浙江大学

本科实验报告

课程名称：远程声控系统实现

姓　　名：

学　　院：信息与电子工程学院

专　　业：信息工程

学　　号：

指导老师：张朝阳

2023 年 6 月 4 日

一、 问题背景

实现一个远程声音控制系统。首先采集不同的语音指示信号，进行适当压缩; 然后通过噪声信道实现远程传输，远端接收后再通过适当计算识别出是何指示, 最后送入一个处于未知状态、但能控/能观的控制系统，完成不同的控制动作

二、 问题分析

分析此问题，可以分为信号采集、信号传输、语音识别、指令控制四部分。

具体流程图如下：

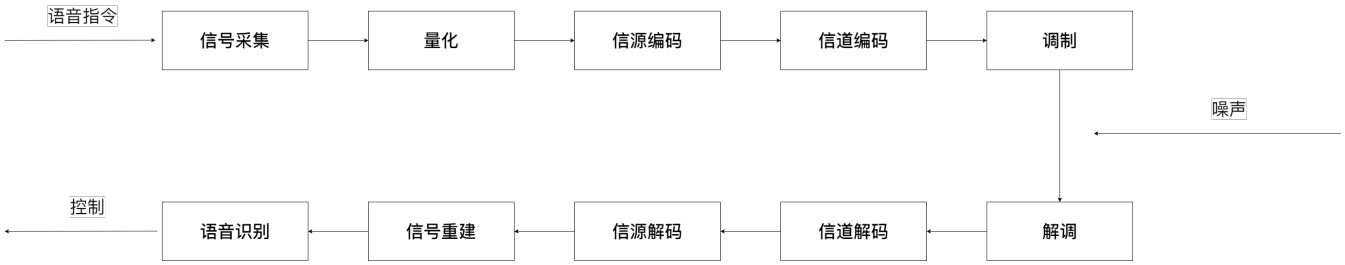

图 1: 流程图

其中信号采集可以直接使用 matlab 自带的函数实现。

信号传输包括了信号量化、信源编码、信道编码、调制解调、信道译码、信源译码六个过程。其中信源编码我们采用了霍夫曼编码，信道编码采用了卷积码，调制选用了 BPSK 调制，并人为引入了高斯白噪声模拟传输时的噪声。

语音识别部分通过 Matlab 的已经训练好的模型进行识别。

三、 具体实现过程

1. 语音指令信号采集

由于人发出的信号被麦克风采集到时，是连续的模拟信号，所以需要将其转换为数字信号，才能被我们后续的处理。

这里我们直接采用了 Matlab 中的 audiorecorder 函数进行了录制，设置采用率为 16000Hz，采样位数 16bits，通道数 1，录音时长 1s。此处我们录制右转指令“right”，得到时域图和频域图如下：

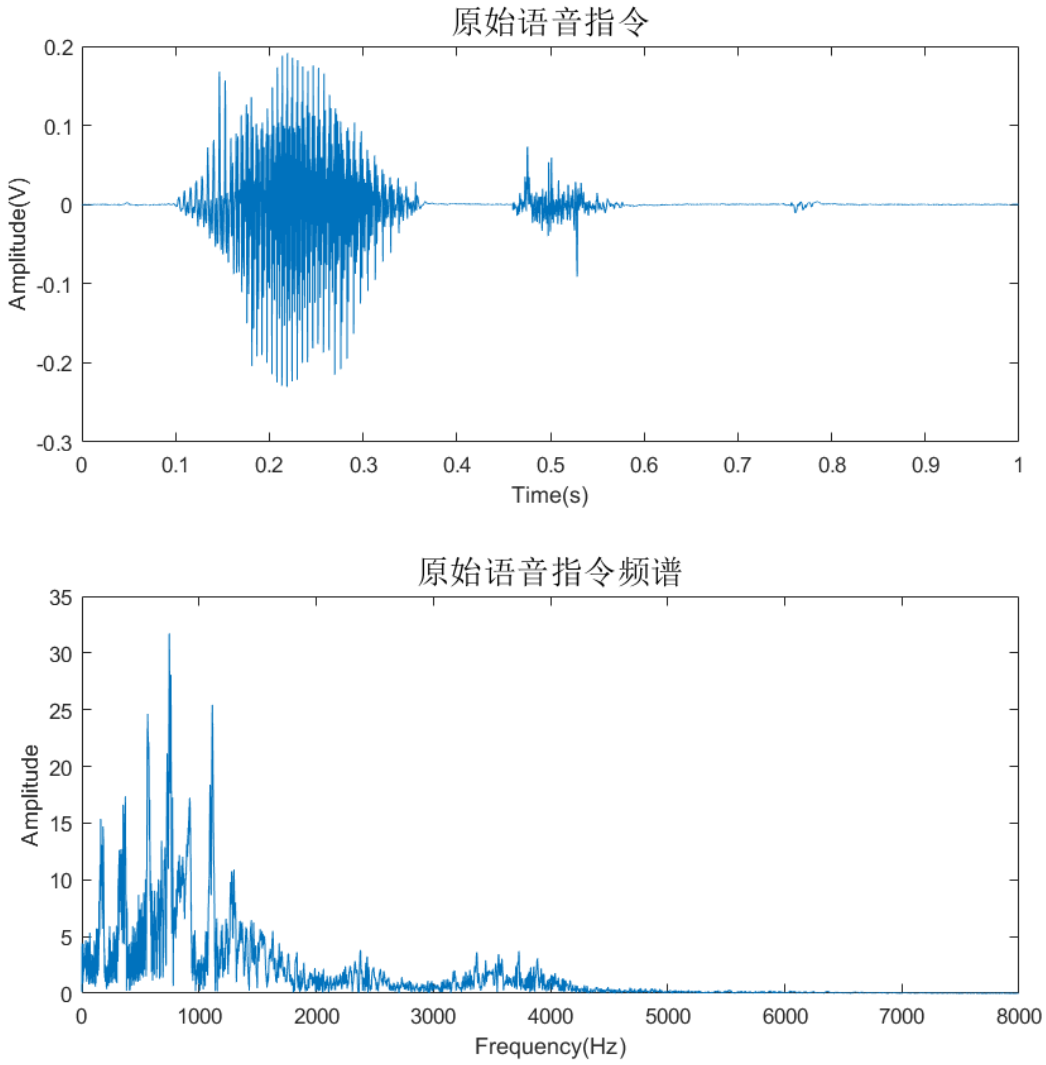

图 2: 采集到的语音指令信号时域和频域图

2. 信号量化

采样到的电压幅度值是模拟的，为了后续的处理，我们还需要对其进行量化编码，使其电压幅度值映射到离散电平，并用二进制表示。此处我们使用了量化等级为 8 的量化器进行量化，具体分辨率为为

$$q = \frac{V_{max} - V_{min}}{2^N} = \frac{1-(-1)}{2^8} = 3.90625 \times 10^{-3} V$$

3. 信源编码

信源编码又称为数据压缩，是一种以提高通信有效性为目的而对信源符号进行的变换，或者说为了减少或消除信源冗余度而进行的信源符号变换。

而观察我们的信号频域图可以发现，信号在不同的频率处高度并不一样，而且差异很大，所以我们选择了我们所学过的不等长编码中的霍夫曼编码，用以压缩信号和节省带宽。

实验中我们先计算各个负号出现的概率，再利用 Matlab 自带的函数 huffmandict 生成字典，在通过 huffmanenco 生成霍夫曼编码。

以下为编码结果：

```
----- Source Encode -----
平均码长 : 2.181875
信源熵 : 2.102782
编码效率 : 0.475560
编码前字符串总长度 : 128000
编码后字符串二进制总长度 : 34910
压缩率 : 3.666571
```

4. 信道编码

实际信号传输的过程中，信道存在噪声和干扰，会使得我们接受的码字和发送的码字之间存在错误。所以可以通过信道编码，在发送的信息码元中加入一些冗余码元，进行监督，从而能够在出错时进行纠正，达到改善传输和纠错的目的。

由于课程中并没有讲信道编码，所以在查阅资料后，我采用了卷积码对信道进行编码。

卷积编码属于信道编码的一种，信道编码的作用是通过增加冗余来提高传输的准确性，当然这样会导致信息传输速率的下降。卷积码是一种前向纠错码，比一般的分组码可以拥有更好的性能，因为卷积编码输出的每个码即和当前输入数据有关，也和之前输入的数据有关。

一般用（n,k,L）来描述卷积码，即我们输入 k 个信息比特经过卷积码编成 n 个比特（n 一定是大于 k 的，就是加进去了冗余来进行纠错），L 是编码的约束长度。编码器由 n 个模 2 加法器和 L*k-1 级移位寄存器组成。

此处我们采用了（2,1,7）卷积编辑器进行编码，而 Matlab 中有内置的卷积吗实现的函数即：

```
trellis = poly2trellis(ConstraintLength, CodeGenerator);
code_data = convenc(P_data, trellis);
```

其中 poly2trellis 函数将 (2,1,7) 卷积码生成了 Matlab 的网格表达式，随后利用 convenc 函数对输入序列进行了编码。

编码结果如下：

```
----- Channel Encode -----
信道编码前字符串长度 : 34910
信道编码后字符串长度 : 69820
编码速率：0.5
```

5. 调制解调

由于课程中并没有讲调制和解调，所以在查阅资料后，我采用了相移键控进行调制。此处我们采用二进制相移键控 BPSK，并且为了模拟实际传输过程中的噪声干扰，我们在调制好的信号中加入了加性高斯白噪声，取信噪比为 10dB。

此处用 Matlab 中的函数

```
bpsk_out = pskmod(channel_encode,2);
```

进行 BPSK 调制

```
awgn_out = awgn(bpsk_out, 10);
```

进行高斯噪声的加入。

而解调时任然使用 pskmod 函数。

结果如下：

```
-------- Modulation --------
调制解调出错数 : 0
调制解调误码率 : 0.000000
```

6. 信道译码

信道编码采用的是卷积码，译码时使用 vitdec 函数进行译码。

结果如下：

```
-------- Channel Decode --------
信道解码出错数 : 0
信道解码误码率 : 0.000000
```

7. 信源译码

信源编码采用的是霍夫曼编码，译码采用 huffmandeco 函数。

结果如下：

```
-------- Channel Decode --------
信道解码出错数 : 0
信道解码误码率 : 0.000000
```

四、 信号重建

信号重建只需要将信源译码得到的离散信号和量化分辨率相乘再减一。

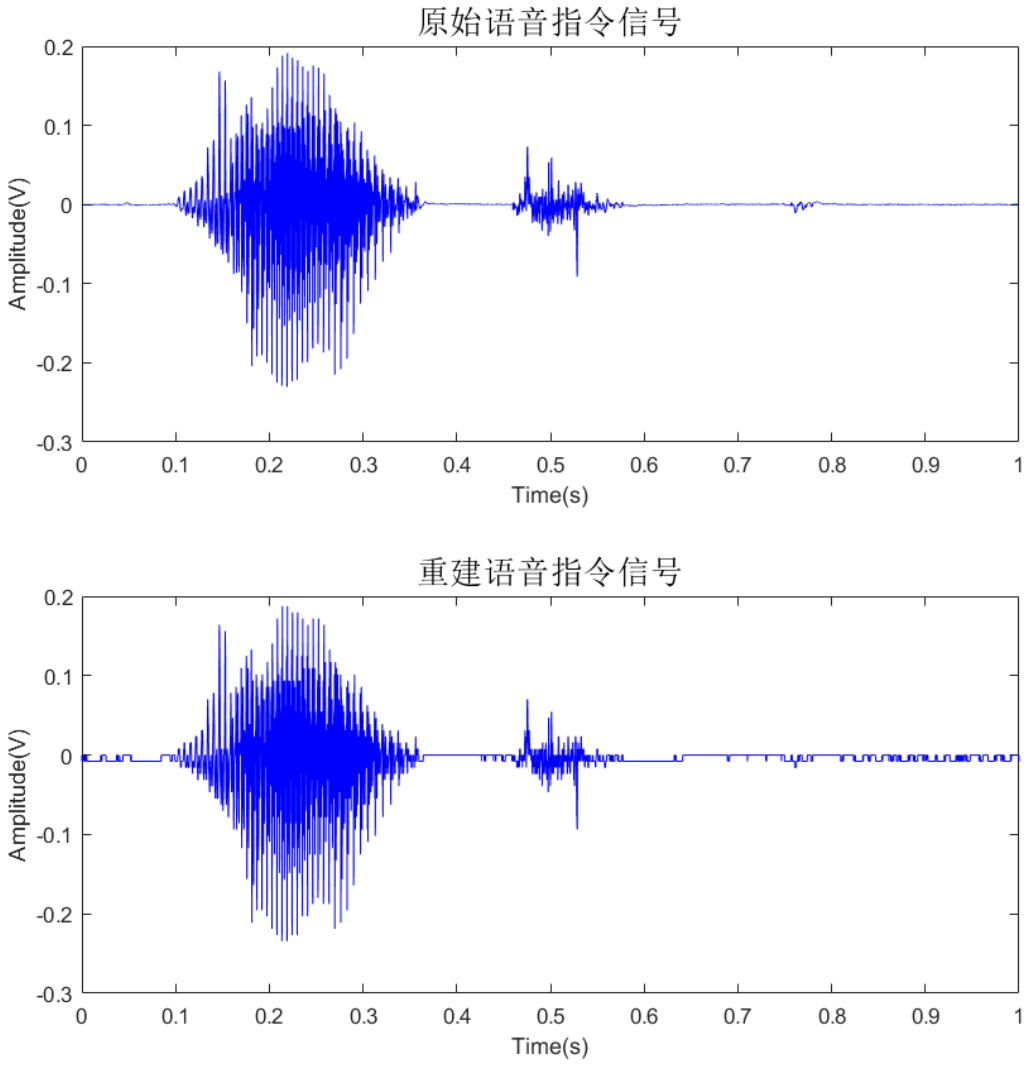

图 3: 原始信号和重建信号时域对比

此外我们使用了 Matlab 中的 mse 函数计算了原始信号和重建信号的均方误差。结果如下：

```
-------- Mean Square Erro --------
均方误差: 0.000022
```

可以看出，无论是通过信号时域和频域对比图还是通过均方误差来看，原始信号和重建信号的差异都非常小。

1. 语音识别

语音识别部分，通过查阅资料学习，我们采用了现在普遍采用的 CNN 模型进行处理，此处我们直接使用了 Matlab 的 Deep Learning Toolbox 中预先训练好的模型。改模型可以识别简单的 go，stop，yes，no，up down，left，right，on，off 十条指令。并且本次实验我们以指令“right”为例。最后识别结果如下：

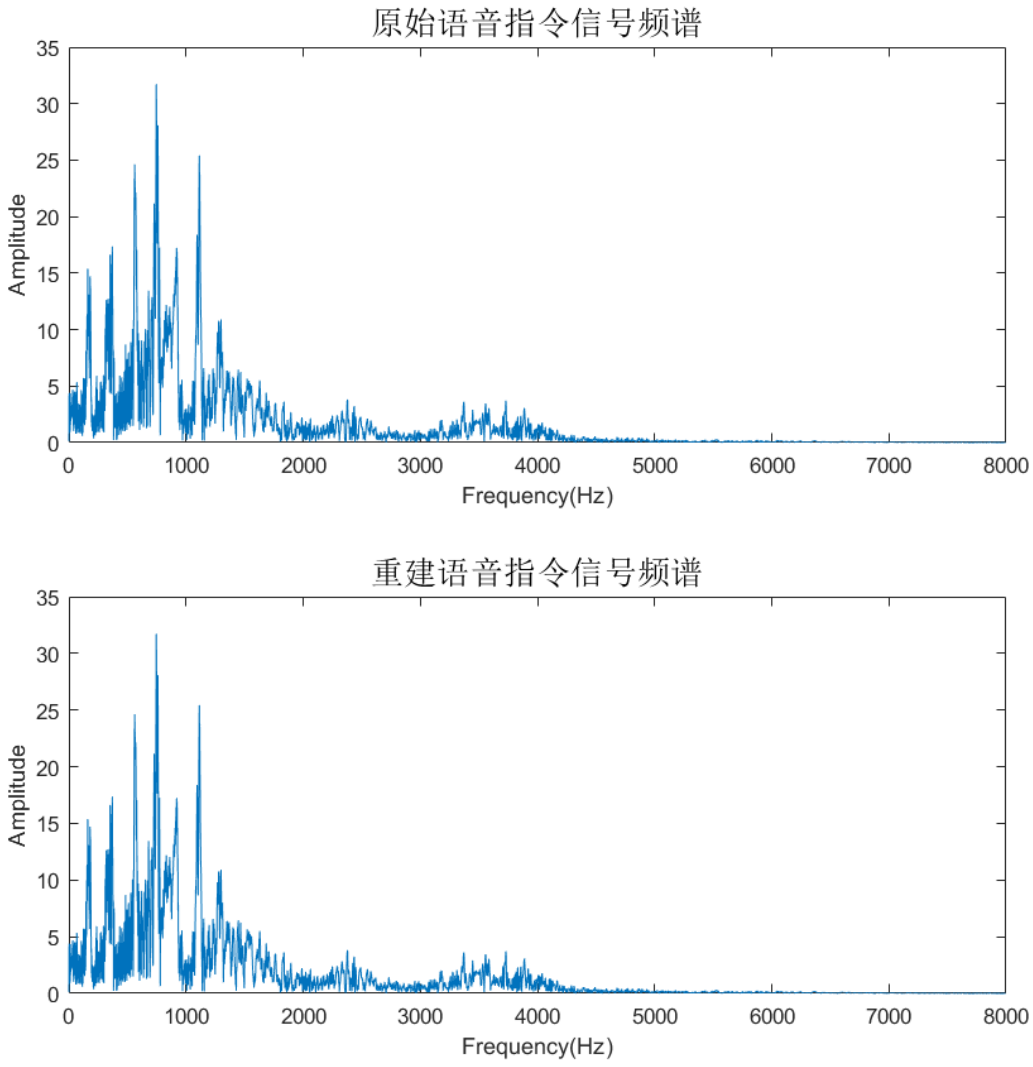

图 4: 原始信号和重建信号频域对比

```
-------- Speech Recognition --------
识别结果：right
```

2. 控制

控制阶段，我们假设了一个有限状态机进行动作的控制，状态机根据输入的指令以及当前的状态共同决定下一个状态。从而进行控制。

五、 心得体会

刚开始本次试验时，我觉得十分棘手，虽然课程中已经讲了许多的相关知识，但是自己缺乏对整个信号传输以及控制的系统性认知，所以导致刚开始时无从下手。

在此基础上，我向同学以及学长请教，首先大概理清了整个问题大概的实现过程。之后，我开始着手一步步对，每一个过程进行分析，期间除了回顾学过的信源编码等已经上过的知识，也查阅学习了如信道编码和调制解调等许多新知识。

同时也查阅了相关实验的文章、博客教程，最后通过 Matlab 对该问题进行了简单的实现。

六、 参考文献

1. matlab 有关卷积码编译码的函数.https://blog.csdn.net/xiangziling/article/details/106908643
2. matlab 实现卷积编码' 适合小白理解学习'.https://blog.csdn.net/wang6350339/article/details/94837370
3. PSK 原理及 MATLAB 仿真.https://www.docin.com/p-680830415.html
4. Pskmod 函数 _Matlab 笔记——AWGN 函数详解与实例——AWGN 加性高斯白噪声 +QPSK 调制.https://blog.csdn.net/weixin_39585675/article/details/113312105
5. 薛年喜.MATLAB 在数字信号处理中的应用 [M]. 北京: 清华大学出版社,2003.
6. 郑君里, 应启衍, 杨为理. 信号与系统 [M]. 北京: 高等教育出版社,2000

七、 附录：实验代码

实验代码

```matlab
clear;
clc;
close all;
%% 录制语音信号
    % 采样频率16000Hz，采样位数16bits，通道数1，录音时长1s
    Fs = 48000;
    R = audiorecorder(Fs, 16 ,1) ;
    disp('Start')
    recordblocking(R,2);
    disp('End');
    cmd = getaudiodata(R);
    audiowrite('MyAudio.flac', cmd, Fs);

%% 指令读取
    [MyAudio1, fs] = audioread('MyAudio.flac');
```

```
    AudioLength1 = length(MyAudio1);
    MyAudio = MyAudio1(30000:60000);
    AudioLength = length(MyAudio);
    t = (0:AudioLength-1)/fs;
    fft_MyAudio = fft(MyAudio, fs);
    f = fs*(0:fs/2-1)/fs;
    f1 = figure(1);
    subplot(211);
    plot(t, MyAudio);
    title('原始语音指令','Fontsize', 16);
    xlabel('Time(s)');ylabel('Amplitude(V)');
    subplot(212);
    plot(f, abs(fft_MyAudio(1:fs/2)));
    title('原始语音指令频谱','Fontsize', 16);
    xlabel('Frequency(Hz)');ylabel('Amplitude');

%  量化
    % 量化bit
    [Vmax, ~] = max(MyAudio);
    [Vmin, ~] = min(MyAudio);
    qN = 4;
    q = (Vmax-(Vmin))/(2^qN-1);
    qAudio = [];
    for i = 1:AudioLength
        temp = (MyAudio(i)+Vmax)/q;
        qAudio = [qAudio; floor(temp)];
    end

    qMin = min(qAudio);
    qAudio2 = qAudio - qMin;
    tx_bit_audio = [];
    flag = [];
    for i = 1:length(tx_bit)
        temp = dec2bin(qAudio2(i),4)';
        temp = str2num(temp);
        if length(temp) > 4
            flag = [flag, i];
        end
        tx_bit_audio = [tx_bit_audio; temp];
    end

    tx_bit_aduio = tx_bit_audio(20000:52767);
    save tx_bit_aduio.mat tx_bit_aduio;

%  信源编码
    % unique函数：对数组按升序排序，并去掉多余的重复的值
    % cell数组：元胞数组，每个单元可以储存不同类型的变量
    % numel(A)函数：Numbers of Elements返回数组A中元素的个数
    % find(A = b)：返回一个列向量，记录了数组A中等于b的元素的位置（按列数）

```

```
66      len = length(qAudio);
67      unique_x = unique(qAudio);
68      unique_len = length(unique_x);
69
70      symbols = cell(1, unique_len);
71      p = zeros(1, unique_len);
72
73      for i = 1:unique_len
74          symbols{1,i} = unique_x(i);
75          p(i) = numel(find(qAudio==unique_x(i))) / len;
76      end
77
78      [dict, avglen] = huffmandict(symbols, p);
79
80      source_encode = huffmanenco(qAudio, dict);
81      fprintf('------ Source Encode ------\n');
82      fprintf('平均码长：%f\n', avglen );
83      fprintf('信源熵：%f\n', sum(p.*(-log2(p))) );
84      fprintf('编码效率：%f\n', 1/sum(p.*(-log2(p))) );
85      fprintf('编码前字符串总长度：%d\n', len*qN);
86      fprintf('编码后字符串二进制总长度：%d\n', length(source_encode));
87      fprintf('压缩率：%f\n', len*qN/length(source_encode));
88
89  % 信道编码
90      % 使用了matlab自带的卷积码的函数生成卷积码
91      % trellis = poly2trellis(ConstraintLength，CodeGenerator)，其中trellis是一个网格表，用来规
            定我们使用的卷积编码的规则，ConstraintLength是描述每一路输入的长度，CodeGenerator描述输入和模
            2加法器的连接关系（8进制计数），此处用了（2，1，7）卷积编辑器规则
92      % code = convenc(source_encode, trellis)输出编码后的序列
93      trellis = poly2trellis(7, [171 133]);
94      channel_encode = convenc(source_encode, trellis);
95      fprintf('------ Channel Encode ------\n');
96      fprintf('信道编码前字符串长度：%d\n', length(source_encode));
97      fprintf('信道编码后字符串长度：%d\n', length(channel_encode));
98      fprintf('编码速率：%.1f\n',length(source_encode)/length(channel_encode));
99
100
101 % BPSK调制解调
102     % 选用了二进制相移键控BPSK进行调制，并加入了高斯白噪声，信噪比为10dB
103     bpsk_out = pskmod(channel_encode,2);
104     awgn_out = awgn(bpsk_out, 10);
105     % 仍然使用pskmode 函数进行解调
106     demodulation_out = pskdemod(awgn_out,2);
107     % 计算错误率
108     [erros, ratio] = biterr(demodulation_out, channel_encode);
109     fprintf('--------- Modulation ---------\n');
110     fprintf('调制解调出错数：%d\n', erros);
111     fprintf('调制解调误码率：%f\n', ratio);
112
113 % 信道译码
```

```
114     channel_decode = vitdec( demodulation_out,trellis,32,'trunc','hard');
115     % 计算错误率
116     [erros2, ratio2] = biterr(channel_decode, source_encode);
117     fprintf('————————— Channel Decode —————————\n');
118     fprintf('信道译码出错数：%d\n', erros2);
119     fprintf('信道译码误码率：%f\n', ratio2);
120 
121 % 信源译码
122     source_decode=huffmandeco(channel_decode,dict);
123     % 计算错误率
124     [erros3, ratio3] = biterr(source_decode, qAudio);
125     fprintf('————————— Source Decode —————————\n');
126     fprintf('信源译码出错数：%d\n', erros3);
127     fprintf('信源译码误码率：%f\n', ratio3);
128 % 信号重建
129     restore_out = source_decode * q - 1;
130     fprintf('————————— Mean Square Erro —————————\n');
131     fprintf('均方误差: %f\n', mse(restore_out - MyAudio));
132     audiowrite('Restore_MyAudio.flac', restore_out, fs);
133 
134 
135 % 对比图
136     figure(2);
137     subplot(211);
138     plot(t, MyAudio, 'b')
139     title('原始语音指令信号','Fontsize', 16);
140     xlabel('Time(s)');ylabel('Amplitude(V)');
141     subplot(212);
142     plot(t, restore_out, 'b')
143     title('重建语音指令信号','Fontsize', 16);
144     xlabel('Time(s)');ylabel('Amplitude(V)');
145 
146     figure(3);
147     fft_restore = fft(restore_out, fs);
148     subplot(211);
149     plot(f, abs(fft_MyAudio(1:fs/2)));
150     title('原始语音指令信号频谱','Fontsize', 16);
151     xlabel('Frequency(Hz)');ylabel('Amplitude');
152     subplot(212);
153     plot(f, abs(fft_MyAudio(1:fs/2)));
154     title('重建语音指令信号频谱','Fontsize', 16);
155     xlabel('Frequency(Hz)');ylabel('Amplitude');
156 
157 % 语音识别
158     load('commandNet.mat');
159     auditorySpect = helperExtractAuditoryFeatures(restore_out,fs);
160     command = classify(trainedNet,auditorySpect);
161     fprintf('————————— Speech Recognition —————————\n');
162     fprintf("识别结果：%s\n", command);
```