Sampling Rate Conversion

ELEC-E5620 - Audio Signal Processing, Lecture #9

Vesa Välimäki

Sound check

March 24, 2017

DEMO

Plate Reverb

by Zhicun Xu & Aleksi Ö
## Course Schedule in 2017

<table>
<thead>
<tr>
<th>Course</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>0. General issues (Vesa &amp; Fabian)</td>
<td>13.1.2017</td>
</tr>
<tr>
<td>1. History and future of audio DSP (Vesa)</td>
<td>20.1.2017</td>
</tr>
<tr>
<td>2. Digital filters in audio (Vesa)</td>
<td>27.1.2017</td>
</tr>
<tr>
<td>3. Audio filter design (Vesa)</td>
<td>3.2.2017</td>
</tr>
<tr>
<td>4. Analysis of audio signals (Vesa)</td>
<td>10.2.2017</td>
</tr>
<tr>
<td>No lecture (Evaluation week for Period III)</td>
<td>17.2.2017</td>
</tr>
<tr>
<td>5. Audio effects processing (Fabian)</td>
<td>24.2.2017</td>
</tr>
<tr>
<td>6. Synthesis of audio signals (Fabian)</td>
<td>3.3.2017</td>
</tr>
<tr>
<td>7. 3-D sound (Prof. Ville Pulkki)</td>
<td>10.3.2017</td>
</tr>
<tr>
<td>8. Physics-based sound synthesis (Vesa)</td>
<td>17.3.2017</td>
</tr>
<tr>
<td>9. Sampling rate conversion (Vesa)</td>
<td>24.3.2017</td>
</tr>
<tr>
<td>10. Audio coding (Vesa)</td>
<td>31.3.2017</td>
</tr>
</tbody>
</table>

## Outline

- **Sampling rate conversion methods for audio signals**
  - Audio sampling rates & conversion ratios
  - Basic operations: decimation & interpolation
  - Synchronous vs. asynchronous conversion
    - Integer and rational conversion ratios (e.g., 1/4 or 2/11)
    - Incommensurate ratios (e.g., 44.1/48)
  - Polyphase filters
  - Multistage methods
  - Smith-Gossett algorithm
- **Two demos:** Plate reverb modeling, time-stretching
Audio Sampling Rates

- **44.1 kHz**: “Consumer” sampling rate
  - Sampling rate of CD standard (*Compact Disk*)
    - Optimum sample rate for the European video recorders would be 44.05594 kHz, but the simpler figure 44.1 kHz was chosen
  - MiniDisk and DAT (*Digital Audio Tape*) players

- **48 kHz**: “Professional audio” sample rate
  - Chosen to be compatible with common video standards
  - Used in recording studios for digital mastering on magnetic tape
  - DAT players
  - European digital radio DAB (*Digital Audio Broadcasting*)
  - Soundtrack in digital video, such as DVD (Dolby AC-3, MPEG-1 & 2, DTS)
  - Mobile phones, portable audio devices, such as music players and games

More Audio Sampling Rates

- **32 kHz**: “Digital radio” sample rate
  - Signal bandwidth 15 kHz — the same as in the FM radio
  - BBC and EBU (European Broadcasting Union) standard (4 X 8 kHz)
  - One option in DAT recorders
  - NICAM digital TV sound (14 bits/sample)
  - Digital sound on video-8 cassettes
  - Originally suggested for DAB, but not in use

- **24 kHz**: “Half-rate”
  - European digital radio DAB

- **22.05 kHz**: 44.1 kHz / 2
  - PC sound effects, computer game sounds

- **11.025 kHz**: 44.1 kHz / 4
  - PC sound effects, computer game sounds
Still More Sampling Rates

- **8 kHz**: “Toll quality”
  - Digital telephone signals, e.g., GSM mobile phones
  - Not really for audio (narrowband, not wideband)
- **44.1-kHz family**: **88.2 kHz, 176.4 kHz**
  - DVD-Audio disks
  - MPEG-2 AAC standard
- **48-kHz family**: **96 kHz, 192 kHz**
  - DVD-Audio disks
  - MPEG-2 AAC standard
  - 96 kHz has become popular in the recording industry
    - 96 kHz and 24 bits!

Strange Sampling Rates

- Some systems have their own, strange sample rates
- For example, the acoustic measurement software MLSSA gives you $f_s = 75.47$ kHz
  - when desired bandwidth is 25 kHz and an 8-pole Chebyshev anti-aliasing filter is selected
  - Also 14.98 kHz, 20.00 kHz, 30.08 kHz, 45.98 kHz, 60.61 kHz, 80.00 kHz, 90.01 kHz, 117.6 kHz, and 160.0 kHz are available (that is, just happen)
- An ultrasonic measurement system UltraSoundGate uses 50 kHz, 62.5 kHz, 75 kHz, 100 kHz, 125 kHz, 150 kHz, 166.6 kHz, 187.5 kHz, 200 kHz, 214 kHz, 250 kHz, 300 kHz, 333.3 kHz, 375 kHz, 400 kHz, 500 kHz, 750 kHz, and **1 MHz**!
What is a Sampling Rate Converter?

• Device or software that changes the sample rate (or interval)
  – Accepts a digital signal at one sampling rate as input
  – Outputs a digital signal at a different sampling rate

If the sample rate is lowered, signal bandwidth is reduced to avoid aliasing.

If the sample rate is increased, signal bandwidth is not changed (oversampling).

Changes in signal quality should not be too severe
  – Allowing the largest acceptable error to occur helps in reducing the computational cost
Basic Operations

- Two basic operations in sample rate conversions:
  1) Decimation
  2) Interpolation
- **Decimation** = Lower the sampling rate
  - Get rid of samples
  - Also called, e.g., downsampling, subsampling, or sampling rate compression (that last one is not recommended in this context!)
- **Interpolation** = Increase the sampling rate
  - Invent new samples between original ones
  - Also called, e.g., upsampling or sampling rate expansion
- **Resampling** may refer to any of the two operations
  - Possibly time-varying sampling rate conversion

Decimation by Integer Factor $M$

- **Lowpass filter** the signal & **retain every $M$th sample**
  - The rest of the samples are disregarded
- The lowpass filtering is done to avoid aliasing
  - Remove everything above the new Nyquist limit $f_s/2M$ **BEFORE** disregarding the $M - 1$ samples
  - Remember the Shannon sampling theorem:
    “The sampling frequency must be at least 2 times that of the highest frequency present in the signal”
Frequency-Domain View of Decimation

\[ M = 2 \]

\[ f_s' = \frac{f_s}{2} \]

Incorrect and Correct Decimation by \( M \)

1) Original signal \((f_s = 44.1 \text{ kHz})\)
2) The wrong way: decimate without lowpass filtering \((f_s/M = 8820 \text{ Hz})\)
   - Lowpass filter was forgotten, only decimate by 5
3) Hear only the aliased signal \((f_s/M = 8820 \text{ Hz})\)
   - Highpass filter (cutoff at 4 kHz), then decimate by 5
4) Correctly decimated by factor 5 \((f_s/M = 8820 \text{ Hz})\)
   - Lowpass filter (cutoff at 4 kHz), then decimate by 5
Interpolation by Integer Factor $L$

- Basic idea: Insert $L – 1$ zeros between each sample & smooth the signal with a lowpass filter.
- The lowpass filter suppresses image frequencies.
  - Otherwise images of the baseband signal will be present above the original Nyquist limit!
- Note that the signal must be amplified by factor $L$.
  - When signal samples are destroyed, the signal power gets smaller.
  - Gain $L$ be embedded in the passband gain of the lowpass filter.

Frequency-Domain View of Interpolation

- $L = 2$
- $f'_s = 2f_s$
- Lowpass filtering
- $f'_s$ = 2$f_s$
Incorrect and Correct Interpolation by $L$

1) Original signal ($f_s = 11.025$ kHz)
2) **The wrong way**: expand by $L = 4$ without lowpass filtering ($Lf_s = 44.1$ kHz)
   - 3 zeros inserted between each original sample
   - Lowpass filter was forgotten
3) Hear the aliased components ($Lf_s = 44.1$ kHz)
   - Insert zeros and highpass filter (cutoff 5.2 kHz)
4) **Correctly interpolated** by factor 4 ($Lf_s = 44.1$ kHz)
   - Zeros inserted and then lowpass filtered with cutoff at 5 kHz
   - Also multiply by $L$ to restore signal amplitude

Avoid Extra Operations

1. Don’t compute the samples that will be disregarded
2. Don’t multiply by zero
   - In decimation, only 1 out of $M$ output samples is needed
     - Just skip those $M - 1$ filter computations
   - In interpolation, $L - 1$ samples out of $L$ are zero
     - Just skip those filter coefficients
   - A polyphase filter implements both ideas
Polyphase Filter

- Design a prototype filter: long FIR filter with a sinc-type impulse response
- Pick every $L$th sample to form **polyphase filter branches**
  - First start at the last coefficient, then at the 2nd last, etc.
  - You will obtain $L$ fractional delay filters
- Used one after the other for decimation or interpolation

![Prototype FIR filter](image)

Polyphase Interpolation Filter

- Each filter branch computes every $L$th sample
  - Zero samples are not processed at all (not inserted, not filtered)
  - Output samples are picked at a rate $Lf_s$
Many Faces of Interpolation

• Interpolation may mean one of several operations
  1) Find a value between known sample values
     • Basic definition used in mathematics
  2) Upsampling: calculate and insert \( L - 1 \) new samples
     between each two original samples
     • In the context of sampling rate conversion in DSP
  3) Calculate one signal value at a non-integer point between
     sample points
     • Fractional-length delay line in audio signal processing
     • Fixed or time-varying

Integer Conversion Ratios in Audio

• Integer ratios are rare in digital audio
• **DAT-to-telephone**: 8000/48000 = 1/6
  – Decimation by factor \( M = 6 \)
• Conversion between multiples of 44.1 kHz and 48 kHz
  – E.g. 88.2 kHz \( \rightarrow \) 44.1 kHz or 96 kHz \( \rightarrow \) 48 kHz
• And that’s it!
Rational Ratios Are Common in Audio

- **CD-to-DAT**: 48000/44100 = 160/147 ≈ 1.08844
  - A very common but difficult case
  - Interpolation by $L = 160$ followed by decimation by $M = 147$
  - Very close to 48000/44000 = 12/11 = 1.09090909... (Error of 0.2%)

- **Master-to-CD**: 44100/48000 = 147/160 = 0.91875

- **CD-to-telephone**: 8000/44100 = 80/441 ≈ 0.181406
  - Very close to 8000/44000 = 2/11 = 0.18181818... (error of 0.2%)

- Typically ratios of large integer numbers
  - Ratios of large integers may well be approximated with ratios of smaller integers

Rational Sampling Rate Conversion

- Sampling rate conversion by a rational factor $L/M$
- First upsampling by $L$, then downsampling by $M$
  - In reverse order, the signal bandwidth would be reduced too much
- One lowpass filter is sufficient (cutoff $\approx 0.9 \times Lf_s/2M$)
- Be clever: calculate only the samples to be retained
  - The rest will be disregarded anyway, so they are not calculated at all
Polyphase Implementation

- Conversion by \( L/M \) can be implemented with a polyphase filter
  - Commuter skips the output samples that are not needed
- Prototype lowpass filter designed to avoid aliasing
- Interpolation ratio \( L \) determines number of branches
- Decimation ratio \( M \) determines how many branches are skipped at the output

\[
\begin{align*}
\text{Commuter} & \quad H_0(z) \quad f_s \quad \frac{L}{M} \quad H_{L}(z) \\
& \quad H_1(z) \\
& \quad H_2(z) \\
& \quad \vdots \\
& \quad H_{L-1}(z)
\end{align*}
\]

Multistage Structures

- Interpolation with a large factor \( L \) can be efficiently implemented using a multistage structure
  - For ex. \( L = 16 = 2 \times 2 \times 2 \times 2 \)
- The same idea applies to decimation with a large factor \( M \)
- Efficiency comes from the fact that all filters have a low order
  - Filter design gets easier when the passband and transition band get wider

\[
\begin{align*}
\uparrow 16 & = \uparrow 2 \quad \uparrow 2 \quad \uparrow 2 \quad \uparrow 2
\end{align*}
\]
Comparison of Multistage Structures

- Four configurations to resample from 44.1 kHz to 48 kHz:
  1. Direct: $L = 160$ and $M = 147$
  2. Full M-stage (only primes):
     - $L = 2 \times 2 \times 2 \times 2 \times 2 \times 5$
     - $M = 7 \times 7 \times 3$
  3. 2-stage: $L = 2 \times 80$, $M = 49 \times 3$
  4. 3-stage:
     - $L = 2 \times 5 \times 16$, $M = 7 \times 7 \times 3$
- $N$ = filter order(s) used for resampling
- No. of additions and multiplications was counted (MIPS = Million ops/sec)
- It is best to use only 4-5 stages!

Asynchronous Conversion

- Irrational or time-varying conversion
- Any resampling operation is possible, if the underlying analog signal is first reconstructed using sinc functions
  - Then it can be sampled again at any time instants
  - Remember the sampling theorem
    - Sample fast enough or lowpass filter before resampling

Matlab demo:
sinreconstr
Approximate Signal Reconstruction

- Digital approximation of reconstructed analog signal
- Increase the sampling rate with a large enough factor $L$
  - Enough is when the difference between 2 consecutive samples is less than the quantization step $Q$
- Interpolation with large factor $L$ can be implemented in 2 stages
  - First upsample by integer factor $L_1$ (such as 2 or 3) using a long FIR filter and then apply low-order FIR interpolation

Implementing Asynchronous Conversion

- Filter the signal with a long FIR interpolation filter
- At each step, compute convolution between the FIR filter coefficients and the signal samples
  - Weighted sum of samples (filter coefficients are weights)
  - Weighted sum of filter coefficients (sample values are weights)
    (Convolution is commutable)
- Each interpolation point can be chosen freely
  - Different FIR filter for every output sample, if desired!
  - Fractional delay $d$ can be chosen freely ($0 < d < 1$ plus integer delay)
  - However, the sampling theorem should not be forgotten! (Remember the lowpass filter when sampling rate is lowered)
Polynomial Interpolation Methods

- Use Lagrange or spline interpolation instead of the sinc function
  - When the order of Lagrange or spline interpolator is taken to infinity, they approach sinc interpolation
  - Implementation: at every time step, a different fractional delay FIR filter is used
- Low-order polynomial interpolation is easy, but not very accurate
  - Aliasing is not well suppressed
  - Amplitude is attenuated at high frequencies

Lagrsinc.m
March 24, 2017 ©2001-2017 Vesa Välimäki 31

Lagrange Interpolation

<table>
<thead>
<tr>
<th>Polynomial interpolation</th>
<th>Output sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>0th ( y = ax^0 ) ( y = a )</td>
<td><img src="time" alt="Diagram" /></td>
</tr>
<tr>
<td>1st ( y = ax + bx^1 )</td>
<td><img src="time" alt="Diagram" /></td>
</tr>
<tr>
<td>2nd ( y = ax + bx^1 + cx^2 )</td>
<td><img src="time" alt="Diagram" /></td>
</tr>
<tr>
<td>3rd ( y = ax + bx^1 + cx^2 + dx^3 )</td>
<td><img src="time" alt="Diagram" /></td>
</tr>
</tbody>
</table>

March 24, 2017 ©2001-2017 Vesa Välimäki 32
Application example: Varispeed

- Continuously variable sample-rate conversion, where the sample rate (play back speed) remains constant, however
  - The play back speed can then be freely varied

Demo by Simo Broman & Heidi-Maria Lehtonen, TKK 2004

DEMO

Time-Stretching

by Mats & Robert
Efficient Resampling Algorithm

- Originally proposed by Smith and Gossett (1984)
- Simulates the analog signal reconstruction and resampling
  - Approximate reconstruction & resampling at the same time
  - Clever version of the multistage sample rate converter: only 1 FIR filter, but coefficients computed in 2 stages (stored and interpolated)
- Store a very long sinc-like function in a table
  - A windowed sinc function
  - For example, 13 zero-crossings included (in both directions)
  - The sinc function may have to be upsampled by factor 65536

Windowing the Sinc Function

- Window function
- Sinc function
- Windowed sinc function
Smith-Gossett Algorithm

- Much data, but can be reduced in two ways
  1. Store only one half of the sinc function, because it is symmetric
  2. Since the function is smooth, interpolation of values is possible

Linear Interpolation of Tabulated Sinc Values

Figure by Lasse Kärkkäinen & Teemu Sipilä, 2005
Smith-Gossett Algorithm (2)

- **A trick to reduce table size:** decimate the sinc function and compute the missing values with linear interpolation
  - Upsampling factor only 256 or 512 (not 65536)
  - Store both sinc and its first difference sinc(n)–sinc(n–1) to get efficient hardware implementation
    - If memory is cheap, but additions and multiplications consume time

**Linear interpolation**

\[
x(n) \rightarrow z^{-1} \rightarrow h(0) \rightarrow h(1) \rightarrow y(n)
\]

\[
h(0) = 1 - \frac{1}{D}, \quad h(1) = D
\]

Variable Filter in Smith-Gossett Algorithm

- In the case of decimation, the sinc function can be stretched/compressed to change the cutoff frequency by indexing
  - Only one sinc table is needed (for all conversion factors)
  - When the cutoff frequency is decreased, the sinc function gets fat
  - Stretching/shrinking happens by scaling the sampling step (index hops)
Smith-Gossett Testing 1

- Linear sine sweep from 0 Hz to 22050 Hz at the sample rate of 44.1 kHz

Demo by Samu Kaajas & Jussi Virolainen, TKK, 2001

Smith-Gossett Testing: References Case

- Convert the sine sweep from 44.1 kHz to 32 kHz sample rate properly
- Target: Linear sine sweep from 0 Hz to 16000 Hz
- Note: silence after 4.5 s (lowpass filtering)

Demo by Samu Kaajas & Jussi Virolainen, TKK, 2001
Smith-Gossett Example #1

- Sample rate conversion from 44.1 kHz to 32 kHz
  - Low-order Smith-Gossett alg: length 4

Demo by Samu Kaajas & Jussi Virolainen, TKK, 2001

Smith-Gossett Example #2

- Sample rate conversion from 44.1 kHz to 32 kHz
  - High-order Smith-Gossett alg: length 46

Demo by Samu Kaajas & Jussi Virolainen, TKK, 2001
Smith-Gossett Sound Examples

• Original piano recording (CD, 44.1 kHz)
• Converted to 32 kHz
  – Conversion ratio 44100/32000 = 441/320 = 1.378
  1. Smith-Gossett algorithm: filter length 4
     (Notice the aliasing!)
  2. Smith-Gossett algorithm: filter length 46

Demo by Samu Kaajas & Jussi Virolainen, TKK, 2001

1 Million-Hz Recording

• Recording of the ringing of door keys using an ultrasound microphone at the 1 MHz sample rate shows energy up to about 200 kHz!!!
Conclusion

- Many sampling rates used in digital audio
- Many possible conversion ratios
  - Often ratios of large integers, like 48000/44100 = 160/147
  - Also irrational ratios, like 48000/75470 = 0.63601431
- Decimation
  - Lowpass filtering before disregarding samples
- Interpolation
  - Image frequencies must be lowpass filtered
- Different algorithms for rational and irrational ratios
  - Economical polyphase and multistage filters
  - Smith-Gossett algorithm
References