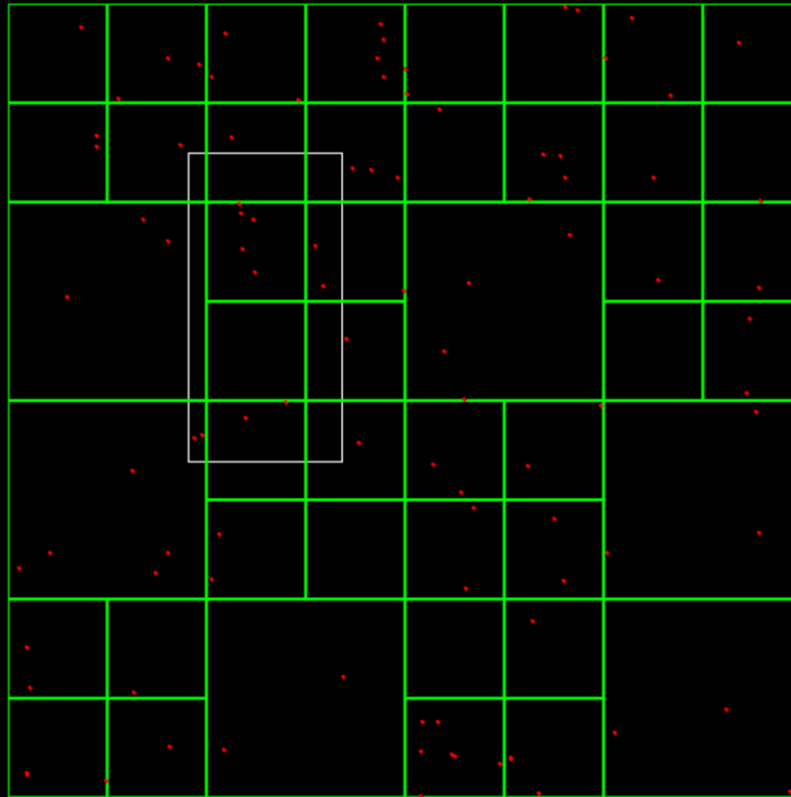
# Quadtree, Tree

Normal Search Time: 0.142112 ms, Points in Range = 11 | Quadtree Search Time: 0.067503 ms, Points in Range = 11 | Quadtree AABBs = 60 | Quadtree checks = 29 | Accepted points 100/100 | Mesh step = 20



# Quadtree, Tree

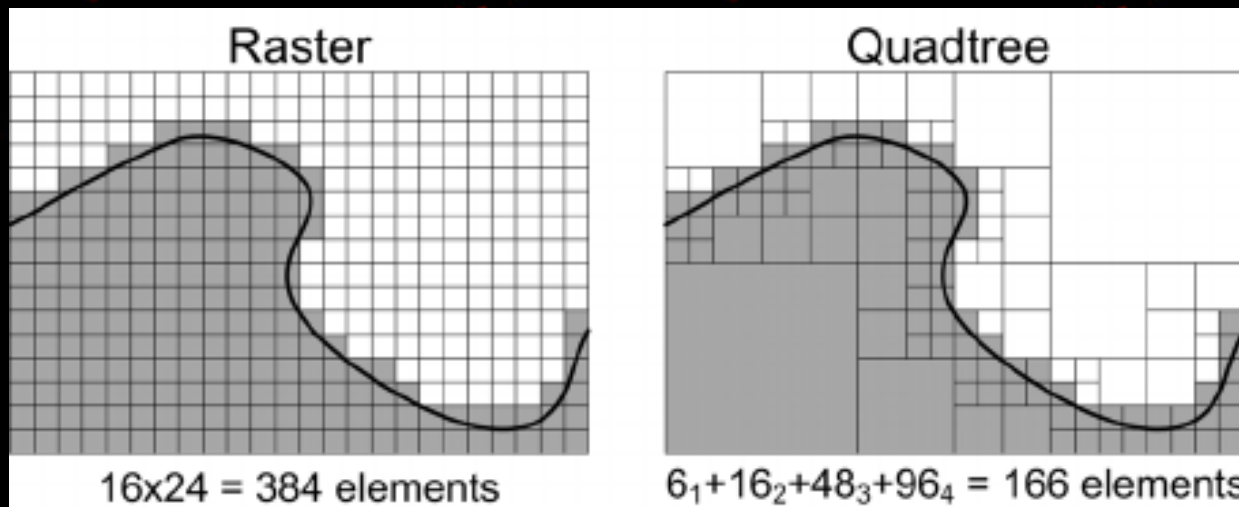
Normal Search Time: 0.142112 ms, Points in Range = 11 | Quadtree Search Time: 0.067503 ms, Points in Range = 11 | Quadtree AABBs = 60 | Quadtree checks = 29 | Accepted points 100/100 | Mesh step = 60





# Quadtree, Where is used?

Image processing/compression



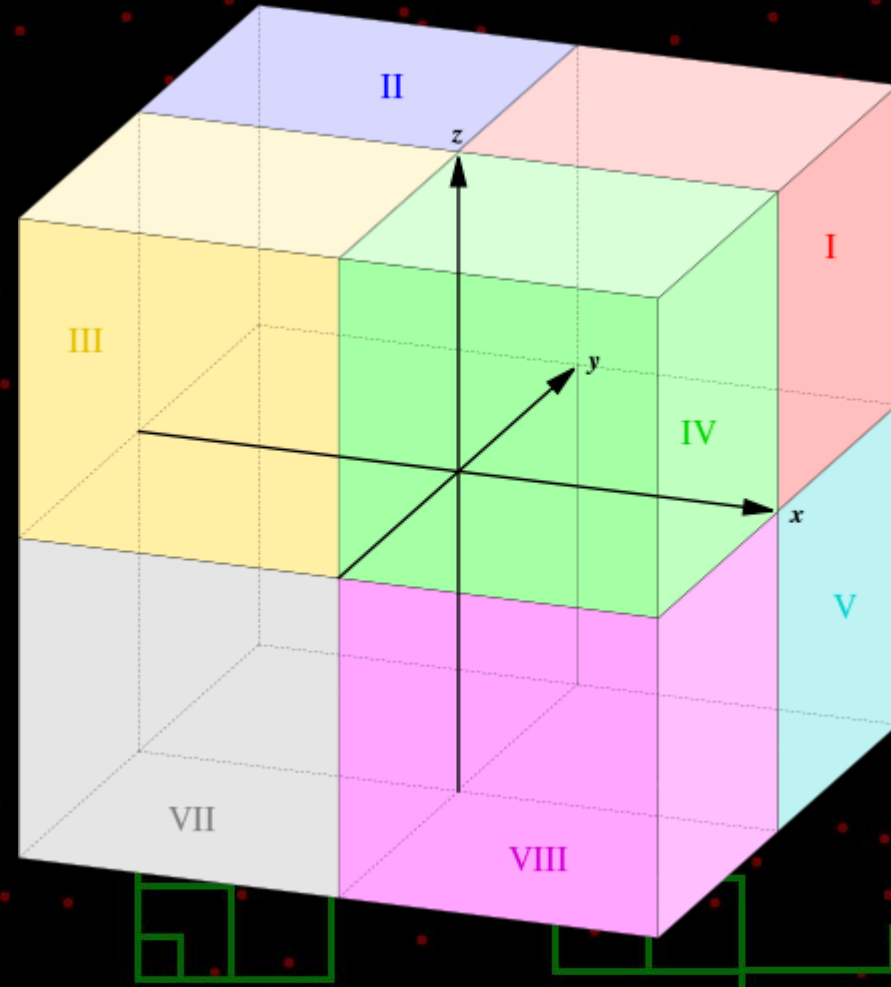
# Quadtree, Where is used?

- Collisions
- Camera culling



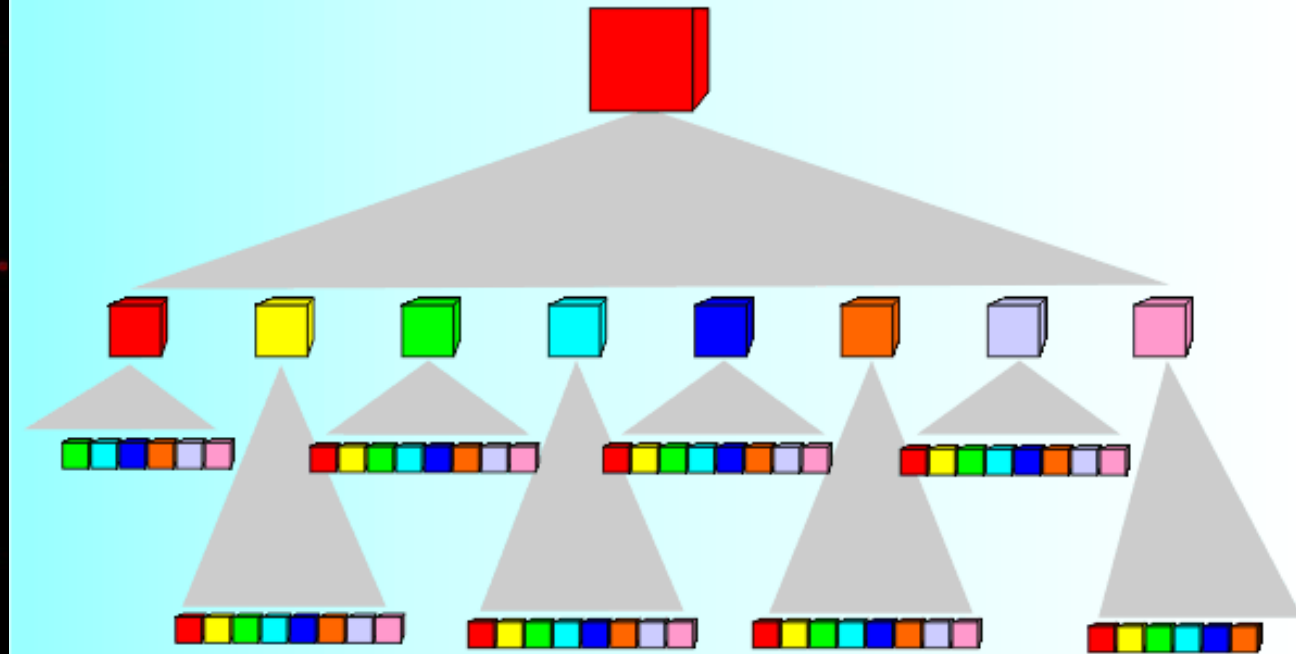
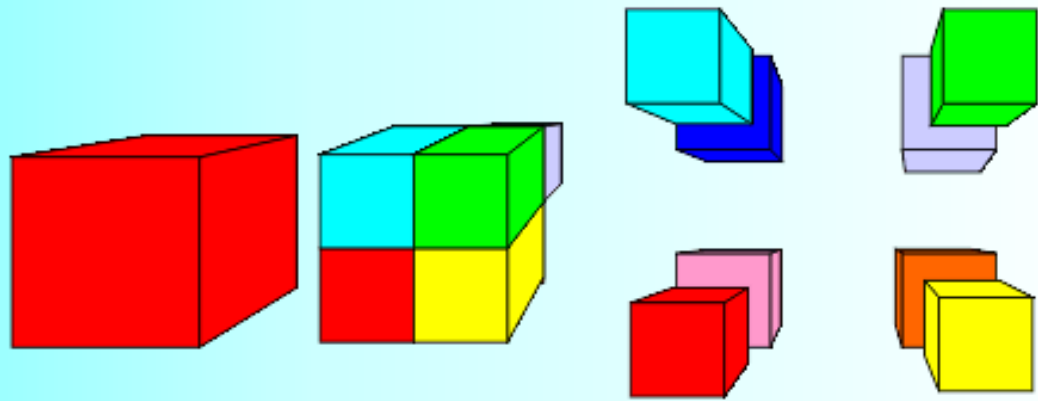
# Octree

- Quadtree analogue in 3D
- 3D graphics and video game engines



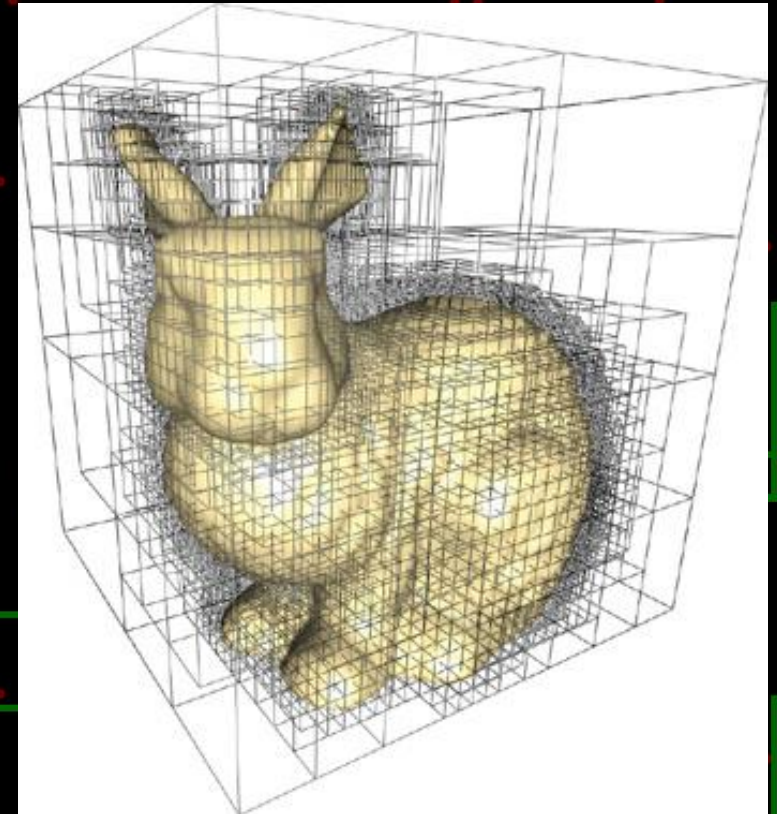
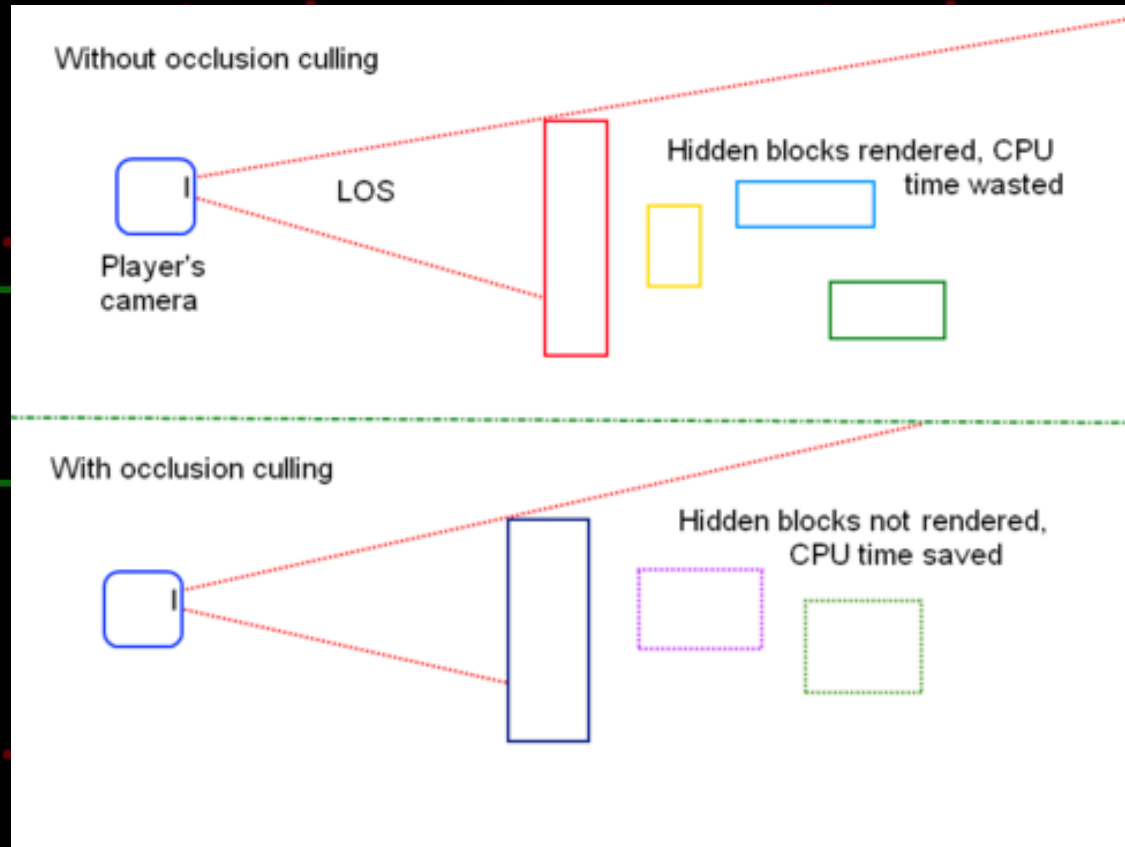


# Octree, Tree

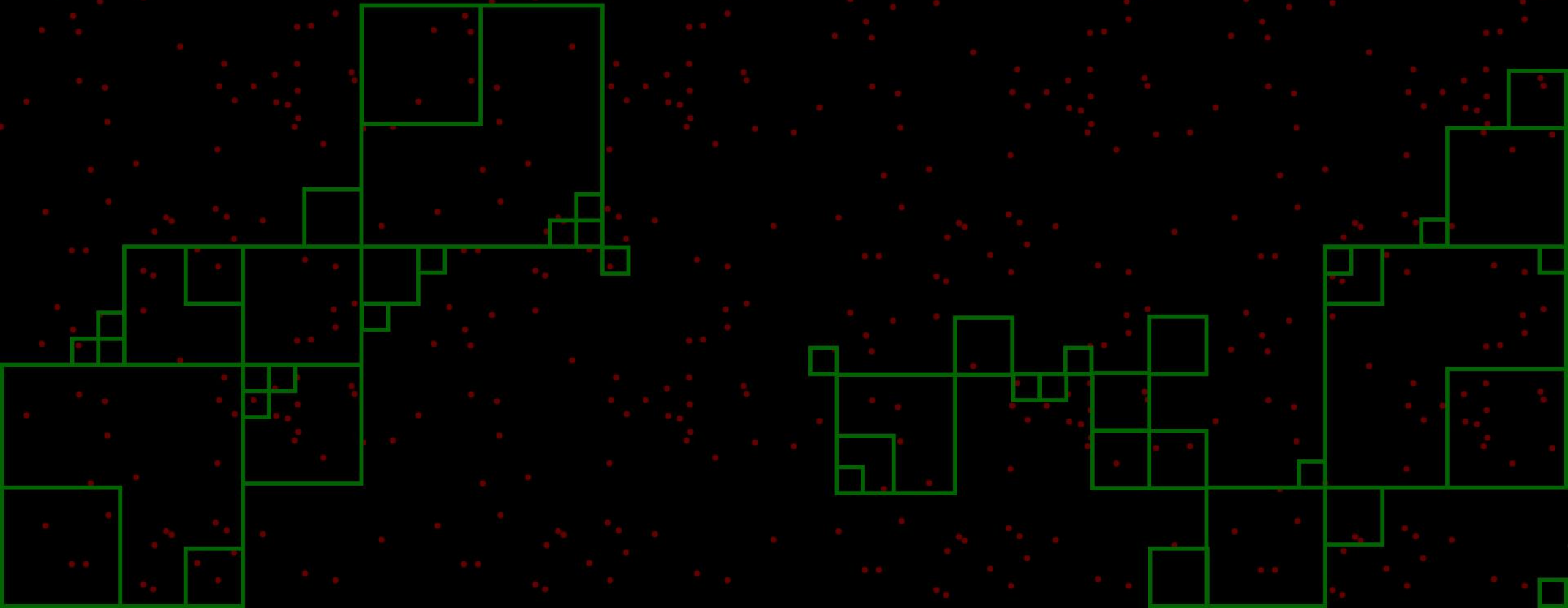


# Octree, Where is used?

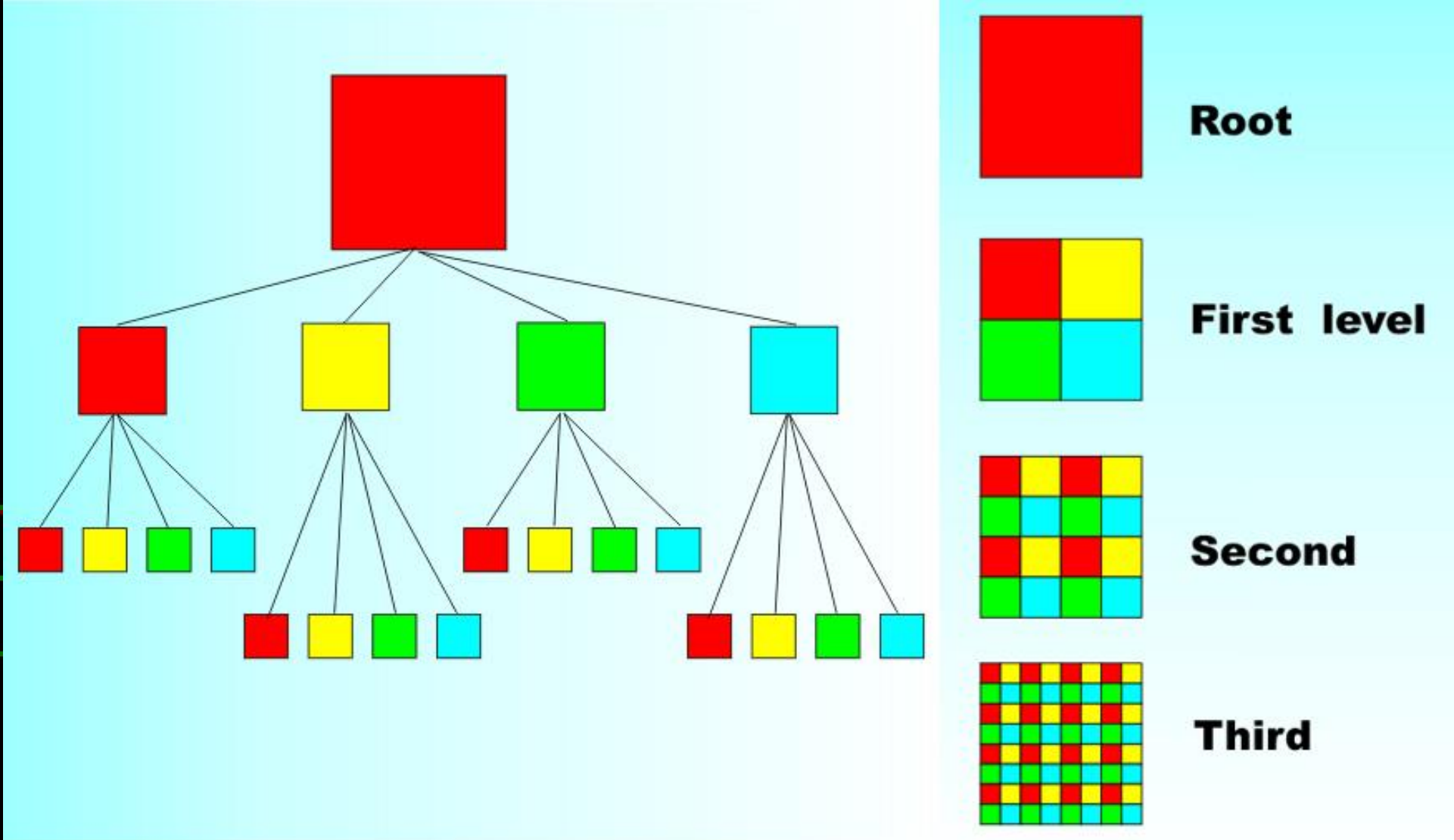
- 3D Graphics
- Efficient collision detection in three dimensions
- Occlusion Culling (OC)

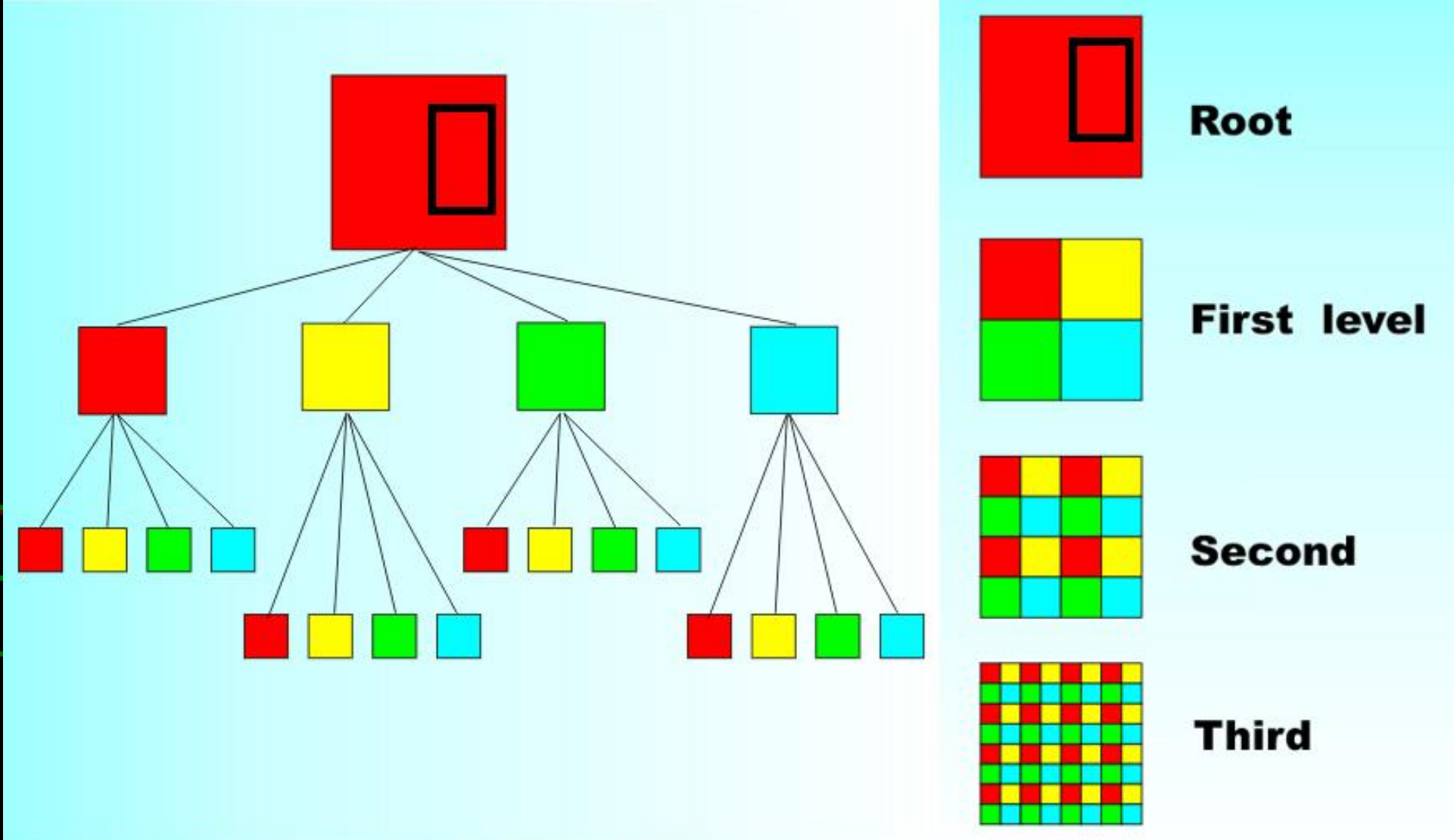


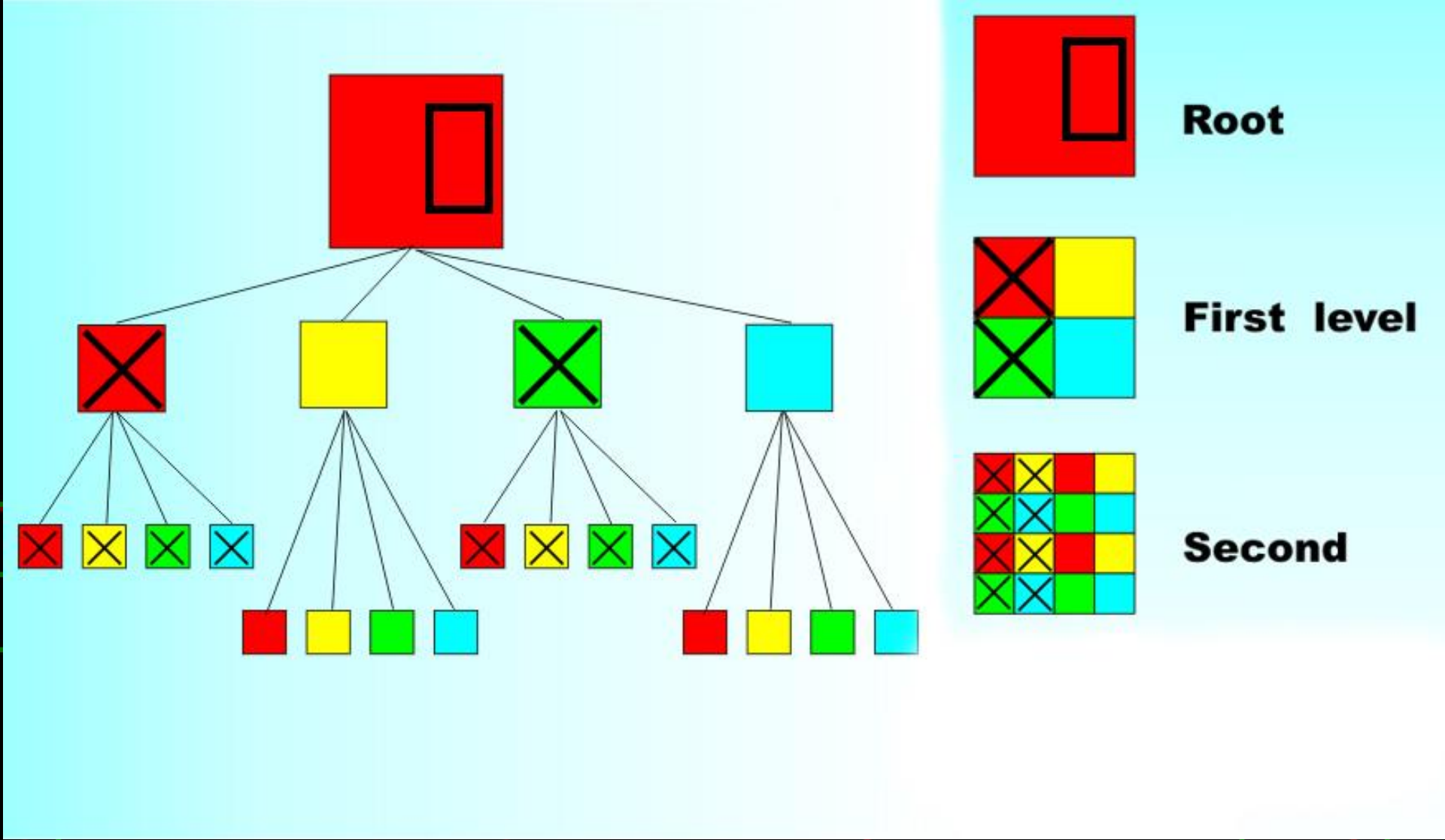
Why they are faster?

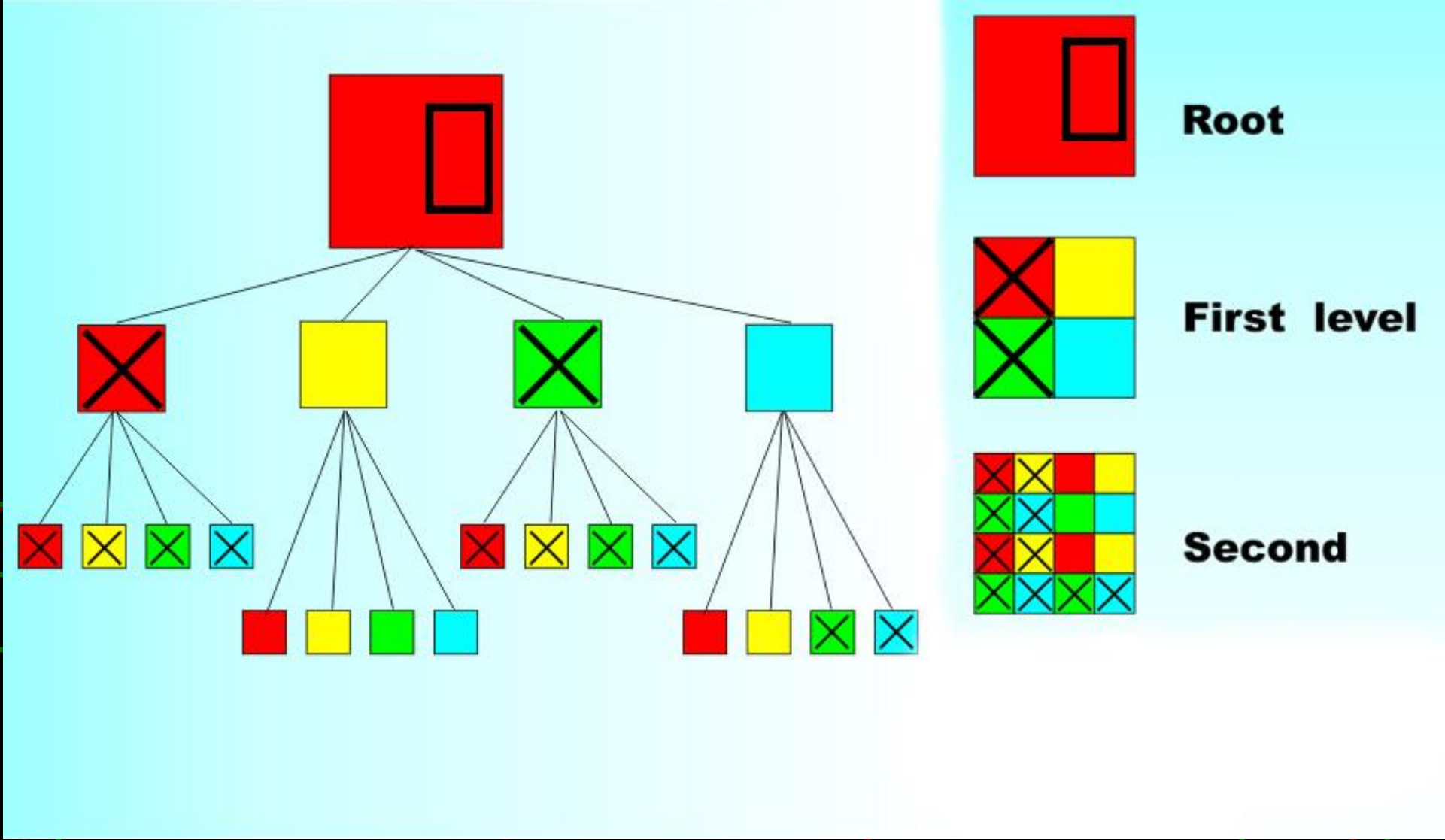






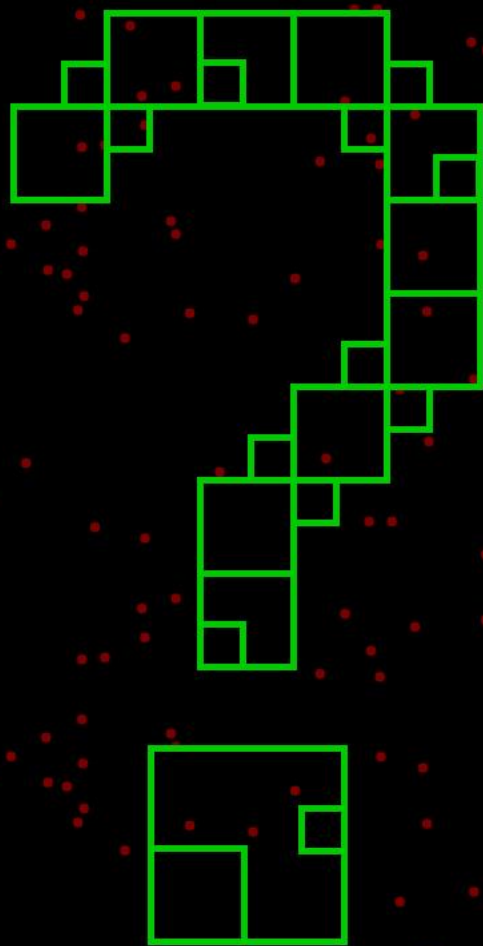















Any question?



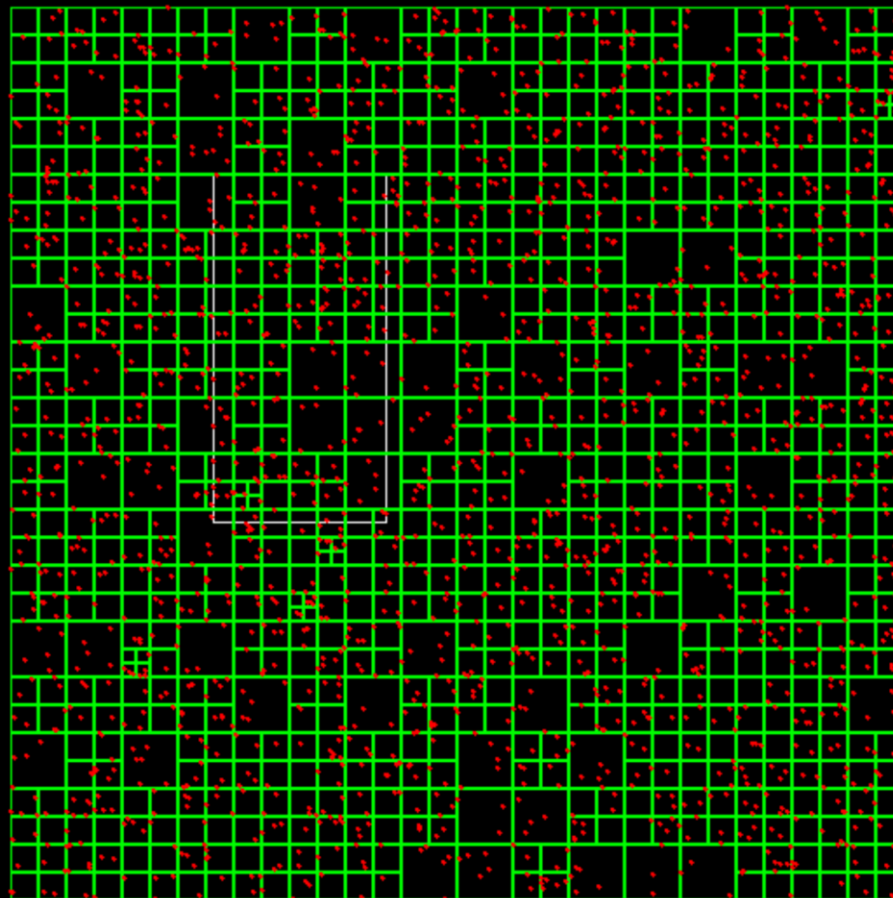
# Before TODOs

-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_second\_Iteration\_faster
-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_second\_Iteration\_faster\_with\_TODOs
-  Olivenza\_Xavier\_Research\_Release
-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_ENG
-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_ESP
-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_PPTX
-  Olivenza\_Xavier\_Research\_Optimized\_Search\_Manager\_PPTX

R,S,F1,F2,Q,A

# Before TODOs

Normal Search Time: 1.242694 ms, Points in Range = 162 | Quadtree Search Time: 0.471734 ms, Points in Range = 162 | Quadtree AABBs = 1164 | Quadtree checks = 165 | Accepted points 2000/2000 | Mesh ste... — □ ×

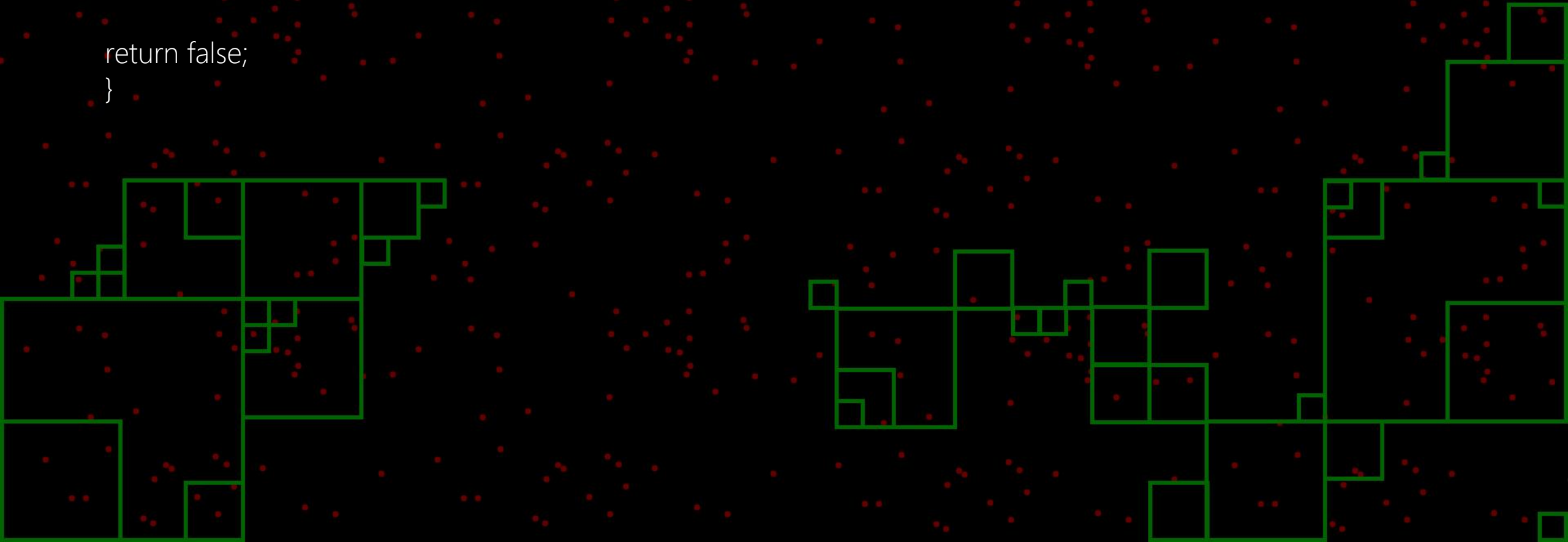


# TODOs wth the Quadtree implementation

Now get the release of my repository and  
proceed to do the TODOs in `SDLQuadtree.cpp`

# TODO 1

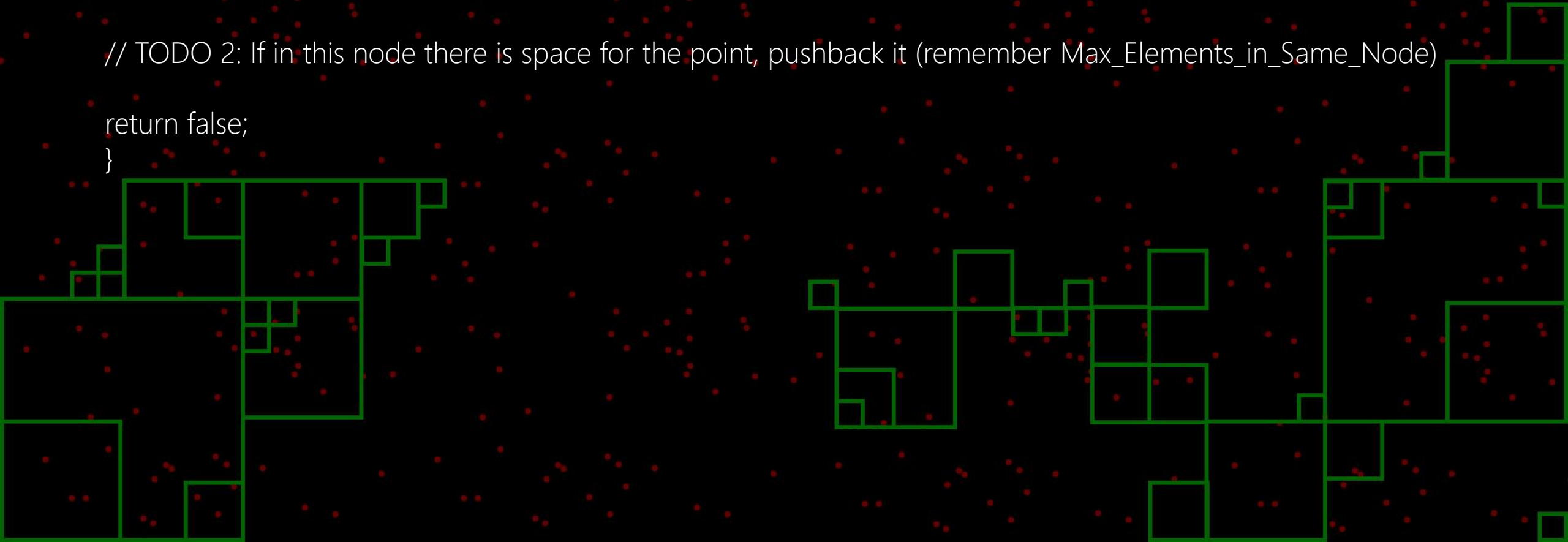
```
bool AABB::Insert(iPoint* newpoint)
{
    // TODO 1: If new point is not in the quadtree AABB, return
    return false;
}
```





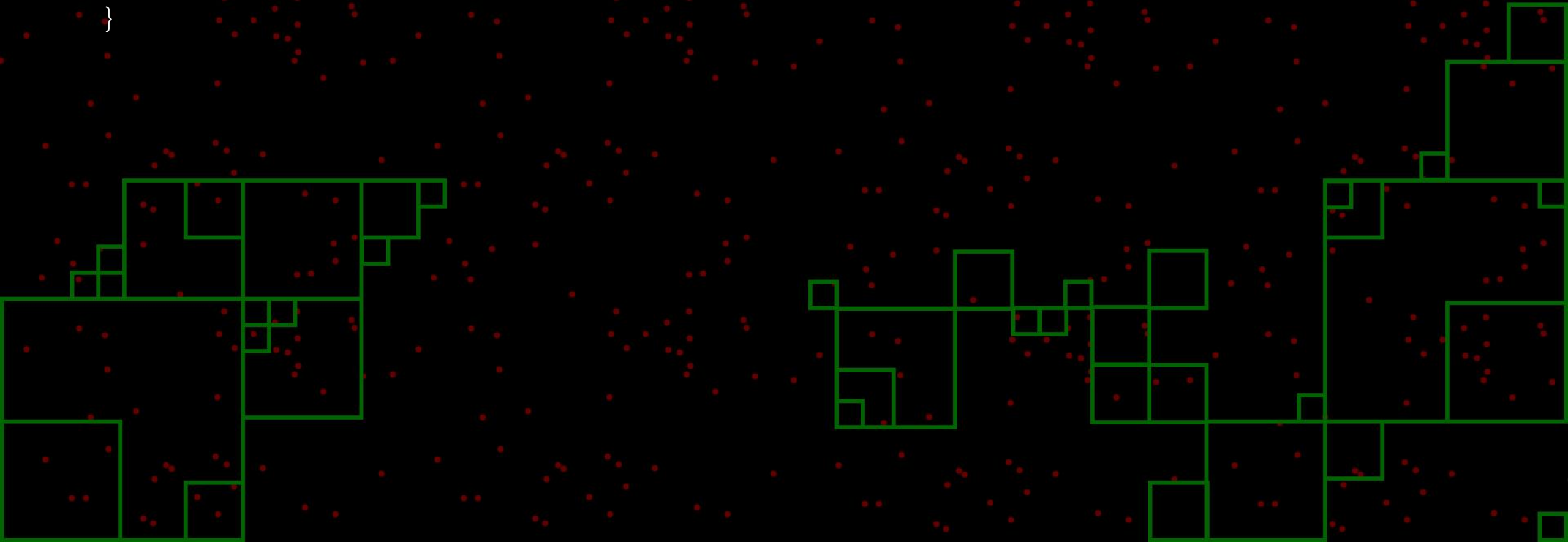
# TODO 2

```
bool AABB::Insert(iPoint* newpoint)
{
    // TODO 1: If new point is not in the quadtree AABB, return
    // TODO 2: If in this node there is space for the point, pushback it (remember Max_Elements_in_Same_Node)
    return false;
}
```



# TODO 3

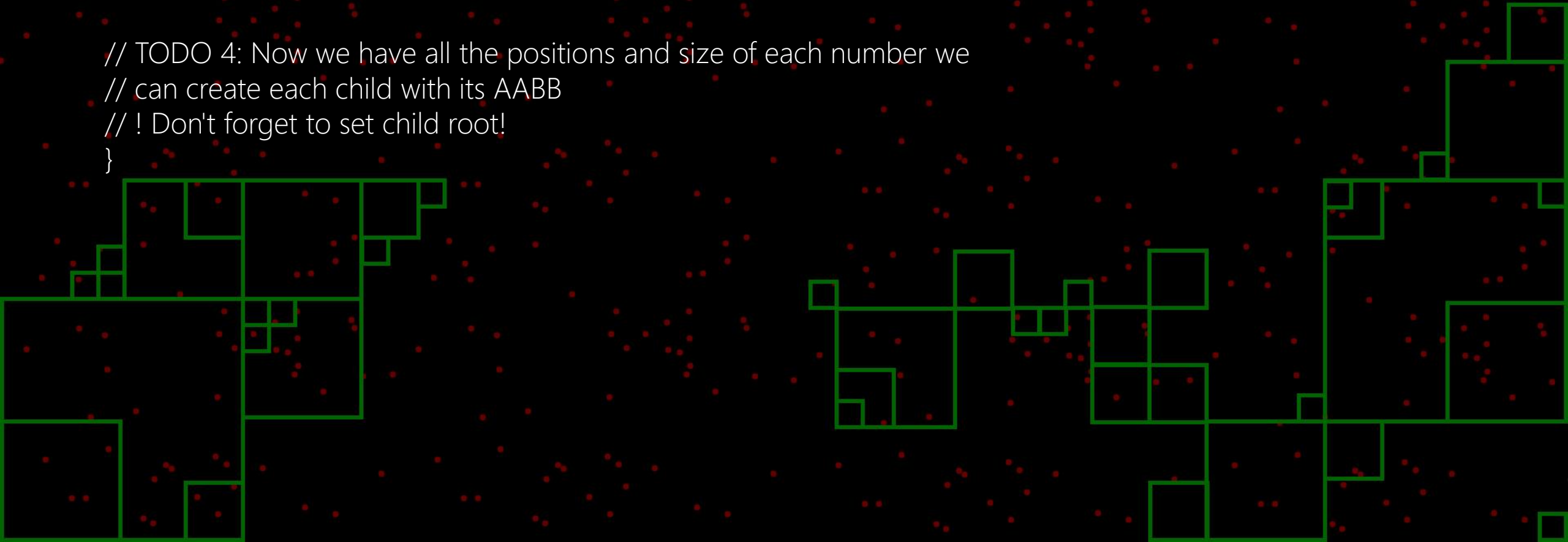
```
void AABB::subdivide()  
{  
    // TODO 3: Calculate the size and position of each of the 4 new nodes  
}
```



# TODO 4

```
void AABB::subdivide()
{
    // TODO 3: Calculate the size and position of each of the 4 new nodes

    // TODO 4: Now we have all the positions and size of each number we
    // can create each child with its AABB
    // ! Don't forget to set child root!
}
```



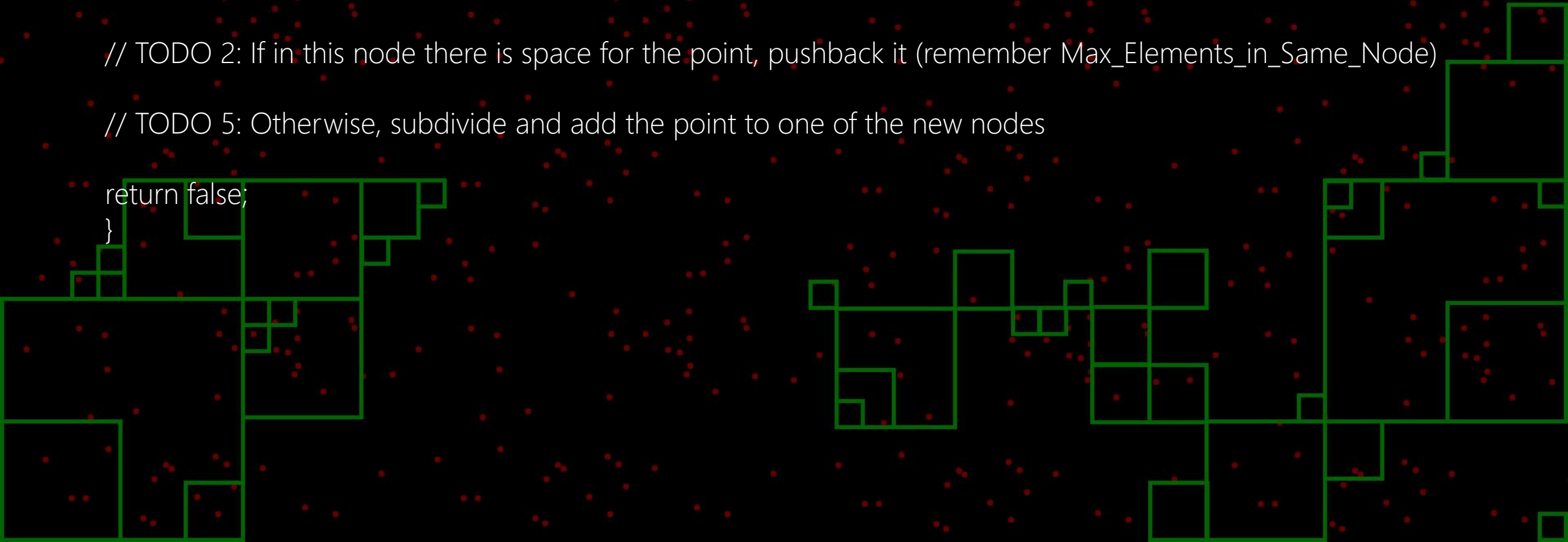
# TODO 5

```
bool AABB::Insert(iPoint* newpoint)
{
    // TODO 1: If new point is not in the quadtree AABB, return

    // TODO 2: If in this node there is space for the point, pushback it (remember Max_Elements_in_Same_Node)

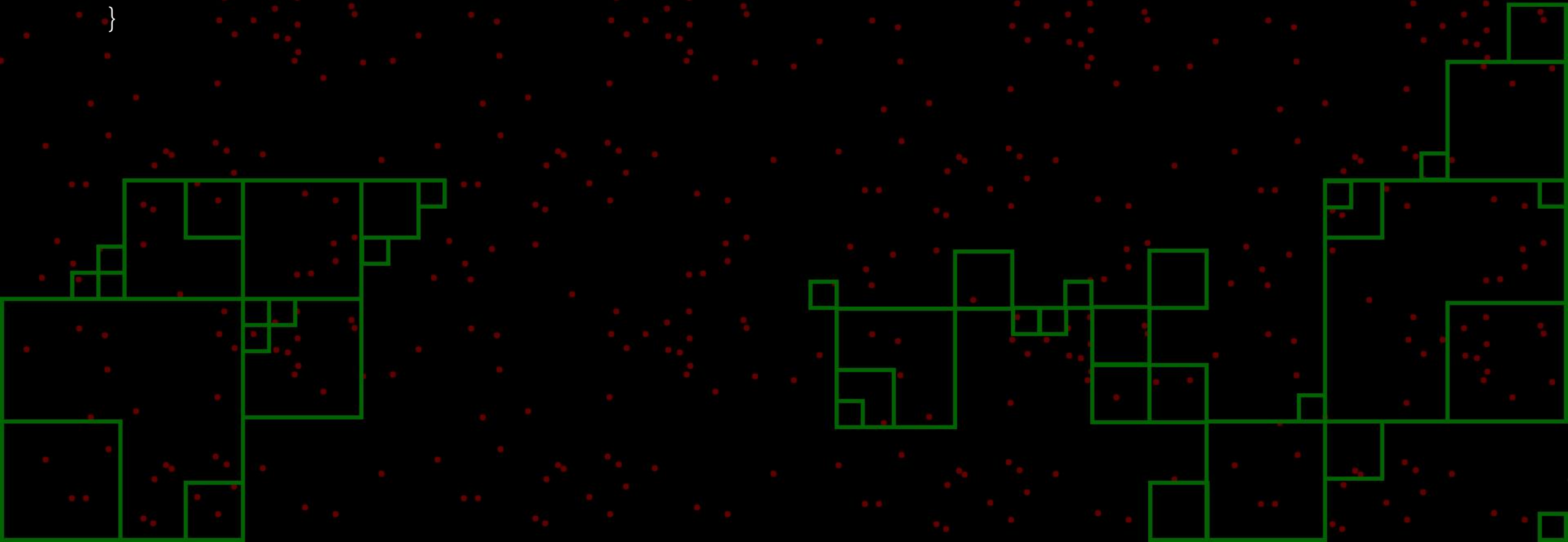
    // TODO 5: Otherwise, subdivide and add the point to one of the new nodes

    return false;
}
```



# TODO 6

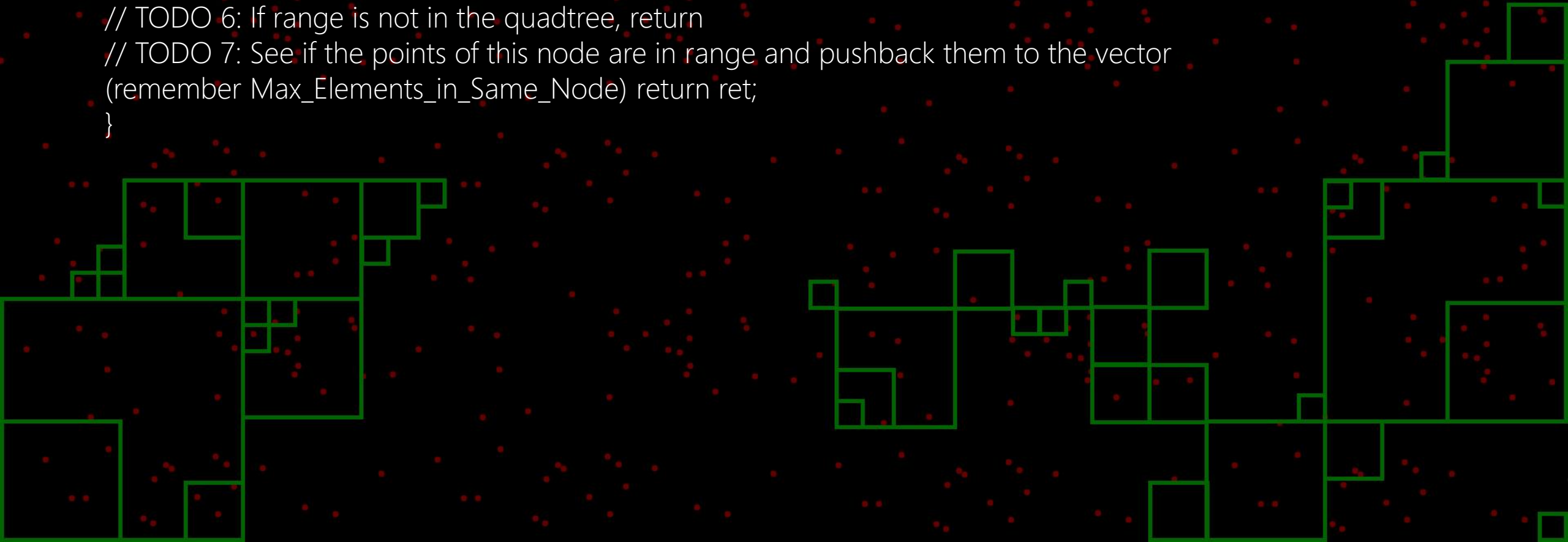
```
int AABB::CollectCandidates(std::vector< iPoint* >& nodes, const SDL_Rect& r)
{
    uint ret = 1;
    // TODO 6: If range is not in the quadtree, return return ret;
}
```





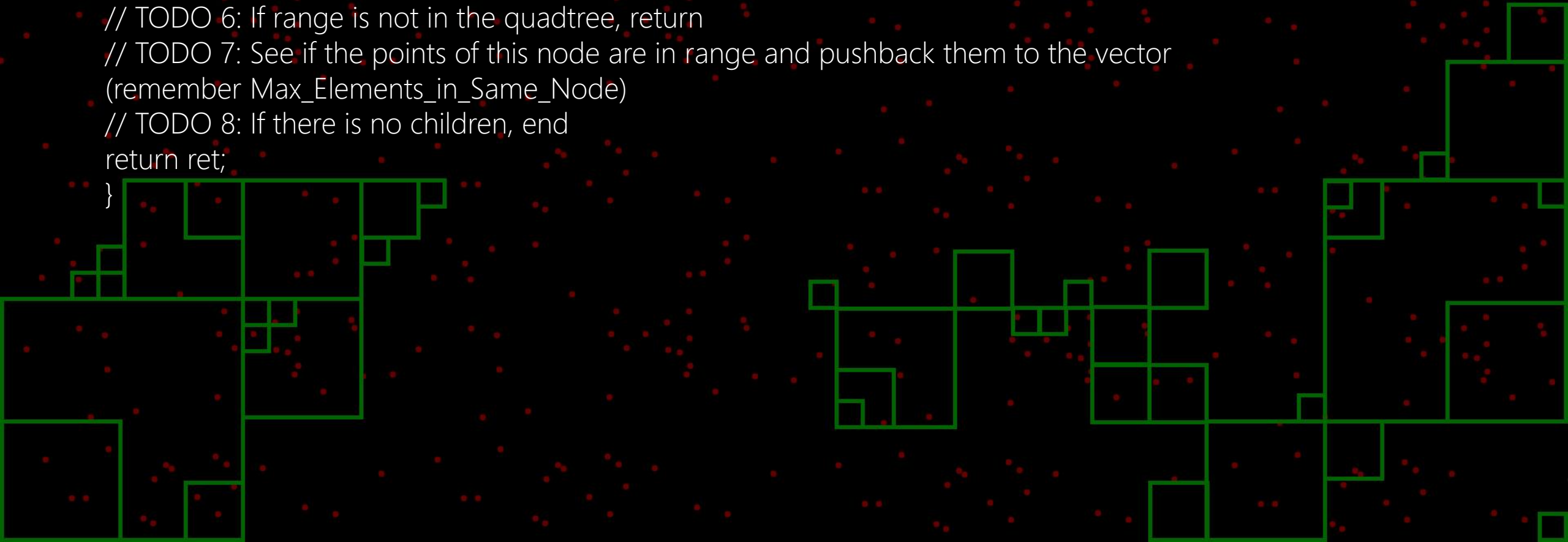
# TODO 7

```
int AABB::CollectCandidates(std::vector< iPoint* > & nodes, const SDL_Rect& r)
{
    uint ret = 1;
    // TODO 6: If range is not in the quadtree, return
    // TODO 7: See if the points of this node are in range and pushback them to the vector
    (remember Max_Elements_in_Same_Node) return ret;
}
```



# TODO 8

```
int AABB::CollectCandidates(std::vector< iPoint* > & nodes, const SDL_Rect& r)
{
    uint ret = 1;
    // TODO 6: If range is not in the quadtree, return
    // TODO 7: See if the points of this node are in range and pushback them to the vector
    (remember Max_Elements_in_Same_Node)
    // TODO 8: If there is no children, end
    return ret;
}
```

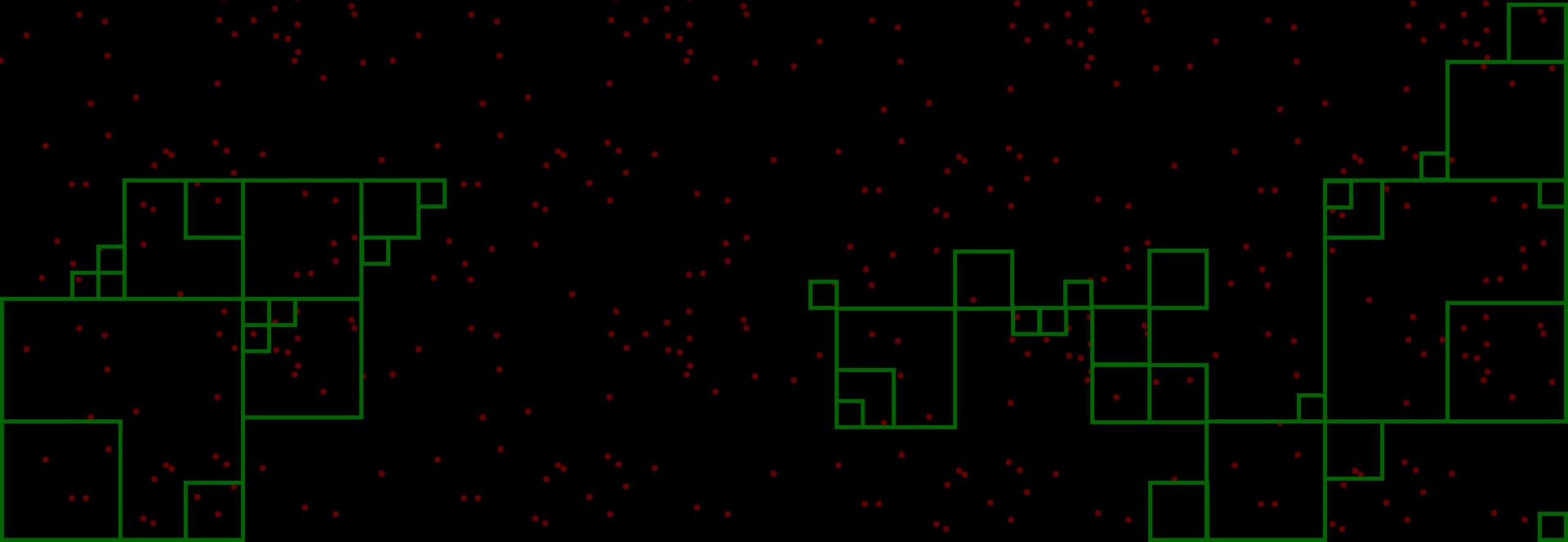


# TODO 9

```
int AABB::CollectCandidates(std::vector< iPoint* > & nodes, const SDL_Rect& r)
{
    uint ret = 1;
    // TODO 6: If range is not in the quadtree, return
    // TODO 7: See if the points of this node are in range and pushback them to the vector
    (remember Max_Elements_in_Same_Node)
    // TODO 8: If there is no children, end
    // TODO 9: Otherwise, add the points from the children (Recursive)
    return ret;
}
```

# Optional Homework

- Moving entities?  
Adapt the code for it.  
Maybe you will need a Remove method...



Any last question?

