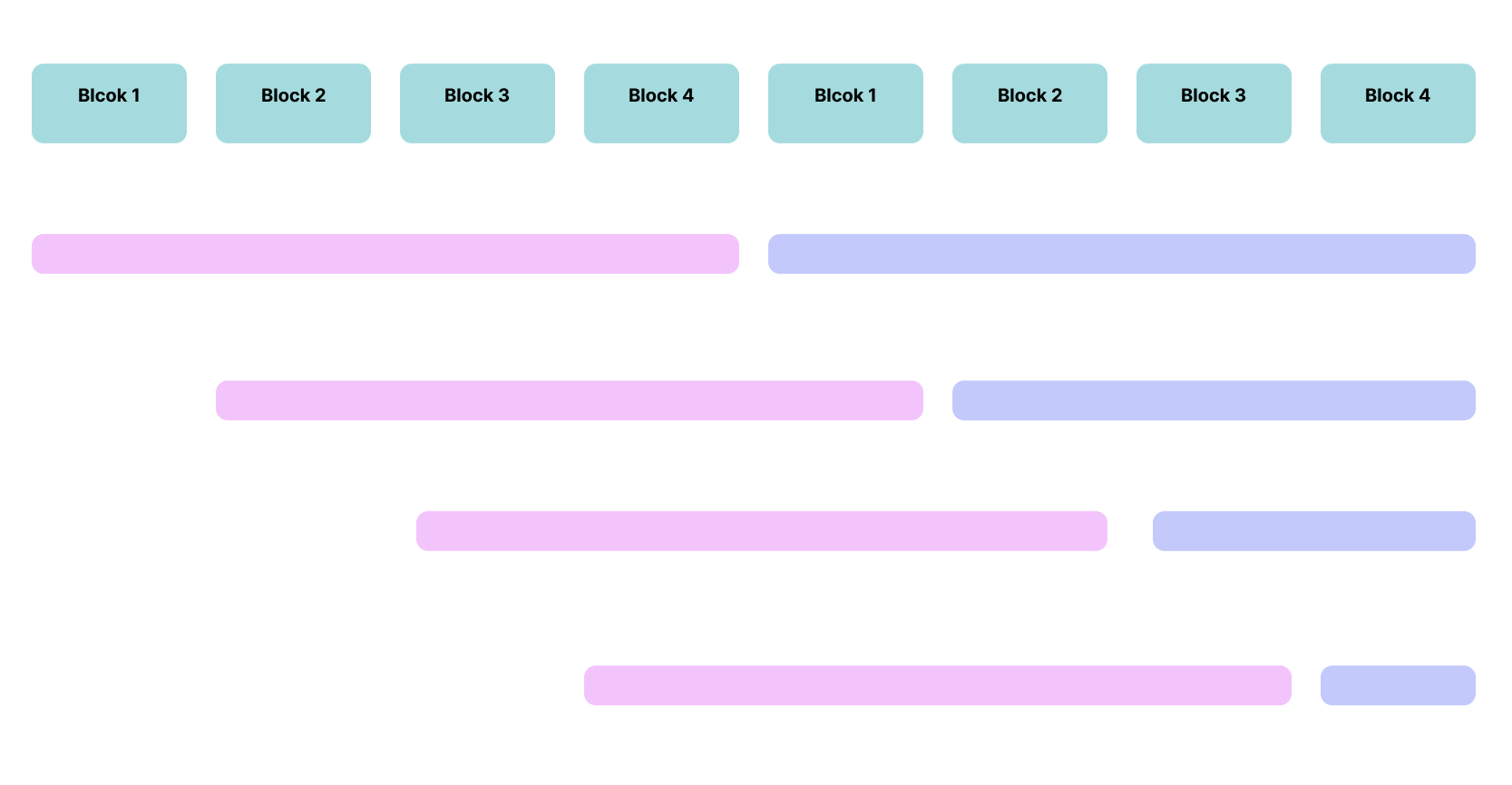
Description

I made a simple FFT pitch shift audio effect. I got FFT and IFFT algorithm from this [[1]](#footnote-1)article. It uses Recursive Cooley-Tukey FFT which provides optimized computation suited for real-time process. I overlapped four FFT blocks with ¼ Hop size. The FFT size will be the same as DAW’s I/O setting. So, a larger I/O rate that is at least 1024 is desirable for a better result. Since one FFT block is the same as one I/O block, it must add one block worth of delay to the sound.



Each process, it adds previous processed audio samples to the output buffer (from the second block 1 to block 3). This is processed data even before the pink blocks above. And then store current pink blocks of audio sample to the same position (from the second block 1 to block 3). Now, it is ready to process FFT with samples from the first block 1 to second block 3. Hann window is applied before and after the FFT process. Using this frequency domain data (std::complex), this plugin applies pitch shift with pitch parameter. The method was obtained from this [[2]](#footnote-2)tutorial video. And then fill each block with new input data from second block 1 to block 4 (color blue). When blue blocks 2, 3 and 4 are filled with new data, pink residual samples will be played (from at second block 1 to 3). There is a portamento in parameter change for a smoother effect. Important file to look at is “FFTPitchShift/source/processor.cpp”.

Github Link : <https://github.com/Toby-SeoKyeong-Kim/FFTPitchShift/tree/main>

1. Felipe, A.: Fast Fourier transform (2018), https://rosettacode.org/wiki/Fast\_Fourier\_transform#C++ [↑](#footnote-ref-1)
2. Bela Platform.: 20: Phase Vocoder (part 3), C++ Real-Time Audio Programming with Bela, Youtube Video, https://youtu.be/2p\_-jbl6Dyc?si=k1xaQeVlzOt9rFwU&t=1742 [↑](#footnote-ref-2)