The team I lead developed this VR video game called VRtist. VRtist was funded by KOCCA*(The Korea Creative Content Agency is a South Korean government agency which is affiliated with the Ministry of Culture, Sports and Tourism and is charged with governing cultural content).* VRtist is an environment where players can share artistic experience in VR. We needed to narrow down our focus because this was a short-term project from June to November 2021. So, we only developed DJing part.

This project was developed based on Unreal Engine(UE). Most of the functions are coded by me with C++ and then got integrated into UE to call them.

This is a multi-player game. A player can host his own server and other players can browse servers and join in. It uses steam advanced session. In a multi-player setting, each client sends their hand controllers’ world positions to the server and the server replicate all of each players’ hand controller position every frame so that each client can see another players’ hand controller.

When developing DJ Controller, I had a lot of obstacles because basic functions that Unreal Engine gives is not sufficient for me to develop good enough DJ Controller my team needed. So, I needed to study how other commercial software works to the low level.

The first problem I faced was that if you want to use wav file that’s in your computer instead of pre-imported data in the game package folder (in this case Unreal Engine automatically changes its format, not wav) when you play this DJ Controller, I needed to parse that wav file PCM data so that I can play it on the game environment in real-time. Fortunately, I found a plug-in that can parse any wav data (regardless of its BitsPerSample rate) into 32bit wav data format. However, since the parsed data is an unsigned integer, I needed to convert that data into float so that the data goes in between -1.0 and +1.0 to make DSP and visualization process a lot easier. So, I injected a block of code that converts uint8 into float64. And inserted the return data into my custom class that inherited by IsoundGenerator (from unreal engine) to do all the DSP works and finally produce actual sound in the game environment.

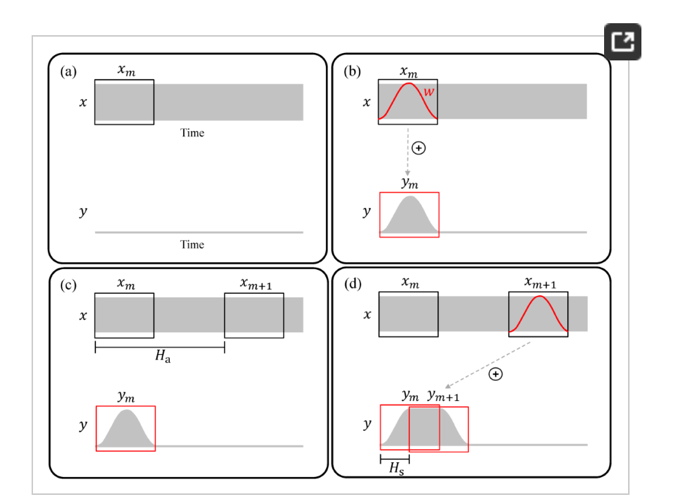
The second problem I faced was pitch shift and time stretch. UE DOES provide a similar function but pitch and playback speed is codependent. So, I had to learn about Phase Vocoder.

So, for the Pitch Shift, it goes,

Original sample

* Copy 1024 samples from original sample
* FFT
* Apply Phase Vocoder
* IFFT
* Write into Circular Buffer and move Write Pointer of circular buffer
* Feed speaker 1024 samples when it needs more data and move Read Pointer of circular buffer

And for the time stretch, I used Hann window. Before I copy 1024 samples from original sample, I decide where to get next 1024 samples to make it either faster or slower.



The third problem I faced was filter. Filter system UE provides works a little bit weird. Every time I adjust its frequency, it doesn’t just change its value, it deletes current filter and re-assign new filter with changed value. It sounded like ‘click, click’. So, I had to study biquad-fomulas and assigned that code right before the sample gets sent to the speakers. And for the Frequency input slider, the frequency value change has to be exponential, I used Bezier formula to obtain smoother change.

And then I used GCP to run GO HTTP Server and MongoDB database server. And connected those servers to the game so that users can store their customizing data under their steam account username.