文章编号: 1001-0920(2005)09-1052-04

基于粒子群优化算法的移动机器人全局路径规划

孙 波, 陈卫东, 席裕庚
(上海交通大学 自动化研究所, 上海 200030)

摘 要: 提出了一种基于粒子群优化算法的移动机器人全局路径规划方法. 该方法首先进行环境地图建模, 通过坐标变换在路径的起点与终点之间建立新地图, 然后利用粒子群优化算法获得一条全局最优路径. 该方法模型简单, 算法复杂度低, 收敛速度快, 而且模型不依赖于障碍物的形状. 仿真实验证实了该方法的有效性.

关键词: 移动机器人; 路径规划; 粒子群优化算法
中图分类号: TP24 **文献标识码**: A

Particle Swarm Optimization Based Global Path Planning for Mobile Robots

SUN Bo, CHEN Wei-dong, XI Yu-geng
(Institute of Automation, Shanghai Jiao Tong University, Shanghai 200030, China. Correspondent: CHEN Wei-dong, E-mail: wdchen@sjtu.edu.cn)

Abstract: A global path planning approach based on particle swarm optimization (PSO) is presented. The first step is to make a new map between starting-point and goal-point through coordinate system transferring. Then the PSO is introduced to get a global optimized path. This algorithm has a simple model, low complexity, rapid convergence and no restrict on the shapes of obstacles. Simulation results are provided to verify the effectiveness and practicability.

Key words: Mobile robots; Path planning; Particle swarm optimization (PSO)

1 引 言

路径规划是移动机器人研究的基本问题之一. 根据环境地图提供的信息, 路径规划可分为全局和局部方法两类, 其中全局路径规划需要在规划前获得完整的环境信息. 已有的全局路径规划方法包括启发式图搜索算法、人工势场法、可视图法、神经网络法等, 都已经得到了广泛的应用[1], 这些方法都具有各自的优点, 但均存在着一定的局限性[2]. 根据不同的环境特点和性能指标选取不同的算法是提高路径规划性能的一个有效途径, 它要求进一步丰富路径规划的方法, 不断引入新算法.

粒子群优化算法(PSO)是基于群体智能的一种进化计算方法, 由Eberhart 和Kennedy 于1995 年提出[3]. 与一般的进化算法相比, PSO 概念简单、容易实现并且需要调整的参数少[4], 目前广泛应用于函数优化、模式分类、优化调度、神经网络训练、模糊系统控制等领域. 秦元庆等[5]将PSO 引入移动机器人路径规划中, 开创了很好的先例, 但其存在一些不足. 文中首先使用Dijkstra 法获得链接图的最短路径, 然后利用PSO 对该路径中几个节点的位置进行优化. 这样不仅限制了PSO 的使用范围, 而且其最终结果有可能不是最优的路径. 为克服这种缺陷, 本文的算法将PSO 贯穿路径规划始终, 充分发挥PSO 的优点, 高速准确地进行路径规划.

本文首先进行了环境地图建模, 为PSO 应用提供条件; 然后设计了相应的PSO 算法实现; 最后进

收稿日期: 2004-09-20; **修回日期**: 2005-02-22
基金项目: 国家自然科学基金项目(60105005, 60475032).
作者简介: 孙波(1977—), 男, 山东临沂人, 博士生, 从事智能机器人控制与多机器人协作的研究; 陈卫东(1968—), 男, 黑龙江哈尔滨人, 副教授, 从事智能机器人、多机器人协调与合作等研究.

行了仿真实验, 取得了很好的效果

2　问题描述与建模

对于移动机器人, 路径规划就是寻找其在环境中移动时所必须经过的点的集合. 如图 1 所示, 在全局坐标系 O-XY 中, S 为机器人的出发点, G 为终点. 图中黑色实心填充的物体表示障碍. 机器人的路径规划即为寻找一个点的集合

$$P = \{S, p_1, p_2, \cdots, p_m, G\}, \tag{1}$$

其中 $(p_1, p_2, \cdots, p_m)$ 为全局地图中一个点的序列, 即规划目标. 对点 p_j 的要求是: p_j 为非障碍点, p_j 与相邻点的连线上不存在障碍点

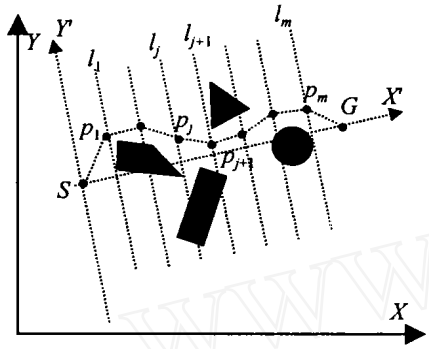


图 1　路径产生过程

在全局地图中建立一个新的坐标系, 以 SG 作为 X' 轴, 垂直于 X' 且经过 S 点的直线作为 Y' 轴, X', Y' 轴的方向如图 1 所示. 对应的坐标变换为

$$\begin{bmatrix} x \\ y \end{bmatrix}' = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_S \\ y_S \end{bmatrix}. \tag{2}$$

其中: (x, y), (x', y') 分别为地图中某一点在不同坐标系 O-XY 和 S-$X'Y'$ 下的坐标, α 为坐标轴 X 与 X' 的夹角, (x_S, y_S) 为 S 点在坐标系 O-XY 下的坐标

将线段 SG 进行 $(m+1)$ 等分, 在每一个等分点作垂线, 得到平行直线族 $(l_1, l_2, \cdots, l_m)$, 它们与路径 P 的交点即为目标点序列 $(p_1, p_2, \cdots, p_m)$. 定义 S 为 p_0, G 为 p_{m+1}, 路径 P 的长度 L_P 为

$$L_P = L_{Sp_1} + \sum_{j=1}^{m-1} L_{p_jp_{j+1}} + L_{p_mG} = \sum_{j=0}^{m} L_{p_jp_{j+1}}. \tag{3}$$

其中 $L_{p_jp_{j+1}}$ 表示点 p_j 与点 p_{j+1} 间的距离. 以 (x', y') 坐标表示, 式(3) 为

$$L_P = \sum_{j=0}^{m} \sqrt{(x'_{p_j} - x'_{p_{j+1}})^2 + (y'_{p_j} - y'_{p_{j+1}})^2} = \sum_{j=0}^{m} \sqrt{\left(\frac{L_{SG}}{m+1}\right)^2 + (y'_{p_j} - y'_{p_{j+1}})^2}. \tag{4}$$

最终路径规划转变为对式(3) 的函数进行优化, 即在 $y'_{p_j}(j = 1, 2, \cdots, m)$ 的取值空间中寻找最小的 L_P. 在这个模型描述中, 没有出现与障碍物有关的参数, 障碍物的信息体现在对 y'_{p_j} 的约束中, 即 L_P 在的计算过程中, y'_{p_j} 和 $y'_{p_{j+1}}$ 的连线(含两端点) 上不能存在障碍点

3　基于 PSO 的路径规划

3.1　PSO 算法

PSO 模拟鸟群的捕食行为, 采用速度 - 位置 $(v$-$x)$ 搜索模型. 每一个备选解称为一个粒子, 粒子的优劣程度由适应度函数 $F(x)$ 决定. 每一个粒子都有一个速度决定更新的方向和大小, 粒子们追随当前最优粒子通过迭代在解空间中搜索. 每一次迭代, 粒子通过跟踪两个极值更新自己的速度和位置: 粒子本身所找到的最优解 pBest 和整个种群目前找到的最优解 gBest. 定义种群中存在 n 个粒子, 每个粒子 m 维, 其速度和位置的更新方法为[6]

$$v_{i,j}^{t+1} = \omega v_{i,j}^{t} + c_1 r_1 \frac{(p_{i,j}^{t} - x_{i,j}^{t})}{\Delta t} + c_2 r_2 \frac{(p_{g,j}^{t} - x_{i,j}^{t})}{\Delta t}, \tag{5}$$

$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1}\Delta t \tag{6}$$

其中: $v_{i,j}^{t}$, $x_{i,j}^{t}$ 分别表示粒子 $i(i = 1, 2, \cdots, n)$ 第 j $(j = 1, 2, \cdots, m)$ 维分量在 t 时刻的速度和位置; $p_{i,j}^{t}$ 表示粒子 i 第 j 维分量到 t 时刻为止搜索到的最优位置; $p_{g,j}^{t}$ 表示种群中所有粒子第 j 维分量到 t 时刻为止搜索到的最优位置; Δt 为单位时间长度; r_1, r_2 为 (0～1) 之间的随机数; c_1, c_2 为加速常数, 表示每个粒子受 pBest 和 gBest 位置吸引的加速项的权重, 一般取 $c_1 = c_2 = 2$; ω 为惯性权重, ω 较大则算法具有较强的全局搜索能力, ω 较小则算法倾向于局部搜索, 一般是将 ω 随迭代次数线性减小[7], 即

$$\omega = \omega_{\max} - \text{iter} \cdot \frac{\omega_{\max} - \omega_{\min}}{\text{iter}_{\max}}. \tag{7}$$

其中: iter 为当前迭代次数, $\text{iter}_{\max}$ 为总的迭代次数, $\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$

3.2　算法实现

应用 PSO 算法解决优化问题的两个重要步骤是: 问题解的编码和适应度函数的选择

考虑图 1 中的平行直线族 $(l_1, l_2, \cdots, l_m)$, 如果将粒子的各维位置分量对应于直线上的点, 则不仅可找到问题解的表达, 而且粒子各维位置分量均具有明确的物理意义. 定义 n 个 m 维的粒子, 粒子每一维位置坐标 $x_{i,j}$ 取值在相应直线 l_j 上. 第 i 个粒子的适应度函数选为

$$F_i(x) = \sum_{j=0}^{m} \sqrt{d + (x_{i,j} - x_{i,j+1})^2}. \tag{8}$$

其中: i 为粒子序号, j 为维序号; $d = \left(\frac{L_{SG}}{m+1}\right)^2$ 为定

值; $x_{i,j}$ 等于式(4)中的 y'_{p_j}; 两端约束 $x_{i,0} = x_{i,m+1} = 0$; 边界约束 $x_j^{\min} \le x_{i,j} \le x_j^{\max}$, $x_j^{\min}$ 和 $x_j^{\max}$ 取值由平行直线族(l_1, l_2, …, l_m)和地图边界的交点决定

具体的 PSO 算法为:

Step 1: 根据 S, G 点的位置, 由式(2)进行坐标变换, 获得新地图. 以下各步均在该地图中进行, 得到最终路径后通过坐标反变换到原地图中, 反变换公式为

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} x' - x_S \\ y' - y_S \end{bmatrix}. \quad (9)$$

式(9)中各变量定义同式(2).

Step 2: 出于避障安全性的考虑, 对障碍进行膨胀

Step 3: 在 S, G 间进行(m + 1)等分, 在等分点做垂线(l_1, l_2, …, l_m). 由 l_j 和地图的交点求出 $x_j^{\min}$ 和 $x_j^{\max}$, 进而求出速度极值 $v_j^{\max}$. $v_j^{\max}$ 影响到对最优解的搜索能力: $v_j^{\max}$ 过大粒子会掠过最优解, $v_j^{\max}$ 过小粒子会陷入局部优值. 本文将其取为位置变化范围的 10%[8], 即

$$v_j^{\max} = (x_j^{\max} - x_j^{\min}) \times 0.1. \quad (10)$$

Step 4: 初始化粒子速度和位置 $v_{i,j}^0$, $x_{i,j}^0$

$$v_{i,j}^0 = -v_j^{\max} + r \cdot 2v_j^{\max}, \quad (11)$$

其中 r 为(0～1)之间的随机数

因为粒子位置的取值涉及到障碍物, $x_{i,j}^0$ 在初始化过程中不仅要考虑边界约束, 还要考虑避障. $x_{i,j}^0$ 计算流程如图 2 所示

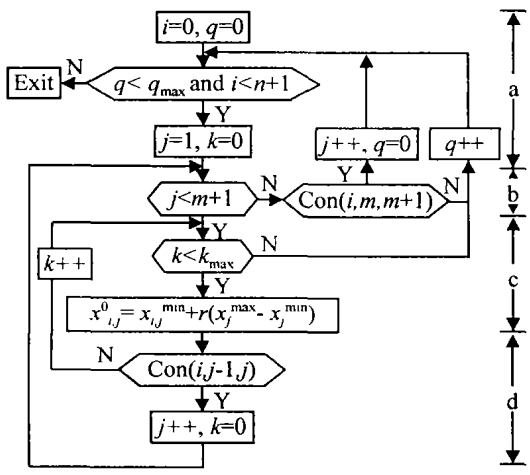


图 2 粒子位置初始化流程

q 和 k 为计数器. $q \ge q_{\max}$ 表示初始化失败, $k \ge k_{\max}$ 表示第 j 维 $x_{i,j}^0$ 进行的 k 次初始化尝试失败, 要从第 1 维 $x_{i,1}^0$ 重新开始. 函数 Con(i, j, j + 1)表示第 i 个粒子第 j 维分量和第(j + 1)维分量分别对应的地图中两个点之间是否为直线连接无障碍的, 特别 Con(i, m, m + 1)验证是否满足终端约束(算法从一端 $x_{i,0}^t = 0$ 开始迭代, 至 $x_{i,m+1}^t = 0$ 止). 整个流程可以分为 abcd 四部分: 循环控制、终端约束、$x_{i,j}^0$ 生成和障碍验证. 得到 $v_{i,j}^0$, $x_{i,j}^0$ 后, 根据式(8)求各粒子的适应度 $F_{i,j}^0$, 求出 $p_{i,j}^0$ 和 $p_{g,j}^0$.

Step 5: 由式(5)更新粒子各维的速度 $v_{i,j}^t$, 并进行边界约束(即超出边界[$-v_m^{\max}$, $v_j^{\max}$]的值以边界值代替). 由式(6)更新粒子各维的位置 $x_{i,j}^t$, 其更新流程与 $x_{i,j}^0$ 的更新流程相似, 也要考虑两端和障碍约束

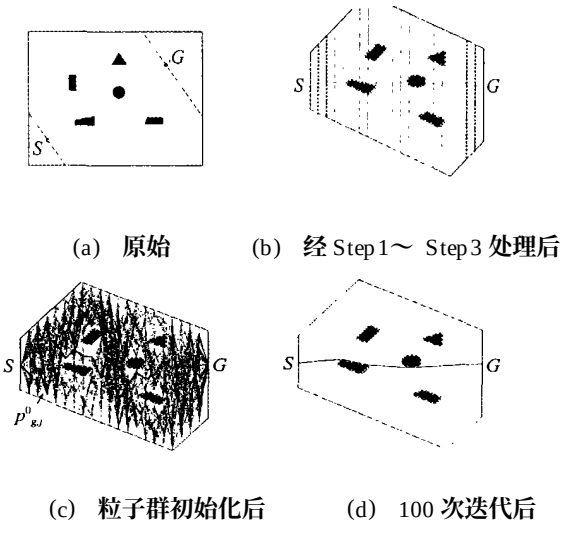
Step 6: 对每一个粒子, 由式(8)求其适应度 $F_{i,j}^t$, 更新 $p_{i,j}^t$ 和 $p_{g,j}^t$.

Step 7: 转 Step 5 进行迭代, 直到达到最大迭代次数 $\text{iter}_{\max}$ 或满足精度要求(收敛).

4 仿真研究

4.1 算法过程演示

如图 3 所示为路径规划的仿真全过程. 图 3(a)为原始地图, 图 3(b)为经 Step 1～Step 3 处理后的地图. 原始全局地图大小为 500 × 400. S (54, 325), G (394, 100)为起始点和目标点. n = 60, m = 20, α = 33.65°, L_{SG} = 407.71, 障碍物向外膨胀 4 格. 图 3(c)为粒子群初始化后地图, 图 3(d)为经 100 次迭代后地图. gBest^0 = 1 364.92, gBest^{100} = 413.27. 在第 54 步开始收敛. 重复实验 50 次, gBest^{100} 平均值为 415.14(区间[411.67, 418.11]), 一般在 50 步左右开始收敛



(a) 原始 (b) 经 Step 1～Step 3 处理后

(c) 粒子群初始化后 (d) 100 次迭代后

图 3 路径规划仿真过程

4.2 “陷阱”环境仿真

由第 3.2 节的算法实现过程可知, 本文的路径规划方法不依赖于障碍物的形状, 不需要对 S, G 之间的障碍物加以限制和约束. 这就为放置“陷阱”提供了可行性, 而“陷阱”环境正是许多路径规划算法在建模时就刻意避免的. 图 4 为仿真实验的环境地图和应用 PSO 算法寻找到的最优路径

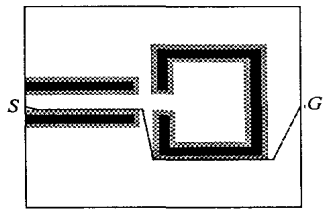


图 4 “陷阱”环境仿真结果

4.3 多步路径规划

第 3.2 节的算法对于某些环境存在着不足. 如图 5(a) 所示的情况, S, G 由于障碍物的包围使得其在坐标变换后的新地图中始终无法连通. 这种情况可用第 3.2 节算法分步实现, 具体的思路是: 双向等长度延长 SG 连线, 延长量为 d', 定义

$$d' = r \cdot \frac{L_{SG}}{m+1}. \tag{12}$$

其中 $r(r \geq 0)$ 为非负整数. 延长后的新端点定义为 M 和 M'. 在 SM (M) 和 M (M) G 均可连通的情况下, 进行两步规划. 连通的判断是第 3.2 节算法 Step 4 粒子初始化是否可以完成. 具体算法步骤为:

Step 1: 执行第 3.2 节 Step 4 粒子初始化, 如果 $q < q_{\max}$, 则继续执行(相当于 $r = 0$ 的多步规划); 否则, 表示初始化失败, 转入多步规划, $r = 1$.

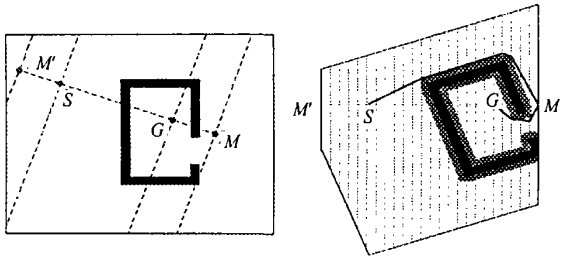
Step 2: 延长 SG 得到 MM'.

Step 3: 在 SM 与 MG 间分别进行粒子初始化. 如果可以实现, 则继续第 3.2 节算法求出 SM 与 MG 间的最优路径, 算法结束; 否则至本算法 Step 4.

Step 4: 在 SM' 与 MG 间再分别进行粒子初始化. 如果可以实现, 则继续第 3.2 节算法, 求出 SM' 与 MG 间的最优路径, 算法结束; 否则, 至本算法 Step 5.

Step 5: $r = r + 1$ 转本算法 Step 2 进行迭代, 直到 M 或 M' 超出地图范围时中止.

图 5(a) 和(b) 分别为初始地图和仿真结果. 图中 M 点是 G 点沿 SG 延长 $5 \times \frac{L_{SG}}{m+1}$ 找到的点.



(a) 初始地图　　(b) 仿真结果

图 5 多步路径规划仿真

4.4 算法性能

本文的基于 PSO 路径规划算法, 其复杂度为 $O(n^* m^* I)$, 取决于粒子群的规模(n)、粒子的维数(m) 和收敛时迭代的次数(I). 对图 4 所示地图, $I \approx 50$, 复杂度为 $O(60\,000)$. 如果采用 A^* 算法, 取节点数 $N = 50 \times 40 = 2\,000$, 其复杂度 $O(N^2 \log N)$ 远远超过 $O(n^* m^* I)$.

对第 4.1 节仿真, 50 次实验中, 完成 100 次迭代的平均时间为 0.39 s(区间[0.33, 0.45]), 完成 50 次迭代(平均收敛时刻) 时的平均时间为 0.22 s(区间[0.19, 0.27]). 仿真运行的硬件条件为: PC 机, CPU 奔四 1.6 GHz, 内存 DDR 266 512M. 软件环境为 Window 2000 和 Visual C++ 6.0. 在相同硬件、软件、同一地图和相同 S, G 点设置的条件下, 使用 A^* 算法进行规划(50 次实验) 所需要的平均时间为 0.42 s.

5 结　论

本文提出了一种基于 PSO 的移动机器人全局路径规划算法, 旨在丰富路径规划算法, 为移动机器人在特定的环境下规划路径提供一种有效的选择. 该算法主要具有以下 3 个特点: 1) 建模简单, 变量具有实际的物理意义, 需要调整的参数少; 2) 对起点与终点间的障碍未加区分和限制, 不受障碍形状的影响; 3) 计算简单, 容易实现, 并且收敛速度快.

参考文献(References)

[1] Zhang C G, Xi Y G. Robot Path Planning in Globally Unknown Environments Based on Rolling Windows[J]. *Science in China*, 2001, 44(2): 131-139.

[2] Henroch D. Fast Motion Planning by Parallel Processing —— A Review[J]. *J Intelligent and Robotic Systems*, 1997, 20(1): 45-69.

[3] Kennedy J, Eberhart R C. Particle Swarm Optimization [A]. *Proc of IEEE Intl Conf on Neutral Networks* [C]. Perth, 1995: 1942-1948.

[4] Angeline P J. Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Difference [A]. *Annual Conf Center on Evolutionary Programming* [C]. San Diego, 1998: 601-610.

[5] 秦元庆, 孙德宝, 李宁, 等. 基于粒子群算法的移动机器人路径规划[J]. *机器人*, 2004, 26(3): 222-225.
(Qin Y Q, Sun D B, Li N, et al. Path Planning for Mobile Robot Based on Particle Swarm Optimization [J]. *Robot*, 2004, 26(3): 222-225.)

[6] Gerhard V, Jaroslaw S S. Particle Swarm Optimization [A]. *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conf* [C]. Denver, 2002: 282-290.

(下转第 1060 页)

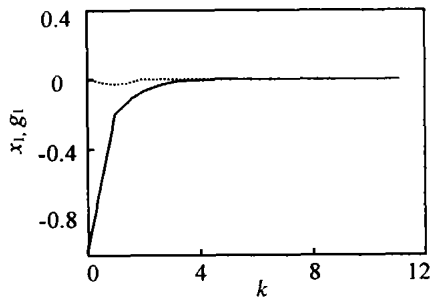


图 1 $x_1(k)$ 的状态响应

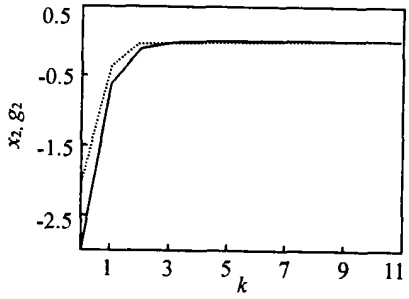


图 2 $x_2(k)$ 的状态响应

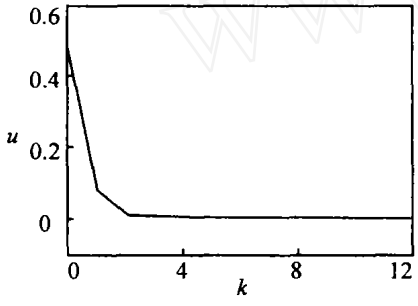


图 3 控制输入曲线 u

5 结 语

本文利用T-S 模型描述一类不确定离散非线性系统, 基于李亚普诺夫理论和线性矩阵不等式, 进行了稳定性分析并提出了模糊鲁棒控制器、观测器的设计方法. 该方法不仅考虑了系统状态不完全直接可测问题, 而且考虑了模糊模型的不确定性问题, 为离散非线性系统的控制提供了一条有效途径.

参考文献(References)

[1] Tseng C S, Chen B S. H_∞ Decentralized Fuzzy Model Reference Tracking Control Design for Nonlinear Interconnected Systems [J]. *IEEE Trans on Fuzzy Systems*, 2001, 9(6): 795-809.

[2] Zhang H G, Bien Z N. Adaptive Fuzzy Control Based on MIMO Nonlinear Systems [J]. *Fuzzy Sets and Systems*, 2000, 115(2): 191-204.

[3] Wu S J, Lin C T. Optimal Fuzzy Controller Design [J]. *IEEE Trans on Fuzzy Systems*, 2000, 8(2): 171-185.

[4] Tseng C S, Chen B S, Uang H J. Fuzzy Tracking Control Design for Nonlinear Dynamic Systems Via T-S Fuzzy Model [J]. *IEEE Trans on Fuzzy Systems*, 2001, 9(3): 381-392.

[5] Lee Hao Jae, et al. Robust Fuzzy Control of Nonlinear Systems with Parametric Uncertainties [J]. *IEEE Trans on Fuzzy Systems*, 2001, 9(2): 369-379.

[6] Hyeongcheol Lee, Masayoshi Tomizuka. Robust Adaptive Control Using a Universal Approximator for SISO Nonlinear Systems [J]. *IEEE Trans on Fuzzy Systems*, 2000, 8(1): 95-106.

[7] 张化光, 何希勤. ***模糊自适应控制理论及其应用*** [M]. 北京: 北京航空航天大学出版社, 2002: 244-247.
(Zhang H, He X Q. *The Theories and Applications of Adaptive Fuzzy Control* [M]. Beijing: University of Aviation and Aerospace Press, 2002: 244-247.)

(上接第 1055 页)

[7] Shi Y, Eberhart R. Parameter Selection In Particle Swarm Optimization [A]. *Proc of the 7th Annual Conf on Evolutionary Programming* [C]. New York, 1998: 591-600.

[8] Shi Y. Krohling R A. Co-evolutionary Particle Swarm Optimization to Solve Min-max Problems [A]. *Proc of the IEEE Cong on Evolutionary Computation* [C]. Honolulu, 2002: 1682-1687.