

Equalizer

Ein Equalizer-Objekt beinhaltet zwei Buffer (je einen Buffer pro Audiokanal) in Form von float-Arrays. Zusätzlich existiert noch ein dritter Buffer (derselben Grösse) für die FFT. Darin werden die Frequenzanteile zwischengespeichert. Des weitem implementiert die Equalizer-Klasse folgende Methoden:

`void write(float sample)`

Mit dieser Methode werden die Buffer für den rechten und den linken Kanal gefüllt. Portaudio wechselt dabei ständig zwischen den beiden Kanälen ab. Die Samples an den ungeraden Positionen gehören zum rechten Kanal, diejenigen an geraden Positionen zum linken Kanal. (beginnend mit 1 = ungerade = rechter Kanal)

`float read()`

Solange die Buffer leer sind, wird 0.0 zurückgegeben. Sind die Buffer gefüllt, so wird Sample für Sample (von Anfang an) zurückgegeben. Die Verzögerungszeit ist daher von der Buffergrösse proportional abhängig.

Nach jedem vollständigen Durchlauf eines Buffers wird eine Fourier-Analyse (FFT) über diesen Buffer durchgeführt. Nun werden die erhaltenen Werte (in der Frequenz-Array zwischengespeichert) mit den entsprechenden Faktoren (von den Reglern) verrechnet. Anschliessend wird das Frequenzspektrum per inverser Fourier-Transformation (IFFT) wieder in den entsprechenden Sample-Buffer zurücktransferiert. Nach der Rücktransformation muss unbedingt ein „rescale“ erfolgen, um die erhaltenen Werte entsprechend der Ausgangssituation zu skalieren.

Durch die vorgenommenen Änderungen treten beim Signal von Block zu Block Sprungstellen auf. Dies führt zu unangenehmem „Knacksen“ in regelmässigen Abständen. Um dies zu unterbinden wird eine Windowing-Technik angewendet. Diese Technik verändert das Signal so, dass es bei den Blockübergängen gegen 0 konvergiert und somit keine Sprungstellen mehr aufweist. (Details, siehe Windowing)

`changeValue(int sliderNr, int value)`

Definitionsbereiche:

sliderNr:	1 ... 10
value:	-100 ... 100

`doHanning(int size)`

size: Anzahl Samples, die durch „Windowing“ verändert werden; beginnend von aussen (an den Rändern); in der Mitte wird nichts verändert.

Um den Equalizer zu testen habe ich ein Qt-Programm geschrieben, welches Audiosignale von Line-In empfängt, durch ein „equalizerClass“-Objekt durchschlauft und die Signale schliesslich wieder ausgibt. Mit den zehn Reglern können nun die entsprechenden Frequenzbereiche während des Betriebes verstärkt, gedämpft oder ganz ausgeschaltet werden.