Audio Engineering Society

Convention Paper

Presented at the 146th Convention
2019 March 20–23, Dublin, Ireland

jReporter: A Smart Voice-Recording Mobile Application

Lazaros Vrysis[1], Nikolaos Vryzas[2], Efstathios Sidiropoulos[3], Evangelia Avraam[4] and Charalampos Dimoulas[5]

[1, 2, 3, 4, 5] *Aristotle University of Thessaloniki, 54124, Greece*

lvrysis@auth.gr

ABSTRACT

The evaluation of sound level measuring mobile applications, shows that the development of a sophisticated audio analysis framework for voice-recording purposes may be useful for journalists. In many audio recording scenarios, the repetition of the procedure is not an option, and under unwanted conditions the quality of the capturing is possibly degraded. Many problems are fixed during post-production, but others may make the source material useless. This work introduces a framework for monitoring voice-recording sessions, capable of detecting common mistakes and providing the user with feedback to avoid unwanted conditions, ensuring the improvement of the recording quality. The framework specifies techniques for measuring sound level, estimating reverberation time and performing audio semantic analysis by employing audio processing and feature-based classification.

1 Introduction

The main objective of some previous works of ours was the design and implementation of mobile applications for smartphones that were featuring sound metering and audio semantics analysis capabilities [1]-[2]. The evaluation of these software solutions proved that the development of this kind of software is feasible and can be useful for the targeted professionals. An extension of these works could be the establishment of a smart audio framework, servicing the voice recording needs of journalists and reporters. In most voice recording situations, there is no opportunity of repeating a session, therefore, possible mistakes that cannot be fixed during production may downgrade the quality of a recording, making the material useless for production. Moreover, there may be a wide range of voice recording applications available on mobile stores (App Store, Google Play), but the majority of them does not provide enough functionality to facilitate the needs of professionals. This study introduces a voice recording management framework for mobile computing devices, capable of detecting common recording mistakes, informing the operator, and proposing appropriate tips in order to improve the overall quality of the capture.

2 Related Work

User Generated Content (UGC) and Citizen Journalism, as well as the constant rise of the use of web for publishing changes the whole landscape of content creating and publishing [3]. This change requires the collaboration of different disciplinary specializations and the adaptation of traditional newsrooms [4]. Crowdsourcing is part of the practices of the biggest news organizations [5].

Great advances of content creation do not concern solely the utilization of resources created and provided by non- professional users. Mobile Journalism (MoJo) is an emerging field, finding applications mostly in live reporting and breaking news. Professional journalists tend to make use of the sensor and connectivity capabilities of modern smart mobile devices, in a similar way citizens do in citizen journalism. Modern devices provide effective packaging of

cameras and microphones, as well as Global Positioning Sensors (GPS) for geographical localization, accelerometers and gyroscopes to monitor the movement and direction of the device [6]-[7]. Even in the common case of having access to professional high-end equipment, MoJo offers versatility, affordability and promotes rapid publishing [8].

Meanwhile, smart and automatic procedures can be applied for the handling of vast amounts of content collected through crowdsourcing [9]. Recent developments of microelectronics have brought such processing power and hardware capabilities to mobile devices that make them comparable to personal computers. This unleashes the potential of mobile software, allowing heavy-duty tasks, such as semantic audio analysis, to be processed in real time. Among others, mobile audio sensors can be exploited along with their corresponding software control and signal processing functions in sophisticated audio analysis applications, including the implementation of mobile sound level meters [1].

In this context, many audiovisual recording procedures (photography, audio and video recording) can be carried out without deploying professional equipment; typical consumer mobile computing devices, like smartphones, are capable of producing high fidelity multimedia content [2]. In most situations, smart devices may also simplify a recording procedure, because of practicality due to compact size, low weight and user-friendly interface. Moreover, a whole new market of gadgets and add-ons has emerged in order to improve the quality of multimedia production with mobile devices. In terms of hardware, this is supported by connecting external professional quality specialized microphones, camera lenses etc. [10]. The newsrooms adapt and centralize their infrastructure for content gathering and publishing as interconnection between reporters and editors is essential [5]. However, features that professional equipment support, are being implemented into mobile applications rarely, even if the software migration is not a complex process.

The main target of this work is the investigation and design for automating techniques that can assist on detecting and correcting common audio recording issues. These techniques can be embedded in mobile software applications and should be exploited for semi-professional use, enabling audio recording of adequate quality without requiring special equipment (voice recorder) or operator (sound engineer).

3 Proposed Framework

The proposed framework is founded on three processing pipelines: the continuous inspection of the raw input signal, the accurate sound level measuring and the semantic analysis of the audio content. Finally, a decision-making algorithm is capable of detecting inappropriate recording conditions, taking into account the individual outputs of these modules. After investigation for a concrete classification of unwanted recording conditions, five categories were formed: (a) clipping of the signal, (b) high background noise, (c) low signal-to-noise ratio, (d) presence of high-level noise and (e) inappropriate room acoustics with high reverberation time.

One the one hand, audio input waveform monitoring and sound level metering units can easily be implemented as modules of a mobile application [1]-[2]. Furthermore, the additional requirement for accurate audio semantics analysis is a more demanding task, without being unachievable; mobile applications that incorporate this kind of processing units can achieve adequate audio classification performance [12]. In this work, a lightweight *kNN* classifier along with an extended temporal feature integration methodology are deployed, delivering robust performance without requiring high processing power [13]-[15]. Audio-classification unit is capable of detecting noise sources, exploiting a generic sound classification scheme. This scheme includes patterns like wind blowing, noise from common electrical devices (i.e. air-condition, refrigerator, power supplies), noise due to defective or damaged cables, electromagnetic interference etc. Furthermore, sound level metering and audio classification processes are performed on a dual channel-basis: the use of a secondary, narrowband processing pipeline, restricted in the frequency range of speech, favours high or low frequency noises that may not necessarily be dissuasive for capturing a satisfactory recording. All these analyses are performed in real-time during a voice recording session, resulting to a visual indication for the user, along with the appropriate suggestion to solve or suppress the presence of any issue (by applying a windscreen, switching off other devices, checking the wiring etc.). Finally, a reverberation time estimation modality can be

deployed separately, before the conduction of a recording, by capturing a short, impulsive, audio stimuli (i.e. handclapping) [2]. An autocorrelation algorithm can analyse the test recording and produce a rough estimation of the reverberation profile of the space.

4 Application Development

In the same context, a pilot application that implements the aforementioned specifications was designed and implemented, under the codename *"jReporter"*. In addition to the audio-analysis engine, the application features multiple recording sessions, automatic spatio-temporal annotation and support for external stereo microphones.

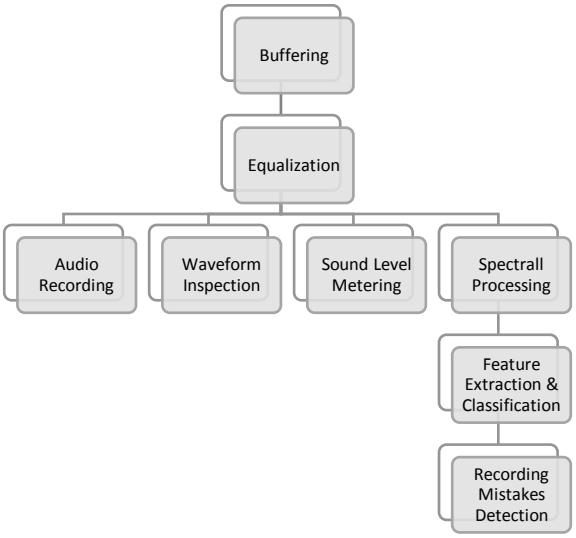

Figure 1. Hierarchical audio processing architecture

Development process followed a waterfall model, including the typical phases of analysis, design, implementation and verification. Coding is platform-specific, while extended emphasis is given to performance optimizations. The application is structurally split into two main components, the back end and the frond end. The back end performs the audio playback, recording, and analysis procedures, while the front end adds the user interface and the audio session management functionality. The GUI is implemented using the OpenGL libraries while the code is written in C++ and Objective C languages in order to maximize performance, allowing algorithms to be processed in real time [16].

As a first step, the process of real-time capturing is carried out, making use of the *AudioQueues* services included in *AudioToolbox*. *AudioQueues* provide direct access to raw sample data that are temporally stored in the input buffers of the audio adapter, eliminating any pre-processing algorithm that operating system may deploy and bypassing *Automatic Gain Control (AGC*) [1].

Next, audio data are stored in circular buffer, facilitating the further processing and spectral equalization. In general, the samples being captured are stored temporally into a 64-bit floating-point data structure; the signal is being processed spectrally, calibrated and filtered. Frequency analysis is performed using a 4096-point RDFT-FFT algorithm, calibration is applied in order to cancel the non-linearity of the device's audio interface and filtering is performed by applying functions that are defined by the A/B/C/D weighting filters. These data are routed to four discrete processing units: (i) audio file formation and storage, (ii) sound level metering, (iii) waveform inspection and (iv) semantic analysis, as Figure 1 demonstrates. In more details:

(i) Audio Storage: This module forms the structure of the audio file (header, *RIFF* structure, writing data to storage device) and ensures the continuity of audio information. The produced files are coded as PCM Waveform with sampling rate of *44.1 kHz* and *32 bits* of depth. The module is capable of producing both stereo and mono audio files, in respect to the number of channels that input device supports. Thus, when internal or external monophonic microphones are used, the derived files are monophonic; a stereo file is produced when input is stereo as well.

(ii) Sound Level Metering: This module implements *A/B/C/D* weighting and *Impulsive/Fast/Slow* responses, according to ANSI standards [17], delivering an accuracy of *±1.5dB* in frequencies from *50Hz* to *18kHz* ranging from *30* to *110dB(A) [1]*. Taking into consideration the requirements of this particular application, and focusing on the detection of broadband noise sources, a decision for unweighted, slow response sound metering was made.

(iii) Waveform Inspection: This module attempts to detect waveform clipping of the captured signal. In

particular, in iOS audio sample data are represented as floating point numbers in the range from *-1* to *1*. Thus, values with absolute value close to *1* indicate a possible clipping and an appropriate warning is presented to the user.

(iv) Semantic Analysis Unit: This module performs real-time non-optimal audio recording conditions detection. Specifically, it can identify situations with high background noise, low *Signal-to-Noise Ratio (SNR)* and the presence of high-energy noise sources. Buffered audio data, after equalizing, are routed to a dual-channel (bandpass/broadband) feature extraction unit. A baseline set of *22* audio-features are estimated: *Spectral Centroid, Spectral Spread, Spectral Kurtosis, Spectral Skewness, Spectral Flatness, Spectral Slope, Spectral Roll-off, Spectral Crest Factor, Spectral Flux, Zero Crossing Rate and 12 Mel-Frequency Cepstral Coefficients (MFCCs).* All but *MFCCs* are calculated on both analysis channels; *MFCCs* are computed exclusively from the broadband channel, forming a *32*-dimensional feature vector. Feature extraction is performed in a *50ms/25ms* frame-size/step basis, and after a process of temporal feature integration, where *Mean Value* and *Variance* are derived for each feature on a *1s*-long texture window, the final *64*-dimensional feature-vector is derived [13]. This feature-vector serves as input to the classification unit, which is an implementation of a *k Nearest Neighbors (kNN)* classifier with *k* set to *1*. Trying to meet the needs of the users (i.e. journalists), a *three-class* classification scheme was decided. The categories are *Speech*, *Music* and *Noise*. The algorithm classifies unknown input by comparing it to ground-truth patterns that is stored into a sound samples library. Initially, this library contains *30* sound patterns (*10* from each class) but is dynamic, meaning that new samples can be easily captured by the user.

A decision-making logic takes into account the output of these four units, in order to detect possible recording problems. For example, when *Sound Pressure Level (SPL)* is in the mid-range (*~40-60dB*) and semantic analysis unit classifies input as *"Noise"*, then a warning for *"High Background Noise"* is displayed to the user, whereas when input is classified as *"Speech"* the warning notifies for *"Low Signal-to-Noise Ratio"*. In a similar way, when high-energy noise is detected (*>60dB SPL*), the user is informed about the presence of a noise source with high sound level.

By populating the dynamic samples library and modifying the decision-making algorithm to meet new requirements, the non-optimal recording conditions that can be detected can be increased.

Along with the aforementioned real-time processing flows, the software features an additional facility, capable of measuring the reverberation profile of the recording space, by measuring *Reverberation Time (RT60)*. This process can be carried out following the interrupting-noise method that previously was implemented in iSMARTer [10], or a by exploiting a more user-friendly autocorrelation strategy [18].

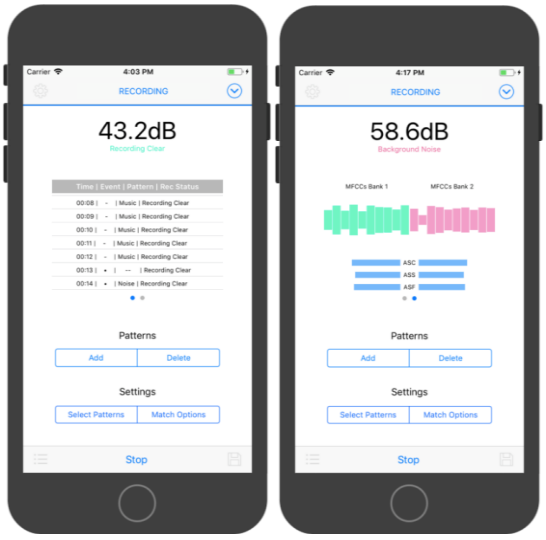

Figure 2. Standard/Professional faces of the main view of *jReporter*.

Figure 2 presents the view of the main screen of *jReporter*. This view provides the necessary functionality for conducting audio capturing: the usual controls for starting/stopping a recording is placed on the bottom of the screen, whereas a rich set of visual indicators lay in the middle for displaying the current status of the session. On the top of the display, an alphanumerical indicator is placed, showing Input Sound Level and a label that notifies about a possible recording issue. The main graphical component is a timeline which logs the most important information regarding the recording progress in a one-second basis. This in-

cludes a timestamp, the type of content (Speech/Music/Noise), event detection and recording status. Underneath of this there are some controls for managing the sound-samples library and classification module options.
The views for storing/viewing the recordings are designed according to the modern UX/UI standards, ensuring improved usability and facilitating common metadata structures, including tags, photos, geolocation.

5 Model Evaluation

Last but not least, a series of tests was conducted in order to evaluate the performance of the semantic analysis unit. The objective of these tests was the classification of audio content following a three-class scheme (Music, Speech, and Noise). The duration of the test dataset is about 30 minutes, equally containing equivalent content of each class. Standard accuracy metric was deployed for the evaluation and the audio signal was split to samples of 1-second duration.

Scenario	Audio Input	Noise
A	MacBook Air	-
B	MicW i436	6-16kHz Pink Noise
C	MicW i436	Broadband Pink Noise
D	Logitech H540	-
E	Logitech H540	-

Table 1. Specifications for the five test scenarios.

Aiming at a more solid evaluation, the experiment was repeated multiple times, each one featuring different configuration, as specified by five test scenarios. The scenarios specify the presence of background noise, the type of the noise, the type of the input device (microphone) and the signal-to-noise ratio.
Table 1 and Table 2 carry information about the audio input being used, the type of background noise, SNR and the classification accuracy. The first thing that someone can note is that, in general, classification accuracy for the proposed classification scheme is satisfactory.

Scenario	SNR	Accuracy
A	30dB	0.93
B	18dB	0.93
C	24dB	0.78
D	15dB	0.93
E	9dB	0.81

Table 2. Classification results for each test scenario.

Temporal feature integration reduces variability inside a class, resulting in finer modelling of the common attributes amongst the samples of the same sound category and, additionally, many short-time instances are grouped together, reducing the amount of data derived to the -usually- slow classifiers and relieving the classification process. Additionally, the deployment of a dual-channel semantic analysis increases the robustness of the system, offering steady performance when tested under different input devices. It is worth noting, that without deploying the narrowband semantic analysis flow, classification accuracy decreases about *10* percentage points.

6 Conclusions

Preliminary evaluation results are very promising, and as *jReporter* is a product based purely on software, facilitates any debugging or upgrading process. Semantic analysis unit provides adequate accuracy but more sophisticated classification algorithms can be implemented in order to boost performance. A dual-channel semantic analysis workflow, which will bring spatial -stereo- semantic analysis is currently under development. This new feature can perform really well when combined with stereo microphones, like *Zoom IQ6* and *IQ7* [19]-[20]. Additionally, more types of smartphone-targeted microphones (like *Rode VideoMic* [21]) can be tested and be recommended for *jReporter*. Concerning *GUI/UX Design* aspects, next development phase specifies usability testing, targeted at journalists and reporters.

7 Acknowledgements

This research has been supported by the Operational Program "Human Resources Development, Education and Lifelong Learning" and is co-financed by the European Union (European Social Fund) and Greek national funds.

References

[1] L. Vrysis, C. Dimoulas, G. Kalliris, and G. Papanikolaou, "SPL Meter: Sound Pressure Level Meter software development on a smartphone (iPhone)" *Helina 6th National Conference Acoustics 2012*, Kerkyra, Greece, (2012).

[2] L. Vrysis, C. Dimoulas, G. Kalliris, and G. Papanikolaou, "Mobile audio measurements platform: bringing audio semantic intelligence into ubiquitous computing environments", *134th AES Convention*, pp. 183-189 (2013).

[3] L. Palen, K.M. Anderson, G. Mark, J. Martin, D. Sicker, M. Palmer, and D. Grunwald, "A vision for technology-mediated support for public participation & assistance in mass emergencies & disasters." *In Proceedings of the 2010 ACM-BCS visions of computer science conference*, British Computer Society , pp. 8 (2010).

[4] S. Umair, "Mobile Reporting and Journalism for Media Trends, News Trans-mission and its Authenticity." *Journal of Mass Communications and Journalism*, 6, 323 (2016).

[5] J. Mills, P. Egglestone, O. Rashid, and H. Väätäjä, "MoJo in action: The use of mobiles in conflict, community, and cross-platform journalism." *Continuum*, vol. 26, no. 5, pp 669-683 (2012).

[6] F. Al-Turjman, "Impact of user's habits on smartphones' sensors: An overview." *In HONET-ICT*, IEEE, pp. 70-74 (2016).

[7] C. Dimoulas, A. Veglis, and G. Kalliris, "Application of Mobile Cloud-Based Technologies in News Reporting: Current Trends and Future Perspectives." *In Mobile Networks and Cloud Computing Convergence for Progressive Services and Applications*, IGI Global, pp. 320-343 (2014).

[8] F. Guribye, and L. Nyre, "The Changing Ecology of Tools for Live News Reporting." *Journalism Practice*, vol. 11, no. 10, pp 1216-1230 (2017).

[9] L. Vrysis, N. Tsipas, C. Dimoulas, and G. Papanikolaou, "Crowdsourcing audio semantics by means of hybrid bimodal segmentation with hierarchical classification." *Journal of the Audio Engineering Society*, vol. 64, no. 12, pp. 1042-1054 (2016).

[10] S. Quinn, and I. Burum, *Mojo: The mobile journalism handbook: How to make broadcast videos with an iphone or ipad*, Focal Press. (2015).

[11] N. Vryzas, E. Sidiropoulos, L. Vrysis, E. Avraam, and C. Dimoulas, "A Mobile Cloud Computing Collaborative Model for the Support of On-Site Content Capturing and Publishing." *Journal of Media Critiques*, 4, 14 (2018).

[12] L. Vrysis, N. Tsipas, C. Dimoulas, and G. Papanikolaou, "Mobile audio intelligence: From real time segmentation to crowd sourced semantics", *Audio Mostly 2015*, ACM, p. 37 (2015).

[13] N. Tsipas, L. Vrysis, C. Dimoulas, and G. Papanikolaou, "Efficient audio-driven multimedia indexing through similarity-based speech/music discrimination." *Multimedia Tools and Applications*, vol. 76, no. 24, pp. 25603-25621 (2017).

[14] L. Vrysis, N. Tsipas, C. Dimoulas, and G. Papanikolaou, "Extending Temporal Feature Integration for Semantic Audio Analysis", *Audio Engineering Society Convention 142* (2017).

[15] N. Tsipas, L. Vrysis, C. Dimoulas, and G. Papanikolaou, "MIREX 2015: Methods for Speech/Music Detection and Classification"

[16] Apple «iOS Developer Library», https://developer.apple.com/library/ios/ (last access 20/7/2018.

[17] ANSI Standards S1.4-1983 and S1.42-2001.

[18] S. Thomas, S. Ganapathy, and H. Hermansky, H. "Recognition of reverberant speech using frequency domain linear prediction" (No. LIDIAP-REPORT-2008-063), IDIAP. (2008).

[19] Zoom, "IQ6 Professional Stereo Microphone", http://bit.ly/2RoFnjX, last access 1/1/2019.

[20] Zoom, "IQ7 Professional Stereo Microphone", http://bit.ly/2CKp7kR, last access 1/1/2019.

[21] Rode "VideoMic Me", http://bit.ly/2s3GVBk, last access 1/1/2019.