doi: 10.3969/j.issn.1006-1576.2011.07.010

虚拟场景中路径自动选择算法

施斌[1]，刘惠义[2]，赵建平[1]，年福纯[1]
（1. 中国卫星海上测控部技术部，江苏 江阴 214431；2. 河海大学计算机及信息工程学院，南京 210098）

摘要：针对传统路径规划方法存在的问题，对机器人全局路径规划中的栅格建模方法进行改进。从场景建模描述、邻域查找以及路径搜索策略 3 个方面进行深入研究，采用线性八叉树法对场景进行建模，给出基于线性八叉树的层次编码体系，引入路径搜索因子对启发式函数进行重构。通过基于八叉树的场景分解，基于线性八叉树编码特性的邻域查找，以及改进的 A*算法路径搜索，实现了三维场景路径规划和自动漫游。实验结果表明：该算法的时间和空间效率较好。

关键词：虚拟场景；线性八叉树；路径规划；A*算法
中图分类号：TP391.9 **文献标志码**：A

An Algorithm on Path Planning in Virtual Scene

Shi Bin[1], Liu Huiyi[2], Zhao Jianping[1], Nian Fuchun[1]
(1. Technology Department, Satellite Maritime Tracking & Controlling Department of China, Jiangyin 214431, China;
2. College of Computer Science & Information Engineering, Hohai University, Nanjing 210098, China)

Abstract: To solve the problems existing in those traditional path planning methods, the algorithm improves grid modeling method for global robotics path planning. Deep researches on scene modeling description, neighbor search and path search strategy are made. First, the linear octree method is applied to scene modeling. Second, hierarchical coding system based on linear octree is presented. Third, heuristic function is reconstructed by introducing path planning factor. In short, by hierarchical decomposition of scene based on octree, neighbor search based on linear octree codes, and improved A* algorithm, this algorithm can complete path planning and automated navigation in 3D virtual scene. The experiment results indicate that the algorithm is good in storage and time efficiency.

Keywords: virtual scene; linear octree; path planning; A* algorithm

0 引言

路径规划一直是机器人学中重要的研究领域，广泛应用于移动机器人的避碰、导航，机械臂运动以及机器人足球等热点领域。将路径规划算法应用到虚拟现实中，实现虚拟场景的三维路径规划有重要的理论意义和实用价值。由于虚拟场景中所有物体的信息对系统来说都是已知的，相比机器人学路径规划问题，虚拟场景的路径规划有其特殊性[1]。

目前，路径规划领域有很多典型的方法，部分算法也扩展到了三维虚拟场景路径规划问题。在国外，Salomon 等[2]提出了基于路径图(roadmap)的概率路径图(probabilistic roadmap，PRM)算法，可以进行交互或自动路径规划，但该算法效率易受采样粒度影响，不够稳定。Srikanth Bandi 等[3]提出了基于精确栅格法的简单林区着火(simple bushfire)路径规划算法，能扩展应用到多目标以及动态场景的复杂 3D 环境下，但其效率受场景复杂程度的影响。J Vörös[4]提出了通过构造基于非精确栅格的树型结构(四叉树或八叉树)的距离图(distance map)来搜索获取合理的路径算法，效率较高，但只能够实现简单场景的路径规划。在国内，史红兵等[5]尝试了将机器人的运动路径规划算法和摄像机控制的美学标准结合起来，采用基于八叉树的环境模型层次分解建立连接图，然后利用 IDA*算法搜索最优路径，最后对运动轨迹进行了优化，能应用在较复杂的实际三维虚拟场景中。由畅宇等[6]提出了一种基于遗传算法的三维路径规划方法，通过对机器人的运动行为进行编码，从而将路径规划问题转化为行为编码的遗传进化问题，可以规划出合理可行的运动轨迹，但该算法存在容易陷入 U 型障碍物空间的问题。刘利强等[7]以蚁群算法中的 ACS 算法为基础设计了一种路径优化搜索算法，能够方便有效地解决水下潜器三维空间路径规划问题，但该算法存在运行时间较长、规划出的航线非最优航线的问题。综合国内外研究现状来看，三维虚拟场景规划算法还比较少，部分算法存在效率不稳定的问题，且大多没有应用于实际的复杂三维虚拟场景，仅停留在理论模型的探讨上。

因此，笔者围绕路径规划过程中的场景建模描

收稿日期：2011-03-10；修回日期：2011-05-26
作者简介：施斌(1982—)，男，江苏人，硕士，工程师，从事计算机图形学、智能虚拟环境研究。

述、邻域查找以及路径搜索策略3个问题进行研究。

1 基于八叉树的场景分解

三维场景的模型描述是三维路径规划中很重要的一步，直接影响后续问题搜索的规模，从而影响路径规划的整体效率。非精确的空间分解法能够充分利用空间障碍物密度信息，灵活地描述障碍物分布不均匀的场景，大大减少描述场景所需单元的数目，从而有效减少了问题的规模，八叉树法就是空间分解法的典型方法。

1.1 决定分解的判断标准

八叉树的叶子节点单元有障碍节点和自由节点2种。在节点单元区域内，如果包含障碍物，则称为障碍节点，如果不包含任何障碍物，则称自由节点，如果未达到分解终止条件，需要进一步分解。最终，整个场景就由障碍节点和自由节点构成。

为了描述节点的分解过程，设定2个链表object、vex分别表示八叉树单元中容纳的模型对象和模型顶点，density表示单元内顶点数，max_depth表示最大树深。对于任意一个八叉树单元节点 o，分解判断过程用伪码表示如下：

```
if(o->density == 0)     //单元内无顶点
{
    if(o->object.IsEmpty())
    //单元内模型对象链表为空
    {
        o->flag = EMPTY;     //自由节点
        //记录并存储节点信息
    }
    else       //单元内模型对象链表非空
    {
        o->flag = FULL;       //障碍节点
    }
}
else        //单元内有顶点
{
 if(o->density > MAX_DENSITY)
 //单元内顶点密度超过密度阈值 MAX_DENSITY
    {
    if(depth < max_depth) //未达最大树深，分解
     {
       //节点空间范围分割
       for(int oc = 0; oc < 8; oc++)
        {
        //判断子节点分解情况并记录子节点信息
        }
       //子节点空间释放
     }
    else        //达到最大树深，终止分解
     {
       o->flag = FULL;              //障碍节点
     }
    }
 else //单元内顶点密度未超过密度阈值 MAX_DENSITY，终止分解
    {
      o->flag = FULL;              //障碍节点
    }
}
```

密度阈值 MAX_DENSITY 由经验和实际效果决定。笔者采用计算顶点数目方式确定分解阈值，精确程度可以达到要求，效率较高。

1.2 基于八叉树的场景分解

八叉树的根节点对应于整个场景立方体空间，其确定方法是：遍历场景中所有对象的坐标值，获得模型的空间范围，然后以场景中心点为中心，以 X、Y、Z 轴最大跨度为边长，将顶点区域扩展为立方体区域。

场景八叉树建模的过程即根据 1.1 节分解的判断标准进行场景八叉树分解的过程：从根节点开始，如果它是障碍节点，将该节点标记为 FULL；如果它是自由节点，将该节点标记为 EMPTY；如果它是待分割节点，将它分割成8个子立方体节点，对每一个子立方体节点进行同样的处理。

最后，以链表的形式记录下所有的自由节点单元，作为后续邻域查找和路径搜索的基础。

2 基于线性八叉树的邻域查找

八叉树的邻域查找的效率高低也是影响路径规划整体效率的重要因素之一。很多的邻域查找算法都比较复杂，不便于编程实现。笔者借鉴肖乐斌等[8]提出的一种邻域搜索的思想，应用于基于线性八叉树的空间邻域查找过程中，充分利用线性编码蕴含的层次、大小和方向特性，直接计算节点邻域编码，能够较快查找到邻域，算法复杂程度适中，便于编程实现。

2.1 线性八叉树的编码体系

在基于八叉树的场景分解的同时，需要对所有八叉树节点进行编码。线性八叉树[9-10]的编码基准

体系的划分，如图 1。

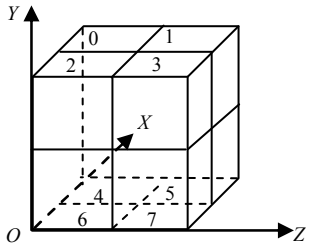


图 1　线性八叉树的编码体系

线性八叉树的编码（八进制）体系可以定义为：

北部 N = {0, 1, 4, 5}；南部 S = {2, 3, 6, 7}；

西部 W = {0, 2, 4, 6}；东部 E = {1, 3, 5, 7}；

上部 U = {0, 1, 2, 3}；下部 D = {4, 5, 6, 7}。

2.2　基于线性八叉树的邻域查找算法

基于线性八叉树的邻域查找按以下流程进行：首先，排除不可能存在的邻域；然后，判断是否存在相同尺寸的邻域；最后，进一步确定是否存在不同尺寸的邻域。

1) 排除不可能存在的邻域

对于任意一个八叉树单元，编码为 $M=Q_1Q_2\cdots Q_n$，其中(i =1, 2, …, n)为八进制数。判断该单元的某个方向的邻域是否存在，其规则如下：如果对于任意的 Q_i，$Q_i\in$ N，则对应的 N 邻域不存在。同理，对于 $Q_i\in$ S，$Q_i\in$ E，$Q_i\in$ W，$Q_i\in$ U 和 $Q_i\in$ D 的情况，对应的 S，E，W，U，D 邻域也不存在。

判断出不可能存在的邻域后，这些邻域就无需计算，提高了查找效率。

2) 相同尺寸邻域的确定

首先，确定水平方向的东、南、西、北 4 个方向的邻域。

① 如果编码末位 Q_n=0，则可直接计算得到其东部邻域为 M+1，南部邻域为 M+2，西部邻域的编码值可以用式 (1) 表示：

$$M_W = M - (8^{n-j} - \sum_{k=0}^{n-j-1} 8^k) \tag{1}$$

北部邻域的编码值可以用式 (2) 表示：

$$M_N = M - (2\times 8^{n-j} - \sum_{k=0}^{n-j-1} 2\times 8^k) \tag{2}$$

② 如果 Q_n=1，则西部邻域为 M−1，南部邻域为 M+2，东部邻域的编码值可以用式 (3) 表示：

$$M_E = M + (8^{n-j} - \sum_{k=0}^{n-j-1} 8^k) \tag{3}$$

北部邻域的编码值可以用式 (2) 表示。

③ 如果 Q_n=2，则北部邻域为 M−2，东部邻域为 M+1，南部邻域的编码值可以用式 (4) 表示：

$$M_S = M + (2\times 8^{n-j} - \sum_{k=0}^{n-j-1} 2\times 8^k) \tag{4}$$

西部邻域的编码值也可以用式 (1) 表示。

④ 如果 Q_n=3，则西部邻域为 M−1，北部邻域为 M−2，南部邻域的编码值可以用式 (4) 表示，东部邻域的编码值可以用式 (3) 表示。

⑤ Q_n 为 4，5，6，7 的邻域的计算方法与 0，1，2，3 的计算方法完全相同。

其次，确定垂直方向的上下邻域。

如果 Q_n 属于上部集合，下部邻域为 M+4，上部邻域的编码值可以用式 (5) 表示：

$$M_U = M - (4\times 8^{n-j} - \sum_{k=0}^{n-j-1} 4\times 8^k) \tag{5}$$

如果 Q_n 属于下部集合，上部邻域为 M−4，下部邻域的编码值可以用式 (6) 表示：

$$M_D = M + (4\times 8^{n-j} - \sum_{k=0}^{n-j-1} 4\times 8^k) \tag{6}$$

最后，通过遍历场景八叉树自由节点单元链表，检验求出的相同尺寸邻域编码在链表中是否存在对应的节点单元。若存在，则该节点单元即为相同尺寸邻域；否则，不存在相同尺寸邻域。

3) 不同尺寸邻域的确定

由于线性八叉树分解的自适应性，某方向上的相同尺寸邻域不存在，却可能存在更大或更小尺寸的邻域。不同尺寸邻域的确定是以相同尺寸邻域为基础的。相同尺寸邻域记为 O，假设不同尺寸邻域存在，不同尺寸邻域按如下规则确定：

① 产生 O 的各级祖先节点的编码，即截取 O 的编码的前缀，长度从 1 位到 O 的编码长度减 1 位。遍历场景八叉树自由节点单元链表，依次检验各级祖先节点编码在链表中是否存在对应的节点单元。若存在，即找到不同尺寸邻域，记录下该节点单元，终止查找，否则转②；

② 产生 O 的各级子孙节点的编码，即在 O 的编码上加上后缀，总长度从 1 位到最大编码长度。后缀编码位的数值必须取与所求邻域方向相反方向的集合中的数值。依次检验各级子孙节点编码在链表中是否存在对应的节点单元。若存在，即找到不同尺寸邻域，并记录下所有经检验存在的节点单元。

最终，经过①和②检验找不到存在的节点单元，则不存在不同尺寸邻域。如果某方向的相同和不同尺寸邻域均不存在，则该方向不存在邻域。

3 基于改进的 A*算法的路径搜索

启发式搜索能充分利用问题定义提供的信息进行搜索，效率较高。A*算法[11]就是典型的启发式搜索算法。三维场景分解后产生了障碍节点和自由节点，其中全体自由节点的集合即 A*算法要搜索的范围。

3.1 规划问题的描述

三维场景的全局路径规划的主要任务即在一个指定起点 S 和终点 G 的三维场景中，利用 A*算法搜索出一条从 S 到 G 的无碰撞最短路径，并实现沿路径的自动漫游。为了简化问题的复杂性，特作以下假设：

1) 场景模型中包括的建筑物对象就是场景的障碍物，其位置和大小范围已知，并且在规划和漫游过程中其位置和大小均不发生变化。

2) 不考虑虚拟对象(机器人或虚拟替身)形状和大小，将虚拟对象缩为一个点。

3.2 启发式函数的设计

搜索路径的代价一般都是由节点间的距离来衡量的。Euclidean 距离是实际距离的真实反映，有清晰的物理意义，精确度高。笔者采用 Euclidean 距离对路径代价进行估计。

对于启发式函数式 $f(n)=g(n)+h(n)$，$f(n)$表示从起点 S 经过中间节点 n 到终点 G 的距离估计，$g(n)$表示起点 S 到达中间节点 n 时已通过的路径距离，$h(n)$ 表示从中间节点 n 到终点 G 将要通过的路径距离的估计。

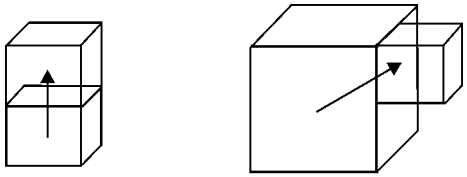
3.3 路径影响因子的引入

基于 3.2 节定义的启发式函数实现 A*搜索时，要考虑虚拟场景的特殊性，防止选择不合理的空中路径的情况。选择空中路径有 2 种情况：一是在扩展后继节点时选择了垂直方向相同尺寸的上部邻域节点，如图 2(a)；二是在扩展后继节点时选择水平方向不同尺寸的上部邻域节点，如图 2(b)。

解决方法是：分别增加选择垂直方向相同尺寸上部邻域节点和水平方向不同尺寸上部邻域节点的代价。即在计算 $g(n)$时引入路径影响因子 w，得到式(7)：

$$g(n)=g(m)+w\times d(m,n) \tag{7}$$

式中：m 和 n 分别代表前后相邻的 2 个节点，$d(m,n)$表示 m 和 n 之间的距离值。在上述 2 种选择空中路径的情况下，将 w 设为通常情况下的 2 倍或多倍，以达到惩罚的目的，从而避免选择不合理的空中路径。



(a) 垂直方向相同尺寸的上部邻域　(b) 水平方向不同尺寸的上部邻域

图 2 选择空中路径的 2 种情况

对于路径起点 S 和终点 G 分处于不同的楼层时，要将整个路径进行分段，楼梯部分路径专门设为一段，路径影响因子 w 要设为正常值，允许楼梯部分选择空中路径。分段求解后，再进行路径合成，即可解决虚拟场景的三维路径规划问题。

4 实验结果

使用 Visual C++ .NET 和 OpenGL 图形库，在 AMD Athlon 1800+ 1.54 GHz、内存 512 M 的计算机上，基于本算法构建了一个三维虚拟场景下的路径规划和自动漫游的实验原型。

实验中指定路径起点 S 和终点 G 的三维坐标，S ={10.0, −1.0, 9.0}，G={15.0, −1.0, 30.0}。图 3～图 6 为算法的实验结果。图 3 为三维场景最大分解次数(即最大树深)为 8 的八叉树分解效果；图 4 为三维路径规划的结果；图 5(a)表示未引入路径影响因子 w 时，出现不合理的空中路径；图 5(b)是引入路径影响因子 w 的路径搜索结果，避免了不合理的空中路径；图 6 所示为自动漫游实验效果。图 4～图 6 中，连续的立方体框代表路径经过的八叉树自由节点单元，折线代表搜索得到的路径。

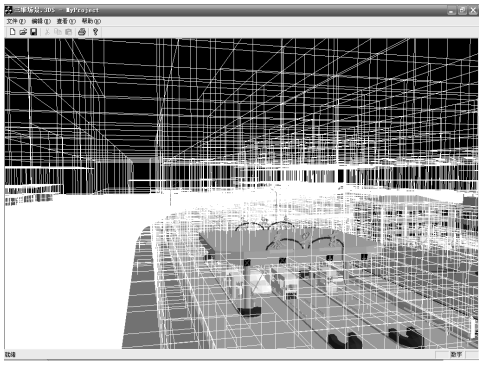


图 3 最大分解次数为 8 的基于八叉树的场景分解效果

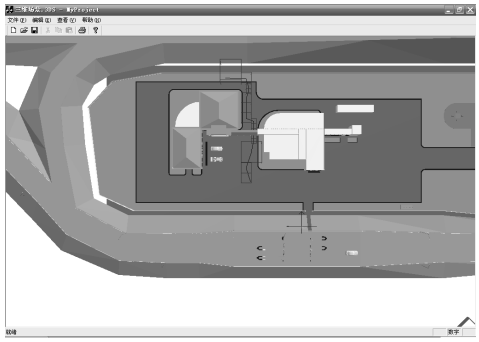
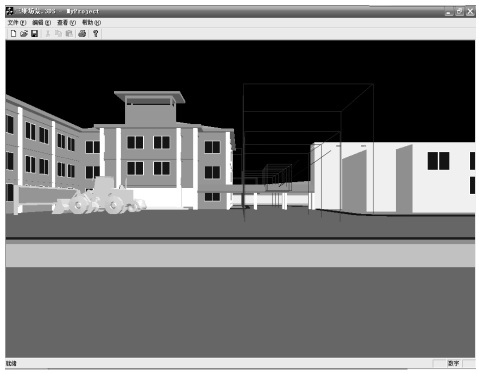
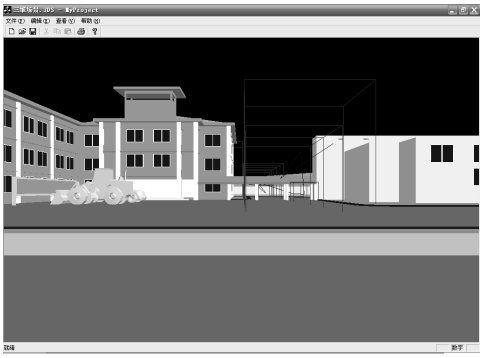


图 4　三维场景路径规划的结果



(a)　未引入路径影响因子 w 的路径搜索结果



(b)　引入路径影响因子 w 的路径搜索结果

图 5　路径搜索结果

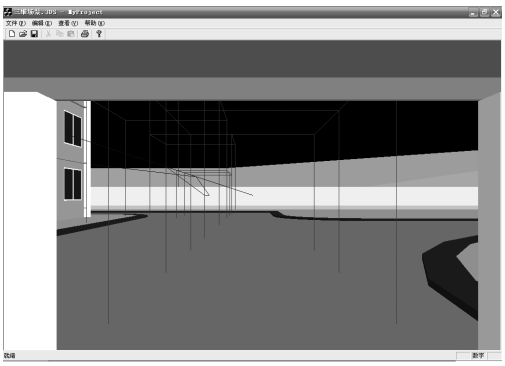


图 6　第一人称自动漫游的中间过程效果

实验原型中的场景模型包含 508 个模型对象，48 235 个顶点，100 764 个三角面。经过分解共产生 10 767 个八叉树节点，其中 7 462 个自由节点。在上面的实验中，基于八叉树的场景分解过程耗时约 8 s，基于 A*算法的搜索过程耗时约 12 s，整体路径规划算法总耗时约 20 s，内存消耗约 16 820 k。实验结果验证了本算法的正确性和有效性。

5　结论

在场景建模描述方面对机器人路径规划中的栅格建模方法进行改进，采用线性八叉树法对场景进行建模，大大减少了描述虚拟场景的单元节点个数。在邻域查找方面，给出了基于线性八叉树的层次编码体系，提高了邻域查找效率。在路径搜索策略方面，对 A*算法进行改进，引入路径搜索因子，避免选择不合理的空中路径。通过缩减场景节点数量、提升邻域查找效率及利用场景距离信息搜索，提高了路径搜索的效率，实现了三维场景路径规划及自动漫游。仿真实验结果表明：该算法正确、有效，算法的时间、空间效率较好，具有良好的应用前景。

参考文献：

[1] 陈勇, 王栋, 陈戈. 一种三维虚拟场景自动漫游的快速路径规划算法[J]. 系统仿真学报, 2007, 19(11): 2507-2510.

[2] Brian Salomon, Maxim Garber, Ming C.Lin, Dinesh Manocha. Interactive Navigation in Complex Environments Using Path Planning[C]. Proceedings of Symposium on Interactive 3D Graphics. Monterey, California: 2003: 41-50.

[3] Srikanth Bandi, Daniel Thalmann. Path finding for human motion in virtual environments[J]. Computational Geometry, 2000(15): 103-127.

[4] Jozef Vörös. Low-cost implementation of distance maps for path planning using matrix quadtrees and octrees[J]. Robotics and Computer Integrated Manufacturing, 2001(17): 447-459.

[5] 史红兵, 张毅彬, 童若锋, 等. 虚拟场景自动漫游的路径规划算法[J]. 计算机辅助设计与图形学学报, 2006, 18(4): 592-597.

[6] 由畅宇, 韩建达. 一种移动机器人在三维动态环境下的路径规划方法[J]. 计算机应用与软件, 2009, 26(8): 223-224.

[7] 刘利强, 于飞, 戴运桃. 基于蚁群算法的水下潜器三维空间路径规划[J]. 系统仿真学报, 2008, 20(14): 3712-3716.

[8] 肖乐斌, 龚建华, 谢传节. 线性四叉树和线性八叉树邻域寻找的一种新算法[J]. 测绘学报, 1998, 27(3): 195-203.

[9] David F.Rogers. 计算机图形学的算法基础[M]. 北京: 机械工业出版社, 2002.

[10] 彭群生, 鲍虎军, 金小刚. 计算机真实感图形的算法基础[M]. 北京: 科学出版社, 1999.

[11] Stuart Russell, Peter Norvig. 人工智能: 一种现代方法[M]. 第 2 版. 北京: 人民邮电出版社, 2004.