(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 104462764 A
(43)申请公布日 2015.03.25

(21)申请号 201410615853.6

(22)申请日 2014.11.06

(71)申请人 中国人民解放军第二炮兵工程设计研究所
地址 100011 北京市东城区安德里北街18号
申请人 南开大学

(72)发明人 费允锋 田庆龙 宋银灏 孙广毅 李吉 孙沁璇 张晨峰 谭可可 刘瑞萍 李国文 耿彤 常正阳 王聪 李伟 占金春

(74)专利代理机构 中国人民解放军第二炮兵专利服务中心 11040
代理人 李丽梅

(51)Int.Cl.
G06F 19/00(2011.01)

权利要求书1页 说明书5页 附图2页

(54)发明名称

一种虚拟现实仿真中物体表面间最小距离的快速计算方法

(57)摘要

本发明属于一种虚拟仿真技术领域，特别涉及一种虚拟模型空间距离计算方法。一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征包括以下步骤：A、将虚拟物体模型离散成顶点和多边形面片的形式，标记各顶点的序号及多边形面片对应顶点索引信息；B、构建树型空间结构遍历后找出两模型间的最近顶点对，并计算该最近点对在欧式空间直线距离；还包括C、在两个模型上分别找出各自最近顶点所在的全部多边形面片；D、分别计算两个模型上最近顶点到对方模型上最近顶点所在全部多边形面片的距离；E、重复步骤C和D，找出最小值即为两物体间的最小间隙。本发明加速了最近顶点的搜索过程，提高了估算精度，加强了对不同类型物体模型的适应性。

CN 104462764 A

1. 一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是包括以下步骤：

步骤 1. 将虚拟物体模型 A 和 B 离散成顶点和多边形面片的形式，并标记模型中各顶点的序号以及多边形面片对应的顶点索引信息；

步骤 2. 对模型 B 的所有顶点，构建树型空间结构，以实现三维空间的二分查找树的数据存储结构；

步骤 3. 对于模型 A 上的每一个点 a_i，在欧式空间中遍历步骤 2 所述的模型 B 的树型空间结构中各节点，并在其中找到距 a_i 最近的点 p_i，并计算 a_i 与 p_i 之前的直线距离 d_i；

步骤 4. 遍历步骤 3 中得到的所有距离 d_i，找出最小值 d_{min}，并记录最小值 d_{min} 对应的顶点 a_i 和 p_i，则最小值 d_{min} 为两个模型的最近距离的初步估计值，对应的顶点 a_i 和 p_i 为初步估计的两个模型的最近点。

2. 根据权利要求 1 所述的一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是还包括以下步骤：

步骤 5. 在模型 B 上确定顶点 p_i 所在的全部 n 个多边形，分别在每个多边形面上找到步骤 3 所述的距离模型 A 上顶点 a_i 最近的点 $q_1, q_2 \cdots q_n$，并分别计算出 a_i 到 $q_1, q_2 \cdots q_n$ 的直线距离 $l_1, l_2 \cdots l_n$；

步骤 6. 在模型 A 上确定顶点 a_i 所在的全部 m 个多边形，在每个多边形上找到距离模型 B 上顶点 p_i 最近的点 $r_1, r_2 \ldots r_m$，并分别计算出 p_i 到点 $r_1, r_2 \ldots r_m$ 的直线距离 $k_1, k_2, \ldots k_m$；

步骤 7. 遍历所有距离 $l_1, l_2 \cdots l_n$，找到最短距离 l_{min}，则 l_{min} 所对应的点 q_j 即为当前最邻近点；

步骤 8. 遍历所有距离 $k_1, k_2, \ldots k_m$ 找到最短距离 k_{min}，$k_{min} <= k_j$，$J = 1, \ldots m$，则 k_{min} 所对应的点 r_j 即为当前最邻近点；

步骤 9. 比较 k_{min} 与 l_{min} 的大小，两者的最小值即为两个模型的最近距离的精确估计值，对应的顶点 a_i 与 q_j、或 p_i 与 r_j 为精确估计的两个模型的最近点。

3. 根据权利要求 1 所述的一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是：步骤 1 所述的将虚拟模型离散的过程中，根据不同的应用场景以及实时性和精度要求进行参数的设定，达到适用其应用场景的模型设定。

4. 根据权利要求 1 所述的一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是：步骤 2 所述构建树型空间结构的方法为采用最近邻搜索算法构建 kd 树的空间结构。

5. 根据权利要求 1 所述的一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是：步骤 3 所述遍历模型 B 的树型空间结构中各节点，并在其中找到距 a_i 最近的点 p_i 时用采用并行计算的方法加以实现，包括基于 GPU 的并行搜索加速、基于 OpenMP 的并行计算、多线程计算、分布式计算的并行计算方法。

一种虚拟现实仿真中物体表面间最小距离的快速计算方法

技术领域

[0001] 本发明属于一种虚拟仿真技术领域，特别涉及一种虚拟模型空间距离计算方法。

背景技术

[0002] 随着虚拟仿真技术的发展，虚拟模型所包含的信息越来越丰富，同时数据量也不断增大。现有技术中，计算两个虚拟物体模型间的最小距离主要有以下两种方案：

[0003] 第一类是穷举的方法，即采用暴力搜索(Brute-force)算法遍历构成两个模型的所有顶点，计算两两顶点之间的最近距离并进行排序比较，找出其中最短的距离，并记下最短距离对应的顶点对，即为两个虚拟模型的最近距离和最近点。这种方法思路简单，容易实现，假设搜索空间包括n个点，则该算法将计算所有可能的n(n-1)/2对点之间的距离，并在其中挑选出最小距离的一对。该算法由于需要遍历所有可能的点对，是一种线性搜索算法，搜索效率很低，其算法复杂度为$O(n^2)$。在实际应用中仅适用于搜索空间较小的情况。

[0004] 第二类是GJK(Gilbert-Johnson-Keerthi)算法，一种碰撞检测算法。用GJK算法来计算的最近距离即两个虚拟模型的明可夫斯基差形状到原点的最近距离。这种方法通过构建明可夫斯基差形状，用迭代的算法避开了对点的遍历，可以获得很高的速度。但GJK算法只适用于凸多边形或凸多面体，所以在实现的应用中可能多数情况都不能直接运行该算法来进行距离的计算，涉及到如何将复杂物体分割成多个凸多面体的问题，增加了其复杂度的不确定性。

[0005] 本发明是针对以上方法的不足，提供一种能快速、精确计算两个虚拟模型之间最小距离以及最邻近点的方法。

发明内容

[0006] 本发明的目的是：提供一种能快速、精确计算两个虚拟模型之间最小距离以及最邻近点的方法。

[0007] 本发明的技术方案是：

[0008] 一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是包括以下步骤：

[0009] 步骤1. 将虚拟物体模型A和B离散成顶点和多边形面片的形式，并标记模型中各顶点的序号以及多边形面片对应的顶点索引信息；

[0010] 步骤2. 对模型B的所有顶点，构建树型空间结构，以实现三维空间的二分查找树的数据存储结构；

[0011] 步骤3. 对于模型A上的每一个点a_i，在欧式空间中遍历步骤2所述的模型B的树型空间结构中各节点，并在其中找到距a_i最近的点p_i，并计算a_i与p_i之前的直线距离d_i；

[0012] 步骤4. 遍历步骤3中得到的所有距离d_i，找出最小值d_{min}，并记录最小值d_{min}对应的顶点a_i和p_i，则最小值d_{min}为两个模型的最近距离的初步估计值，对应的顶点a_i和p_i为初步估计的两个模型的最近点。

[0013] 在模型采样率足够高的前提下，完成步骤 1 至步骤 4 后找出的最近顶点和最短距离便能够近似地表示两个模型的最近点和最小距离。但一方面当模型的采样率比较低的时候，如果直接步骤 1 至步骤 4 后找出的最近顶点计算出的最近距离来表示两个模型之间的最近距离，就会有很大的偏差。另一方面，当虚拟模型表面曲率变化不大的时候，若采样率过高，会使数据量大幅度增加，从而增加运算量，无端增加系统开销。为了避免这样的情况出现，本发明设计了步骤 5- 步骤 9，分别计算两个模型上最近顶点到对方模型上最近顶点所在全部多边形面片的距离，寻找其中的最近点，作为两个虚拟模型之间的最邻近点与最小距离，使结果更加精确，其实现方法为 ：

[0014] 步骤 5. 在模型 B 上确定顶点 p_i 所在的全部 n 个多边形，分别在每个多边形面上找到步骤 3 所述的距离模型 A 上顶点 a_i 最近的点 q_1，$q_2 \cdots q_n$，并分别计算出 a_i 到 q_1，$q_2 \cdots q_n$ 的直线距离 l_1，$l_2 \cdots l_n$ ；

[0015] 步骤 6. 在模型 A 上确定顶点 a_i 所在的全部 m 个多边形，在每个多边形上找到距离模型 B 上顶点 p_i 最近的点 r_1，$r_2 \ldots r_m$，并分别计算出 p_i 到点 r_1，$r_2 \ldots r_m$ 的直线距离 k_1，k_2，$\ldots k_m$ ；

[0016] 步骤 7. 遍历所有距离 l_1，$l_2 \cdots l_n$，找到最短距离 l_{min}，则 l_{min} 所对应的点 q_j 即为当前最邻近点 ；

[0017] 步骤 8. 遍历所有距离 k_1，k_2，$\ldots k_m$ 找到最短距离 k_{min}，$k_{min} <= k_j$，$j = 1, \ldots m$，则 k_{min} 所对应的点 r_j 即为当前最邻近点 ；

[0018] 步骤 9. 比较 k_{min} 与 l_{min} 的大小，两者的最小值即为两个模型的最近距离的精确估计值，对应的顶点 a_i 与 q_j、或 p_i 与 r_j 为精确估计的两个模型的最近点。

[0019] 进一步的，步骤 1 所述的将虚拟模型离散的过程中，根据不同的应用场景以及实时性和精度要求进行参数的设定，达到适用其应用场景的模型设定。

[0020] 进一步的，根据权利要求 1 所述的一种虚拟仿真中物体间最小距离的快速估算方法，其特征是 ：步骤 2 所述构建树型空间结构的方法采用最近邻搜索算法构建 kd 树的空间结构。

[0021] 进一步的，根据权利要求 1 所述的一种虚拟仿真中物体间最小距离的快速估算方法，其特征是 ：步骤 3 所述遍历步骤 2 所述的模型 B 的树型空间结构中各节点，并在其中找到距 a_i 最近的点 p_i 时用采用并行计算的方法加以实现，包括基于 GPU 的并行搜索加速、基于 OpenMP 的并行计算、多线程计算、分布式计算的并行计算方法。

[0022] 本发明与直接暴力搜索遍历方法相比，树空间结构（如 kd 树）的构建极大地加速最近顶点对的搜索过程。找到树结构中距离查询点最近的三维空间点，在模型采样率足够高的前提下，这样找出的最近顶点和最短距离便能够近似地表示两个模型的最近点和最小距离。

[0023] 本发明在点到点最短距离计算的基础上，又实现了点到多边形边和面以及多边形面到多边形面之间最短距离的计算，摆脱了虚拟物体模型建立和多边形离散化时模型分辨率或采样率对算法精度、稳定性和鲁棒性的影响。尤其是在一些曲率变化不大的模型表面可以使采样率尽可能降低，减少多边形数量，以节省存储空间，提高计算效率，同时更好地实现实时性。

[0024] 本发明与 GJK 算法相比，该方法对虚拟模型的具体形状没有任何要求，可以适用

于任意复杂形状（非凸几何体）的模型，具有普适性。

[0025] 本发明在虚拟现实、3D 游戏、场景模拟、工程仿真等领域都有着重要的实用价值和良好的应用前景。

附图说明

[0026] 图 1 为本发明流程图；

[0027] 图 2 为本发明所述的 kd 树对应的三维空间切割示意图；

[0028] 图 3 为点到三角形切片的最短距离计算过程示意图。

具体实施方式

[0029] 实施例 1：如图 1 所示一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是包括以下步骤：

[0030] 步骤 1. 将虚拟物体模型 A 和 B 离散成顶点和多边形面片的形式，并标记模型中各顶点的序号以及多边形面片对应的顶点索引信息；设模型 A 和 B 离散后的顶点数为为 N_A 和 N_B；虚拟模型离散的过程中，根据不同的应用场景以及实时性和精度要求进行参数的设定，达到适用其应用场景的模型设定。

[0031] 步骤 2. 对模型 B 的所有顶点，采用构建树型空间结构，以实现三维空间的二分查找树的数据存储结构；其中：

[0032] 构建相应的树型空间结构选用最近邻搜索算法构建 kd 树的空间结构。kd 树是用来快速搜索近邻的一种数据结构，对于三维空间的点，其基于点的空间位置信息，通过二分法迭代划分三维空间，实现最优存储与快速搜索。三维空间的划分示意图如图 2 所示。由于模型 B 的顶点数为 N_B，则在 kd 树上进行 k 近邻查找的时间复杂度为 $O(\log_2 N_B)$。对于该方法中的三维空间 kd 树的构建过程如下：

[0033] a1. 在三维数据集合（模型 B 顶点集合）中选择具有最大方差的维度 m，在该维度上以其中值为中心对该数据集合进行划分，得到两个子集合，并创建一个树结点 node 用于存储；

[0034] a2. 每个子集合重复步骤 a1 直到所有子集合都不能再划分为止。

[0035] 步骤 3. 对于模型 A 上的每一个点 a_i，在欧式空间中遍历模型 B 步骤 2 所述的树型空间结构中节点，并在其中找到距 a_i 最近的点 p_i，并计算 a_i 与 p_i 之前的直线距离 d_i；最终得到点对集合为 $\{a_i, p_i | a_i \in A, p_i \in B\}$，$i = 1 \cdots N_A$，对应的距离值为 $\{d_i\}$，$i = 1 \cdots N_A$；其中：

[0036] 对于 kd 树，可以基于欧式距离进行 k 近邻的查找，但在该方法中只需要得到模型 B 中距离 a_i 最近的点，所以取 $k = 1$ 即可。

[0037] 对于查询点 a_i 和模型 B 构建好的 kd 树，最近邻查找的过程如下：

[0038] b1. 从根结点开始，将查询点 a_i 与各个结点进行比较，并根据比较结果向下访问 kd 树，直到达到叶子结点；

[0039] b2. 到达叶子结点后进行回溯操作，判断未被访问过的分支里是否还有离查询点 a_i 更近的点，从而找到离查询点 a_i 更近的最近邻点。

[0040] 所述遍历模型 B 的树型空间结构中各节点，并在其中找到距 a_i 最近的点 p_i 时用采

用并行计算的方法加以实现，包括基于GPU的并行搜索加速、基于OpenMP的并行计算、多线程计算、分布式计算的并行计算方法。

[0041] 步骤4. 遍历步骤3中得到的所有距离 d_i，找出最小值 d_{min}，并记录最小值 d_{min} 对应的顶点 a_i 和 p_i，则最小值 d_{min} 为初步估计的两个模型的最近距离，对应的顶点 a_i 和 p_i 为初步估计的两个模型的最近点。

[0042] 实施例2：一种虚拟现实仿真中物体表面间最小距离的快速计算方法，其特征是在实施例1的基础上还包括以下步骤：

[0043] 步骤5. 在模型B上确定顶点 p_i 所在的全部n个多边形，分别在每个多边形面上找到步骤3所述的距离模型A上顶点 a_i 最近的点 $q_1, q_2 \cdots q_n$，并分别计算出 a_i 到 $q_1, q_2 \cdots q_n$ 的直线距离 $l_1, l_2 \cdots l_n$；

[0044] 该步骤也就是求模型A上顶点 a_i 到的模型B上确定顶点 p_i 所在的全部n个多边形的最近点及距离，即求点到面的最短距离。可以采用点到点在面上的投影点之间的距离来计算。多边形以三角形为例。点Q到三角形ABC之间的最短距离的流程如下：

[0045] c1. 令 $v_0 = A-B$，$v_1 = A-C$，则 (v_0, v_1) 为该三角形所确定平面的一个基，此时平面上任意一点可以表示为 $Pt = A+t_0 \cdot v_0+t_1 \cdot v_1$。

[0046] c2. 设点Q在三角形ABC所在平面上的投影为P，则有 $Q-P = Q-(A+t_0 \cdot v_0+t_1 \cdot v_1)$，且 $\begin{cases}(Q-P)\cdot \mathbf{v_0}=0\\(Q-P)\cdot \mathbf{v_1}=0\end{cases}$，即 $\begin{cases}(Q-A)\cdot v_0=t_0\cdot v_0\cdot v_0+t_1\cdot v_1\cdot v_0\\(Q-A)\cdot v_1=t_0\cdot v_0\cdot v_1+t_1\cdot v_1\cdot v_1\end{cases}$，解方程组即可得到 t_0 和 t_1 的值，代入 $P = A+t_0 \cdot v_0+t_1 \cdot v_1$ 便得到点P的坐标值。

[0047] c3. 若有 $\begin{cases}0\le t_0\le 1\\0\le t_1\le 1\\t_0+t_1\le 1\end{cases}$，则 $P = A+t_0 \cdot v_0+t_1 \cdot v_1$ 在三角形ABC内部，那么点P即为点Q在三角形上的最近点，|QP|即为最短距离；否则P不在三角形ABC内部，执行步骤c4；

[0048] c4. 根据参数 t_0, t_1 的范围划分为以下三种情况，如图3所示：$t_0 < 0$，将点P投影在边AC上，即 $t_0 = 0$，并将 t_1 限制在[0,1]范围内；或 $\begin{cases}t_0>0\\t_1<0\end{cases}$，将点P投影在边AB上，即 $t_1 = 0$，并将 t_0 限制在[0,1]范围内；或 $\begin{cases}t_0>0\\t_1>0\\t_0+t_1>1\end{cases}$，将点P投影在边BC上，即 $t_0+t_1 = 1$，并将 t_0, t_1 限制在[0,1]范围内。这三种情况下点Q在三角形某条边上的投影点即为点Q在三角形上的最近点，相应的距离即为最短距离。

[0049] 步骤6. 在模型A上确定顶点 a_i 所在的全部m个多边形，在每个多边形上找到距离模型B上顶点 p_i 最近的点 $r_1, r_2 \ldots r_m$，并分别计算出 p_i 到点 $r_1, r_2 \ldots r_m$ 的直线距离 $k_1, k_2, \ldots k_m$；

[0050] 步骤7. 遍历所有距离 $l_1, l_2 \cdots l_n$，找到最短距离 l_{min}，则 l_{min} 所对应的点 q_j 即为当前最邻近点；

[0051] 步骤8. 遍历所有距离 $k_1, k_2, \ldots k_m$ 找到最短距离 k_{min}，$k_{min} <= k_j$，$j = 1, \ldots m$，

则 k_{min} 所对应的点 r_j 即为当前最邻近点；

[0052] 步骤 9. 比较 k_{min} 与 l_{min} 的大小，两者的最小值即为两个模型的最近距离的精确估计值，对应的顶点 a_i 与 q_j、或 p_i 与 r_j 为精确估计的两个模型的最近点。

[0053] 实验结果：对若干具有不同顶点数的虚拟模型完成步骤 1-4 的运算速度的统计如下表所示：

[0054]

实验序号	模型 A 顶点数 N_A	模型 B 顶点数 N_B	实时性 (fps)
1	24	88	～ 100
2	88	24	～ 100
3	347	644	～ 90
4	644	347	～ 90
5	4648	4996	～ 70
6	4996	4648	～ 40
7	1028	17473	～ 60
8	17473	1028	～ 20
9	17473	17473	～ 10

[0055] 由统计结果可以看出，随着顶点数量的增加，处理时间会增加，但都能基本满足实时性的要求。另外，实验中用来构建 kd 树的是模型 B 中的顶点，而模型 A 中的顶点则是用来作为查询点，所以当两个模型的顶点数量不同时，选择顶点数量少的那个作为模型 A 可以获得更高的实时性。

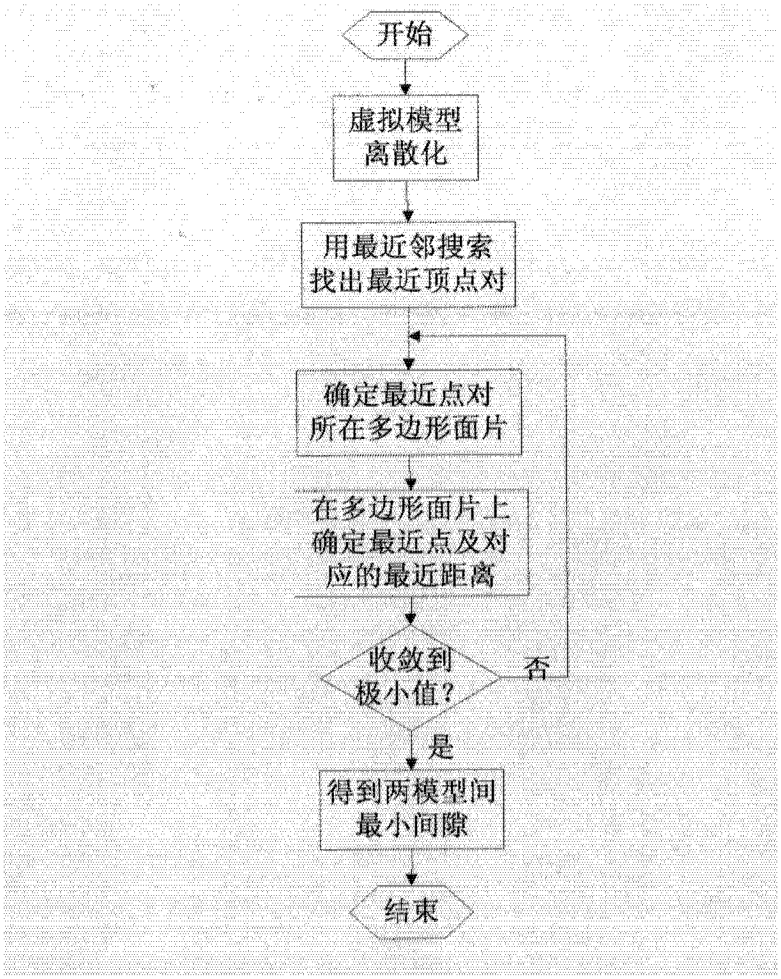
开始
虚拟模型离散化
用最近邻搜索找出最近顶点对
确定最近点对所在多边形面片
在多边形面片上确定最近点及对应的最近距离
收敛到极小值?
否
是
得到两模型间最小间隙
结束

图 1

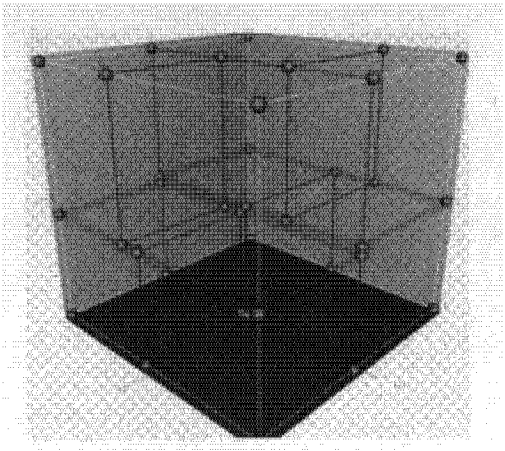

图 2

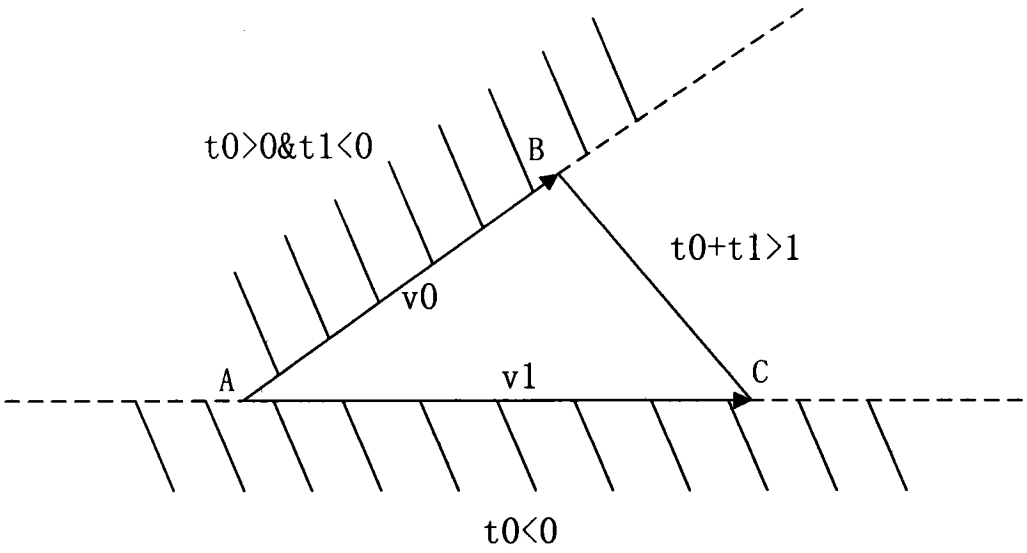

图 3