文章编号: 1002-0446(2002)02-0130-04

基于神经网络的机器人操作手 IKP 精确求解*

陈学生　陈在礼　谢　涛

(哈尔滨工业大学机械电子工程教研室　150001)

摘　要: 结合位置正解模型, 利用 BP 网络求解了机器人逆运动学问题(IKP). 为提高求解结果精度, 采用迭代计算进行误差补偿, 计算结果表明, 该法迭代次数少, 计算精度高且计算速度接近机器人实时控制的要求.

关键词: 机器人操作手; 逆运动学问题(IKP); 神经网络; 误差补偿

中图分类号: TP24　　文献标识码: B

AN ACCURATE SOLUTION TO THE INVERSE KINEMATIC PROBLEM OF A ROBOT MANIPULATOR BASED ON THE NEURAL NETWORK

CHEN Xue-sheng　CHEN Zai-li　XIE Tao

(*Mechanism Electronic Engineering Specialty, Harbin Institute of Technology*　150001)

Abstract: In this paper, IKP of the robot manipulator is solved by using BP neural network and the forward kinematic model. To improve the accuracy of the solution, an iterative approach is used to compensate for the offset error. Numerical results have shown that the accurate solutions can be obtained by performing only a few iteration steps and the computation speed can meet the requirements for the robot's real time control system.

Keywords: robot manipulator, inverse kinematics, neural network, error compensation

1　引言(Introduction)

逆运动学是给定机器人末端操作器的位姿, 计算对应关节变量, 与正运动学不同, 逆运动学问题要复杂得多, 表现为逆解的存在性和唯一性问题. 逆解的存在性定义了手的操作空间. 进一步, 逆解通常是非唯一的, 且不一定存在闭式逆解. 业已证明, 具有球形手腕的操作器存在闭式逆运动学解. 对于一般结构的 n 自由度操作器, 逆解要通过求解一个 $2n$ 阶多项式方程才能获得. 当 $n>2$ 时无解析解. 在此情况下, 我们不得不使用某种数值算法, 逆运动学问题成为一个非线性超越方程的数值求解问题. 困难之一是数值解法一般不能给出全部可能的解; 困难之二是现在的机器人控制系统通常要求实时计算逆运动学, 因而必须具有快速性质, 而常规的数值解法的迭代性质使得解的精度不高或花时太多.

人工神经网络在解决非线性映射方面的问题时有强大的逼近能力, 在机器人领域可用于操作手运动学及动力学模型的自动辨识、障碍回避与路径优化以及机器人控制等一系列问题[1~5]. 结合位置正解模型, 采用 BP 网络, 通过对大量样本的训练学习, 实现机器人从工作变量空间到关节变量空间的非线性映射, 从而可以避免求解位置逆解时公式推导和编程计算等繁杂的过程. 但常规的 BP 网络的求解精度难以满足机器人精确控制的要求. 为提高逆解结果精度, 可以采用迭代计算进行误差补偿. 本文采用 BP 网络与误差补偿结合的方法, 以 PUMA560 机器人为例, 得到了比较精确的位置逆解求解结果.

2　BP 神经网络模型 (Model of BP ANN)

BP 神经网络的结构参数参见文献[6]. *BP* 网络的学习过程由前向过程和误差反向传播过程组成. 其前向计算过程如下.

(a) 输入层节点 i 的输出 O_i 等于其输入 L_i.

(b) 隐层节点 j 的输入、输出分别为

$$net_j = \sum_i \omega_{ji} o_i + \theta_j$$

$$o_j = f(net_j) = [1 + \exp(- net_j)]^{-1}$$

* 收稿日期: 2001-03-26

式中, ω_{ji}为隐层节点 j 与上一层节点 i 之间的连接权重; θ_j 为隐层节点 j 的阈值; f 为非线性 Sigmoid 函数.

(c) 输出层节点 k 的输入、输出分别为

$$net_k = \sum_j \omega_{kj} o_j + \theta_k$$

$$o_k = g(net_k) = net_k$$

式中, ω_{kj}为输出层节 k 点与上一层节点 j 之间的连接权重; θ_k 为输出层节点 k 的阈值. g 为线性 purelin 函数.

对于给定的训练样本集$(x_{p1}, x_{p2}, \cdots, x_{pN}) \rightarrow (t_{p1}, t_{p2}, \cdots, t_{pK})$, $p = 1, 2, \cdots, P$ 为样本序号, 网络运算结果与训练样本目标输出之间的均方误差和, 即网络的系统误差表示为

$$E = (1/P)\sum_{p=1}^{P} E_p$$

$$E_p = (1/\pi)\sum_{k=1}^{K} (t_{pk} - o_{pk})^2$$

式中, P 为训练样本数; t_{pk}和 o_{pk}分别为 p 第样本的第 k 输出单元的目标输出和网络运算结果.

网络训练学习的过程就是通过调节网络内部连接权重使网络误差最小化的过程. BP 网络的连接权重调整过程也就是误差反向传播过程.

对于输出层与隐层之间的权值 ω_{kj}有

$$\omega_{kj}(n+1) = \omega_{kj}(n) + \eta\delta_k o_j + \alpha[\omega_{kj}(n) - \omega_{kj}(n-1)]$$

$$\delta_k = g(net_k)(t_k - o_k) = (t_k - o_k)$$

式中 t_k 为节点 k 的目标输出.

对于隐层与输入层之间的权值 ω_{ji}有

$$\omega_{ji}(n+1) = \omega_{ji}(n) + \eta\delta_j o_i + \alpha[\omega_{ji}(n) - \omega_{ji}(n-1)]$$

$$\delta_j = f'(net_j)\sum_k (\omega_{kj}\delta_k) = o_j(1 - o_j)\sum_k (\omega_{kj}\delta_k)$$

式中, n 为迭代运算次数; η 为权值的增益系数, 即学习率, $0 < \eta < 1$; α 为惯性系数, 用以调整学习的收敛速度.

BP 网络是目前比较成熟且广泛应用的一种网络. 它把一组训练样本的输入输出问题变为一个非线性优化问题. 从映射的角度看, 神经网络实现了输入模式空间到输出模式空间的非线性映射关系.

3　基于 BP 网络的 PUMA560 机器人 IKP 求解实例(Example application of BP ANN to the IKP of PUMA560)

PUMA560 机器人的具体结构及杆件坐标系见文献[7], D-H 参数如表 1.

表 1　PUMA560 机器人的 D-H 参数

Table 1 D-H parameters of the PUMA560

关节 i	θ_i(°)	α_i(°)	a_i(mm)	d_i(mm)
1	$\theta_1 \in [-160, +160]$	90	0	0
2	$\theta_1 \in [-225, 45]$	0	431.8	149.09
3	$\theta_1 \in [-45, 225]$	0	-20.32	0
4	$\theta_1 \in [-110, 170]$	-90	0	433.07
5	$\theta_1 \in [-100, 100]$	90	0	0
6	$\theta_1 \in [-266, 266]$	0	0	56.25

为了训练网络, 由循环程序给出机器人的关节变量$(\theta_{1p}, \theta_{2p}, \theta_{3p}, \theta_{4p}, \theta_{5p}, \theta_{6p})$, 然后通过正解运算求得相应的末端操作器位姿$(x_p, y_p, z_p, O_p, A_p, T_p)$. 从而得到学习样本

$$\{x_p, y_p, z_p, O_p, A_p, T_p\} \rightarrow \{\theta_{1p}, \theta_{2p}\theta_{3p}, \theta_{4p}, \theta_{5p}, \theta_{6p}\}$$

本文在整个机器人工作空间内选取了 3840 个学习样本. 网络结构选用输入层与输出层均有 6 个节点, 分别与 6 个输入和输出变量对应. 隐层节点的选用由于没有理论指导可以借鉴, 在对各种结构进行大量学习试验的基础上, 经过比较, 选取两个隐层, 节点数分别为 19 和 17. 训练算法采用 Levenberg-Marquardt 优化方法. 经过大约 200 次迭代学习后, 网络的系统误差约为 10^{-5}. 当继续对网络进行训练学习时, 其系统误差降低的速度已经十分缓慢. 该网络训练学习迭代次数 N 与与系统误差 E 之间的关系如图 1 所示.

网络训练成功后, 测试数据由以下方法给出, 工作空间中从位姿(-0.1, 0.36, 0.32, -0.5, 0.2, 0)延直线运动到(0.45, 0, -0.33, 0.26, -0.28, 1.28)过程中 11 个均匀位置及姿态作为网络的输入, 而通过 PUMA560 机器人逆解解析解求得的相应关节向量作为网络的期望输出. 网络计算结果的误差用下式计算

$$\Delta E = [(\theta_{1t} - \theta_{1bp})^2 + (\theta_{2t} - \theta_{2bp})^2 + (\theta_{3t} - \theta_{3bp})^2 + (\theta_{4t} - \theta_{4bp})^2 + (\theta_{5t} - \theta_{5bp})^2 + (\theta_{6t} - \theta_{6bp})^2]^{1/2}$$

式中$(\theta_{1t}, \theta_{2t}, \theta_{3t}, \theta_{4t}, \theta_{5t}, \theta_{6t})$, 为机器人给定的关节向量, 即期望输出; $(\theta_{1bp}, \theta_{2bp}, \theta_{3bp}, \theta_{4bp}, \theta_{5bp}, \theta_{6bp})$, 为网络计算出的机器人关节向量, 单位采用弧度制. 附表 1 为有关求解结果. 11 组解的平均误差为 0.0166rad, 最大误差达到了 0.0212rad, 即平均角度误差为 0.953°, 最大角度误差为 1.213°. 显然, 单纯的

BP 网络求解精度难以满足机器人控制的精度要求, 必须进行误差补偿.

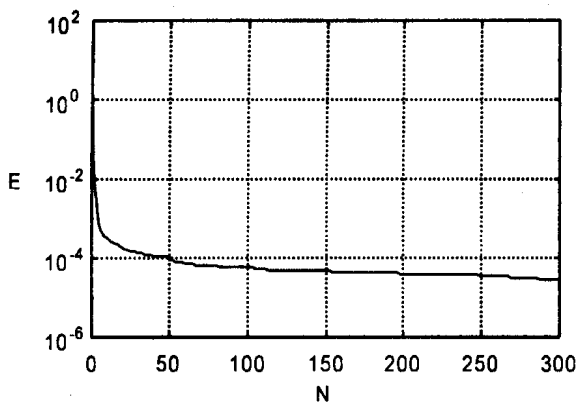

图 1 E 与 N 的关系

Fig. 1 Relation between E and N

4 误差补偿算法 (Error compensation algorithm)

为了得到满意的精度, 采用迭代计算进行误差补偿. 误差补偿算法流程如图 2 所示.

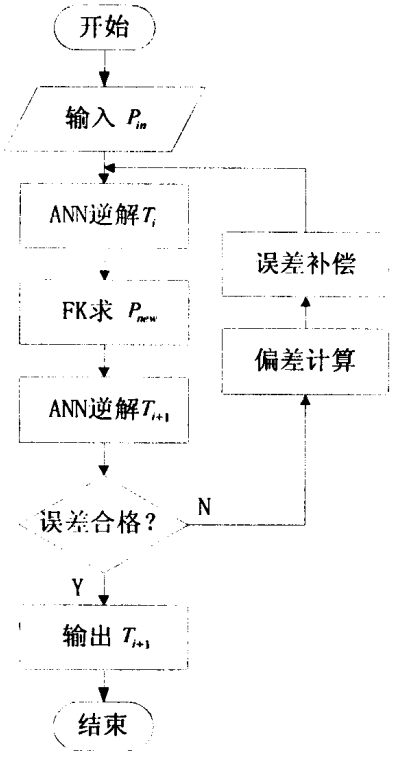

图 2 误差补偿算法流程

Fig. 2 Flowchart of error compensation method

其中, P_{new} 是对 T_i 进行正解运算所得位姿向量; T_{i+1} 为以 P_{new} 为输入经网络逆解运算所得关节向量; 误差 $Err = T_{i+1} - T_i$; 偏差 $\Delta P' = P_{new} - P_{in}$; 误差补偿是采用 $P_{in} - \Delta P'$ 作为下一循环的网络输入;

采用图示误差补偿算法之后, 逆解计算的精度得到大幅提高, 且只需很少几次迭代. 利用上述算法对测试数据进行了验证计算. 计算结果误差如附表 2 所示. 经过 3 次迭代以后, 最大误差仅为 1.507×10^{-5} rad, 已能满足机器人控制精度要求. 在 CPU 为 Pentin II 400M 的计算机上测试, 三次迭代运算时间约为 2.43ms 也已接近机器人实时控制的要求.

5 结论(Conclusion)

通过本文的计算实例可以看出, 利用神经网络和误差补偿算法求解机器人 IKP 是可行的. 与传统的求解方法相比, 它有下列特点.

(a) 神经网络通过利用正解结果训练学习, 可以实现从工作变量空间到机器人关节变量空间的复杂非线性映射关系, 避免了求解 IKP 时公式推导和编程计算等繁杂的过程. 求解计算简单, 实用性好. 同时也适用于逆解无解析解的一般机器人以及冗余机器人.

(b) 单纯的 BP 网络的求解精度难以满足机器人精确控制的要求. 可用误差补偿的方法来提高计算精度.

(c) 神经网络对信息的处理及推理的过程具有并行的特点. 因此基于神经网络的 IKP 求解是极具发展前途的方法, 因为它使得利用 VLSI 技术制造 IKP 求解器成为可能, 从而可以达到更高的求解速度, 从根本上解决 IKP 的实时求解.

总之, 应用神经网络求解机器人 IKP 是一种全新的、简单可行的方法, 同时还有一些问题有待解决. 例如应用神经网络求解机器人 IKP 的多解性问题, 奇异性等.

参考文献 (References)

1 Y F Lou, P Brunn. A hybrid artificial neural network inverse kinematic solution for accurate robot path control. Proc Instn Mech Engrs 1999, **213**(1): 23- 32

2 L Acosta, G N Marichal, *et al*. A robotic system based on neural network controllers. Artificial Intrlligence in Engineering. 1999 (13): 393- 398

3 Simon X Yang, Max Meng. An efficient neural network method for real-time motion planning with safety consideration. Robotics and Autonomous Systems, 2000(32): 115- 128

4 Bekir Karlik, Serkan Aydin. An improved approach to the solution of inverse kinematics problems for robot manipulators. Engineering Applications of Artificial Intelligence, 2000(13): 159-

164

5 D T Pham, Sahin Yildirim. Control of the trajectory of a planar robot using recurrent hybrid networks. International Journal of Machine Tools & Manufacture, 1999(39): 415- 429

6 张际先, 宓霞. 神经网络及其在工程中的应用. 北京: 机械工业出版社, 1996

7 付京逊, 杨静宇. 机器人学: 控制、传感技术、视觉、智能. 北京: 中国科学技术出版社, 1989

作者简介:

陈学生 (1975-), 男, 博士研究生. 研究领域: 并联机器人.

陈在礼 (1935-), 男, 教授, 博士生导师. 研究领域: 超声驱动器, 航天地面模拟器.

谢　涛 (1965-), 男, 副教授. 研究领域: 机器人, 航天地面模拟器.

附表 1　BP 网络对 PUMA560 机器人逆解的求解结果

Appendix 1　Results of PUMA560 IKP by using BP network　(rad)

序 号		θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	ΔE
1	目标值	- 0.1000	0.3600	0.3200	- 0.5000	0.2000	0	0.0162
	计算值	- 0.0972	0.3504	0.3165	- 0.5066	0.1916	- 0.0062	
2	目标值	- 0.0450	0.3240	0.2550	- 0.4240	0.1520	0.1280	0.0164
	计算值	- 0.0468	0.3142	0.2544	- 0.4326	0.1423	0.1275	
3	目标值	0.0100	0.2880	0.1900	- 0.3480	0.1040	0.2560	0.0193
	计算值	0.0038	0.2781	0.1946	- 0.3580	0.0943	0.2608	
4	目标值	0.0650	0.2520	0.1250	- 0.2720	0.0560	0.3840	0.0212
	计算值	0.0562	0.2423	0.1340	- 0.2811	0.0492	0.3921	
5	目标值	0.1200	0.2160	0.0600	- 0.1960	0.0080	0.5120	0.0198
	计算值	0.1103	0.2073	0.0698	- 0.2018	0.0061	0.5215	
6	目标值	0.1750	0.1800	- 0.0050	- 0.1200	- 0.0400	0.6400	0.0169
	计算值	0.1655	0.1743	0.0026	- 0.1218	- 0.0375	0.6497	
7	目标值	0.2300	0.1440	- 0.0700	- 0.0440	- 0.0880	0.7680	0.0140
	计算值	0.2226	0.1443	- 0.0644	- 0.0425	- 0.0829	0.7770	
8	目标值	0.2850	0.1080	- 0.1350	0.0320	- 0.1360	0.8960	0.0128
	计算值	0.2857	0.1153	- 0.1287	0.0354	- 0.1304	0.9012	
9	目标值	0.3400	0.0720	- 0.2000	0.1080	- 0.1840	1.0240	0.0185
	计算值	0.3500	0.0819	- 0.1907	0.1121	- 0.1802	1.0189	
10	目标值	0.3950	0.0360	- 0.2650	0.1840	- 0.2320	1.1520	0.0140
	计算值	0.3904	0.0413	- 0.2567	0.1880	- 0.2328	1.1442	
11	目标值	0.4500	0	- 0.3300	0.2600	- 0.2800	1.2800	0.0140
	计算值	0.4513	- 0.0022	- 0.3349	0.2639	- 0.2871	1.2900	

附表 2　误差补偿计算后逆解的求解误差及运算时间

Appendix 2　Errors and computing time of IKP after offset error compensation　(rad)

序 号	$\Delta E_1(10^{-2})$	$\Delta E_2(10^{-3})$	$\Delta E_3(10^{-4})$	$\Delta E_4(10^{-6})$	$\Delta E_5(10^{-7})$	$\Delta E_6(10^{-8})$
1	0.1311	0.0199	0.0066	0.0221	0.0072	0.0024
2	0.1638	0.0308	0.0094	0.0285	0.0084	0.0025
3	0.3436	0.1042	0.0497	0.2214	0.1025	0.0472
4	0.6351	0.2798	0.1507	0.8669	0.5818	0.3915
5	0.0872	0.0243	0.0077	0.0278	0.0101	0.0036
6	0.0838	0.0175	0.0046	0.0156	0.0056	0.0019
7	0.0984	0.0087	0.0009	0.0041	0.0014	0.0004
8	0.1093	0.0202	0.0059	0.0224	0.0077	0.0028
9	0.1822	0.0274	0.0080	0.0275	0.0106	0.0035
10	0.1706	0.0234	0.0046	0.0093	0.0016	0.0002
11	0.1787	0.0315	0.0082	0.0209	0.0053	0.0012
运算时间(ms)	0.9064	1.6713	2.4331	3.1789	3.9116	4.6539

注: 附表 2 中 ΔE_i, i= 1, 2, … 6, 其中 i 为迭代次数.