Participatory musical improvisations with Playsound.space

Ariane Stolfi
Music Department, ECA USP
arianestolfi@gmail.com

Alessia Milo
Centre for Digital Music, Queen Mary University of London
a.milo@qmul.ac.uk

Miguel Ceriani
Centre for Digital Music, Queen Mary University of London
m.ceriani@qmul.ac.uk

Mathieu Barthet
Centre for Digital Music, Queen Mary University of London
m.barthet@qmul.ac.uk

ABSTRACT

Playsound.space is a web-based tool to search for and play Creative Commons licensed-sounds, which can be applied to free improvisation, experimental music production and soundscape composition. It provides fast access to about 400k non-musical and musical sounds provided by Freesound and allows users to play/loop single or multiple sounds retrieved through text-based search. Sound discovery is facilitated by the use of semantic searches and sound visual representations (spectrograms). After feedback gathered from user tests and practices with the tool as an instrument, we identified several directions to develop the expressive and collaborative capabilities of the tool. We present additional features for more complex audio processing, and also to enhance participation trough a chat system that allows users to share sound sessions and exchange messages while playing.

1. INTRODUCTION

Playsound [21] is a music making tool that works by querying sounds from the online sound database Freesound.org. Until the development of sound recording systems in the early 20th century, experience of music only occurred through situated performance [15]. Traditionally, the ability to perform classical music requires to gain sufficient theoretical and technical skills to understand, decode and interpret written music through the practice of a musical instrument. Such acquisition of musical knowledge and the cost of musical instruments can be seen as barriers to participate in music making activities.

Despite the process of digitization of music, which brought access to music production for a larger number of users together with the dissemination of personal computers, most interfaces for music making are complex to operate [23], or too incipient in terms of music expressiveness, acting more like toys [18]. Little attention has been paid for novice musical practitioners [19]. Our first design goal is to develop a music making tool providing access to a rich palette of sounds that can be used by people unfamiliar with music production techniques. The system we developed, Playsound, relies on the use of semantic queries as a means to access sound content.

2. DESIGN

Playsound was developed to support the main author's musical practice, including experimental music and free improvisation, with emphasis on the playing process [8] in musical performance. The main idea for the development of the tool was to build a platform that could allow playing, looping and processing sound samples from the Freesound.org online database, a content provider part of the Audio Commons Ecosystem [13] developed by the UPF-Music Technology Group. It was designed to be used in live performance, so that the musician/performer could access huge number of sounds without having a large personal sound collection. It also derived from a difficulty found in using samples during live performances, specially when browsing for sounds in large sound databases, where besides the information contained in the bibliographic metadata, not much describes the creative content of the sounds. Therefore, users generally require to listen to a large amount of sounds in order to choose the ones that can satisfy their needs. This can be a special problem in contexts such as free improvisation, where the sound response needs to be quite immediate for the musician.

One of the main features of Playsound is to offer a visual tool for searching sounds, such as fast access to about 400k non-musical and musical sounds from Freesound, relying on the spectrogram graphics already provided by the API [3], enabling to present the results as a collection of images. After acquiring experience in reading this spectrogram representations, it is then possible to identify some aspects of the sonic quality and form impressions of how given samples will sound like before playing them, especially in terms of timbre [5] and texture [17].

Web Audio Conference WAC-2018, September 19–21, 2018, Berlin, Germany.

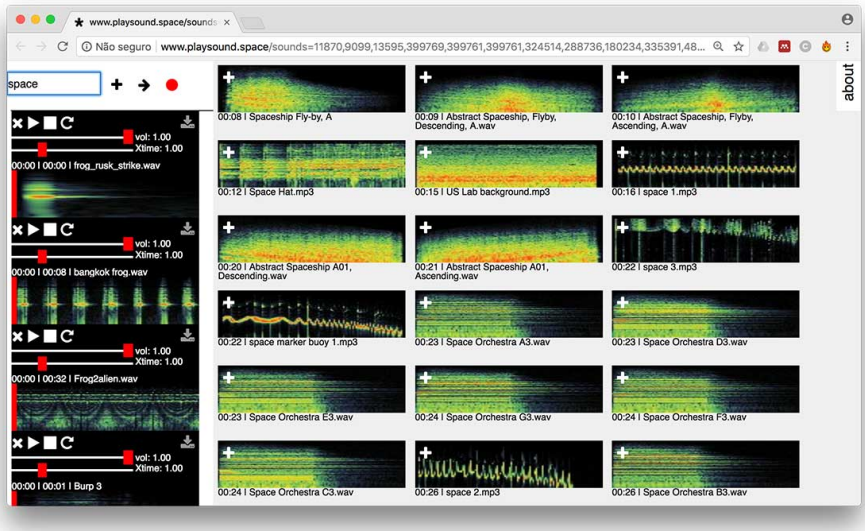

Figure 1: Screenshot of the tool

As an instrument, Playsound can be used to support collocated musical performance by running on different browsers on the client side. In the future, we would like to connect clients over the web to share a same musical session, in a context of networked performances similar to those described by [14] and [4] (see e.g. [20] for a review on network music performance technologies).

2.1 Development

We followed an inclusive design approach to create a tool that could suit both non musicians and musicians. Software development followed the agile methodology with an initial phase supported by frequent feedback from six users working in media and arts technology and performing arts. The prototype was then assessed by 18 non musician and musician participants during free music improvisation and live soundscape composition sessions lasting 15 to 30 mins [21]. The evaluation was based on several human computer interaction frameworks including the system usability scale (SUS) [10] and the creativity support index (CSI) [11]. We also conducted inductive thematic analyses [9] on focus group discussions and self-reports related to workflow, hedonic quality, engagement, learning, contexts of use and improvements, applied by online questionnaires after the music sessions. The prototype obtained high usability (M = 82.5/100, SD = 8.94) and creativity support index scores (M =71.7, SD =15.6) and no significant differences were found between non musicians and musicians. Analyses of log behavioral data indicated that all the participants engaged creatively during the sessions: number of sound queries (M=8, SD=2) and number of sounds played and repetitions (M= 24, MIN=9, MAX=101, SD=18). The results favorably supported the inclusive design goal as the prototype proved to enable engaging creative musical collaborations without requiring prior musical skills. Moreover, we identified further design challenges both at prototype and project levels. The Playsound prototype could provide better audio expressive controls (e.g. volume, loop, effects, timing) and improve the sense of identification of played content and co-presence between performers. To improve sound retrieval, participants wished to have access to filtering and clustering techniques and to be able to search for sounds by features, e.g. by timbre.

2.2 User Interaction

To use Playsound, the user needs a browser compatible with HTML5 and the Web Audio API[1]. The interface is currently optimised for the Google Chrome browser. It does not require authentication or any license and is opened to everyone. Users just need to type in the URL *playsound.space* in the browser window to access the site. A demo video for the tool is available at: https://youtu.be/yv8T70rawzs.

In the upper left corner of the website, as displayed in Figure 1, there is an input field for searching sounds. When the user types an arbitrary search query, the results are shown in the right side of the page as a collection of spectrograms. By clicking on a play icon on the image result, the corresponding sound starts to play and a related player is shown in the left column, below the search result. Any sound can be played and paused individually, whilst the curved arrow below the play button triggers the loop, on and off, for each sample. The first version allowed only the playback, pause, and looping of the sounds.

After the user tests, as well as performance practices with the instrument, we identified that it would be important to have more control on the sound processing. To this end, we implemented: a queue function, that sends by default the sound to the playlist without playing, by clicking on the image, allowing the user to choose the desired speed and volume before playing the file. We also implemented a playback rate control that allows to individually change the speed of the samples.

Besides the search input there is a plus button, which opens a new page with an empty query. When the search has more than 40 results, navigational arrows are also displayed in the menu for browsing into the search results. A built-

in recorder generates a Wav file directly from the browser, that can be also further layered with other sounds the user is playing.

Since we aim at developing tools to facilitate correct licensing for the generated work, we provide a list of credits for each sound that requires attribution. Sounds from the Freesound library can be used in various ways depending on requirements from the Creative Commons licenses they are assigned to[1]. Thus, to avoid misuse, we exclude sounds that could not be remixed, tweaked, or built upon commercially, and provide access to sounds in the public domain (cc0) or with the CC Attribution license (cc-by), which allows transformations and reuse, provided that credit is given to the original author. In order to reference completely the source, we implemented a link to access the original file uploaded to freesound.org, to allow the user to download the sound at the original or desired quality. To save a selection of samples, the user just needs to save the URL generated by the queries. Once the page is loaded, all the sounds will be loaded in the left column of the site.

2.3 System Architecture

The system works as a single page application[16] programmed mostly in JavaScript, using the Angular.js framework on the client side and Node.js on the server side. The two way data binding provided by the Angular framework allows a fast response from the Freesound REST API, meaning results can be obtained directly while typing a search query. The majority of processing happens on the client side and the back-end part of the code deals only with the authentication process with the Freesound API and the retrieval of the information from the REST API. Recent lines of development have seen also the implementation of a sound recommendation system [24] using similarities based on the spectral centroid (see e.g. [6] for a discussion on the spectral centroid). We also integrated a translation feature [22] and a chat system relying on sockets.

When a sound is loaded in the playlist, the identifier (ID) from Freesound is added to the URL and a custom object is generated with the corresponding controls. Sounds can be also removed from the playlist, or faded using the volume controls. In the current development of the system, the main page serves as a single player tool, to explore the different sounds generated by the query and mix them in a creative way. With respect to the version developed during the user testing in [21], additional features were developed such as a seek cursor which updates the position of the playhead along the sound duration. The cursor is also updating as the sound plays, giving feedback to the user on how fast the sound is being played. This simple feedback from the playback, together with the manipulation of the playback rate, is visualised directly on the image of the sound loaded, allowing the user to identify the different sounds and mixing them during their rendering. Particular care has been taken into making the application scalable both for the use on a computer or a smartphone, customising the CSS layout with visible coloured elements and text showing the information about the sound and the controls values beside the controls. After implementing a custom loop selection and a panning slider, the interface was redesigned to show panning and playback rate only when hovering on the icon of a menu, next to the other controls.

[1]https://creativecommons.org/licenses/

2.4 Audio Processing

The audio processing is relying on the Web Audio API to connect the sounds together in the `AudioContext` through multiple `GainNode` objects. The system, originally based on individual HTML media players, employing `MediaElementAudioSourceNode`, was changed to allow the control of the `AudioBuffer` and the `AudioBufferSourceNode`, as referenced in the Web Audio API [1]. Many properties from the HTML player in the original version were found convenient for our uses, such as `loop`, `playbackRate`, `volume`, `currentTime`, and event listeners allowing immediate binding with AngularJS. However, in order to answer the desire of manipulating the buffers without having to rely on the media object, we decided first to replace the layout of the interface of the original HTML object, hiding its controls, then we ported all the properties of the player to the `AudioBufferSourceNode`. The current system available at Playsound.space is employing the `AudioBuffer` object, triggering a secondary Ajax request to retrieve the sound file from Freesound, decoding the data with `audioContext.decodeAudioData()`, creating an `AudioBufferSourceNode` and passing the buffer to the Node.

Additionally, a `StereoPanner` Node allows the user to select the panning position of each sound in the master mix, effectively remixing sounds on the fly in the browser. This mixing feature, together with the possibility to record the audio in a wave file storing the IDs of the sounds, and a master volume control on the top, enhances the potential that the instrument offers to play live and document the performances taking place.

We decided to avoid external libraries to manipulate the buffers to allow the buffering processes to be exposed in their simple administration. We noticed that simple tools available in the `HTMLAudioElement` as the ability to play and pause, or seek using the `currentTime`, became more difficult to manage when using the `AudioBuffer`, specifically due to the dependency of the `SourceNode` from the `AudioContext` such as the necessity of calculating during the playback the time elapsed from the beginning of the creation of the `SourceNode`, which happens when a sound is clicked. Moreover the Event Listeners available on the `HTMLAudioElement`, retrievable also when the original controls are hidden, are to our knowledge not fully supported on the `AudioBufferSourceNode`, which might have encouraged others to develop additional libraries for playback and manipulation.

In the player interface we implemented a slider to control playback speed, draggable cursors to set the loop size, plus panning and volume control on the sound, as described in [22]. Every player instance then connects to the final destination, whose gain is controlled by a master volume slider, and the session is rendered on the client side in the browser. After porting the functionalities of the original HTML player to the `AudioBuffer` and testing them in real use cases, we noticed, especially with slow network, that the `AudioBuffer` was less convenient when the sounds loaded presented a longer duration. Knowing that the HTML Audio Element offers the possibility of playing the sound while it loads in the buffer, we are currently considering to manage the samples differently according to their duration.

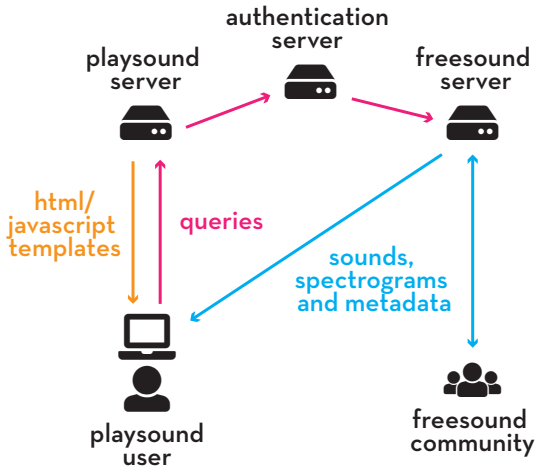

Figure 2: System Diagram

2.5 Code, Participatory Chat and Recommendation

Playsound.space is an open source project and its supporting code can be accessed at https://github.com/arianestolfi/audioquery-server. The new features presented here, allowing more control of the playback of the different sounds, following a series of observations during musical sessions between users, can be trialled by playing with the system at the URL Playsound.space.

Technology-mediated participatory music is a growing field investigating the design of systems to include audiences in live music performance (see e.g. [25] for a review). The addition of collaborative features into Playsound, such as a multi-user chat, enables to use the tool in a participatory context. The chat feature, together with the custom definitions for the audio buffers, has first been implemented in the development branch https://github.com/arianestolfi/audioquery-server/tree/chat and employs the `socket.io` [2] library for Node.js to allow users to exchange text messages while visiting the corresponding URL. The chat environment can be experienced at Playsound.space/chat. The text delivered through the messages is turned into hypertext and can trigger queries to the Freesound database upon click.

Audio content recommendation has been traditionally applied for the selection of recorded content by media producers [7] [12] but it can also be applied to enrich the selection of sounds during the compositional stage. In order to improve the relevance of the system, we implemented a recommendation system [24] to search into other Audio Commons resources (e.g. Jamendo[2], Europeana[3]) using semantic web technologies, and to find similar sounds using acoustical similarity measures. To improve accessibility, we included a translation feature using Yandex to translate queries from other languages [22].

[2]https://www.jamendo.com/
[3]https://www.europeana.eu/portal/en

3. FUTURE WORK

Playsound can be considered as a work-in-progress prototype and new features are developed as new requirements are outlined by its users. We identified several lines of development which could improve the usability of the interface. The next priorities for the development of the system should be first to enhance the chat system to allow the users to create different rooms to chat and share sounds, and second to interact within the same sound session; this development is aimed at fostering not just collocated but also networked performance. As we already implemented envisioned features such as a recommendation system, a translation service, panning and loop manipulation, we would like to focus on testing connection speed differences in the playback of the content using AudioBuffers in comparison to the HTMLMediaElements, as the latter might be more convenient for samples with a duration longer than one minute.

4. CONCLUSION

We discussed hereby novel features of *Playsound.space*, a web-based tool to query and play Creative Commons licensed-sounds, suitable for free improvisation, experimental music production and soundscape composition. These features, including playback rate control and a seekable cursor placed on the spectrogram of the sound, provide additional control for the manipulation of about 400k non-musical and musical sounds provided by Freesound, allowing users to play/loop single or multiple sounds retrieved through text-based search. Whilst the sound discovery is facilitated by use of semantic searches and sound visual representations (spectrograms), participation in collaborative musical explorations is being supported by the introduction of an additional chat system allowing the users to exchange messages during the sonic interactions.

5. ACKNOWLEDGMENTS

We acknowledge support from University of São Paulo's Nu-Som Research group and the CAPES PDSE grant awarded to Ariane Stolfi. This work is also supported by the EU H2020 Audio Commons Initiative grant (No. 688382).

6. REFERENCES

[1] Web audio API W3C working draft. https://www.w3.org/TR/webaudio/, 2015.
[2] Socket.IO. https://socket.io/, 2018.
[3] V. Akkermans, F. Font Corbera, J. Funollet, B. de Jong, G. Roma Trepat, S. Togias, and X. Serra. Freesound 2: An improved platform for sharing audio clips. *ISMIR Conference Proceedings*, 2011.
[4] J. J. Arango, M. Tomoyoshi, F. Iazzetta, and M. Queiroz. Brazilian challenges on network music. In *Proc. of the Sound And Music Computer Conf. (SMC)*, pages 1–7, 2013.
[5] M. Barthet, P. Depalle, R. Kronland-Martinet, and S. Ystad. Acoustical correlates of timbre and expressiveness in clarinet performance. *Music Perception: An Interdisciplinary Journal*, 28(2):135–154, 2010.
[6] M. Barthet, R. Kronland-Martinet, and S. Ystad. Improving musical expressiveness by time-varying

brightness shaping. In *International Symposium on Computer Music Modeling and Retrieval*, pages 313–336. Springer, 2007.

[7] C. Baume, G. Fazekas, M. Barthet, D. Marston, and M. Sandler. Selection of audio features for music emotion recognition using production music. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society, 2014.

[8] C. Bergstrøm-Nielsen. Keywords in musical free improvisation. *Music and Arts in Action*, 5(1):11–18, 2016.

[9] V. Braun and V. Clarke. Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2):77–101, jan 2006.

[10] J. Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[11] E. Cherry and C. Latulipe. Quantifying the Creativity Support of Digital Tools through the Creativity Support Index. *ACM Transactions on Computer-Human Interaction*, 21(4):1–25, jun 2014.

[12] G. Fazekas, M. Barthet, and M. B. Sandler. Demo paper: The bbc desktop jukebox music recommendation system: A large scale trial with professional users. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–2. IEEE, 2013.

[13] F. Font, T. Brookes, G. Fazekas, M. Guerber, A. La Burthe, D. Plans, M. D. Plumbley, M. Shaashua, W. Wang, and X. Serra. Audio commons: bringing creative commons audio content to the creative industries. In *Audio Engineering Society Conference: 61st International Conference: Audio for Games*. Audio Engineering Society, 2016.

[14] R. Haefeli. netpd-a collaborative realtime networked music making environment written in pure data. In *Linux Audio Conference 2013*, volume 1. Citeseer, 2013.

[15] F. Iazzetta. A Música, o Corpo e as Máquinas. *Revista Opus*, 4:1–20, 1997.

[16] M. A. Jadhav, B. R. Sawant, and A. Deshmukh. Single page application using angularjs. *International Journal of Computer Science and Information Technologies*, 6(3):2876–2879, 2015.

[17] L. Lu, L. Wenyin, and H.-J. Zhang. Audio textures: Theory and applications. *IEEE transactions on speech and audio processing*, 12(2):156–167, 2004.

[18] J. McDermott, T. Gifford, A. Bouwer, and M. Wagy. Should Music Interaction Be Easy? 2013.

[19] E. M. Miletto, M. S. Pimenta, F. Bouchet, J.-P. Sansonnet, and D. Keller. Principles for Music Creation by Novices in Networked Music Environments. *Journal of New Music Research*, 40(3):205–216, sep 2011.

[20] C. Rottondi, C. Chafe, C. Allocchio, and A. Sarti. An overview on networked music performance technologies. *IEEE Access*, 4:8823–8843, 2016.

[21] A. Stolfi, M. Ceriani, L. Turchet, and M. Barthet. Playsound.space: Inclusive Free Music Improvisations Using Audio Commons. In *Proc. Nime*, 2018.

[22] A. Stolfi, V. F. Milo, Alessia, M. Ceriani, and M. Barthet. Playsound.space: An Ubiquitous System in Progress. In *Proc. 8th Workshop on Ubiquitous Music*, 2018.

[23] A. D. S. Stolfi. Graphic Interfaces for Computer Music: Two Models. In *CMMR*, pages 1–8, 2016.

[24] F. Viola, A. Stolfi, A. Milo, M. Ceriani, M. Barthet, and G. Fazekas. Playsound.space: enhancing a live performance tool with semantic recommendations. In *Proc. 1st SAAM Workshop*. ACM, 2018.

[25] Y. Wu, L. Zhang, N. Bryan-Kinns, and M. Barthet. Open symphony: Creative participation for audiences of live music performances. *IEEE MultiMedia*, 24(1):48–62, 2017.