基于遗传算法的机器人任务优化调度研究

翟 帆 [1], 谢显华 [2]

(1. 郑州升达经贸管理学院 基础部, 河南 郑州 451191)
(2. 赣南师范大学 数学与统计学院, 江西 赣州 341000)

摘 要: 工业机器人应该在尽可能短的周期内完成复杂的任务, 以获得高生产率. 确定机械手末端执行器访问多个任务点的最佳路径的问题与著名的旅行商问题 (TSP) 相似, 但并不完全相同. 为了使 TSP 适应机器人技术, 需要优化的度量是时间而不是距离. 另外, 任意两点之间的行程时间受机械手构型的选择影响较大. 因此, 需要考虑运动学逆问题的多重解. 逆运动学问题的求解是机器人控制的基础. 许多传统的逆运动学问题的解决方法, 如几何、迭代和代数方法, 对冗余机器人来说是不够的. 以此为出发点, 基于遗传算法提出了一种新的编码方法来考虑逆运动学问题的多重解, 方法适用于任何非冗余度的机械手. 仿真实验结果显示, 提出的方法能在一定的时间内快速找到最优解或近似最优解. 且任何非冗余度机械手的多重配置很容易体现在遗传算法的编码中.

关键词: 遗传算法; 仿真平台; 任务优化; 机器学习

1 引言

机器人目前被应用于工业和医疗领域, 这些领域对操作的准确性、重复性和稳定性都有很高的要求 [1]. 随着现代制造技术的发展, 微装配在某些敏感任务上也变得越来越重要, 如半导体加工装配、小零件装配、精密材料加工等 [2−5]. 尽管许多工业机器人的操作精度很高, 但为了进一步提高精度, 机器人规划和操作的研究仍在进行中. 特别是在夹具、执行机构控制系统和视觉系统的设计等领域, 以便在微装配中实现更精确的机器人应用. 为使机器人能在最短的时间和高精度内完成复杂的任务, 因而目前机器人研究的目标之一是使机器人在考虑到其的物理特性所施加的限制条件下尽可能快地执行所需任务. 然而, 在许多工业操作中, 机械手 “访问” 多个任务点并返回起点的顺序不是预先确定的, 因而, 需要确定最优的顺序确保搜索最小化预先分配任务执行时间的顺序. 机械手任务点的最优排序问题可以看作是著名的旅行商问题 (TSP) 的扩展. 而进一步考虑到城市之间的距离, 优化的目标是找到应该遵循的最优总行程. 结合实际情况将 TSP 应用到机器人上, 最小化问题的重点是时间的优化而不是对距离的优化 [6]. 而国内外已有大量的学者对此开展过研究. Dubowsky 和 Blubaugh 使用 Little 算法来确定了机械手任务的最短时间, 并通过三个案例解决了 TSP 的任务点可以以任何顺序访问, 并测试了算法的可行性. Abdel-Malek 和 Li 开发了一种寻找机器人任务性能最佳排序的方法. 其在考虑机器人构型多样性的情况下确定了最小循环时间. 该算法在机器人

收稿日期: 2019-11-28
资助项目: 江西省教育厅科技项目 (GJJ170820)

上进行了测试, 但是, 该算法没有扩展到具有两个以上配置的机器人.

上述这些研究为我们提供了将遗传算法应用于四种不同构型的类 PUMA 机器人并得到三维空间中七个点和十四个点的遗传算法的可能性. 以此为基础, 本文建立了一种基于 GAs 的非冗余机械手末端执行器优化序列的确定方法. 在该方法中, 每个染色体形成的人口以一种特定方式代表序列的机械手到达 N 任务分及机械手逆运动学的解决方案的配置导出在每个任务点. 该算法考虑了逆运动学问题的多重解, 可以应用于任意非冗余度机械手, 而不需要考虑其复杂性, 具有较好的通用性.

2 目标函数规划

2.1 问题阐述

表示末端执行器的位置和方向的向量 s 与关节位移向量 q 之间的关系通常是非线性的[7−10]. 该机械手的运动学关系式为:

$$s = f(q) \tag{1}$$

当用向量 q 表示关节位移时, 其对应的函数 s 具有唯一性且计算较为简单. 当给定机械手一个任务时, 可获得其末端执行器的位置和方向 s 或轨迹 $s(t)$, 则式 (1) 的解可以形式化地表示为:

$$q = f^{-1}(s) \tag{2}$$

然而, 解 q 不一定存在, 即使存在, 它通常也不是唯一的. 求给定 s 对应的 q 的问题则称为逆运动学问题.

为此, 本文讨论了这样一个问题: 让一个机械臂访问 N 个点 (每个点恰好一次), 然后返回初始点. 要求确定机械手的行程, 使总循环时间最小且操作空间可以达到有限数量的配置. 这意味着在多个配置的情况下, 最优序列肯定会受到配置选择的影响. 因此, 给定每一种构型的关节位移, 计算任意两点间移动所需的时间是较容易的. 完成完整的运动规划所需的总时间构成了要优化的目标函数.

2.2 目标函数

假设在 m 维空间 $(n < m)$ 中的 N 自由度机械手需要访问 N 个作用点, 且作用点的位置已知, r 为逆运动学问题解的个数. 图 1 所示为 N 自由度机械手, 该机械手需要访问 m 维空间中的 N 个任务点. 一组 r 配置 (表示逆运动学问题的 r 个解) 对应于 N 个任务点中的每一个作用点. 目标是找到访问所有任务点的路径 (每个任务点一次), 从而在考虑每个任务点对应的机器人配置的多样性的条件下确定最小的旅行时间.

上述问题首先应计算关节坐标 $(q \in R^n)$, 求解 m 维空间各点的逆运动学问题. 因此, 由机械手时间 t_i 从任务点 $(i-1)$ 使用 q^l 组关节坐标计算 c-configuration 任务点. 使用 q^k 组关节坐标计算 k-configuration 任务点, 因此需要进行优化的目标函数为:

$$t_i = \max\left(\frac{|q_{ji}^k - q_{j(i-1)}^l|}{\dot{q}_j}\right), \quad \begin{matrix} j = 1, 2, \cdots, n \\ k, l = 1, 2, \cdots, r \end{matrix} \tag{3}$$

其中 q_{ji} 为第 i 个末端执行器位置的第 j 个关节位移, $\dot{q}_j$ 为关节 j 的平均速度. 式 (3) 表示两个位置之间的运动时间由最慢的机械手的关节决定. 由此也可反映出, 对于多解的情况, 机械

手构型的选择对行程时间有显著的影响. 因此, 访问所有任务点所需的总旅行时间 t_c 为:

$$t_c = \sum_{i=2}^{N} t_i \tag{4}$$

将该公式进一步优化, 可以写成

$$t_{opt} = \min \sum_{i=2}^{N} t_i \tag{5}$$

该函数的优化变量为每个任务点对应的机器人配置. 函数的最优值可表示为 N 个点之间的最小旅行时间. 因为机器人拥有的所有任务点的数量是有限的, 从而使得这个函数的搜索空间是离散的.

在上述方法中, 问题的复杂性取决于任务点的数量和可能的机械手配置的数量. Tsai 和 Morgan 证明了六转动关节机械手的逆运动学问题最多有 $2^8(=256)$ 个解. 在最后三个轴相交的特殊情况下, 可能的构型为 $2^4(=16)$. 因此, 在这种情况下, 解的最大数目表示为 2^d, 其中 d = 1,2,3,4. 所有可能的浏览次数为 $(N-1)!/2$. 所有可能的配置的数量与机械手可以达到 N 个点是 $(2^d)^N$. 对于每次浏览在二维逆运动学范围内都存在解决方案. 因此, 搜索空间的距离 R_s 为:

$$R_s(N,d) = \frac{(N-1)!}{2}(2^d)^N \tag{6}$$

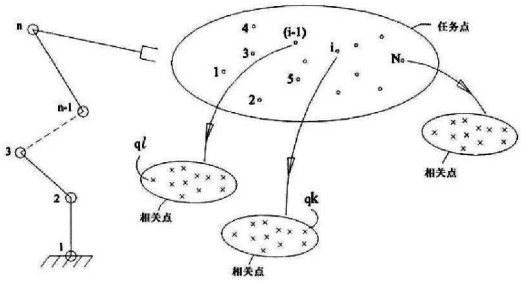

图 1 一个 N 自由度机械手的示意图

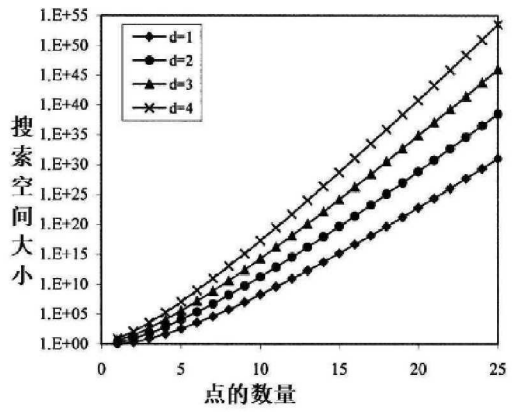

图 2 搜索空间的总体数量与配置点的数量

一个二自由度的机械手在二维空间中运行, 需要访问十个点, 每个点对应两个不同的配置, 搜索空间的总体距离 $R_s\,(10,1) = 1.85 \times 10^8$. 在六自由度机械手操作的情况下的三维空间访问十个点八组不同的配置搜索空间距离 $R_s\,(10,3) = 1.94 \times 10^{14}$. 由此可见可能的解决方案的数量在很大程度上取决于机械手的自由度及其可能的配置情况. 图 2 表示搜索空间的总体和任务点的数量, 对于机械手必须访问空间中具有 2、4、8 和 16 种可能配置的点的情况.

3 基于 GA 算法的优化方法

3.1 遗传算法

1975 年美国 Holland 教授受到达尔文进化论思想的启发, 根据适者生存, 优胜劣汰的自然法则, 发明了遗传算法 [11]. 该算法主要规则是模仿自然界优胜劣汰、适者生存的规则 —

越能适应环境的物种, 生存下去的几率就越高. 遗传算法将生物进化过程中的生育、自然选择、杂交、基因突变以及竞争等概念引入到算法中. 如果将问题的解看作是一个一个的个体, 并用一定的适应度函数来评价这些个体适应环境的能力, 这些个体通过互相交换自己的基因或者由于某些外部因素导致个体的基因产生突变, 就会得到新的个体. 个体为了适应环境的变化, 自己的基因也会跟随着变化, 通过一代又一代的交配与突变, 下一代个体相比较于上一代个体, 适应度会得到提高, 得到了能更加适应环境的个体, 即可以得到更加满足要求的解. 物种的表现是由他的基因决定的, 且物种通过交配或变异, 生成新的物种. 遗传算法模拟生物界物种进化的机理, 在生成的初始种群的基础上, 对种群进行选择、交叉、变异等操作, 改变种群里的基因, 通过适应度函数来评价个体的好坏. 通过每一次的选择、交叉、变异, 生成新的、适应度更高的种群, 以此来达到找寻最优解的目的. 目前, 遗传算法广泛应用于函数优化、组合优化、车间调度、图像处理等问题中.

遗传算法减少了计算时间, 甚至对大量的点也具有收敛性. t_c 的总时间传递两个点之间的路径距离成正比. 因而, 成本函数被定义为沿着测地线在惯性空间中的距离, 由于计算测地线距离在这个空间就相当于计算通过该路径的时间. 该算法在确定机器人通过任务点的路径时, 考虑了机器人的运动学和动力学特性. 但是, 在寻找最优路径的过程中, 没有提到是否考虑了可能的配置. 得到了一个时间复杂度仅为 $O(N^2)$ 的快速算法, 其中 N 为任务点的个数, 但该方法并不总能得到最短的循环. 这意味着 NN 算法不能保证找到最优解.

作为一种源自统计力学的技术, 模拟退火 (SA) 目标是类比物理退火过程的行为. Dissanayake 和 Gal 使用 SA 确定冗余度机械手的最优行程序列 [12−16]. 针对优化问题的非线性, 采用序贯二次规划方法求解给定任务序列下机器人的最优配置和工作单元的最优位置. SA 应用于三自由度平面机器人 (3R、RPR 和 PRR), 需使用工具在二维空间中通过 9 个位置, 将工具返回到原始位置. 该算法对 9 个位置问题进行了约 18000 个序列的搜索, 直到收敛. 然而, 据估计, 对于 19 个工作场所问题, 该算法应该搜索大约 45,000 个序列. 因此, 为了在合理的时间内得到一个解决方案, 需要一台计算能力强大的工作站.

Petiot 等人使用弹性网络方法 (ENM) 来最小化机器人任务的周期时间. 该方法较好地解决了机械臂末端执行器逐点运动的最优排序问题. 这是通过使用改进的梯度法最小化一个能量函数 E 来实现的. 然而, 由于算法参数选择错误, 算法可能不收敛.

前面的算法对二自由度或三自由度的机器人均能很好地工作, 但存在一定的困难, 因为时间成本, 或不能推广到自由度较大的机器人上. 其中一些算法考虑了逆运动学问题的多重解. 在讨论的所有例子中, 二维空间中的每个点都可以用两种构型来表示. 此外, 这些算法不需要太多的时间来找到最优的解决方案的点数不超过 10 个. 若超过 10 点, 时间成本会显著增加.

以此为基础, 众多学者提出了解决 TSP 的几种方法, 并在某些情况下进行了修改, 以确定机械手的最小周期时间. 尽管一些研究人员已经使用 GA 来处理 TSP, 但还没有考虑到机器人及其多种配置. 与其它优化方法相比, GA 优化方法的主要优点是不需要目标函数的导数. 因此, 能够找到非连续函数和过程函数的最优解. 遗传算法的基因型是一组整数, 表明了一个潜在的解决方案. 该方法计算了由机器人末端执行器到达 N 个点并最终选择出最小行走时间的逆运动学问题所产生的所有可能构型的组合. 在这些方法中, 考虑了将关节从机器人到达任务点的配置移动到离开任务点的配置所需的时间.

3.2 代表机制

GAs 是基于自然选择和适者生存的原理和机制的自适应搜索技术. 遗传算法与一定数量的染色体并行工作, 每个染色体由一组有限的符号 (基因) 组成, 代表给定目标函数的可能解. 在每一次迭代中发生变化的一组染色体称为群体, 群体中的成员称为个体. 与其它优化算法一样, 遗传算法首先确定适应度函数和控制参数. 它像其他任何优化算法一样, 停止检查收敛性. GA 的表示机制是该方法的主要创新之处, 本文对其进行了详细的描述. 应用遗传算法的第一步是选择合适的编码表示当前优化问题的可能解. 解决方案表示为特定字母表上的字符串. 解决方案的位串表示主导遗传算法的编码, 因为二进制字母表提供了任何编码的每字节信息的最大模式数. Holland 给出了使用二进制编码的理论依据. 他将二进制编码中可用的模式数量与非二进制编码中可用的模式数量进行了比较, 其中两种编码具有相同的信息承载能力. Holland 的计数模式显示了 GAs 在多字符编码上的性能要比在二进制编码上差. 此外, 一些测试表明非二进制编码比二进制编码执行得更好.

对于一个二自由度的机械手, 它必须访问二维空间中的 N 个点, 每个点对应两种不同的构型, 每个染色体由 $2N$ 个基因组成. 由 N 个符号组成的字符串的第一部分, 表示机械手到达 N 个任务点的顺序. 对于这一部分, 使用的是整数字母表. 第二部分由 N 个符号组成的字符串, 表示机械手的构型. 第二部分使用二进制字母. 因此, 给定连杆的长度 L_1 和 L_2 和 N 个作图点的位置坐标 x y, 则关节变量 q_1 和 q_2 的逆运动学问题的解为:

$$\begin{cases} q_1 = \tan^{-1}(y/x) \pm \cos^{-1}\left(\frac{x^2+y^2+L_1^2-L_2^2}{2L_1\sqrt{x^2+y^2}}\right) \\ q_2 = \cos^{-1}\left(\frac{x^2+y^2-L_1^2-L_2^2}{2L_1L_2}\right) \end{cases} \tag{7}$$

$\pm$ 符号表示联合变量 q_1 有两种不同的解决方案, 从而导致两种不同的配置. 因此, 第一配置 (图 3 中的实线) 具有解 (q_{1-}, q_2), 而第二配置 (图 3 中的虚线) 具有解 (q_{1+}, q_2). 因而与二维空间中的任务点相对应的二进制数字值确定了操纵器的两种配置中的一种. 特别是配置 (q_{1-}, q_2) 对应于二进制数字 0, 而配置 (q_{1+}, q_2) 对应于二进制数字 1.

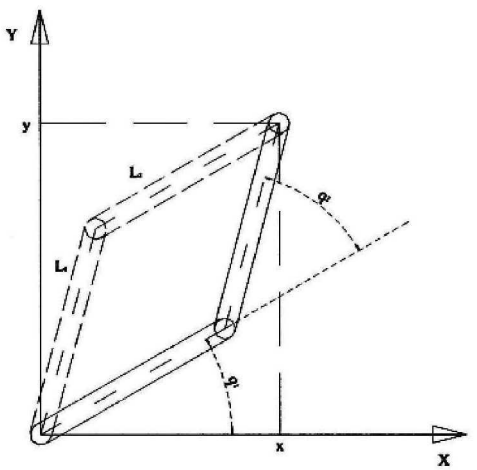

图 3 在二维空间中运行的 2-DOF 机械手的两种不同配置

让一个二自由度操纵器必须访问二维中的 9 个点空间, 然后染色体可以形成为: 如图 4(a) 所示.

这意味着操纵器与二进制数字相对应的配置到达任务点 5, 然后到达与二进制数字 1 相对应的配置的点 9, 依此类推. 让一个六自由度操纵器必须访问 N 个任务点具有与每个任务点相对应的八个不同配置的三维空间. 字符串的第一部分由 N 个符号组成, 表示操纵器到达 N 个点的顺序. 对于此种情况, 应使用整数字母. 字符串的第二部分由 3 个字节至 N 个字节组成, 其表示操纵器的配置. 此外, 与三维空间中的任务点相对应的 (00, 001, 010, .., 111) 的每个字节都以类似的方式确定了机械手的八种配置中的一种对于二自由度机械手的情况. 让一个必须在三维空间中访问 9 个点的操纵器, 可以将染色体形成为: 如图 4(b) 所示.

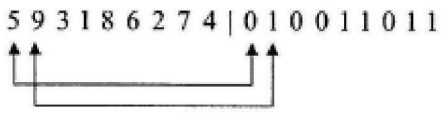

图 4(a) 染色体形成

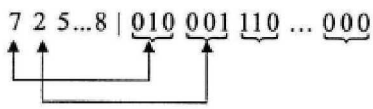

图 4(b) 染色体形成

这意味着操纵器以对应于字节 (010) 的配置到达点 7, 然后到达对应于字节 (001) 的配置到达点 2, 依此类推.

通常, 对于必须访问 m 维空间中的 N 个点并以 2d 种不同配置到达每个任务点的操纵器, 染色体形成如如图 4(c) 所示.

对于第二部分, 使用整数进行编码较为适用. 对于这种情况, 染色体形成如如图 4(d) 所示. 其中箭头显示一种配置与相应任务点的对应关系.

其中 p_u 是介于 1 到 2d 之间的整数, 并且对应于特定的配置. 在这一点上, 应该区分用于表示 TSP 的整数编码和用于机器人配置的整数编码. 与机器人必须 "访问" 的任务点相对应的整数不能重复, 因为机器人必须精确地到达每个任务点一次. 另一方面, 机器人可以采用任何配置到达任务点, 因此可以重复与机器人配置相对应的整数. 这意味着应用于染色体的两个部分的遗传算子 (交叉和突变) 不能相同.

图 4(c) 染色体形成

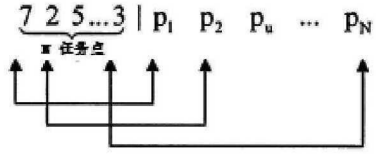

图 4(d) 染色体形成

尽管二进制编码似乎比整数编码更复杂, 但它的性能要好得多. 此外, 二进制编码的代码更加简单, 因为不必将二进制数与机器人的配置相对应 (q1, q2, y), 如上文所述.

4 GA 算法的优化流程

本节介绍遗传算法的建模和应用. 遗传算法是一种源于进化论的自适应系统工具, 遗传算法的基本原理最早由霍兰德提出. 从本质上说, 产生了许多解决办法, 构成了一个群体, 在这个群体中经过几代人的发展, 产生了更好的解决办法. 遗传算法要求首先对问题进行编码以适应算法. 编码后, 将遗传算法算子应用于染色体. 然而, 由于交叉和突变操作的活性, 不能保证获得的新后代将是良好的解. 交叉、突变和繁殖的过程不断进行, 直到找到最优的解决方案. 在本研究中, 案例解表明这些解在机器人的末端执行器和目标之间产生较低的定位误差. 该误差作为适应度函数用于遗传算法设计. 本文基于遗传算法的轨迹优化基本原理流程图如图 5 所示.

4.1 初始种群

通常, 初始种群是随机选择的, 以便均匀分布所选染色体 (搜索空间上的溶液). 在某些情况下, 将从其他优化算法获得的解决方案用于播种初始种群. 尽管上述做法存在将优化过程误导为局部最优的风险, 但事实证明, 播种方法在某些情况下非常有效. 在本文中, 使用案例

检索作为初始种群播种条件以加快 GA 找到解决方案的速度.

4.2 评价机制

评估机制为适应度函数, 可对总体的所有染色体进行评估. 一个染色体的适应度函数可反映该染色体对环境的适应程度, 即该染色体在下一代中的生存和繁殖能力. 在本文, 目标函数处理操纵器访问 N 个任务点并返回到起点所需的时间, 而目标是使时间最小化. 获取目标函数的过程将在下文中详细描述. 适应度函数是目标函数的逆函数:

$$fitness = \frac{1}{t_c} \tag{8}$$

式 (4) 给出了 t_c. 由于 $t_c \neq 0$; 不必采取任何预防措施来避免无穷大的值.

4.3 繁殖

繁殖的目的是保留人口中好人所具有的优良特质, 并将其以较高的比例传播到下一代人口中. 因此, 繁殖不会产生新的个体.

在遗传算法中, 根据适应度函数的标准化值 (而非绝对值), 将染色体从上一代复制到下一代. 其比例选择基于轮盘赌策略, 即将按照与它们的适合度成正比的比率进行选择. 这意味着选择具有高适应性的染色体进行繁殖的概率要高于具有较低适应性的染色体.

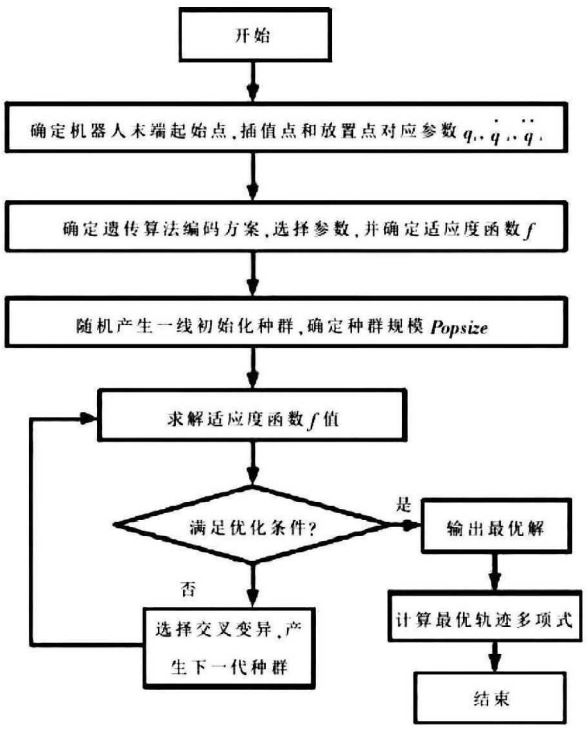

图 5 GA 算法的优化流程

4.4 控制参数

交叉是重组运算符, 并遵循复制操作. 该操作的作用是将几个人的部分联系在一起, 以便为下一代生产新的个体. 根据预定义的概率 (交叉率) 随机选择个体. 用于染色体第一部分 (由十进制数字组成) 的交叉算子为顺序交叉 (OX), 对于染色体的第二部分, 则使用单点交叉.

应用突变以较小的预定义概率 (突变率) 对一些个体执行随机修改. 用于染色体第一部

分的由十进制数字组成的变异算子是倒置的. 倒置仅适用于染色体, 并保证所产生的后代代表 “合法” 游览. 对于染色体的第二部分, 应用变异算子将数字值 “0” 的随机基因更改为 “1”, 反之亦然, 以进行二进制编码; 而对于整数编码, 则将随机基因更改为介于 1 和 2 之间的另一个值 2D.

控制参数的数量影响遗传算法的过程, 因此收敛速度和最终的结果必须被定义. 最重要的控制参数如下: 1) 种群大小, 种群大小决定了染色体的数量, 因此在遗传搜索过程中有多少遗传物质可用. 2) 交叉率, 交叉率决定了交叉算子应用于种群染色体的频率, 从而产生新的种群. 交叉率越高, 新种群中引入的个体越多. 交叉率通常在 0.6 到 1.0 之间, 在本文的案例中, 经过大量的试验, 交叉率被选择为 0.8. 3) 突变率, 突变率决定了一个基因在染色体上的值发生改变的概率. 突变引入了未探索的搜索空间的新领域. 然而, 突变率不应太高, 因为其可能增加搜索的随机性. 突变率通常小于 0.4, 在本文的案例中, 经过多次试运转后, 突变率被选择为 0.1.

4.5 收敛

在许多情况下, 通过预定义最大迭代次数 (生成次数) 完成收敛. 然而, 最大迭代数的预先确定意味着遗传搜索的持续时间是固定的, 而不考虑搜索成功与否. 此外, 很难事先确定需要多少迭代才能找到接近最优的解决方案. 因此, 应该对遗传算法的质量水平进行在线评估.

在本文的方法中, 终止演化所需满足的条件是对预定义的迭代数进行相同解决方案的迭代. 因此, 该算法通过提前定义相同染色体不断出现的最优染色体的迭代次数来终止.

本文提出的遗传算法来求解基于末端执行器误差最小化的机器人逆运动学问题. 由于机器人的结构特点, 使得决策空间较大, 同时也增加种群规模的大小, 种群规模过大虽然可以获得高精度的逆运动学解, 但是在保证全局最优的情况下由于计算复杂度增加难以满足.

5 实验

本节通过四组仿真实验对机器人任务调度优化方法进行了测试. 具体通过对染色体进行了两种编码, 即二进制编码和整数编码进行了测试, 以便比较每种情况下获得的最佳循环时间, 研究了控制参数对三自由度和六自由度机械手的遗传算法结果的影响, 并将结果与其他方法进行了比较. 在以下的所有实验中, 由于不能证明所找到的解是最优解, 所以用 “接近最优” 来表示所找到的最佳解.

5.1 提出的遗传算法的编码

对于这两种编码, 染色体的第一部分都使用整数字母. 对于染色体的第二部分, 第一种编码采用二进制字母表, 而对于第二种编码则使用整数字母表. 利用这两种编码, 对同一点、同一控制参数的三自由度和六自由度机械手进行了实验. 表 1 和表 2 解释了播种百分比、近似最优解用时和近似最优解播种数量之间的关系. 显示了接近最优的解决方案以及几种不同比例的方案首次出现的时间. 表 1 和表 2 所示的结果对应于三自由度和六自由度机器人在三维空间中必须达到十个任务点的情况. 从结果中可以看出, 种子数量对算法求得的最优解有一定的影响, 且各情况下的影响是不同的. 因此, 对于三自由度机器人, 最佳 “近似最优” 的解决方案是播种百分比为 70%, 而对于六自由度机器人, 最佳 “近似最优” 的解决方案是播种百分比为 45%.

表 3 和表 4 的结果显示在几乎所有比较循环时间和遗传算法收敛的情况下, 代表机械手配置的二进制编码都优于整数编码. 因此, 在实施的所有测试中, 染色体的第二部分都使用了二进制字母表.

5.2 控制参数的影响

将所提出的遗传算法应用于需要在三维空间中访问 10 个点的三自由度机械手. 对控制参数的各种组合进行测试, 并将同一染色体的最大数设置为 200. 给出 “最佳” 近似最优解的控制参数组合为: 种群规模 =450, 交叉率 =0.9, 突变率 =0.25, 最小循环时间为 2.92 s. 这种最优解在迭代 35 次中首次出现. 第一行控制参数的最佳循环时间为 3.32 s, 即: $\frac{|3.32-2.92|}{2.92}100\% = 13.69\%$.

图 6 为遗传算法收敛过程中总循环时间的最佳值、平均值和最差值, 与上述控制参数值一致.

表 1 三自由度机器人的接近最优解与播种百分比

播种百分比	近似最优解用时(S)	近似最优解播种数量
5	3.14	72
10	3.24	2
15	3.09	196
20	3.01	96
25	2.92	358
30	2.92	38
35	3.06	31
40	3.27	11
45	3.01	155
50	3.20	44
60	3.14	160
70	2.92	35
80	3.08	184
90	3.25	52
100	3.24	12

表 2 六自由度机器人接近最优解与播种率的关系

播种百分比	近似最优解用时(S)	近似最优解播种数量
5	5.07	166
10	6.26	2
15	5.87	7
20	6.26	2
25	6.26	2
30	6.26	2
35	6.10	3
40	6.26	2
45	4.82	3
50	4.66	125
60	4.66	241
70	6.26	2
80	6.05	219
90	6.10	3
100	6.10	3

表 3 循环次数采用二进制和整数编码的三自由度机械手

点数	二进制编码		整数编码	
	循环时间 (S)	传播次数	循环时间 (S)	传播次数
5	2.31	3	2.35	18
10	2.92	35	4.86	48
15	3.10	1	8.65	212
20	3.57	14	9.57	140
25	4.02	87	10.62	123

表 4 循环次数采用二进制和整数编码的六自由度机械手

点数	二进制编码		整数编码	
	循环时间 (S)	传播次数	循环时间 (S)	传播次数
5	2.31	3	2.35	18
10	2.92	35	4.86	48
15	3.10	1	8.65	212
20	3.57	14	9.57	140
25	4.02	87	10.62	123

在仿真实验中, 对一个需要在三维空间中访问 10 个点的六自由度机械手进行了测试. 测试了各种控制参数, 而算法在同一条染色体的 200 次迭代后终止. 所选结果如表 5 所示.

表 5 循环次数采用二进制和整数编码的三自由度机械手

种群规模	交叉率	变异率	接近最优解时间 (S)	接近最优解传播次数
150	0.8	0.10	3.32	4
150	0.9	0.25	3.32	4
250	0.8	0.10	3.01	3
250	0.9	0.25	3.16	59
350	0.8	0.10	3.13	13
350	0.9	0.25	3.29	111
450	0.8	0.10	3.01	53
450	0.9	0.25	2.92	35

最佳参数组合的依据是使用遗传算法过程中, 算法必须是收敛的. 同时选取总循环时间的最佳值、平均值和最差值都应该控制在合理的范围内同时达到一定的全局最优. 本研究的目的在于找出一个最小化适应度函数的逆运动学解. 笛卡尔坐标信息可以用直接运动学方程计算得到的任何新的子代. 其次, 利用三维空间中两点之间的三维 (3-D) 距离方程, 可以方便地得到位置误差的度量形式, 具体如下: $Error = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$, 其中 (x, y, z) 为空间坐标.

5.3 计算时间的结果

如图 7 所示, 二自由度机械手 (一个移动关节和一个转动关节), 它必须 "访问" 二维空间中的 10 个点. 该机械手通过求解逆运动学问题, 可以得到具有两种不同构型的空间点. CPU 时间为 20 秒, 算法在 VAX 4500 计算机上运行, 而使用 Little 方法的 CPU 时间为 1280 秒.

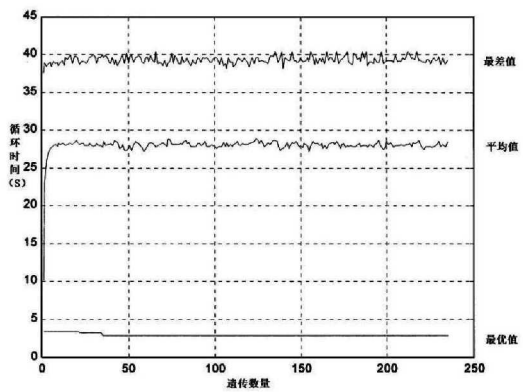

图 6 自由度机械手的情况, 提出了遗传算法的收敛性

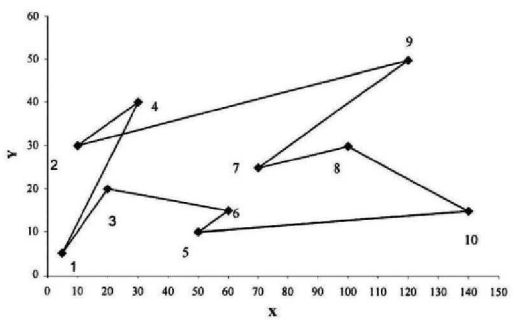

图 7 二自由度机械手在 x-y 平面上 10 个点的最优路径

利用该方法, 末端执行器路径在二维空间中访问 10 个点的最优序列如图 9 所示. CPU 时间为 3.91 s, 算法在 Matlab 平台下运行在 1.8 GHz 的 Pentium 4 PC 上. 计算时间性能显示计算时间随任务点数量的变化. 图 8 和图 9 显示了这种变化, 它与计算时间结果近似线性. 与点的数量相比, CPU 时间的增长速度并不快. 此时需要说明的是, 最优机器人调度问题要比经典的 TSP 问题复杂得多, 因为同时考虑了机器人的多种构型. 尽管搜索空间随着任务点

的增加呈指数级增长, 但 CPU 时间几乎是线性增长的. 该算法的 CPU 时间随点的数量线性增加, 计算时间随着任务点的数量显著增加 (几乎呈指数增长).

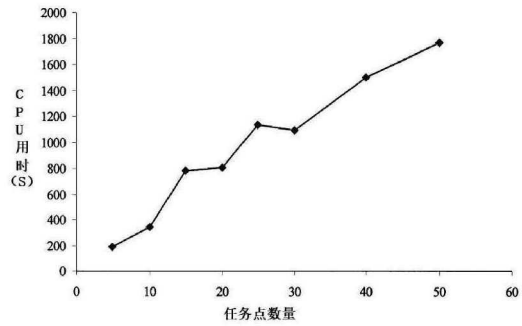

图 8 三自由度机械臂的 CPU 时间与三维空间中任务点的数量之比

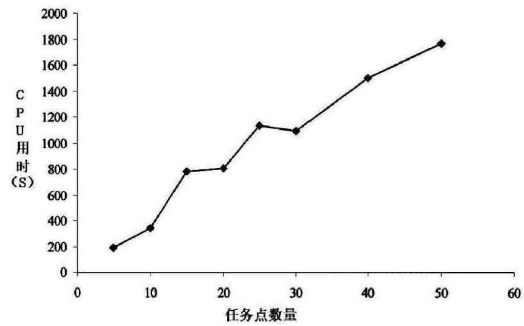

图 9 CPU 时间与六自由度机械手在三维空间中的任务点数量的关系

6 结论

1) 本文提出了一种基于 GA 的机械手末端执行器优化排序方法. 该方法以 GA 为基础, 主要在遗传算法的编码上进行了创新, 在计算总循环时间时考虑了逆运动学问题的多个解. 在本文方法中, 染色体由两部分组成. 字符串的第一部分表示机械手到达 N 个任务点的顺序, 第二部分表示机械手对应于每个任务点的配置. 第一部分使用整数字母, 而第二部分使用二进制字母. 该模型考虑了逆运动学问题的多重解, 具有通用性, 可方便地应用于任意非冗余度机械手.

2) 对于遗传算法的结构, 控制参数的选择对优化问题的求解有很大的影响. 尝试了几种控制参数的组合, 最终选择出 “最佳” 近似最优解的组合.

3) 仿真实验显示, 本文提出的方法能在一定的时间内快速找到最优解或近似最优解. 且任何非冗余度机械手的多重配置很容易体现在遗传算法的编码中.

4) 本文提出的方法可以在合理的 CPU 时间内求出六自由度机械手在三维空间内的最优操作序列. 考虑到未来的研究工作, 可以对该算法进行扩展, 使其能够考虑避障问题.

参考文献

[1] 杨浩, 邹鲲, 袁培峰, 等. 基于遗传算法的化纤落卷任务优化 [J]. 东华大学学报 (自然科学版), 2019, 45(1): 90-95.

[2] 黄宛宁, 龚建伟, 王鹏辉. 基于改进遗传算法的多机器人任务分配方法 [J]. 计算机仿真, 2006, 23(11): 164-167.

[3] 王坤. 基于遗传算法的移动机器人路径规划研究 [D]. 哈尔滨工业大学, 2012.

[4] 浦定超. 基于遗传算法的移动机器人路径规划的研究 [D]. 合肥工业大学, 2010.

[5] 姜明洋. 基于遗传算法的移动机器人路径规划方法的研究 [D]. 沈阳理工大学.

[6] 田欣. 基于改进遗传算法的移动机器人路径规划研究 [D]. 2016.

[7] 李霞, 张基宏, 谢维信. 基于遗传算法的任务分配问题求解 [J]. 数据采集与处理, 1999, 14(3): 293-297.

[8] 任金霞, 黄艺培, 钟小康. 基于遗传算法的云任务调度改进算法 [J]. 江西理工大学学报, 2018, 39(3): 90-94.

[9] 琚忠明, 孟令, 方小马, 等. 基于遗传算法的任务分配与调度 [J]. 自动化应用, 2018(6): 69-71.
[10] 马俊涛, 陈业恩, 胡国杰, 等. 云计算环境下基于遗传算法的任务调度技术研究 [J]. 软件导刊, 2016, 15(1): 51-53.
[11] 孙俊, 须文波. 一种基于遗传算法的分布式系统的任务调度 [J]. 计算机工程与应用, 2003, 39(21): 105-106.
[12] 孙志峻, 潘全科, 朱剑英. 基于遗传算法的多资源作业车间智能优化调度 [J]. 中国机械工程, 2002, 13(24): 2104-2107.
[13] 段勇, 王宇, 喻祥尤. 基于免疫遗传算法的多机器人环境探索 [J]. 沈阳工业大学学报, 2018, 40(3): 62-66.
[14] 杜宗宗. 基于遗传算法的移动机器人路径规划研究 [D]. 南京师范大学, 2009.
[15] 周友行, 何清华, 谢习华. 基于遗传算法的凿岩机器人孔序规划 [J]. 机器人, 2002, 24(1): 62-65.
[16] 杨煜俊, 龙传泽, 陶宇. 作业车间类型多机器人制造单元调度算法 [J]. 计算机集成制造系统, 2015, 21(12): 3239-3248.

Study on Optimal Robot Task Scheduling Based on Genetic Algorithms

ZHAI Fan[1], XIE Xian-hua[2]

(1. Department of Basic, Zhengzhou Shengda University Of Economics,Business & Management, Zhengzhou 451191, China)
(2. School of Mathematics and Statistics, Gannan Normal University, Ganzhou 341000, China)

Abstract: Industrial robots should complete complex tasks in as short a period as possible to achieve high productivity. The problem of determining the optimal path for the manipulator end-effector to access multiple task points is like, but not identical to, the famous traveling salesman problem (TSP). In order to adapt the TSP to robotics, the metric that needs to be optimized is time, not distance. In addition, the travel time between any two points is greatly affected by the choice of manipulator configuration. Therefore, we need to consider the multiple solutions of the inverse kinematics problem. The solution of inverse kinematics problem is the foundation of robot control. Many traditional solutions to inverse kinematics problems, such as geometry, iteration, and algebra, are not enough for redundant robots. Based on the genetic algorithm, a new coding method is proposed to consider the multiple solutions of inverse kinematics problem. Simulation results show that the proposed method can find the optimal solution or approximate optimal solution in a certain time. And the multiple configuration of any non-redundant manipulator is easily reflected in the coding of genetic algorithm.

Keywords: genetic algorithm; simulation platform; task optimization; machine learning