Random Access Remixing on the iPad

Jon Forsyth, Aron Glennon, Juan P. Bello
Music and Audio Research Lab (MARL)
New York University, New York, NY USA
{jpf211, apg250, jpbello}@nyu.edu

ABSTRACT

Remixing audio samples is a common technique for the creation of electronic music, and there are a wide variety of tools available to edit, process, and recombine pre-recorded audio into new compositions. However, all of these tools conceive of the timeline of the pre-recorded audio and the playback timeline as identical. In this paper, we introduce a dual time axis representation in which these two timelines are described explicitly. We also discuss the random access remix application for the iPad, an audio sample editor based on this representation. We describe an initial user study with 15 high school students that indicates that the random access remix application has the potential to develop into a useful and interesting tool for composers and performers of electronic music.

Keywords

interactive systems, sample editor, remix, iPad, multi-touch

1. INTRODUCTION

The remixing and processing of audio samples[1] is a common feature in the creation and production of popular music, especially electronic music. Artists use a multiplicity of hardware and software tools to edit, process, and recombine prerecorded audio to create entirely new compositions. Many of these tools operate under the "mixing console" paradigm, wherein audio is recorded onto separate audio tracks, usually corresponding to a mixer channel. Most software tools feature waveform editing capabilities, and allow for a variety of processes, e.g. filtering or other audio effects, to be applied to tracks, either individually or in groups.

However, all these applications operate on a single playback timeline in which audio samples are placed. Such an approach fails to explicitly represent the timeline of the original samples. In this paper we introduce a *dual time axis* representation in which the playback and the sample timelines are placed on separate axes in a two-dimensional space, and present a *random access remix* application for the iPad, an implementation based on this representation. We argue that the dual time axis representation can provide users with an intuitive interface for the temporal restructuring of audio samples, and can therefore support the development of novel tools for the composition and performance of electronic music.

The rest of this paper is organized as follows. Section 2 discusses a few commercial and non-commercial sample-editing tools currently available to users. Section 3 introduces and justifies our approach, and discusses some of its implications. In section 4 we present an iPad implementation of our design, and provide a detailed description of the user experience. Section 5 documents our initial user tests in the context of an educational activity for high school students. Section 6 concludes our presentation and discusses ideas for future work.

2. RELATED WORK

There are many commercial applications that can be used for creating remixes and loop-based compositions from prerecorded audio. Many of these applications are based on the mixing console paradigm, in which the application serves as a virtual recording studio. Typically, audio is recorded into tracks that correspond to mixer channels, processed in a variety of ways, and placed at arbitrary points along a common playback timeline. Some examples include commercial applications, such as Logic, GarageBand, and ProTools, and free- and share-ware software such as Audacity[2]. These applications often feature a waveform editor, which allows the user to directly edit an audio sample, although the editor is generally not central to the user interface. Other commercial applications are designed more specifically for the task of waveform editing. For example, Propellerhead's ReCycle[3] is a devoted sample editor that segments a sample, and allows the user to modify each segment independently, e.g., by altering its start and end points and by applying audio processing.

The academic community has produced a number of sample editing tools. For example, the waveTable [6] is a sample editor in which the waveform is displayed on a multi-touch tabletop interface. Touch gestures serve as navigation commands, allowing the user to scroll through the waveform and zoom in or out. A set of tangible tools allows the user to edit the sample and add various audio effects. The Slidepipe [1] is a physical controller consisting of a number of horizontal pipes, each with a set of paddles and ropes, that are used to control audio effect parameters settings and the start and end points of a sample. A similar system, the Chopping Board [4], uses a touch sensitive pad as the physical interface. An audio sample is mapped to the pad, with the location at which the user touches the pad indicating the start point of the sample. Two faders and a knob are used to control effects settings and select samples.

[1]The term sample is used here in the colloquial sense; i.e., a segment of audio.

[2]`http://audacity.sourceforge.net`

[3]`http://www.propellerheads.se/products/recycle`

NIME'11, 30 May–1 June 2011, Oslo, Norway.

In addition, there are a number of loop-based sample remix systems. The Beat-Sync-Mash-Coder [3] is a web-based system in which users can combine a number of pre-recorded audio loops into "mash-ups." The user can also alter the overall tempo. The music loop explorer system [7] also allows a user to mix together pre-recorded samples to create mash-ups. In this case, the system automatically segments music tracks and computes a similarity measure between the segments. The user can combine different loops together and set a master tempo, with the system automatically adjusting the tempo of each segment. Unlike the systems discussed above, the Beat-Sync-Mash-Coder and the music loop explorer system treat a sample as an atomic entity; i.e., it cannot be edited, only combined with other samples and globally modified.

3. APPROACH

3.1 Dual Time Axis Representation

The systems discussed above allow the user to temporally restructure audio, for instance by segmenting the sample and then rearranging the resulting segments, or to combine different audio samples. However, all of these systems treat the audio sample timeline and the playback timeline as collinear; in other words, these two timelines are identical. While this conception of time is appropriate for certain tasks, it also obscures the fact that the sample and playback timelines are distinct. A paradigm that makes this distinction explicit could facilitate new and interesting manipulations of an audio sample.

Perhaps the most straightforward of these manipulations is the temporal restructuring of an audio sample. We want to have the ability to place any region of a sample at any point in the playback timeline. This requires precise control over both the start and end times of each audio event, as well as the start and end times of the desired region from the source audio sample. Both of these sets of parameters are temporal, and both represent positions on separate timelines: the first set of parameters specifies a location on the playback timeline, and the second specifies a location on the source audio timeline.

The above observations suggest a representation that uses two time axes. One such representation is the *recurrence plot*, a tool used to analyze and visualize nonlinear dynamic systems [5]. A standard recurrence plot is a two dimensional, square, binary matrix generated by comparing the state of a dynamic system at a particular time with the states of the system at all times. If a state at time i is the same (within a margin of error) as another state, at time j, a value of 1 is entered into the matrix at position (i, j); otherwise, the value at (i, j) is set to 0. Thus, the row and column indices of the matrix represent time.

Using the recurrence plot as inspiration, we developed a representation consisting of two temporal axes, which we refer to as the *dual time axis* representation. Unlike the standard recurrence plot, in which both dimensions represent a position along the same timeline, here the horizontal axis describes playback time, and the vertical axis describes the timeline of the source sample. This representation allows *random access* to the source audio sample. That is, any portion of the sample can be played back at any position along the playback timeline.

3.2 Interactions

Figures 1 and 2 depict the dual time axis representation. In these figures, the waveforms of the source audio and the output audio are shown to illustrate the interactions. Time 0 for both axes is in the upper left corner. The basic interactions available to the user are the ability to draw, resize, and move blocks. In order to specify a particular region of audio from the source sample to playback, the user draws a block, as shown in Figure 1. The start and end times of the desired region of audio from the source sample are specified by the vertical coordinates of the block, and the start and end times at which this region of audio is played back are specified by the horizontal coordinates. For example, in Figure 1, if the user draws the block as shown, the region of the source sample between 1.0 sec and 2.0 sec is played back starting at time 2.5 sec.

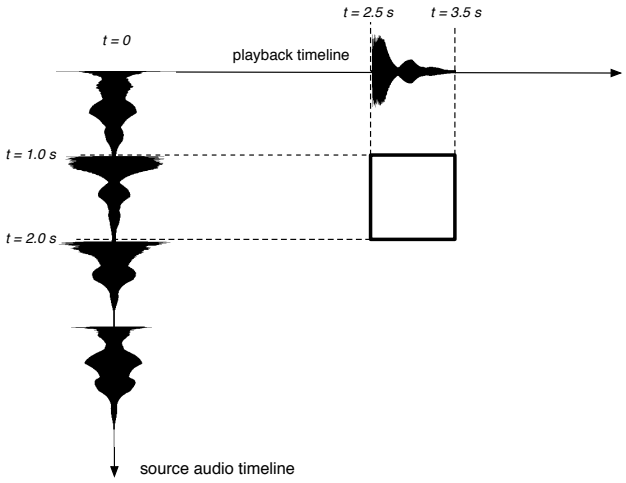

Figure 1: Dual time axis representation.

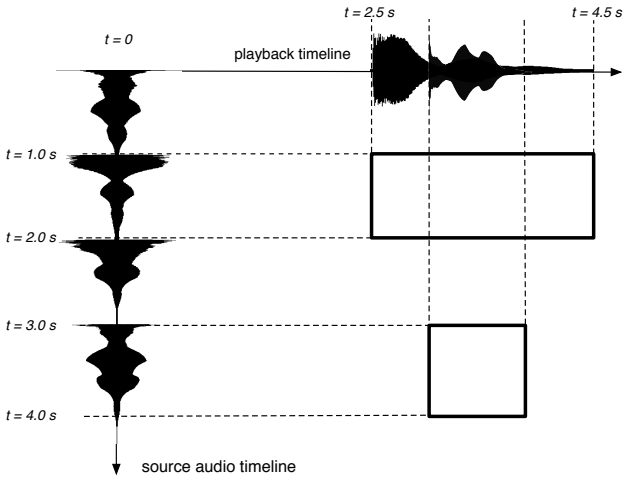

Figure 2: Block in figure 1 time stretched, with additional block.

The user is not restricted to drawing square blocks, but can either draw a rectangular block or resize an existing square block into a rectangle. In the case of a rectangular block, the length of the region from the source audio will not equal the length along the playback timeline. Thus, we must either time- stretch or time-compress the specified region of source audio so that it fits within the specified playback time. Figure 2 shows the effect of resizing the block from Figure 1 into a rectangle. As in Figure 1, the region of source audio is between 1.0 sec and 2.0 sec, a duration of 1 sec. However, the rectangle indicates that this region of audio should be played back between 2.5 sec and 4.5 sec, a duration of 2 sec. Therefore, the region of source audio is time-stretched by a factor of 2.

We can think of each block as having a local copy of the

region of audio defined by its vertical coordinates. Therefore, when we modify the content of a block, for instance by time-stretching or time-compressing it, we are altering only the local copy, while the original source sample is left unaffected. As a consequence, any modifications to a particular block's audio leaves the others' audio unaffected, allowing us to apply any type of audio processing independently. The user can also draw multiple blocks, as shown in Figure 2; here, another block has been added to the configuration shown in Figure 1. The resulting output is a sum of each block's audio.

We refer to an application based on the representation of time described above as a *random access remix application*. It is our contention that such an application would allow a user to remix an audio file with precision equal to that available in typical remix or sample editors. Further, we believe that the dual time axis representation will encourage the user to view an audio sample as random access data instead of data that must be accessed sequentially, and that this new perspective will inspire new ways of thinking about sample editing and remixing.

4. IMPLEMENTATION

4.1 Technical Details

We chose to implement the random access remix application on an Apple iPad, a multi-touch device with a rich development environment (APIs, libraries, frameworks, etc.), including native support for a wide variety of gestures. The relatively large screen and processing power of the iPad also make it an excellent platform on which to develop a musical application with an intuitive and natural graphical user interface (GUI). However, the basic design does not require a touchscreen, and could be implemented on a desktop or laptop computer.

We implemented a prototype application in Objective-C and C/C++, using a combination of Apple's UIKit framework and OpenGL ES to implement the GUI. OpenGL ES is an implementation of OpenGL for mobile devices, including the iPad. It is efficient, powerful, and portable. In addition, we used MoMu, an open source application development toolkit for mobile devices [2]. MoMu consists of APIs and utilities that support the development of interactive mobile applications on iOS, including a layer of abstraction that greatly simplified the handling of audio input and output. We subsequently developed a second version of the application, using OpenGL ES, MoMu, and Cocos2D[4], a free, open source framework for developing graphical applications for the iPhone and iPad. Cocos2D provides functionality similar to the UIKit, but with greater flexibility.

4.2 User Interface

The main user interface for the second version of the random access remix application is shown in Figure 3. The interface consists of four main elements: the file select/effects edit region (① in Figure 3), the waveform display (②), the block editing region (③), and the toolbar (④). To load a sample, the user selects a file listed in the file select region. Upon doing so, the audio waveform is displayed in the waveform display region, with time 0 at the top. The block editing region is the area in which the user creates and modifies blocks. As in Figures 1 and 2, the horizontal axis represents the playback timeline (with time increasing from left to right), and the vertical axis represents the sample timeline (with time increasing from top to bottom).

The toolbar (③ in Figure 3) allows the user to control playback, clear the block editing region, copy or delete a

[4]http://www.cocos2d-iphone.org/

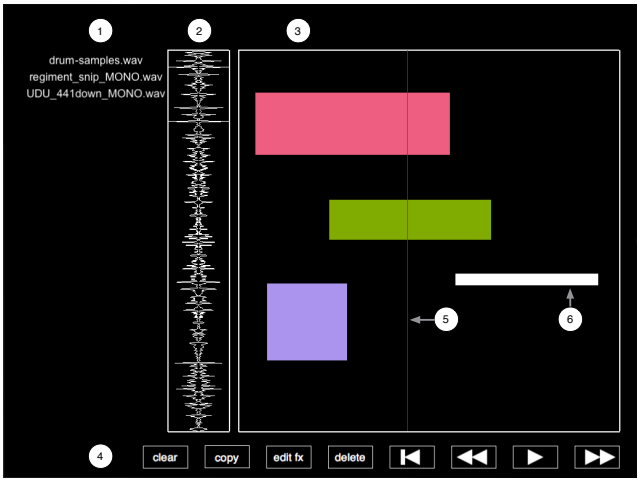

Figure 3: User interface.

block, and edit the effects settings for a block. The playback controls consist of the standard set of tape deck controls: reset playback position to time 0, rewind, play/pause, and fast forward. The audio output is looped, so that when playback reaches the end of its timeline (the rightmost edge of the block editing region), it immediately returns to time 0 (the leftmost edge of the block editing region). The vertical red line in the block editing region (⑤ in Figure 3) indicates the current playback position, sweeping from left to right during playback.

The user can edit the effects processing applied to a particular block by pressing the "edit fx" button in the toolbar (④ in Figure 3). Pressing this button reveals a set of sliders, used to adjust effects parameters, in the file select/effects edit region (①). The list of available files is displayed if the user presses this button again.

The block editing region responds to standard iOS gestures. To draw a block, the user places a finger in an empty region of the block editing region, dragging until the block is the desired size. In order to select an existing block, the user places a finger on the block; the block color changes to white to indicate selection. Figure 3 shows a selected block (⑥). Once selected, the user can copy, delete, or set the effects parameters applied to the block using the corresponding buttons in the toolbar. In addition, the user can move a block by dragging it to the desired location.

5. INITIAL EXPERIENCES

5.1 Context

The first implementation of the random access remix application[5] was used as part of a four session long workshop for high school students developed by New York University and the Institute for Collaborative Education. The general objective of the workshop was to use music as a basis for teaching the high school students various scientific concepts. Each session, the students attended a lecture, and then performed a hands-on activity related the lecture material. The workshop participants consisted of 15 students total, all in either the 10^{th} or 11^{th} grade. The group was split roughly evenly between males and females, and between students with an interest in science and those with experience in music.

The final session focused on digital audio, and included

[5]This implementation is similar to the one described in Section 4, with the most notable difference being the lack of a waveform display region.

a discussion of quantization, sampling, and the ease with which digital audio can be manipulated. Because the random access remix application is essentially a tool for the temporal restructuring of audio, its functionality aligned well with the lecture topics. In addition, we implemented a number of relevant audio effects, specifically bit crushing and downsampling. The bit crushing effect allowed the user to change the bit depth of the audio associated with a block from 16 bits to 1 bit, thus creating audible distortion at lower quantization levels. The downsampling effect allowed the user to change the sampling rate from 44100 Hz to 4410 Hz, thus raising the pitch of the audio segment as well as its overall duration. This change in duration was reflected in the block editing region of the application: lowering the sampling rate setting for a particular block reduced that block's width, thus allowing the students to both hear and see the results of downsampling a segment of audio.

5.2 Activity

We provided the students with iPads, with each iPad being shared by a group of two or three students. We briefly introduced the application to the students, describing its basic interactions, as well as the dual time axis representation. We then allowed them to freely explore the application for roughly 20 minutes, providing assistance when necessary. Although some students initially found the dual time axis representation to be confusing, they were able to understand the concept after a few minutes of explanation. Other students were comfortable with the application from the outset, and began to produce sounds soon after receiving the iPad. Most students focused on creating visual patterns of blocks and listening to the results.

When the students seemed comfortable with the application, we presented them with a musical task. We played a recording of a simple four-note target melody, and asked them to approximate it using an audio file that consisted solely of a sample of a single note of a vibraphone. In order to complete the task, the students had to complete a number of sub-tasks. First, they had to locate the beginning and end of the note in the source sample. Next, they had to place a block along the playback timeline corresponding to each note of the target melody. Finally, in order to approximate the pitches of the target melody, the students had to alter the sampling rate of each block. This task required an understanding of how digital audio can be manipulated, as well as how altering the sampling rate of digital audio alters the pitch. Most of the students were able to perform the task.

5.3 Observations

Although we were not able to obtain any quantitative data from this experience, we were able to make a number of observations. The application was, in general, positively received by the students and the teachers who accompanied them; most seemed to enjoy using it, and two or three expressed interest in obtaining a copy. However, the activity exposed a number of design flaws. In particular, during the melody creation task, the students found it difficult to properly locate the region of the audio sample corresponding to the note. This difficultly was likely due to the lack of visual feedback indicating the content of the audio sample. This problem has been remedied in the second version of the application with the waveform display, as shown in Figure 3. In addition, the students occasionally created small blocks in the interface, for instance through an accidental touch or by reducing the sampling rate to the minimum value; it was not possible to select (and thereby delete) the smallest of these blocks. Although the user could clear all the blocks, this was not a satisfactory solution. Instead, we could impose a minimum size restriction on the blocks, and increase the minimum value of the sampling rate. Although we did not implement this feature, some students inquired if was possible for the application to render the output to an audio file that they could take home.

6. CONCLUSIONS AND FURTHER WORK

The positive reaction from the high school students indicates that the dual time axis representation has merit. Although the activity with the students exposed some of the application's shortcomings, some of these problems have been fixed in the second version of the application, while others should be relatively easy to remedy. There are a number of further enhancements that we would like to make to the application, such as implementing additional audio effects. We also feel that the application should allow the user to define the length of the remix, instead of limiting it to the length of the source sample. In addition, we plan to add tools to analyze the input audio signal in order to extract high-level musical information. For instance, we feel that beat tracking and rhythmic quantization would greatly enhance the application, as it would allow the user to more easily create musical coherence between the various blocks.

While our initial experiences are encouraging, it is necessary to conduct more extensive user tests in order to produce a quantitative evaluation of the application. Such testing should identify areas of relative strength and weakness, and point the way towards further improvements. While our initial experiences indicate that our implementation is appropriate for relative novices, our hope is that an application that combines an intuitive graphical user interface with intelligent processing can achieve sufficient depth and sophistication to make it a useful tool for more experienced musicians and technologists. Further user testing could begin to provide some answers to this question.

7. REFERENCES

[1] M. Argo. The slidepipe: A timeline-based controller for real-time sample manipulation. In *Proceedings of the 2004 Conference on New Interfaces for Musical Expression (NIME04), Hamamatsu, Japan*, 2004.

[2] N. Bryan, J. Herrera, J. Oh, and G. Wang. Momu: A mobile music toolkit. In *Proceedings of the 2010 International Conference on New Interfaces for Musical Expression (NIME2010), Sydney, Australia*, 2010.

[3] G. Griffin, Y. Kim, and D. Turnbull. Beat-sync-mash-coder: A web application for real-time creation of beat-synchronous music mashups. In *2010 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Dallas, Texas*, 2010.

[4] J. Lee. The chopping board: Real-time sample editor. In *Proceedings of the 2006 Conference on New interfaces for musical expression (NIME '06), Paris, France*, 2006.

[5] N. Marwan, M. Romano, M. Thiel, and J. Kurths. Recurrence plots for the analysis of complex systems. *Physics Reports*, 438(5-6):237–329, 2007.

[6] G. Roma and A. Xambò. A tabletop waveform editor for live performance. In *Proceedings of the 2008 Conference on New interfaces for musical expression (NIME '08), Genoa, Italy*, 2008.

[7] S. Streich and B. Ong. A music loop explorer system. In *Proceedings of the 2008 International Computer Music Conference (ICMC), Belfast, Northern Ireland*, 2008.