Дигитална обрада звука и слике 5

Саша Малков
Математички факултет
2023/2024

Дигитална обрада звука и слике

Тема 2
Дигитална обрада звука

Разумевање јачине звука и *dB*

- Јачина у децибелима се изражава као
  - *dbЈачина = log$_{10}$(Snaga) x 10*
- Т.j. “два пута јачи звук” има око 3*dB* више
- Али сигнал обично не описује снагу него потенцијал/напон
- А снага зависи квадратно од напона:
  - *Snaga ~ Napon x Napon*
- Сигнал који је два пута јачи, производи два пута јачи напон и 4 пута јачи звук, па има око 6*dB* више
- При обради сигнала, “два пута већа амплитуда сигнала” има око 6*dB* више
  - али то није 2x него 4x јачи звук

Још пар техника у алату *Audacity*

- Лабеле
  - “запис” са ознакама
  - служи за лакше сналажење у већим пројектима
  - означавају се тачке или сегменти
  - сегменти могу да се спајају у тачкама (претходни сегмент се завршава а наредни сегмент почиње)
  - нова лабела се прави са *Ctrl+B*
  - *Edit / Labels / Label Editor*

Још пар техника у алату *Audacity*

- *Time Track*
  - “запис” са подешавањем темпа
  - може да се успорава или убрзава емитовање свих записа

Дигитална обрада звука и Пајтон

- Има више библиотека за рад за звучним записима
- На пример:
  - *simpleaudio*
  - *pyo*
  - *pydub*
  - *ffmpeg*
- Упознаћемо основне могућности библиотеке *pydub*

Библиотека *pydub*

- Основни елементи библиотеке су аудио-записи
- Неки елементи библиотеке су имплементирани помоћу других библиотека, као што су *simpleaudio* и *ffmpeg*

Читање и емитовање, основне карактеристике

- Основни тип је *AudioSegment*
  - представља један запис
  - може да се чита из фајла
    - *AudioSegment.from_wav(...)*
  - за неке формате може да буде неопходан *ffmpeg*
- За емитовање се користи
  - *playback.play(...)*

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

print( "Channels:          ", track.channels )
print( "Sample width:      ", track.sample_width, "B" )
print( "Frame rate:        ", track.frame_rate )
print( "Loudness:          ", track.dBFS, "dB" )
print( "RMS:               ", track.rms )
print( "Max value:         ", track.max )
print( "Max amplitude:     ", track.max_dBFS, "dB" )
print( "Duration:          ", track.duration_seconds, "s"
)

playback.play( track )
```

Сегменти

- Оператор индексирања може да се користи за издвајање сегмената записа
- Јединица је милисекунда

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

# izdvajanje segmenata, u ms
playback.play( track[4400:] )
playback.play( track[2200:4400] )
playback.play( track[:2200] )
```


Појачавање

- Промена јачине се одвија или методом *apply_gain(...)* или додавањем броја
  - у оба случаја је параметар у децибелима

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

# promena jacine
playback.play( track[4400:] - 12 )
playback.play( track[2200:4400] )
playback.play( track[:2200].apply_gain(12) )
```


Комбиновање записа

- Записи се комбинују методом *t1.overlay(t2)*
- Основни запис *t1* мора да буде дужи или ће се резултат скратити
- Опциони параметри:
  - *position = ...*
    - од које позиције се преклапа
  - *times = ...*
    - колико пута се понавља

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

# kombinovanje snimaka
t1 = track[4400:] - 12
t1 = t1.overlay( track[2200:4400] )
t1 = t1.overlay( track[:2200] + 12 )

playback.play( t1 )

# dodajemo drugi snimak 1s kasnije
track = track.overlay( track, position=1000 )
playback.play( track )
```


Померање на леви или десни канал

- Звук може да се помери улево или удесно
  - *t1.pan(c)*
- наводи се вредност од -1 до 1
  - -1 – скроз улево
  - 1 – скроз удесно

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

# kombinovanje snimaka sa izborom kanala
t1 = (track[4400:] - 12).pan(-1)
t1 = t1.overlay( track[2200:4400] )
t1 = t1.overlay( (track[:2200] + 12).pan(1) )

playback.play( t1 )
```

Снимање

- Запис може да се сними у различитим форматима
- Учесталост узорака или дубина записа се мењају пре снимања

```
import pydub
from pydub import AudioSegment
from pydub import playback

track = AudioSegment.from_wav("glas_16_44.wav")

# kombinovanje snimaka sa izborom kanala
t1 = (track[4400:] - 12).pan(-1)
t1 = t1.overlay( track[2200:4400] )
t1 = t1.overlay( (track[:2200] + 12).pan(1) )

# snimanje
t1.export( "output.wav", format="wav" )
t1.export( "output.mp3", format="mp3", bitrate="192k" )

# t1.set_channels(1)
t1 = t1.set_frame_rate( 48000 )
t1 = t1.set_sample_width( 3 )
t1.export( "output_2.wav", format="wav" )
```


Генерисање тонова

- Има више начина генерисања
- Основно је генерисање синусног тона дате фреквенције, трајања и јачине

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

# pravljenje tona
result = AudioSegment.silent( 0 )
for n in range(15):
    # ton frekvencije n * 220 Hz
    sound = generators.Sine( 220 * n ).to_audio_segment(
duration=100, volume=-3*n )
    # dodajemo novi ton na kraj
    result += sound

playback.play( result )
```


Дописивање на крај

- Оператором сабирања
- Методом *append*

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators

# pravljenje tona
result = AudioSegment.silent( 50 )
for n in range(15):
    # ton frekvencije n * 220 Hz
    sound = generators.Sine( 220 * n ).to_audio_segment(
duration=100, volume=-3*n )
    # dodajemo sa preklapanjem
    result = result.append( sound, 50 )

playback.play( result )
```


Преклапање са прелазима

- Преклапањем више генерисаних тонова можемо да направимо нпр. хармонике

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

# pravljenje tona
result = AudioSegment.silent( 2000 )
for n in range(15):
    # ton frekvencije n * 220 Hz
    sound = generators.Sine( 220 * n ).to_audio_segment(
duration=100, volume=-3*n )
    # preklapamo sa ublazavanjem
    result = result.overlay( AudioSegment.silent(n*100)
+ sound.fade_in(50).fade_out(50) )

playback.play( result )
```

Ниско-пропусни и високо-пропусни филтер

- Методи *low_pass_filter* и *high_pass_filter* имплементирају ниско-пропусни и високо-пропусни филтер

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

# pravljenje tona
result = AudioSegment.silent( 2000 )
for n in range(10):
    # ton frekvencije n * 220 Hz
    sound = generators.Sine( 440 * (n+1) ).to_audio_segment(
duration=1000, volume=-3*n-13 )
    sound = sound.fade_in( 100 ).fade_out( 100 )
    # preklapamo harmonike
    result = result.overlay( sound )

playback.play( result )
playback.play( result.low_pass_filter(3000) )
playback.play( result.low_pass_filter(1000) )
playback.play( result.low_pass_filter(500) )

playback.play( result )
playback.play( result.high_pass_filter(500) )
playback.play( result.high_pass_filter(1000) )
playback.play( result.high_pass_filter(3000) )
```


Нормализација

- Метод *normalize* нормализује јачину записа
- Опциони аргумент одређује колико простора се оставља до максимума

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

track = AudioSegment.from_wav("glas_16_44.wav")
track = track.normalize()
print( track[:2200].max, track[2200:4400].max, track[4400:].max)
playback.play( track )

# normalizacija se podrazumevano odnosi na amplitude
# tj. max = max0 / 10^ (n/20)

track = track.normalize(6)
print( track[:2200].max, track[2200:4400].max, track[4400:].max)
playback.play( track )

track = track.normalize(3)
print( track[:2200].max, track[2200:4400].max, track[4400:].max)
playback.play( track )
```


Компресија

- Компресија је слична као код програма *Audacity*
- Први аргумент (*threshold*) је амплитуда изнад које се врши редукција јачине
- Други аргумент (*ratio*) је фактор редукције јачине
- Трећи аргумент (*attack*) је трајање јаког тона да би се започело стишавање
- Четврти аргумент (*release*) је трајање тихог тона да би се прекинуло стишавање

```
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

track = AudioSegment.from_wav("glas_16_44.wav")
track = track.normalize(0.2)
print( track[:2200].max, track[2200:4400].max, track[4400:].max)
playback.play( track )

t1 = track.compress_dynamic_range( -20, 4 ).normalize(0.2)
print( t1[:2200].max, t1[2200:4400].max, t1[4400:].max)
playback.play( t1 )

t1 = track.compress_dynamic_range( -20, 10 ).normalize(0.2)
print( t1[:2200].max, t1[2200:4400].max, t1[4400:].max)
playback.play( t1 )
```


Промена брзине

- Промена брзине се своди на сецкање записа на мале сегменте (подразумевано 150мс) и избацивање једног броја сегмената
  - За боље резултате може да се користи мања величина сегмента
  - За још боље резултате може да се врши преклапање парчића

```
import pydub
import pydub
from pydub import AudioSegment
from pydub import playback
from pydub import generators
from pydub import effects

track = AudioSegment.from_wav("glas_16_44.wav")
track = track.normalize(0.2)
print( track.duration_seconds)
playback.play( track )

# podrazumevani parcici su 150ms
t1 = track.speedup(2)
print( t1.duration_seconds )
playback.play( t1 )

# kada se skrate parcici, postaje razumljivije
t1 = track.speedup(2, chunk_size=50 )
print( t1.duration_seconds )
playback.play( t1 )

# kada se parcici preklope sa ublazenim prelazima postaje jos
razumljivije
t1 = track.speedup(2, chunk_size=50, crossfade=10 )
print( t1.duration_seconds )
playback.play( t1 )
```

Литература

- **Документација за *Pydub***
  - https://github.com/jiaaro/pydub