Computational Perception
15-485/785

Assignment 1
Sound Localization

due: Thursday, Jan. 31

Introduction

This assignment focuses on sound localization. You will develop Matlab programs that synthesize sounds using ITD, IID, and HRTFs, compute the azimuth from the ITD, as well as think through problems in sound localization. If you are not familiar with Matlab or Fourier analysis, it is recommended that you work through the accompanying tutorial before starting this assignment.

Many of the problems ask for explanations of findings. Your answers to these questions should identify the key concepts with complete but concise explanations of the underlying phenomena. You should make sure to describe all important issues and cases that need to be considered. Answers that to not show understanding will not be given credit. As an example, we have provided some good and bad answers to a sample question.

In some of the questions you will be turning in Matlab code. The code should be commented and readable. The code will be evaluated based on wether it gives the correct answer to the problem. If you have any questions or problems, email me (`lewicki@cs.cmu.edu`).

All assignments should be submitted via email as a zipped attachment. Send it by email to `lewicki@cs.cmu.edu`. The zip file should consist of a single text document and a set of matlab files named as indicated in each problem. You only need to answer the questions listed under "*What to turn in*" for each problem.

All the files you will need to complete this assignment are available in the zip file `''hw1files.zip''` on the Blackboard page under Assignments. The file size is 2.8 MB due to the HRTF data you will need.

1 Perplexed by Duplex?

In class we discussed figure 1a, which shows the minimum audibly detectable change in angle of a sound (i.e. the minimum audible angle or MAA) versus that sound's frequency plotted for various azimuths. Figure 1b shows how interaural intensity differences (IIDs) change as one varies the azimuth and frequency of a test signal.

Example question and answer

e.g. (*10 points*). In figure 1a, the MAA increases (in general) with the deviation of the angle from 0 degrees. Explain this finding using the data plotted in figure 1b and your knowledge of sound propagation.

Wrong answer:

> Because at larger angles there is more variation across frequency.

Poor answer:

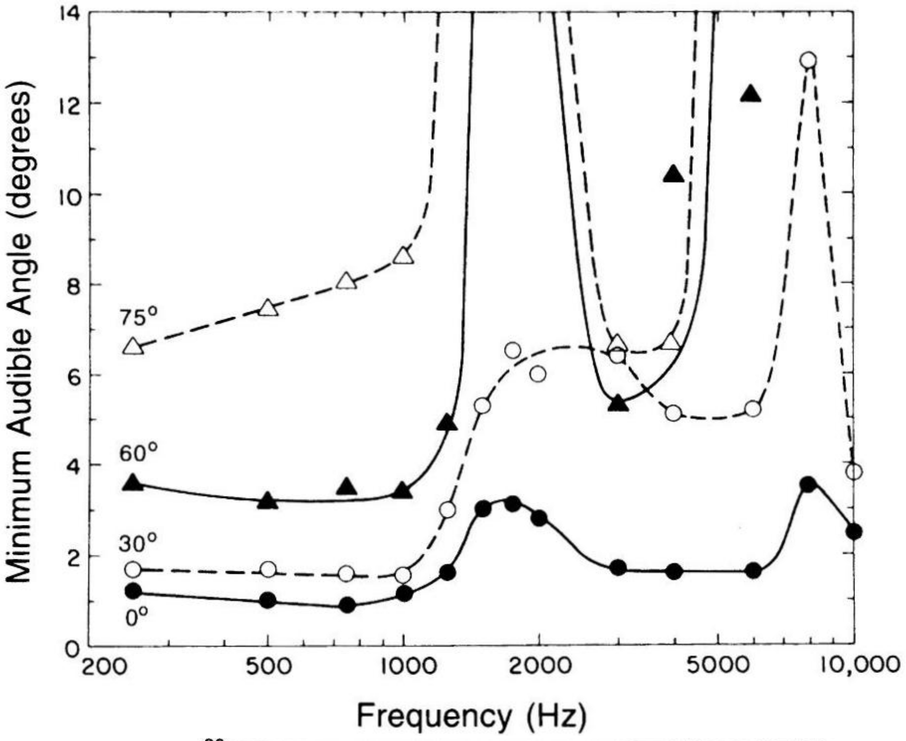
> Because the curves become more flat at the larger angles.

Good answer:

> In order to be able to discriminate changes in the sound position, there must be a perceptible difference in the IID of the two different positions. The curves plotted in figure 1b, have the greatest slope near zero across nearly all frequencies, which means the positions near zero degrees will be the most discriminable. As the azimuthal angle increases, the slopes decrease which means that the same change in the sound position becomes less discriminable thus increasing the MAA.

Note also that not all questions have clear answers and the true explanation could involve several factors. An excellent answer would not limit the answer to just IIDs, but also explain the other factors involved. It would also add more detail and observations:

> Note it is not always the case that the steepest slope is around 0 degrees. For example, these curves would predict that for 2500 Hz between 150 and 170 degrees would yield high MAAs. Also, these data apply to a single subject and the MAA curves for other subjects could be different.
>
> At lower frequencies ($<$ 900 Hz), the slope of the IID curves around zero become smaller, but are presumably still perceptible if the intensity difference is great than 1dB (the smallest detectable change). An additional acoustic cue that is used at lower frequencies is ITD. The slope of the ITD curve (shown in lecture) decreases slightly with increasing angle, but it is not clear whether this change is sufficient to explain the increase in MAA with increasing azimuthal angle at the lower frequencies. Another possible factor is experience, which would suggest that the reason we're more accurate at angles closer to zero is that we have more experience with sounds at that location.



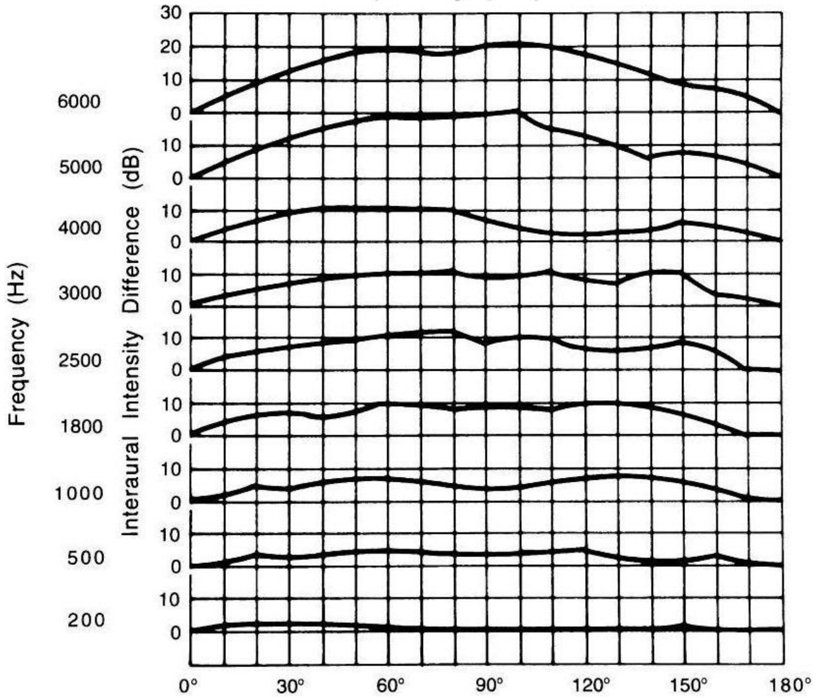


Figure 1: (a) Minimum audible angle vs. Frequency (ploted by azimuth); (b) Interaural Intensity Difference vs. Azimuth (plotted by frequency)

In grading this example, the wrong answer would receive no credit, and the poor answer might receive a point. How many points your answer receives depends on its correctness, thoroughness, and quality of explanation. Here the good answer is correct and well-explained, but it was limited to IIDs, so it might receive 6-8 points, depending on the completeness of the answer. The combined good and excellent answers would receive full credit.

Questions

1. (*15 points*) Describe three features of figure 1 which would be predicted by duplex theory.
2. (*5 points*) Describe a feature of the curves is not explained by duplex theory. What might be the source of this feature?

2 Bend me your ear

Binaural features such as IID or interaural time difference (ITD) are calculated in the brain of mammals. Before sound signals ever reach a neuron, though, they pass the pinna or outer ear. What is the functional significance of the pinna?

Select the best answer: (5 points)

a The pinna reflects sounds waves from over its concave surface into the ear cannal. This process acts to amplify the signals over biologically important frequency ranges.

b Sounds resonate within the pinna (even before it reaches the ear cannal.) The spectral profile of these resonances is used for distance and source angle judgements.

c The pinna is too small compared to behiorally meaningful wavelengths to meaningfully affect sounds, particulary sounds in the frequency range of human speech. Its priniple role is likely just to protect the ear cannal and tympanic membrane from physical trauma.

d The function of the pinna is revealed by studying its impulse response in the time domain. This analysis demonstrates that it works to generate interference patters via reflections. These patterns are used for many localization tasks.

3 Interaural Time Differences

Note: This and future problems will contain some questions in the text to get you to think about how you would explain your own perceptions. You are not expected to write answers for these, but you are expected to be able to answer them, perhaps on future exams. For the

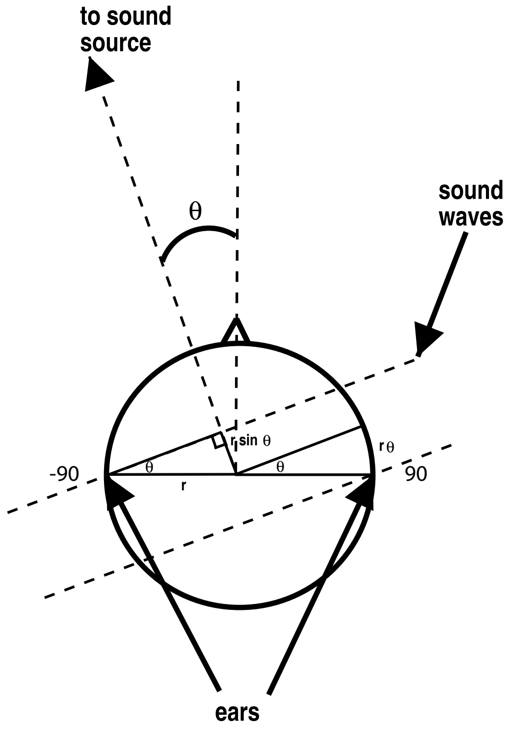


Figure 2: Azimuth conventions

assignments, you are only expected to turn in answers for questions that are enumerated and have assigned point values. Also, as this problem involves writing code, it will be graded primarily on whether it produces the correct output on test cases. Code that does not produce the correct output will receive little or no credit.

As you may have noticed from the handouts given in class, many experiments in sound localization (or lateralization) use sinusoidal signals. In this problem you will create a set of such signals and "play" a bit with them. You need to use headphones (or earphones) for this, so either use your own or borrow some from a friend. The quality is not that important.

We will define that locations in front of the head have 0° azimuth and locations in front of the left ear have -90° azimuth. Assume a simplified model of the human head in which the head is spherical. Also assume that sound sources are infinitely far away, so that sound reaches the ears in straight lines (see figure 2). With this model the ITD (Δt) is:

$$\Delta t = \frac{r}{c}(\theta + \sin(\theta)) \tag{1}$$

where $r = 9$ cm is the radius of the head and $c = 345$ m/s is the speed of sound (at 23 °C, if you were to step outside to our -10 °C degree weather of late, the speed of sound would slow to 325 m/s).

Write a Matlab function that takes an angle in degrees as input and creates a stereo sound composed of both the left and right signals that reach the ears of a listener. Your function should create a signal y, such that y=[left signal; right signal] and where the left and right signals are sine waves of the same frequency displaced by some amount relative to each other.

Begin by calculating the ITD of a sound that comes from angle theta. You will need to convert this time difference into a samples difference. You should use the sampling rate for that. (When choosing a sampling rate you should guarantee that the frequencies you want to work with can be represented by that sampling rate. The highest frequency that can be represented by a given sampling rate is called the Nyquist frequency and it is equal to half the sampling rate. For this homework we will use a sampling rate of 44.1 kHz, the standard for CD recordings.) Then you should create a function that given the signal that reaches one ear computes the signal that reaches the other ear. In this simplified example, the signal that reaches the other ear is just a copy of the first signal shifted by a given delay.

You can test your function with the provided sounds `sound.wav` and `note.wav` which can be loaded with the Matlab command `wavread`. You should also create a sine wave using the following code (also in the provided file `sineWave.m`):

```
function wave = sineWave(hz,d,fs)
%input:
% hz - frequency in Hz of output signal
% d - duration of signal in seconds
% fs - sampling rate
%
%output:
% wave - vector of duration d containing a sine wave
% of frequency hz sampled with sampling rate fs.
% The vector has size d*fs.

step = 1/fs;
t = 0:step:d; %time steps
wave = sin(t*hz*2*pi);
```

If you play a signal created by the previous function, you will probably notice onset and offset transients. Sharp onsets and offsets can play a role in sound localization (and lateralization), but here we do not want to have that effect into account. You should change the signals so that no sharp onsets or offsets are perceived. One way of removing them is to ramp the amplitude of the signal up at the beginning and down at the end using a function that smoothly goes from 0 to 1 such as a Hanning window. The Matlab function `hann(n)` is provided in the Matlab signal processing toolbox. Write a function called `ramp` that modifies the output of the function `sinWave` so that it produces signals with gradual onsets and offsets. The function should create a 10 msec Hanning window (for example, `hann(441)` will create a 10 msec hanning window if your sampling rate is 44.1 kHz.) Apply the first (rising) half of the Hanning window to the first 5 msec of your signal (using point by point multiplication.) Do the same using the second half of the Hanning window and the end of your signal. Make sure that each half of the ramp goes from zero to one. It's a good idea to plot it using different values of `n` so you can see the shape.

Using the functions you created above, create a set of signals with a given frequency that come from different angles (for instance, you can create signals of 200Hz that go from 0 to 90). Also, using the same or different angles, create sounds of different frequencies (use at least 200Hz and 1500Hz).

Show the output of your function by plotting the pure tone sounds with the left and right channels in different colors on the same axes. You can use the Matlab function `hold` to plot both signals in the same axis. Make the x-axis units in μsecs, so that the relative alignment of the left and right channels are easy to see.

Listen to those sounds with headphones. The Matlab function `soundsc(signal,fs)` can be used to play the sounds. Do you perceive the sounds as coming from the same location or different locations?

Now try the 'note.wav' and 'sound.wav' sounds. Load them into your funciton and listen to the output for different azimuths. How does the perception of spatial position of the sine waves compare to the sine waves?

For the curious, try generating a narrow band sound spanning 1400 to 1600 Hz. How does your perception of this sound compare to the 1500 Hz pure tone?

What you should turn in:

1. (*10 points*) Submit your code for generating stereo signals. The main function should be called `createITD` (see below for example.) It will take three input arguements: an angle in degrees, a time in seconds and a frequency in Hz. It will create a pure tone (sine wave) of length n samples based on the input frequency with its onset and offset modulated by a Hanning window. It should output a 2 x n matrix, y, representing a stereo sound signal such that $y =$ [left signal; right signal]. In other words, the left and right signals are sine waves of the input frequency displaced by the amount relative to each other appropriate for the input angle.

```
function y = createITD(theta,d,hz)
%input:
% theta - angle in degrees
% d - duration of signal in seconds
% hz - frequency in Hz
%
%output:
% y - double row vectors (i.e. a 2 x n matrix) of
% duration d containing sine waves of hz Hz sampled
% with a sampling rate of 44.1 kHz.
% The vector has size n = d*44100.
```

2. (*10 points*) In addition to returning the vector y, the function should produce a plot of the left and right channels overlaid in different colors on the same axes. In order to clearly see the phase difference, the x-axis should be plotted units in μsecs and range from -800 μsecs to +800 μsecs. Also make sure that the signals are centered on the plot, so that you're not just plottting the initial ramp.

4 Computing ITDs

In the previous problem, you synthesized sounds using a given ITD; now you will go the other direction. You will write a Matlab function which, given a stereo sound as input, determines the ITD, Δt, and then inverts the model in equation (1) to estimate θ.

Given a sound, you should estimate the interaural time delay Δt by looking for peaks in the cross correlation between the left and right channels. The Matlab function `xcorr(x,y)` computes the cross correlation of x and y. The function `[m,i]=max(v)` returns the maximum m and its index i from the vector v. The IDT should be in seconds, so do not forget to convert the number of samples by which one channel leads the other to the Δt in seconds.

Since the function for Δt in terms of θ is not algebraically reducible to an analytic expression for θ in terms of Δt, you will need to numerically estimate this function. You can do this, for instance, by developing a lookup table taking different values of Δt to θ, by iteratively searching for θ by repeatedly computing $\Delta t(\theta)$ for some estimate of θ and using the result to improve your estimate, or by iteratively solving the function with the Matlab function `fsolve`.

Using the sounds you generated in the previous problem, check that your function computes (approximately) the same angles that you used as parameters. Are there instances when it is wrong? What do you think happens in these cases?

What you should turn in:

1. (*20 points)* Write a Matlab function that computes ITD. The funcition should be named `computeITD`. It will take a stereo sound as input and produce the appropriate azmimuth, θ, as an output (see below for example.)

```
function theta = computeITD(y)

%theta = computeITD(y)
%input:
% y - double row vectors containing a stereo
% waveform sampled at a rate of 44.1 kHz.
%
%output:
```

```
% theta - angle in degrees
```

5 Interaural intensity differences

If a sound is to the right or left of the midline, the head will "shadow" high frequencies on their way to the more distant ear. Low frequencies are less affected. This means the sound will appear louder to the ear nearer the source, but the difference in sound level will be frequency dependent. A simple model of this interaural intensity difference (IID) (which can be worked out analytically under the assumption that the head is a solid sphere) is expressed as a pair of transfer functions that specify, for any angle theta, how much each frequency s is boosted or attenuated by head shadowing:

$$H_L(s,\theta) = \frac{(1+\cos(\theta+\pi/2))s+\beta}{s+\beta}; \quad H_R(s,\theta) = \frac{(1+\cos(\theta-\pi/2))s+\beta}{s+\beta},$$

where $\beta = 2c/r$. The function $H_L(s,\theta)$ is for the left ear, and $H_R(s,\theta)$ is for the right ear. We can use these functions to derive frequency domain filters indexed by Fourier number k (instead of frequency s) by recalling that $s = k/(N\delta)$ (where δ=1/44100 sec. is the interval between samples, and N is the number of samples in the sound to which we're applying the filter). Call these filters $H_L(\theta)$ and $H_R(\theta)$. Now, given a sound $x(t)$ (in time domain) at angle θ with Fourier coefficients $a(k) = \mathcal{F}(x)$, we can apply the filters to obtain left and right channel signals:

$$a_L = aH_L(\theta); \quad a_R = aH_R(\theta).$$

You will write a Matlab function that, given a monaural sound (i.e., just one channel rather than stereo) and an angle, creates the left and right channel signals. You should use the two filters, H_L and H_R, creating the left and right channel signals by filtering the input signal with each filter respectively. So, the left and right channel signals should be similar to the inputs signal with their relative intensity determined (in a frequency dependent manner) by the equations above. Refer to the Frequency domain and Fourier transform tutorial for how to filter signals using Matlab.

Test your function with a sine wave or the signals provided for problem 3. It is a good idea to plot the resulting left and right channel signals so that you can see the effect of applying the filters to the original signal.

What you should turn in:

1. (*20 points*) Write code for a function named `createIID`. It will take two input arguments: an angle in degrees and a monaural waveform. It should output a 2 x n vector, y, representing a stereo sound signal such that y = [left signal; right signal].

```
function y = createIID(theta, x, fs)
```

```
%y = createIID(theta,x,fs)
%input:
% theta - angle in degrees
% x - the monaural input waveform
% fs - sampling frequency in Hz
%
%output:
% y - stereo output waveform
```

2. (*5 points*) Using the above code, produce a plot of the IID cuves arrayed like those in figure 1b but using the idealized model you just coded. To make a similar looking graph, you should plot all the curves on the same figure, offseting each curve by a sufficient amount. Make sure that each curve has is plotted using the same vertical scale, so that they can be compared.

3. (*5 points*) What significant feature of the empirically derived IID curves is missing in the curves produced by the idealized model? What is the reason for this difference?

6 3D sound localization

The transfer functions you used in the previous question were derived analytically assuming a spherical head, which is a crude approximation to the actual filtering caused by the head, body, and pinna. An alternative method of deriving the transfer functions is to measure them. The files

```
horiz_hrir_l.mat
horiz_hrir_r.mat
```

contain the left and right head-related transfer functions (HRTFs, also known as head-related impulse response functions or HRIRs) for 45 different subjects (one of which is a mannequin) at various points around the horizontal plane. You can read them into Matlab with the command `load`. The azimuths used are:

$$[-80, -55, -25, 0, 25, 55, 80, 100, 125, 155, 180, 205, 235, 260]$$

This starts from a little in front of the left ear (direct left and right was not sampled in this database) and going around toward the front and then behind the head. Each impulse response function is 200 samples long. The format of the (3D) array is

```
horiz_hrir_l(subject,azidx,1:200)
```

You can select a particular impulse response function using

```
hl = squeeze(horiz_hrir_l(i,j,:));
```

You can test localization in elevation using the files

```
overhead_hrir_l.mat
overhead_hrir_r.mat
```

The elevation in these files is 90° and the azimuths used are:

$$[-80, -55, -25, 0, 25, 55, 80]$$

i.e. they start from the left and sweep overhead to the right.

Familiarize yourself with the (time domain) transfer functions by plotting them using common axis limits so they can be more easily compared. The Matlab function `pause` is helpful for pausing a loop that plots the different functions. Try to see if you can observe any relationships between the properties of the curves and their corresponding spatial positions.

Write a function that takes a (non-stereo) sound and the left and right HRTFs and returns the sound as it appears at the left and right ears. Select a pair of HRTFs and listen to the two provided sounds using headphones or earphones. Does your percept match the direction of the given parameters? Think about why it might not.

You should also try writing a few simple for loops to generate a series of virtual sounds. For example, an auditory stimulus that sweeps all the way around the head in the horizontal or vertical plane using the locations specified above. Note that playing the sounds one at a time can generate audio artifacts, so it is usually better to generate a single stimulus by concatenating the results into one long stereo sound. Are the sounds where the ”should” be? You could also try listen to a series of sounds that loops through the subjects, in effect listening “through their ears”. Are some more accurate than others? Do you experience externalization?

What you should turn in:

1. (*15 points*) Sumbit a function called `applyHRTF`. It will take four inputs: a sound vector (non-stereo), a subject number, an azimuth and an elevation. It should also check to make sure that the inputs are valid and return and error if not. Do not include the HRTFs or sound files with your code.

```
function y = applyHRTF(x,subject,theta,phi)

%applyHRTF(x,theta,phi)
%
%input:
% x - a monaural sound vector
% subject - subject number
% theta - an azimuth angle in degrees
% phi - a horizatal angle in degrees
```

```
    %output:
    % y - the stereo sound
```

2. (*10 points*) Give an outline of an algorithm that would allow you to determine that spatial position of a white noise sound filtered by one of the HRTFs. You do not need to write any code.

HRTF Database

If you are interested in exploring the database used in this problem further, the website is

`http://interface.cipic.ucdavis.edu/CIL_html/CIL_HRTF_database.htm`