SoundLog
Make More Noise!

André Ferreira
Instituto Superior Técnico
Universidade Técnica de Lisboa
afmf.pt@gmail.com

ABSTRACT

Sound is traditionally used for its content along with automated speech recognition techniques to understand what is said. But sound, besides its semantics, can be used to gather useful information.

We propose SoundLog, a system that analyses sound and crosses it with contextual and autobiographic information to gather insights from different situations (public presentations and meetings). Our goal is to provide users knowledge regarding those situations, both in real-time and later on, without the need of training or use of devices. This can help them to correct or adapt their behaviour, and to review past events in meaningful ways.

Author Keywords
Low Sound Features, Subject Identification, Speech Recognition, Scenarios, Framework and Interface

INTRODUCTION

The sound captured through a microphone is a rich information source that can be used to withdraw exact inferences about the speaker, its environment and social events. It is a modality that supports a wide set of features, such as conversation detection, activity recognizer, localization classification, social net structure discovery. However, it is traditionally used for its content, resorting to techniques like Automated Speech Recognition to understand what it’s said.

But the sound itself, regardless of its semantics, can provide information. For example, in the course of a class, if the level of noise increases, this can indicate that some slide generated comments or that the students lost interest. This can be used to create a log of the student’s interest in the several parts of the class, to improve it where needed. Similarly, in a meeting or lecture, it can indicate when a polemic subject was referred. If we go deeper, and analyse the sound’s frequency, we can identify when a person is speaking and by analysing the timbre, we can recognize who spoke.

Despite sound’s potential as a robust way to obtain information, microphones and inferences based on it have not been given much attention in previous research. Our goal, in SoundLog, is to create a system that analyses sound and relates the information obtained from it with contextual and autobiographic information to gain insights about several situations. More than just providing the framework to analyse the sound in rich and relevant ways, we present that information to the users in meaningful ways, based on different scenarios. We will take care to do so in such a way as not to unduly distract the users from their tasks but still make the information readily available if needed. Our final goal is to create a system that doesn’t need any type of training (for example training for voice identification or speech recognition) and the use of any type of device (head sensors, video cameras, etc.). With the application in place, users only have to start/stop the information capture and then can end.

In order to find out what kind of information can be relevant, we surveyed several previous works in the area. We focused on solutions that analyse the sound itself, oral and written conversations. We decided to analyse oral and written conversations because we wanted to see how to represent dialogues (and both sound and text are orthogonal to the visualization). From the first we determined the most important features (volume for a possible identification of what’s problematic or understood; frequency to differentiate between voice and music or noise, timbre for voice identification, etc.), that can be extracted from sound captured in realistic contexts. From the works that analyse oral conversations, we conclude that voice identification (with subsequence recognition of speaker) and speech recognition must be made in all scenarios (to see who spoke and what was said). The works that analyse textual conversations showed that all the collected information should be logged and how the exchanges between participants can be shown, making salient patterns apparent.

Having identified the requirements and effective sound analysing system should possess, we decided to create a reusable framework that analyses the sound characteristics and provides an abstraction layer that enables the quick development of sound-based applications. We will then consider three different scenarios: public presentations (class and lecture) and meetings. These scenarios will allow us to test the versatility of the framework by making use of a wide range of sound-based information, in different contexts. We will create a tool that has an extensible
structure on which different details will be given more emphasis depending on the scenario that is considered (e.g. who spoke is more relevant in a meeting than in a lecture). The interface will adapt accordingly. There will be a real time interactive visualization during the events. Afterwards it will be possible to interactively analyse the logs, and produce reports.

In this work, our main contributions are:

- The identification of which sound features we must collect that might be relevant to understand the surrounding context.
- An effective way to visualize sound-based contextual cues in such a way as to provide deep insights to the user.
- A prototype system embodying our ideas, that allows sound to be analysed in three different scenarios: meetings, classes, and public presentations.

In the next section, we will describe relevant related work in the area focusing on works that analyse sound and works that analyse oral and textual conversations. Then we talk about SoundLog implementation by describing the scenarios where our system is applied, its requirements and architecture. After the complete description of our solution, we present evaluation results and conclude enumerating pointing relevant future work.

**RELATED WORK**

We study works that analyse the sound and works that analyse oral and written conversations to determinate which sound information we should capture and how to display it. We study both types of conversations because we wanted to see how to represent dialogues, and being sound or text it’s orthogonal to the visualization.

Regarding works that analyse the sound itself we highlight SoundSense [8], a scalable framework for modelling sound events through mobile phones. The system runs uses its microphone to capture sound (without any other auxiliary system), and can identify general types of sound (e.g. music, voice). Also giving sound a central role, NoiseSpy [7] is a noise monitoring system that allows users to explore a city area while collaboratively collecting and visualising urban noise levels in real-time. It looks at individual journeys to access personal exposure level to visualize if the users are prone to the risk of serious noise pollution.

Since we need to apply our application to different scenarios (classes, lectures and meetings) we decided to study works that analyse oral and written conversations.

Conversation Clusters [2] is a system used in meetings that through Speech Recognition, creates sets of conversation topics. The sets are arranged in a shared tabletop allowing its visualization by participants of the meeting. It has also the thread visualization type, where there are displayed lines of topics that were discussed over time (from left to right).

In Conversation Clocks [3] we visualize a conversation up to four people seated at a table that has a projection surface. During the group interaction, microphones capture the sound stream of all participants. The visualization of such flows is then projected, in real time, on a projection table. The flow of each person sound captured has a different colour and is placed in a circular timeline in the form of rectangles (whose size depends on an average amplitude of speech). Conversation Votes [4] is an extension of the Conversation Clock. As the last, through visualization provides indication of the domain, changing shifts, mimicry, and other aspects that allows people to make an assessment in the third person of their participation. Then extends this social mirror, each participant has two buttons, which allow good or bad feedback to be shown in the visualization. Participants may encourage current speaker giving positive votes to continue or negative votes to stop. In Visualizing Remote Voice Conversations [9], conversations are visualized through its volume, pitch and content. It allows portraying sarcasm, subtle clues and questions in a way that its transcription can’t.

Turning into more complex systems, we have Egocentric Analysis and Visualization of Instant Messaging Activity [1]. It’s a work that focuses on privacy issues using the communication of an individual with its contacts. To do so, (a) it analyses the social and semantic aspects of the communication; detect features such as frequency, dominance, density and longevity; and (b) build a compact structure where you can view all these resources. The structure consists of a ring that has the individual at the center represented in black. All its contacts are around it and are represented by garnet circles, whose size varies depending on the number of communications. Points on the radial lines from the center of the ring (the individual) and the contact represent conversations between both (timeline) and the more brilliant they are; the greater is the conversation activity.

The works that more resembles our goals are the 2010 works Human Semantic Interactions in Meetings [10] and PrimaVista [6]. In the first is proposed an approach for capture, recognition and visualization of human interactions. They consider that the interactions involved are consumed with semantic, i.e., user intention or attitude towards a topic and are different from physical intentions (for example turn talking or addressing). With this in mind, they developed a system which consists on the following three layers. Interaction Capture consists in the physical layer that deals with capture environment, devices and methods. Video cameras, microphones and motion sensors, are used to record the meeting content and tracking participants’ head movement. Interaction Recognition
consists in the structural layer which analyses content extracted from the audio-visual and motion data. Speech recognition engine extracts the speech features from audio data. Motion Data Processor is responsible for analysing people’s head gestures. Annotator is used for label features from these audio-visual data. Interaction Visualization consists in the presentation layer, which offers an interactive user interface for browsing human interactions as well as meeting content.

PrimaVista is a system that contains different visualization techniques to support a meeting. It’s made automatic speech recognition with multi-sensory data captured from the meeting into a unified visualization interface that provides real time interactions and also browsing of past meetings.

However, none of the works is identical to what we intend to implement. None of them makes use of all the information that sound can provide (low sound features, voice identification and speech recognition) – usually they only use one of them. Also, despite most of the works have a real time interface; none have it applied to different scenarios. The status/past report is never produced and in some works it is necessary to do training or use devices.

THE SOUNDLOG SYSTEM

The work studied allowed us to identify what we can extract from the different scenarios and a list of requirements from systems that use sound as capture basis. In this section, we’ll start by describing those scenarios and requirements, talking about what information needs to be captured, and how to visualize and analyse it. After that we describe the overall architecture and implementation of the system. This consists on a framework for information capture and an interface for visualization an analysis of that information.

The SoundLog Scenarios

We’ll use our application in public presentations (classes and lectures) and in meetings. We decided to address these scenarios because we can extract enough information through for analysis, thus helping to improve them.

In the public presentation scenarios, the important thing is to realize what parts of it have generated more controversy and what were better perceived (this is easily done by analysing the noise/volume). It is also useful to find out who made more questions and if the answers were long (for this we must have a voice identification mechanism). Another important point will be to discover what slides were more complicated to describe and at the same time trying to figure out where there were delays in the presentation.

In the meeting scenario we must have an idea of what happened along the same. By elaborating topics through what is recognized. Having a sense of who’s talking about and how much time it spoke (or even speech percentage itself) it’s also be relevant. It enters again the voice identification mechanism but this time combined with a speech recognition mechanism.

The SoundLog Requirements

The system requirements were made taking in mind the information obtained from the work study and all SoundLog system goals. We divide the system requirements into different categories (capture, visualization, analysis and others).

One of our major goals is to capture information that comes from sound itself. To accomplish this, and according to the described works, we can start by capturing the sound low feature characteristics: volume, frequency and finally timbre. To get more information from sound, we need to discover who said it and what is said. For this, we need find a way to identify different voices for the speaker identification and we need to implement an automatic speech recognition engine that recognizes what is said. By doing this, we can easily discover who spoke during a meeting and calculate oral rates (since in a public presentation the oral speaker is the most intervenient one).

We also contextualize all the scenarios though the different recognized words. In order to get even more contextual and auto-biographic information, we need to go one step forward and capture information that comes from presentations. Information about what happened in the different slide presentation is a quick way to discover problematic slides (problematic parts of a class or lecture).

So that we can see all the captured information, we first need to build an interface that captures all this data in real-time. The interface also needs to be able to support the three different scenarios (class, lecture and meeting). This can be acquired by developing a different set of data visualization plugins. Each plugin will demonstrate different data and then we can choose which plugins to select, accordingly to the type of scenario. All the information must be accurate and for that, the system needs to have temporal notion. Knowing what happened is important, but without knowing when happened it becomes useless.

To improve the different types of scenarios, we need to get different tools to properly analyse what happened. But for that, we first need to persist the scenarios. The system needs to capture the information in real-time, pass the visualization to the different plugins to visualize it in different ways and finally persist all the data in order to allow future analysis. For an easy and fast analysis, we need to access all the information from a scenario in a quick way. That is done by allowing the user to use pan and zoom commands in the interface and also by using functionalities.

One of these functionalities may be a total status report that can be used to reflect all the scenario status.

We also have to work in a way that none special device (besides the computer itself) is needed in order to not
deflect the user and other participant’s attention. By doing this we are capable to use our system in any situation. To make it easy to install the application whatever we intend we need to build it in such a way that will make it independent. Our desire is to build an independent real-time information capture interface that allows users to use it anywhere and analyse it later on.

**The SoundLog Implementation**

The solution that we propose assume the format of a Python application that accomplishes the goals and requirements detailed in the previous chapter. These requirements can be divided in two types: the capture of information and the visualization and analysis of it. This separation eventually dictates the architectural decisions that we have taken.

So, in order to capture the information, we decided to implement a generic, reusable, Framework. This Framework (.dll), that meets all the capture requirements previously described, was implemented in the C# language. Thought it we can get the low sound features, make voice identification and speech recognition. In order to get info from presentation, we develop a PowerPoint add-in that provides us with images and textual content.

Afterwards, we develop the real-time capture interface in Python language. This interface uses the Framework to get the information and with some processing we can visualize and analyse it. It is also made in a way that the different scenarios can use it. This is done because the interface consists on different plugins where each of them demonstrates different types of information. We then can choose which plugins to use in each type of scenario (classes, lectures or meetings).

We have also always in mind, throughout the development of this system, that we wanted to implement an independent application that doesn’t need any training or devices to work properly. These requirements are ones of most importance because all the systems that we describe that were trying to do something like we have done were always too dependent (programmatically or physically) that simply wouldn’t be possible to use them in a real live class, lecture or presentation.

**Framework**

The first thing that we have done in the framework, was to capture the sound from the microphone and calculate the volume of it. For that, we decided to collect 44100 samples at a sample rate of 16Hz and with 2bits per sample. This are the number of samples required, per second, to transform a continuous audio signal from a computer microphone, into a discrete audio signal $^1$.

With the sound samples captured, we needed to find a way to calculate the sound volume (or sound level) in dB and the sound frequency in Hz. That was done by passing the discrete audio signal captured into a digital signal that can be properly analysed. To do this we had to apply a Fourier Transform algorithm. Since the normal Fourier Transform consists in a $O(N^2)$ algorithm, we decided to use an optimized version of the algorithm called Fast Fourier Transform (FFT) $^2$. For this part, we didn’t reinvent the wheel and decided to use the Math.NET Iridium – Numeric Foundation dll $^3$.

With the low sound characteristics volume and frequency being captured, and knowing when a person speaks because of the frequency, our next step was to discover who spoke. For that we turned into another low sound feature, the timbre. People can speak with the same frequency but the timbre usually is pretty different. Through research, we identified that the most common way to obtain timbre in real time was to use the Mel Frequency Czestrum Coefficients algorithm $^5$. Because we didn’t find it in any framework, at least available for use, we decided to implement the algorithm. This was promising because the first step of the algorithm is to calculate the Fourier Transform, which is one step that we already have done. After that we had to apply triangular overlapping windows in the spectrum obtained in order to map its powers onto Mel scale. After some extra calculations, we obtain the timbre that consists in a large set of values. We then use the Vector Quantization algorithm $^5$ which reduces the size of this set to only sixteen values. Of the sixteen, only the largest of them needs to be considered because the other fifteen are too small and can be discarded. We consider this acquired value to be the timbre obtained in real time.

When a voice is identified, its timbre is persisted in a identified voices array. This array is populated every time the framework is initialized. We decided to persist the array because the same user usually deals with the same voices. In that way, an identified voice in one scenario is already identified in the futures scenarios. This is useful because we can label the identified voice in the Python. So, if one voice is labelled in one scenario (e.g. John), if its timbre is identified in future scenarios, we already have its voice label and John will instantly be identified.

Knowing who spoke, we passed into knowing what is said. So, the speech recognition has not been forgotten and this is

---

$^1$ Sampling Rate Obtained from https://secure.wikimedia.org/wikipedia/en/wiki/Sampling_rate in February 2010


done through the Microsoft Speech API (SAPI) 4. We build it to recognize the English language because Portuguese is still not available. We decided to use the Microsoft speech API because it’s already available in Windows OS (Vista or above). Since it’s the OS present in almost every computer we didn’t need to install any extra software to make it work (and so our implementation remains independent).

PowerPoint Add-In
Before we start the build of the interface, and to obtain more contextual information, we develop a PowerPoint add-in to collect information from presentations (Figure 37). We extract the textual content of the slides (plus notes) and also images of them. These images are then shown in the Python application every time that is made a slide transaction in the PowerPoint. Both the slide transactions and also the file presentation name are also provided to the Python interface. This is done by writing info in a file that the interface will read every second.

Python Interface (Scenarios)
This interface handles with the Visualization and Analysis requirements of the SoundLog system. Needless to say that will also handle the Other requirements, which are present during all the code created. So, we need to get the captured information, we need to visualize it and finally we need to analyse it. This made our python application be divided into three packages: Auxiliar, Plugins and Tools. The Auxiliar Package is responsible by making available to Python all the information captured. This package is also responsible to provide the needed information to the Framework or the PowerPoint Add-in so that they could work properly and by making the data available for future analysis. The Plugins Package and the Tools Package are both responsible by the visualization of the information and analysis of the information.

The auxiliar package main functions are to get the information captured by the framework and in the presentations, to provide information to the Framework and to save the captured information for future analysis. To create an independent system we needed to create an interface that only depended on Python modules. If the interface only depends on these modules, we can create an executable which contains the necessary Python modules inside (and then we don’t need to install Python). With this in mind, we had to put aside the idea of using IronPython 5. As we only wanted to use Python modules, we had to resort to the ctypes module. The problem of using this module is that it only supports calls to a C++ language dll (and not our C# language dll). We had to change our Framework so it became a Com InterOp dll, something we already knew we would have to occur. We also needed to develop a C++ dll (the one who is called by the Python) that consists in a Wrapper dll. This dll is called by though methods defined in the Auxiliar Package.

To access the PowerPoint Add-in information was simply to access the files that are in the directory created for the presentation (the directory with the hash name). Every time that we are in a PowerPoint presentation full mode and change the slide, the directory name, the presentation name and the slide to which the transition was made, are written into a .txt file by the PowerPoint add-in and that file is parsed by the interface.

In the Speech Recognition, we decided to use grammar (which contains a set of words to recognize) instead of recognizing all words spoken (the current speech recognition engine doesn’t properly work in this conditions). The chosen words consist in the most frequent words from presentations and other documents provided (and we remove the most common ones).

The Auxiliar Package accomplishes the real-time capture requirement because it provides the SoundLog application, at every second, with all the collected information. Furthermore, it also provides the means to the collection of data to work properly and the means to analyse the information in the future (record everything requirement). By helping capturing information and doing the speech recognition grammar.

The Plugin Package deals with the plugins and temporal notion system requirements of visualization type. It also provides the means for the information to be analysed. We initially started by developing the Noise, Speech and Subjects plugins whose aims are to, respectively, provide information about sound level, recognized words and identified subjects (voices). Later on we develop the Timeline plugin that provides temporal information and also the Slides plugin that demonstrates information about the slides of the PowerPoint presentations that are visualized during each scenario. The current SoundLog layout is displayed in Figure 1.

We develop a main class plugin whose provides a set of functionalities that are available for the different plugins. As the total information existing in a scenario can be huge (remember that a class usually takes 90 minutes), it was developed pan and zoom operations. We can execute them through the toolbar buttons or by direct manipulation of the plugins. Without these pan and zoom commands, it would be impossible to clearly analyse the captured information. The zoom operation has four types of granularity. The

---

4 Microsoft Speech API (SAPI) 5.3 Obtained from http://www.microsoft.com/speech/default.mspx in November 2009
5 IronPython Obtained from http://ironpython.net/ in December 2009
system starts with the lowest one (one second), but we can then zoom out through the ten seconds, thirty seconds or one minute granularities.

If the user wants to dispose the plugins in a different order, he may swap them with drag and drop commands or hide them by closing the one pretended. The plugins order and visualization are also available in the Plugins Tool.

Concerning the Noise plugin, this is intended to demonstrate the volume in decibels at every instant. When the plugin is selected (with the mouse over it), a red vertical bar is shown. It corresponds to a decibel scale that is used in the plugin. As the volume of the human voice is typically between 40dB and 60dB, we climbed this bar so it would have a greater range for this interval.

In the Subjects plugin, we visualize the identified subjects in each instant. When a subject is identified for the first time, a coloured horizontal line will appear in that instant. Behind that line is displayed the name of the subject whose label can be defined in the Subjects Tool. From that moment on, in every instant where the subject is identified, the horizontal line will be continuous, otherwise it will be dotted. Depending on the height of the plugin, a different set of subjects may appear. If there are more subjects identified that we potentially can demonstrate, arrows will appear in the left of the plugin to see the other subjects.

In this plugin, we identify if a subject is identified through the Framework. When the granularity is different from one second, we make a logic OR of the corresponding range in the smaller granularity. So, if Subject 1 is identified in the fiftieth second, in the smaller granularity it will only appear in that second, but in the higher granularity the line will appear continuous for that entire minute.

Concerning the Highlight functionality, we can left click the name of one identified subject and all the instants where that subject was identified will be highlighted (in Figure 1, the word 'framework' is highlighted). By clicking again the highlight is removed. Through direct manipulation we can only make unique selections. If we want to make compositions of subjects we can access the Highlight Tool with a right click over the subject’s label.

For the Sliders functionality, we use the counter at the right of the plugin. This counter is limited between zero and the maximum number of subjects identified during the scenario, in one slide. There are only visible the slides that have an equal number or more persons identified, then the defined in the Sliders Subjects counter.

In the Words Plugin, we visualize the recognized words at each instant. Like we previously explained, the recognized words consist in a set of words that are passed to the speech recognition. These words are provided by the presentation and/or by documents inserted in Documents Tool.

Because there may be a lot of words recognized in each instant with granularity bigger than one second, we defined that only the three most important words are displayed (when the instant has more than three recognized words). The importance rate is related with the words frequency in the provided files or presentations.

The colour of each recognized word will be white or black depending on the PowerPoint presentation number. For

![Figure 1: SoundLog layout](image-url)
each presentation is defined a distinct sequential identifier. When the presentation number is even, the word is white and black otherwise. This helps to easily identify the moment where the user switched presentations. There is still one other possibility. The colour of the word can be grey if it belongs two both presentations. All these colours differ when we highlight one. Clicking on one word will change its colour to green, red or blue it the word is in, respectively the most recognized ones or none of the previous. In figure 51, we used the Highlight Tool to highlight the fala and frameword words. Fala is one of the most recognized words and Framework is in the middle set.

The Sliders functionality is also present in the Words plugin. It works exactly like in the Subjects plugin but instead of subjects, deals with words. The counter is now limited by the maximum number of words recognized in one slide.

The Slides plugin contains detailed information about what happened in the scenario, providing more contextual knowledge. As previously noticed, it consists in a plugin that interacts with other plugins through the Sliders functionality. Whenever occur a slide transition in the PowerPoint presentation, a miniature image of the slide to which the transition went is visualized. Also the grey dotted vertical transition line is defined in all plugins.

According to the granularity, it may exist a lot of transitions in each instant. When we don’t have enough space to display all of them, we group them in the last miniature. If we click one of these miniatures, a middle size image of the slide, or set of slides, appears. The visualization of several slides is dynamic. The size of each middle image starts with the default one. If needed, the slides shrink in order to be able to see all of them. With the middle size slides image, we can access in the slides information by clicking on it. It’s then created a new frame with the slide image selected and with information about it: presentation name; number and appearance of slide; max noise level detected; identified subjects; and recognized words.

The use of the sliders is attached with the information obtained for each slide. This functionality is one of the most important of the SoundLog system. We can easily detect the most discussed, most relevant or most noisy slides. When we know which slides brought complications to a class, we can change them in order to make a better presentation. For example, if a slide is the noisiest one, that could mean that students weren’t paying attention. Or if a slide has a lot of identified subjects (most discussed) it could mean that students don’t understand it and have doubts about that part of the class. On the other hand it could also mean that students like that part of the class. Through the different plugins we can contextualize the situation and after that, provide the students with a better presentation.

The Timeline Plugin is responsible by the temporal notion of the system. It provides information about time through two different timelines: one that indicates the actual time that we see displayed in the middle of the plugin (actual timeline); and another that provides information about where we stand in the presentation displayed in the bottom of the plugin (total timeline).

The actual timeline contains time information in the last instant of the scenario and in every x instants. In the total timeline, a green rectangle is filled and placed over it representing what we are seeing, according to all the time of the presentation. Besides this, we can click over the total time timeline and we pan to the area that was clicked. The total timeline also provides information about highlight instants. When we highlight something, all the instants that have that something are referenced in the timeline with a green vertical line. The timeline plugin provides a lot of temporal information. It provides the user with the temporal notion needed in these kinds of systems.

All the described plugins display the information captured by the Framework. The Plugins Package is clearly most related with the visualization requirements of the system but it has some of the analysis requirements linked to it. The pan and zooms operations, who are crucial to analyse the system, are present in all the plugins. The plugins are also related between them. If we pan or zoom in one plugin, all of them pan or zoom. Also, the Timeline plugin is related to all the others and the Sliders plugin is mostly related with the noise, subjects and words plugins. The plugin combination is then present. Some functionalities were also defined in them but they are deeper in the Tools Package.

The Tools Package is responsible by the rest of the analysis requirements: functionalities and status report. Besides these requirements, the different tools that were previously introduced are all part of this package. This package helps controlling the flux of the program with the New and Load Presentation Tools. It provides the means to analyse the system with the Highlight and Report Tools (functionalities). The Documents and Subjects Tools allow contextualizing the information. The Plugins Tools provide supports the system plugins layout and finally the Help Tool provides tips about the entire system. Simply explained, this package assists the user in several aspects besides the analysis requirements.

Both New and Load Presentations Tools are used to control the initial flux of the program.

After the creation of a new scenario, and before we start the information capture, we can use the Documents Tool to had more information to the Speech Recognition. We can insert PDF files (.pdf), Word files (.doc and .docx), Excel files (.xls and .xlsx), PowerPoint files (.ppt, .pptx, .pps and .ppsx) and Plain Text files (.txt and .rtf). The text from
these files is extracted into a .txt file and then we built a grammar that we sent to the Speech Recognition application to parse. With these files we aid the Speech Recognition and by doing that, we aid our system. It is a very important tool because without it Speech Recognition wouldn’t be available in a scenario that doesn’t have presentations.

Concerning the functionalities that support the information analysis, we built the Highlight Tool and the Report Tool. The highlight is available in the words and subjects plugins through direct manipulation but there only a unique selection can be made. If we want to make multiple selection of words and/or subjects to highlight, we only can do it using the corresponding tool. Through the Highlight Tool we can visualize all the identified subjects and all the recognized words and afterwards we can select the ones we want to highlight. There are also available options that allow us to select sets containing the most and least identified subjects and recognized words. As previously explained, the highlight is then propagated throughout all the plugins and the highlight instants are visualized in the timeline plugin.

The other tool built to analyse the system was the Report Tool. After the ending of a scenario, either when we stop the capture of information or when we load a scenario, we can access this tool that provides a summary of what happened during the scenario and has two visualization methods: textual report and visual report. Both reports provide information about: the maximum and minimum noise level detected; the top three identified subjects and the top three recognized words (both per number of times); and finally information about the slides: most and least discussed, relevant, noisy and longest.

Although we specify the default plugins for each type of scenario, sometimes the user might need to use different plugins from the ones that the configuration file dictates. For that reason, we built the Plugins Tool. A tool where we can define the order of the plugins and which to show or hide.

In order to give more information to the Subjects plugin, we needed to find a way to label the identified subjects. That is the main purpose of the Subjects Tool. The identified subject is provided by the Framework and we can label it. As related before, all this subjects’ information is persisted and if intended, we can use subjects identified in previous scenarios. This particularly good because the system normally used by the same person and we don’t need to be labelling him all the times.

EVALUATING SOUNDLOG
The SoundLog system was developed in two iterations: first functional prototype and final prototype. When we ended the development of the first functional prototype, we made a heuristic evaluation of it and afterwards we developed the final functional prototype fixing the errors found. When we ended the final prototype, we tested it in two types of scenarios (classes and meetings) in order to check if the SoundLog system really works.

**Final Functional Prototype Evaluation**
All the work done so far helped us to identify the major priorities of the system. In order to conclude if the system really works and if it can provide correct information we first define the issues that must be evaluated. So, we need to see if the system can be used in a non intrusive way, if we can log all the information captured and if that information is well captured: sound level, voice identification, speech recognition and slide management.

For that purpose, we had one live class test at Instituto Superior Técnico (IST) and two live meeting tests. The first also at IST with three participants and the other at BES ESData with two participants.

**Noise Level**
Through the analysis of the recorded data we could clearly identify that when the noise level of the class or meeting was higher (when participants talked), the noise level went up and when they didn’t stay down. In order to analyse this data with more efficiency, we decided to make another test (an independent one). In this test we decided to use the SoundLog system to capture information. In every two seconds we snapped our fingers in order to see if the system calculated the noise from the snaps. The finger snaps were correctly calculated in every second and with a dB level average of about 30dB, which means that has a good accuracy. In the live tests, the sound level was also correctly calculated because it was between the 40 and 60 decibels of the human voice.

**Speech Recognition**
Taking in consideration that we needed to test the speech recognition where we have documents to build a grammar to aid the speech recognition application, we purposely used the class scenario to test this system attribute. Using the PowerPoint presentations, SoundLog built the grammar and then provided it to the Speech Recognition application.

During the entire class the speech recognition only identified the word não. This was a poor result but, unfortunately, one that we already expected. The Speech Recognition, besides being a technology that still needs a lot of improvement, in our case, is associated with the English language. So, with Portuguese presentations (and words) the recognition becomes quite hard.

So, in order to properly test our system speech recognition, we decided to do another independent test. To do the test we created a PowerPoint presentation with ten slides. Each slide as only one word on and we pass by all of them and read the word: Sound, capture, framework, real, time, interface, noise, speech, subjects and slides. We have done
this test with five different times, on each we decreased the duration on each slide transition by one second, starting with five seconds and ending with one second per slide. For each interval we tested the system speech recognition five times. The ratios for each interval are displayed in the following graphic (Graphic 1).

![Graphic 1: SoundLog Recognized Words Ratio per Interval](image1)

As told before, we decided to use the grammar mode to improve the words recognition. Nowadays, the speech recognition engines, at least the one that we can access free of charges, are still very inefficient when they try to recognize everything that is said. Another way to improve the recognition is to do a lot of training by reading for the computer’s speech recognition application. Although we couldn’t test the words recognition in a real situation, with our test we determinate that the probability of a word to be recognized when spoken after another important one, is bigger than eighty percent when the range between them is higher than three seconds.

Subjects Identification

The identification of different subjects consists on one of the most important attributes in a meeting. In public presentations the oral speaker is the most intervenent, but in meetings there are many people and by simply calculating the oral taxes of them, we can improve the meeting. We tested this attribute during the meetings tests. The first meeting had three participants and the second meeting had two participants.

In the first meeting, there was miss recognition in one instant defining Subject 3 appearance. Besides this instant, the others went quite well and the subjects were correctly identified. Both subjects were identified in the first ten seconds and since then correctly identified during the meeting. In order to avoid lots of miss recognitions, we built an algorithm that only identifies one subject when its timbre is detected during several seconds on a row. This helps with the correct identification of one subject. In other words, when a subject is identified, it is correctly identified. Although, for a subject to be identified for the first time it takes some seconds of talking. In the second meeting we had three participants but most of the talk was made by only two. After their identification in the very first seconds of the meeting, the third participant, the one who talked less time, it was only identified at the end of the meeting, despite talking brief periods of time during all of it. To build a system that doesn’t needed any type of training or device attached to the different participants; we had to decide between good or lots of identifications.

Analysing the sound data recorded from the two meetings we concluded that participants were correctly identified over eighty percent of the time (when a participant was identified, it corresponded to the participant who spoke) but they were only identified sixty percent of the times (Graphic 2).

![Graphic 2: Subject Identification in the Two Meetings](image2)

In this plugin our main goal is to reflect the PowerPoint presentation transitions by displaying the image of the slide to which the transition was made. We also need to associate the information capture during each slide to the slide itself.

Slides & Timeline

Both Slides and Timeline plugins only depend on programmatic events. In the Slides Plugin all the slide transitions were correctly parsed and recorded and the information presented in each slide also corresponded with the information displayed for that time interval.

Summary

The next table contains the satisfactory questionnaire results.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Captured Information</td>
<td>3.67</td>
</tr>
<tr>
<td>Visualization (Plugins)</td>
<td>3.67</td>
</tr>
<tr>
<td>Functionalities</td>
<td>4</td>
</tr>
<tr>
<td>Tools</td>
<td>3.5</td>
</tr>
<tr>
<td>Overall</td>
<td>3.8</td>
</tr>
</tbody>
</table>

Table 1: SoundLog Satisfactory Questionary Results
Associating the different functions of the prototype we got an overall result of 3.8 (in five). This helps to prove that SoundLog can really improve the scenarios and also to aid in the answer of the previous questions.

CONCLUSIONS

In this paper, we presented a new way to capture contextual and sound information in order to gain insight from different situations (class, lectures or meetings) and with the purpose to improve those situations.

Through the analysis of different works, we discovered that the most relevance sound features that we needed to achieve were the ones that allow us to calculate the sound level, identify different subjects (identify who said) and recognize different words (recognize what was said).

We then created a prototype that used a Framework to capture all this information and also was aid by a PowerPoint Add-in to get more contextual information (information from presentations). The information obtained was then visualized by different plugins: noise, words, subjects, slides and timeline. At the end of each scenario (whose are persisted), we could analyse that information.

Although we got satisfactory results from the evaluation of the prototype, we also found that there is space to improve both speech recognition and subjects identification. In the first case, we might find a way to recognize the Portuguese language and get the truly most important words from the data presented. In the second case, the identification of different subjects works properly because identifies correctly the ones who spoke. However, an improvement in the algorithm might occur in order to make a bigger number of identifications. Another feature that might be useful is a way to compare different scenarios. With this document we concluded that we can improve different scenarios. So, if a professor uses the SoundLog application in different classes about the same theme, he might want to compare them in order to achieve the better one.

REFERENCES


6. Heimonen, Ovaska, T., Turunen, S., Hakulinen, Rajaniemi, J., Räihä, J., & Kari-Jouko (July 2010). Visualization of Multi-Sensory Meeting Information Support Awareness (PrimaVista), In Information Visualisation (IV), 2010 14th International Conference, At London, United Kingdom, ISSN 1550-6037, Pages 194 – 199


