一种CMAC网络模型及其在机器人控制中的应用

孙 炜 王耀南
(湖南大学电气与信息工程学院,长沙 410082)

摘要 小脑模型关节控制器(CMAC)具有学习算法简单、在线学习速度快的优点,非常适于机器人等复杂系统的自适应控制.本文阐述了CMAC的原理,证明了其收敛性,提出了一种适合于机器人轨迹跟踪控制的CMAC,并给出了仿真实验结果.

关键词 CMAC,神经网络,机器人

引言

小脑模型关节控制器(CMAC)是由Albus[1]最初于1975年基于神经生理学提出的,它是一种基于局部逼近的简单快速的神经网络,类似于Perceptron的相联记忆方法,能够学习任意多维非线性映射.迄今已广泛用于许多领域.特别是Miller[2]等的突破性应用研究,已使CMAC受到越来越多的关注.

与BP网络之类的全局逼近方法不同,CMAC具有许多优点,它具有局部逼近能力,使其每次修正的权极少,学习速度快,适合于在线学习;具有一定的泛化能力,相近输入给出产生相近输出,不同输入给出不同输出;具有连续(模拟)输入输出能力;具有寻址编程方式,在利用串行计算机仿真时,它使回响速度加快.

机器人是一种具有高度非线性、强耦合的对象,且具有诸如摩擦、负载变化等不确定因素.传统的基于对象的控制方法很难精确地控制机器人的跟踪轨迹.用神经网络对其进行控制,可以利用神经网络的自学习能力取得较好的控制效果.多层反传网络收敛速度慢,所需迭代次数多,不适合机器人轨迹跟踪控制的在线学习.而CMAC学习的速度非常快,网络收敛所需的训练次数大大少于BP网络,可有效地用于机器人的实时在线控制.本文将CMAC用作伺服控制器控制机器人的轨迹跟踪,仿真实验结果表明,这种控制方法具有很好的动静态性能和很强的鲁棒性.

2005-02-11收到第1稿,2005-06-09收到修改稿.

1 CMAC技术

1.1 CMAC的原理

CMAC是一种模仿人类小脑的学习结构.如图1所示,在这种技术里,每个状态变量被量化并且问题空间被划分成离散状态.量化的输入构成的向量指定了一个离散状态并且被用于产生地址来激活联想单元中存储的联想强度从而恢复这个状态的信息.对于输入空间大的情况,联想单元数量巨大,为了节省存储空间,Albus提出了hash编码,将联想强度存于数量大大少于联想单元的hash单元中,联想单元中只存储hash单元的散列地址编码.

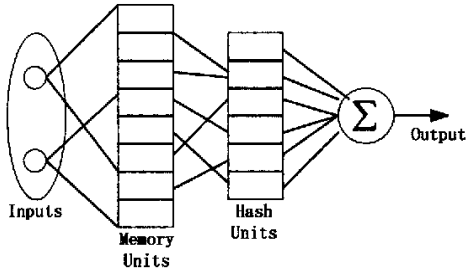

图1 CMAC的一般结构

Fig.1 Common structure of CMAC

图2描述了CMAC的空间划分和量化机制.这个简单的例子有两个状态变量(v_1 和 v_2),每个被量化为两个离散的区域叫做"块".块的宽度影响CMAC的概括能力.为了能够用较简洁的矩阵形式来描述CMAC的机制和特性,在这里我们将块的数量限定为2.譬如,v_1 被划分为A、B;v_2 被划分

为 a、b. 区域 A_a, A_b, B_a 和 B_b 被称作"超立方体"(hypercubes). 通过将每个变量平移一小段间隔(称为'元素':图 2 中 1,2,3),可以获得不同的超立方体. 例如,v_1 通过平移后的区域 C,D 和 v_2 的 c,d 组成新的超立方体 C_c,C_d,D_c,D_d. 我们规定,超立方体必须由相应的量化方式组成,例如 v_1 的第 p 种量化方式与 v_2 的第 p 种量化方式对应. 用这种方法分解,我们可以看出一共有 N_e 层的超立方体. N_e 是完整的块中的元素数量(图 2 中为 3). 第 p 层超立方体由第 p 种量化方式确定. 图 2 中可看出,两变量的每一种状态(1,2,…,16)都被 N_e 个不同的超立方体覆盖,每个超立方体覆盖 1 次. 也就是说,两变量的每一种状态将激活 N_e 个不同的超立方体. 一个输入上的块的总数和一个块的元素数决定了层数和概括的程度. CMAC 将每个超立方体和物理的存储单元(联想单元)联系在一起,每个离散状态的信息被分布存储在联想单元中,联想单元和覆盖该状态的超立方体相联系.

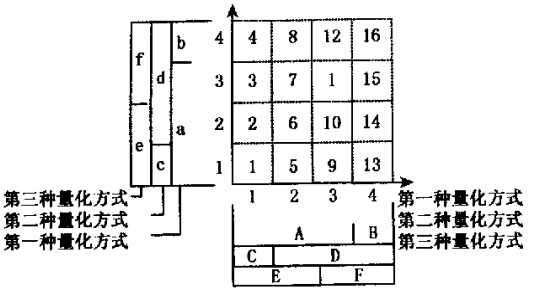

图 2　双变量 CMAC 的块划分

Fig. 2　Block division of CMAC for a two-variable case

对于输入空间较大的情况,为了减少存储空间,几个超立方体可以通过 hash 映射被分配同一个存储单元. 每个超立方体与一个"任意的但确定性的"物理存储地址(hash 单元)相联系. 当存储器空间不够时才需要 hash 映射,它增加了分析 CMAC 行为的难度.

1.2　CMAC 学习的数学推导

我们已经简要介绍了 CMAC 技术,下面我们将给出 CMAC 技术的数学公式.

1)无 hash 映射的 CMAC

每个量化的状态的信息被分布存储在 N_e 个存储位置. 假设 N_h 是超立方体的数量,该数量与没有 hash 映射时的物理存储空间大小一致. 用 CMAC 技术,一个存储的数据 ys 可以被表示为

$$y_s = C_s^{\mathrm{T}} W = [c_{s,1} \quad c_{s,2} \quad \cdots, c_{s,N_h}] \times \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{N_h} \end{bmatrix} = \sum_{j=1}^{N_h} c_{s,j}\omega_j \tag{1}$$

式中 W 是代表存储内容(联想强度)的向量,C_s 是存储单元激活向量,该向量包含 N_e 个 1.

在决定了块的划分方式后,对于指定的量化状态,单元激活向量 C_s 也随之确定. 假如存在 N_s 个训练样本,这些样本的存储数据可以用矩阵形式表示为

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_{N_s} \end{bmatrix} = \begin{bmatrix} C_1^{\mathrm{T}} \\ \vdots \\ C_{N_s}^{\mathrm{T}} \end{bmatrix} W = CW \tag{2}$$

对于图 2 中的 CMAC,有 16 个离散状态,并且安排了标号. 有 12 个联想单元分别对应超立方体 A_a, A_b, B_a, B_b, C_c, C_d, D_c, D_d, E_e, E_f, F_e 和 F_f. 这些单元从 1 到 12 按升序排列. 对每个量化空间有一个样本,用对应的状态编号来表示样本编号,存储空间的应用可以用矩阵 C 表示为

$$C = \begin{bmatrix} C_1^{\mathrm{T}} \\ C_2^{\mathrm{T}} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ C_{16}^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} 1&0&0&0&1&0&0&0&1&0&0&0 \\ 1&0&0&0&0&1&0&0&1&0&0&0 \\ 1&0&0&0&0&1&0&0&0&1&0&0 \\ 0&1&0&0&0&1&0&0&0&1&0&0 \\ 1&0&0&0&0&0&1&0&1&0&0&0 \\ 1&0&0&0&0&0&0&1&1&0&0&0 \\ 1&0&0&0&0&0&0&1&0&1&0&0 \\ 0&1&0&0&0&0&0&1&0&1&0&0 \\ 1&0&0&0&0&0&1&0&0&0&1&0 \\ 1&0&0&0&0&0&0&1&0&0&1&0 \\ 1&0&0&0&0&0&0&1&0&0&0&1 \\ 0&1&0&0&0&0&0&1&0&0&0&1 \\ 0&0&1&0&0&0&1&0&0&0&1&0 \\ 0&0&1&0&0&0&0&1&0&0&1&0 \\ 0&0&1&0&0&0&0&1&0&0&0&1 \\ 0&0&0&1&0&0&0&1&0&0&0&1 \end{bmatrix} \tag{3}$$

第一行 C_1^{T} 表示状态 1 激活了联想单元 1,5 和 9(对应于 A_a, C_c, E_e).

2) 有 hash 映射的 CMAC

hash映射将几个联想单元和一个物理存储位置(hash单元)相对应.Hash单元中存储联想强度,而此时的联想单元与超立方体一一对应,是虚拟的存储空间,只存储hash单元的散列地址编码.有hash映射的CMAC特别适用于存储空间小于超立方体数量时的情况.存储的信息从数学上可以表达为

$$y_s = [c_{s,1} \quad c_{s,2} \quad \cdots \quad c_{s,N_h}] \times \begin{bmatrix} h_{11} & \cdots & h_{1,M_p} \\ \vdots & & \\ h_{N_h,1} & \cdots & h_{N_h,M_p} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{M_p} \end{bmatrix} = C_s^{\mathrm{T}} HW \tag{4}$$

式中 Mp 是hash单元的大小,它小于超立方体数 N_h. $h_{ij} = 1$ 表示联想单元 i 激活hash单元 j.由于每个联想单元仅仅和一个hash单元相对应,所以hash矩阵 H 的每一行仅有一个单元等于1,其余的都等于0.和式(2)相似,对于训练样本集的公式为

$$Y = \begin{bmatrix} C_1^{\mathrm{T}} \\ C_2^{\mathrm{T}} \\ \vdots \\ C_s^{\mathrm{T}} \end{bmatrix} HW = CHW \tag{5}$$

没有hash映射的式(2)和有hash映射的式(5)可以合并为一个式子

$$Y = AW = \begin{bmatrix} a_1^{\mathrm{T}} \\ a_2^{\mathrm{T}} \\ \vdots \\ a_{N_s}^{\mathrm{T}} \end{bmatrix} W \tag{6}$$

式中 $A \equiv CH$, $H = \begin{cases} I_{N_h \times N_h} & \text{无 hash 映射} \\ H_{N_h \times M_p} & \text{有 hash 映射} \end{cases}$

假如CMAC的结构如图2所示,C 如式(3)所示,下面给出一个 H 和 A 的例子

$$H = \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1&0&1&0&1&0&0&0 \\ 1&0&1&0&0&1&0&0 \\ 1&0&0&0&1&1&0&0 \\ 0&1&0&0&1&1&0&0 \\ 1&0&1&0&0&0&1&0 \\ 1&0&1&0&0&0&0&1 \\ 1&0&0&0&1&0&0&1 \\ 0&1&0&0&1&0&0&1 \\ 1&1&0&0&0&0&1&0 \\ 1&1&0&0&0&0&0&1 \\ 1&0&0&0&0&0&1&1 \\ 0&1&0&0&0&0&1&1 \\ 0&1&1&0&0&0&1&0 \\ 0&1&1&0&0&0&0&1 \\ 0&0&1&0&0&0&1&1 \\ 0&0&0&1&0&0&1&1 \end{bmatrix} \tag{7}$$

3) 学习

CMAC用迭代算法来产生存储信息.尽管在实际的CMAC学习过程中,是从训练样本中随机选取标本,但在研究其收敛特性时,这样做是非常困难甚至是不可能的.因此,在这次学习中,我们将 N_s 个训练数据重复用于学习.在第 i 次迭代中用第 s 个样本学习的迭代算法为

$$W_s^{(i)} = W_{s-1}^{(i)} + \Delta W_{s-1}^{(i)} = W_{s-1}^{(i)} + \frac{\alpha}{N_e} a_{s-1} (\hat{y}_{s-1} - a_{s-1}^{\mathrm{T}} W_{s-1}^{(i)}) \tag{8}$$

式中下标 $s-1$ 和 s 表示样本数,上标 i 表示迭代次数,为学习率,$\hat{y}_{s-}$ 是样本 $s-1$ 的目标值.注意 $\hat{y}_{s-} - a_{s-1}^{\mathrm{T}} W_{s-1}^{(i)}$ 是样本 $s-1$ 的误差.向量 $W_s^{(i)}$ 给出了在第 i 次迭代时用第 s 个样本进行学习时的联想强度.

1.3 学习算法的收敛性分析

关于CMAC学习的收敛性问题,Wong和Sideris[3]进行了证明但只局限于存储空间大于联想强度数量并且没有hash映射的情况.Parks和Miltizer[4]给出了较完整的证明,他们定义了一个Lyapunov函数并用它证明了CMAC在学习率为1时可以收敛.本文中,我们清楚地用数学公式定义了CMAC技术.基于这些公式,我们进一步证明了当学习的迭代次数趋于无穷并且学习率趋于0时,学习结果收敛于最小方差值.

定理1 当迭代次数趋于无穷且学习率趋于0时,CMAC的学习结果收敛于最小平方差.

证明　由式(8)得出两次连续迭代之间的存储内容之差为

$$\begin{aligned}Dw_s^{(i)} &= W_s^{(i+1)} - W_s^i = W_{s-1}^{(i+1)} + \Delta W_{s-1}^{(i+1)} - \\ &(W_{s-1}^{(i)} + \Delta W_{s-1}^{(i)}) = Dw_{s-1}^{(i)} + \\ &\frac{\alpha}{N_e}a_{s-1}(\hat{y}_{s-1} - a_{s-1}^{\mathrm{T}}w_{s-1}^{(i+1)}) - \\ &\frac{\alpha}{N_e}a_{s-1}(\hat{y}_{s-1} - a_{s-1}^{\mathrm{T}}w_{s-1}^{(i)}) = \\ &Dw_{s-1}^{(i)} - \frac{\alpha}{N_e}a_{s-1}a_{s-1}^{\mathrm{T}}(w_{s-1}^{(i+1)} - \\ &w_{s-1}^{(i)}) = \left(I - \frac{\alpha}{N_e}a_{s-1}a_{s-1}^{\mathrm{T}}\right)Dw_{s-1}^{(i)} \end{aligned} \tag{9}$$

对于 $s = 1$ 的情况,定义 $Dw_0^{(i+1)} \equiv Dw_{N_s}^{(i)}, a_0 \equiv a_{N_s}$.

通常定义

$$E_s \equiv I - \frac{\alpha}{N_e}a_s a_s^{\mathrm{T}} \tag{10}$$

$$Dw^{(i)} = [Dw_1^{(i)}, Dw_2^{(i)}, \cdots, Dw_{N_s}^{(i)}] \tag{11}$$

由式(9) ~ 式(11)可推导出

$$\begin{aligned}Dw^{(i)} &= [Dw_1^{(i)}, Dw_2^{(i)}, \cdots, Dw_{N_s}^{(i)}] = \\ &[E_{N_s}Dw_{N_s}^{(i-1)}, E_1Dw_1^{(i)}, E_2Dw_2^{(i)}, \\ &\cdots, E_{N_{s-1}}Dw_{N_s-1}^{(i)}] = \\ &[E_{N_s}E_{N_{s-1}}Dw_{N_s-1}^{(i-1)}, E_1E_{N_s}Dw_{N_s}^{(i-1)}, \\ &\cdots, E_{N_s-2}E_{N_s-3}Dw_{N_s-3}^{(i)}, \\ &E_{N_s-1}E_{N_s-2}Dw_{N_s-2}^{(i)}] = \\ &[(E_{N_s}E_{N_{s-1}}\cdots E_1)Dw_1^{(i-1)}, \\ &(E_1E_{N_s}\cdots E_2)Dw_2^{(i-1)}, \cdots, \\ &(E_{N_s-1}E_{N_s-2}\cdots E_{N_s})Dw_{N_s}^{(i-1)}] \end{aligned} \tag{12}$$

为了简化,接着定义

$$G_s \equiv E_{s-1}E_{s-2}\cdots E_1E_{N_s}\cdots E_s \tag{13}$$

则

$$\begin{aligned}Dw^{(i)} &= [G_1Dw_1^{(i-1)}, G_2Dw_2^{(i-1)}, \\ &\cdots, G_{N_s}Dw_{N_s}^{(i-1)}] = [G_1^2Dw_1^{(i-2)}, \\ &G_2^2Dw_2^{(i-2)}, \cdots, G_{N_s}^2Dw_{N_s}^{(i-2)}] = \\ &[G_1^iDw_1^{(0)}, G_2^iDw_2^{(0)}, \cdots, \\ &G_{N_s}^iDw_{N_s}^{(0)}] \end{aligned} \tag{14}$$

由式(9)得

$$\begin{aligned}Dw_s^{(0)} &= W_s^{(1)} - W_s^{(0)} = W_{s-1}^{(1)} + \Delta W_{s-1}^{(1)} - \\ &W_s^{(0)} = W_1^{(1)} + \Delta W_1^{(1)} + \cdots + \\ &\Delta W_{s-1}^{(1)} - W_s^{(0)} = W_{N_s}^{(0)} + \Delta W_{N_s}^{(0)} + \\ &\Delta W_1^{(1)} + \cdots + \Delta W_{s-1}^{(1)} - W_s^{(0)} = \\ &W_s^{(0)} + \Delta W_s^{(0)} + \Delta W_{s+1}^{(0)} + \cdots + \\ &\Delta W_{N_s}^{(0)} + \Delta W_1^{(1)} + \cdots + \Delta W_{s-1}^{(1)} - \\ &W_s^{(0)} = \Delta W_s^{(0)} + \Delta W_{s+1}^{(0)} + \cdots + \\ &\Delta W_{N_s}^{(0)} + \Delta W_1^{(1)} + \cdots + \Delta W_{s-1}^{(1)} \end{aligned} \tag{15}$$

$$\begin{aligned}W_s^{(i+1)} &= Dw_s^{(i)} + W_s^{(i)} = Dw_s^{(i)} + Dw_s^{(i-1)} + \\ &W_s^{(i-1)} = Dw_s^{(i)} + Dw_s^{(i-1)} + \cdots + \\ &Dw_s^{(1)} + Dw_s^{(0)} + W_s^{(0)} = \\ &\sum_{k=0}^{i} W_s^{(k)} + W_s^{(0)} \end{aligned} \tag{16}$$

将式(14)和式(15)代入式(16)得

$$\begin{aligned}W_s^{(i+1)} &= \sum_{k=0}^{i}(G_s^kDw_s^{(0)}) + W_s^{(0)} = \\ &\sum_{k=0}^{i}G_s^k(\Delta W_s^{(0)}) + \Delta W_{s+1}^{(0)} + \cdots + \\ &\Delta W_{N_s}^{(0)}) + \Delta W_1^{(1)} + \cdots + \Delta W_{s-1}^{(1)}) + \\ &W_s^{(0)} \end{aligned} \tag{17}$$

由式(8)得

$$\Delta W_s^{(i)} = \frac{\alpha}{N_e}a_s(\hat{y}_s - a_s^{\mathrm{T}}W_s^{(i)}) \tag{18}$$

将式(8),式(15),式(18)代入式(17)得

$$\begin{aligned}W_s^{(i+1)} &= \sum_{k=0}^{i}G_s^k(\Delta W_s^{(0)} + \Delta W_{s+1}^{(0)} + \cdots + \\ &\Delta W_{N_s}^{(0)} + \Delta W_1^{(1)} + \cdots + \Delta W_{s-1}^{(1)}) + W_1^{(0)} + \\ &\Delta W_1^{(0)} + \cdots + \Delta W_{s-1}^{(0)} = \sum_{k=0}^{i}G_s^k\frac{\alpha}{N_e}[a_s(\hat{y}_s - \\ &a_s^{\mathrm{T}}W_s^{(0)}) + \cdots + a_{N_s}(\hat{y}_{N_s} - a_{N_s}^{\mathrm{T}}W_{N_s}^{(0)}) + \\ &a_1(\hat{y}_1 - a_1^{\mathrm{T}}W_1^{(1)}) + \cdots + a_{s-1}(\hat{y}_{s-1} - \\ &a_{s-1}^{\mathrm{T}}W_{s-1}^{(1)})] + W_1^{(0)} + \frac{\alpha}{N_e}[a_1(\hat{y}_1 - \\ &a_1^{\mathrm{T}}W_1^{(0)}) + a_1(\hat{y}_2 - a_2^{\mathrm{T}}W_2^{(0)} + \cdots + \\ &a_{s-1}(\hat{y}_{s-1} - a_{s-1}^{\mathrm{T}}W_{s-1}^{(0)})] = \sum G_s^k\frac{\alpha}{N_e}\times \\ &[(a_1\hat{y}_1 + a_2\hat{y}_2 + \cdots + a_{N_s}\hat{y}_{N_s}) - a_1a_1^{\mathrm{T}}W_1^{(1)} + \\ &a_{s-1}a_{s-1}^{\mathrm{T}}W_{s-1}^{(1)} + a_sa_s^{\mathrm{T}}W_s^{(0)} + \cdots + \\ &a_{N_s}a_{N_s}^{\mathrm{T}}W_{N_s}^{(0)})] + W_1^{(0)} + O(\alpha) \end{aligned} \tag{19}$$

下面进一步对式(19)中的 $a_ja_j^{\mathrm{T}}W_j^{(1)}(j = 1,2,\cdots,s-1)$ 和 $a_ka_k^{\mathrm{T}}W_k^{(0)}(k = s, s+1, \cdots, N_s)$ 进行推导.

$$\begin{aligned}a_ja_j^{\mathrm{T}}W_j^{(1)} &= a_ja_j^{\mathrm{T}}(W_{j-1}^{(1)} + \Delta W_{j-1}^{(1)}) = \\ &a_ja_j^{\mathrm{T}}(W_1^{(0)} + \Delta W_1^{(0)} + \cdots + \Delta W_{N_s}^{(0)} + \end{aligned}$$

$$\Delta W_1^{(1)} + \cdots + \Delta W_{j-1}^{(1)} = a_j a_j^{\mathrm{T}}(W_1^{(0)} +$$
$$a_j a_j^{\mathrm{T}} \frac{\alpha}{N_s}[a_1(\hat{y}_1 - a_1^{\mathrm{T}} W_1^{(0)}) + \cdots +$$
$$a_{j-1}(\hat{y}_1 - a_{j-1}^{\mathrm{T}} W_{j-1}^{(1)}] \equiv$$
$$a_j a^{\mathrm{T}} W_1^{(0)} + \alpha Z_j$$
$$a_k a_k^{\mathrm{T}} W_k^{(0)} = a_k a_k^{\mathrm{T}}(W_{k-1}^{(0)} + \Delta W_{k-1}^{(0)}) =$$
$$a_k a_k^{\mathrm{T}}(W_1^{(0)} + \Delta W_1^{(0)} + \cdots + \Delta W_{k-1}^{(0)}) =$$
$$a_k a_k^{\mathrm{T}} W_1^{(0)} + a_k a_k^{\mathrm{T}} \frac{\alpha}{N_e}[a_1(\hat{y}_1 -$$
$$a_1^{\mathrm{T}} W_1^{(0)}) + \cdots + a_{k-1}(\hat{y}_1 - a_{k-1}^{\mathrm{T}} W_{k-1}^{(0)})] \equiv$$
$$a_k a^{\mathrm{T}} W_1^{(0)} + \alpha Z_k$$

由此式(19)可以简化为

$$W_s^{(i+1)} = \sum_{k=0}^{i} G_s^k \frac{\alpha}{N_e}[A^{\mathrm{T}}\hat{Y} - (a_1 a_1^{\mathrm{T}} W_1^{(0)} +$$
$$a_2 a_2^{\mathrm{T}} W_1^{(0)} + \cdots + a_{N_s} a_{N_s}^{\mathrm{T}} W_1^{(0)}) -$$
$$\alpha \sum_{j=1}^{N_s} Z_j] + W_1^{(0)} = \sum_{k=0}^{i} G_s^k \frac{\alpha}{N_e}[A^{\mathrm{T}}\hat{Y} -$$
$$A^{\mathrm{T}} A W_1^{(0)} - O(\alpha)] + W_1^{(0)} O(\alpha) \quad (20)$$

当迭代次数趋于无穷并且学习率 α 趋于零时,等式变为

$$\lim_{\alpha\to 0}\lim_{i\to\infty} W_s^{(i+1)} = \lim_{\alpha\to 0}\lim_{i\to\infty}\left\{\sum_{k=0}^{i} G_s^k \frac{\alpha}{N_e}[A^{\mathrm{T}}\hat{Y} -\right.$$
$$A^{\mathrm{T}} A W_1^{(0)} - O(\alpha)] + W_1^{(0)} + O(\alpha)\} =$$
$$\lim_{\alpha\to 0}\{(I - G_s)^{-1}\frac{\alpha}{N_e}[A^{\mathrm{T}}\hat{Y} - A^{\mathrm{T}} A W_1^{(0)} -$$
$$O(\alpha)] + W_1^{(0)} + O(\alpha)\} \quad (21)$$

由式(10)和式(13)得

$$G_s \equiv E_{s-1}E_{s-2}\cdots E_1 E_{N_s}\cdots E_s =$$
$$\left(I - \frac{\alpha}{N_e} a_{s-1} a_{s-1}^{\mathrm{T}}\right)\left(I - \frac{\alpha}{N_e} a_{s-2} a_{s-2}^{\mathrm{T}}\right)\cdots$$
$$\left(I - \frac{\alpha}{N_e} a_s a_s^{\mathrm{T}}\right) = I - \frac{\alpha}{N_e}(a_1 a_1^{\mathrm{T}} +$$
$$a_2 a_2^{\mathrm{T}} + \cdots + a_{N_s} a_{N_s}^{\mathrm{T}}) + O(\alpha^2) =$$
$$I - \frac{\alpha}{N_e}(A^{\mathrm{T}} A) + O(\alpha^2) \quad (22)$$

将式(22)代入式(21)得

$$\lim_{\alpha\to 0}\lim_{i\to\infty} W_s^{(i+1)} = \lim_{\alpha\to 0}\left\{\left[\frac{\alpha}{N_e}(A^{\mathrm{T}} A) -\right.\right.$$
$$\left.O(\alpha^2)\right]^{-1}\frac{\alpha}{N_e}[A^{\mathrm{T}}\hat{Y} - A^{\mathrm{T}} A W_1^{(0)} -$$
$$\left.O(\alpha)]\right\} + W_1^{(0)} + \lim_{\alpha\to 0} O(\alpha) =$$
$$\lim_{\alpha\to 0}\{[A^{\mathrm{T}} A - O(\alpha)]^{-1}[A^{\mathrm{T}}\hat{Y} -$$
$$A^{\mathrm{T}} A W_1^{(0)} - O(\alpha)]\} + W_1^{(0)} =$$
$$(A^{\mathrm{T}} A)^{-1} A^{\mathrm{T}}\hat{Y} - (A^{\mathrm{T}} A)^{-1} \times$$
$$A^{\mathrm{T}} A W_1^{(0)} + W_1^{(0)} = (A^{\mathrm{T}} A)^{-1} A^{\mathrm{T}}\hat{Y}$$
(23)

式(23)证明了当迭代次数趋于无穷且学习率趋于零时,CMAC 的学习结果等价于 $(A^{\mathrm{T}} A)^{-1} A^{\mathrm{T}}\hat{Y}$,收敛于最小平方差.在实际应用中,往往先采用一个较大的学习率,然后再将其逐渐减小,以获得近似的最小平方差结果.

2 机器人的动态控制问题

对于具有多个自由度的多关节机器人来说,每个关节的驱动力矩都由伺服控制器根据各个关节的期望轨迹给定.在本文中伺服控制器采用 CMAC.由于机器人本身具有固有的动态特性,所以为了使各个关节能够以理想的动态性能无静差地跟踪期望轨迹,伺服控制系统必须采用反馈结构.而反馈伺服系统的参考输入信号,就是由监控系统根据上位计算机的操作指令而生成的期望轨迹.

下面以二关节机械手为例,给出控制系统的框图和机械手的数学模型.

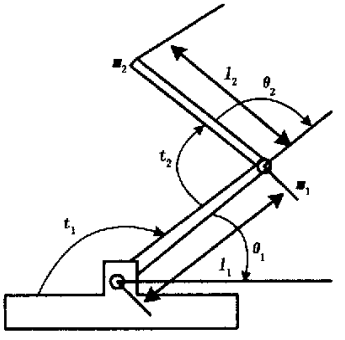

图 3 二关节机械手

Fig. 3 Robotic manipulator with two links

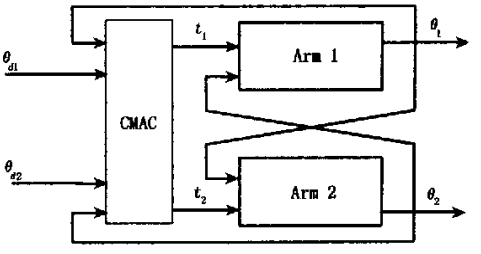

图 4 机械手控制系统框图

Fig. 4 Control system of robotic manipulator

图 3 是二关节机械手的示意图,图中 m_1, m_2 是杆 1 和杆 2 的质量.假设质量集中于杆的末梢. l_1, l_2 分别为杆 1 和杆 2 的长度. t_1, t_2 分别为作用在杆 1 和杆 2 上的驱动力矩. θ_1, θ_2 为杆 1 和杆 2 在 t_1, t_2 作用下转过的角度.

图 4 给出了机械手控制系统的框图,图中 θ_{d1} 和 θ_{d2} 分别是杆 1 和杆 2 的期望转角.

图 3 所示的机械手的数学模型可描述为

$$M(\Theta)\ddot{\Theta} + V(\Theta,\dot{\Theta})\dot{\Theta} + G(\Theta) + F(\dot{\Theta}) + T_d(\Theta,\dot{\Theta}) = T \tag{24}$$

式中, $\Theta = [\theta_1 \quad \theta_2]^{\mathrm{T}}, \dot{\Theta} = [\dot{\theta}_1 \quad \dot{\theta}_2]^{\mathrm{T}}, \ddot{\Theta} = [\ddot{\theta}_1 \quad \ddot{\theta}_2]^{\mathrm{T}}, T = [t_1 \quad t_2]^{\mathrm{T}}$. $M(\Theta)$ 是惯性矩阵, $V(\Theta,\dot{\Theta})$ 是离心力和哥氏力矩阵, $G(\Theta)$ 是重力矩阵, $F(\dot{\Theta})$ 是静态和动态摩擦矩阵, $T_d(\Theta,\dot{\Theta})$ 代表所有由负载变化或建模误差所引起的扰动.

令 c_i 代表 $\cos\theta_i$, s_i 代表 $\sin\theta_i$, c_{ij} 代表 $\cos(\theta_i + \theta_j)$. 则

$$M(\Theta) = \begin{bmatrix} m_1 l_1^2 + m_2(l_1^2 + l_2^2 + 2l_1 l_1 c_2) & m_2 l_2^2 + m_2 l_1 l_2 c_2 \\ m_2 l_2^2 + m_2 l_1 l_2 c_2 & m_2 l_2^2 \end{bmatrix} \tag{25}$$

$$V(\Theta,\dot{\Theta}) = \begin{bmatrix} m_1 l_1 l_2 s_2 \dot{\theta}_2^2 - 2m_2 l_1 l_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ m_2 l_1 l_2 s_2 \dot{\theta}_1^2 \end{bmatrix} \tag{26}$$

$$G(\Theta) = \begin{bmatrix} m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1 \\ m_2 l_2 g c_{12} \end{bmatrix} \tag{27}$$

3　用于机器人轨迹跟踪控制的 CMAC

本节以两关节机械手为例来说明用于机器人轨迹跟踪控制的 CMAC 的结构.图 5 所示是用于两关节机械手轨迹跟踪控制的 CMAC 的结构,CMAC 在控制系统中充当伺服控制器的角色.该控制器由输入、联想单元、加法器和输出四部分构成.由于输入空间不是很大,所需存储空间不大,所以没有使用 hash 映射.

输入空间由各关节的误差、误差变化率构成,每个关节的误差、误差变化率构成一个输入子空间.每个子空间均为二维空间,对于每个子空间分别采用 2.1 节中图 2 描述的方法进行空间划分.由于各关节之间存在耦合作用,所有的输入子空间对每个输出均有不同程度的影响,对于每个输出都有一组联想单元与之对应,用来存储所有输入对其的联想强度.

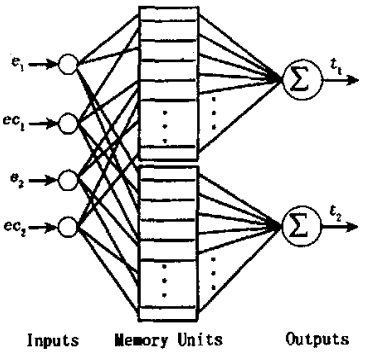

图 5　用于两关节机械手控制的 CMAC

Fig. 5　CMAC for robotic manipulator with two links

在控制过程中,各关节的误差、误差变化率经空间划分后对应于相应的状态,进而激活联想单元中联想强度,经求和后得到各关节的期望转矩.各组联想单元中存储的联想强度根据式(8)进行学习,学习的初始阶段,学习率取较大值,随后逐渐减小,以使 CMAC 最终收敛于期望的平方误差.

4　仿真实验结果

本文对一个两关节机械手(如图 3)进行仿真以验证所提方法的正确性.用于仿真的机械手的参数为 $m_1 = 10$ kg, $m_2 = 2$ kg 和 $l_1 = 1.1$ m, $l_2 = 0.8$ m.初始条件为 $\theta_1(0) = \theta_2(0) = 0$ rad, $\dot{\theta}_1(0) = \dot{\theta}_2(0) = 0$ rad/s.期望轨迹为 $\theta_1^d(t) = \theta_2^d(t) = \sin(2\pi t)$,采样周期为 0.0005 s.摩擦项和扰动项分别为 $F(\dot{\Theta}) = 0.5\ \mathrm{sign}(\dot{\Theta})$, $T_d(\Theta,\dot{\Theta}) = \begin{bmatrix} 5\cos(5t) \\ 5\cos(5t) \end{bmatrix}$ N·m.在对 CMAC 的输入空间进行划分时,块的数量为 2,块中的元素数 N_e 为 3,状态数为 16.图 6 和图 7 分别给出了关节 1 和关节 2 的跟踪轨迹曲线.从图 6 和图 7 可看出机械手的各关节能很好地跟踪期望轨迹,且具有很好的抗干扰性能.

5　结论

本文详细阐述了小脑模型关节控制器(CMAC)的工作原理和数学描述,证明了其收敛性.利用其结构简单、学习速度快、适合于在线学习的优点,提出了一种用于机器人轨迹跟踪控制的 CMAC,并将之用于对两关节机械手的控制.仿真实验结果表明,所提控制器具有较好的动静态性能和较强的鲁棒性,

是一种有效的控制器.

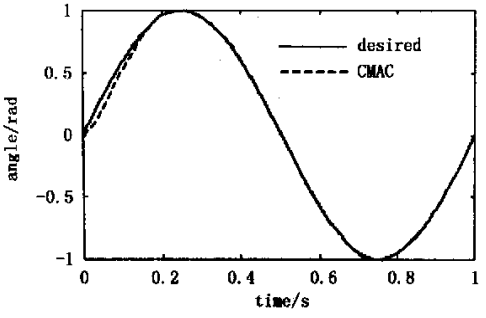

图6 关节1的轨迹跟踪曲线

Fig.6 Tracking curve of link1

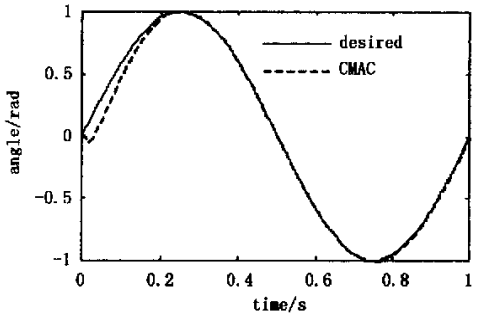

图7 关节2的轨迹跟踪曲线

Fig.7 Tracking curve of link2

参 考 文 献

1 Albus JS. A new approach to manipulator control: The cerebellar model articulation controller(CMAC). *Journal of Dynamic Systems, Measurement, and Control Transactions of ASME*, 1975, (12): 220~227

2 Millerm TW, Glanz FH, Kraft LG. An associative neural network alternative to backpropagation. *Proceedings of IEEE*, 1990, (78): 1561~1567

3 Wong Y, Sideris A. Learning convergence in the cerebellar model articulation controller. *IEEE Trans. on Neural Networks*, 1992, (3): 115~121

4 Parks PC, Miller J. Convergence properties of associative memory storage for learning control system. *Automation and Remote Control*, 1989, (50): 254~286

5 Lin CS, Chiang CT. Learning convergence of CMAC technique. *IEEE Trans. on Neural Network*, 1997, (8): 1281~1292

6 GE S. Advanced control techniques of robotics manipulators. *New York: Proceedings of Americal Control Conference*, 1998

7 王耀南. 计算智能信息处理技术及其应用. 长沙: 湖南大学出版社, 1999(Wang Yaonan. Computational intelligent information processing: technology and applications. Changsha: Hunan University Press, 1999(in Chinese))

AN CMAC NETWORK MODEL AND ITS APPLICATION TO ROBOTIC CONTROL

Sun Wei Wang Yaonan

(*College of electrical and informational engineering, Hunan University, Changsha* 410082, *China*)

Abstract The cerebellar model articulation controller (CMAC) has simple learning algorithm and high on-line learning speed. It is very useful to the self-adaptive control of complicated plant such as robot. In this paper, the principle of CMAC was described, and the leaning convergence was proved. An CMAC designed for robotic manipulator tracking control was proposed, and the simulations results were presented.

Key words CMAC, neural network, robot

Received 11 February 2005, revised 09 June 2005.