ICSC 2024

7th INTERNATIONAL CSOUND CONFERENCE

September 17th – September 20th 2024

Vienna, Austria

PROCEEDINGS

Stadt Wien

DEPARTMENT OF MUSIC ACOUSTICS WIENER KLANGSTIL | IWK

mdw university of music and performing arts vienna

mdw Bild: shutterstock / riddboy

LOCATION

mdw – University of Music and Performing Arts Vienna
Anton-von-Webern-Platz 1, 1030 Vienna, Austria
Klangtheater (Sound Theater), Future Art Lab
Conference room AW K0101
Conference room AW M0107

CONFERENCE WEBSITE

https://www.mdw.ac.at/icsc2024/

ORGANISING COMMITTEE

Alex Hofmann, Sonja Stojak, Vasileios Chatziioannou, Oskar Gigele, Werner Goebl, Tim-Tarek Grund, Alessio Lampis, Titas Lasickas, Ewa Matusiak, Alexander Mayer, Montserrat Pàmies-Vilà, Paul Schuster, Dustin Zorn

PAPER REVIEW COMMITTEE

Øyvind Brandtsegg, Michael Gogins, Alex Hofmann, Tarmo Johannes, Luis Jure, Victor Lazzarini, Steven Yi, Rory Walsh

MUSIC REVIEW COMMITTEE

Michele Abondano, Jeanette C., Joachim Heintz, Oscar Pablo di Liscia, Feliz Macahis, Ghazaleh Moqanaki, Peter Plessas, Robert Rehnig, Dustin Zorn

SUPPORTED BY

Department of Music Acoustics – Wiener Klangstil (mdw – University of Music and Performing Arts Vienna)
Stadt Wien (City council of the city of Vienna)

PROGRAM EDITED BY

Alex Hofmann, Sonja Stojak
Department of Music Acoustics – Wiener Klangstil
mdw – University of Music and Performing Arts Vienna
Anton-von-Webern-Platz 1, 1030 Vienna, Austria

IMPRINT

Proceedings of the 7th International Csound Conference

edited by Sonja Stojak and Alex Hofmann

Department of Music Acoustics – Wiener Klangstil

mdw – University of Music and Performing Arts Vienna

Anton-von-Webern-Platz 1, 1030 Vienna, Austria

December 2024

DOI: 10.21939/ICSC2024

Exploring the Expressive VR performance of Csound Instruments in Unity

Ken Kobayashi

Berklee College of Music
kkobayashi4@berklee.edu

Abstract. Electronic music instruments have revolutionized musical performances. These gadgets allow musicians to perform using sound synthesis, unlocking infinite possiblities from countless algorithms, from those explored thoroughly to the cutting edge. However, as sound synthesis technologies evolve, such digital instruments must also be reimagined. A instrument that fully utilizes the capabilities of modern synthesizer technology should allow one to perform not just novel sounds, but be more expressive with their performance. This paper explores such expressiveness through an instrument created in VR, the *Laser Synth*.

Keywords: CsoundUnity, VR, Performance, Csound, Unity

1 Introduction

The *Laser Synth* [1] is a VR instrument focused on playability. It is distinct from VR installations, which enable a user to create evolving soundscapes with high degrees of randomness, such as those based on physics or hordes of objects. Instead, the *Laser Synth* will be playable in the traditional sense; a performer plays with intent for specific rhythm, pitch and synth modulations.

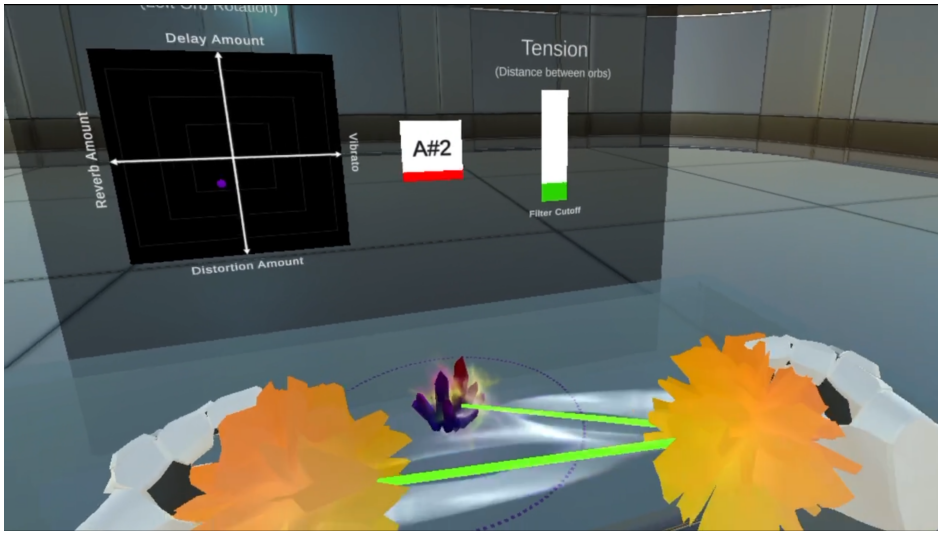

Fig. 1. *Laser Synth* being performed. Shown are both orbs in hands and the UI.

2 Instrument Control

The instrument consists of a stationary crystal and 2 virtual orbs, one to be grabbed by each hand. (See Fig. 1) The relative position of these 3 objects, along with the rotation of one, controls all elements of the instrument. The positional relationship of the 3 objects are shown by the laser connecting them. A laser connects the crystal and right orb, and the 2 orbs.

2.1 Pitch

Accurate pitch control is crucial for a melodic performance. Streamlining it acts as a neccessary foundation for all other features.

The distance between the crystal and the right orb controls pitch. The crystal remains stationary, while the right orb's position determines pitch, similar to a theremin. Simplifying this control by narrowing down the scope to a single element eliminates randomness, allowing performers to focus solely on their right hand's position for pitch adjustment.

To futher assist the performer for an accurate performance, the pitch is quantized to the 12 tone equal temperament scale. Because of the positional control, it is near impossible to play pithces accurately in a 3D, virtual space. Pitch quantization gives the performer significant margin of error to play their desired note. To compliment this, a visual tuner is shown in the center of the UI in front of them. The pitch is shown, with filling bar that represents how close to the next note it is. As shown in Fig. 1, the tuner shows A#2, with the bar being low and red, showing the performer is close to going down a pitch to A2 if the right orb is brought closer to the crystal.

Finally, to make transitions between pitches smoother, a portamento was added. This is to address the issue with large jumps in pitch being difficult to make sound elegant. It wasn't physically feasible to move fast enough. The portamento smooths out the frequencies between pitches, so the performer doesn't need to move too quickly, and can prioritize pitch accuracy.

2.2 Filter

Filter control is an invaluable tool to have in a live performance. The distance between the two orbs controls the filter cutoff frequency. This use of relative positioning of the two orbs makes filter control intuitive for the performer. If they intend to keep the filter constant through pitch changes, the performer will move both arms in identical motion. To open the filter, they will open their arms. Leaving the left orb stationary will provide a natural key mapping, opening the filter as the pitch increases.

Similar to pitch, the filter control has visual aids for the performer. On the right side of the UI (See Fig. 1), a bar reflects the cutoff of the filter. In Fig. 1, the green bar on the right is very low, reflecting the close distance between the two orbs. In addition, the color of the laser between the orbs will turn red as the filter nears the maximum frequency cutoff.

2.3 XY-Pad

For a truly expressive instrument, only being able to control a filter is not satisfactory. The rotation of the left orb controls an XY pad, enabling up to 4 parameters to be adjusted in real time. The axis (X and Y, both positive and negative) can indiviually be mapped to any parameter (See Fig. 2). The *Laser Synth* currently features 10 different parameters, including vibrato, pitchbend, reverb, and distortion (See Fig. 3).

The XY-Pad also has a visual aid for the performer. Located on the left side of the interface, a XY coordinate plane is labled with synth parameters along each axis (See Fig. 1). In Fig. 1, a dot can be observed to the bottom left of the origin of the pad. This is the value outputted by the rotation of the left orb pictured.

Combined with the filter control, the XY-pad feature adds immense expressive ability through the left hand. To manipulate the sound, the performer can use smooth and fluid gestures, such as a shoveling motion, or twisting your wrist in a figure eight motion. Simple gestures as such affect the filter and XY-pad simultaneously, resulting in an expressive sound, while being responsive to the movements of the performer. The *Laser Synth* is an virtual Csound instrument that is emotive with intension of the musician.

3 Implementation

The instrument was built using mainly two programs: Unity game engine and Csound. Some accompanying technologies were used to supplement and hasten the development process.

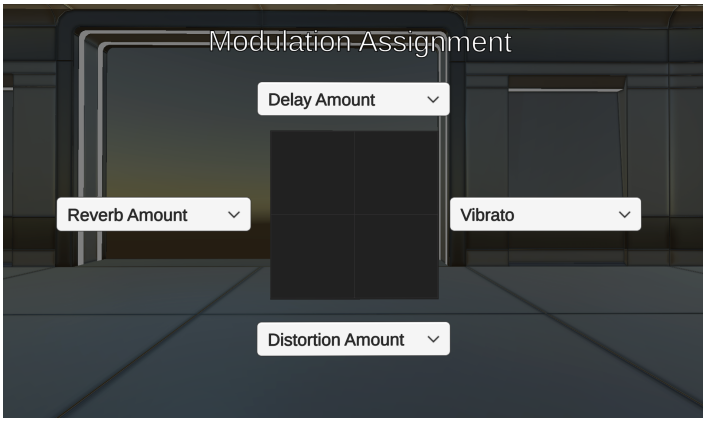

Fig. 2. XY-Pad customizable in VR through dropdowns on each axis.

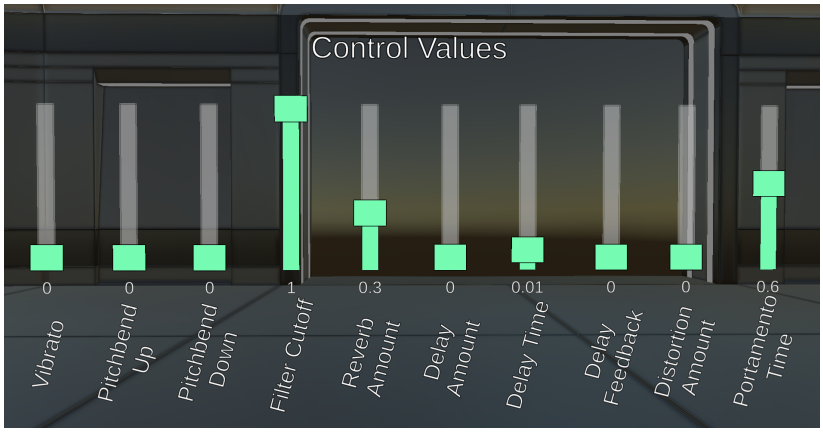

Fig. 3. List of all synth parameters and their respective values, shown in VR.

3.1 Unity

Unity is a game engine, commonly used for game development purposes. For the *Laser Synth*, Unity was used to create the 3D virtual environment. The visual design of the program, as well as the logic of the controls were created inside of Unity.

3.2 Csound

Csound drives nearly all of the sounds produced by the instrument. The program runs a monophonic lead sound, using the *poscil* opcode.

```
aOut poscil kamp, kfreq * kpitchbend + ksine, 1
```

The Cabbage editor [3], created by Rory Walsh and others, was used, allowing easy testing and implementation. One element heavily tested through Cabbage was the *note on* for the synth. A Csound channel controls turning on and off the synth. This was achieved through a listener instrument triggering a Csound event. By using Csound events, the synth is able to use the *madsr* opcode to implement an evelope, and allows easy implementation into Unity.

```
// When the noteon channel value is changed to 1, create an instance of intrument 1.
// When changed to 0, the instance is ended.
 instr 22
   knoteon = chnget:k("noteon")
   ktrigger = changed2:k(knoteon)

   if ktrigger == 1 then
       if knoteon == 1 then
           event "i", 1, 0 , -1
       else
           event "i", -1, 0, 1
       endif
       ktrigger = 0
   endif
 endin
```

All audio effects and their parameters were exposed to live manipulation by Unity through Csound channels.

```
kdist = port:k(chnget:k("distortion"), 0.01)
```

In above example, Unity is able to change the distortion amount of the Csound instument through the *"distortion"* channel. The *port* opcode is often combine with the *chnget* opcode to smooth out input values.

3.3 CsoundUnity

To allow the Csound instrument to play live audio and interact with Unity, CsoundUnity, created by Rory Walsh and others, was used. Through CsoundUnity [4], the Csound instrument is packaged within the Unity project, performing real-time audio synthesis. More importantly, it implements the *SetChannel* method into Unity to send values to the Csound instrument.
`BeamManager.cs (Ln 137)`

Setting frequency of Csound instrument through Unity

```
// convert midi note to frequency and send to csound
if (noteOn)
{
 freq = MidiToFrequency(midiNote);
 csound.SetChannel("freq", freq);
}
```

When a note is played, the played MIDI note value is converted into Hz. This value is then sent to the "freq" Csound channel to be used as the pitch.

3.4 Other Tools

Many other programs and assets were used to expedite the development process. Many of the 3D objects, visual elements, and packages are purchased from the Unity Asset Store. One such package is Auto Hand VR [5]. It handled the implementation of the VR headset and controller inputs, as well as interacting with objects through VR. It is an invaluable reasource, along with all of the other Unity Asset Store items which saved much time and effort for the project.

4 Conclusion

Expressiveness of an instrument is its ability to fulfill the performer's intention. The controls must not impede on the performer and be natural to use. In addition, it must provide a form of expression through easy, real-time parameters. The *Laser Synth* was designed with both rules in mind, making it a vessel for musical expression. The instrument is planned to allow for the sound generator to be customized for drastically personalized sounds, as well as tools for new users to understand the instrument, such as presets to jumpstart them into the experience and tutorials on the possibilities of the *Laser Synth*. This iteration of the *Laser Synth* is the start of a study of *how to play sound*, and how virtual instruments can be designed for different performances, performers, and sounds.

References

1. Ken Kobayashi. VR Laser Synth Github Page. `https://github.com/kencula/VRLaserSynth`
2. Boulanger, R.: The Csound Book: Perspectives in software synthesis, sound design, signal Processing, and Programming. MIT. (2000)
3. Cabbage Website, `https://cabbageaudio.com`
4. CsoundUnity Github site, `https://github.com/rorywalsh/CsoundUnity`
5. Auto Hand VR Unity Asset Store page, `https://assetstore.unity.com/packages/tools/game-toolkits/auto-hand-vr-interaction-165323`
6. The Canonical Csound Reference Manual, `https://csound.com/docs/manual/index.html`
7. The Csound FLOSS Manual, `https://flossmanual.csound.com`