文章编号: 1002-0446(2001)01-0040-05

复杂环境下路径规划问题的遗传路径规划方法°

陈 刚 沈林成

(国防科技大学自动控制系 长沙 410073)

摘 要: 本文主要研究复杂环境下路径规划问题的遗传算法求解方法. 介绍了适于求解路径规划问题的遗传算法, 针对复杂环境的特点设计了有效的路径遗传算子, 在此基础上提出一种新的度量路径个体适应度的计算方法. 试验表明, 该算法有很强的鲁棒性, 适合于复杂环境下的路径规划.

关键词: 遗传算法; 路径规划

中图分类号: TP24 文献标识码: B

GENETIC PATH PLANNING ALGORITHM FOR COMPLEX ENVIRONMENT PATH PLANNING

CHEN Gang SHEN Lin-cheng

(*National University of Defense Technology, Auto Control Department* 410073)

Abstract: This article mainly discuss the genetic algorithm solution method for planning problem under complex environment. We put forward a genetic algorithm which is suitable for solving path planning problem, and have designed an effective path planning genetic operator pointing to the characteristics of complex environment. On the basis of above discussion, we put forward a new operator method for measuring path individual body's degree of adaptability. The experiments show this method has very good robustness. It is fit for solving path planning problem under complex environment.

Keywords: genetic algorithm, path planning

1 引 言

复杂环境下的路径规划问题一直是个没有解决的难题. 人们尝试用遗传算法来解决路径规划问题并取得一定成功. **Celeghon**(1988)用遗传算法来求解平面避障问题, 其后 **Davier**(1991)、**Joslon**(1993)、**Chen** 等陆续展开了用遗传算法来进行路径规划的研究. 但是复杂环境下的路径规划问题却一直没有得到很好的解决. 究其原因, 困难主要集中在以下 4 个方面:

1) 路径个体编码设计不合理, 进化效率低, 进化过程中经常产生非法个体.

2) 个体适应度函数设计不准确, 无法度量不同类型路径个体的优劣.

3) 遗传算子设计不合理, 进化效果不明显.

4) 规划过程没有利用背景知识, 进化效率不高.

本文正是从以上几点出发, 通过修改遗传算法来解决复杂环境下的路径规划问题, 通过性能分析表明, 这种算法具有很强的通用性.

2 遗传算法求解路径规划问题的基本方法

遗传算法求解路径规划问题的基本思想是: 将路径个体表达为路径中的一系列中途点, 并转换为二进制串. 首先初始化路径群体, 然后进行遗传操作, 如选择、交叉、复制、变异. 经过若干次代的进化以后, 停止进化, 输出当前最优个体, 其过程如算法 1 所示.

算法 **1**

开始:

随机初始化群体 $P(0)$;

计算群体 $P(0)$ 中个体的适应度;

° 收稿日期: 1999-09-18

t: = 0
while(不满足终止准则) do
{
由 $P(t)$ 通过遗传操作形成新的种群 $P(t+1)$;
计算 $P(t+1)$ 中个体的适应度, t: = $t+1$;
}

由于路径规划问题的复杂性, 遗传算法应用于复杂环境下路径规划存在很大的缺陷, 表现如下:

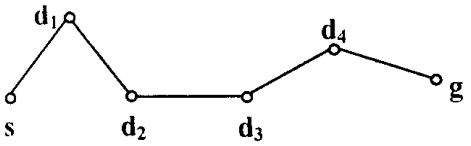
1 对于二进制串编码, 若路径中途点个数较多, 则编码非常长, 降低算法搜索效率; 采用二进制串编码, 相邻整数存在 hamming 悬崖, 如 15 和 16 的二进制表示分别为 01111 和 1000, 因此从 15 到 16 将改变所有的位, 导致进化效率过低 ; 采用二进制串表示路径坐标有一定的冗余度, 因此变异会产生非法个体.

2 交叉采用等长交叉策略, 路径个体的中途点不会增加或减少. 交叉只导致路径个体坐标发生改变, 在复杂环境下, 路径个体中途点数目不变, 有时根本无法进化出合法路径.

3 变异采用随机变异. 由于一次只能变异一位, 只能使路径中途点的 x 坐标或 y 坐标单个变化, 对于复杂环境, 这样的变异效率太低, 无法达到有效搜索未知区域的作用.

3　复杂环境下的遗传规划算法

对于复杂环境, 遗传算法必须有合适的编码方式, 准确的适应度函数, 有效的遗传算子, 对背景知识的较强利用. 本文采用大范围初始化, 实数直接编码, 锦标赛选择方式, 有效的基因突变、构造基因库等策略. 同时, 提出用翻越障碍物的能力来表达个体适应度, 取代了用穿越障碍物的长度来表示个体适应度的方法.



R=<s.d1.d2,…dn, g>

图 1　路径的自由编码表示

个体编码采用实数直接编码, 路径个体表示为从出发点到目标点的一系列中途点. 如图 1 所示, $R=\langle sd_1d_2\cdots d_ng\rangle$, 其中 s 为路径的起点, g 为路径的终点, d_i 其中的中途点.

直接编码的有效克服了二进制编码的缺点, 直接将中途点坐标(x, y)表示为一个基因片. 个体长度只与中途点有关, 变异时可以对(x, y)坐标同时进行改变, 有效地扩大了搜索范围.

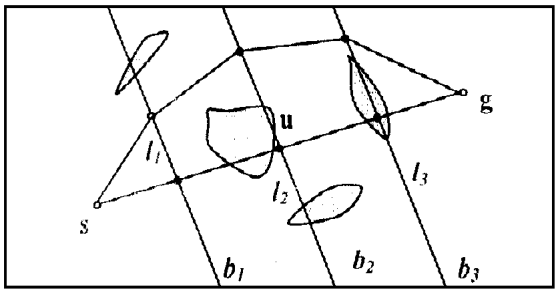


图 2　大范围初始化路径

在复杂环境下, 为了保证初始化的广泛性, 必须采用大范围初始化, 如图 2 所示. 将 s(出发点)、g(目标点)连线成$\overline{sg}$, 将$\overline{sg}$均匀分成 n 份, 依次做垂直于直线$\overline{sg}$的直线 $b_i(i=1, n-1)$. 初始化路径时在直线 b_i 上随机选择点, 然后将这些点连成直线.

选择算子采用锦标赛选择(Tournment). 它的基本思想是自然界中一个生物只与群体中有限的一部分进行竞争. 联赛选择是一种基于局部竞争机制的选择, 其操作思想是从群体中选择 k 个个体进行比较, , 选择适应度最好的个体进入下一代. 参数 k 称为联赛规模, 常取 $k=2$. 显然, 这种方式使适应度值较好的个体具有较大的"生存"机会; 同时, 它只使用适应度的相对值作为选择标准, 而与适应度值的大小不成比例, 因而它能避免群体中超级个体的出现, 在一定程度上避免了过早收敛和停滞现象的发生. 在复杂环境下进化很容易陷入局部极值点, 因此这对于复杂环境下的路径进化尤为重要.

交叉算子采用非对称单点杂交策略. 非对称是指进行杂交的路径个体染色体长度不必相等, 杂交点不必在同一位置上. 如图 3 所示, 在路径 P_1 中随机选择杂交点 $i_1=$ rand$[1, m_1]$, 在路径 P_2 中随机选择杂交点 $i_2=$ rand$[1, m_2]$, 互相交换 $i_k(k=1, 2)$ 之后的路径部分, 得到两个新的路径个体.

变异算子有增加一个点、减少一个点、移动一个点以及替换一段路径 4 种方式. 在路径规划中, 变异的主要作用是探知未知区域, 提供杂交所需新材料, 构造可行路径. 变异采用启发式变异, 即先对穿越障碍物的点进行变异. 只有路径个体中不存在穿越障碍物的点后, 才随机变异路径个体的中途点.

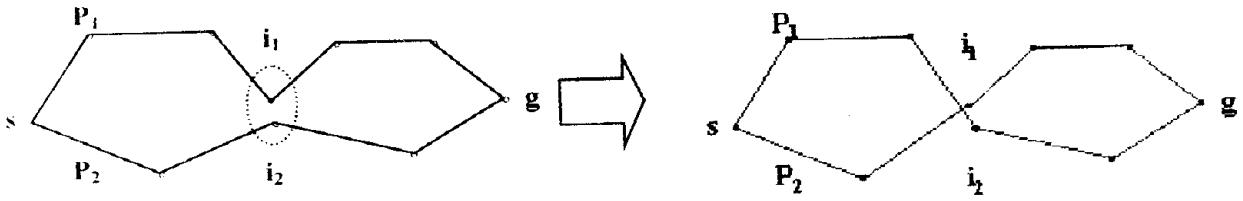


图 3 非对称单点杂交示意图

(a) 移动一个点

对于一条路径 R_i, 将其中穿越障碍物的线段的端点记录到 Rec 中. 如果 Rec 不为空, 在 Rec 中随机选取一个点进行移动; 否则, 则随机选取路径个体中一点进行移动.

(b) 删除一个点

对于路径 R_i, 若其中途点 p_i 与 p_{i+2} 之间连线没有穿越障碍物, 则将点 p_i 记录到 Rec 中. 如果 Rec 不为空, 则从 Rec 中随机选取一点进行删除. 否则, 随机选取路径中一点删除.

(c) 增加一个点

对于路径 R_i, 统计穿越障碍物的线段并记录到 Rec 中. 如果 Rec 不为空, 随机从 Rec 中取一段并在这一段之间增加一点; 否则, 随机选取路径 R_i 中一段并增加一个点.

(d) 替换一段路径

从基因库中随机选取一段路径, 随机从路径 R_i 中选取两点进行替换. 基因库的产生可以采用对障碍物的边缘进行跟踪, 生成若干小路径片段的方法.

以前的遗传路径规划算法用穿越障碍物的长度来度量路径个体的好坏, 如图 4 所示.

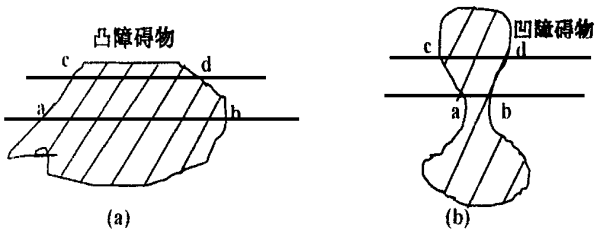


图 4 遗传路径规划

在障碍物是凸面体时, 路径如图 4(a) 所示, 此时可以明显看出路径 cd 优于路径 ab. 而对于凹多面体, 如图 4(b), 可以看出, 路径 cd 明显优于路径 ab, 因为路径 cd 更加容易翻越障碍物. 而此时路径 cd 穿越障碍物的长度大于路径 ab, 由此可见用穿越障碍物的长度来度量路径个体的优劣是不正确的. 如果我们采用翻越障碍物能力来作为度量路径个体的适应度函数, 此时无论图 4(a) 或图 4(b) 路径 cd 都优于路径 ab, 因为它很容易翻越障碍物. 因此我们考虑用路径如 cd 到障碍物两端的最小长度 l 来作为度量路径翻越能力的标准, l 越短, 路径翻越障碍物越容易. 为此得算法如下:

步骤:

$S = 0$;

对路径中的每一段, 执行以下循环:

{

如果线段没有穿越障碍物, 则 S 不变

否则:

计算从障碍物两端到线段的距离, 并求得最小值 S_{min};

如果障碍物两端没有靠近地图边界, 则

$S = S + S_{min}$;

否则:

设没有接触边界一端到线段的距离为 T_{min};

$S = S + T_{min}$;

}

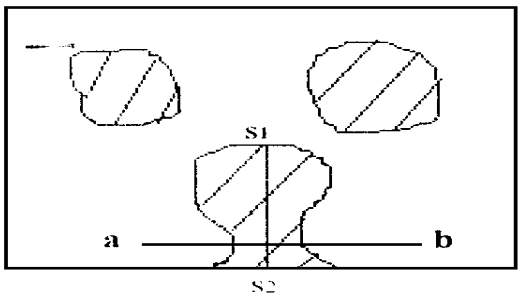


图 5 障碍物邻接边界

说明: 考虑障碍物一端接触边界的情况, 是为了防止出现如图 5 所示的情况. 此时只有从障碍物的上方 才能避开障碍物. 但此时 $s_1 > s_2$. 如果再采用 S_{min} 来衡量路径的好坏, 基于选择的压力, 进化的结果导致路径向下移动, 但向下路径无法翻越障碍物.

路径规划的首要问题是避障, 其次是要求路径的长度尽可能的短. 因此, 适应度函数中必须还包括

路径经过的长度. 对于路径个体

$R_i = \langle sv_1v_2 \ldots v_ng \rangle = \langle p_0p_1 \ldots p_{n+1} \rangle$

$\text{fitness}(R_i) = \omega_1 k1/(S+1) + \omega_2 k2(\text{sum} - d)/d$

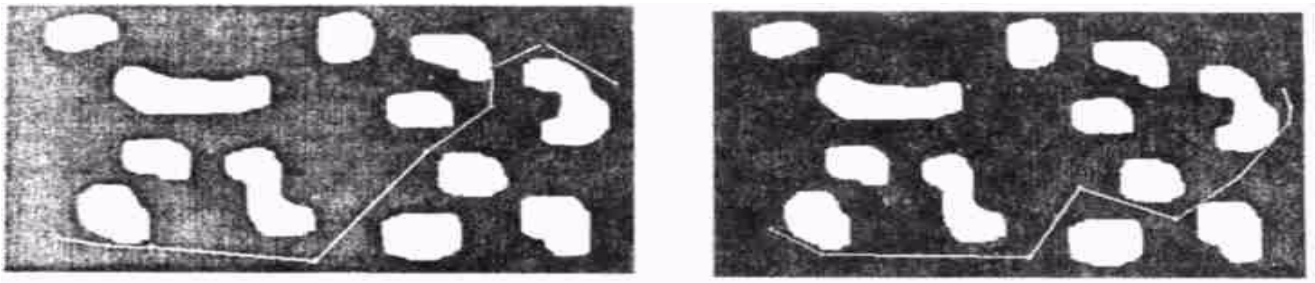
其中 S 是路径翻越障碍物的能力, $\text{sum} = \sum_{i=0}^{n} \text{length}(\overline{p_ip_{i+1}})$ 是路径长度. ω_1 与 ω_2 是衡量避障与路径长度之间的权重且 $\omega_1 + \omega_2 = 1$, $k1$, $k2$ 是比例系数, d 是从出发点到目标点的距离.

4　试验测试

试验测试包括算法可行性测试与性能测试. 可行性测试是测试算法是否能规划出一条路径, 性能测试则测试遗传算子对算法性能的影响.

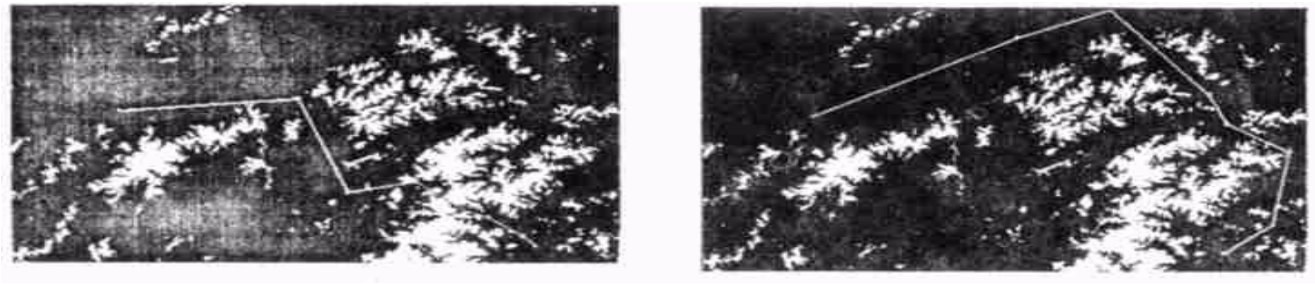
4.1　可行性测试

在 Win98 环境下用 Visual C++ 6.0 实现了上述算法. 采用简单与复杂环境来对比算法性能. 简单环境由人工构造, 复杂环境采用真实地形. 图中黑色是自由区域, 白色是障碍, 连线是进化后的路径. 图 7 是简单环境, 此时无论采用基本遗传规划算法或复杂环境下的遗传规划算法都可规划出一条可行路径. 但在复杂环境下使用基本遗传规划算法, 如图 7(a) 可见, 进化陷入局部极值点, 无法规划出一条可行路径. 图 7(b) 使用的是复杂环境下的遗传规划算法在进化 500 代后的结果, 可以看出, 复杂遗传规划算法效果很好, 可以规划出一条较好的路径.



(a) 基本遗传规划算法　　(b) 复杂环境下的遗传规划算法

图 6　简单环境下的路径规划



(a) 基本遗传规划算法　　(b) 复杂环境下的遗传规划算法

图 7　复杂环境下的路径规划

4.2　算法性能测试

评价遗传算法性能一般采用离线性能与在线性能, 前者测量算法的收敛性, 后者测量算法的动态性能. 在性能试验中, 采用复杂环境, 采用联赛选择加最佳个体保留策略的方法进行选择, 联赛规模 $k=2$, 路径中途点的个数为 15, 群体规模为 80, 交叉概率 0.75. 杂交采用单点非对称杂交, 变异采用启发式变异. 图中横坐标代表进化的代数, 纵坐标代表路径的代价.

4.2.1 变异概率对算法性能的影响

可以看出, 当 $P_m = 0.1$ 时算法的性能较好, P_m 过高或过算法性能都不好. P_m 过低达不到搜索未知空间的效果. 变异概率太大时趋于随机搜索性能较低. $P_m = 0.1$ 是一个较好的选择.

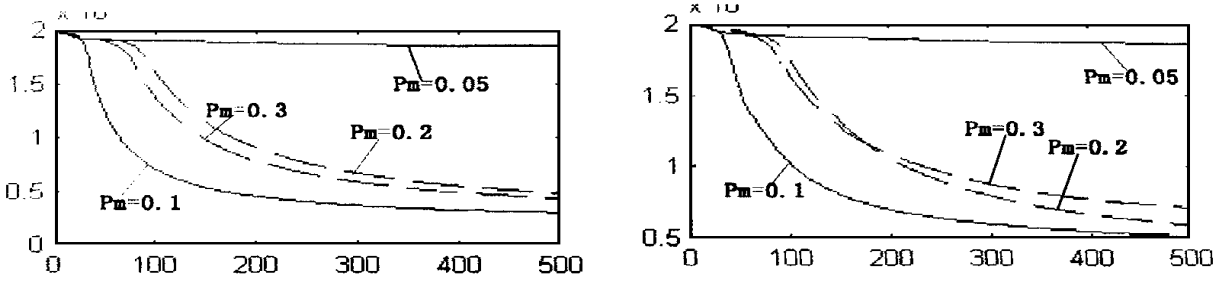
4.2.2 启发式变异与随机变异对算法性能的影响

从图 9 可以看出, 采用启发式变异的算法的性能明显优于随机变异. 这是因为在启发式变异中我们加入了对背景知识的运用, 避免了变异的盲目性.

4.2.3 基因库的运用对算法性能的影响

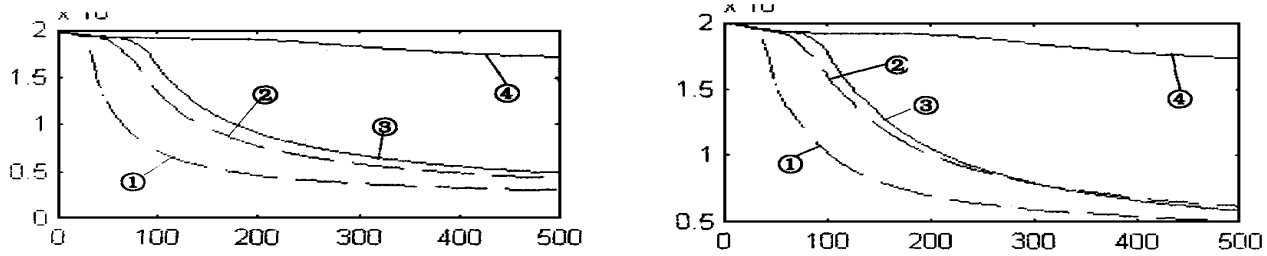
在其他测试条件保持不变的前提下, 我们测试了在变异中加入了基因库与取消基因库对算法性能的影响. 图 10 可以看出, 基因库操作对算法性能有显著的影响, 由于基因库是地图中较好的路径, 加入基因库操作就引入了对背景知识的利用, 大幅度提

高了算法的性能.



(a) 离线性能 (b) 在线性能

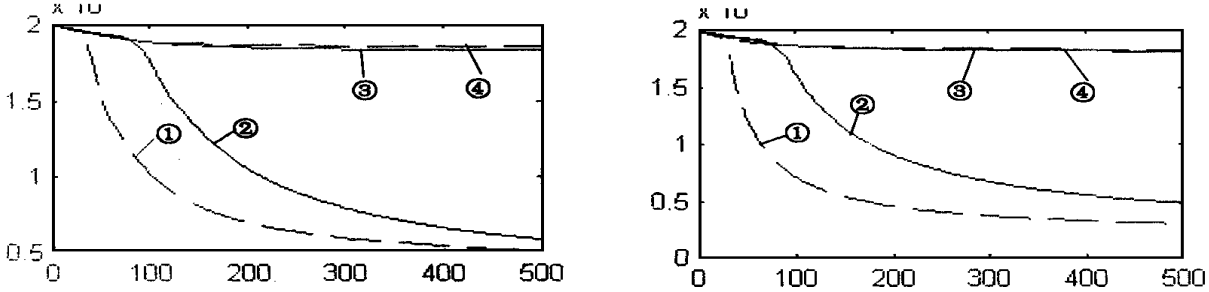
图 8 变异概率对算法性能的影响



(a) 离线性能 (b) 在线性能

①启发式变异, P_m= 0.1 ②启发式变异, P_m= 0.2 ③随机变异, P_m= 0.1 ④随机变异, P_m= 0.2

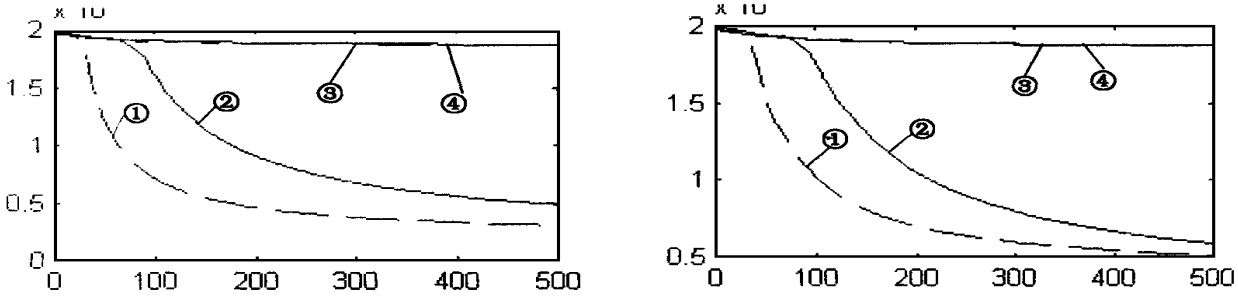
图 9 启发式变异与随机变异对算法性能的影响



(a) 离线性能 (b) 在线性能

①变异带基因库, P_m= 0.1 ②变异带基因库, P_m= 0.2 ③变异无基因库, P_m= 0.1 ④变异无基因库, P_m= 0.2

图 10 基因库操作对算法性能的影响



(a) 离线性能 (b) 在线性能

①新适应度函数, P_m= 0.1 ②新适应度函数, P_m= 0.2 ③旧适应度函数, P_m= 0.1 ④旧适应度函数, P_m= 0.2

图 11 适应度函数对算法性能的影响

(下转第 50 页)