文章编号:1007-1180(2011)04-0050-06

语音识别算法的VC++实现

乔 兵,吴庆林*,阴玉梅
(中国科学院 长春光学精密机械与物理研究所,吉林 长春 130033)

摘 要:随着语音识别算法的不断发展,其识别率不断提高,逐步达到可以应用的阶段。本文利用VC++软件实现了一种语音识别算法,并对其识别能力进行了测试。结果表明,该软件实现的算法识别成功率较高,短词可达95%以上,长词可达90%以上;同时识别延迟<50 ms,识别效率高,可满足应用需要。

关键词:语音识别;VC++
中图分类号:TP391 **文献标识码**:A
DOI:10.3788/OMEI20112804.0050

Implementation of Speech Recognition Algorithm in VC++

QIAO Bing, WU Qing-lin, YIN Yu-mei
(*Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun* 130033, *China*)

Abstract: With the continuous development of speech recognition algorithms, the recognition rate is increasing, and can be applied in practice gradually. In this paper, a speech recognition algorithm is implemented by VC++ and its recognition ability is tested. The results show that the algorithm has high recognition success rate, up to 95% in short term, and up to 90% in long term. The algorithm also has low recognition delay less than 50 ms, which can meet the application needs.

Keywords: speech recognition; VC++

1 引 言

语音识别[1-2]是一门交叉学科。近20年来，语音识别技术取得了显著进步，开始从实验室走向市场。人们预计，未来10年内，语音识别技术将进入工业、家电、通信、汽车电子、医疗、家庭服务、消费电子产品等各个领域。语音识别听写机在一些领域的应用被美国新闻界评为1997年计算机发展十件大事之一。很多专家都认为，语音识别技术是2000年至2010年间信息技术领域十大重要科技发展之一。语音识别技术主要包括特征提取技术、模式匹配准则及模型训练技术3个方面，所涉及的领域包括：信号处理、模式识别、概率论、信息论、发声机理、听觉机理、人工智能和语言控制等。

本文利用VC++实现了一种语音识别算法，并对其识别能力进行了分析。

2 解决思路

本文将利用VC++编程[3-4]实现语音识别算法，整个程序大致可划分为3个模块：预处理、特征值提取、匹配识别，其结构框图如图1所示。

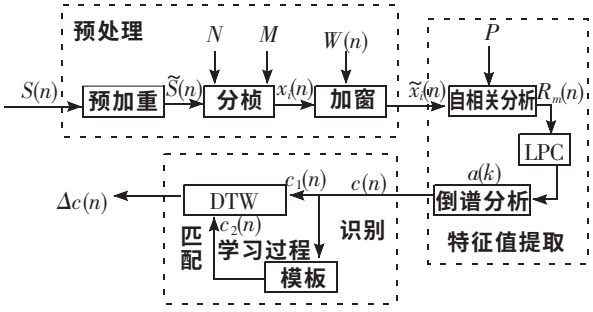

图1 语音识别算法结构框图

3 模块实现

3.1 声音采集

我们利用声卡录音的低层操作技术将声音信号送入计算机，即对winmm.lib进行API调用，编程时这一部分被写在一个类中（Soundin类）。

在构造函数中设定包括最大采样率（11 025 Hz）、数据缓存（作为程序一次性读入的数据，2 048）以及声卡本身所带的一些影响采样数据的各种参数。

调用API函数waveInGetNumDevs检查并打开声音输入设备，即声卡；使用waveInGetDevCaps得到声卡的容量；创建一个叫WaveInThreadEvent的事件对象，并赋予一个Handle，称作m_WaveInEvent，开始利用线程指针m_WaveInThread调用自定义的线程WaveInThreadProc；对结构WAVEFORMATEX中的WaveInOpen开始提供录音设备。注意设备句柄是通过对HWAVEIN型数据m_WaveIn的引用得到的。

由于通过这种方式进行录音的文件格式是.wav，所以要先设置录音长度，以及对头文件进行一些设置：包括buffer的地址（InputBuffer的初始地址）、大小（录音长度的两倍）和类型。使用waveInPrepareHeader为录音设备准备buffer，然后使用waveInAddBuffer函数为录音设备送出一个输入buffer，最后使用waveInStart (m_WaveIn)打开设备。

程序中WaveInThreadProc需要特别说明，因为通过这个线程可以实现采样和数据提取。该线程首先定义一个指向CsoundIn类的指针pParam，并将其宏定义为PT_S；而线程参数即为空指针pParam。使用WaitForSingleObject将录音过程设置为一旦开始就不中止（除非中止线程）。在此线程中做如下两个工作：将数据送入buffer，并将数据传入某个参数（其调用一个函数，将buffer中的数据送入该函数的参数*pt），而这些数据正是我们要利用和处理的数字化的语音信息。

3.2 声音预处理

声音信息的预处理主要包括音头和音尾的判断，声音的预加重，分帧处理和窗化处理[5]。

3.2.1 音头音尾的判断与提取

由于输入的声音信号不是连续的，音头音尾的判断是算法的一个难点；还有噪声的影响，必须通过适当的方法来判断采集的数据是否为所要的声音控制信号，这是该项目的另一个难点。若声音指令信号提取得不恰当，那么采样所得的数据就会与实

际的语音信号有很大的出入，这样不但会延迟语音识别的时效性，而且会降低对这些声音信号的识别率。对声音信号的提取主要是确定音头、音尾的位置，常用的方法有过零率和短时距能量等。我们采用的是过零率法，首先对噪声取样，从这些噪声样本中得到噪声的上下限，将实时信号与这个门限进行比较，得到过零率。定义过零率 Zcr 如下：

$$Zcr=\sum_{m=-\infty}^{\infty}Sgn[x(m)]-Sgn[x(m-1)] \quad (1)$$

其中：

$Sgn[x(n)]=1$ $x(n)>$NoiseMax (NoiseMax 为噪声上限)

$Sgn[x(n)]=-1$ $x(n)<$NoiseMin (NoiseMin 为噪声下限)

$Sgn[x(n)]=0$ otherwise

利用过零率的大小来判断是否有声音信号进入，若 $Zcr>Zcr\_gate$ (预设的过零率值)，则表示有声音信号进入，即找到了音头。在找到音头的情况下，若 $Zcr<zcr\_gate$，则表示声音结束，也就找到了音尾。在环境噪声较大且比声音指令小得多的情况下可以对这个门限加以修正。音头和音尾之间的部分就是我们用以作为识别用的声音指令信号。由于一般情况下人们所发出的单音都有一定的时间长度，而大的噪声则大多是突发的，持续时间较短，所以我们可以再对所得到的声音指令信号做一次筛选，若得到的声音信号的长度小于预设值，就可认为是噪声干扰而舍弃；若得到的声音信号的长度大于预设值，则将其作为有用信号存储。实验表明，利用过零率和预设长度相结合提取声音指令信号的方法很有效。

3.2.2 语音信号的预加重

本算法所采用的预加重方法是较为常用的网络，传递函数为：

$$H(z)=1-0.937\,5z^{-1} \quad (2)$$

得到的信号为：

$$\tilde{S}(n)=S(n)-0.937\,5S(n-1) \quad (3)$$

预加重的目的在于滤除低频干扰，尤其是 50 Hz 或 60 Hz 的工频干扰，将对语音识别更为有用的高频频谱进一步提升。在计算短时能量之前应用该滤波器，还可以起到消除直流漂移、抑制随机噪声和提升清音部分能量的效果。

3.2.3 分帧处理

在计算各个系数之前，要先将语音信号作分帧处理。语音信号是瞬时变化的，但在 10~20 ms 内是相对稳定的，而设定的采样频率为 11 025 Hz，所以我们对预处理后的语音信号 $S_1(n)$ 以 300 点为一帧进行处理，帧移为 100 个采样点。

$$x_l(n)=\tilde{s}(Ml+n) \quad (4)$$

$n=0,1,\cdots,N-1$，$l=0,1,\cdots,L-1$ ($N=300$)

3.2.4 窗化处理

为了避免矩形窗化时对 LPC 系数在端点产生误差，我们采用了汉明窗函数来进行窗化。即：

$$\tilde{x}_l(n)=x_l(n)w(n) \quad 0\leqslant n\leqslant N-1 \quad (5)$$

其中 $w(n)=0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right)$ $0\leqslant n\leqslant N-1$ (6)

3.3 语音数据的特征提取

语音信号的特征有多种度量标准，我们采用的是比较常用的倒谱特征[6]。

语音信号是一种典型的时变信号，然而，如果把观察时间缩短到几十毫秒，则可以得到一系列近似稳定的信号。人的发音器官可以用若干段前后连接的声管进行模拟，这就是所谓的声管模型。全极点线性预测模型[7]（LPC）可以对声管模型进行很好的描述，每段声管对应一个 LPC 模型的极点。一般情况下，极点的个数在 12~16 个之间即可足够清晰地描述语音信号的特征。

语音信号经过预处理，它的每个样值均可由过去若干个样值的线性组合来逼近，同时可以采用使实际语音抽样与线性预测抽样之间的均方差最小的方式来解出一组预测的系数 a，这就是 LPC 提取出来的信号的初始特征。

预测值时域表达式为：

$$\tilde{S}(n)=\sum_{i=1}^{p}a_iS(n-i) \tag{7}$$

其中，a_i 为加权系数，即 LPC 系数。

预测误差为：

$$e(n)=S(n)-\tilde{S}(n)=S(n)-\sum_{i=1}^{p}a_iS(n-i) \tag{8}$$

使 $e(n)$ 在均方误差最小的条件下，可求得唯一的 a_i，此过程即为 LPC 的分析过程。

这里采用的是 Levinson-Durbin 法，由（8）式得：

$$R(j)-\sum_{i=1}^{p}a_iR(j-i)=0 \tag{9}$$

$$R(0)-\sum_{i=1}^{p}a_iR(i)=Ep \tag{10}$$

其中，$R(j)$ 为待分析与引信号的自相关序列：

$$R(j)=\sum_{n=i}^{N-1}S(n)S(n-j)=\sum_{n=0}^{N-j-1}S(n)S(n+j) \tag{11}$$

Levinson-Durbin 算法为：

（1）初始化：$E_0=R(0)$

（2）迭代计算：对于 $1\leqslant j\leqslant p$

$$k_i=\left[R(i)-\sum_{j=1}^{i-1}a_j^{(i-1)}R(i-j)\right]/E^{(i-j)}$$

$$a_j^{(i)}=\begin{cases}k_j & j=i\\ a_j^{(i-1)}-k_ia_{j-1}^{(i-1)} & 1\leqslant j\leqslant i-1\end{cases}$$

$$E^{(i)}=(i-k_i^2)E^{(i-1)}$$

（3）最后计算

$$a_i=a_j^{(p)} \qquad 1\leqslant j\leqslant p \qquad G^2=E^{(p)}$$

以上式中的 k_i 为反射系数，$|k_i|<1$, $1\leqslant i\leqslant p$；$E^{(p)}$ 为最小预测误差，随着阶数的增加而减少；G 为模型增益常量。

在语音识别系统中，很少直接使用 LPC 系数，而是由 LPC 系数推导出另一种参数——线性预测倒谱系数（LPCC）。倒谱实际上是一种同态信号处理方法，标准的倒谱系数计算流程需要进行 FFT 变换、对数操作和相位校正等步骤，运算比较复杂。在实际计算中，它不是由原始信号 $x(n)$ 得到的，而是由 LPC 系数 a_i 得到的。

计算出 LPC 系数后，就可以直接进行倒谱系数 $C(k)$ 的计算，其迭代算法如下：

（1）初始化：$C(0)=\log G^2$

（2）迭代计算：

$$\begin{cases}C(m)=a_m+\sum_{k=1}^{m-1}\frac{k}{m}a_{m-k}C(k) & 1\leqslant m\leqslant p\\ C(m)=\sum_{k=1}^{m-1}\frac{k}{m}a_{m-k}C(k) & m>p\end{cases}$$

这里 $C(0)$ 实际上就是直流分量，在识别中通常是不需要的，也不必计算。综合考虑识别误差和识别速度的影响，我们在计算 LPC 时，LPC 系数的阶数取为 8，而 LPCC 系数的阶数取为 12。

3.4 DTW 算法

语音识别程序的核心部分即采用合适的算法来识别不同的语音信号，在特定人语音识别算法中，对于孤立词语语音识别而言，最为简单的方法是采用 DTW（Dynamic time warping，动态时间弯折）算法[8]，该算法基于动态规划（DP）的思想，解决了发音长短不一的模本匹配问题，是语音识别中出现较早、较为经典的一种算法。本文采用的就是 DTW 算法。

用 R 表示已存在的参考模板，T 表示待识别的测试模板，$R(1)$, $R(2)$, …, $R(m)$, $T(1)$, $T(2)$, …, $T(n)$ 分别表示参考模板和测试模板中的各语音帧，$d[T(n),R(m)]$ 表示这两帧特征矢量之间的距离（DTW 算法中通常采用欧氏距离）。为了比较 R 和 T 之间的相似度，可以计算他们之间的距离 $D(T, R)$，距离越小则表明相似度越高。$D(T, R)$ 的计算通常采用的是动态规划的方法。

将 R 和 T 的各个帧号分别在直角坐标系的横轴和纵轴上标出，可得到如图 2 的一个网格，网格中各点表示 R 和 T 中一帧的交汇点。DP 算法可以归结为寻找一条通过此网格中若干格点的路径，使得沿路径的累积距离达到最小值。

为了使路径不至于过分倾斜，可以约束斜率在 0.5~2 范围内，如果路径已经通过了格点 (n_{i-1},m_{i-1})，那

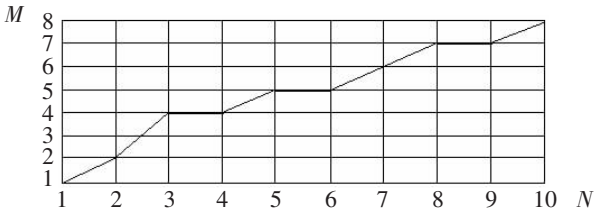

图 2 DTW 算法搜索路径

么下一个通过的格点 (n_i,m_i) 只能是下列 3 种情况之一：

$$(n_i,m_i)=(n_{i-1}+1,m_{i-1}+2)$$
$$(n_i,m_i)=(n_{i-1}+1,m_{i-1}+1) \quad (12)$$
$$(n_i,m_i)=(n_{i-1}+1,m_{i-1})$$

搜索最佳路径的方法如下：搜索从 (n_1,m_1)开始，网格中任意一点只可能有一条路径通过。对于 (n_i,m_i)，可达到该格点的前一格点只可能是 (n_i-1,m_i) 、(n_i-1,m_i-1)或 (n_i-1,m_i-2)，那么 (n_i,m_i)选择这 3 个距离中的最小者所对应的格点作为其前续格点。若用 (n_{i-1},m_{i-1})代表此格点，并将通过该格点的路径延伸而通过 (n_i,m_i)，此路径的累积距离为：

$$D[(n_i,m_i)]=d[T(n_i),R(m_i)]+D[(n_{i-1},m_{i-1})] \quad (13)$$

其中的 $n_{i-1}=n_i-1,m_{i-1}$ 由下式决定：

$$D[(n_{i-1},m_{i-1})]=$$
$$\min\{D[(n_{i-1},m_i)],D[(n_{i-1},m_{i-1})],D[(n_{i-1},m_{i-2})]\} \quad (14)$$

这样可以从初始点出发依次搜索直至搜索到终点 (n_N,m_M)，便可得到最佳路径。

3.5 软件设计流程图

本软件的设计流程如图 3 所示，主要包括语音采集、预处理、特征提取和识别几个过程。

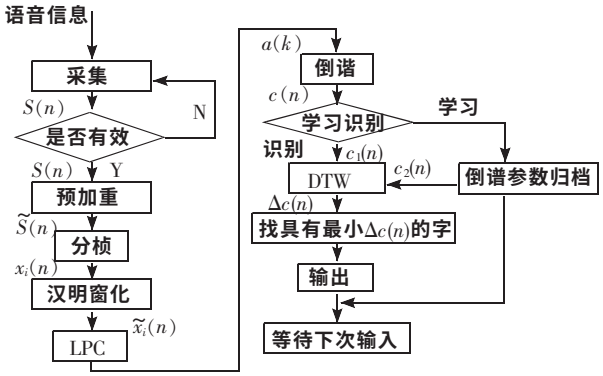

图 3 语音识别软件流程图

4 结论分析

根据语音识别针对不同样本具有个体差异的特点，我们随机选择了 5 名测试者对软件的识别能力进行了测试，测试结果如表 1 所示。

表 1 语音识别软件识别效果统计表

测试者	性别	短词识别率/%	长词识别率/%
A	女	98	93
B	男	97	92
C	女	97	91
D	男	95	90
E	男	98	91

其中，短词是指 1~2 个字组成的词语，长词为 3 个字以上的词语。从表 1 可以看出，该软件的识别率较高，短词语达到 95%以上，较长的词语也能达到 90%以上；同时，识别的延迟时间<50 ms，识别成功率和效率都较高，能够达到简单应用的程度。不过，该软件暂时只能实现对词语的准确识别，对于整句的识别将有待于我们对算法的进一步改进，这也将是我们今后的研究重点。

参考文献

[1] 张雄伟. 现代语音处理技术及应用[M]. 北京：机械工业出版社，2003.
[2] 赵力. 语音信号处理[M] . 北京：机械工业出版社，2009.
[3] 孙鑫，余安萍. VC++深入详解[M]. 北京：电子工业出版社，2006.
[4] 宋金珂，高丽华，张迎新. VC++程序设计基础教程 [M]. 北京：清华大学出版社，2010.
[5] 顾亚强. 非特定人语音识别关键技术研究[D]. 长沙：国防科学技术大学硕士学位论文，2009.

[6] 时晓东. 孤立词语音识别系统设计研究[D]. 杭州：浙江大学硕士学位论文，2006.
[7] 刘承玺. 语音识别算法的确定与实现[J]. 天津通信技术，1995，8(1)：22-25.
[8] 万春. 基于 DTW 的语音识别应用系统研究与实现[J]. 集美大学学报(自然科学版)，2002，10(2)：33-37.

作者简介：乔兵（1964-），女，山东定陶人，学士，高级工程师，主要从事电子设计研究。
吴庆林*（1980-），男，黑龙江嫩江人，博士，助理研究员，2009年于中国科学技术大学获得博士学位，主要从事机电控制、伺服控制的相关研究。E-mail：wuql@ciomp.ac.cn