aSeamless Scripting and Recording of Audio Narratives

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

ACM Classification Keywords
H.5.2. Information Interfaces and Presentation (e.g. HCI): User Interfaces - Graphical user interfaces

Author Keywords
Audio recording, scripting, transcript-based editing

INTRODUCTION

Audio recordings are a common form of communication used in voice-overs, podcasts, audio books and e-learning. Closest to everyday conversation, audio recording is a medium with relatively low barriers to entry. So, it is used by many laymen who are not professional producers or writers to distribute information. A common workflow for creating such audio recordings involves three main steps: writing a script, recording audio and editing audio. To create a compelling audio recording, users go through several iterations of these steps.

Consider the case of recording a voice-over for a video. The narrator does an initial recording based on a prepared script. Afterwards, while placing it on top of the video, the producer wants to make some changes to parts of the narration e.g. to control the timing of a specific word, or to match changes made in the shots after the audio was recorded. While some of these edits can be done from the existing recording using an audio editing software, others require editing the script and re-recording the altered parts. Similarly, consider an audio recording of an online lecture. After the initial publication, the lecturer may want to re-record or add parts e.g. to keep examples up to date, or to address common questions that came up from viewers. These are but a few of many scenarios where producers iterate back and forth between script writing, audio recording and audio editing.

Most existing audio editing systems provide functionalities necessary to support the latter two steps: recording and editing audio. However, the first step, writing or editing the script, is usually overlooked or treated as a completely separate task. This is the case even when scripts play a key role in recording and editing speech.

In this paper, we present an interface that supports and links all three steps of the aforementioned workflow. Our system addresses challenges that span the process of creating audio narratives, including (1) iterating back and forth between script and audio recordings, and (2) combining multiple audio recordings into a single final track. Our interface is inspired from familiar document editors and text merge tools, which are easy to learn.

RELATED WORK

Adobe Story [2], FinalDraft [6] and Celtx [5] are examples of software applications dedicated to script writing. They support collaboration, automatic formatting, navigation and planning for future production, but they treat the script as a text document that is essentially separate from the recordings. In fact, in our preliminary interview of lay and professional audio producers, we found that many of them use general-purpose document editors like Google Docs [8] or Microsoft Word [11] to prepare their scripts.

At the recording and editing stage, Adobe Audition [1], Avid ProTools [4], GarageBand [7] and Audacity [3] are among the most popular digital audio workstations (DAWs). These tools allow users to edit audio by manipulating waveforms in a multi-track timeline interface. They also provide a wide variety of low-level signal processing functions. However, since they are designed to serve as general-purpose audio production systems, they include many features that are not directly relevant for creating audio narratives whose main content is speech. Hindenburg Systems [9] develops tools that are specifically targeted for audio narratives. Still, they are primarily concerned only with the audio and they do not deal with the script directly.

Recently, several researchers have explored using audio transcripts to support text-based navigation and editing of audio. Whittaker and Amento [16] demonstrate that users prefer editing voicemail through its transcript instead of its waveform. Inspired by similar intuition, Casares et al. [13] and Berthouzoz et al. [12] enable video navigation and editing through time-aligned transcripts. Rubin et al. [15] extend this approach to audio narratives and propagate edits in the transcript text to the corresponding speech track. These systems all focus on editing the audio on the assumption that all the recording is done beforehand. *Narration Coach* developed by Rubin et al. supports an iterative narration recording process, but instead it assumes that the user has a fixed input script. Its focus, providing capture-time feedback to improve speech performance, is also very different from ours. Our work also takes advantage of text-based navigation and editing, but unlike these systems, we support an dynamic workflow where both the audio recordings and the underlying script can be continuously updated.

KEY IDEAS

To learn about current practices and challenges for recording audio narratives, we interviewed ten professional lecturers and two video producers who regularly created audio recordings as part of online lectures that were published on platforms, including YouTube, Udacity, EdX and MITx. Following are several key insights we gained from the pilot interviews.

Scripts play a major role during recording. All of the lecturers prepared written materials about what they were going to speak before they started recording. The format and level-of-details of these scripts varied. For instance, one lecturer used his lecture slides containing images and a list of bullet points as his script. Another lecturer typed a thorough word-for-word transcription of what he was going to say in a text document. Another person used a handwritten outline. In all cases, while they were recording, they kept the scripts within their view and depended on them to construct their narrative.

Script planning, recording, and editing is an iterative process. Even when a thorough word-for-word script is prepared beforehand, often the recording does not follow the script exactly. The speaker may remember and add more details while recording, or find a more natural way of saying a written sentence. In some cases, major changes are made to the script after some recording has happened. The iteration may happen over a longer period of time. One lecturer noted that he periodically revisited and re-recorded parts of the lecture to include up-to-date examples.

The final track is created by mixing and matching multiple recordings. Most users recorded multiple takes, and then edited them using an audio editing software to produce the final recording. Many of them noted that aligning the multiple takes, finding the best take of a given part, and seamlessly merging the cuts were the most time consuming and tedious tasks.

USER INTERFACE

Motivated by these insights, we developed the VoiceScript interface to support iterative script writing and audio recording/editing. As shown in Figure **??**, our interface consists of two types of documents: a *Master Script* document (left) displays the current status of the final track, including what has been recorded and what was planned to be recorded but has not been recorded yet (i.e. the original script), while a *Transcript* document (right) displays the verbatim transcript of individual audio takes. As the user records new takes, our tool aligns the audio transcripts to the master script so that it is easy to compare each take with the master script and with each other. It also partitions each recording into segments that can be seamlessly joined between takes. At any point during the recording process, the user can edit the script or the final track by editing the master script like a text document. The rest of the section describes the interface using an example workflow.

In Figure **??**, the user begins by writing on the master script an outline of points to record. The text appears in light grey to indicate that these parts have not been recorded yet. At this stage, the master script is like an ordinary word document or script.

Once the user starts recording, the audio is transcribed in real time and a verbatim text corresponding to each take appears in a separate transcript document tab. Each transcript is time-aligned with the corresponding recording, so that users can quickly navigate to specific part of the audio by clicking on a word in the transcript. If the *compare-view* is turned on, our system aligns segments of the transcript to corresponding segments in the master script, (in this case the points in the original outline), and shows them side-by-side (Figure **??**). If a segment in the transcript does not correspond to any of the segments in the master-script (e.g. when the speaker improvises or ad-libs), it is highlighted in yellow.

When the user has multiple takes, in addition to each of the transcripts, the *all-tab* provides a summary of all of the takes. As in the *compare-view*, for each segment in the master script, it displays corresponding transcript segments, this time from all of the audio takes. A drop-down button indicates that there are multiple versions (or takes) of the same segment. Clicking on the button opens a list showing the alternative versions from different takes (Figure **??**). The user can listen to any of these takes and select one without having to search through individual takes. If none of the recordings contains a part corresponding to a segment in the master script, it is highlighted in red. In this way, the user can easily keep track of what has been recorded, and what still needs to be recorded.

The user can *accept* audio segments into the final track by clicking on a button next to each transcript segment either from the individual transcript tab or the all-tab. If there is a corresponding master script segment, the transcript segment replaces it. If the segment was improvised the transcript segment is simply inserted into the master script. The text of this segment in the master script now appears in darker color to indicate that it has been recorded. Now, the master script contains both recorded text that is accepted to the final track, and unrecorded text from the original script. Playing the master script will play only the darker, recorded text.

At any stage, the user can edit the master script like a text document. Edits made to the recorded part of the text is reflected in the audio. For instance, the user can delete a recorded word from the text, and it will be deleted from the audio. Mark as dirty. Correct transcription. Users can also simply insert more text to record (which appears in lighter color), or make changes to unrecorded text.

The user iterates between all of these steps, editing the master script, recording audio takes, and accepting segments of the takes into the final track to produce the final recording.

ALGORITHMIC METHODS

Transcribing the audio recording

We use IBM Speech to Text Service [10] to obtain a verbatim transcript of each audio take in real time. The service produces a time stamp for each word indicating its start and end time. It also segments the transcript into *utterances* where each utterance is separated by a longer silent gap in the speech (longer than 500 ms). While automatic speech recognition is imperfect, we have found that the results were accurate enough for the purpose of alignment (next section) and for understanding the transcript.

Aligning the transcript to the master script

Once we have a verbatim transcript of an audio take, we compute the global word-to-word alignment between the transcript and the master script using the Needleman-Wunsch (NW) algorithm [14]. NW allows for insertions and deletions, which accounts for differences in the two documents.

Co-segmenting the transcript and the master script

RESULTS

INFORMAL USER EVALUATION

To gauge the utility of our interface, we conducted an informal evaluation with two users. We started each session with a brief demonstration of our interface, and then asked the participants to create an audio recording. We examined their workflow, and the number/type of features they used. We also solicited written qualitative feedback about our interface at the end of the session. Each session lasted about 50 minutes.

We also conducted a pilot study to compare our interface with a state-of-the-art transcript-based speech editing interface [15]. We recruited four participants, none of whom had experience using text-based audio editing systems. We gave them a script with bullet points outlining a mini lecture on a science subject (e.g. *gravity* and *dark matter*) and two audio takes roughly corresponding to that script. Their task was to cut and merge the two takes to produce a recording that contained all the contents listed in the script and only those contents. The two takes were similar, but both takes had some missing content from the outline and one had some extra content. So, the participants had to choose parts from each take and combine them to get the final result. We encouraged the users to focus on having a complete content, rather than the details of the audio quality (e.g. tempo, diction, flow of speech etc.).

Each participant completed the task twice with different outlines, once using our interface and the other using Rubin et al.'s interface. The subject of the outline and the order of the interface was counter-balanced. We examined the time time they spent editing, the number/type of functions they used, and the quality of the final recording. After the session, participants gave written qualitative feedback about the two interfaces. In total, each session lasted one hour.

Overall, the results from the study were extremely encouraging. Each of the four participants preferred VoiceScript over Rubin et al.'s interface for the given task, and noted they would use our interface to edit audio recordings. Every participant also completed the task faster using our interface (avg. 7.4 ± 1.6 min) than Rubin et al's interface(9.9 ± 1.5 min). Table **??** summarizes the participants' usage of VoiceScript during the editing session.

Explain usage

Positive Feedback

CONCLUSIONS

ACKNOWLEDGMENTS

REFERENCES

1. 2016a. Adobe Audition. `http://www.adobe.com/products/audition.html`. (April 2016). Accessed: 2016-04-02.
2. 2016b. Adobe Story. `https://story.adobe.com/en-us/`. (Apr 2016). Accessed: 2016-04-02.
3. 2016. Audacity. `http://www.audacityteam.org/`. (April 2016). Accessed: 2016-04-02.
4. 2016. Avid Protools. `http://www.avid.com/en/pro-tools`. (April 2016). Accessed: 2016-04-02.
5. 2016. Celtx. `https://www.celtx.com/index.html`. (April 2016). Accessed: 2016-04-02.
6. 2016. Final Draft. `https://www.finaldraft.com/`. (April 2016). Accessed: 2016-04-02.
7. 2016. GarageBand. `http://www.apple.com/mac/garageband/`. (April 2016). Accessed: 2016-04-02.
8. 2016. Google Docs. `https://www.google.com/docs/about/`. (April 2016). Accessed: 2016-04-02.
9. 2016. Hindenburg Journalist Pro. `http://hindenburg.com/products/hindenburg-journalist-pro`. (April 2016). Accessed: 2016-04-02.
10. 2016. IBM Speech to Text Service. `https://www.ibm.com/smarterplanet/us/en/ibmwatson/developercloud/doc/speech-to-text/`. (April 2016). Accessed: 2016-04-02.
11. 2016. Micorsoft Word. `https://products.office.com/en-us/word`. (April 2016). Accessed: 2016-04-02.
12. Floraine Berthouzoz, Wilmot Li, and Maneesh Agrawala. 2012. Tools for placing cuts and transitions in interview video. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 67.
13. Juan Casares, A Chris Long, Brad A Myers, Rishi Bhatnagar, Scott M Stevens, Laura Dabbish, Dan Yocum, and Albert Corbett. 2002. Simplifying video editing using metadata. In *Proceedings of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques*. ACM, 157–166.
14. Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology* 48, 3 (1970), 443–453.
15. Steve Rubin, Floraine Berthouzoz, Gautham J Mysore, Wilmot Li, and Maneesh Agrawala. 2013. Content-based tools for editing audio stories. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. ACM, 113–122.

16. Steve Whittaker and Brian Amento. 2004. Semantic speech editing. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 527–534.