

# 哈尔滨工业大学

# 实验报告

## 实验（一）

题 目 语音信号的端点检测

专 业 计算机科学与技术

学 号 1160300424

班 级 1603106

学 生 付惠珊

指 导 教 师 郑铁然

实 验 地 点 G709

实 验 日 期 2018.10.15

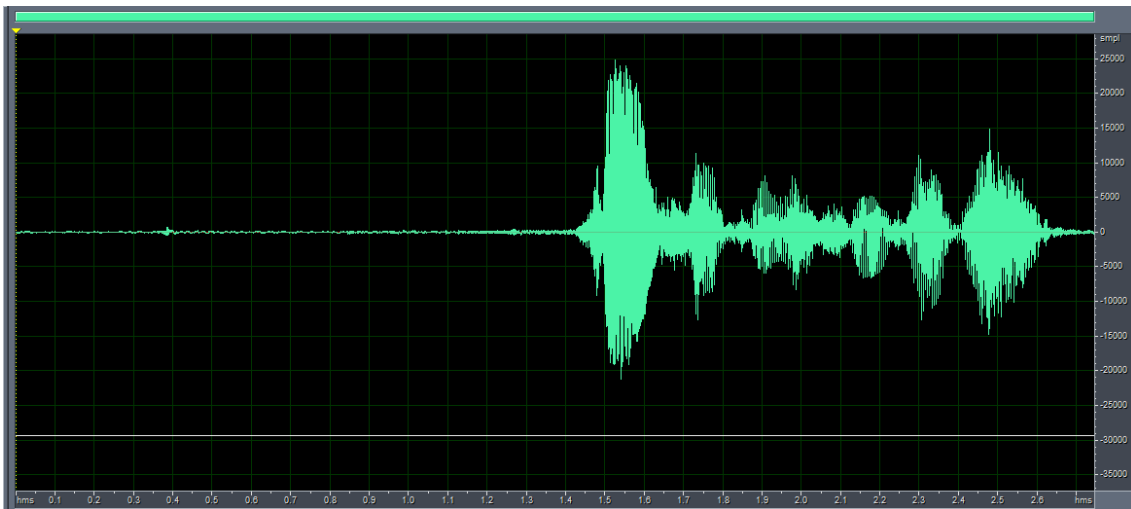
计算机科学与技术学院

## 目录

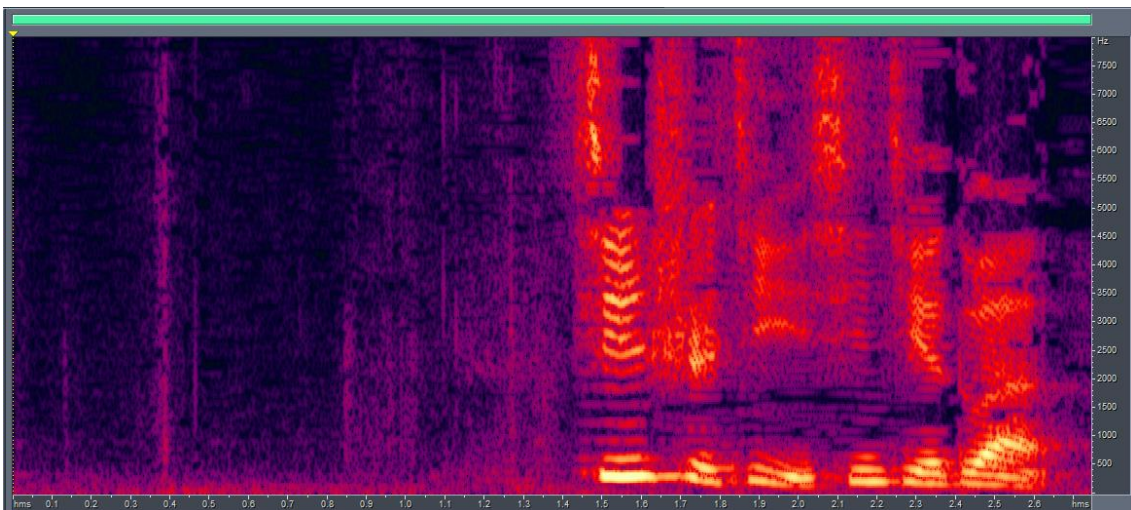
一、 语音编辑和处理工具的使用 .....	- 3 -
1.1 语音文件的时域波形截图 .....	- 3 -
1.2 语音文件的语谱图截图 .....	- 3 -
1.3 第一个音节的时域波形截图 .....	- 3 -
1.5 语料的格式 .....	- 4 -
二、 能量和过零率特征提取 .....	- 5 -
2.1 能量特征 .....	- 5 -
2.1.1 算法 .....	- 5 -
2.2.2 代码 (MATLAB) .....	- 5 -
2.2 过零率特征 .....	- 5 -
2.2.1 算法 .....	- 6 -
2.2.2 代码 (MATLAB) .....	- 6 -
三、 端点检测算法 .....	- 7 -
3.1 算法描述 .....	- 7 -
3.2 算法 .....	- 7 -
3.3 代码 (MATLAB) .....	- 8 -
四、 计算检测正确率 .....	- 9 -
4.1 “1.WAV”语料去除静音后的时域波形截图 .....	- 9 -
4.2 正确率 .....	- 10 -
五、 总结 .....	- 10 -
5.1 请总结本次实验的收获 .....	- 10 -
5.2 请给出对本次实验内容的建议 .....	- 10 -

## 一、 语音编辑和处理工具的使用

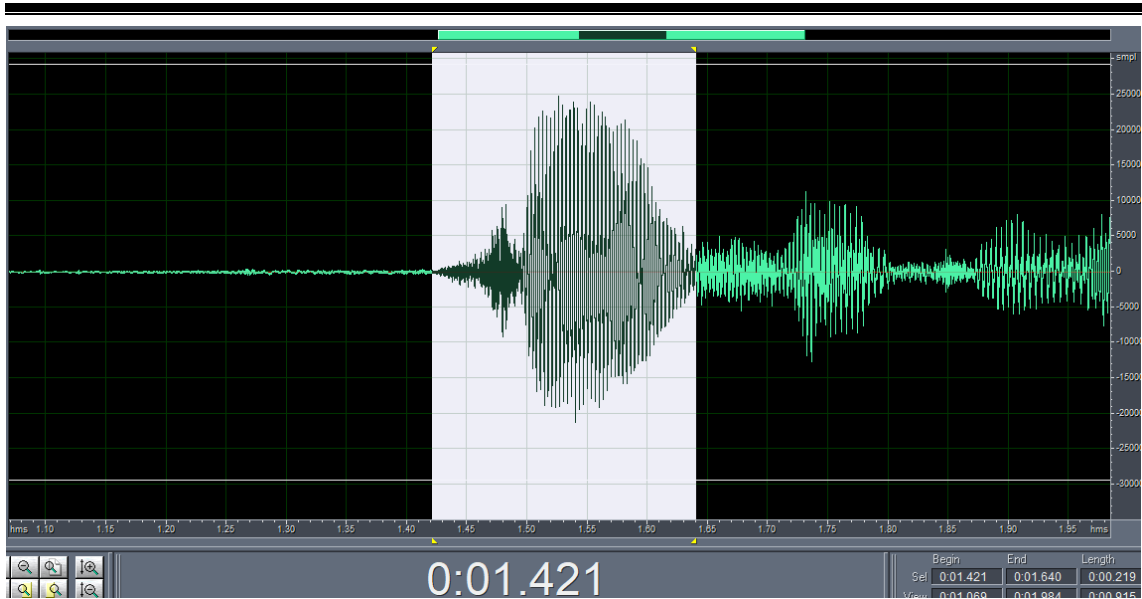
### 1.1 语音文件的时域波形截图



### 1.2 语音文件的语谱图截图



### 1.3 第一个音节的时域波形截图



## 1.5 语料的格式

采样频率 =16000Hz

量化比特数=16bit

声道个数 =84K

## 二、能量和过零率特征提取

### 2.1 能量特征

#### 2.1.1 算法

假设语音数据是  $n \times \text{frame}$  的矩阵（ $n$  为每帧样本数,  $\text{frame}$  为帧数）

$$E_n = \sum_{m=-\infty}^{\infty} [x(m)w(m)]^2$$

由短时能量计算公式：

可知：对每帧数据的每个样本点加窗后，计算加窗后的平方，然后对这一帧内所有样本点求和即可得到此帧的能量

由于样本是存储在矩阵内的，矩阵的每一列就是一帧，所以由  $\text{frame}$  列，所以构造一个方窗矩阵,也是  $n \times \text{frame}$  的，然后进行矩阵点乘后平方，再对每列求和，得到的一个行向量就是能量行向量（行向量的每一列对应样本矩阵每一列的能量）

#### 2.2.2 代码（MATLAB）

（计算的能量是归一化数据之后的）

```
function energy = Energy(xx)
%计算能量，每一帧的能量存储在一列里，数据
[N,frame]=size(xx); %N 行,frame 列

W=ones(N,frame);%方窗函数
afterWindow=W.*xx; %每一帧都加窗
square=afterWindow.*afterWindow; %平方

%计算每一帧的能量，每一帧的能量存储在一行里
energy=sum(square); %计算平方后的每一列之和，也就是短时能量
    %Energy 是 1*frame 的行向量，为了让每一帧的能量都占一行，需要转置
    %一下然后写入文件
end
```

### 2.2 过零率特征

### 2.2.1 算法

由过零率计算公式

$$Z_n = \sum_{m=-\infty}^{\infty} |\text{sgn}[x(m)] - \text{sgn}[x(m-1)]| w(m)$$

$$\text{sgn}[x(n)] = \begin{cases} 1 & x(n) \geq 0 \\ -1 & x(n) < 0 \end{cases} \quad w(n) = \begin{cases} 1/2N & 0 \leq n \leq N-1 \\ 0 & \text{其它} \end{cases}$$

可先得到一个 sig 函数，然后如果相邻两点函数值之差为 0，说明这两点之间并没有零点，如果之差的绝对值为 2，则说明两点之间一定存在一个零点，将每一帧内样本数据相邻两数据之差的绝对值求和，然后乘以  $1/(2*n)$  即可得到这一帧的过零率。

对样本  $n*frame$  的矩阵来说，sig 函数也为  $n*frame$  的矩阵（矩阵值跟与样本值计算得到），然后对整个矩阵用第  $i+1$  行-第  $i$  行得到一个新的矩阵  $Z(n-1)*frame$ ，此矩阵内存储的是零点信息，然后对矩阵按列求和，再点乘  $[1/(2*n)]$ ，得到的行向量存储的就是过零率。

### 2.2.2 代码（MATLAB）

```
function zero = Zero(xx)
%计算过零率
[N,frame]=size(xx); %N 行,frame 列
wn=1/(2*N);
sgn=ones(N,frame); %将所有转化为符号函数计算后的值
for i=1:N
    for j=1:frame
        if(xx(i,j)<0)
            sgn(i,j)=-1;
        end
    end
end
Z=zeros(N-1,frame);
%计算有多少个零点
for j=1:frame
    for i=2:N
        Z(i-1,j)=abs(sgn(i,j)-sgn(i-1,j));
    end
end
zero=sum(Z).*wn; %按列求和，得到过零率
end
```

## 三、 端点检测算法

### 3.1 算法描述

在开始进行端点检测之前，首先为短时能量和过零率分别确定两个门限：一个是比较低的门限，其数值比较小，对信号的变化比较敏感，很容易就会被超过。

另一个是比较高的门限，数值比较大，信号必须达到一定的强度，该门限才可能被超过。

低门限被超过未必就是语音的开始，有可能是时间很短的噪声引起的。高门限被超过则可以基本确信是由于语音信号引起的。

假设本次实验设计的门限 过零率:  $zcrLow$ ,  $zcrHigh$ ; 能量:  $enLow$ ,  $enHigh$  起点  $head$ , 终点  $tail$  语音帧计数  $count$ , 静音帧计数  $silence$ , 最大静音长度  $maxSilence$ , 最短语音帧数  $minlen$  (小于此值则默认为不是语音)

整个语音端点检测可以将每段分为 4 段: 静音, 过渡段, 语音段, 结束。程序中使用一个变量  $state$  来表示当前帧所处的状态。在静音段, 如果短时能量或过零率, 超越了低门限, 就应该开始标记起点  $head$ , 进入过渡段。在过渡段中, 由于参数的数值比较小, 不能确信是否处于真正的语音段, 因此只要两个参数的数值回落的到低门限以下, 就将当前状态恢复到静音状态, 而如果过渡段中两个参数中任意一个超过了高门限, 就可以确信进入了语音段, 如果过渡段静音长度超过了最大静音长度( $maxSilence$ ), 则认为语音截至, 标记  $tail$ 。

### 3.2 算法

在该矩阵中就是对所有帧进行循环判断, 先初始化数值, 假设  $state=0$ ,  $count=0$ ,  $silence=0$ , 还有所有的门限。

然后进入循环:

对每一帧, 用一个  $switch\ state$  的语句判断  $state$  的值 (此时  $state$  存储的是当前帧的上一帧的状态,

如果  $state=0$  或者 1, 说明上一帧处于静音或者过渡阶段:

    如果当前帧的能量高于  $enHigh$

        更新  $state=2$ , 进入语音状态

        设置起始点  $Head=\max(i-count-1, 1)$ ;  $i$  是已经判断完了的帧数,  $count$  是目前为止走过的语音帧数, 只有在这里才能判断是起始点

        语音帧计数+1 :  $Count+1$

        重新开始记录静音段的帧数  $silence=0$

    如果当前帧的能量或者过零率有一个超过了最低门限, 说明这一帧可能处于过渡段

更新 state=1, 进入过渡状态  
 语音帧计数+1 : Count+1  
 否则, 说明这一帧处于静音状态  
 更新 state=0  
 语音帧计数归零  
 如果 state=2 说明上一帧处于语音段  
 如果当前帧的能量和过零率都高于于低阈值, 则认为当前帧也为语音段  
 语音段计数+1  
 否则; 该帧则进入了静音段  
 静音段计数+1;  
 如果静音段的长度小于 maxSilence, 则该帧仍属于语音段, count+1  
 如果静音段长度大于 maxSilence 但是语音段计数小于 Minlen, 则  
 认为之前判断的语音段不是语音 (可能是突发噪声), 将  
 count, silence 置 0, 此帧状态值 0.  
 否则该帧为截止帧 (截至帧需要减去尾部的静音帧)

### 3.3 代码 (MATLAB)

```
function [head,tail]=VAD(xx)

en=Energy(xx);
zero=Zero(xx);

%设置门限
maxSilence=10; %允许的最大静音长度
minlen = 20; % 最短语音长度
zcrLow=5; %短时过零率低门限
zcrHigh=10; %短时过零率高门限
steLow=min(0.002,max(en)/4); %短时能量低门限
steHigh=min(10,max(en)/8); %短时能量高门限
[~,frame]=size(xx);

state=0; %初始状态为 0->静音状态 1->可能语音 2->语音状态 3->语音状态结束
count=0; %语音段计数 (包括大于能量高门限的和大于能量低门限或者
        %大于过零率低门限的) =>非静音状态的片段
silence = 0; %初始静音段长度为 0

head=0;
for i=1:frame %对每一帧都进行判断
    switch state
        case {0,1}
            if(en(i)>steHigh) %该帧能量大于能量高门限, 进入语音阶段
                state=2; %标志为语音阶段
                head = max(i-count-1,1);
                silence=0;
                count=count+1;
            elseif(en(i)>steLow||zero(i)>zcrLow) %可能处于语音阶段
                state=1; %标志状态为可能处于语音状态
```



```

        count=count+1;
    else %处于静音状态
        state=0;
        count=0;
    end
case 2
    if (en(i)> steLow||zero(i)>zcrLow) % 保持在语音段
        count = count + 1;

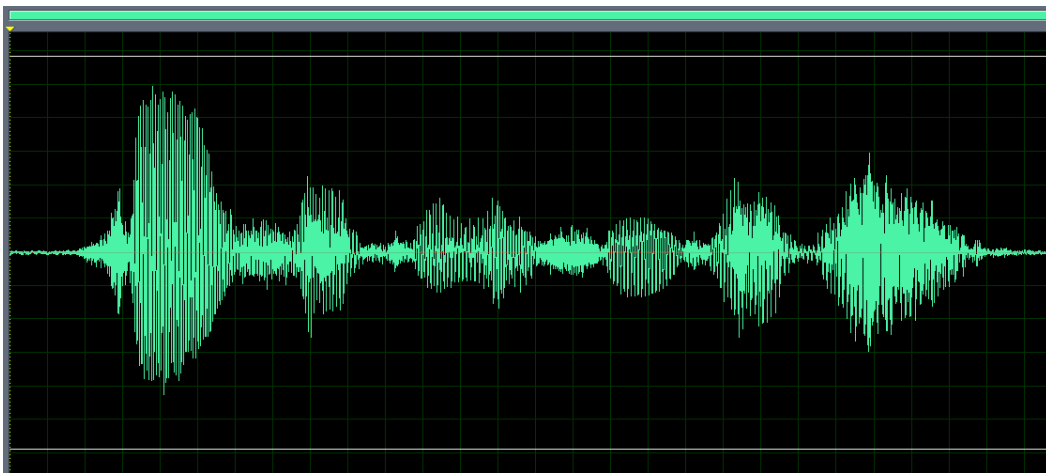
    else % 语音将结束
        silence = silence+1; %静音片段+1
        if silence < maxSilence % 静音还不够长, 尚未结束
            count = count + 1;
        elseif count < minlen % 语音长度太短, 认为是噪声, 需要将所有
的计数归零
            state = 0;
            silence = 0;
            count = 0;
        else % 静音足够长了, 语音结束
            state=3;
        end
    end
case 3
    break;
end

end
count=count-silence/2;
tail=head+count-1;
end

```

## 四、 计算检测正确率

### 4.1 “1. wav” 语料去除静音后的时域波形截图



## 4.2 正确率

正确检出文件的个数:

正确率= 60 %

# 五、 总结

## 5.1 请总结本次实验的收获

1. 本次实验让我对语音处理有了初步的了解,掌握了短时能量和过零率在端点检测(双门限法)中的应用以及双门限算法的基本流程和 MATLAB 实现。
2. 实验中将读取的音频是 wav 格式,最后存储端点检测之后的音频是 raw pcm 格式。完成实验过程中我对 wav, pcm 格式有了较深入的了解,学会了如何在 pcm 与 wav 格式之间转换。
3. 本次实验我是在 MATLAB 平台上完成的, MATLAB 用于处理语音方面相关内容的库可以说是非常强大了。比如用于读取音频文件的 `audioread` 函数,可以根据参数选择读取的音频向量是原始数据类型还是归一化为 `double` 的类型。而 MATLAB 处理矩阵的强大能力也使得将从 wave 里得到的数据分帧,以及提取特征,得到端点检测后的数据变得十分方便。

## 5.2 请给出对本次实验内容的建议

希望老师以后实验要求可以更具体,比如处理数据是否需要归一化计算,处理文件格式可用什么格式等。


### 附录


\_VAD.pcm 文件存储的是进行端点检测之后的预料转化为 raw pcm 格式存储


\_VAD.wav 文件存储的是进行端点检测后的语料转化为 wav 格式存储


\_en.txt 存储每一帧的能量，按行存储


\_zero.txt 存储每一帧的过零率，按行存储


 Energy.m


 getData.m


 main.m


 VAD.m

 vadData.m

 writeFile.m

 writePcm.m

 writeWav.m

 Zero.m

Energy, Zero 用于计算能量,过零率, getData 用于从音频文件中得到各种数据

Main 是主函数, VAD 是端点检测函数, vadData 是得到端点检测后的数据

WriteFile 用于将数据写入 txt 文件（能量，过零率） writePcm,writeWav 将数据存储为 Pcm,wav 格式