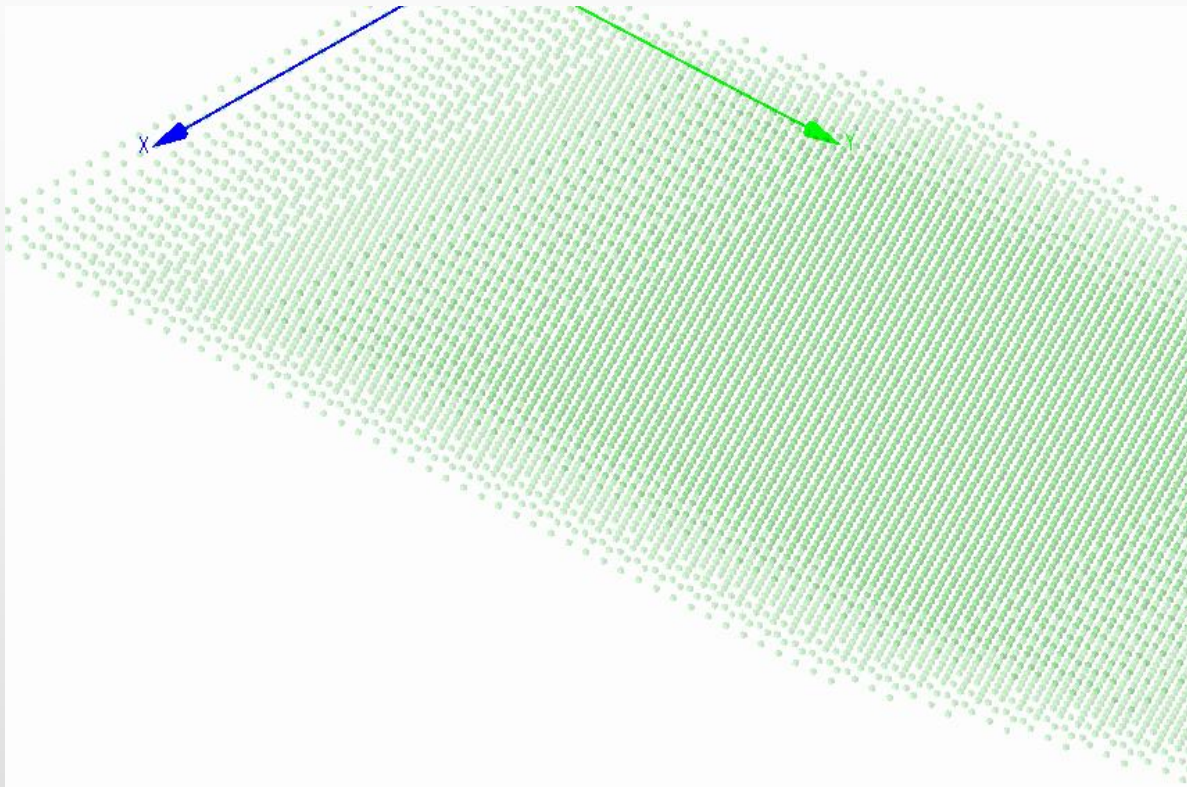


OCTree

1.Question

- One million points in a cube, How to find two points that distance is shortest?



2. Purpose


- Understand Octree Structure
- Use OCtree to solve real problem in project
- Create/Expand OCtree data structure

3. Octonary Tree / Octree

Octonary

英 / ˈɒktənəri / 

美 / ˈɑːktəˌneri / 

全球发音  >



简明

新牛津

柯林斯

例句

百科

adj. [数] 八进制的; 八的; 由八组成的

n. 八个一组; 八行诗

octree



简明

例句

百科

八叉树

3.1.OCtree Structure

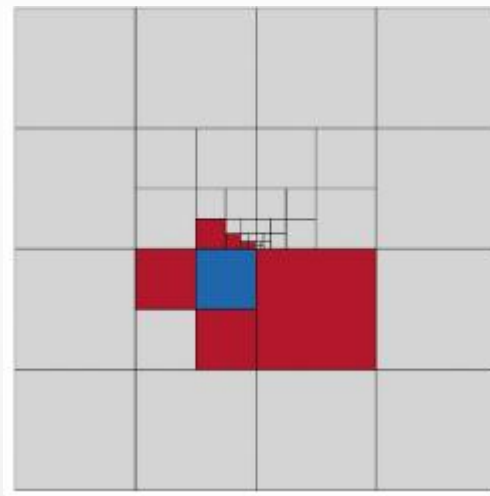
An octree is an object that represents a **spacial partitioning**.
It is made up by a **tree data** structure in which each node has exactly **eight children**.

3D space managment and acceleration

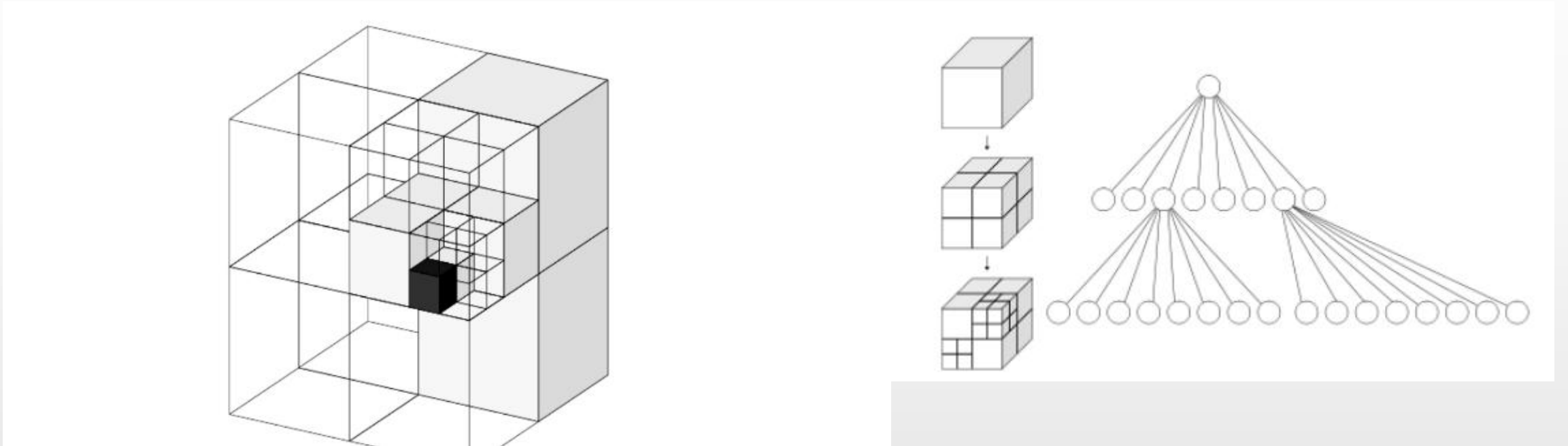
Massive objects in 3D space

3.1.OCtree Structure

- Quad Tree extension
- Reduce unnecessary test
- Partition [一篇文章入门仿真软件性能优化](#)



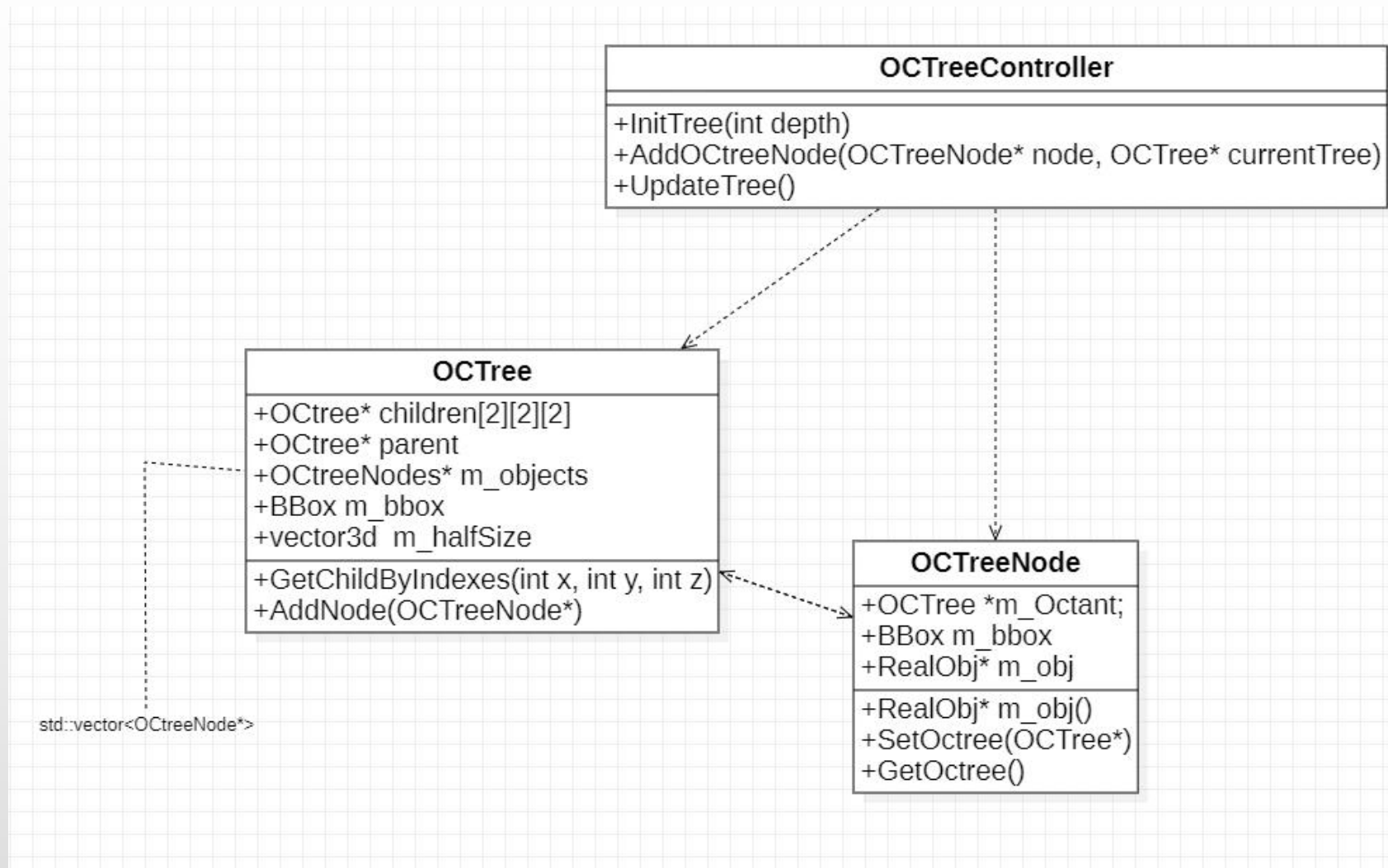
3.1. Octree Structure



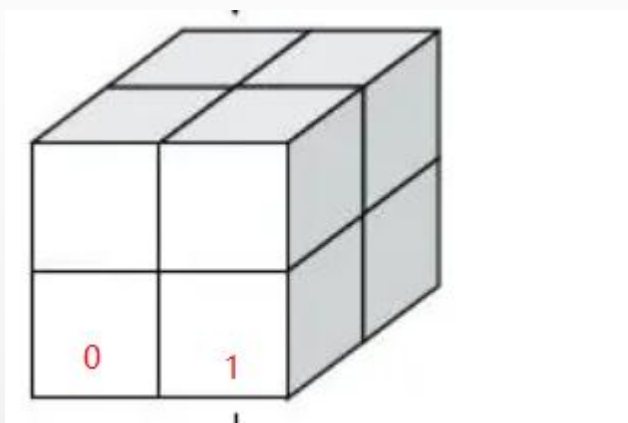
3.2.Implementation:major class

- OCTree
- OCTreeNode
- OCTreeController

3.2.Implementation: class diagram



Set sub-node arrangement



```
void OCTree::AddNode(OCTreeNode* currentNode)
{
    currentNode->SetOCTree(this);
    m_objects.push_back(currentNode);
}
```

```
if (x == 0)
{
    tmin.SetX(minData.X()); tmax.SetX((minData.X() + maxData.X()) / 2);
}
else
{
    tmin.SetX((minData.X() + maxData.X()) / 2); tmax.SetX(maxData.X());
}
```

```
void OctreeController::AddOctreeNode(OctreeNode* treeNode, Octree* currentOctant, int treeDepth)
{
    //Add more Filter //currentOctant->IsTwoSize(bx);
    const bbBox& bx = treeNode->GetBBox();
    if ((treeDepth < m_TreeMaxDepth) && currentOctant->IsTwoSize(bx))
    {
        int x, y, z;
        currentOctant->GetChildIndexes(bx, &x, &y, &z);
        if (currentOctant->m_children[x][y][z] == nullptr)
        {
            currentOctant->m_children[x][y][z] = new Octree();
            currentOctant->m_children[x][y][z]->setParent(currentOctant);

            const vector3d& minData = currentOctant->m_bbox.GetMinPoint();
            const vector3d& maxData = currentOctant->m_bbox.GetMaxPoint();
            vector3d tmin, tmax;
            if (x == 0) { ... }
            else { ... }
            if (y == 0) { ... }
            else { ... }
            if (z == 0) { ... }
            else { ... }
            currentOctant->m_children[x][y][z]->m_box.Set(min, max);
            currentOctant->m_children[x][y][z]->m_halfSize = (max - min) / 2;
        }
        AddOctreeNode(treeNode, octant->m_children[x][y][z].get(), ++depth);
    }
    else
    {
        currentOctant->AddNode(treeNode);
    }
}
```

Determine put which sub-octree

Create new Octree

put treeNode into current Octree

One API example

```
void OctreeController::FindIntersectNodes(const vector3d& startPt, const vector3d& endPt, std::list< OctreeNode* >& list, Octree* octant)
{
    bbBox obox;
    octant->GetBBox(&obox);
    bool isInter = obox.IsIntersect(startPt, endPt);
    if (!isInter)
    {
        return;
    }
    auto it = octant->m_nodeList.begin();
    while (it != octant->m_nodeList.end())
    {
        OctreeNode* on = (*it);
        if (on != nullptr)
        {
            bbBox box2Cal = on->GetBBox();
            bool nsect = box2Cal.IsIntersect(startPt, endPt);
            if (nsect)
                list.push_back(on);
        }
        ++it;
    }

    Octree* child;
    if ((child = octant->m_children[0][0][0].get()) != 0)
        OctreeFindNodesBy2Pt(startPt, endPt, list, child);

    if ((child = octant->m_children[1][0][0].get()) != 0)
```

```
        if (nsect)
            list.push_back(on);
    }
    ++it;
}

Octree* child;
if ((child = octant->m_children[0][0][0].get()) != 0)
    OctreeFindNodesBy2Pt(startPt, endPt, list, child);

if ((child = octant->m_children[1][0][0].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[0][1][0].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[1][1][0].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[0][0][1].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[1][0][1].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[0][1][1].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);

if ((child = octant->m_children[1][1][1].get()) != 0)
    FindIntersectNodes(startPt, endPt, list, child);
}
```

Reference

- [一篇文章入门仿真软件性能优化](#)
- [OGRE 场景管理](#)

- **OpenSource:**
 - OpenFOAM
 - OCC(BVH Bounding Volume Hierarchy)
 - VTK
 - OGRE

Homework

- Test OCtree performance

Q&A