A ROS framework for audio-based activity recognition

Theodoros Giannakopoulos
Institute of Informatics and Telecommunications
NCSR "Demokritos"
Athens, Greece
tyiannak@gmail.com

Georgios Siantikos
Institute of Informatics and Telecommunications
NCSR "Demokritos"
Athens, Greece
siantikosg@gmail.com

ABSTRACT

Research on robot perception mostly focuses on visual information analytics. Audio-based perception is mostly based on speech-related information. However, non-verbal information of the audio channel can be equally important in the perception procedure, or at least play a complementary role. This paper presents a framework for audio signal analysis that utilizes the ROS architectural principles. Details on the design and implementation issues of this workflow are described, while classification results are also presented in the context of two use-cases motivated by the task of medical monitoring. The proposed audio analysis framework is provided as an open-source library at github (https://github.com/tyiannak/AUROS).

Categories and Subject Descriptors

H.5.5 [**Information Systems**]: Sound and Music Computing—*Signal analysis, synthesis, and processing*
; I.5.4 [**Computing Methodologies**]: Applications—*Signal processing*

Keywords

ROS, audio analysis, open-source,feature extraction,audio segmentation,classification

1. INTRODUCTION

The overwhelming majority of the research efforts in robot perception focuses on computer vision systems along with scanning procedures that map and visualize the robotâĂŹs environment. This is due to the fact that perception is mostly used as a preprocessing step to allow robots to recognize people and obstacles. Audio-based perception in the context of robotic applications is mostly speech-related, since the speech channel of information has been widely used in human-robot interaction applications. However, non-verbal audio analysis can also play an important role in perception of robotic systems, as it may lead to perception of rather important attributes of the environment that are hardly or impossibly recognizable by the visual sensors. Even in cases that visual information is sufficient for recognizing events and objects, machine hearing can be an important complementary channel of perception, especially in the context of multimodal fusion techniques.

Robot Operating System (ROS, [10]) is a framework that facilitates rapid prototyping of robotic software. It provides OS-like functionalities such as hardware abstraction. A publish/subscribe messaging model is applied for communication between different ROS-based processes (nodes). In addition, ROS nodes do not necessarily exist on the same PC, which means that this architecture is designed with distributed computing in mind. Nodes communicate through "topics", i.e. asynchronous many-to-many communication streams. Nodes are not aware of which other nodes they are communicating with. Instead, nodes that are interested in particular data *subscribe* to the respective topic. Similarly, nodes that generate data *publish* to the respective topic. According to the ROS architecture, there can be multiple publishers and subscribers to a topic.

ROS Open-source Audio Recognizer [11] provides a supervised framework for audio classification along with a feature extraction stage. Sound is accessed through ALSA (Advanced Linux Sound Architecture), while noise removal is achieved through a spectral substraction technique. Perceptual Linear Prediction [7], [2] coefficients are used as features, calculated through a Matlab - Octave module. Classification is achieved through an one-class Support Vector Machine Classifier. The module has been evaluated on a simple audio event recognition task. In addition, contextual information has been used within an autonomous task execution scenario. The particular package is deprecated as it only functions with ROS Diamondback and below with Ubuntu 12.04 and below.

In [3] a framework for detecting, classifying and recognising novel non-verbal sounds on an Aldebaran Nao humanoid robot has been proposed. New sounds are entered in the training process through a speech-based interaction that is applied between the robot and the user. MFCCs are used as features, while a simple distance-based and thresholding method has been adopted as classifier. The implemented approach cannot be executed online and realtime for events larger than one second on a 1.6gHz CPU of the Nao robot.

In [9], everyday sound events are recognized in the context of a home environment using a consumer robot (NAO). Audio information is represented through the stationary auditory image method (SAI). The method has been evaluated

PETRA '16, June 29-July 01, 2016, Corfu Island, Greece
 ISBN 978-1-4503-4337-4/16/06...$15.00
DOI: http://dx.doi.org/10.1145/2910674.2935858

with an annotated dataset of audio information recorded with a humanoid robot in a house environment, using 12 audio classes.

In this work, we propose a workflow for audio signal classification that utilizes the ROS architecture by ensuring that the most computationally burdensome stages (the audio feature extraction in particular) is placed in series to the audio acquisition stage. In this way, the joint audio acquisition-feature extraction module is capable of streaming signal representations instead of raw audio samples. Such information is then used by all segmentation and/or classification ROS modules ensuring reuse of the audio representations without the need of recalculation. Implementation details, computational costs along with an experimental evaluation on particular use cases are provided in this paper.

2. PROPOSED ARCHITECTURE

The proposed conceptual architecture is presented in 1. Two main components (nodes) are implemented in the context of the core of the proposed architecture: (a) *Audio Acquisition and Feature Extraction Node (AAFE)*: this node is responsible for recording the audio data from the soundcard and extracting the adopted audio features in an online mode. AAFE node published the audio features along with temporal information to the respective topic. (b) *Audio Segmentation & Classification Node (ASC)*: the ASC node subscribes to the topic generated by AAFE and generates segmentation and classification decisions based on the input audio features. The extracted class labels per audio segment are published in a new topic (labels topic).

Both nodes broadcast (a feature or a class label message respectively) with a resolution equal to the short-term frame, which is equal to 50 mseconds as a default value (see next Section). Any (ongoing and future) modules, e.g. Speaker Identification and Speech Emotion Recognition, will subscribe to both topics in order to obtain access to feature representations along with class information. In this work, we provide a full description of the core audio analysis ROS functionality, i.e. the AAFE and ASC nodes. Audio acquisition is described in subsection 3.2 while the feature extraction submodule in subsection 3.2. ASC is described in Section 4.

3. AUDIO ACQUISITION AND FEATURE EXTRACTION

3.1 Audio Acquisition

All audio input/output operations are handled by the PortAudio library [1]. This is a highly-regarded audio library, widely used in both open-source and commercial applications. It is implemented in C (with C++ and Python bindings) and it is supported on all major operating systems. A unique feature of the library is the support for *blocking audio input/output*. This attribute can be very useful in the project's setting since having a contiguous audio block for processing without having real-time constraints is more important than playback or recordings and thus it is allowed to drop frames if necessary.

3.2 Feature Extraction

3.2.1 Short and Mid-term feature extraction

In this Section we describe the audio features extracted in order to lead to an informative representation with respect to the desired properties of the original audio data [6, 8, 12, 4]. The signal is first divided into short-term non-overlapping windows (frames) and a set of features is computed per frame. This processing stage generates a sequence, $\mathbf{F}$, of feature vectors per audio signal.

Another common technique in audio analysis is the processing of the feature sequence on a mid-term basis. According to this step, the audio signal is also divided into mid-term segments (windows) and then, *for each segment*, the short-term processing stage is carried out. Then, the feature sequence, $\mathbf{F}$, which has been extracted from a mid-term segment, is used for computing feature *statistics*, e.g., the average value of the zero-crossing rate. After this two-stage process (mid-term and short-term analysis), *each mid-term segment is represented by a set of statistics* which correspond to the respective short-term feature sequences.

3.2.2 Audio Features

This section describes the adopted audio features extracted per short-term frame. Before proceeding, let $x_i(n), n = 1, \ldots, W_L$ be the sequence of audio samples of the $i-$th frame (i.e. short-term window), where W_L is the length of the frame in number of samples. The following time-domain features are extracted:

- Short-term energy: $E(i) = \frac{1}{W_L} \sum_{n=1}^{W_L} |x_i(n)|^2$
- Zero Crossing Rate (ZCR): $Z(i) = \frac{1}{2W_L} \sum_{n=1}^{W_L} \mid sgn[x_i(n)] - sgn[x_i(n-1)] \mid$, where $sgn(\cdot)$ is the sign function.
- Entropy of Energy: short-term frame is divided in K sub-frames of fixed duration. Then, for each sub-frame, j, we compute its energy and normalize it by the total frame energy ($E_{shortFrame_i}$): $e_j = \frac{E_{subFrame_j}}{E_{shortFrame_i}}, j = 1, \ldots, K$. Next, the entropy, $H(i)$ of the sequence e_j is computed according to the equation: $H(i) = -\sum_{j=1}^{K} e_j^2 \cdot log_2(e_j^2)$.

Additionally to the aforementioned time-domain features, we extract a range of spectral features, which are based on the Discrete Fourier Transform (DFT), which provides a representation of the distribution of the frequency content of sounds (audio spectrum). Such features are called frequency-domain or spectral audio features. In order to proceed, let $X_i(k)$, $k = 1 \ldots, Wf_L$, be the magnitude of the DFT coefficients of the $i-th$ audio frame, where $Wf_L = W_L/2$ is the number of spectral coefficients. The following spectral-domain features are extracted:

- Spectral centroid and the spread: $C_i = \frac{\sum_{k=1}^{Wf_L}(k+1)X_i(k)}{\sum_{k=1}^{Wf_L} X_i(k)}$ and $S_i = \sqrt{\frac{\sum_{k=1}^{Wf_L}((k+1)-C_i)^2 X_i(k)}{\sum_{k=1}^{Wf_L} X_i(k)}}$. Both features are normalized in the range $[0, 1]$, by dividing their values by $\frac{Fs}{2}$.
- Spectral Entropy: A spectral counterpart of the entropy of energy is the feature of the spectral entropy.
- Spectral flux: $Fl_{(i,i-1)} = \sum_{k=1}^{Wf_L}(EN_i(k) - EN_{i-1}(k))^2$, where $EN_i(k) = \frac{X_i(k)}{\sum_{l=1}^{Wf_L} X_i(l)}$, i.e., $EN_i(k)$ is the k-th normalized DFT coefficient at the i-th frame.

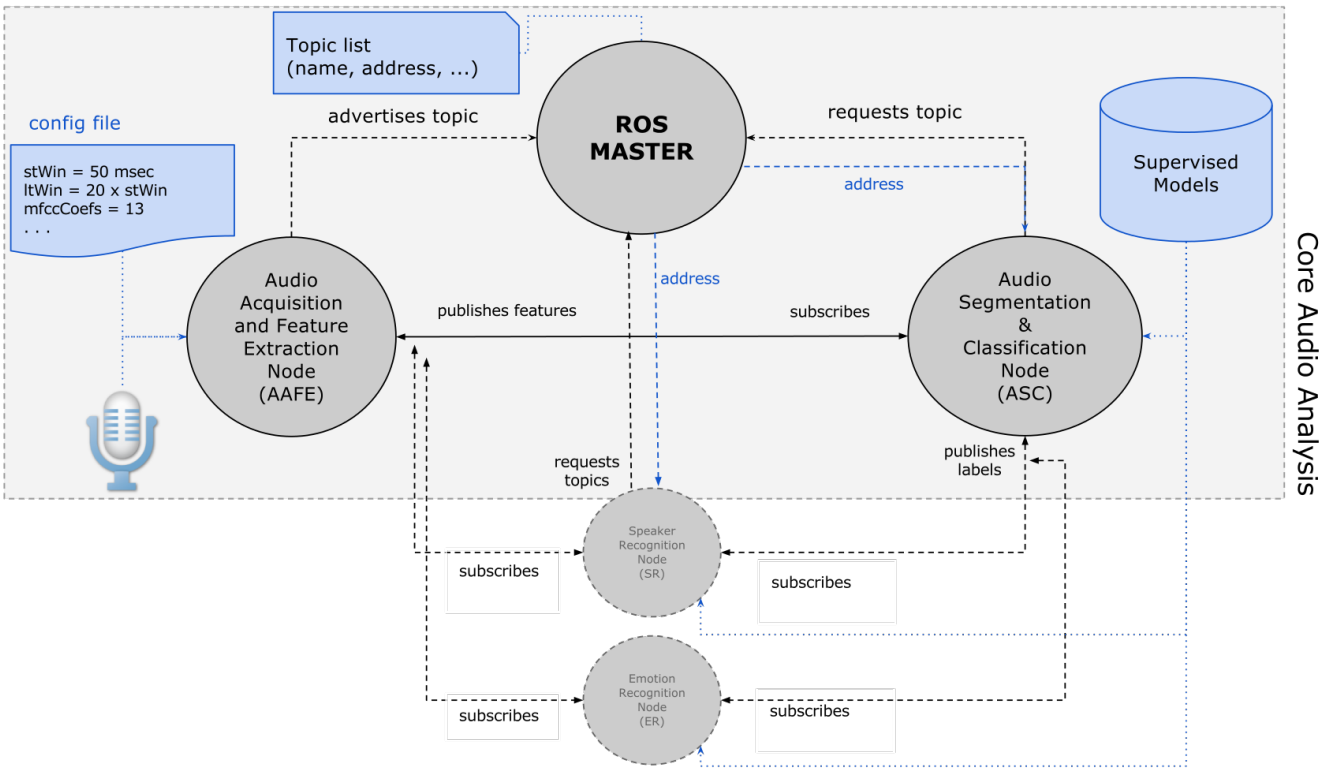

Figure 1: Conceptual architecture of the proposed ROS workflow. The core functionality is provided by the Audio Acquisition and Feature Extraction Node and the Audio Segmentation & Classification Node. Future and ongoing modules (e.g. speaker recognition and emotion recognition) will subscribe to topics stemming from both nodes. The ROS Master provides naming and registration services to the rest of the nodes in the ROS system and it is responsible for tracking publishers and subscribers to topics and services.

- Spectral Rolloff: the frequency below which a certain percentage (e.g 90%) of the magnitude distribution of the spectrum is concentrated. If the m-th DFT coefficient corresponds to the the spectral rolloff of the i-th frame, then it satisfies the equation $\sum_{k=1}^{m} X_i(k) = C \sum_{k=1}^{Wf_L} X_i(k)$, where C is the adopted percentage.
- The Mel-Frequency Cepstrum Coefficients (MFCCs): a cepstral representation of the signal, where the frequency bands are distributed according to the mel-scale. To compute, the DFT is extracted and provided as input to a mel-scale filter bank. The MFCCs are computed as the discrete cosine transform coefficients of the aforementioned mel-scale log-power spectrum.

More information and explaination of the adopted audio features can be found in [12], [8], [6], [5].

3.2.3 *Implementation Issues*

The 21, in total, audio features described in Section 3.2.2 are extracted for each short-term frame. Then, as explained above, a mid-term analysis is adopted, by computing the mean and standard deviation statistics per 1-second mid-term segment. This leads to a $2x21 = 42$ feature statistic representation per 1-second audio segment. In the context of the ROS architecture (as described in Section 2) the following attributes are broadcasted, *per short-term frame*, in the respective topic:

- *timestamp*: the workstation's clock. Used by the communicating topics for synchronization.
- *current short-term feature vector*: the 21 features that represent the current short-term frame
- *current mid-term feature vector*: the 42 feature statistics that represent the last 1-second segment. Since the time resolution of the topic is equal to the short-term frame (50 mseconds), the amount of overlap is large. The statistics are computed in an incremental manner to reduce computational complexity.

The whole feature extraction submodule is implemented in C++. In addition, the FFTW library (http://www.fftw.org/) is adopted for computing the DFT and the Eigen C++ library (http://eigen.tuxfamily.org/) for numerical procedures and linear algebra tasks.

4. AUDIO SEGMENTATION AND CLASSIFICATION

A separate ROS node implements the classification of each mid-term segment, given the respective feature statistics, to a set of predefined audio classes. Towards this end a Python ROS node has been implemented based on the pyAudioAnalysis open-source library [5]. This node:

- subscribes to the features topic: based on the input mid-term audio feature representations it classifies the respective audio segment to a set of predefined classes. Support Vector Machines are used as classifiers

- performs joint segmentation, based on the extracted audio class labels. This is achieved in two ways: either by successive segments merging or by HMM-based smoothing of the soft outputs of the SVM classifier.

- trains the SVM classifier: towards this end, we have implemented a wrapper of the training procedure using cross-validation for tuning the SVM parameters C.

5. USE CASES AND EVALUATION

The motivating use cases for our approach are defined in the context of medical monitoring. Specifically, we base our evaluation setup on allowing elderly people with mild cognitive impairment to maintain an independent life, at their own home, for longer than what is safely possible today. In order to have a guideline about what information is used by medical doctors to assess such conditions, we use the *interRAI Long-Term Care Facilities Assessment System (interRAI LTCF)*, which enables comprehensive, standardized evaluation of the needs, strengths, and preferences of persons receiving care. interRAI has been analysed previously in order to identify assessment items, such as mood and ADL logs, that can be automatically recognized and are useful to medical personnel.

Among the assessment items defined in the context of the interRAI process, we identified the following use cases:

- bathroom scenario: these items can be extracted by processing very sensitive content. The adopted audio classes are: Silence - no sound, Flushing water, Shower, Tap water, Other activities

- general scenario: this task covers a wide range of audio content possible to appear in the rest of the house. The adopted classes are: silence, speech, music, phone ring, coughing, crying, yelling, laughing, objects-activity

Cross validation has been applied to evaluate the performance of the segmentation-classification task on both scenarios. Towards this end, two datasets of total duration of 45 minutes have been compiled. The resulting F1 measure for the 5-class scenario (bathroom) was found to be equal to 89%, while for the 9-class scenario (general) the F1 measure was 81%

6. CONCLUSIONS AND FUTURE WORK

We have presented an audio-based activity recognition workflow that is based on the ROS architectural principles. The proposed architecture tries to ensure that the most computationally demanding modules of the audio analysis workflow are merged to the audio acquisition stage. In other words, we propose using a ROS node that streams audio feature representations (both mid-term and short-term) instead of raw audio information. This representations are used by all segmentation and/or classification ROS nodes ensuring reuse of the audio representations without the need of recalculation or the transmition of bandwidth demanding raw information. We provided implementation details, along with an experimental evaluation on particular use cases from the health monitoring domain.

It has to be noted that the proposed workflow is provided as an open-source library at github (https://github.com/tyiannak/AUROS). Ongoing work is conducted to extend the classification/segmentation node with more advanced audio analytics (emotion recognition and speaker diarization in particular).

7. ACKNOWLEDGMENTS

This paper is supported by the project "Robots in assisted living environments: Unobtrusive, efficient, reliable and modular solutions for independent ageing - RADIO", which has received funding from the EU's Horizon 2020 research and innovation programme under grant agreement No 643892. More details in http://www.radio-project.eu

8. REFERENCES

[1] R. Bencina and P. Burk. Portaudio–an open source cross platform audio api. In *Proc. 2001 Intl. Computer Music Conf.(ICMC-01)*, 2001.

[2] J. Benesty, M. M. Sondhi, and Y. Huang. *Springer handbook of speech processing*. Springer Science & Business Media, 2007.

[3] T. DâĂŹArcy, C. Stanton, and A. Bogdanovych. Teaching a robot to hear: a real-time on-board sound classification system for a humanoid robot. In *Proceedings of Australasian Conference on Robotics and Automation*, 2013.

[4] T. Giannakopoulos. *Study and application of acoustic information for the detection of harmful content, and fusion with visual information*. PhD thesis, Dpt of Informatics and Telecommunications, University of Athens, Greece, 2009.

[5] T. Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PloS one*, 10(12):e0144610, 2015.

[6] T. Giannakopoulos and A. Pikrakis. *Introduction to Audio Analysis: A MATLAB® Approach*. Academic Press, 2014.

[7] H. Hermansky. Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[8] K. Hyoung-Gook, M. Nicolas, and T. Sikora. *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. John Wiley & Sons, 2005.

[9] M. Janvier, X. Alameda-Pineda, L. Girinz, and R. Horaud. Sound-event recognition with a companion humanoid. In *Humanoid Robots (Humanoids), 12th IEEE-RAS International Conference on*, pages 104–111, 2012.

[10] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[11] J. M. Romano, J. P. Brindza, and K. J. Kuchenbecker. Ros open-source audio recognizer: Roar environmental sound detection tools for robot programming. *Autonomous Robots*, 34(3):207–215, 2013.

[12] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, 2008.