Audio Engineering Society

Convention Paper 10708

Presented at the AES 156th Convention
2024 June 15–17, Madrid, Spain

This paper was peer-reviewed as a complete manuscript for presentation at this convention. This paper is available in the AES E-Library (http://www.aes.org/e-lib), all rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Remote Collaborative Mixing with 3D Audio

Daniel Faronbi[1] and Phil Brown[2]

[1]*Music and Audio Research Laboratory, New York University, New York, USA*
[2]*Dolby Laboratories, Inc., San Francisco, California, USA*

Correspondence should be addressed to Daniel Faronbi (danielfaronbi@nyu.edu)

ABSTRACT

Professional mixing and mastering of recorded audio is a challenging task. The process involves creating a mix that performs well across various listening environments, which can be quite cumbersome. With existing tools, audio engineers must render the audio, test it in different settings, and then remix it. This iterative process continues until the audio track is completed. The introduction of 3D spatial audio has further complicated matters, resulting in even more complex and varied mixes due to different speaker configurations. To address this, we propose an application that enables users to stream their mix in spatial audio to another location. By doing so, the mix can be evaluated simultaneously in multiple listening environments without the need for rendering and re-mixing. In this paper, we delve into the system design of our approach, compare it to similar products, analyze the time delay and synchronization of the audio and control stream, and examine potential race conditions when multiple users attempt to mix simultaneously.

1 Introduction

Tools for remote collaboration between audio engineers are limited [1, 2, 3]. In recent years, there have been online digital audio workstations (DAWs), audio recorder streamers, and high quality remote audio transfer tools. Although these tools have proven successful in improving the efficiency of some tasks in audio recording, current tools fail to provide remote mixing with 3D audio controls.

There is also a lack of tools used to aid spatial audio mixing. This is significant because, in the past decade, spatial audio has become more prominent in the process of music production. What used to be a niche field is quickly becoming an integral part of mixing audio. Spatial audio mixes have become especially accessible to listeners with the inclusion of Dolby Atmos in Apple Music streaming in 2021. Because of the growing popularity of spatial audio mixes, mixing and mastering engineers will need tools to be able to produce high quality spatial audio mixes. Mixing for a variety of speaker configurations has proven to be a challenge. With current tools, producers must mix a song, render the audio, then evaluate the mix in different environments. If a flaw in the mix is found, the producer must remix and re-render the audio and evaluate it in different listening environments again. This is, obviously, very cumbersome and non-intuitive.

Our goal was to find a solution to these common problems in the field of audio engineering. We wanted to make sure that our solution was:

- **Expandable to many platforms**: The solution needs to be built in a way that it can be easily ported from Mac to PC, to iOS, to android, etc...
- **Compatible with common standards**: The solution should not rely on custom or obscure standards for its communications to make the technology more approachable.
- **Scalable**: The solution should be built with the future in mind. Extra features should be able to be added fairly easily.

To address these problems, we created MixShare: a set of host and client applications that allow engineers to mix and evaluate audio in multiple speaker configurations at once. One app is a virtual instrument host that is used in conjunction with a DAW. The second app is a stand-alone client that receives audio from the host. Gain and 3D panning data is streamed bidirectionally between the two applications to allow for real-time collaborative mixing. MixShare is built using JUCE [4], which makes it easily expandable to many other platforms. It uses Dolby Atmos to handle object based spatial audio and UDP to handle sending web packets, which makes it compatible with common standards. Finally, the app was designed so that it could be easily tweaked to handle more channels or clients for scalability.

The rest of this paper will give an overview of spatial audio for context, go over the system design of MixShare, and evaluate the performance of MixShare by comparing it to past applications, observing the timing of audio and control stream packets, and evaluating race conditions when two users edit the same parameter at once.

2 Spatial Audio

2.1 Overview

Spatial audio aims to replicate a complete acoustic environment or synthesize a realistic new one [5]. Throughout the years, there have been a variety of different technologies developed to accomplish this goal [5]. These tools are used to either record or reproduce acoustic environments. The format for this differs depending on the context. In binaural settings, the recording and reproduction is tailored for two ears [6, 7]. Although these are normally meant for headphones [8, 9], they can also be used in loudspeakers [10, 11]. In soundfield settings, physical models are used to synthesize acoustic wave fields [12, 13, 14, 15]. This is usually done with recordings of spherical microphone arrays (like Eigenmike [16]) and multiple loudspeakers for reproduction. Ambisonics is widely used to encode and decode a variety of audio signals used to get 3D spherical coordinates [17].

2.2 Object Based Audio

A common method of reproducing spatial audio is object-based reproduction. In this format, an audio signal is combined with metadata describing its location [18]. This is in contrast to channel-based audio, where audio channels correspond to specific directions. The metadata describing the audio location is either included with the recording or created. The positional metadata is used to render the audio. How this rendering is performed differs depending on the speaker configuration. The rendering can be done by applying perceptual cues to the audio with head-related transfer functions (HRTFs) or multichannel surround systems [19, 20] or by generating a full sound field with Ambisonics [17] or Wave Field Synthesis [21].

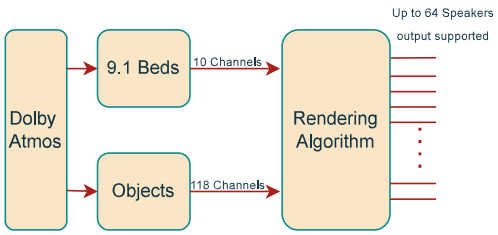


Fig. 1: Overview of Dolby Atmos Rendering. A total of 128 channels of audio are rendered in real time

2.3 Dolby Atmos

Dolby Atmos Platform encompasses a variety of products and technologies to deliver a high quality and flexible spatial audio experience [22]. It was originally created to support the immersive sound needs of cinema, but has more recently been applied to a variety of audio listening environments. When mixing in Dolby Atmos, there are two distinct audio track types: beds

and objects (fig 1). The 10 bed channels are tied to specific speakers in a 9.1 array (a 7.1 surround speaker array with 2 overhead speakers). Objects can be moved around freely in space. The rendering algorithm determines which speakers will be used to reproduce the sound object depending on its metadata. In practice, two pieces of Dolby software are often used to get spatial audio working with a traditional mixing setup: the Dolby Atmos Music Panner[23] and the Dolby Atmos Renderer[24] .

- The *Dolby Atmos Music Panner* is a virtual effect plugin used in DAWs that allows users to pan any audio channel in a three-dimensional audio field. The user can set the specific channel (11-128) of the audio data and the metadata is sent to the Dolby Atmos Renderer to be processed.
- The *Dolby Atmos Renderer* renders up to 128 inputs (beds and objects) and allows for monitoring and visualization of all audio objects in a scene. Although metadata is streamed automatically from the Dolby Atmos Music Panner, audio must be routed to either the Dolby Audio Bridge (on Mac) or Asio Link pro (on Windows) IO devices.

Some DAWs (Pro Tools Ultimate, Steinberg Nuendo, Apple Logic Pro Merging Pyramix and DaVinci Resolve) support Dolby Atmos natively. Most other DAWs require to use of both the panner and renderer for spatial audio mixing.

3 System Design

3.1 Overview

MixShare is made up of two individual applications: The host app and the client app (fig 2) The host app is a multi-input and output virtual effect that can be hosted by a DAW. Specific audio channels can be bused in and out of the host to be heard individually. The output channels must be sent to either the Dolby Audio Bridge (on Mac) or Asio Link Pro (on Windows) IO devices for the audio to be sent to the renderer.

The client app is a stand-alone application. The user must also select the aforementioned audio devices for the sound to be sent to the renderer. The user is asked to input the IP address to connect to the renderer. Once the device is connected, audio from the host is streamed to the client. Gain and pan parameters are also streamed bidirectionally. Each of the applications has multiple channels that have gain control, and a button to open the Dolby Atmos Music Panner (fig 3). When a user changes one of these parameters on the host app, an interrupt will stream the change to the client and vice versa. This ensures that mix parameters are synchronized between the two applications.

This application can be extended to support multiple clients, but for this prototype only one client is used. We also limited our test by only streaming 8 channels of audio over local host (shared Wi-Fi router) but intend to expand to accommodate more channels. Because we are streaming lossless pulse code modulation (PCM), there is an upper channel limit that is dependent on the router, sample rate and bit-depth.

3.2 Software Tools

We used the software framework JUCE [4] to create both applications. JUCE is a C++ framework that makes it easy to create audio plugins and stand-alone audio applications. It has a vast library for importing digital signal processing (DSP) functions as well as a wide variety of other libraries for commonly used functions in many applications (graphical user interface - also know as a GUI, features, sockets, timers, etc...).

3.3 Threads

Multithreading allows for a concurrent programming approach where multiple lines of code can be executed at once [25]. Threads run at the system level and allow one application to perform multiple tasks at the same time by quickly switching between different programming threads. By default, JUCE applications have separate threads to process audio and GUI changes. For our application, we add stream threads for audio and control messages.

On the host side, the audio stream thread sends sound from a FIFO buffer[26] to the client. This FIFO buffer is filled during the audio callback of the host application. The audio stream thread sends the audio data over a network to the client once the FIFO buffer is filled with at least 512 samples. Mutex locks are used to make sure the buffer contents are not being read at the

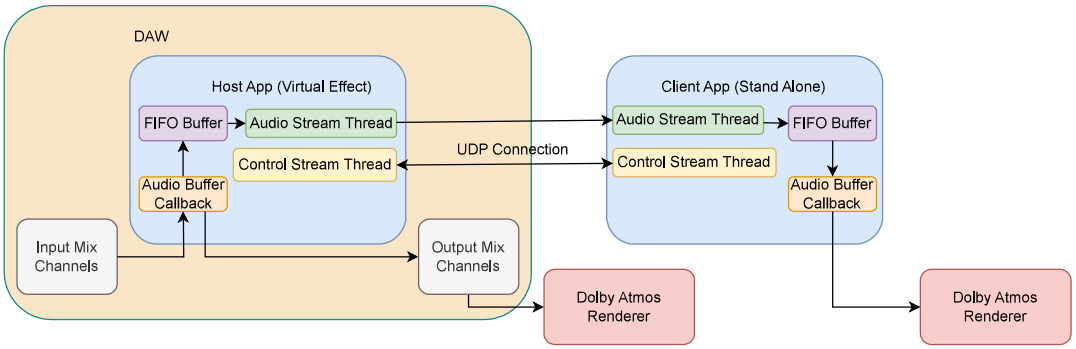


Fig. 2: MixShare streaming between host app and client app

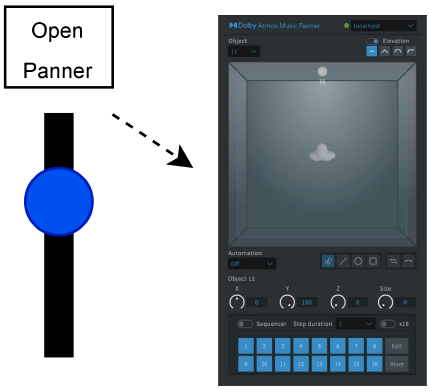


Fig. 3: Controls for each mix channel for each app. The (Open Panner) button reveals the Dolby Atmos Panner Window

same time it is being filled. Try locks (data locks that do not block the program) are used to be sure real time operation of the program is not halted. In the control stream thread, network messages are received and either the gain of a specific channel or a parameter of the hosted Dolby Atmos Music Panner is updated.

On the client side, the audio stream thread receives audio blocks from the network and fills the FIFO buffer. The audio buffer callback of the app fills the samples used by the program in real-time with samples from the FIFO Buffer (try locks are used again to avoid multi-access problems). The control stream thread of the client functions identically to that of the host.

3.4 UDP Streaming

To handle fast transfer of data through a network socket, the User Datagram Protocol (UDP) [27] was used to send our network messages between the two applications. This protocol was chosen in contrast to the Transmission Control Protocol (TCP) because it is faster (thus useful for real time streams) despite it being less accurate.

For programming with network sockets, they must be attached to specific network ports. Both the host and client apps use three different ports to transfer audio. The ports for sending connection messages (The initial message to establish a connection between client and host), audio samples, and control messages (both gain and panning controls).

4 Performance Analysis

To examine the performance of our application, we compare it to previous methods that attack similar problems, analyze the time synchronization of the stream, and predict race condition problems if multiple clients were added to the protocol.

4.1 Comparison to Previous Approaches

To our knowledge, an application for collaboratively streaming spatial audio mixing audio and controls is almost completely novel. However, there have been past applications that have tackled similar problems:

- *Source-Connect Pro [28]* is a monthly subscription service available on macOS and Windows that is tailored for remote recording sessions with high quality audio (up to 192kHz sample rate). It can connect to up to 4 other computers and send audio in full duplex. It can also be used for timeline locking. However, it lacks the ability to control mixer faders.
- *VST Connect Pro [29]* is a product by Steinberg that allows for streaming of high quality audio and midi of multichannel audio (up to 16 tracks) between two devices.
- *Avid Cloud Collaboration [30]* is a tool that allows for easy cloud collaboration with other engineers using Avid Pro Tools Artist, Pro Tools Studio, or Pro Tools Ultimate. You can share midi tracks, edits, and mix changes. However, the mix parameters are not changed in real time.
- *Soundtrap [31]* is an online DAW with built-in collaboration features. The DAW is fully featured with virtual instruments, recording, and audio editing. All of these can be changed synchronously by multiple users.
- *Audio Movers [32]* is an application that allows streaming of audio or midi to up to 150 users from a VST, DAW, or browser. It also supports Dolby Atmos (pre-prints not full object metadata).
- *BandLab [33]* is a free online DAW also available on Android and IOS. It allows for basic mixing and audio editing features, so simple songs can be produced. It has synchronized collaborative features that work with almost all the parameters of the DAW.
- *Soundwhale [34]* is a collaborative production environment designed for multiple collaborators to listen, record, and communicate. The audio is streamed uncompressed and virtual sessions can be joined to easily produce music with others.
- *Lisento [3]* is a browser-based application that allows for collaboration in real-time in a fully-featured online DAW using Web Real-Time Communication. The app also allows for videoconferencing and control data streaming.
- *Sessionwire Studio [35]* is a collaboration platform specifically made for producers and songwriters. It allows for streaming of video, voice, text, files, storage, and high quality audio for a seamless collaborative experience.
- *ConnectionOpen [36]* is a voice focused collaboration app that allows for uncompressed audio to be streamed with minimal latency. Webcam video and screen-sharing are also included as part of the application.
- *SoundNation [37]* is a browser based DAW with collaboration tools. Users can change mixes, instruments, and audio edits in real time.
- *DAW Collaboration Framework (DCF) [2]* is an approach to collaborative music making that streams low bandwidth control data to allow for many users to receive high-resolution audio.

We would like to compare these applications to our new approach. We came up with 6 desirable features that a collaborative mixing app should have and analyze the presence of these features in all of these approaches. These are *Real Time* (does the app stream audio instantly?), *Lossless* (does the app stream uncompressed audio?), *Multi-Client* (does the app can stream to more than 1 client?), *Mix Controls* (does the app stream mixer controls?), *Spatial Audio* (does the app stream spatial audio content?), and *Extra Communication* (does the app include text, voice, or video communication in addition to audio stream?). We analyze each app and take note of whether it does or does not have each of these features and plot the results in Table 1.

Despite the wide variety of tools available for streaming production audio, our approach is the only one that

App	Features					
	Real Time	Lossless	Multi-Client	Mix Controls	Spatial Audio	Extra Communication
Source-Connect Pro	✓	✓	✓			
VST Connect Pro	✓	✓				
Avid Cloud Collaboration		✓				✓(text)
Soundtrap			✓	✓		✓(video and text)
Audio Movers	✓	✓	✓		✓(only preprints)	
Band Lab	✓		✓	✓		✓(text)
Sound Whale	✓	✓				✓(video, audio, and text)
Listento	✓	✓	✓			✓(video)
Sessionwire Studio	✓	✓				✓(screen, video, text)
Connection Open	✓	✓				✓(video, screen, text)
Sound Nation	✓	✓	✓	✓		✓(chat)
DCF	✓	✓	✓	✓		✓(video)
MixShare	✓	✓		✓	✓	

Table 1: Feature comparisons between collaborative mixing apps throughout the years. A wide variety of features have been explored to improve the art of mixing, but none have fully implemented spatial audio.

fully supports spatial audio. Audio Movers has some support, but the client users do not have the ability to change the spatial properties of the sound object. Our approach is novel because both the host and client can alter the spatial properties of an audio signal and hear the changes in real time for the first time.

4.2 Time Synchronization

To ensure that our application is usable, we need to make sure that the host and client send information to each other quickly enough that mixing between the two appears seamless. For a control change to appear seamless, it would have to occur before the next audio buffer call back to the app it is being streamed to. During this call back, all the processing for the audio is done, so it is important that the correct parameters are set for the audio to be processed properly. For our tests, our audio block-size is 512 samples and our apps are running at a sample rate of 48 kHz. This means there are 93.75 blocks processed every second, so there is an audio call back almost every 10 ms. We tried our app on an internet connection with an upload speed averaging about (1 KB/s) and a download speed averaging about (2 KB/s). Control change messages were received within 10 ms and the majority of audio stream packets were sent without hiccups. A timing diagram of an ideal performing host and client app is shown in fig 4.

4.3 Race Conditions

Finally, we discuss what happens when multiple users attempt to alter the same parameter at once. Let us consider a change in gain control of a slider. Control stream changes trigger an interrupt that updates the status of the slider regardless of the user input. In the meantime, the user input change is streamed to the other app. Because of this, the values of the host and the client would be different. In this special case, the mixes cease to be synchronized as seen in fig. 5. However, the probability of a stream change and user change happening at the exact same time is very low.

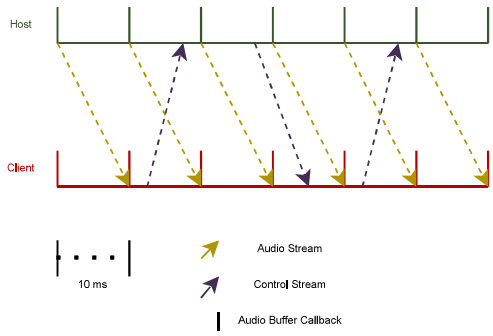


Fig. 4: Timing diagram of streaming content between host and client MixShare apps. With the right upload connection speed, control messages are sent in time (before 10 ms), so the next audio block on both apps is processing the audio block the same way.

In most cases, one change will finish before the other and will synchronize the mixes on both applications.

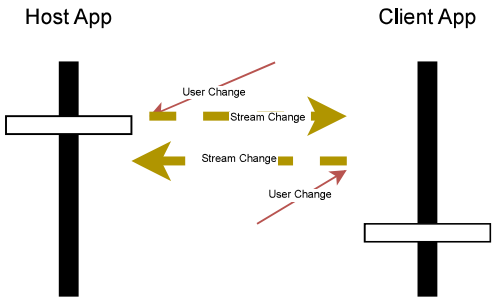


Fig. 5: When a user and stream change are received at the same time, stream changes are prioritized. This results in the host and client app having separate values (an un-synchronized mix).

5 Conclusion

In summary, we have presented our contribution to remote collaborative mixing tools. MixShare is the first tool to completely support synchronous mixing with spatial audio using Dolby Atmos. This basic version of the app proves that the concept works. However, there are many improvements that could be made. More features could be added that aid collaborative mixing (video, audio, and screen share). We could also allow the app to stream to more than a single client. Despite some of these shortcomings, we believe MixShare is a great step forward in mixing tools for audio engineers and look forward to seeing how it can be improved upon in the future.

References

[1] S. Stickland, R. Athauda, and N. Scott, "Professional Views of Digital Audio Workstations and Collaborative Audio Mixing," *Chroma: Journal of the Australasian Computer Music Association*, vol. 38, no. 1 (2022).

[2] S. Stickland, R. Athauda, and N. Scott, "Design and evaluation of a scalable real-time online digital audio workstation collaboration framework," *Journal of the Audio Engineering Society*, vol. 69, no. 6, pp. 410–431 (2021).

[3] S. Stickland, R. Athauda, and N. Scott, "Design of a real-time multiparty DAW Collaboration Application using Web MIDI and WebRTC APIs," presented at the *Proceedings of the International Web Audio Conference*, pp. 59–64 (2019).

[4] JUCE, "JUCE," URL `https://juce.com/`, (accessed on Feb. 27th, 2024).

[5] W. Zhang, P. N. Samarasinghe, H. Chen, and T. D. Abhayapala, "Surround by sound: A review of spatial audio recording and reproduction," *Applied Sciences*, vol. 7, no. 5, p. 532 (2017).

[6] D. Hammershøi and H. Møller, "Methods for binaural recording and reproduction," *Acta Acustica united with Acustica*, vol. 88, no. 3, pp. 303–311 (2002).

[7] H. Møller, "Fundamentals of binaural technology," *Applied acoustics*, vol. 36, no. 3-4, pp. 171–218 (1992).

[8] R. Ranjan and W.-S. Gan, "Natural listening over headphones in augmented reality using adaptive filtering techniques," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1988–2002 (2015).

[9] H. Jianjun, E. L. Tan, W.-S. Gan, *et al.*, "Natural sound rendering for headphones: integration of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 100–113 (2015).

[10] B. B. Bauer, "Stereophonic earphones and binaural loudspeakers," *Journal of the Audio Engineering Society*, vol. 9, no. 2, pp. 148–151 (1961).

[11] Y. Huang, J. Benesty, and J. Chen, "On crosstalk cancellation and equalization with multiple loudspeakers for 3-D sound reproduction," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 649–652 (2007).

[12] T. D. Abhayapala and D. B. Ward, "Theory and design of high order sound field microphones using spherical microphone array," presented at the *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. II–1949 (2002).

[13] J. Meyer and G. Elko, "A highly scalable spherical microphone array based on an orthonormal decomposition of the soundfield," presented at the *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. II–1781 (2002).

[14] M. A. Poletti, "Three-dimensional surround sound systems based on spherical harmonics," *Journal of the Audio Engineering Society*, vol. 53, no. 11, pp. 1004–1025 (2005).

[15] P. N. Samarasinghe, T. D. Abhayapala, and M. A. Poletti, "Spatial soundfield recording over a large area using distributed higher order microphones," presented at the *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 221–224 (2011).

[16] MHAcoustics, "EigenMike," URL https://mhacoustics.com/products#eigenmike1, (accessed on Feb. 27th, 2024).

[17] E. G. Williams and J. A. Mann III, "Fourier acoustics: sound radiation and nearfield acoustical holography," (2000).

[18] C. Kim, "Object-based spatial audio: concept, advantages, and challenges," in *3D Future Internet Media*, pp. 79–84 (Springer, 2013).

[19] D. R. Begault and L. J. Trejo, "3-D sound for virtual reality and multimedia," Tech. rep. (2000).

[20] J. Blauert, *Spatial hearing: the psychophysics of human sound localization* (MIT press, 1997).

[21] A. J. Berkhout, D. de Vries, and P. Vogel, "Acoustic control by wave field synthesis," *The Journal of the Acoustical Society of America*, vol. 93, no. 5, pp. 2764–2778 (1993).

[22] E. Pfanzagl-Cardone, "The DOLBY®"Atmos™" System," in *The Art and Science of 3D Audio Recording*, pp. 143–188 (Springer, 2023).

[23] D. A. Panner, "Dolby Atmos Panner," URL https://professional.dolby.com/product/dolby-atmos-content-creation/dolby-atmos-music-panner, (accessed on May. 8th, 2024).

[24] D. A. Renderer, "Dolby Atmos Renderer," URL https://professional.dolby.com/product/dolby-atmos-content-creation/dolby-atmos-renderer, (accessed on May. 8th, 2024).

[25] A. D. Birrell, *An introduction to programming with threads* (Digital Systems Research Center, 1989).

[26] Q. P. Herr and P. Bunyk, "Implementation and application of first-in first-out buffers," *IEEE transactions on applied superconductivity*, vol. 13, no. 2, pp. 563–566 (2003).

[27] L. Kalita, "Socket programming," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 3, pp. 4802–4807 (2014).

[28] SourceConnetPro, "Source-Connect Pro," URL https://store.source-elements.com/subscription-options/source-connect-pro-subscription, (accessed on Feb. 27th, 2024).

[29] Steinberg, "VSTConnect," URL https://www.steinberg.net/vst-connect/, (accessed on Feb. 27th, 2024).

[30] Avid, "CloudCollaboration," URL https://www.avid.com/pro-tools/cloud-collaboration, (accessed on Feb. 27th, 2024).

[31] SoundTrap, "SoundTrap," URL https://www.soundtrap.com/musicmakers, (accessed on Feb. 27th, 2024).

[32] AudioMovers, "AudioMovers," (2023 Sep), URL https://audiomovers.com/, (accessed on Feb. 27th, 2024).

[33] BandLab, "BandLab," URL https://www.bandlab.com/?lang=en, (accessed on Feb. 27th, 2024).

[34] SoundWhale, "SoundWhale," URL https://soundwhale.com/, (accessed on Feb. 27th, 2024).

[35] SessionWire, "SessionWire," URL https://www.sessionwire.com/, (accessed on Feb. 27th, 2024).

[36] ConnectionOpen, "ConnectionOpen," URL https://www.connectionopen.com/, (accessed on Feb. 27th, 2024).

[37] Soundation, "Soundation," URL https://soundation.com/, (accessed on Feb. 27th, 2024).