文章编号:1671-251X(2013)08-0016-06　　DOI:
樊炳辉,周凯,纪鹏. 特定人语音识别算法的 VC++实现[J]. 工矿自动化,2013,39(8):16-21.

特定人语音识别算法的 VC++实现

樊炳辉,　周凯,　纪鹏

(山东科技大学 信息与电气工程学院,山东 青岛　266590)

摘要:针对现有的非特定人语音识别系统存在词汇数据库庞大、训练过程复杂的问题,采用 VC++实现了一种特定人语音识别算法。该算法采用先预加重后端点检测的方法来消除低频噪声的影响;选择短时能量和短时过零率 2 个指标作为语音信号端点检测的依据;加入了可容忍静音时间的判断来保证检测到的语音数据的完整性;采用改进的动态时间规整算法进行模式匹配,在不影响计算结果的前提下减少了运算量。测试结果表明,该算法能够对短词和短句进行实时、准确识别,并具有较快的识别速度。

关键词:语音识别;端点检测;短时能量;过零率;梅尔频率倒谱系数;动态时间规整;VC++
中图分类号:TD67　　文献标志码:A　　网络出版时间:
网络出版地址:

Implementation of speaker-depended speech recognition algorithm in VC++

FAN Bing-hui,　ZHOU Kai,　JI Peng
(College of Information and Electrical Engineering , Shandong University of Science and Technology, Qingdao 266590, China)

Abstract: In view of problems of huge words database and complex training process existed in current speaker-independent speech recognition system, a speaker-depended speech recognition algorithm was implemented by using VC++. The algorithm uses method of pre-emphasis first then endpoint detection to eliminate influence of low-frequency noise; selects two indexes of short-time energy and short-time zero-crossing rate as criterion of endpoint detection for speech signal; adds judgment of tolerable mute time to ensure integrity of speech data; uses improved dynamic time warping algorithm to make pattern matching which reduces computation under premise of not influencing calculating results. The test result shows that the algorithm can recognize short words and short sentences real-timely and accurately with faster recognition speed.

Key words: speech recognition; endpoint detection; short-time energy; zero-crossing rate; mel-frequency cepstrum coefficient; dynamic time warping; VC++

0　引言

语音识别是一门交叉学科,涉及生理学、语言学、信号处理、人工智能等诸多领域。作为智能计算机研究的主导方向,语音识别技术在工业、交通、军事、医学、民用等方面都有着广泛应用[1]。将来,语音识别技术有可能作为一种重要的人机交互手段,辅助甚至取代传统的键盘、鼠标等输入设备,在个人计算机上进行文字录入和操作控制[2]。因此,开发出一个能够在普通 PC 机上运行的、小型实用的语音识别系统具有重要意义。现有的非特定人语音识别系统词汇数据库庞大、训练过程复杂,笔者采用

收稿日期:2013-05-14。
基金项目:青岛市科技计划基础研究项目(12-1-4-6-(5)-jch);青岛经济技术开发区重点科技发展计划项目(2011-2-49);博士点基金资助项目(20093718110007)。
作者简介:樊炳辉(1958-),男,山东聊城人,教授,博士研究生导师,研究方向为机器人技术,E-mail:fanbh58@163.com。通讯作者:周凯。

VC++设计出一种数据库小、识别速度快、简单轻巧的特定人语音识别系统,详细阐述了特定人语音识别算法在 VC++中的实现过程,并对算法的识别能力进行了测试和分析。

1　算法总体思路

特定人语音识别算法采用 VC++编程实现,其总体结构可以分为采样、预处理、提取特征参数和模式匹配 4 个部分,如图 1 所示。

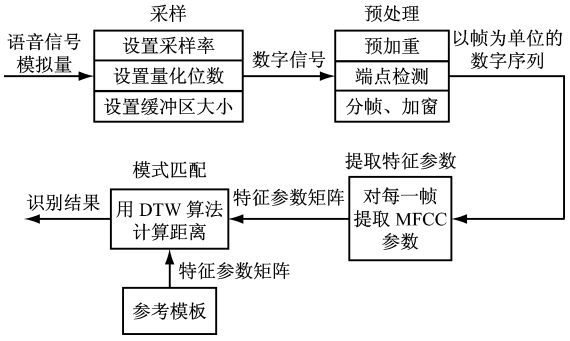


图 1　基于 VC++的特定人语音识别算法总体结构

2　算法功能模块实现

2.1　采样

首先在 VC++工程中引入“mmsystem.h”头文件和“winmm.lib”库,以便进行与音频相关的操作。定义一个 WAVEFORMATEX 结构体变量,利用该变量将采样频率设置为 8 000 Hz,量化位数设置为 16 位。然后通过“winmm.lib”库提供的 waveInOpen 函数打开音频输入设备。

由于端点检测算法的计算单位就是 1 个缓冲区的大小,故缓冲区越小,检测出的有效语音段的长度就越准确,而缓冲区太小,又会加重程序的运行负担。综合考虑,将缓冲区的大小配置为 256 B,即 1 个缓冲区可装入 128 个采样点的值,该配置通过初始化 WAVEHDR 结构体变量实现。调用 waveInPrepareHeader 和 waveInAddBuffer 函数将输入缓冲区添加至录音设备。最后调用 waveInStart 函数开始音频采集。

当 1 个输入缓冲区被填满之后,操作系统会产生 WIM_DATA 消息。在 WIM_DATA 消息处理函数[3-4]中读取缓冲区中的数据,再次为设备添加缓冲区,并对数据进行预处理。

2.2　预处理

对声音信号的预处理包括 4 部分内容:预加重、端点检测、分帧和加窗[1]。为了保证语音识别的时效性,本算法采取的策略是先对数据进行预加重和端点检测,当获得语音段后,再对其分帧和加窗。预处理程序流程如图 2 所示。

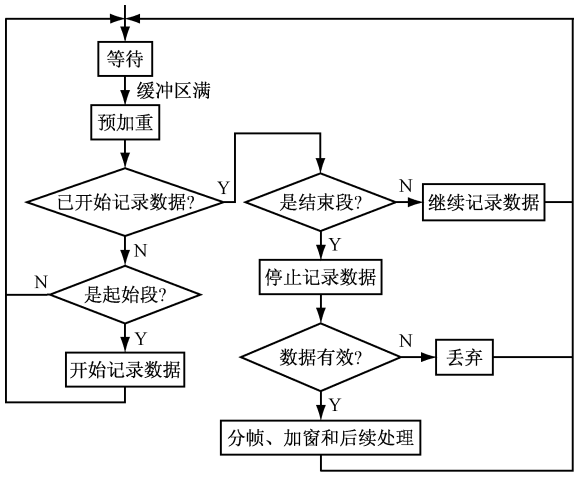


图 2　预处理程序流程

2.2.1　预加重

预加重的作用有 2 个:一是对能反映语音特征的高频部分进行加强;二是滤除低频干扰,尤其是 50～60 Hz 的工频干扰[2]。预加重后再检测端点可消除低频噪声对端点检测过程的影响,提高检测的准确性[5]。

预加重的方法是把从缓冲区中读出的数据通过一个一阶高通滤波器,其系统函数为 $1-0.937\ 5z^{-1}$。设 $x_0(n)(0\leqslant n<128)$ 为刚从缓冲区中读出的数据,$x_1(n)$ 为预加重后的数据,则预加重的实现过程可由式(1)描述:

$$x_1(n)=\begin{cases}x_0(0)-0.937\ 5x_0(-1), n=0\\ x_0(n)-0.937\ 5x_0(n-1), 1\leqslant n<128\end{cases} \tag{1}$$

式中:$x_0(-1)$ 为上一次读取的缓冲区中的最后一个采样值,这样可以保证滤波的连续性。对于第 1 次读取的数据,$x_0(-1)$ 取 0。

2.2.2　端点检测

端点检测就是确定语音段的起始点和结束点,常用方法有短时能量法和短时过零率法[6]。本算法采用短时能量和短时过零率 2 个特征确定语音端点。

语音和噪声的主要区别是能量大小不同,一般来说,语音段的能量比噪声段大[7]。对于预加重后的数据 $x_1(n)$,短时能量的计算方法:

$$E=\sum_{n=0}^{N-1}|x_1(n)| \tag{2}$$

式中:$N=128$。

短时过零率定义为一段信号中波形穿越零电平的次数[2],它对清音的检测效果很明显[7]。在实际应用中,为了避免静音段的随机噪声产生过高的过

零率，需要先设置一个门限 T_H，当前后 2 个采样值的符号不同，而且差值大于该门限时，再将短时过零率加 1。对于预加重后的数据 $x_1(n)$，短时过零率的计算方法：

$$Z=\sum_{n=0}^{N-1}\frac{1}{2}\left[\,|\mathrm{sgn}(x_1(n))-\mathrm{sgn}(x_1(n-1))|\,\right]\times u\left[\,|x_1(n)-x_1(n-1)|-T_H\right] \tag{3}$$

其中：

$$\mathrm{sgn}(x)=\begin{cases}1, & x\geqslant 0\\ -1, & x<0\end{cases} \tag{4}$$

$$u(x)=\begin{cases}1, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{5}$$

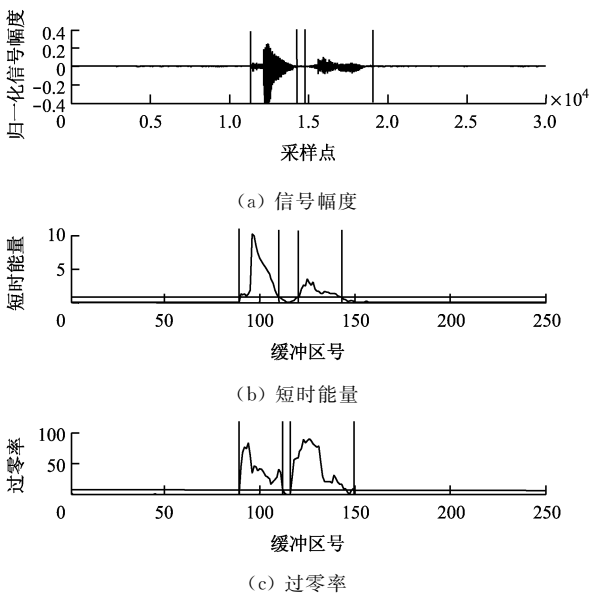
分别对语音信号和静音段的噪声信号进行取样，计算并分析它们的短时能量和过零率，从而选取合适的短时能量阈值 E_{gate} 和短时过零率阈值 Z_{gate}。

每当从缓冲区中读出一次数据并经过预加重处理后，就计算一次短时能量和过零率，在程序中判断，只要两者中有一个超过相应阈值，就将此段数据作为语音的起始段，从此处开始记录数据。此后计算过程中若短时能量和过零率都回落到阈值以下，就将此时的数据段作为语音的结束段，停止记录数据。

由于人用自然语速说话时，字与字之间有一定时间长度的间隔，而在这段间隔内短时能量和短时过零率有可能会回落到阈值以下，这样程序会误认为此处是语音的结束段，从而导致检测到的有效语音数据不完整。图 3 为以较慢的语速说出的“开始”信号经预加重之后的信号幅度和相应的端点检测结果，图中纵坐标短时能量无单位，过零率为整数，图 4 与其相同。

从图 3 可看出，对于一个完整的词“开始”，算法将它误检测为 2 个词“开”和“始”，这样会造成后面的语音匹配错误。为避免这种情况的出现，引入了可容忍静音时间[8] $t_{between}$，并将 $t_{between}$ 设置为 15 个缓冲区数据段长度(0.24 s)。在记录的过程中，当短时能量和短时过零率回落到阈值以下之后，若在 $t_{between}$ 时间内没有短时能量或过零率超过阈值，则认为语音结束，停止记录，否则继续记录。引入可容忍静音时间后的“开始”语音信号幅度和端点检测结果如图 4 所示。

一些突发性的噪声也可引起短时能量或过零率的数值很高，但是往往不能维持足够的时间。为了消除这种噪声的影响，程序中设置了最短时间门限

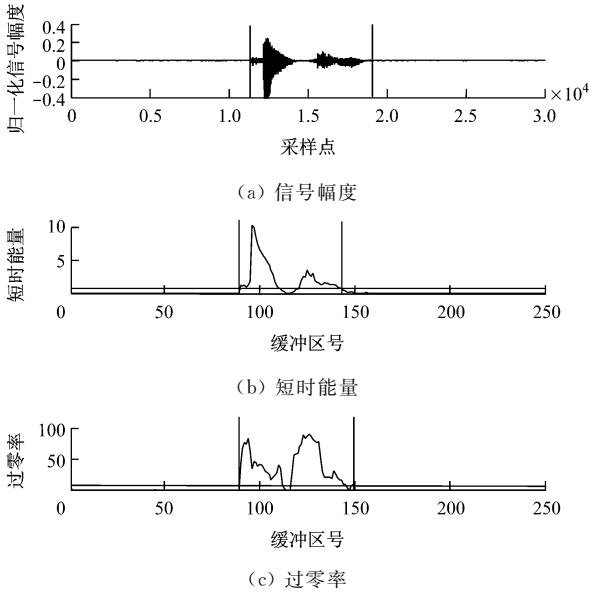


(a) 信号幅度

(b) 短时能量

(c) 过零率

图 3 “开始”语音信号预加重后的幅度和相应的端点检测结果



(a) 信号幅度

(b) 短时能量

(c) 过零率

图 4 引入可容忍静音时间后的“开始”语音信号幅度和端点检测结果

t_{min}。当语音段数据记录完成后，判断其长度是否大于 t_{min}，若大于 t_{min}，则认为是有效数据，否则认为是噪声，将其丢弃。

2.2.3 加窗和分帧

通过端点检测获得语音段后，以 256 个采样点为帧长，100 个采样点为帧移对语音段数据分帧。然后对每一帧进行加窗操作，以减少由截断处理导致的 Gibbs 效应[9]。加窗的方法是让每一帧通过一个汉明窗：

$$\widetilde{x}_m(n)=x_m(n)W_{HM}(n) \tag{6}$$

$$W_{HM}(n)=0.54-0.46\cos\left(\frac{2\pi n}{N-1}\right) \tag{7}$$

式中:m 为帧号,$0 \leqslant m \leqslant n$;$n$ 为帧数,$0 \leqslant n \leqslant N$;$N$ 为帧长,这里 $N=256$;$W_{HM}(n)$为汉明窗。

2.3 提取特征参数

语音识别中常用的参数有 LPCC(Linear Prediction Cepstrum Coefficient,线性预测倒谱系数)[10] 和 MFCC(Mel-Frequency Cepstrum Coefficient,梅尔频率倒谱系数)[11-13]。其中 LPCC 是基于人的发音模型建立的,是一种基于合成的参数。而 MFCC 是基于人耳的听觉特性提出的。大量研究表明,MFCC 能够比 LPCC 更好地提高系统的识别性能[2]。故本算法选用 MFCC 作为特征参数。MFCC 是按帧计算的,其计算步骤如下:

(1) 对加窗之后的每一帧数据 $\widetilde{x}_m(n)$计算 FFT(Fast Fourier Transform,快速傅里叶变换),取模的平方得到离散功率谱 $S(i)(0 \leqslant i < 256)$。本算法采用基二的时域抽取法 FFT 来计算 $S(i)$。

(2) 在梅尔尺度频域内设计 K 个滤波器(这里 K 取 24),并将它们映射到线性频域内。梅尔频率和线性频率 f 的转换关系:

$$f_{mel} = \mathrm{Mel}(f) = 2\,595\lg\left(1+\frac{f}{700}\right) \tag{8}$$

24 个滤波器的形状都为等腰三角形,它们在梅尔频率轴上均匀分布且相互重叠。第 n 个滤波器的起始频率等于第$(n-1)$个滤波器的中心频率。本算法中每个滤波器的中心频率:

$$f_{center}(k) = \mathrm{Mel}^{-1}\left[\frac{\mathrm{Mel}(f_s/2)}{K+1}(k+1)\right] \tag{9}$$

式中:$0 \leqslant k < K$;f_s 为采样频率,$f_s=8\,000$ Hz。

得到中心频率后,还要计算每个中心频率对应的离散频率点的值 f_{center_k}:

$$f_{center_k} = \mathrm{round}\left(\frac{f_{center}(k)}{f_s}N\right) \tag{10}$$

式中:round(·)为四舍五入法取整函数。

第 k 个滤波器在线性频域内的表达式:

$$\mathrm{filter}_k(i) = \begin{cases} \dfrac{i-f_{center_{k-1}}+1}{f_{center_k}-f_{center_{k-1}}+1}, & f_{center_{k-1}} \leqslant i \leqslant f_{center_k} \\ 1-\dfrac{i-f_{center_k}}{f_{center_{k+1}}-f_{center_k}+1}, & f_{center_k} < i \leqslant f_{center_{k+1}} \\ 0, & 0 \leqslant i < f_{center_{k-1}} \text{ 或 } f_{center_{k+1}} < i < 128 \end{cases} \tag{11}$$

式中:i 为频率采样点;$k=0$ 时的 $f_{center_{k-1}}$ 被初始化为 0,$k=K-1$ 时的 $f_{center_{k+1}}$ 被初始化为 128。

(3) 舍弃代表低频成分的 filter_0。分别计算 $S(i)$和 $\mathrm{filter}_k(i)$在各离散频率点上的乘积之和,得到$(K-1)$个参数 P_k(P_k 为数字量):

$$P_k = \sum_{i=0}^{127} S(i)\mathrm{filter}_k(i) \tag{12}$$

(4) 计算 P_k 的自然对数,得到 L_k:

$$L_k = \ln P_k \tag{13}$$

(5) 对 L_k 做离散余弦变换[14],得到 D_j:

$$D_j = \sum_{k=1}^{K-1} L_k \cos\left[\frac{\pi j}{K-1}(k-0.5)\right] \tag{14}$$

式中:$1 \leqslant j \leqslant 12$。

将 D_j 作为对应帧的 MFCC 特征参数。

2.4 模式匹配

通过计算得到语音段所有帧的 MFCC 特征参数后,就形成了一个 M 行 12 列的参数矩阵(M 表示语音段的总帧数,12 表示每一帧的 MFCC 特征参数的个数),在程序中进行判断,如果当前处在训练过程中,则将此矩阵作为参考模式存入参考模式库中;如果处在识别过程中,就将此矩阵作为待识别模式,计算它与模式库中的各个参考模式之间的距离,从而获得识别结果。

在特定人孤立词语音识别中,进行模式匹配最为简单有效的方法是 DTW(Dynamic Time Warping,动态时间规整)算法。该算法基于动态规划的思想,解决了发音长短不一的模板匹配问题,是语音识别中出现较早、较为经典的一种算法[3]。本文采用一种改进的 DTW 算法进行模式匹配。

一个待识别模式可表示为 $\boldsymbol{T}=\{\boldsymbol{T}(1),\boldsymbol{T}(2),\cdots,\boldsymbol{T}(n),\cdots,\boldsymbol{T}(N)\}$,其中 n 为待识别语音的帧号,$\boldsymbol{T}(n)$为第 n 帧的 MFCC 特征参数矢量,N 为待识别模式所包含的语音帧总数。一个参考模式可表示为 $\boldsymbol{R}=\{\boldsymbol{R}(1),\boldsymbol{R}(2),\cdots,\boldsymbol{R}(m),\cdots,\boldsymbol{R}(M)\}$,其中 m 为训练语音的帧号,M 为参考模式所包含的语音帧总数。计算待识别模式和参考模式间的距离 $D[\boldsymbol{T},\boldsymbol{R}]$,要从 $\boldsymbol{T}$ 和 $\boldsymbol{R}$ 中各对应帧之间的距离算起[15]。设 n 和 m 分别为 $\boldsymbol{T}$ 和 $\boldsymbol{R}$ 中任意选择的帧号,$D[\boldsymbol{T}(n),\boldsymbol{R}(m)]$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)。

改进的 DTW 算法原理如图 5 所示。待识别模式的各个帧号 $n(1\sim N)$在二维直角坐标系中的横轴上标出,参考模式的各帧号 $m(1\sim M)$在纵轴上标出,网格中的每一个交叉点(n,m)表示待识别模式中第 n 帧与参考模式中第 m 帧的交汇点。DTW 算法可归结为寻找一条通过网格中若干格点的路径,路径通过的格点即为参考模式和待识别模式中进行

距离计算的帧号。

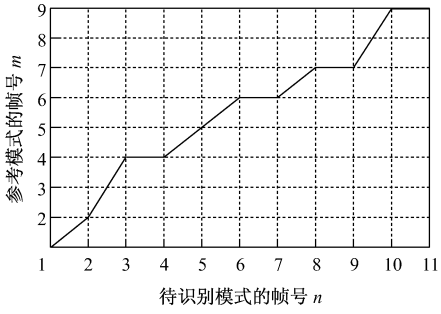


图 5 改进的 DTW 算法原理

路径不是随意选择的,任何一种语音的发音快慢有可能变化,但是其各部分的先后顺序不可能改变,因此,所选的路径必定是从左下角出发,在右上角结束。

搜索路径可由式(15)所示的函数表示:

$$m_i = \Phi(n_i) \tag{15}$$

式中:$n_i = i, i = 1, 2, \cdots, N$;$\Phi(1) = 1, \Phi(N) = M$。

为了使路径不至于过分倾斜,这里约定如果路径已通过格点(n_{i-1}, m_{i-1}),那么下一个通过格点(n_i, m_i)只可能是式(16)所列的 3 种情况中的 1 种:

$$\begin{cases} (n_i, m_i) = (n_{i-1} + 1, m_{i-1} + 2) \\ (n_i, m_i) = (n_{i-1} + 1, m_{i-1} + 1) \\ (n_i, m_i) = (n_{i-1} + 1, m_{i-1}) \end{cases} \tag{16}$$

也就是说,对于(n_i, m_i),其前一个格点只可能是$(n_i - 1, m_i)$、$(n_i - 1, m_i - 1)$和$(n_i - 1, m_i - 2)$,这 3 个中拥有最小积累距离的格点将作为(n_i, m_i)前续格点,若用(n_{i-1}, m_{i-1})代表该格点,并将通过该格点的路径延伸至通过(n_i, m_i),这时该路径的积累距离为

$$D[n_i, m_i] = D[\boldsymbol{T}(n_i), \boldsymbol{R}(m_i)] + D[n_{i-1}, m_{i-1}] \tag{17}$$

式中:$n_{i-1} = n_i - 1$;m_{i-1}由式(18)决定:

$$D[n_{i-1}, m_{i-1}] = \min(D[n_{i-1}, m_i], D[n_{i-1}, m_i - 1], D[n_{i-1}, m_i - 2]) \tag{18}$$

这样可以从$(n_1, m_1) = (1, 1)$出发,逐一计算每一个格点的积累距离,每个格点对应唯一的一条搜索路径。当计算到(n_N, m_N)时,只保留一条最佳路径,而在(n_N, m_N)格点的积累距离就是利用 DTW 算法得到的参考模式和待识别模式之间的距离。

由于式(16)的限制,并不是所有格点的帧匹配距离和积累距离都需要计算。因为路径是从(1,1)开始到(N, M)结束,而路径中每一段的最大斜率为 2,所以,图 6 中第 1 条斜虚线以上和第 2 条斜虚线以下的格点是路径达不到的格点。

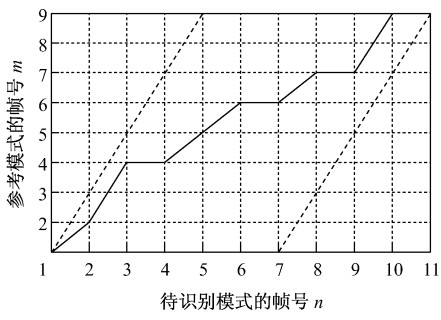


图 6 路径约束示意图

2 条约束线的表达式为

$$y_{\max} = 2x - 1, \quad 1 \leqslant x \leqslant \frac{1}{2}(M + 1) \tag{19}$$

$$y_{\min} = 2x + (M - 2N), \ N - \frac{1}{2}(M - 1) \leqslant x \leqslant N \tag{20}$$

这样,在程序中只需计算 2 条约束线之间(包括约束线上)的格点的帧匹配距离和积累距离,就可得到最终的积累距离 $D[n_N, m_N]$,与待识别模式距离最短的参考模式所对应的说话内容即为识别结果。这种方法减小了运算量,而不会对计算结果产生影响。

3 算法软件流程

基于 VC++的特定人语音识别算法软件流程如图 7 所示。在训练阶段,以文本文件的形式保存参考模式矩阵和相对应的说话内容;在识别阶段,以特定的格式将各个参考模式矩阵从文本文件中读取出来,并分别计算它们与待识别模式矩阵的距离。由于用 DTW 算法进行模式匹配需要一定的时间,而且参考模式库越大,需要的时间越长,为了避免在计算的过程中发生输入缓冲区溢出问题,在程序中通过端点检测获得语音段之后,先关闭音频输入设备,当计算结束并输出识别结果后,再重新打开设备接收数据。

4 测试结果

随机选择了 6 名测试者对基于 VC++的特定人语音识别算法的识别能力进行了测试。在训练阶段,为每一名测试者分别建立一个参考模式库,每个模式库中保存有 20 个短词和 20 个短句的 MFCC 特征参数矩阵,测试结果见表 1。其中短词为 1~3 个字组成的词语,短句为 4~10 个字组成的句子。

从表 1 可看出,基于 VC++的特定人语音识别算法对短词的识别率在95%以上,对短句的识别

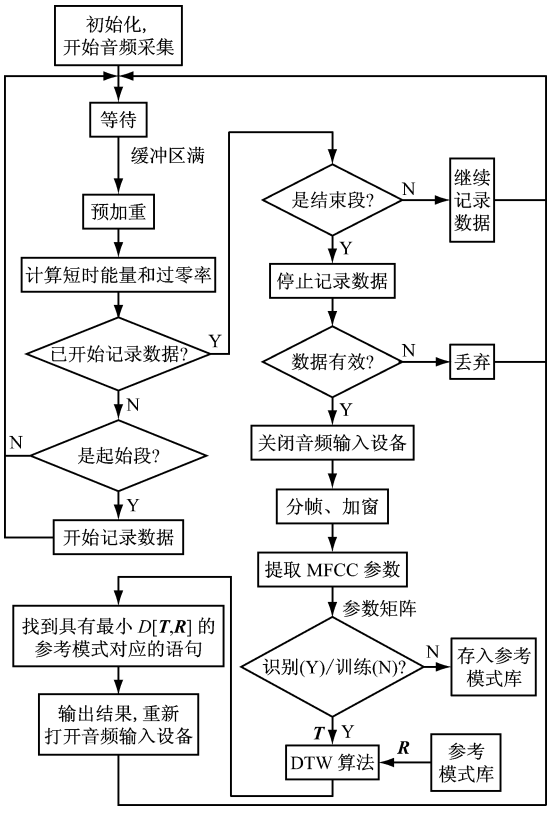


图 7　基于 VC++的特定人语音识别算法软件流程

表 1　测试结果

测试者	性别	短词识别率/%	短句识别率/%
A	男	97.5	95.0
B	男	95.0	90.0
C	男	97.5	95.0
D	女	100	97.5
E	女	97.5	92.5
F	女	100	90.0

率在 90%以上,识别准确率较高。由于算法中加入了可容忍静音时间的判断,识别的延迟时间大约为 0.3 s。

5　结语

分模块介绍了特定人语音识别算法的各个步骤在 VC++中的具体实现。采用先预加重后端点检测的方法来消除低频噪声对检测算法的影响;加入了可容忍静音时间的判断来保证检测到的语音数据的完整性;采用改进的 DTW 算法进行模式匹配,在不影响计算结果的前提下减少了运算量。最后通过不同的测试者对软件的识别效果进行了测试,结果表明,基于 VC++的特定人语音识别算法能够对短词和短句进行实时、准确识别,可达到简单应用的程度。而如何在强背景噪声的情况下有效地提取语音信号并进行准确识别,将是下一步研究的重点。

参考文献:

[1] 彭辉,魏玮,陆建华. 特定人孤立词的语音识别系统研究[J]. 控制工程,2011,18(3):397-400.

[2] 何强,何英. MATLAB 扩展编程[M]. 北京:清华大学出版社,2002.

[3] 孙鑫,余安萍. VC++深入详解[M]. 北京:电子工业出版社,2006.

[4] 范文庆,周彬彬,安靖. 精通 Windows API——函数、接口、编程实例[M]. 北京:人民邮电出版社,2009.

[5] FEI W C,BAI L. Pattern recognition method for size series of cocoon filament[J]. Japan Silk Science and Technology,2005,14(9):81-85.

[6] 胡航. 语音信号处理[M]. 哈尔滨:哈尔滨工业大学出版社,2000.

[7] SAMBUR M R, RABINER L R. A speaker-independent digit-recognition system[J]. Bell System Technical Journal,1975,54(1):81-102.

[8] 胡金平,陈若珠,李战明. 语音识别中 DTW 改进算法的研究[J]. 微型机与应用,2011,30(3):33-35.

[9] 胡广书. 数字信号处理理论、算法与实现[M]. 2 版. 北京:清华大学出版社,2003.

[10] 张杰,黄志同,王晓兰. 语音识别中隐马尔科夫模型状态数的选取原则及研究[J]. 计算机工程与应用,2000,36(1):67-69.

[11] 赵力. 语音信号处理[M]. 北京:机械工业出版社,2009.

[12] YOU K H,WANGH H C. Robust features for noisy speech recognition based on temporal trajectory filtering of short-time autocorrelation sequences[J]. Speech Communication,1999,28(1):13-24.

[13] 俸云,景新幸,叶懋. MFCC 特征改进算法在语音识别中的应用[J]. 计算机工程与科学,2009,31(12):146-148.

[14] 吕霄云,王宏霞. 基于 MFCC 和短时能量混合的异常声音识别算法[J]. 计算机应用,2010,30(3):796-798.

[15] MIZUHARA Y, HAYASHI A, SUEMATSUN. Embedding of time series data by using dynamic time warping distances[J]. Systems and Computers in Japan,2006,37(3):1-9.