遗传算法和 Dijkstra 算法在动态权值系统中的比较

马 超
(西北大学 软件学院,陕西 西安 710100)

摘 要:针对遗传算法和 Dijkstra 算法在求解动态权值系统中最短路径时的性能问题,采用比较法,将两种算法应用在同一个实际游戏模型中,对其算法的稳定性、智能性、时间复杂度进行对比测试。游戏模型模拟了各种条件下的动态权值系统。为了使遗传算法更加可靠,通过优化其变异过程使得收敛速度更快,可靠性更高。实验数据表明,遗传算法在每张地图上的得分数以及算法所用时间普遍高于 Dijkstra 算法,从而得出遗传算法在求解动态权值系统中最短路径问题时稳定性和预期效果明显好于 Dijkstra 算法,但其时间复杂度较高的结论。

关键词:遗传算法;Dijkstra 算法;最短路径;动态权值
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2012)09-0021-04

Comparison of Genetic Algorithm and Dijkstra Algorithm in Dynamic Weight System

MA Chao
(College of Software, Northwest University, Xi'an 710100, China)

Abstract: Used a comparative approach to compare the performance of the genetic algorithm with the Dijkstra algorithm when solve the shortest path problem in the dynamic weight system. Did an experiment in the actual model with these two algorithms in order to test their stability, intelligence and time complexity. The game model makes many kinds of dynamic weight system. In order to make the genetic algorithm more reliable, the new algorithm gets a way to optimize the process of mutation to make the speed of the genetic algorithm faster and the reliability better. The experiment data shows that most data of the genetic algorithm is higher than the Dijkstra algorithm. The experiment makes a conclusion that the stability and expected result of the genetic algorithm is better than the Dijkstra algorithm in the dynamic weight system, but the time complexity of algorithm is higher than the Dijkstra algorithm.

Key words: genetic algorithm; Dijkstra algorithm; shortest path; dynamic weight

0 引 言

随着科技日新月异的进步,尤其是各种网络技术的飞速发展,最短路径(Shortest Path, SP)[1,2]问题已经越来越多地被应用在了各项技术中,例如:城市交通规划、网络路由、项目进度规划、游戏人工智能等。

最短路径问题是图论研究中的一个经典算法,旨在寻找图中两节点之间的最短路径。传统解决最短路径问题的算法有:Dijkstra, A *, SPFA, Bellman-Ford, Floyd-Warshall 等,其中以 Dijkstra 算法作为代表。

但是,随着人们面临的实际问题越来越复杂,传统的算法在解决动态权值[3]系统中最短路径问题时显得越来越力不从心,使用效果往往不尽人意。此时,一批新的算法逐渐崭露头角。其中,遗传算法(Genetic Algorithms, GA)是这些新型算法中比较具有代表性的。

为了验证遗传算法在求解动态权值系统中最短路径问题时所具备的优越性,笔者在一个自己开发的游戏场景中进行实验,将遗传算法与 Dijkstra 算法进行测试和比较,最终分析实验数据并得出结论。

1 算法简介

1.1 遗传算法简介

遗传算法是由美国密执安(Michigan)大学的 Holland 教授在 1975 年提出的。算法建立在达尔文(Drawin)生物进化论和孟德尔(Mendel)遗传学说的基础上,经过后人的不断改善变得逐渐完善。

遗传算法的基本思想是[4]:将待解决问题解的参数编辑成自定义编码,即基因(gene)。将解的所有参

收稿日期:2012-02-02;**修回日期**:2012-05-08
基金项目:陕西省科技攻关项目 (2009k01-53)
作者简介:马 超(1990-),男,陕西安康人,软件工程师,研究方向为网络与信息安全。

数连接在一起形成一个编码链,即染色体(chromosome)。许多染色体之间进行类似于自然选择的操作。任意挑选一对染色体进行配对交叉和变异运算从而形成新的染色体。每条染色体都有一个适应值,适应值越高的染色体被保留下来的概率越高,反之被淘汰的概率越高。经过若干代的选择直到最终结果收敛。算法流程由图 1 所示。

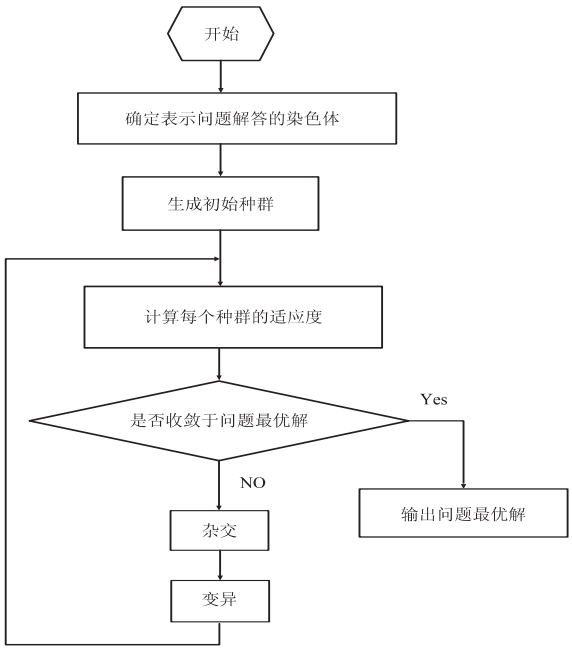

图 1 遗传算法程序流程

1.2 Dijkstra 算法简介

Dijkstra 算法[2]是典型的最短路径算法,用于计算一个节点到其它节点的最短路径。主要的特点是以起始点为中心向外层层扩展,直到扩展到终点为止。Dijkstra 算法能得出静态权值条件下最短路径的最优解,但是由于遍历节点多,通常效率比较低。

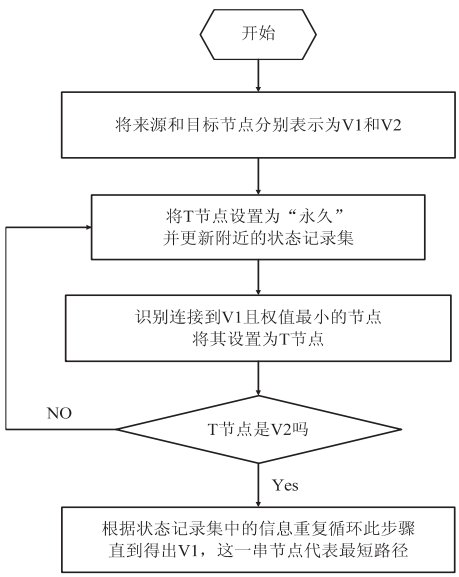

图 2 Dijkstra 算法流程图

Dijkstra 算法将图中节点分成 3 部分:未标记节点,临时标记节点和永久标记节点。图中所有节点首先初始化为未标记节点,在搜索过程中和最短路径中的节点相连通的节点为临时节点,每次循环都是从临时节点中搜索距离源点路径长度最短的节点作为标记节点,直到找到目标节点或者所有节点都成为永久标记节点来结束算法。算法流程由图 2 所示。

2 提出实验场景

2.1 实验目的

通过模拟实验对遗传算法和 Dijkstra 算法进行比较,分析实验数据从而得出两种算法在解决动态权值条件下最短路径问题时性能的优劣。

2.2 实验场景基本信息

实验场景是由 C#开发的一个测试平台,提供 DLL 接口供实验者开发并运行不同的算法。

实验场景模拟实现了一个“英雄救美”的游戏场景。场景由一张矩形地图组成,矩形地图被平均划分为 20×20 个小矩形,每个小矩形称为一个节点。节点类型包括:路(Path),墙(Wall),英雄(Hero),公主(Princess),怪兽(Monster),宝石(Treasure)。用户可以自己设计地图并编写算法,算法控制英雄找到一条救出公主的路线(从起始点(0,0)到公主所在节点,再由公主所在节点回到起始点)。

各类节点的详细信息:英雄(生命值=100,攻击力=10,防御力=5),怪物和宝石同样具有生命值、攻击力、防御力三种属性,属性值不固定(可以在设计地图时自行设置),其中宝石的各项属性允许为负值。

英雄不能穿越墙,遇到怪物会与其战斗。遇到宝石可以提升自身属性。战斗规则:英雄和怪物轮流攻击对方。英雄首先发动攻击。每次攻击都会使对方生命值减少,减少值为攻击者的攻击力减去被攻击者的防御力。如果防御力大于等于对方的攻击力,则攻击无效。当有一方生命值小于等于 0 时,战斗结束。英雄胜利后,原怪物节点变为路节点。战斗失败则游戏失败。如果双方的攻击力都小于对方的防御力,则游戏也算失败。遇到宝石后英雄的各项属性加上宝石对应的各项属性(宝石属性若为负则减去),然后宝石节点变为路节点。

游戏胜负判定:在英雄救出公主的前提下,要求最终得分尽可能高。

最终得分 = 英雄最终所剩生命值 - 英雄移动的步数。

3 问题分析

3.1 问题抽象

本实验可以抽象成求解最短路径问题。英雄节点

即为起始节点也为终止节点,路径节点中必须包含公主节点。节点与节点之间的权值可以由游戏规则确定。实验要求实验者编写的算法最终得到一条复合游戏规则的最短路径(使得最终得分尽可能的高)。

3.2　难点分析

本实验抽象出来的最短路径问题与传统的最短路径问题最大的不同在于权值的动态性。传统的最短路径问题可以很容易地将节点与节点之间的固定权值确定下来。而由于本实验场景规则的复杂,在地图确定以后无法将其转化成一个具有固定权值的有向图。比如说,英雄战胜一个怪物节点所需要的代价与英雄当前的属性息息相关。由于无法知道战斗前英雄的属性值,所以到达该怪兽节点所需的代价也是不固定的。计算三种不同情况下两节点之间的权值:

X 节点为路节点,Y 节点为怪物节点。

① 英雄属性为(100,10,5),Y 节点属性为(20,7,5),则 $X -> Y$ 的权值 $W = 1 + Fight$; $Fight = (20/(10 - 5) - 1) \times (7 - 5) = 6$,故 $W = 7$。

② 英雄属性为(100,10,10),Y 节点属性为(20,7,5),则由游戏规则可知英雄不会有生命值的损失,故此时 $X -> Y$ 的权值 $W = 1$。

③ 英雄属性为(100,5,5),Y 节点属性为(20,10,6),则由游戏规则可知英雄取得胜利,故此时 $X -> Y$ 的权值 $W = \infty$。

可以看出,英雄属性的不确定性对节点之间权值的影响是十分巨大的,并且实际情况还将包括宝石节点,并且宝石节点的属性允许为负,所以实际问题的复杂性将会更高。

如果近似生成一套固定权值然后采用传统的 Dijkstra 算法进行求解,所得结果势必无法达到预期的效果,即所求得的最短路径并不一定是该地图理论上真正的最短路径,并且有可能由于权值误判带来算法的失败(例如算法选择了无法战胜的怪物节点)。

对于遗传算法来说,算法无需知道各个节点之间的固定权值,而是采用随机的思想对结果进行筛选。对于本实验来说,遗传算法的适应度计算函数也是明确的,即 $f(x) = health - steps$。其中 health 为英雄最终所剩的生命值,steps 为英雄所走的步数。所以我们猜想,遗传算法在求解该动态权值最短路径问题的性能会好于 Dijkstra 算法。加以实验证明了提出的猜想。

4　算法设计

4.1　Dijkstra 算法设计

4.1.1　算法流程

①根据地图信息将地图转化成一张有向图。根据地图节点具体信息设置节点之间的关系和近似权值。

②利用 Dijkstra 算法[5,6] 获得一条从英雄的起始节点到公主节点的最短路径。

③更新地图信息,将英雄走过的节点设置为路节点,并更新地图权值。

④继续利用 Dijkstra 算法确定一条从公主节点到英雄初始节点的最短路径。

⑤对路径进行优化[7,8]。由于 Dijkstra 算法限定了算法无法“走回头路”。即在一次 Dijkstra 算法中无法重复遍历一个节点。这导致了英雄无法走到类似于被三面墙包围的宝石节点。为了对路径进行优化,算法遍历剩下的宝石节点,判断每个节点是否可以“插入”到之前确定下来的路径中。

4.1.2　节点权值具体设置方法

每个节点最多能和四个节点有联系,即上,下,左,右四个方向的邻接节点。若邻接节点为墙,则直接舍去该节点。

如果两个节点都为路节点,则这两个节点之间相互权值 $W = 1$。

如果相邻两个节点分别为路和怪兽,则路节点到怪兽节点的权值 $W = 1 + F$,其中 F 代表战斗代价,即为英雄所失去的生命值;怪兽节点到路节点之间的权值 $W = 1$。

如果相邻两个节点分别为路和宝石,则路节点到怪兽节点的权值 $W = 1 - H$, 其中 H 为宝石节点的生命值属性;宝石节点到路节点的权值 $P = 1$;

4.1.3　算法分析

算法在对待怪兽和宝石这两种特殊节点时,都是采用了权值近似处理。算法无法知道英雄在遇上怪兽节点时当前的生命值,所以都以 100 处理。除此之外,在选择宝石节点时则是采用了生命值优先的近似,但实际情况可能是攻击力优先或者防御力优先更加合适。由于这样的节点权值的设定方式,预期本算法获得的最短路径大多数情况下并不一定是最优解。

4.2　遗传算法设计

4.2.1　算法流程

①进行初始化工作。扫描整个地图,将重要节点进行标志。重要节点包括:怪兽,宝石,公主三种节点。求出两个重要节点之间的绝对最短路径。这里的绝对最短路径即物理长度上的最短路径,即路径中不包括其它重要节点。

②产生初始种群。算法采用随机原理在重要节点中任意挑选若干节点进行组合(允许有重复节点)。组合路径的首节点和末节点都为英雄的初始节点,并且路径中包含公主节点。设置初始种群规模为 4000。

③进行交叉迭代[9]。算法在父代两条染色体中任意抽出一段首尾节点相同的染色体段进行交叉变

换。交叉变换算法由图 3 所示。

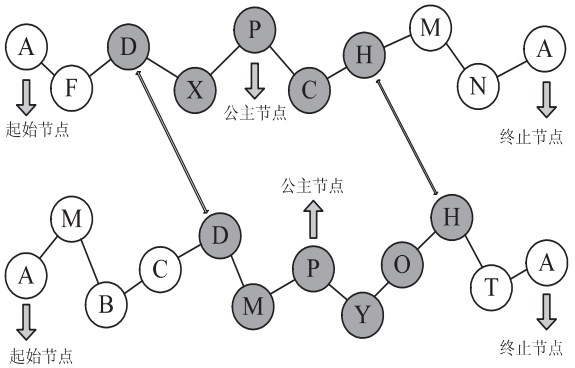

图 3 交叉变换算法

④变异操作[10,11]。算法在新生成的染色体序列中任意抽取两点,用 A * 算法[12]快速求得两节点间的最短路径,并和原有路径进行替换(算法同时必须保证公主节点不会被替换掉)。变异操作由图 4 表示:

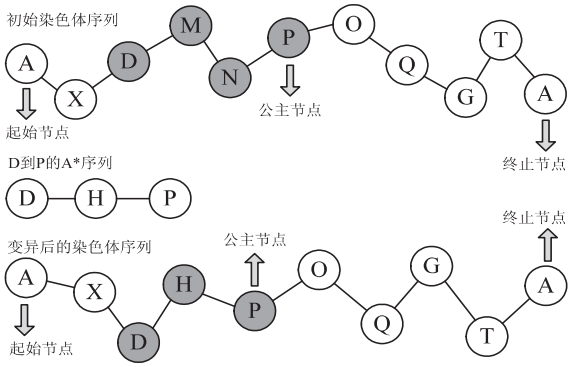

图 4 变异操作

⑤适应度判断。计算新生染色体的适应度,适应度的评定函数即为最终得分的计算函数。将新生染色体与其父代染色体适应度进行比较,淘汰适应度最低的染色体。

⑥判断收敛性。如果种群收敛,则得出最终最优路径,如果没有,回到③。

4.2.2 算法分析

该遗传算法巧妙地避免了权值设定这一步骤,直接对结果进行求解,预期稳定性和效果会强于 Dijkstra 算法。

5 实验与分析

5.1 进行试验

对上述两种算法进行编码,编译成为两个 dll 文件用于实验平台的加载,编码采用 C#语言。

将两种算法在自行设计的 20 张具有代表性的地图上进行测试,测试数据如表 1 所示。

5.2 数据分析

对两种算法在 20 张地图上运行的数据进行分析,可以得出以下结论:

表 1 两种算法在 20 张地图上的得分

地图	是否成功		运行时间 s		路径节点数		所剩生命值		总得分	
编号	D	G	D	G	D	G	D	G	D	G
01	成功	成功	3.21	5.84	80	78	115	117	35	39
02	成功	成功	0.49	5.12	54	74	124	158	70	84
03	成功	成功	2.38	6.91	150	144	250	267	100	123
04	成功	成功	2.60	3.42	104	82	152	133	48	51
05	成功	成功	1.45	5.94	124	116	157	152	33	36
06	成功	成功	1.13	4.91	108	104	157	156	49	52
07	失败	成功	失败	6.4	失败	120	失败	138	失败	18
08	成功	成功	6.13	5.20	72	148	100	193	28	45
09	成功	成功	1.91	5.93	100	68	163	137	63	69
10	成功	成功	3.15	4.43	98	86	148	146	50	60
11	成功	成功	3.55	6.15	140	146	160	181	20	35
12	成功	成功	4.24	4.19	280	108	177	139	-103	31
13	成功	成功	1.78	6.05	120	88	165	165	45	77
14	失败	成功	失败	2.27	失败	72	失败	200	失败	128
15	成功	成功	2.90	6.67	110	100	147	152	37	52
16	成功	成功	2.09	5.71	82	86	121	131	39	45
17	成功	成功	3.02	4.85	90	86	130	132	40	46
18	成功	成功	2.56	6.11	68	96	119	161	51	65
19	失败	成功	失败	4.54	失败	82	失败	404	失败	322
20	成功	成功	2.12	8.03	108	80	173	148	65	68

1. Dijkstra 算法有 3 次算法失败,成功率为 85%;遗传算法全部成功,成功率 100%。

2. Dijkstra 算法的平均运行时间为 2.63s;遗传算法的平均运行时间为 5.43s。

3. Dijkstra 算法的平均总得分为 33.5(失败按 0 分计算),遗传算法的平均总得分为 72.3。

两种算法在 20 张地图上的得分折线图如图 5 所示。

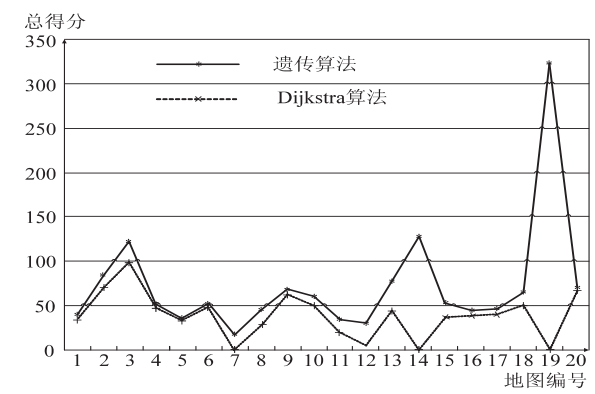

图 5 两种算法总得分比较图

5.3 实验结论

实验数据表明:遗传算法在解决动态权值系统下最短路径选择问题时所表现出的稳定性,解集的优越性都要高于传统的 Dijkstra 算法,但是在相同问题下算法运行的时间复杂度一般要高于 Dijkstra 算法。

(下转第 28 页)

表 2 水平偏转人脸特征点定位平均时间和正确率

试验方法	正确率	平均时间(ms)
文献[5]中方法	75.31%	4022
文中方法	85.05%	1134

4 结束语

文中提出了一种基于偏转角度的人脸定位方法,与单纯使用反向组合方法的 AAM 人脸特征点定位相比,采用 Adaboost 和 YCbCr 色彩空间得到了人脸的大致位置,进一步找到了眼睛和嘴的位置,解决了拟合中心初始位置问题,根据偏转角度,给出相匹配的模板,减少了迭代次数,提高了人脸特征点定位的速度和准确度。文中方法对有单一偏转角度的人脸特征点定位有较好的定位效果,但当偏转角度过大时效果不理想。还不能解决水平加俯仰的人脸特征点定位,这是下一步需要解决的问题。

参考文献:

[1] Edwards G J,Taylor C J,Cootes T F. Interpreting face images using active appearance models[C]//3rd IEEE International Conference on Automatic Face and Gesture Recognition Proceedings. Washington D C, USA: IEEE Computer Society, 1998:300-305.

[2] Blanz V,Vetter T. A morphable model for the synthesis of 3D faces[C]//Proceeding of SIGGRAPH. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co,1999:187-194.

[3] Matthews I,Baker S. Active appearance models revisited[J]. International Journal of Computer Vision,2004,60(2):135-164.

[4] 牛 星. 基于改进 AAM 的人脸特征点提取[J]. 应用科技,2011,38(4):35-38.

[5] 呼月宁,张艳宁. AAM 在多姿态人脸特征点检测中的应用[J]. 计算机工程与应用,2010,46(12):161-165.

[6] Yang J. Two-dimensional PCA: A New Approach to Appearance-based Face Representation and Recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005,26(1):131-137.

[7] 蔡雪君,谢松云,张 波. 一种改进的利用五官特征的人脸识别方法[J]. 计算机仿真,2009(11):228-230.

[8] Baker S, Matthews I. Lucas-Kanade 20 years on: a unifying framework[J]. International Journal of Computer Vision, 2004,56(3):221-255.

[9] 武 勃,黄 畅. 基于连续 Adaboost 算法的多视角人脸检测[J]. 计算机研究与发展,2005,42(9):1612-1621.

[10] Jeng S H,Liao H Y M,Han C C,et al. Facial feature detection using geometrical face model: an efficient approach[J]. Pattern Recognition,1998,31(3):273-282.

[11] 彭一凡,张 翼,宋明黎. 基于特征跟踪和融合的人脸风格化动画的研究[J]. 计算机技术与发展,2009,19(2):127-129.

[12] 时书剑,马 燕. 基于 Gabor 滤波和 KPCA 的人脸识别方法[J]. 计算机技术与发展,2010,20(4):92-95.

(上接第 24 页)

6 结束语

遗传算法在动态权值系统下最短路径问题所表现出的优越性是有目共睹的,但不能否认,Dijkstra 算法在解决静态权值最短路径[13]问题时也有着不可替代的优势。这两种算法都还有大量提升和优化的空间。就遗传算法而言,尽量减小算法的时间复杂度,避免结果早熟等都是优化的手段。对于 Dijkstra 算法,如何为算法构造动态权值库也将成为算法优化的关键。

参考文献:

[1] 沙宗尧,边馥苓. 单源最短路径算法的图示教学设计与实践[J]. 测绘通报,2010(4):58-61.

[2] 严蔚敏,吴伟民. 数据结构(C 语言版)[M]. 北京:清华大学出版社,1997.

[3] 鲍培明. Dijkstra 算法在动态权值系统中的应用[J]. 计算机工程,2000,26(4):11-12.

[4] 徐庆征,柯熙政. 求解最短路径的遗传算法中若干问题的讨论[J]. 计算机工程与设计,2008,29(6):1507-1509.

[5] 乐 阳,龚健雅. Dijkstra 最短路径算法的一种高效率实现[J]. 武汉测绘大学学报,1999,24(3):209-212.

[6] 郝春梅. 一种改进的 Dijkstra 算法的分析及程序实现[J]. 计算机与现代化,2011(1):36-38.

[7] 陈益富,卢 潇,丁豪杰. 对 Dijkstra 算法的优化策略研究[J]. 计算机技术与发展,2006,16(9):73-75.

[8] 叶仕灏,王伊蕾. 一种优化 Dijkstra 算法的研究[J]. 计算机应用与软件,2011,28(9):272-274.

[9] 康晓军,王茂才. 基于遗传算法的最短路径求解[J]. 计算机工程与应用,2008,44(23):22-23.

[10] 李 擎,张 伟,尹怡欣,等. 一种用于最优路径规划的改进遗传算法[J]. 信息与控制,2006,35(4):444-447.

[11] Gen M,Cheng R W,Wang D W. Genetic algorithms for solving shortest path problems[C]//Proceedings of the 1997 IEEE International Conference on Evolutionary Computation. Piscataway, NJ, USA: IEEE, 1997:401-406.

[12] Wu W, Ruan Q Q. A gene constrained genetic algorithm for solving shortest path problem[C]//Proceedings of the 2004 7th International Conference on Signal Processing. Piscataway, NJ, USA: IEEE, 2004:2510-2513.

[13] Kajiya J T, Kay T L. Rendering fur with three dimensional textures[C]//Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH. New York: ACM Press, 1989: 271-280.

遗传算法和 Dijkstra 算法在动态权值系统中的比较

作者：马超
作者单位：西北大学 软件学院,陕西 西安 710100
刊名：计算机技术与发展
英文刊名：Computer Technology and Development
年，卷(期)：2012(9)