北京大學

本科实验报告

课程名称：	机器感知实验
姓　　名：	金镇雄
学　　院：	元培
系：	智能科学系
专　　业：	智能科学与技术
年　　级：	19
学　　号：	1900094619
指导教师：	曲天书
职　　称：	副教授

2022 年　　3 月　　29 日

实验二、虚拟声音实验

一、实验目的

- 了解虚拟声的定义及原理
- 了解头相关传输函数的定义及其数据库的使用
- 掌握信号处理用于静止和运动的三维虚拟声音的基本原理和方法
- 练习在 matlab 平台上对虚拟声的实现、

二、实验要求

- 界面清晰美观
- 实现静态虚拟声音
- 实现动态虚拟声音
- 实验结果分析
- 实验讨论

三、实验原理

若将声源到达耳膜之间传输路径视为一个滤波器，则此滤波器的频率响应就包含了传输路径和耳郭对声音的共同响应，而此频率响应是头相关传输函数。自由场情况下，声波从声源到双耳鼓膜处的传输函数称为头相关传输函数（Head-Related Transfer Function，HRTF），对应的时域冲激响应称为头相关冲激响应 （Head-Related Impulse Response，HRIR）。人耳的听觉特性决定了听觉响应实际上是基于频谱的响应，而 HRTF 中包含了人体结构对声音信号的频率响应、人体的各个部位对不同频率的信号有着不同的响应。这些响应有些是有方向性的，如躯干、头、肩部和耳郭、耳腔的反射以及头部衍射，有些是无方向性的，如耳腔的回响和耳道、耳膜的阻抗。

HRTF 的原理基于耳间水平差（interaural level difference）和耳间时间差（interaural time difference）。假设声源在人的右侧，因为声音是以声波的形式传来，它首先是到达右耳，然后才到达左耳。另外，声音到达左耳时的效能比右耳更弱。这种音量下降是源于声波的自然消散，以及你的头部吸收和反射了一定的声音。左右耳朵的音量差异据取决于耳间水平差，其中的延迟则称作耳间时间差。而大脑解释声波形状的这种差异，并用来判断声音的来源。

静止虚拟声合成

根据声源方位通过数据库读取该点的左右耳 HRIR 数据，用单通道声音文件分别卷积两耳 HRIR 数据，合成双通道的声音。

运动的虚拟声合成

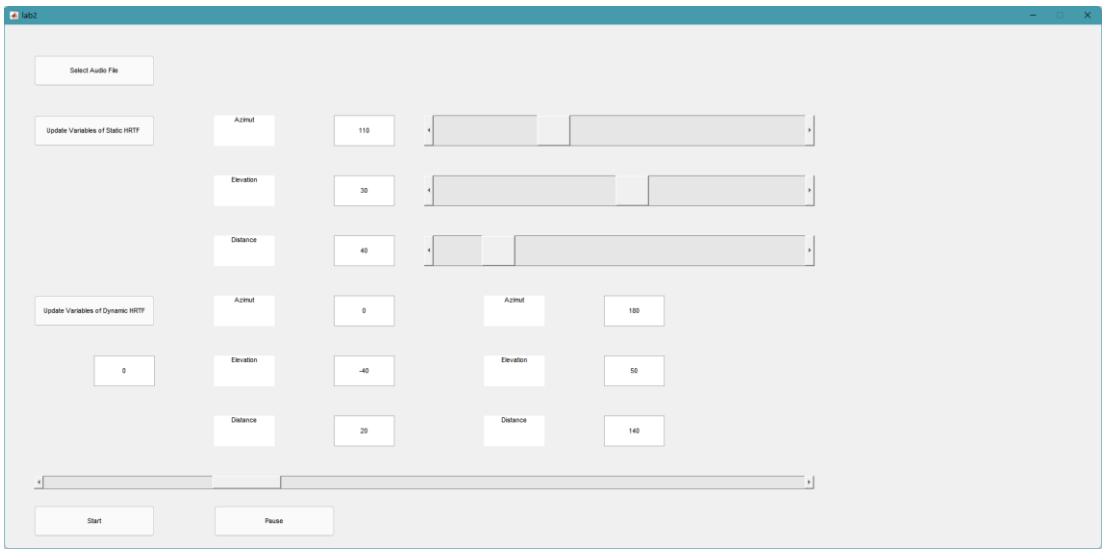
除了以上所述的处理步骤以外，还需要得到虚拟声源运动轨迹上各个间隔点上的头相关传输函数，而头相关传输函数数据库一般测量点间隔太大，需要通过对空间某一个位置的头相关传输函数进行线性插值实现。此外，原始声音经过不同头相关传输函数滤波器之后帧与帧之间需要进行平滑处理，以防止各帧之间声音的突变。

四、实验仪器设备

PC 机、耳机、HRTF 数据库、两个声音文件

五、实验内容和步骤

1. 设计 GUI 界面

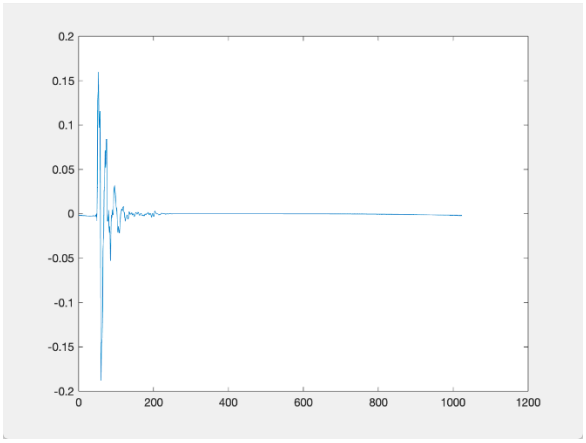


2. 静态虚拟声音合成

A. 从 HRTF 数据库中获取 HRIR 数据

一个 16K 的 HRIR 数据文件包含 2048 个 8 比特数据，其中前 1024 个数据为亚洲人的头相关冲激响应，后 1024 个为西方人的头相关冲激响应。在本次实验中使用了前一半数据。

HRIR 数据的一般波形如下：



HRTF 数据库提供了单耳的传递函数，而本实验中需要分别获取左耳和右耳的 HRIR 数据。这里假设某个方向的声音对某只耳朵的听音效果，等同于对称位置的声音对对侧耳朵的听音效果。假设想合成方位角 60 度的声音，用音频文件和 60 度、300 度的传递函数分别卷积，一个放在左耳的数据上，一个放在右耳的数据上，最后合在一起就得到了 60 度声音对双耳的听音效果，因为 60 度对右耳的传递函数，等于 300 度对左耳的传递函数。于是只用一套单耳的传递函数就可以合成一个位置对双耳的声音。

B. 对音频信号进行分帧

若将音频文件直接分段（分帧处理中无重叠部分，用矩形窗），则各帧之间声音发生突变。因此，为了帧平滑和减少频谱泄露，帧间取一定的重叠部分（overlap），且分段后的音频段需要加窗。因为 HRTF 数据库中数据是时域表示的 1024 个点，所以音频分帧采用 1024 个点的长度。除此以外，为了防止分帧后最后的音频段的长度小于帧长，在音频信号的最后补零。

本实验中每个音频段有 1024 个点，overlap 为 25%，窗函数为汉宁窗。

C. 用单通道音频文件分别卷积两耳 HRIR 数据

因为时域上两个信号做卷积相当于频域上做相乘，所以音频信号与 HRIR 数据的卷积是如下方式进行的：对 HRIR 数据和音频段都进行 fft，两个结果相乘后进行 ifft，得到相当于做卷积的时域信号。

D. 音频段合并，合成双通道的声音

需要注意，分帧时有 overlap，合并也需要 overlap。因为窗函数是对称的，在重叠部分将前一帧信号与后一帧信号直接相加即可。进行上述过程后会得到两个信号：左耳信号和右耳信号。两个信号放在一个 2xlength(audio)的矩阵中，完成双通道声音的合成。

3. 动态虚拟声音合成

A. 头相关传输函数的线性插值

采用的是距离反比加权(Inverse-Distance Weighting)的线性插值方法，这个算法是通过计算目标点最近邻的两个点的值加权得到：

$$\hat{x}_j = \frac{\sum_{i=1}^{2} w_{ij} x_i}{\sum_{i=1}^{2} w_{ij}}$$

此公式中 x 表示传输函数的值，w 表示权重，是目标点与近邻点间的直线距离。

B. 头相关传递函数的平滑

与静态虚拟声音合成类似，为了防止输出声音存在卡顿现象，帧间进行

平滑处理。具体步骤如下：

1. 按窗的形式给权重 w0，w1 赋值，保证合成后的幅值和之前的强度一致，权值的和为 1
2. 取第 i 帧前半帧数据卷积，$x_i*(w_0h_{i-1}+w_1h_i)$ 生成数据，记为 y1
3. 取第 i 帧前后帧数据卷积，$x_i*(w_0h_i+w_1h_{i+1})$ 生成数据，记为 y1
4. 把 y1 和 y2 进行拼接得到输出 y

其实此步骤与静态虚拟声音合成中进行的分帧操作基本上相同，只不过动态虚拟声音合成中要求 overlap 为 50%。

六、实验结果和分析

1. 在附件中有 static_azi250_elev20_dist50.wav 和 dynamic_azi0_elev-40_dist20-azi180_elev50_dist140.wav，分别是静态虚拟声音合成和动态虚拟声音合成的结果。静态虚拟声音合成中声源位置对应的方位角、仰角、距离分别是 250°、30°和 50，输出结果确实有 3D 音效，能感到声源在左上方。动态虚拟声音合成中声源起点位置和终点位置的方位角、仰角和距离分别是 0°、-40°、20 和 180°、50°、140。实验结果发现有马蹄声从前往后移动，声音强度从大到小发生变化。
2. 音频信号与 HRIR 信号的卷积操作

   实验中可采取音频信号与 HRIR 信号在时域上直接做卷积的方法，也可采取做傅里叶变换后在频域上做相乘的方法。结果发现两个方法的结果没有很大的区别，都有 3D 音效，而采取后者的原因是直接时域卷积在帧间平滑和拼接等等操作上不如利用频域卷积更好实现，同时算法复杂度也不如利用频域。
3. 静态虚拟声音合成中帧间平滑处理

   在静态虚拟声音合成中，对音频信号进行无重叠部分、加矩形窗的分帧后结果发现，实际效果并不理想，在每一段的音频中总会听到断点，导致音频不连贯。因此，在每一段的开头和结尾加上淡入淡出的效果，是每一段音频的连接处正常化。其实若用整个音频文件卷积对应的空间位置的 HRIR 数据，则没有必要进行分帧、加窗、合并的操作。
4. 动态虚拟合成中帧间平滑处理

   在动态虚拟合成中做片段淡入淡出效果的原因有两个方面。第一个原因与静态虚拟声音合成做平滑处理一致，与分帧有关。第二个原因是为了让音频在空间位置转化的时候听起来不会出现明显的断层而使用。

七、讨论

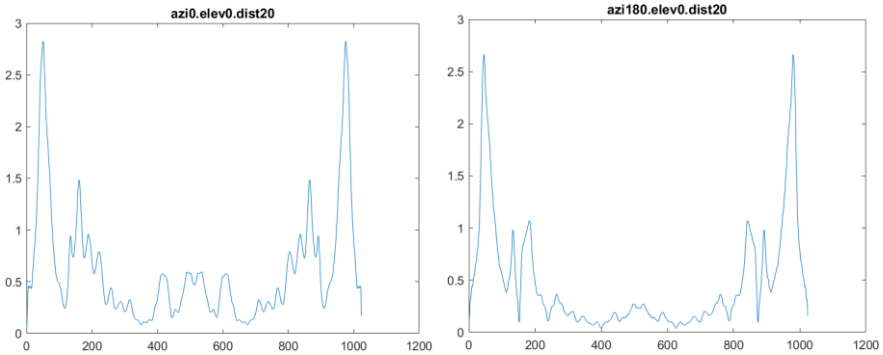
1. 帧间平滑处理时做的加窗操作中，不同的窗函数会不会影响最终输出信号？

   发现使用汉宁窗和海明窗时输出结果没有很大的区别，只有用矩形窗时有声音卡顿的

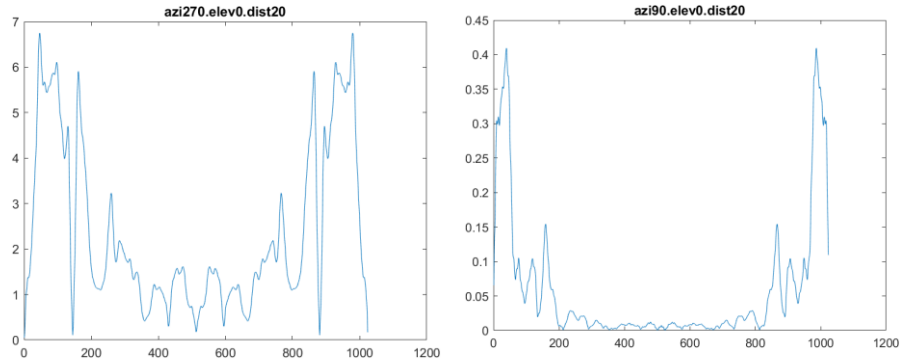
现象。虽然汉宁窗和海明窗的音效没有很大的区别，但具体看两个结果的频谱会有一定的区别，想知道应该使用什么窗函数比较合适。

2. HRTF 是不对称的函数

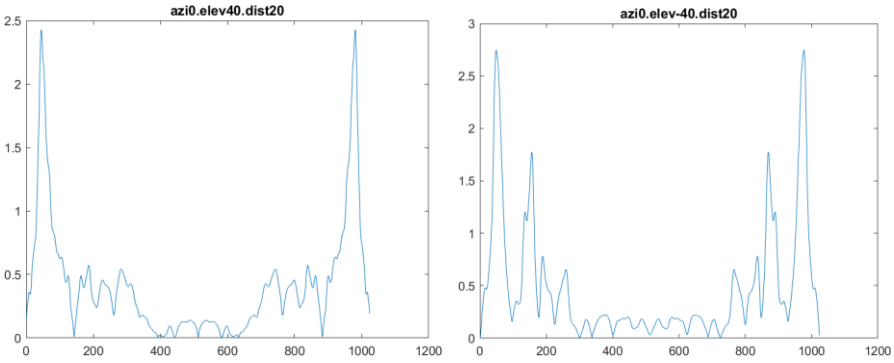
分析 HRTF 函数的频谱结果发现，无论在左右、前后还是上下方向其频谱架构都有差别。声源同侧的 HRTF 函数（左侧，图 3）的强度明显大于背侧 HRTF 函数（右侧，图 4）的强度。并且同侧的 HRTF 函数的波形也较声源背面的波形复杂，起伏变化剧烈，在高频部分能量也更大一些。这是由于人体对声音信号响应具有方向性所造成的。这些都是 HRTF 包含的重要方位信息。HRTF 中包含的这种有方向性的频率响应，使得它成为一个在各个方向上都不对称的函数。



图一和图二，分别是前后的 HRTF 函数



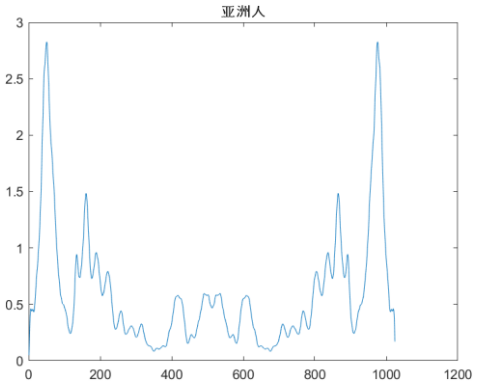
图三和图四，分别是左右的 HRTF 函数

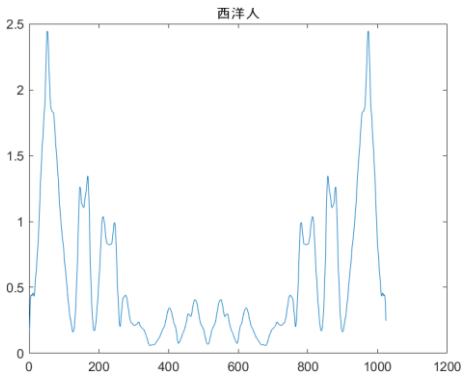


图五和图六，分别是上下的 HRTF 函数

3. 亚洲人和西方人的 HRTF

下面两个图分别是（声源在同一位置下）亚洲人的 HRTF 函数的频谱图和西方人的 HRTF 函数的频谱图。由此可以看出 HRTF 是由被测者特定的响应特性决定的，不同的人有不同的 HRTF。由于 HRTF 包含了个体结构对声音信号的响应，每个人身体结构不一样，对信号的频率响应也千差万别，因此每个人的 HRTF 都是不同的。研究表明，当听音者头部尺寸与测量 HRTF 函数使用的头部模型不相等时，前方范围内的声像位置畸变较小，但侧向的声像位置畸变较大。因而采用 HRTF 进行声源定位的时候听音者头部尺寸的不同是侧向声像位置畸变的重要原因。





4.

八、代码

```
function varargout = lab2(varargin)
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @lab2_OpeningFcn, ...
                   'gui_OutputFcn',  @lab2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function lab2_OpeningFcn(hObject, eventdata, handles, varargin)
% Choose default command line output for lab2
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

```
function varargout = lab2_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

% --- Executes on button press in select_button.
function select_button_Callback(hObject, eventdata, handles)
global audio;
global fs;
[file,~] = uigetfile();          % 选择音频样本
[audio,fs] = audioread(file);    % 读入音频信息

function [closest_loc,second_loc,dist_arr] = find_closest(azi,elev,dist)
prev_azi = azi;
prev_elev = elev;
prev_dist = dist;
dist_list = [20 30 40 50 75 100 130 160];

if dist < 20
    dist = 20;
end
if dist > 160
    dist = 160;
end
if elev < -40
    elev = -40;
end
if elev > 90
    elev = 90;
end
if azi < 0
    azi = 0;
end
if azi > 360
    azi = 360;
end

% 找最近位置的dist
for i = 1:7
    if dist >= dist_list(i) && dist <= dist_list(i+1)
        if dist < (dist_list(i+1) + dist_list(i))/2
            dist = dist_list(i);
        else
            dist = dist_list(i+1);
        end
        break;
    end
end
```

```
% 找最近位置的elev
elev = 10*round(elev/10);

% 找最近两个位置的azi
if elev >= -40 && elev <= 50
    azi = floor(azi/5)*5;
    second_azi = azi;
    if mod(round(azi),5) >= 3
        azi = azi+5;
    else
        second_azi = azi+5;
    end
elseif elev == 60
    azi = floor(azi/10)*10;
    second_azi = azi;
    if mod(round(azi),10) >= 5
        azi = azi+10;
    else
        second_azi = azi+10;
    end
elseif elev == 70
    azi = floor(azi/15)*15;
    second_azi = azi;
    if mod(round(azi),15) >= 8
        azi = azi+15;
    else
        second_azi = azi+15;
    end
elseif elev == 80
    azi = floor(azi/30)*30;
    second_azi = azi;
    if mod(floor(azi),30) >= 15
        azi = azi+30;
    else
        second_azi = azi+30;
    end
elseif elev == 90
    if azi <= 180
        azi = 0;
        second_azi = 360;
    else
        azi = 360;
        second_azi = 0;
    end
end
% 最近两个点的位置
closest_loc = [azi;elev;dist];
```

```
second_loc = [second_azi;elev;dist];
% 计算距离
rel_dist = relative_dist(prev_azi,prev_elev,prev_dist,azi,elev,dist);
sec_dist = relative_dist(prev_azi,prev_elev,prev_dist,second_azi,elev,dist);
dist_arr = [rel_dist,sec_dist];

% 计算距离
function res = relative_dist(azi1,elev1,dist1,azi2,elev2,dist2)
x1 = dist1*cos(azi1)*sin(elev1);
y1 = dist1*sin(azi1)*sin(elev1);
z1 = dist1*cos(elev1);
x2 = dist2*cos(azi2)*sin(elev2);
y2 = dist2*sin(azi2)*sin(elev2);
z2 = dist2*cos(elev2);
res = sqrt((x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2);

% --- Executes on button press in update_static_button.
function update_static_button_Callback(hObject, eventdata, handles)
global st_azi;              %
静态虚拟声音的azi（在GUI界面用户通过slider或editable text给的）
global st_elev;             % 静态虚拟声音的elev
global st_dist;             % 静态虚拟声音的dist

% 找与给定位置最近的HRTF数据库测量点位置
[st,~,~] = find_closest(st_azi,st_elev,st_dist);
st_azi = st(1);
st_elev = st(2);
st_dist = st(3);

% 修改slider位置和text值
set(handles.static_azi,'String',num2str(st_azi));
set(handles.azi_slider,'Value',st_azi);
set(handles.static_elev,'String',num2str(st_elev));
set(handles.elev_slider,'Value',st_elev+40);
set(handles.static_dist,'String',num2str(st_dist));
set(handles.dist_slider,'Value',st_dist-20);

% 静态虚拟声音合成

global output;
global audio;
tmp_audio = audio';
% 读取与给定位置对应的HRIR数据
lf = sprintf('PKU-IOA HRTF
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',st_dist,st_elev,st_azi,st_e
```

```
lev,st_dist);      % 左耳
rf = sprintf('PKU-IOA HRTF
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',st_dist,st_elev,360-
st_azi,st_elev,st_dist);  % 右耳
lfid = fopen(lf,'r');   l_hrir = fread(lfid,1024,'double');
rfid = fopen(rf,'r');   r_hrir = fread(rfid,1024,'double');
fft_lhrir = fft(l_hrir);    % 对HRIR数据进行fft变换
fft_rhrir = fft(r_hrir);

frame_len = 1024;               % 帧长
overlap = 512;                  % 重叠50%
w = hann(frame_len);            % 汉宁窗函数
s = frame_len - overlap;
m = mod(length(tmp_audio),s);
tmp_audio = [tmp_audio zeros(1,frame_len-m)];   %
为数据处理方便，在原信号上补零
len = length(tmp_audio);
output = zeros(2,len,'double'); % 对输出重新分配空间
for i = 0:len / s -2
    seg = tmp_audio((i*s+1):(i*s+frame_len))';  % 当前处理的帧
    seg = seg.*w;                               % 加窗
    % fft
    fft_tmp = fft(seg);
    % HRIR数据与声音信号卷积
    lconv = fft_tmp.*fft_lhrir;
    rconv = fft_tmp.*fft_rhrir;
    % ifft
    lifft = ifft(lconv);
    rifft = ifft(rconv);
    % 有overlap的帧合并
    output(1,1+i*s:(i*s+frame_len)) = output(1,1+i*s:(i*s+frame_len))+lifft';
    output(2,1+i*s:(i*s+frame_len)) = output(2,1+i*s:(i*s+frame_len))+rifft';
end

% --- Executes on button press in update_dynamic_button.
function update_dynamic_button_Callback(hObject, eventdata, handles)
global audio;
global output;
% 起点位置
global dy_s_azi;
global dy_s_elev;
global dy_s_dist;
% 终点位置
```

```
global dy_e_azi;
global dy_e_elev;
global dy_e_dist;

% 动态虚拟声音合成
tmp_audio = audio';
frame_len = 1024;                % 帧长
overlap = 512;                   % 重叠50%
w = hann(frame_len);             % 汉宁窗函数
s = frame_len - overlap;
m = mod(length(tmp_audio),s);
tmp_audio = [tmp_audio zeros(1,frame_len-m)];
len = length(tmp_audio);
n = len / s;
output = zeros(2,len,'double');

% 将运动轨迹分成n个点（n为帧的个数）
azi_list = dy_s_azi:(dy_e_azi-dy_s_azi)/(n-1):dy_e_azi;
elev_list = dy_s_elev:(dy_e_elev-dy_s_elev)/(n-1):dy_e_elev;
dist_list = dy_s_dist:(dy_e_dist-dy_s_dist)/(n-1):dy_e_dist;
loc_mat = [azi_list;elev_list;dist_list];

for i = 0:len/s-2
    % 计算与给定位置最近的两个HRTF数据库测量点位置以及距离
    [first_loc,second_loc,dist_arr] =
find_closest(loc_mat(1,i+1),loc_mat(2,i+1),loc_mat(3,i+1));
    % 权重是距离的反比
    w1 = 1/dist_arr(1);
    w2 = 1/dist_arr(2);
    first_lf = sprintf('PKU-IOA HRTF
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',first_loc(3),first_loc(2),f
irst_loc(1),first_loc(2),first_loc(3));
    first_rf = sprintf('PKU-IOA HRTF
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',first_loc(3),first_loc(2),3
60-first_loc(1),first_loc(2),first_loc(3));
    first_lfid = fopen(first_lf,'r');   first_lhrir =
fread(first_lfid,1024,'double');
    first_rfid = fopen(first_rf,'r');   first_rhrir =
fread(first_rfid,1024,'double');
    % 若在HRTF数据库中正好有间隔点的位置，不需要读取第二个最近点的数据
    if dist_arr(1) > 0
        second_lf = sprintf('PKU-IOA HRTF
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',second_loc(3),second_loc(2)
,second_loc(1),second_loc(2),second_loc(3));
        second_rf = sprintf('PKU-IOA HRTF
```

```
database\\dist%d\\elev%d\\azi%d_elev%d_dist%d.dat',second_loc(3),second_loc(2)
,360-second_loc(1),second_loc(2),second_loc(3));
        second_lfid = fopen(second_lf,'r');   second_lhrir =
fread(second_lfid,1024,'double');
        second_rfid = fopen(second_rf,'r');   second_rhrir =
fread(second_rfid,1024,'double');
        x_l = (w1*first_lhrir + w2*second_lhrir) / (w1 + w2);
        x_r = (w1*first_rhrir + w2*second_rhrir) / (w1 + w2);
    else
        x_l = first_lhrir;
        x_r = first_rhrir;

    end
    fclose('all');
    % 对HRIR数据fft
    fft_lhrir = fft(x_l);
    fft_rhrir = fft(x_r);
    % 当前处理的帧
    tmp = tmp_audio((i*s+1):(i*s+frame_len))';
    tmp = tmp.*w;                % 加窗
    fft_tmp = fft(tmp);          % 对声音信息做fft
    % 卷积
    lconv = fft_tmp.*fft_lhrir;
    rconv = fft_tmp.*fft_rhrir;
    % ifft
    lifft = ifft(lconv);
    rifft = ifft(rconv);
    % 合并
    output(1,1+i*s:(i*s+frame_len)) = output(1,1+i*s:(i*s+frame_len))+lifft';
    output(2,1+i*s:(i*s+frame_len)) = output(2,1+i*s:(i*s+frame_len))+rifft';

    set(handles.update_status,'String','1');
end

% --- Executes on button press in start_button.
function start_button_Callback(hObject, eventdata, handles)
global output;
global fs;
global player;
player = audioplayer(output,fs);
set(player,'TimerFcn',{@play_slider,handles});
set(player,'TimerPeriod',0.05);
player.play;
% audiowrite('result.wav',output',fs);
set(handles.update_status,'String','0');
```

```
function play_slider(hObject,~,handles)
CurrentSample = get(hObject,'CurrentSample');
len = get(hObject,'TotalSamples');
rate = CurrentSample/len;
set(handles.play_slider,'Value',rate);

% --- Executes on button press in pause_button.
function pause_button_Callback(hObject, eventdata, handles)
global player;
player.pause;

% --- Executes on slider movement.
function play_slider_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function play_slider_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



% --- Executes on slider movement.
function azi_slider_Callback(hObject, eventdata, handles)
global st_azi;
st_azi = get(hObject,'Value');
set(handles.static_azi,'String',num2str(st_azi));



% --- Executes during object creation, after setting all properties.
function azi_slider_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
set(hObject,'Value',0);

function static_azi_Callback(hObject, eventdata, handles)
global st_azi;
input = str2double(get(hObject,'String'));
if input <= 360 && input >= 0
    st_azi = input;
    set(handles.azi_slider,'Value',st_azi);
else
    msgbox('invalid value','warning','warn'); %error message window
end
```

```
% --- Executes during object creation, after setting all properties.
function static_azi_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global st_azi;
st_azi = 0;
set(hObject,'string','0');


% --- Executes on slider movement.
function elev_slider_Callback(hObject, eventdata, handles)
global st_elev;
st_elev = get(hObject,'Value')-40;
set(handles.static_elev,'String',num2str(st_elev));


% --- Executes during object creation, after setting all properties.
function elev_slider_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
set(hObject,'Value',0);

function static_elev_Callback(hObject, eventdata, handles)
global st_elev;
input = str2double(get(hObject,'String'));
if input <= 90 && input >= -40
    st_elev = input;
    set(handles.elev_slider,'Value',st_elev+40);
else
    msgbox('invalid value','warning','warn'); %error message window
end

% --- Executes during object creation, after setting all properties.
function static_elev_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global st_elev;
st_elev = -40;
set(hObject,'string','-40');


% --- Executes on slider movement.
```

```
function dist_slider_Callback(hObject, eventdata, handles)
global st_dist;
st_dist = get(hObject,'Value')+20;
set(handles.static_dist,'String',num2str(st_dist));


% --- Executes during object creation, after setting all properties.
function dist_slider_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
set(hObject,'Value',0);

function static_dist_Callback(hObject, eventdata, handles)
global st_dist;
input = str2double(get(hObject,'String'));
if input <= 160 && input >= 20
    st_dist = input;
    set(handles.dist_slider,'Value',st_dist-20);
else
    msgbox('invalid value','warning','warn'); %error message window
end


% --- Executes during object creation, after setting all properties.
function static_dist_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global st_dist;
st_dist = 20;
set(hObject,'string','20');


function start_azi_Callback(hObject, eventdata, handles)
global dy_s_azi;
input = str2double(get(hObject,'String'));
if input <= 360 && input >= 0
    dy_s_azi = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end


% --- Executes during object creation, after setting all properties.
function start_azi_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global dy_s_azi;
dy_s_azi = 0;
set(hObject,'string','0');


function start_elev_Callback(hObject, eventdata, handles)
global dy_s_elev;
input = str2double(get(hObject,'String'));
if input <= 90 && input >= -40
    dy_s_elev = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end



% --- Executes during object creation, after setting all properties.
function start_elev_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global dy_s_elev;
dy_s_elev = -40;
set(hObject,'string','-40');


function start_dist_Callback(hObject, eventdata, handles)
global dy_s_dist;
input = str2double(get(hObject,'String'));
if input <= 160 && input >= 20
    dy_s_dist = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end


% --- Executes during object creation, after setting all properties.
function start_dist_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global dy_s_dist;
```

```
dy_s_dist = 20;
set(h0bject,'string','20');


function end_azi_Callback(h0bject, eventdata, handles)
global dy_e_azi;
input = str2double(get(h0bject,'String'));
if input <= 360 && input >= 0
    dy_e_azi = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end


% --- Executes during object creation, after setting all properties.
function end_azi_CreateFcn(h0bject, eventdata, handles)
if ispc && isequal(get(h0bject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(h0bject,'BackgroundColor','white');
end
global dy_e_azi;
dy_e_azi = 180;
set(h0bject,'string','180');


function end_elev_Callback(h0bject, eventdata, handles)
global dy_e_elev;
input = str2double(get(h0bject,'String'));
if input <= 90 && input >= -40
    dy_e_elev = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end


% --- Executes during object creation, after setting all properties.
function end_elev_CreateFcn(h0bject, eventdata, handles)
if ispc && isequal(get(h0bject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(h0bject,'BackgroundColor','white');
end
global dy_e_elev;
dy_e_elev = 50;
set(h0bject,'string','50');


function end_dist_Callback(h0bject, eventdata, handles)
global dy_e_dist;
```

```
input = str2double(get(hObject,'String'));
if input <= 160 && input >= 20
    dy_e_dist = input;
else
    msgbox('invalid value','warning','warn'); %error message window
end



% --- Executes during object creation, after setting all properties.
function end_dist_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
global dy_e_dist;
dy_e_dist = 140;
set(hObject,'string','140');



function update_status_Callback(hObject, eventdata, handles)



% --- Executes during object creation, after setting all properties.
function update_status_CreateFcn(hObject, eventdata, handles)
% hObject    handle to update_status (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
set(hObject,'string','0');
```