CMLS - HW2 Report

10494722
10800940
10502570
10719249

May 29, 2021

Contents

Chapter 1

HW3 - Drum Kit

1.1 Homework

The aim of Assingnment 1 is to create a set of drum sound, minimum 3 and an interface to control them. Is possible to implement non traditional drum sounds. Is possible to exploit the interactions with MIDI or OSC communication protocols. The GitHub repository of the code is available at this link:

`https://github.com/DavideTommy/Drum_Kit-CMLS_HW3.git`

The repository is already public. Click on link above to reach.

1.2 Idea

In order to achive our homework we needed create a set of sounds and we did it using supercollider SinthDef. After that we needed to create a GUI and we decided to use TouchOscEditor to prepare it. We downloaded the .apk of android app and we uploaded our template on our mobile phones. The final step consists in create listeners able to receive special key words coming from button pressed and call the related Sinth function to play the desired sound.

1.3 Implementation

1.3.1 Sound Processor

The core of the sound processing is Super Collider. We need to define a SynthDef for each sound we wanna obtain. For each sound is possible to set the frequency and the amplitude of the sound we wanna obtain. Is also possible to add different sounds in the same sinth def.

At the end of the Systhesis section we have the series of listeners to call our function as soon as we receive the proper message from the graphic interface.

1.3.2 GUI - TouchOSC

The TouchOsc Editor is an useful tool that we used for the creation of the GUI. This tool allows us to create many different graphic elements, like labels, linear and rotary encoders and so on, to set a drum-kit-like interface. Furthermore, it is possible to move, resize and change colors for each element. The controls are:

- reverb master (slider)
- panning master (slider)
- metronome (number box + slider)
- legend
- master volume (slider)
- gain controls (rotary knobs, one for each instrument)
- frequency master (slider)
- 3 frequency slider, each one for a cymbal

Percussions are:

- drums and cymbals (8 tapable circles):
- High Tom 1
- High Tom 2
- Snare
- Floor Tom
- Kick
- Crash Cymbal
- Hi-Hat
- Ride Cymbal

We have chosen to implement our app to let it be both the more simply and complete for the user. At the top left we have located two master controls through two sliders, one for the reverb and one for the general panning. Immediately below, we put the metronome: it is activated with its slider too and shows the current BPM number. In the center we have the drum-kit, rendered graphically with circles of different colors and sizes: the three yellow ones are the battery cymbal, while the others are the drums. At the bottom left there is a minimal legend which shows the names of each instrument. On the right there are other controls: At the top there is the master volume, which is adjustable thanks to a knob; under it there are other rotary knobs, each one for adjusting the volume of its corresponding drum or cymbal. Below you can find a frequency master, which through a slider sets the globally perceived frequencies, while under it there are other three sliders, relative to each cymbal, to adjust their frequencies.

1.3.3 Listeners

In the second part of the code we prepared the connection with TouchOSC. We have defined a gain variable for each instrument to store the value set from the user thanks to the functions which have the job to collect the datum of each parameter passed from TouchOSC chosen by the user thanks to the value in /textttmsg[1]. Furthermore, the function will post in the Post window of Supercollider the current value for a matter of further precision. In addition to the instrument data, the general values such as the die master values, or those relating to the metronome bpm are also taken.

1.4 Conclusions

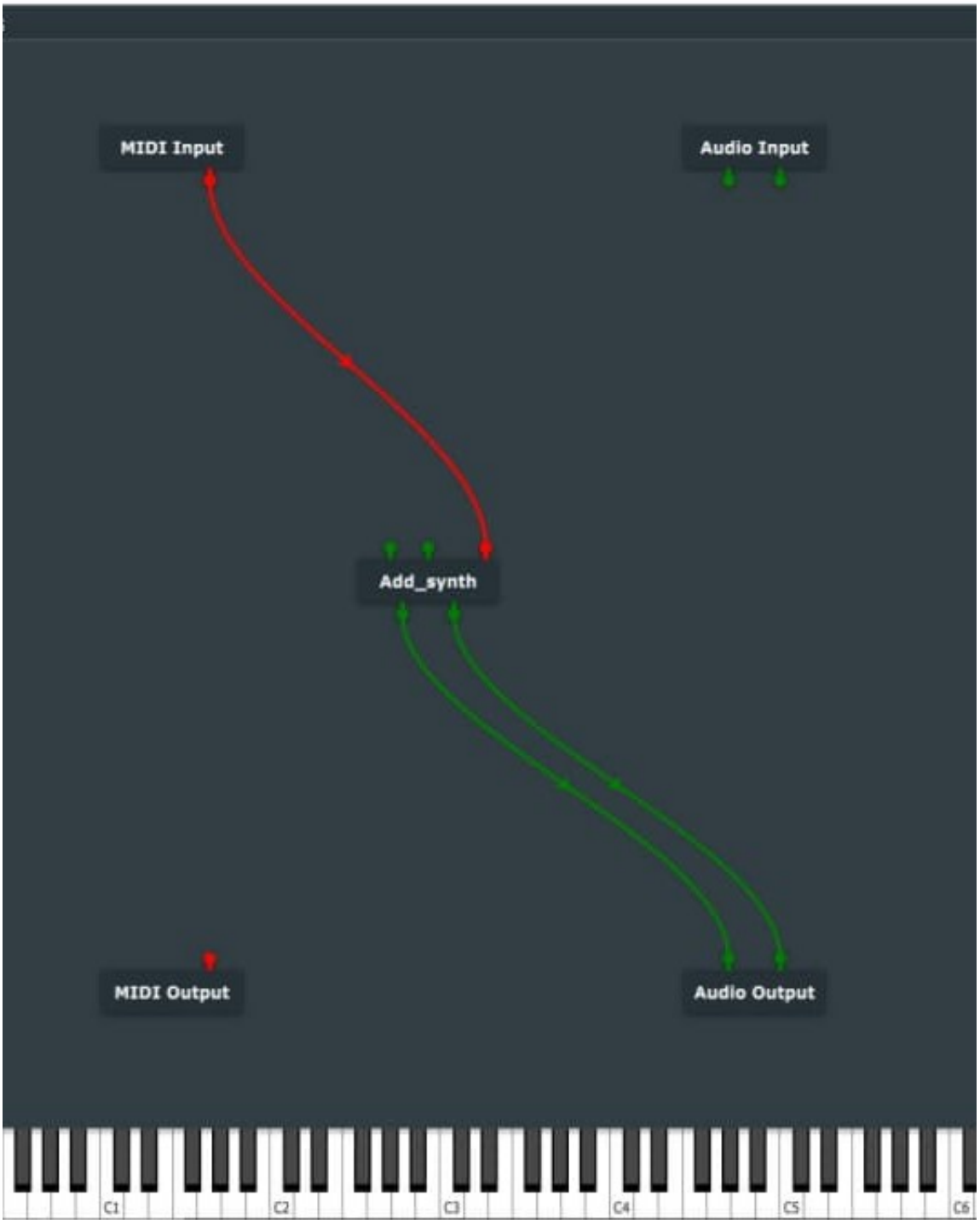
1.4.1 Features

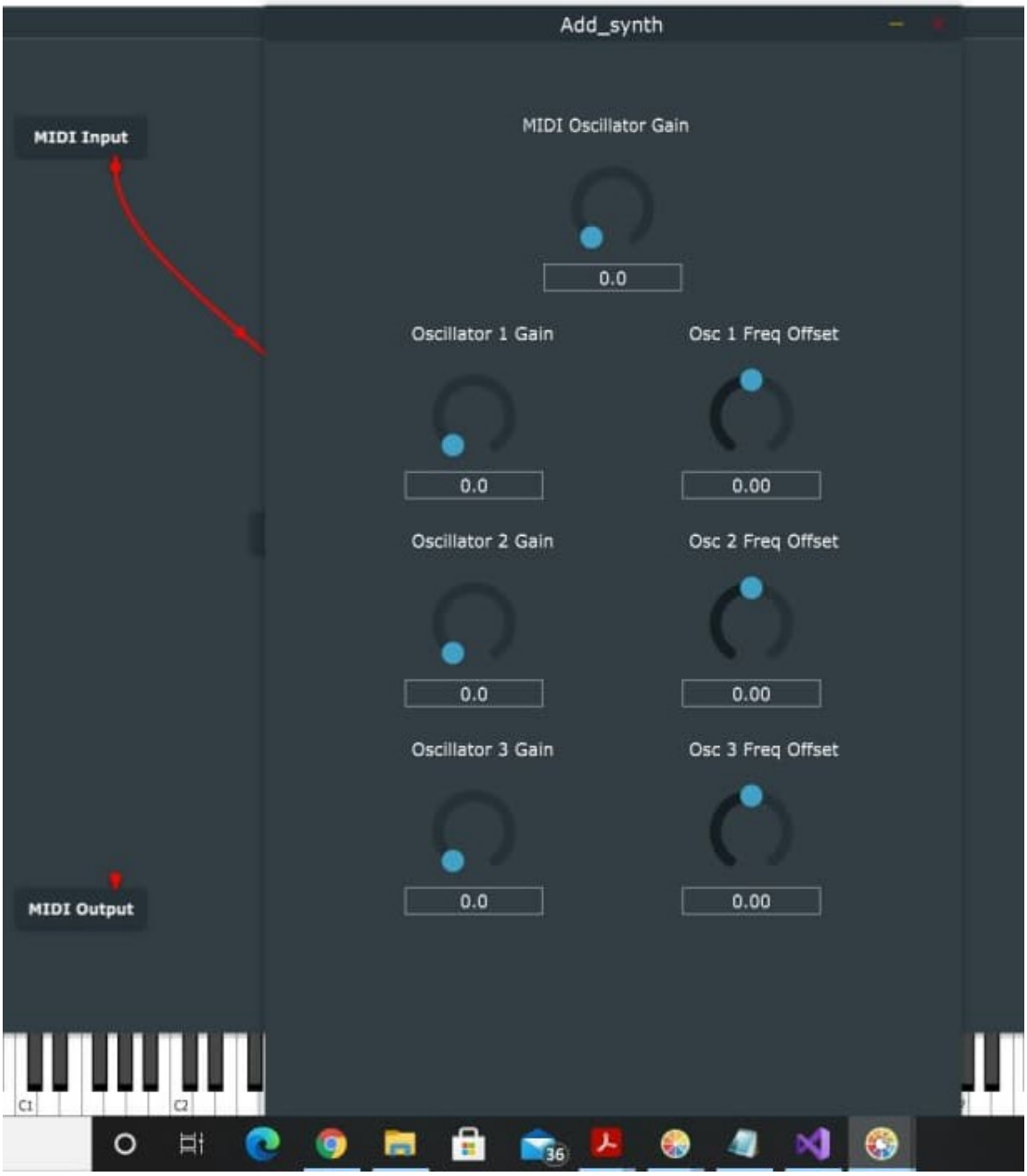
With further implementations is possible to substitute our sounds with pre sampled sounds, transforming our Drum-Kit in a functional effect box that can be used, for example, as musical effect generator for Streamers or DeeJay, instead MIDI box.

We think this should be very usefull in order to controll remotely not only sounds, but also equalizers, sliders, mixers etc. That should be really usefull in applications like the sound tuning and acoustic regulation of an arena, or a theater.

1.4.2 Graphic Screenshots

Here we have some screenshot about the final result.


MIDI Input
Audio Input
Add_synth
MIDI Output
Audio Output
C1
C2
C3
C4
C5
C6


Add_synth
MIDI Input
MIDI Oscillator Gain
0.0
Oscillator 1 Gain
Osc 1 Freq Offset
0.0
0.00
Oscillator 2 Gain
Osc 2 Freq Offset
0.0
0.00
Oscillator 3 Gain
Osc 3 Freq Offset
0.0
0.00
MIDI Output
C1
C2