Signals and Communication Lab
DII University of Brescia - ITALY

# Digital Audio Processing

## Lab.3 - Psychoacoustics Experiments - 2014

*Alessio Degani,* [alessio.degani@ing.unibs.it]
http://www.ing.unibs.it/alessio.degani

## 1 Pure tones masked by narrow-band noise

Load the wave file *track_10.wav* and play it. This file is taken from the CD-ROM of the book **Psychoacoustics - Facts and Models, E. Zwicker and H. Fastl, Springer**. Description of this track (from the aforementioned book): *In this demonstration the masked threshold of pure tones masked by critical-band wide noise (1 kHz, 70 dB) is illustrated. You will hear three series of tone triplets: the first series is played at a level of 75 dB, the second at a level of 60 dB, the third at a level of 40 dB. Each series consists of six tone triplets with the frequencies 600 Hz, 800 Hz, 1000 Hz, 1300 Hz, 1700 Hz, and 2300 Hz. In the second series the third tone triplet at 1000 Hz is masked by the narrow-band noise, and in the third series the third and fourth triplet at 1000 Hz and 1300 Hz (for some persons also the fifth triplet at 1700 Hz) are masked.*

## 2 Beats reloaded: Binaural Beats

As we have seen in the Lab.1 - section 3, a phenomenon named *Beats* occurs when two tones at slightly different frequency are mixed together. A very similar behaviour happens when the sounds are mixed by our brain. Two separate tones, one in the left ear and the other one in the right ear, are in some way mixed by our brain. The difference between sounds picked up by our ears is an important feature for the 3D perception of sound and source localization. Synthesize a 10 seconds stereo sound in this way:

$$b_{left}(t) = \sin(2\pi f_0 t),\; b_{right}(t) = \sin(2\pi f_0 t + \pi t^2)$$

and joint them together to create a stereo sound (in Matlab: $b = [b_{left}\ ;\ b_{right}]'$). Note that the " $'$ " is the Matlab operator for the *matrix transposition* and $b_{right}$ is a *linear frequency sweep.*

## 3 Virtual Pitch: Missing Fundamental

Our perception of pitch is robust to a certain degree of signal degradation. In a complex tone with fundamental frequency $f_0$, his harmonics help us to identify the main pitch. However, if we filter out the fundamental frequency of this tone, we can still perceive his main pitch. Synthesize a complex tone with 20 harmonics in this way:

$$c_1(t) = \sum_i \frac{1}{i} \sin(2\pi i f_0 t),\; i = [1 \dots 20]$$

Now synthesize $c_2$ with $i = [3 \ldots 20]$ (without fundamental and first harmonic) and listen to the differences. For best results try $f_0 = 120$ Hz.

## 4 The Endless Fall: Shepard-Risset glissando

A Shepard tone, named after Roger Shepard, is a sound consisting of a superposition of sine waves separated by octaves. When played with the base pitch of the tone moving upward or downward, it is referred to as the Shepard scale (or glissando in the case the steps between the tones are continuous). This creates the auditory illusion of a tone that continually ascends or descends in pitch, yet which ultimately seems to get no higher or lower. The Shepard scale is the acoustic equivalent to the Penrose scale optical illusion (Figure 1).

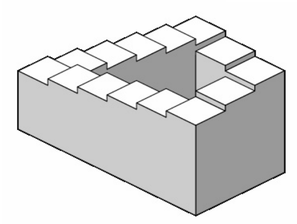

Figure 1: Penrose Stairs optical illusion - L. Penrose (1898-1972)

Open and play the files *shepard.wav*, *rising_melody.wav* and *echoes.wav* (Pink Floyd, 1971) to ear the various facets of Shepard scale. Now, try to synthesize a Shepard downward glissando. This kind of glissando, in his basic form, is formed by 4 tones, separate by an octave ($f_0$, $2f_0$, $4f_0$, $8f_0$), that linear decreases in frequency. Each tone amplitude is scaled by a factor that change in time.
Let $l = 10$ the length of glissando in seconds and $f_0 = 440$:

$$s(t) = \sum_{i=0}^{3} a_i(t) \sin(2\pi 2^i f_0 t - \pi \frac{2^{i-1} f_0}{l} t^2)$$

where

$$\begin{aligned}
a_0(t) &= [0.5 \ldots 0] \text{ in } l \text{ seconds} \\
a_1(t) &= [1 \ldots 0.5] \text{ in } l \text{ seconds} \\
a_2(t) &= [0.5 \ldots 1] \text{ in } l \text{ seconds} \\
a_3(t) &= [0 \ldots 0.5] \text{ in } l \text{ seconds}
\end{aligned}$$

**TIPS:** the $a_i$ arrays can be generated in Matlab with the command *linspace*. Now you have a basic Shepard element. You can concatenate $s(t)$ any times you want to hear the endless falling glissando (eg, for 2-times repetition in Matlab: $shepard = [s(1 : end - 1)\ s(1 : end - 1)]$).