DOI: 10.16511/j.cnki.qhdxxb.2006.01.035

ISSN 1000-0054 清华大学学报 (自然科学版) 2006年 第 46卷 第 1期 35/41
CN 11-2223/N J Tsinghua Univ (Sci & Tech), 2006, Vol. 46, No. 1 133-136

基于多层神经网络控制的动态自适应播放

曾 涛, 戴琼海

(清华大学 自动化系, 北京 100084)

摘 要: 基于流媒体客户端缓冲的被动变化可以通过自适应媒体播放(AMP)技术修改播放速度来调节,为了减轻AMP中速度突变对播放质量的影响,提高变速的性能,提出了一种基于多层神经网络控制的AMP方法,产生随当前缓冲动态变化的速度,并且保持缓冲稳定在一定的范围内。该方法引入了多层神经网络控制结构,并采用反向传播学习算法(BP)进行离线训练。仿真结果表明: 该方法产生的速度平均幅值和变化增量可以减少0~100%,性能比原AMP方法更好。

关键词: 多媒体通信; 缓冲; 自适应媒体播放; 神经网络

中图分类号: TN 919.8 文献标识码: A

文章编号: 1000-0054(2006)01-0133-04

Adaptive media playout based on multiplayer neural network control

ZENG Tao, DAI Qionghai

(Department of Automation, Tsinghua University, Beijing 100084, China)

Abstract Adaptive media playout (AMP) actively adjusts to the passive variations of the streaming client buffer's size by changing the playout speed to improve the stability. An AMP scheme based on the multilayer neural network control was developed to reduce the speed variations in previous AMP systems and improve the performance. The system dynamically exports the speed variations based on the current buffer's status and ensures that the buffer is stable inside a fixed range. The system uses a multilayer neural network architecture trained offline by a backpropagation (BP) learning algorithm. Extensive simulations confirmed that the scheme gives 0~100% reduction of the average speed variations, which is better than the original AMP method.

Key words multimedia communications; buffer; adaptive media playout; neural networks

流媒体数据的实时连续性要求每帧数据必须在其播放之前到达客户端[1],实际网络信道所造成的随机数据丢失带来的抖动可以通过客户端的一段缓冲区平滑. 但是, 较小的缓冲在较差的网络环境中可能很快就会耗尽,即缓冲下溢; 较大的缓冲带来的过分延迟又不适合于实时性要求较高的应用。

自适应媒体播放技术[2-4]的提出为缓冲的随机变化问题找到了一种主动修正的平衡方案。与传统的码率控制[5]的方法不同,它只基于客户端的调节,利用了播放速度对缓冲的相关性,实时减缓缓冲的被动变化。速度加快可以减少缓冲,反之可以增加缓冲。这种方法提高了客户端抵御恶劣网络环境的能力,在下溢概率一定时获得了更小的平均延迟。

AMP-Mean是文[3]中提出的一种使用静态的变速因子的自适应媒体播放(AMP)方法,能在缓冲过多时加速播放来减小平均延迟. 然而,固定的速度因子会造成速度的突然变化,给用户的观看带来较差的效果。本文将针对这个问题提出一种动态自适应播放的方法,实现距稳定值越远的缓冲能获得更大幅度变速的目标,这样可以保证播放速度的变化具有更小的递增量而实现随缓冲渐进的变化,同时也降低了速度的平均幅值。为了实现速度的动态变化,需要考虑更加复杂的控制结构。引入了神经网络控制方法,利用其自适应性、学习能力和函数逼近性的优势,可以产生更为渐变的播放速度,减少了因速度突变造成的对播放质量的损害。

1 神经网络控制系统模型

1.1 系统模型

客户端缓冲可以简化为图1所示的系统模型。缓冲中的阴影部分表示已有的数据,假设为 L_c。L_n 是期望缓冲稳定值,令 $L_t = L_c - L_n$ 表示当前缓冲与稳定值的差值。

收稿日期: 2005-01-10

基金项目: 国家自然科学基金资助项目 (60432030)

作者简介: 曾涛(1980-),男(汉),四川,硕士研究生。

通讯联系人: 戴琼海,副教授,

E-mail: qhdai@mail.tsinghua.edu.cn

假设在一个周期 T 中,会有 $(1-q)T$ 的数据(设网络信道中的数据丢失率为 $q\in[0,1]$)进入缓冲。同时,设客户端从缓冲中取出的数据量为 sT,其中 s 为播放速度因子,慢速播放时 $0<s<1$,快速播放时 $s>1$,定义变速的幅度因子 $u=s-1$,则第 $m+1$ 个周期后缓冲中已有的数据长度更新为

$$L_{c,m+1}-L_{c,m}=(1-q_m)T-s_mT=-(q_m+u_m)T. \tag{1}$$

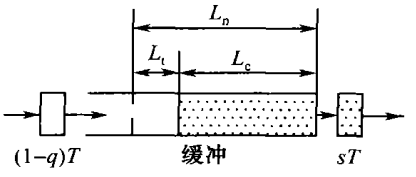

图 1 客户端缓冲模型

1.2 改进策略

为了在保证缓冲稳定性不受影响时获得更好的控制效果,在控制结构设计中还作了如下改进。

1) 限定播放速度的变化范围。播放速度如果变化的幅值过大,就会损害最终的播放质量。为此,在神经网络控制器输出之后引入一个对称饱和线性函数 sat,限制输出在 $[-1,1]$之间,并且乘上一个比例常数 C 保证最终的变速幅度因子 u 始终在 $[u_{\min},u_{\max}]=[-C,C]$中。通常认为当变速幅值不超过 25% 时变速对用户的影响可以忽略[3],这里取 $C=0.25$。为了使控制器输入与 L_n 无关,相应地规定播放速度只能在 $L\in[-L_e,L_e]$时,才会随缓冲动态变化。控制器输入使用了一个更标准化的比例度量 $L_t=L_t/L_e$ 来代替原来的 L_t 的作用。设 $L_c\in[0,2L_n]$时缓冲是非溢出的,则 I_t 的取值范围为 $I_t\in[-L_n/L_e,L_n/L_e]$。

2) 限定播放速度的变化频率。由于每隔周期 T 就会输出一个新的速度,如果 T过小即速度频繁的变化也会损害最终的播放质量。可以设定上下阈值 $L_{\max}$和 $L_{\min}$,只有缓冲长度 L_c在 $[L_{\min},L_{\max}]$之外时才会启动神经网络控制器,否则按照正常速度播放。这样大大减少了控制器的实际作用次数,同时也能保证缓冲稳定在 $[L_{\min},L_{\max}]$中。此外,由于缓冲在逼近稳定值时,变速的幅度会越来越小,直到无法抵消丢包带来的被动变化,导致缓冲无法收敛到理想值。这个限制也避免了这个问题的发生。

1.3 控制模型

这里构建了一个合适的控制模型(如图 2所示)。整个模型可以分为控制器、执行器和学习器。

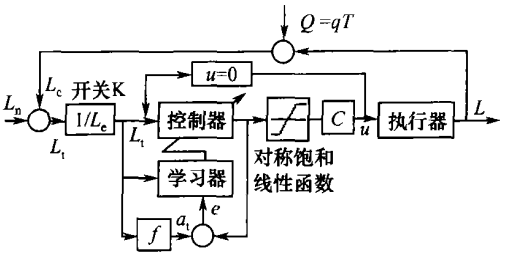

图 2 神经网络控制模型

神经网络控制器(NNC)采用的是多层的神经网络结构。根据一个可以度量当前缓冲满溢程度的输入 I_t,周期性地计算出新的合适的变速幅度因子 u 来抑制由于网络丢包等因素造成的缓冲的被动变化,保证当前缓冲长度 L_c可以趋于 L_n。执行器由客户端的音视频变速播放实现。

为了保证控制器输出的播放速度能达到期望的动态变化,必须要先通过学习器对神经网络进行离线训练,保证控制器对目标函数 f的逼近。初始时输入一些随机的样本 $I_{t,k}$,通过控制器得到一系列的输出 a_k。相应的目标输出值为

$$a_{t,k}=f(I_{t,k}), \tag{2}$$

其中 f 是单调递增函数,可以根据实际控制需要取不同的形式。

在学习器中比较控制器目标输出 a_t和实际输出 a的差值 e,并采用相应的学习算法更新神经网络控制器的权系数直到 e趋于 0。经过训练后的控制器具有收敛的权系数,用于实际的缓冲控制时。输出的播放速度 u将逼近于期望的速度 u_t。

根据上面的模型结构和约束策略,可以定义一个输入输出的映射规则,见表 1,其中 $u=Cf(I_t)$,$u=C\,\mathrm{sat}(\mathrm{nnc}(I_t))$,nnc是控制器函数。

表 1 当前缓冲与期望或实际播放速度的映射关系

L_t	I_t	u_t	u
$L_t>L_e$	$I_t>1$	$u_t>u_{\max}$	$u=u_{\max}$
$0<L_t\leqslant L_e$	$0<I_t\leqslant 1$	$0<u_t\leqslant u_{\max}$	$0<u\leqslant u_{\max}$
$L_t=0$	$I_t=0$	$u_t=0$	$u=0$
$-L_e\leqslant L_t<0$	$-1\leqslant I_t<0$	$u_{\min}\leqslant u_t<0$	$u_{\min}\leqslant u<0$
$L_t<-L_e$	$I_t<-1$	$u_t<u_{\min}$	$u=u_{\min}$

2 控制算法

2.1 控制器算法

作为核心的控制器结构如图 3所示(图中文字说明的上标表示层序号,下标表示神经元序号或迭代次数),这种控制方法可称为 AMP-DML。

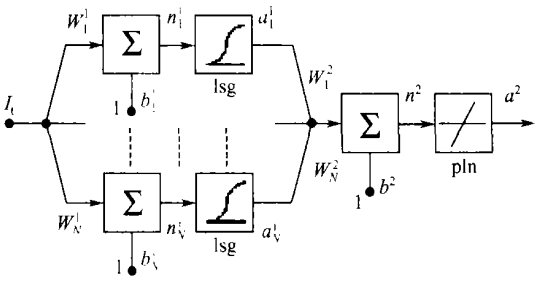

图 3　多层神经网络控制器结构

虽然单层结构收敛较快，却只能模拟线性的目标函数，如果需要速度与缓冲之间是非线性的关系，就必须采用更为复杂的多层神经网络结构来实现[6]。在 AMP-DML控制中的多层网络采用基本的3层前向网络，每层 N 个神经元，这是一种应用广泛的结构，隐层使用了对数-S形激励输出函数 lsg，输出层用的是线性函数 pln。其中各层输出公式为：

$$a_i^1 = \mathrm{lsg}(W_i^1 I_t + b_i^1) = 1\Big/\left[1 + \mathrm{e}^{-(W_i^1 I_t + b_i^1)}\right], \quad (3)$$

$$a^2 = \mathrm{pln}\left(\sum_i W_i^2 a_i^1 + b^2\right) = \sum_i W_i^2 a_i^1 + b^2. \quad (4)$$

当离线学习时，没必要约束播放速度的幅值

$$u = Ca^2 = C\left(\sum_i W_i^2 a_i^1 + b^2\right). \quad (5)$$

由于 3层前向网络可以模拟任意非线性函数，f可以根据需要取多种不同的形式。设 hls为硬极限函数，为了说明不同 f对变速性能的影响，定义 f的函数形式为

$$a_{t,k} = \mathrm{f}(I_{t,k}) = \mathrm{hls}(I_{t,k})U(I_{t,k}/U)^V. \quad (6)$$

这其实是幂函数的一个变形，其中 $U = L_n/L_e$，V的取值会影响曲线的形状，这里考虑 $V \in [0.5, 2]$。

2.2　学习收敛性

这种网络结构的学习算法很多，这里采用了著名的反向传播算法 BP。其原理与最速下降法类似。

$$\text{输出层：}\begin{cases} S_k^2 = -2(a_{t,k}^2 - a_k^2), \\ W_{i,k+1}^2 = W_{i,k}^2 - \eta S_k^2 (a_{i,k}^1)^{\mathrm{T}}, \\ b_{k+1}^2 = b_k^2 - \eta S_k^2, \end{cases} \quad i = 1, \cdots, N.$$

$$\text{隐层：}\begin{cases} S_{i,k}^1 = (1 - a_{i,k}^1) a_{i,k}^1 (W_{i,k}^2)^{\mathrm{T}} S_k^2, \\ W_{i,k+1}^1 = W_{i,k}^1 - \eta S_{i,k}^1 I_{t,k}, \\ b_{i,k+1}^1 = b_{i,k}^1 - \eta S_{i,k}^1, \end{cases} \quad i = 1, \cdots, N. \quad (7)$$

在离线学习的迭代过程中，需要证明网络权值和偏置值在一定条件下将收敛到一个定值。

为了方便，定义输出层权向量 $\boldsymbol{r}$ 和输入向量 $\boldsymbol{z}$：

$$\boldsymbol{r} = [W_1^2, \cdots, W_N^2, b^2]^{\mathrm{T}}, \quad \boldsymbol{z} = [a_1^1, \cdots, a_N^1, 1]^{\mathrm{T}}.$$

由式(4)，有 $a^2 = \boldsymbol{r}^{\mathrm{T}}\boldsymbol{z}$，根据 BP算法有

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \eta S_k^2 \boldsymbol{z}_k = \boldsymbol{r}_k + 2\eta(a_{t,k}^2 - \boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{z}_k)\boldsymbol{z}_k, \quad (8)$$

由于 $\boldsymbol{r}_k^{\mathrm{T}}\boldsymbol{z}_k = \boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{r}_k$，因此式(8)可写作

$$\boldsymbol{r}_{k+1} = (I - 2\eta\boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}})\boldsymbol{r}_k + 2\eta a_{t,k}^2 \boldsymbol{z}_k. \quad (9)$$

这是一个线性动态系统，如果矩阵 $I - 2\eta\boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}}$ 的特征值小于 1，该系统就是稳定的。此时有

$$|1 - 2\eta\lambda_{i,k}| < 1 \Rightarrow 0 < \eta < 1/(2\lambda_{i,k}), \quad (10)$$

令矩阵 $A = \boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}}$，则

$$A^2 = \boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}} = (\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k)(\boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}}) = (\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k)A.$$

可以求得 $\boldsymbol{z}_k\boldsymbol{z}_k^{\mathrm{T}}$ 的最大特征值为

$$\lambda_{\max,k} = \boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k = 1 + (a_{1,k}^1)^2 + \cdots + (a_{N,k}^1)^2. \quad (11)$$

当学习速度满足

$$0 < \eta < 1\Big/\left[2\left(1 + \sum_i (a_{i,\max}^1)^2\right)\right] < 1/(2\lambda_{i,k}) \quad (12)$$

时权向量会收敛，而较大的速度会导致不稳定。

设 $k \geqslant K$ 后输出层权向量将收敛至稳定值 $[W_1^{2*}, \cdots, W_N^{2*}, b^{2*}]^{\mathrm{T}}$，此时 $e_k = a_{t,k}^2 - a_k^2 = 0$，

$$S_k^2 = 0, \quad S_{i,k}^1 = 0, \quad (13)$$

即

$$W_{i,k+1}^1 = W_{i,k}^1 = W_i^{1*}, \quad b_{i,k+1}^1 = b_{i,k}^1 = b_i^{1*}. \quad (14)$$

可以看到，隐层的参数也将收敛至稳定。

多层网络的训练收敛时有一个全局最优，但是在迭代时却可能会陷入一些局部最优值中。选择离收敛值较近的初始值可以减少学习时间，学习速度的合适选择也能优化学习的效率。

3　实验仿真

仿真测试时，缓冲时间的稳定值设为 $L_n = 2.0$ s，初始缓冲时间 $L_c(0) = 1.8$ s，初始播放速度 $u(0) = 0$，周期 $T = 0.1$ s，上下阈值 $L_{\max} = 2.05$ s，$L_{\min} = 1.95$ s，速度变化范围 $C = 0.25$ B/s，$L_e = 0.25$ s，学习速度取满足收敛条件的 $\eta = 0.01$，神经元数 $N = 2$，$V = 0.8$，初始权系数选择较小的随机值。

3.1　离线学习

实际应用之前需要离线训练控制器。缓冲时间 L_c 被限制在 $[0, 2L_n]$之间，并输入一系列随机采样值到控制器。通过相应的学习算法训练使权系数收敛后，控制器能较精确地逼近目标函数。控制器训练 5×10^4 次中止后收敛，均方误差 $E = 0.0378$。BP收

敛较慢会成为训练的瓶颈.

3.2 控制效果

利用训练好后的控制器,可以对处于 $[0, 2L_n]$ 范围内的当前缓冲时间进行控制,输出动态的播放速度来调节缓冲时间至稳定范围 $[L_{min}, L_{max}]$. 这里考虑的网络环境中缓冲还有上升的可能,例如客户端可以实现如重传这样的保护机制,丢失的数据就能重新进入缓冲. 这里用丢包率取 -30%~+30%之间的随机值来模拟这种情况. 图 4比较了这种环境中 AMP-Mean和 AMP-DML的控制输出情况.

在图 4中, AMP-Mean在缓冲低于稳定值时输出最慢速 u_{min}, 超过稳定值输出最快速 u_{max}, 变速时的幅值较大,并且每次变速都是一次较大的突变. 而 AMP-DML在保证了缓冲时间能收敛到 $[L_{min}, L_{max}]$ 之间的同时,还保证了输出速度能线性地随缓冲动态变化. 可以看到,速度的平均幅值减少了,相邻速度的变化也比较均匀. 当丢包率增大时, AMP-DML的变速幅度也会增大,但始终在 $[-C, C]$ 之间.

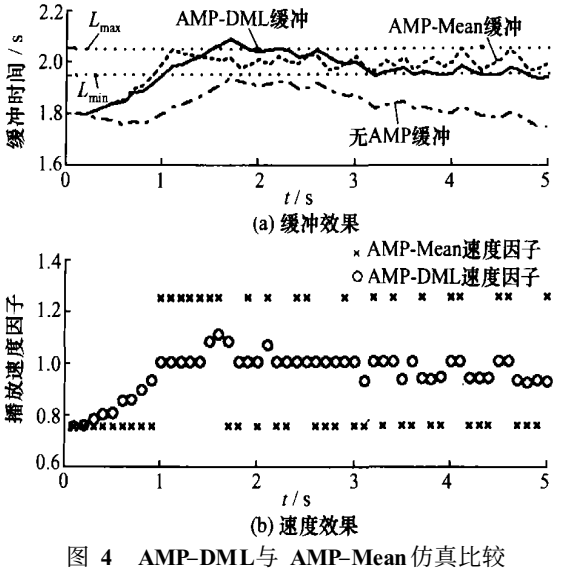

图 4 AMP-DML与 AMP-Mean仿真比较

3.3 变速性能

图 5是 AMP-DML与 AMP-Mean的性能比较.

经过 $M=10^4$ 个周期控制后, AMP-DML的变速性能会随目标函数中的 V值变化. 为了保证平等, AMP-Mean也采用了缓冲的上下限保护.

其中定义速度的平均幅值和平均增量分别为:

$$E[u]=\sum_{m=1}^{M}|u_m|/M,$$

$$E[\Delta u]=\sum_{m=1}^{M}|u_m-u_{m-1}|/M.$$

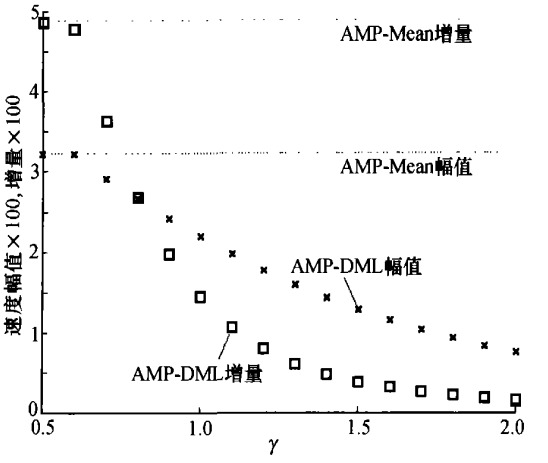

图 5 AMP-DML的速度变化性能

可以看到,随 V的增大,速度的平均幅值和平均增量逐渐减小,性能也更好且始终优于 AMP-Mean.

4 结 论

在神经网络自适应控制流媒体客户端缓冲的方法中,单层和多层的神经网络的引入提高了播放速度的动态变化能力,通过一定的改进策略可以使控制的效果更好. 仿真结果表明: 该方法能产生更加动态均衡的速度,在保证缓冲稳定的同时也减少了由速度变化带来的质量损害,并且可以选择不同的目标函数达到不同的控制效果.

参考文献 (References)

[1] WU Dapeng, HOU Yiwei, ZHU Wenwu, et al. Streaming video over the Internet: approaches and directions [J]. *IEEE Trans on Circuits and Systems for Video Technology*, 2001, **11**(3): 282-300.

[2] Steinbach E G, Farber N, Girod B. Adaptive playout for low latency video streaming [A]. Proc International Conference on Image Processing [C]. Thessaloniki: IEEE Press, 2001. 962-965.

[3] Kalman M, Steinbach E, Girod B. Adaptive media playout for low delay video streaming over error-prone channel [J]. *IEEE Trans on Circuits and Systems for Video Technology, Special Issue on Wireless Video*, 2004, **14**(6): 841-851.

[4] Kalman M, Steinbach E, Girod B. Adaptive playout for real-time media streaming [A]. Proc IEEE International Symposium on Circuits and Systems [C]. Scottsdale, Arizona: IEEE Press, 2002.

[5] 周健, 戴梅萼, 余振建, 等. 远程实时视频传输的自适应技术 [J]. 清华大学学报(自然科学版), 2004, **44**(7): 966-968, 973.
ZHOU Jian, DAI Meie, YU Zhenjian, et al. Self-adaptation technology for video transmission of remote robotically controlled images [J]. *J Tsinghua Univ (Sci & Tech)*, 2004, **44**(7): 966-968, 973. (in Chinese)

[6] Martin T H, Howard B D, Mark H B. Neural Network Design [M]. Boston: PWS Publishing Company, 2002.