Single gesture multi-effect

Single gesture multi-effect

Course main assignment

Description

Design and implement (in C++) a multi-effect of which multiple effect parameters can be controlled by one overarching parameter, thus via a 'single gesture'.

Within this project you can either focus on the creation of:

- a realtime **multi-effect audio plugin that can be controlled by only one knob** (PLUGIN FOCUS), or;
- a realtime **multi-effect console application that can be controlled through OSC by one 'gesture' with a ready-made controller over OSC** in the real world (PLUG&PLAY FOCUS).
- a realtime **multi-effect console application that can be controlled through OSC by one 'gesture' with a custom-made controller over OSC** in the real world (CUSTOM FOCUS).

The latter, the CUSTOM FOCUS, requires a custom design. In other words, simply mapping an existing controller like a mouse or midi-controller is not sufficient for the CUSTOM FOCUS.

Noteworthy, due to the three offered options regarding design focus, the differences in deliverables and requirements for the PLUGIN FOCUS, PLUG&PLAY FOCUS and CUSTOM FOCUS are specified separately below.

Deliverables

Deliverables are submitted to an online git repository.

- **Overall design**
  An overall description of the multi-effect (two paragraphs), which should be informative regarding:
  - The chosen focus (PLUGIN FOCUS / PLUG&PLAY FOCUS / CUSTOM FOCUS) and short explanation of your choice.
  - An overall description of the (proposed) effects in your multi-effect, together with a reasoning behind your selection.
  - A description and justification of the mapping of the one overarching parameter into multiple parameters (can be combined with the block diagram, see further below).
  - PLUG&PLAY FOCUS - Short description of the chosen ready-made controller and its interaction.
  - CUSTOM FOCUS - Thorough interaction design document of the customly designed controller.

- **Diagrams**
    - Block diagrams, one for each implemented effect
    - Overall block diagram displaying the effect chain together with the mapping of the one overarching parameter into multiple parameters.
    - PLUG&PLAY FOCUS & CUSTOM FOCUS - flow / block diagram of the controller interaction and the mapping to one parameter.
    - A basic **class diagram**, displaying inheritance and association relationships between classes

- **Process description**
    - Brief description of the design process of the multi-effect and mapping, including the associated design choices
    - PLUG&PLAY FOCUS - Very brief description of the design process of the mapping of the ready-made controller to one parameter, including the associated design choices.
    - CUSTOM FOCUS - Brief description of the design process of the custom controller, including the associated design choices.

- **Codebase**

- **Selected learning objectives and reflection**, a one page document (+/- 400 a 600 words) describing:
    - The three **selected learning objectives**, provided with a **reflection** per learning goal
    - **Overall reflection**, covering:
        - The design and implementation *(e.g. proud about, possible improvements, etc.)*
        - Learned knowledge/skills *(besides the selected learning objectives)*
        - Process *(e.g. remarkably good planning, struggles)*
    - Reflection transformed into **at least two take-aways**, relevant for the upcoming course

Requirements

Apart from the generic requirements, which can be found in the CSD git repository *("beoordelingscriteria.pdf" covering topics like input validation, an executable and 'sounding' application)*, the implementation should meet the following requirements:

- The customly designed and self-implemented final artefact consists of a realtime **multi-effect**
    - PLUGIN FOCUS: **…** audio plugin that can be controlled by only one knob , or;
    - PLUG&PLAY FOCUS: … console application that can be controlled through OSC by one 'gesture' interaction with a ready-made controller.
    - CUSTOM FOCUS: … console application that can be controlled through OSC by one 'gesture' interaction with a custom-designed and custom-built controller.

- See **Effect implementation difficulty**. The multi-effect consists of at least **three custom-made effects** that **together yield a difficulty level** of at least
    - PLUGIN FOCUS: **six** points
    - PLUG & PLAY FOCUS: **five** points
    - CUSTOM FOCUS: **three** points
- OOP is correctly applied.

Assessment specific criteria

Apart from the generic assessment criteria, which can be found in the CSD git repository *(“beoordelingscriteria.pdf” covering topics like clear naming and comments)* the assessment this course will cover:

- Presentation
    - Realtime demonstration of the final artefact.
    - Chosen focus and the custom design.
    - Reflection and take-aways
- Design
    - Justification of the design choices made with respect to the effect selection and the mapping.
- Documentation
    - Readability and clarity of the required documents
- Codebase
    - OOP
    - DSP effects correctly implemented
    - Clear and correct mapping overarching parameter to separate effect parameters.
    - Code flow is concise; no redundant function calls / lines
    - *Good practice* regarding naming, comments, style,
- Reflection
    - Coverage of the required elements
    - Readability and clarity

Effect implementation difficulty

The marked effects are part of the course code examples. The associated level of difficulty does not count towards the total difficulty level of your multi-effect implementation. A multi-effect that combines the following effects counts for five difficulty level points.

- Waveshaper with wavetable - linear interpolation and at least one other shape (1)

- Chorus (1)
- Bitcrusher (1)
- Noise Gate (2)

Effect Category	Effect	Utilizes	Implementation difficulty
filter	Biquad Filter		4
	Moog Ladder Filter		3
	State Variable Filter		4
	Onepole / RC filter		1
	All Pass Filter	Delay / Biquad	4
	Brick Wall Filter	FFT	Not Allowed
	Butterworth Filter	Biquad	3
	Linkwitz-Riley Filter	Biquad	3
Delay	Feedback Delay - instelbare delay tijd (dus met interpolatie en soort van envelop bij het verstellen van delay tijd) *(=^ Comb Filter)*	(circular buffer)	1.5
	Chorus	Delay, Oscillator	1
	Vibrato	Delay, Oscillator	1
	Flanger	Delay, Oscillator	1
	Doppler effect (basics only)	Delay	2
	Grain delay	(circular buffer)	3
	Pitch Shifter	(circular buffer)	2
	Sampler	(circular buffer)	3
	Tape Delay - met interpolatie	Delay, noise, interpolation	1
	Tape Delay - met interpolatie & instelbare delay tijd	Delay, noise, interpolation	2

	Reversed delay		
Reverb	Plate	All Pass Filter	4
	Convolution	FFT	Not Allowed
	Room	All Pass Filter	5
	Bucket Brigade	Delay	4
	Dattorro	Delay, All Pass Filter	2
	Griesinger	All Pass Filter	3
	Shimmer	All Pass Filter	4
Dynamics	Bit crusher		1
	Noise gate	Delay	2
	Limiter		
	De-esser	Delay, High pass	2
	Compressor, simple	Delay	2
	Compressor, advanced		5
	Waveshaper without wavetable		1
	Waveshaper with wavetable - linear interpolation and at least one other shape	Waveshaper, wavetable	1
Misc.	Tremolo		1
	LFO-based auto-panner		1
	Ringmodulator		1
	Phaser	All Pass Filter	3
	Disperser	All Pass Filter	2
	Multiband ___	Linkwitz-Riley / Or other	3

	Phase vocoder	Short-time FT	Not Allowed

References for self-study

References for self-study

DSP

- Pirkle, W. (2019). *Designing audio effect plugins in C++: for AAX, AU, and VST3 with DSP theory*. Routledge.
- Boulanger, R. (Ed.). (2000). *The Csound book: perspectives in software synthesis, sound design, signal processing, and programming*. MIT press.
- Trevino, J., & Allen, D. (2012). The Audio Programming Book, Edited by Richard Boulanger and Victor Lazzarini. *Computer Music Journal*, *36*(2), 85-89.

C++

- https://en.cppreference.com/
- https://www.tutorialspoint.com/cplusplus/index.htm
- https://www.w3schools.com/cpp/