粒子群算法在三维空间机器人路径规划中的应用

魏代俊[1], 向长城[1,2]

(1. 湖北民族学院 数学系, 湖北 恩施 445000;
2. 重庆大学 自动化学院, 重庆 400030)

[摘要] 将粒子群优化算法应用到粒子群的路径规划中. 首先进行空间坐标变换, 然后将机器人所在的起始点与目的点的连线 *SD* 进行 $(n+1)$等分, 过每个等分点作垂直于 Z轴的 n个平面, 在每个平面上做正多边形的点阵, 构成粒子群算法寻优的路径空间, 最后应用于空间机器人的路径规划. 试验证明该方法在收敛速度和精度方面有效.

[关键词] 粒子群, 路径规划, 机器人
[中图分类号] TP24 [文献标识码] A [文章编号] 1672-1292(2008)03-0077-05

Path Planning in 3-D Space for Robot Based on Particle Swarm Optimization

Wei Daijun[1], Xiang Changcheng[1,2]

(1. Department of Mathematics, Hubei Institute for Nationalities, Enshi 445000, China;
2. Automatic College, Chongqing University, Chongqing 400030, China)

Abstract: The particle swarm optimization is applied to path planning problem in 3-D space for robot in this paper. First of all, the origin reference frame is transformed the new reference frame by the space coordinate transform; The *SD* which is connected the origination spot and the objective spot of the robot is divided into $n+1$ dot, the n vertical space to Z line is taken at the uniform dot, and the lattice of regular polygon is distributed in each space, and all lattices of regular polygon are constructed on the search space of the PSO. Finally, the PSO is applied in the 3-D space path planning, and the experiment results proved that the algorithm is effective in convergence speed and precision.

Key words: particle swarm optimization, path planning, robot

目前, 国内外有许多研究机器人路径规划的算法[1-4]. 这些方法大多应用于二维空间, 部分算法推广到了三维空间. 实际上, 由于三维空间特有的时空复杂性, 对二维路径规划算法简单地加以推广, 效果并不理想. 针对三维空间中的路径规划问题, 目前常采用的方法包括 Warren C W 及孙茂相等使用的人工势场法[1,2]、Carol K P提出的 *A** 搜索[3]、Vasudevan及尚游等基于案例的推理算法[4,5]等. 但具体分析发现, 势场法中不可避免地会陷入局部最小, 而且当采用复杂的优化准则时, 势场法不能直接加以推广; *A** 搜索算法随着维数的增加, 时空要求很难得到满足; 基于案例推理的途径根据局部的障碍物调整路径, 有时不能获得全局最优的路径.

三维空间机器人路径规划的任务是在具有障碍物的环境中, 按照一定的评价标准, 寻找一条从起始位置到目的位置的无碰路径. 解决该问题的常规方法由两部分组成: 建立一个数据结构来代表工作空间的几何结构; 搜索该数据结构以找到一条无碰路径. 常见的有可视顶点图法 (Visibility Graphic)[6]和 Voronoi图法[7]等. 但这些方法需要大量的时间来建立和搜索数据结构, 因此不适用于存在运动障碍物的在线路径规划. 而胡小兵等将蚂蚁算法 (ACO)应用到机器人三维空间的路径规划中, 郝燕玲等将遗传算法应用到水下机器人[8], 仍然存在迭代次数多、耗费时间长的缺陷. 而粒子群算法由于其算法简单、运算时间快, 受

收稿日期: 2007-03-18
基金项目: 湖北省青年基金 (Q200729003)和湖北民族学院青年基金 (myq2006041)资助项目.
通讯联系人: 魏代俊, 硕士, 研究方向: 数学建模与算法设计. E-mail: eswdj@163.com

到了越来越多的研究者的注意. 孙波等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, 该方法进行环境地图建模, 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[10]. 赵先章等针对结构化环境中移动机器人路径规划问题, 提出了一种基于粒子群的路径规划算法[11], 算法利用适应度函数描述环境约束及路径的距离信息, 由路径节点构成粒子, 通过混合粒子群算法进行寻优. 本文在空间坐标变换后将机器人所在的起始点与目的点的连线 SD 进行 $(n+1)$等分, 过每个等分点, 作垂直于 Z 轴的 n 个平面, 在每个平面上做正多边形的点阵, 构成粒子群算法寻优的路径空间, 最后利用粒子群算法解决三维空间机器人路径的规划, 得到了比较理想的寻优速度和精度.

1　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 $O-XYZ$ 中, S为机器人的出发点, D为终点. $O_1, O_2, \cdots, O_k$表示障碍. 为简单起见, 设障碍物为球形, 可表示为: $\{(o_i, r_i) \mid i = 1, 2, \cdots, k\}$, 其中 o_i、r_i 分别表示第 i个障碍物的球心位置和半径长度. 故机器人的路径规划即寻找一个点集合 $P = \{S, p_1, p_2, p_3, \cdots p_n, D\}$, 其中 $\{p_1, p_2, p_3, \cdots p_n\}$ 即为规划目标, 它是全局地图中一个点的序列. 点 $p_j\ (j = 1, 2 \cdots n)$ 的特征为: p_j 为非障碍点, p_j 与相邻点的连线上不存在障碍点. 图 2为在全局路径规划中所示的笛卡尔坐标系 $O'-X'Y'Z'$, 其中 S点为新坐标系的原点, SD 为 Z'轴的正方向, X'轴和 Y'轴可适当选择.

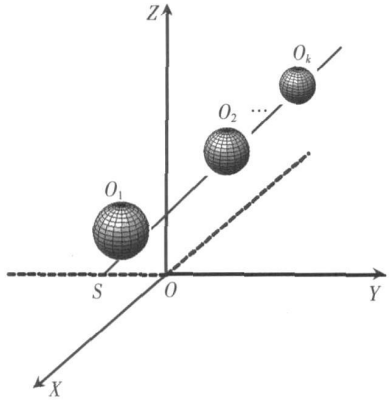

图 1　空间障碍物在坐标系 $O-XYZ$ 中

Fig.1　$O-XYZ$ space frame of axes

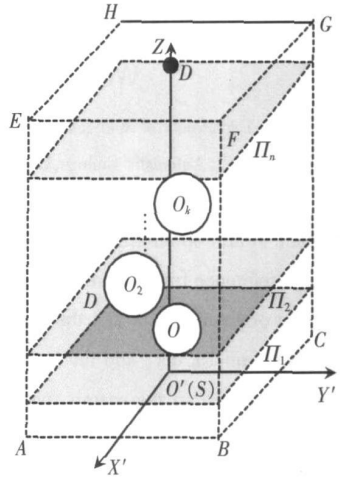

图 2　空间障碍物在坐标系 $O-X'Y'Z'$ 中

Fig.2　$O-X'Y'Z'$ space frame of axes

坐标系 $O'-X'Y'Z'$ 与 $O-XYZ$ 之间的变换关系为:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos\alpha_x & \cos\alpha_y & \cos\alpha_z \\ \cos\beta_x & \cos\beta_y & \cos\beta_z \\ \cos\gamma_x & \cos\gamma_y & \cos\gamma_z \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}, \tag{1}$$

式中, α_x, β_x, γ_x 分别为 X 轴与 X', Y', Z'轴的夹角; α_y, β_y, γ_y 分别为 Y轴与 X', Y', Z'轴的夹角; α_z, β_z, γ_z 分别为 Z轴与 X', Y', Z'轴的夹角.

由式 (1) 可计算出障碍物 $\{(o_i, r_i) \mid i = 1, 2, \cdots, k\}$ 在坐标系 $O'-X'Y'Z'$下的坐标, 设为 $\{(o'_i, r'_i) \mid i = 1, 2, \cdots, k\}$. 因 SD 的长度为 h, 故 D 在坐标系 $O'-X'Y'Z'$ 下的坐标为 $(0, 0, h)$.

在图 2中作立方体 $ABCDEFGH$, 其中 $ABCD$ 面在 $X'O'Y'$ 平面上, 是边长为 $2L$的正方形平面, AB 平行于 Y'轴, BC 平行于 X'轴, 原点 O'在正方形 $ABCD$ 的中心上, 立方体的高 $AE = h$, 定点 H 的坐标为 $(-L, -L, 0)$. 将 SD 进行 $n+1$等分, 过每个等分点, 作垂直于 Z'轴的 n个平面 $\Pi_i\ (i = 1, 2, \cdots, n)$. 以平面 Π_i与 SD 的交点为圆心作一个正六边形的点阵, 如图 3所示.

点阵以交点 $\left(0, 0, \frac{i \cdot h}{n}\right)$ 为圆心, 分别以 $\frac{2L}{m}$, $2\left(\frac{2L}{m}\right)$, $3\left(\frac{2L}{m}\right)$, …, $m\left(\frac{2L}{m}\right)$ 为半径的圆与正多边形的交点集

合, 第 i层平面第 k个正多边形的第 j个顶点在 Π_i (i = 1, 2, …, n) 平面上的极坐标为 $P^i(k_i, j_i) = \left(k\left(\frac{2L}{m}\right), j^*\left(\frac{2\pi}{6k}\right)\right)$, 对应在坐标系 $O'-X'Y'Z'$中的实际坐标为 $(x_{kj}^i, y_{kj}^i, z_{kj}^i) = \left(\frac{2L}{m}\cos\left(j^*\frac{2\pi}{6k}\right), \frac{2L}{m}\sin\left(j^*\frac{2\pi}{6k}\right), \frac{i^*h}{n}\right)$. 其中 j = 1, 2, …, $6k$; k = 1, 2, …, m; i = 1, 2, …, n, 点计数的顺序从 x 轴方向的点开始逆时针方向依次计数. 实际应用中, 还可以做正 12 边形或者 24边形的点阵, 这样平面的点都可以取得, 与平面的网格法比较, 此法取得的点更均匀、合理.

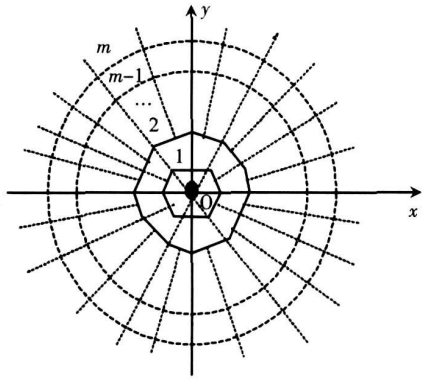

图 3 圆与正多边形的交点点阵集合

Fig.3 Node lattice set of circular and regular polygon

2 基于粒子群算法的路径规划

2.1 *PSO* 算法的基本思想

粒子群最佳化算法 (PSO) 是一种以族群动力学为基础的演化式计算法则[9], 算法中每一个个体在搜寻空间中各自拥有方向和速度, 并且能根据自我过去经验与群体行为进行机率式的搜寻策略调整. 算法开始以随机方式散布粒子, 依各粒子所得的适合度 (fitness) 来进行演化. 其基本算法如式 (2) 所示:

$$\boldsymbol{v}_{id}(t+1) = \omega^* \boldsymbol{v}_{id}(t) + \alpha^* \text{rand}()^* (p_{id} - \boldsymbol{x}_{id}(t)) + \beta^* \text{rand}()^* (p_{g,d} - \boldsymbol{x}_{id}(t)),$$
$$\boldsymbol{x}_{id}(t+1) = \boldsymbol{x}_{id}(t) + \boldsymbol{v}_{id}(t+1), \quad (2)$$

式中, d 是搜寻空间中变量的维度, rand( ) 是一个介于 0与 1之间的随机数, i是群体中的个体, $\boldsymbol{v}_i$ 是速度向量, $\boldsymbol{x}_i$ 是位置向量. p_{id} 是个体所经历过之最佳解位置, p_{gd} 是个体所处之整个邻域所记录的最佳解位置, 参数 α以及 β分别是自我认知与社会模式的学习率, ω为惯性权重. 在每一次迭代当中, 粒子 x_i根据个体过去曾经浏览过的最佳向量值 $\boldsymbol{P}_{\text{best}i}$以及整个群体曾经浏览过的最佳向量值 $\boldsymbol{G}_{\text{best}}$, 计算出粒子的位移向量 $\boldsymbol{v}_i$, 然后再将粒子向量加上此位移向量作为下一次迭代的粒子向量值. 由公式可知 $\boldsymbol{x}_i$向 $\boldsymbol{P}_{\text{best}i}$及 $\boldsymbol{G}_{\text{best}}$的合成向量方向移动, 以增加改善 $\boldsymbol{x}_i$的机会, 同时又设计了随机的干扰量, 以避免掉入局部最佳解. 所以粒子的向量修正原则是取得钻探搜寻以及探索搜寻的平衡, 符合现代经验法则的精神.

2.2 算法实现

有效路径的生成: 在 $O'-X'Y'Z'$ 坐标系中, 机器人从 S(原点) 出发, 首先到达平面 Π_1 上的某点 $P^1(k_1, j_1)$, $k_1 \in \{1, 2, \cdots, m\}$, $j_1 \in \{1, 2, \cdots, 6k_1\}$, 然后再从 $P^1(k_1, j_1)$ 点出发到达平面 Π_2 上的某点 $P^2(k_2, j_2)$, $k_2 \in \{1, 2, \cdots, m\}$, $j_2 \in \{1, 2, \cdots, 6k_2\}$, 依次最后到达平面 Π_{n-1} 上 $P^{n-1}(k_{n-1}, j_{n-1})$ 点, $k_n \in \{1, 2, \cdots, m\}$, $j_n \in \{1, 2, \cdots, 6k_{n-1}\}$. 连接 $P^n(k_n, j_n)$ 与 D 点, 便构成了一条从源点 S 到目的点 D 的路径 $S \rightarrow P^1(k_1, j_1) \rightarrow \cdots \rightarrow P^n(k_n, j_n) \rightarrow D$. 路径的长度: $L_{SD} = L_{SP1} + \sum_{i=1}^{n} L_{P^iP^{i+1}} + L_{P^nD}$, 其中 L_{SP1} 表示 S 到 $P^1(k_1, j_1)$ 的距离, $L_{P^iP^{i+1}}$表示 $P^i(k_i, j_i)$ 到 $P^{i+1}(k_{i+1}, j_{i+1})$, L_{P^nD} 表示 $P^n(k_n, j_n)$ 到 D 的距离. 如果把 S 看成 P^0, D 看成 P^{n+1}, 则 $L_{SD} = \sum_{i=0}^{n+1} L_{P^iP^{i+1}}$, 其中 $L_{SD} = \sum_{i=0}^{n+1} \sqrt{(x_{kj}^i - x_{kj}^{i+1})^2 + (y_{kj}^i - y_{kj}^{i+1})^2 + (z_{kj}^i - z_{kj}^{i+1})^2}$, 即 $L_{SD} = \sum_{i=0}^{n+1} \sqrt{(x_{kj}^i - x_{kj}^{i+1})^2 + (y_{kj}^i - y_{kj}^{i+1})^2 + \left(\frac{h}{n+1}\right)^2}$.

最优路径规划转化为对上述距离的最优化问题, 但是由于 S 与 D 之间存在障碍物, 因此某些路径可能是无效的. 为了尽量生成有效路径, 就要消除平面上障碍点, 因此为平面 Π_i (i = 1, 2, …, n − 1) 上的点 $P^i(k_i, j_i)$ 引入一个允许列表 allowed$^i(u, v)$. allowed$^i(u, v)$ 为平面 Π_{i+1} 上点集的子集, 且连接 $p^i(u, v)$ 与 allowed$^i(u, v)$ 中的任何点都不会穿过障碍物.

允许列表 allowed$^i(k, j)$的计算方法[8]: 设 $P^i(k_i, j_i)$为平面 Π_i (i = 1, 2, …, n) 上的某一点, 现计算该

点的 allowed$^i(u, v)$列表. 对平面 Π_{i+1}上的任意点 $P^{i+1}(k_{i+1}, j_{i+1})$, 如果线段 $P^i(k_i, j_i)P^{i+1}(k_{i+1}, j_{i+1})$不与任何障碍物相交, 则将 $p^{i+1}(k, l)$点加入到 allowed$^i(k, j)$中. 按照此方法, 可以计算出点 $P^i(k_i, j_i)$的所有允许到达的点, 并将其存于 allowed$^i(k, j)$中.

空间粒子的表示: 按照空间路径的生成方法, 每个路径上的点表示组成一个粒子 $\{S, P^1(k_1, j_1), \cdots, P^n(k_n, j_n), D\}$, 而在空间中, 每个点由三维空间坐标组成. 由路径长度的计算公式知, 只需要考虑路径点 $P^i(k_i, j_i)$在平面 $\Pi_i(i = 1, 2, \cdots, n+1)$横坐标 x^i_{kj}和纵坐标 y^i_{kj}的变化, 因为横坐标 x^i_{kj}和纵坐标 y^i_{kj}的变化就可以确定 $P^i(k_i, j_i)$新的位置. 第 i个粒子位置和速度可用式(3)表示:

$$\boldsymbol{X}_i = \begin{bmatrix} 0 & x^1_{k_1j_1i} & x^2_{k_2j_2i} & \cdots & x^{n-1}_{k_{n-1}j_{n-1}i} & 0 \\ 0 & y^1_{k_1j_1i} & y^2_{k_2j_2i} & \cdots & y^{n-1}_{k_{n-1}j_{n-1}i} & 0 \end{bmatrix}, \quad \boldsymbol{V}_i = \begin{bmatrix} 0 & vx^1_{k_1j_1i} & vx^2_{k_2j_2i} & \cdots & vx^{n-1}_{k_{n-1}j_{n-1}i} & 0 \\ 0 & vy^1_{k_1j_1i} & vy^2_{k_2j_2i} & \cdots & vy^{n-1}_{k_{n-1}j_{n-1}i} & 0 \end{bmatrix}, \tag{3}$$

式中, $k_1 \in \{1, 2, \cdots, m\}$, $j_1 \in \{1, 2, \cdots, 6k_1\}$; $k_2 \in \{1, 2, \cdots, m\}$, $j_2 \in \{1, 2, \cdots, 6k_2\}$; …; $k_n \in \{1, 2, \cdots, m\}$, $j_n \in \{1, 2, \cdots, 6k_{n-1}\}$. $\boldsymbol{X}_i$用一个矩阵来表示, 其第一行表示所采路径点的横坐标, 第二行为相应点的纵坐标. 流程如下:

步骤 **1**　初始化

(1) 按式(1)计算出各障碍物在坐标系 $O'-X'Y'Z'$中的坐标;

(2) 作 SD和平面 $\Pi_i(i = 1, 2, \cdots, n)$的交点, 以交点圆心, 分别以 $\frac{2L}{m}$, $2\left(\frac{2L}{m}\right)$, $3\left(\frac{2L}{m}\right)$, …, $m\left(\frac{2L}{m}\right)$为半径的圆与正多边形的交点的点阵集合;

(3) 计算平面 $\Pi_i(i = 1, 2, \cdots, n-1)$上所有点的允许列表 allowed$^i(k, j)$, $j = 1, 2, \cdots, 6k$; $k = 1, 2, \cdots, m$; $i = 1, 2, \cdots, n$;

(4) 粒子群初始化. 随机分配每个粒子的初始位置和速度, 取粒子的维数为 $n+1$, 其中第 1维和第 $n+1$维分别为点 S和点 D, 它们在粒子的速度和位置更新过程中保持不变. 其它维的粒子分别在平面 $\Pi_i(i = 1, 2, \cdots, n-1)$上所有点的允许列表 allowed$^i(k, j)$, $j = 1, 2, \cdots, 6k$; $k = 1, 2, \cdots, m$; $i = 1, 2, \cdots, n$中随机选取. 以此保证随机产生的粒子所代表的路径不可能碰到障碍物.

步骤 **2**　按照粒子群更新的公式对每代种群的第 i个粒子的第 d维向量进行更新如下:

$$\boldsymbol{V}^d_{k_dj_di}(t+1) = \omega * \boldsymbol{V}^d_{k_dj_di}(t) + \alpha * \text{rand}() * (p_{i,d} - \boldsymbol{X}^d_{k_dj_di}(t)) + \beta * \text{rand}() * (p_{g,d} - \boldsymbol{X}^d_{k_dj_di}(t)),$$

$$\boldsymbol{X}^d_{k_dj_di}(t+1) = \boldsymbol{X}^d_{k_dj_di}(t) + \boldsymbol{V}^d_{k_dj_di}(t+1),$$

$$vx^d_{k_dj_di}(t+1) = \omega * vx^d_{k_dj_di}(t) + \alpha * \text{rand}() * (p_{i,d} - x^d_{k_dj_di}(t)) + \beta * \text{rand}() * (p_{g,d} - x^d_{k_dj_dik_dj_di}(t)),$$

$$vy^d_{k_dj_di}(t+1) = \omega * vy^d_{k_dj_di}(t) + \alpha * \text{rand}() * (p_{i,d} - y^d_{k_dj_di}(t)) + \beta * \text{rand}() * (p_{g,d} - y^d_{k_dj_dik_dj_di}(t)).$$

速度更新的范围为正多边形的点阵范围, 以使粒子不会跑到点阵外面. 其中 $\boldsymbol{V}^d_{k_dj_di}(t+1) = \begin{bmatrix} vx^d_{k_dj_di}(t+1) \\ vy^d_{k_dj_di}(t+1) \end{bmatrix}$, $\boldsymbol{X}^d_{k_dj_di}(t+1) = \begin{bmatrix} x^d_{k_dj_di}(t+1) + vx^d_{k_dj_di}(t+1) \\ y^d_{k_dj_di}(t+1) + vy^d_{k_dj_di}(t+1) \end{bmatrix}$, $\boldsymbol{V}^d_{k_dj_di}(t)$为 t代第 i个粒子 d维坐标分别为 k_dj_d的速度, $\boldsymbol{X}^d_{k_dj_di}(t)$为其对应的位置. $vx^d_{k_dj_di}(t)$, $vy^d_{k_dj_di}$分别为该粒子对应在 X方向和 Y方向的速度. $x^d_{k_dj_di}(t)$、$y^d_{k_dj_di}(t)$分别为粒子在 X方向和 Y方向的位置分量

步骤 **3**　对更新后的粒子检查是否遇到障碍物, 即检查更新后的粒子是否在相应的允许列表 allowed$^i(k, j)$中, $j = 1, 2, \cdots, 6k$; $k = 1, 2, \cdots, m$; $i = 1, 2, \cdots, n$. 如果允许列表不为空, 选择下一个平面上的点, 否则该粒子失散; 然后添加通过补充新的粒子添加到粒子群中, 进行优化.

步骤 **4**　对每个粒子进行适应值(即 L_{SD})的计算, 更新个体粒子最优位置和粒子群整体的最优位置.

步骤 **5**　如果搜索到最优路经, 并且连续不再发生变化, 则结束循环输出最优解; 否则转至步骤 2

3　仿真试验

本文实验直接考虑坐标系 $O'-X'Y'Z'$中的情形. 例 1[8]设 3个障碍物 O_1, O_2, O_3的球心坐标和半径分别为: {(20, 25, 60): 30}、{(30, 25, 95): 40}、{(46, 50, 260): 86}, 目的点 D的坐标为(0, 0, 400). $n = 15$, $m = 15$, $L = 150$, 其中粒子群算法的参数设置为 $\omega = 2$, $\alpha = 2$, $\beta = 2$, 粒子数 50

在 PIII 1GMHz, 256M 的 Windows XP 系统中用 Matlab 7.1对实验进行仿真, 算法迭代 578次以后得最优解长度为 488.94, 其路径如图 4所示.

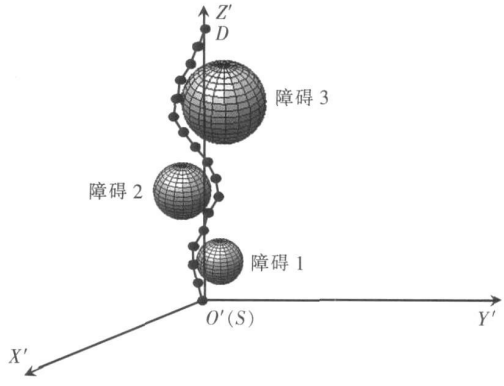

图 4 算法规划出的路径

Fig.4 Planning path based on PSO

4 结论

本文将 PSO 应用于机器人路经规划问题, 提出了一种求解三维空间机器人路径规划问题的 PSO 算法. 该算法首先将机器人所在位置与将要到达的目的地之间的空间用 $n-1$ 个平面划分, 然后在每个平面上构造正多边形点阵, 并将其极坐标表示方式转化为三维空间表示, 同时定义原点与目的点之间的有效路径. 然后用粒子群算法表示这些路径, 根据定义的距离函数作为适应值函数. 实验结果表明, 该算法不仅有效, 而且具有极快的速度. 在该算法中, 多边形点阵的稠密程度决定了算法求解的精度. 空间网格越稠密, 算法的精度越高, 所需时间也越长; 反之则越低, 所花时间越短. 在实际应用中, 可根据对精度的要求, 选用适当的网格的稠密程度.

[参考文献] (References)

[1] Warren C W. A technique for autonomous underwater vehicle route planning [J]. IEEE J of Oceanic Engineering, 1990(15): 199-204.

[2] 孙茂相, 王艳红, 吴学曼, 等. 动态补偿的水下机器人路径规划 [J]. 机器人, 1993(2): 8-12.
Sun Maoxiang, Wang Yanhong, Wu Xueman, et al. Underwater robot path planning with dynamical [J]. Robot, 1993(2): 8-12. (in Chinese)

[3] Carroll K P, McClaran S R, Nelson E L, et al. AUV path planning: an A* approach [C] // Proc Symp on AUV Technology. Washington D C, 1992: 79-84.

[4] Vasudevan C, Ganesan K. Case2based path planning for autonomous underwater vehicles [J]. Autonomous Robots, 1996(3): 79-89.

[5] 尚游, 刘百顺, 张万春, 等. 基于案例的自主式水下机器人全局路径规划的学习算法 [J]. 哈尔滨工程大学学报, 1998, 19(5): 1-7.
Shang You, Liu Baishun, Zhang Wanchun, et al. Autonomous underwater vehicles global path planning using case-based learning algorithm [J]. Journal of Harbin Engineering University, 1998, 19(5): 1-7. (in Chinese)

[6] Lee S, Park J. Neural computation for collisionfree path planning [J]. Journal of Intelligent Manufacturing, 1991(2): 315-326.

[7] Meng Qinghao, Peng Shangxian, Liu Dawei. Mobile robot global path planning using heuristic search on the Q-M graph [J]. Robot (in Chinese), 1998, 20(4): 273-279.

[8] 郝燕玲, 张京娟. 基于遗传算法的 AUV 三维海底路径规划 [J]. 中国工程科学, 2003, 5(11): 56-60.
Hao Yanling, Zhang Jingjuan. AUV path planning in 3D seabed environment using genetic algorithm [J]. Engineering Science, 2003, 5(11): 56-60. (in Chinese)

[9] Kennedy J, Eberhart R C. Particle swarm optimization [C] // Proc IEEE Conf on Neural Networks. Piscataway, N J, 1995: 1 942-1 948.

[10] 孙波, 陈卫东, 席裕庚. 基于粒子群优化算法的移动机器人全局路径规划 [J]. 控制与决策, 2005, 20(9): 1 052-1 054.
Sun Bo, Chen Weidong, Xi Yugeng. Particle swarm optimization based global path planning for mobile robots [J]. Control and Decision, 2005, 20(9): 1 052-1 054. (in Chinese)

[11] 赵先章, 常红星, 曾隽芳, 等. 一种基于粒子群算法的移动机器人路径规划方法 [J]. 计算机应用研究, 2007, 24(3): 181-183.
Zhao Xianzhang, Chang Hongxing, Zeng Junfang, et al. Path planning method for mobile robot based on particle swarm algorithm [J]. Application Research of Computers, 2007, 24(3): 181-183. (in Chinese)

[责任编辑: 严海琳]