Project Report

Conor Holden

February 28, 2021

Abstract

Stuff about Pipes

Contents

1 Introduction

- Create a virtual instrument for the Uilleann pipes
- not many there in advance
- Create a VST plug-in that can be used in digital audio work stations

2 Analysis

2.1 How Uillean Pipes Work

2.2 Simulating Instruments

The main goal of the project is to simulate the uilleann pipes so that they can be used as a virtual instrument. There are many possible ways to do this.

2.2.1 Physical Modelling Synthesis

- Original Plan
- Mathematical Model of the Sound Wave (not simulation)
- Fairly easy on string instruments
- Uillean pipes relatively obscure
- out of scope for the project

2.2.2 Subtractive Synthesis

- Maynooth university thingy
- reasonable successful in pre-made synths
- would end up created a regular but limited subtractive syntheses

2.2.3 Sampling

Sampling is the intuitive and obvious way to create virtual instruments. It involves recording or sampling real instruments in high quality. Then when pressing a key in the virtual instrument, the sample is played back. It sounds simple but in general is not. You can record one sample for each note and assign them to their corresponding key. However, notes can be played at different lengths and there cannot be a sample for every possible note length. There needs to be a loop and getting near perfect loops can be difficult. But notes are not just loops. When a note is hit, they have an attack and when let go, a release. The attack needs to transition to the loop and the loop to the decay seamlessly. For many instruments, legato needs to be taken into account where notes played together directly after one another can affect the pitch of the attack and decay.

Generally, sampler instruments tend to have many samples per note, each corresponding to a certain part of the note. Of course, audio files are large and having lots of samples tends to make sampler instruments much bigger in size. On the other hand, they have better performance than the other calculation heavy synthesis. It is possible to reduce the amount of samples you need by using pitch shifting on a subset of samples. Notes that are one octave apart are either double or half the frequency of each other. It is fairly easy to do these pitch shifts as you either skip a sample point or double it. When trying to pitch to different notes, it creates much more artefacts and similarly does pitching up and down many octaves. It is possible to reduce the number of samples to a multiple of twelve corresponding to each note in the chromatic scale. It should be noted that pitching up will decrease the length of the sample and pitching down will increase it.

2.2.4 Other Synthesis

It is possible to create an instrument using additive synthesis. This involves having many waves, usually sine, and playing them all at once to create a

complex sound. There is a fundamental frequency carrying the note and different harmonics added at different amplitudes along the harmonic scale. According to the Fourier transform, a wave can be represented as a series of sine and cosine waves. Using the transform, it is possible to manually add those waves and there would be a point where it sounds good enough. It is uncertain how easy it would be to do this in practice

Another way that was not looked into, but could be possible is frequency modulation. This involves modulating wave forms by other wave forms. Frequency modulation is difficult to understand and visualize, so no research was done on it.

2.3 Frameworks

There are a few different interfaces that allow plugins to communicate with digital audio workstations(DAW). The most popular is VST or Virtual Studio Technology, developed by Steinberg, a well know virtual instrument and audio application developer. It is a cross platform interface developed for Steinberg's DAW, Cubase and later licensed to other plugin and DAW creators. The vast majority of instrument and effect plugins you can buy today will either be in VST2 or VST3 format.

The second most popular is Audio Unit (AU), originally developed by Apple for their Logic Pro DAW and is now integrated into MacOS and IOS. Many DAWs on MacOS both support VST and AU, and they are, for the most part, interchangeable. It is possible to develop the instrument directly using these instruments, implementing everything yourself. However, for the purpose of this project, higher level frameworks will be used.

The first framework that was found is iPlug2. It is a free and open source framework based on the original iPlug released in 2008, improving on interface design and adding support for distributed plugin formats. IPlug provides many classes for digital signal processing used in audio applications and user interfaces which can be compiled to both VST and AU, as well as stand-alone applications and the new Web Audio Module, allowing you to host instruments online. However, being reactively new, released in 2018, there is not much documentation available and to understand what the code does, you would have to dig through the source code. For someone completely new to audio programming, the lack of documentation and tutorials makes creating a virtual instrument in a reasonable amount of time difficult.

The second framework is Juce and this is the one that was chosen. It is

free to use for students, in open source projects and personally up to $50,000 in revenue and requires a license for other uses. As with iPlug, Juce can be compiled to many different plugin interfaces and a stand-alone application. What differs from iPlug is that Juce has much better documentation. There is an extensive number of classes that covers everything needed for audio programming and building user interfaces. Each class is documented and gives a good explanation of what each class does. On top of that, Juce is reactively popular in the audio programming community and there are numerous tutorials for this framework. The only downside is that these tutorials tend to be for the older Juce 5 when Juce 6 was released in June 2020 and the official tutorials used audio applications rather than plugins which have different formats.

3 Design

The uilleann pipes are to be implemented as

- Drone and chanter
- voices

3.1 Juce Plugin

The synthesiser was implemented as a Juce plugin. This is an application specifically designed for creating plugins for DAWs. It can be compiled to all the plugin formats as well as a stand-alone application, which is useful for testing. Juce uses standard C++.

The plugin is split into two sections: the plugin editor and the plugin processor. The plugin editor is where all the user interface elements go. The dimensions and colour of the plugin window is defined here. All on-screen controls like sliders and buttons are declared here as well as their size and position on screen. The plugin processor on the other hand is where all the sound processing happens. The processor has an input buffer, reads data if necessary, then writes the processed audio to said buffer. Generally the plugin processor calls other classes and functions rather than everything being written in the once processor file.

3.2 Juce Controls

- Enable, disable drone
- change key maybe

3.3 Samples

- 3 samplers, one for each components
- one sample can be pitched

4 Implementation

- classes
- juce plug-in layout
- Synthesiser and voices
- Param tree ==¿ parameter id all caps
- connection to parameter
- importing binary files
- buffers

4.1 Juce Plugin

2 sections

Plugin Editor

- UI
- buttons
- sliders
- Reference to processor

Plugin Editor

- UI
- buttons
- sliders
- Reference to processor

5 Evaluation

6 Conclusions

7 Appendix

- Appendix 1 - How to use Juce – How to setup vst?
- Appendix 1 - Music Example

8 Notes (temporary)

8.1 Things to explain?

Things that people might not understand and may need an explanation

- ADRS envelope – explain at the beginning or in subtractive, is in Sampling
- fundamental, harmonic, harmonic scale – in Other Synthesis
- does juce plugin description make sense and is it enough? – Juce Plugin 3.2

8.2 Improvements

- are there too many titles? hide subsections?
- analysis - talk more about aims. For example in Frameworks.

8.3 References Maybe

- samplers – https://ask.audio/articles/understanding-the-architecture-of-sample-based-virtual-instruments - how samplers work
- subtractive – Subtractive Synthesis Modelling of the Irish Uilleann Pipes - ISSTC 2012