pyrecplay_quantizationblock

January 27, 2017

0.1 PyAudio Example:

Make a modulation between input and output (i.e., record a few samples, modulate them with a sine, and play them back immediately). Using block-wise processing instead of a for loop Gerald Schuller, Octtober 2014.

Import the modules and define the variables.

```
In [1]: import pyaudio
        import struct
        #import math
        #import array
        import numpy as np
        #import scipy

        CHUNK = 5000 #Blocksize
        WIDTH = 2 #2 bytes per sample
        CHANNELS = 1 #2
        RATE = 32000  #Sampling Rate in Hz
        RECORD_SECONDS = 8
```

Iniatialize the sound card.

```
In [2]: p = pyaudio.PyAudio()

        stream = p.open(format=p.get_format_from_width(WIDTH),
                        channels=CHANNELS,
                        rate=RATE,
                        input=True,
                        output=True,
                        #input_device_index=10,
                        frames_per_buffer=CHUNK)
```

Start recording and playback the quantized version of it.

```
In [3]: #Loop for the blocks:
        for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
            #Reading from audio input stream into data with block length "CHUNK":
            data = stream.read(CHUNK)
```

```
            #Convert from stream of bytes to a list of short integers (2 bytes here) in "samples
            #shorts = (struct.unpack( "128h", data ))
            shorts = (struct.unpack( 'h' * CHUNK, data ));
            samples=np.array(list(shorts),dtype=float);

            #start block-wise signal processing:

            #Quantization, for a signal between -32000 to +32000:
            q=5000;

            #Mid Tread quantization:
            indices=np.round(samples/q)
            #de-quantization:
            samples=indices*q;

            #Mid -Rise quantizer:
            #indices=np.floor(samples/q)
            #de-quantization:
            #samples=indices*q+q/2;

            #end signal processing

            #converting from short integers to a stream of bytes in "data":
            data=struct.pack('h' * len(samples), *samples);
            #Writing data back to audio output stream:
            stream.write(data, CHUNK)

        print("* done")

        stream.stop_stream()
        stream.close()

        p.terminate()
```

```
c:\python27\lib\site-packages\ipykernel\__main__.py:28: DeprecationWarning: integer argument exp


* done
```