Design for a System of Multimodal Interconnected ADL Recognition Services

Theodoros Giannakopoulos, Stasinos Konstantopoulos, Georgios Siantikos and Vangelis Karkaletsis

Abstract As smart interconnected sensing devices are becoming increasingly ubiquitous, more applications are becoming possible by re-arranging and re-connecting sensing and sensor signal analysis in different pipelines. Naturally, this is best facilitated by extremely thin services that expose minimal functionality and are extremely flexible regarding the ways in which they can be re-arranged. On the other hand, this ability to re-use might be purely theoretical since there are established patterns in they ways processing pipelines are assembled. By adding privacy and technical requirements the re-usability of some functionalities is further restricted, making it even harder to justify the communication and security overheads of maintaining them as independent services. This creates a design space that each application must explore using its own requirements. In this article we focus on detecting Activities of Daily Life (ADL) for medical applications and especially independent living applications, but our setting also offers itself to sharing devices with home automation and home security applications. By studying the methods and pipelines that dominate the audio and visual analysis literature, we observe that there are several multi-component sub-systems can be encapsulated by a single service without substantial loss of re-usability. We then use this observation to propose a design for our ADL recognition application that satisfies our medical and privacy requirements,

Theodoros Giannakopoulos
Institute of Informatics and Telecommunications, NCSR 'Demokritos', Ag. Paraskevi 15310, Greece, e-mail: tyianak@iit.demokritos.gr

Stasinos Konstantopoulos
Institute of Informatics and Telecommunications, NCSR 'Demokritos', Ag. Paraskevi 15310, Greece, e-mail: konstant@iit.demokritos.gr

Georgios Siantikos
Institute of Informatics and Telecommunications, NCSR 'Demokritos', Ag. Paraskevi 15310, Greece, e-mail: siantikosg@gmail.com

Vangelis Karkaletsis
Institute of Informatics and Telecommunications, NCSR 'Demokritos', Ag. Paraskevi 15310, Greece, e-mail: vangelis@iit.demokritos.gr

makes effient use of processing and transmission resources, and is also consistent with home automation and home security extensions.

Key words: audio-visual analysis, activities of daily life, internet of things, connected sensing services

1 Introduction

As smart, interconnected sensing devices are becoming ubiquitous, more applications are becoming possible by re-arranging and re-connecting sensing and sensor signal analysis in different pipelines. This appears to imply that the finer the grain of the distinct and referenceable services the better, since finer grain allows maximal flexibility; in reality however, there are composite services that make more sense than their components as the finest grain that is exposed. The reasons vary, ranging from technical, to privacy, to business considerations, but the effect is that systems of services need to be properly designed to assume a satisfactory position between being too fine and too coarsely grained.

We shall focus here on detecting *Activities of Daily Life (ADL)* in a smart home setting. Such a setting offers itself naturally to medical applications and especially independent living applications, where ADL logs can establish patterns and identify deviations. These can range from the time spent out of the house or carrying out a given activity in it, to sleeping patterns, to recognizing whether the user has changed clothes, washed, or other crucial indications required by the medical condition that necessitates monitoring.

To motivate our work, let us assume as an example a general-purpose acoustic feature extraction component that can provide input for different classification engines and models. For our application we have established that we will detect acoustic events using an SVN engine and that there is no further component that needs these features. In this situation, it makes sense to encapsulate the audio acquisition component, the feature extraction component, and the SVN engine under a single acoustic event detection service and not provide an interface for the acoustic features or, even more so, for the audio signal. We have based our decision on the observation that we re-analyze the lower-level feature extraction output into a higher-level acoustic event output that is more informative and more straightforward to be consumed by other services.

Let us now assume that we later decide to apply a fusion algorithm that combines audio and vision. The extracted audio-visual events represent the same real-world events as the acoustic and visual events that they fuses, except that they are associated with a higher confidence and/or carry more attributes than either. In such a situation our previous decision to bundle acoustic event extraction as a single service and not as an extraction-classification pipeline restricts us to event-level fusion and makes it impossible to apply feature-level fusion algorithms. On the other hand, if we were to use event-level fusion anyway, the finer-grained feature-level services would be unnessessary overheads.

Naturally, real-world scenarios are bound to be even more complex. To give another example, consider a system for our ADL monitoring application that comprises the following components:

- Waking up and getting out of bed is recognized in the depth modality
- Moving around the house is tracked in the depth modality
- Moving around the house is tracked and the person moving is identified in the image modality

In this scenario, subsequent fused image/depth analysis both confirms and adds attributes (the identity of the person) to the original getting-out-of-bed event. A system that measures the time it takes to transfer out of bed will need both the onset of the first event and the subsequent supporting information. However, this system of analysis components cannot be tried under a single service that log the *bed transfer* ADL, since movement tracking will also be useful for logging numerous other activities.

Further applications can also be envisaged, such as proactively offering automations that are relevant to the current ADL context. The design of this application as an extension of a medical monitoring application is straight-forward, since it is based on analysing high-level ADL logs. Other applications, however, might require more concrete data and, thus, more flexibility regarding the components of the architecture that they will need to access. In general, the choice of how to best wrap the analysis components and pipelines of such components balances between microservices that expose thin slices of functionality and heavier services that wrap larger sub-systems. Naturally, there can be no ideal solution and each application must use its own requirements to explore the design space.

ADL recognition is based on the recognition of events in (possibly multiple) audiovisual signals and on heuristics that characterize sequences or other compositions of events as more abstract ADL events.

In this article, we first present the audio (Section 2) and visual (Section 3) sensors and corresponding recognition methods that are typically used in our application domain. By studying the components that make up these methods and the kinds of information exchanges between them, we propose a conceptual architecture (Section 4). The purpose of our architecture is to integrate existing components developed in different contexts (rootics and the internet of things) and to make effient use of processing and transmission resources. The key design points that this conceptual architecture deals with is establishing appropriate articulation points for segmenting into components the pipelines commonly proposed in the multimodal processing literature and specifying the kind of information that is exchanged at the interfaces between these components.

2 Recognizing Events in Audio Content

Acoustic analysis pipelines include signal aquisition, acoustic feature extraction, and classification of the features into acoustic events. In particular, the following components are typically included in acoustic analysis pipelines:

- audio acquisition: uses the audio signal to produce a stream of short-term audio frames
- short-term feature extraction: uses an audio frame to produce a frame feature vector
- mid-term feature extraction: aggregates multiple frame feature vectors into a mid-term segment feature vector
- audio pattern analysis: uses frame and segment feature vectors to produce an event recognition

In this section we will present these components and related design choices based on our ADL recognition application.

2.1 Feature Extraction

Audio acquisition is based on the cross-platform, open-source PortAudio library [1]. Audio acquisition divides the audio signal into a stream of short-term windows (frames), and short-term features are extracted in an on-line mode from each frame.

Short-term feature extraction produces a stream of feature vectors of 34 elements (Table 1), where each vector is calculated from one frame. The time-domain features (features 1–3) are directly extracted from the raw signal samples. The frequency-domain features (features 4–34, apart from the MFCCs) are based on the magnitude of the Discrete Fourier Transform (DFT). Finally, the cepstral domain (e.g. used by the MFCCs) results after applying the Inverse DFT on the logarithmic spectrum.

These features and the effect of short-term windowing in audio classification have been discussed in more detail by Kim et al. [6] and Giannakopoulos et al. [2, 4]. What is important to note is that the frame size and the selected features depend on each other, so that a *different frame stream, using different frame size, would need to be produced for a different short-term feature extraction component.*

This observation suggests a very strong coupling between audio acquisition and short-term feature extraction. Furthermore, short-term feature extaction results in a drastically smaller stream than the original audio signal, so that there this also a communications overhead advantage in only providing the short-term feature stream as a service without exposing the audio signal.

2.2 Feature Aggregation and Analysis

Another common technique in audio analysis is the processing of the feature sequence on a mid-term basis, according to which the audio signal is first divided into mid-term windows (segments). For each segment, *mid-term feature extraction* uses the short-term feature vector stream to produce a stream of feature statistics, such as the average value of the ZCR (Feature 1). Therefore, each mid-term segment is represented by a set of statistics.

The final stage is the analysis of patterns in the short and mid-term features to infer event annotations. Two types of *audio pattern analysis* are performed:

- *Supervised:* A set of predefined classifiers is trained and used to extract respective labels regarding events [2, 12]. Apart from general audio events regarding activities (ADLs) mood extraction is achieved by applying regression and classification methods trained on speech-based emotion recognition data.
- *Unsupervised:* Apart from predefined taxonomies of audio events and activities, audio features are used in the context of a clustering procedure, according to which the extracted labels are not known a-priori. A typical example is *speaker diarization* or *speaker clustering*, the task of determining who spoke when [3].

For both types for audio pattern analysis, a decision is made for each short-term feature vector takig into account a combined short-term and mid-term feature vector. In a sense, the mid-term features provide a context within with the short-term features are interpreted. The singal energy, for example, is more informative when

Table 1 Audio Features

Index	Name	Description
1	Zero Crossing Rate	The rate of sign-changes of the signal during the duration of a particular frame.
2	Energy	The sum of squares of the signal values, normalized by the respective frame length.
3	Entropy of Energy	The entropy of sub-frames' normalized energies. It can be interpreted as a measure of abrupt changes.
4	Spectral Centroid	The center of gravity of the spectrum.
5	Spectral Spread	The second central moment of the spectrum.
6	Spectral Entropy	Entropy of the normalized spectral energies for a set of sub-frames.
7	Spectral Flux	The squared difference between the normalized magnitudes of the spectra of the two successive frames.
8	Spectral Rolloff	The frequency below which 90% of the magnitude distribution of the spectrum is concentrated.
9-21	MFCCs	Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
22-33	Chroma Vector	A 12-element representation of the spectral energy where the bins represent the 12 equal-tempered pitch classes of western-type music (semitone spacing).
34	Chroma Deviation	The standard deviation of the 12 chroma coefficients.

combined with the average energy in order to detect clangs and bangs even in overall noisy environments as sudden spikes above the average.

Typical values of the mid-term segment size can be from 200 msec to several seconds depending on the events that are being recognized. Segments can be overlapping, using a different sliding mid-term feature vector with each short-text feature vector, or non-overlapping, using the same mid-term feature vector throughout the duration of the segment.

Depending on the technical characteristics of the middleware used, some communication overheads can be reduced by keeping the same mid-term feature vector latched to the bus so that it can be read multiple times by the audio pattern analysis component. This potential gain is, however, relatively small as the information to be communicated has been reduced to 34 floating-point features, i.e. 136 bytes. Furthermore, different event recognizers often work better with different segment sizes, so that multiple (typically two or three) mid-term feature vectors need to be made available to audio pattern analysis. These observations suggest that there is little optimization value in non-overlapping segments, and that the (more accurate) rolling segments should be preferred.

This still leaves open the question of whether mid-term feature vectors should be bundled together with the acquisition/short-term feature extraction components as a complete audio feature extraction service, bundled together with the audio pattern analysis components, or provided as an independent service or services (for different segment sizes).

3 Recognizing Events in Visual Content

Visual information is recorded through a depth camera, therefore two types of information are adopted in this context: colour video and depth video. Both types of visual information are fed as input to a workflow of visual analytics whose purpose is to extract:

- activities of daily living (ADLs): a predefined set of events related to ADLs
- measures related to the user's ability to perform these ADLs

3.1 Preprocessing

In our application area, stereoscopic or structured light visual sensors are used to provide both colour and 3D depth information channels. Both channels are preprocessed in order to provide better representations of the scene. Colour is tranformed and normalized in order to achieve colour consistency and lighting conditions independence. In addition, de-noising and hole removal is applied on each depth frame in order to provide the next steps with smoother depth representations.

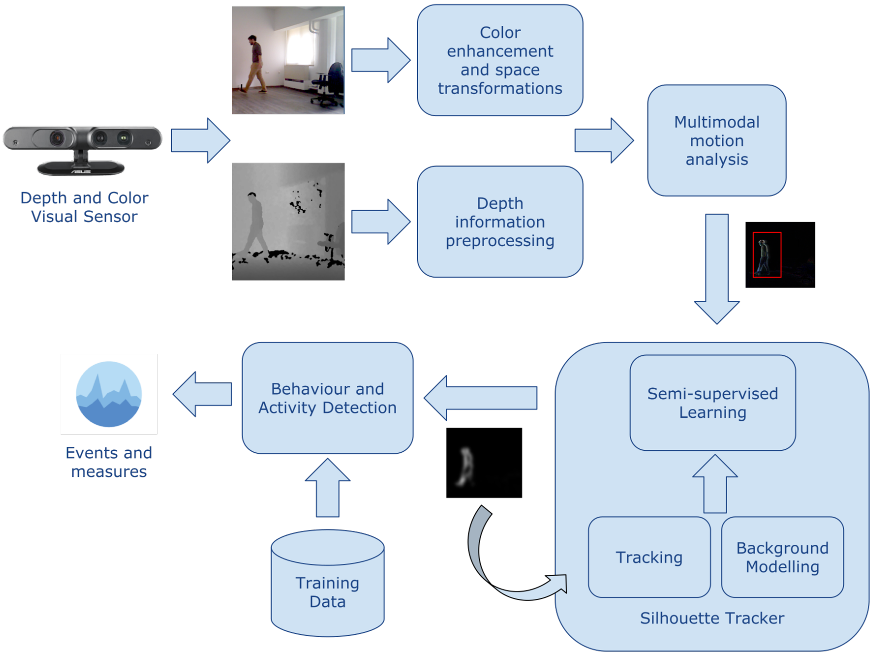
Color enhancement and space transformations
Depth and Color Visual Sensor
Depth information preprocessing
Multimodal motion analysis
Semi-supervised Learning
Behaviour and Activity Detection
Events and measures
Tracking
Background Modelling
Silhouette Tracker
Training Data

Fig. 1 Depth and colour visual analytics workflow

3.2 *Motion Detection*

This submodule aims to function as a triggering functionality for initializing the visual analytics workflow but also as a baseline estimator of the user's spatial information. The adoptation of motion detection as a separate submodule is twofold: (a) to enchance the performance of the next visual analytics steps (e.g. make the bacgkround estimation more robust) and (b) to minimize the computational complexity of the whole visual workflow by triggerning the respective submodules only when necessary.

Motion detection is capable of checking the existence of motion withing the visual information. For each processing time unit (i.e. video frame), a single thresholding rule is adopted for extra low computational complexity, in order to infer the existence of motion. If no motion is detected, then the background model is updated as described in the next paragraph. For better performance, separate channels are used in order to estimate the motion of each frame: depth-based distance metrics along with three different colour-space distance metrics. A simple weighted average fusion approach is used to generate the final 2-dimensional distance metric that is used in the thresholding criterion that detects the motion area. The result of this procedure is a bounding box of the motion area.

Before combining the different modalities it is improtant to note that *visual registration* is applied, in order to align the different representations of the same scene into one common spatial representation. Especially between the depth and colour channels it is very important to apply such geometric transformations, since the two channels are heavily misalligned on most sensors.

3.3 Silhouette Tracking

The goal of this submodule is to detect and track the exact positions of the pixels associated to the person's silhouette. Towards this end the following algorithmic stages are adopted:

- *Background modelling* The goal here is to estimate a statisticall model that describes the background of the visual information, so that it is substracted in the main visual analysis steps. The background subtraction submodule is triggered from the motion detection service as described in the previous paragraph. As soon as the module is triggered, the following steps are executed:
  - the (detected as "motion") frames are captured to memory
  - contrast and light normalization is applied to remove unwanted dependencies
  - gamma correction is applied
  - the background model is extracted using the MOG operator [5]
  - the outline of foreground objects is finally estimated using a set of morphological operations [10]
- *Tracking* The goal of this step is to model the moving object's dynamics in order to track the exact position of the user.
- *Semisupervised Learning* At each stage a clustering algorithm is applied based on the current estimated background, the feedback from the tracking algorithm as well as the raw depth info of the current frame, in order to decide on the final silhouette estimate.

3.4 Behaviour and Activity Detection

Given the position and exact shape of the user's silhouette, a series of supervised and unsupervised machine learning approaches is applied in order to:

- extract a set of predefined activities (stand up, sitting, lying in bed, walking, running, eating, etc). Towards this end, annotated data are used to train the respective classifiers.
- extract body keypoints using supervised models

- detect faces and extract facial features. Also, if provided, the supervised database stores facial features of known users and the respective module also extract user ID (identification).
- extract clothing-related information (i.e. if the user has changed clothes since her last appearence on the sensor) [9]
- estimate metrics related to the user's ability to walk. Towards this end, unsupervised temporal modelling is adopted as the means to extract measures that quantify the gait: average speed, time required to walk four meters, etc.

3.5 Fused Audio-Visual Analysis

Apart from the audio and visual workflows that extract respective high-level information and metadata regarding activities and measures, the *early* and *late fusion* approaches are used to extract information from the combined audio and visual modalities.

On example is combining facial features (cf. Section 3.4) with acoustic features (cf. Section 2.2) in the context of a speaker diarization method that extracts user labels based on speech and facial cues [8]. In this *early fusion* example, the acoustic and visual feature vectors are fused. By constrast, *late fusion* approaches as used for behaviour recognition. In particular, multimodal events can be reconized by combining events in each modality that represent the same physical event.

4 ADL Recognition Architecture

Based on our analysis of the ADL recognition methods above (Sections 2 and 3), we will now proceed to bundle the relevant components into services and to specify these services' information outputs and requirements. These services will be used as the building blocks of our ADL recognition architecture in this section.

Our usage scenario is set in an assisted living environment with static sensors and a mobile robot which acts as a mobile sensing platform. In this setting, the clinical requirement is to monitor activities of daily life in order to report aggregated logs about the occurence of specific ADLs, signs of physical activity, as well as performance measurements such as time needed to get off the bed or to walk a given distance [7].

In our design we foresee acoustic sensors that integrate a microphone with Raspberry Pi, a mobile TurtleBot2 robot[1] that integrates microphone, Xtion depth and colour camera and on-board computer, and a main computer that acts as the gateway to the home and the orchestrator of the overall monitoring and reporting. This

[1] Please cf. `http://www.turtlebot.com` for more details.

physical infrastructure is used to deploy the sensing services and the ADL recognition services that use them.

There is a single acoustic features interface which publishes a stream of triplets of feature vectors. Each message in the stream contains the current short-term frame feature vector and two mid-term rolling averages of different numbers of frames, to accommodate analyses that require deeper or more shallow acoustic contexts.

This interface was chosen because at our 50 Hz frame rate volume of traffic generated by three floating-point feature vectors is insignificant and this interface lifts the requirement to have a middleware that can latch mid-term feature vectors or synchronize mid-term and short-term feature vectors. Exposing the complete acquisition-feature extraction pipeline as a single service also allows us to provide a unified acoustic feature service over two heterogeneous implementations [11]:

- The TurtleBot2 implementation comprises a microphone device driver and a feature extraction component that communicate using the ROS middleware.[2] The service end-point is a bridge that simultaneously connects to the robot-internal ROS middleware and to the home WiFi to access robot-external services.
- The Raspberry implementation comprises a microphone device driver and a feature extraction component that communicate using MQTT.[3] The service end-point is a bridge that simultaneously connects to MQTT and to the home WiFi to access external services.

All instances of the acoustic features service push their vector streams to the audio pattern analysis service. This service implements unsupervised and (previously trained) supervised machine learing methods that recognize ADL events from acoustic feature vectors. The audio pattern analysis service is also distributed, with instances executing at the Raspberry and the robot's computer.

The vision sensing components are analogously implemented as image acquisition, feature extraction, and pattern recognition services. One divergene from the acoustic analysis case is that the graph of dependencies between vision services is not a linear progression from the content to more abstract features and events: motion detection is the only service that constantly consumes features and it triggers more complex analyses as soon as motion is detected. Furthermore, there is no single feature set that is used by all visual analyses and analyses are occasionally stacked more deeply that the features/events/ADLs layers of acoustic ADL detection.

The services and interfaces design described here is also depicted in Figure 2.

[2] The Robot Operating System (ROS) is a set of software libraries and tools for developing distributed applications in robotics; please cf. `http://www.ros.org` for more details.

[3] MQTT is an extremely lightweight publish/subscribe messaging middleware for the Internet of Things; please cf. `http://mqtt.org` for more details.

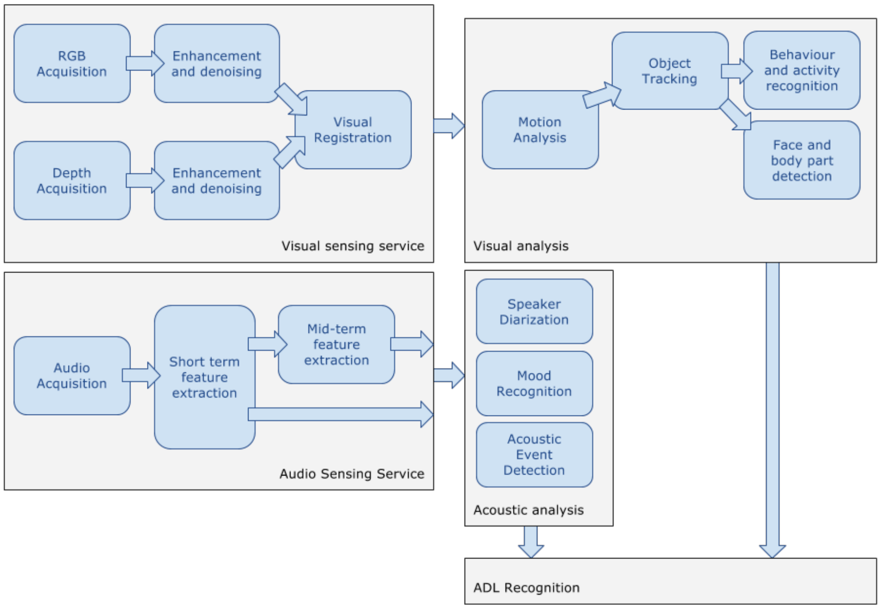

Fig. 2 Conceptual architecture of audio-visual analysis

5 Conclusion

We presented a system of services that interact to recognize ADLs from audio-visual sensors. Our design integrates sub-systems which were originally integrated using heterogeneous middleware infrastructures. We have proposed articulation points for re-structuring these existing pipelines into a new set of services. In order to establish the right level of granularity for the functionality bundled under a single service, we used common patterns in the audio-visual analysis literature to identify services that would practically never need to be broken down into finer services.

The most prominent future research direction is the dynamic handling of early fusion methods. For such methods, the recognition component must have access to both the acoustic and the visual features. In our current design, this is not as issue as in the only situation where this is necessary (scenes captures by the robot) both sensors happen to be on the same ROS middleware and the audio-visual features can be easily consumed by the same component. As this will not be the case in general, our plan is to transfer concepts and technologies from the distributed processing literature. Although developed to address different problems (namely, processing large scale data), transferring such technologies to our application will allow us to develop distributed fusion components. In this manner, an early fusion component can perform calculations over distributed feature vectors without requiring that they are collected at a single computation node, applying the communication overhead optimizations developed by the distributed computation community to minimize the communication of intermediate results.

Another future research direction pertains to incorporating speech recognition in the system. Speech recognition uses radically different features than those computed for acoustic processing. Naturally, one can always concatenate or interleave the feature vectors produced for acoustic and speech processors in order to accommodate both. It, however, worth investigating whether a study of the feature extraction methods proposed in the speech recognition literature can reveal opportunities for a more efficient integration of the acoustic and speech feature extraction components.

Acknowledgements This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 643892. For more details, please visit the RADIO Web site `http://www.radio-project.eu`

References

1. Bencina, R., Burk, P.: PortAudio–an open source cross platform audio API. In: Proceedings of the International Computer Music Conference, Havana, pp. 263–266 (2001)
2. Giannakopoulos, T.: pyAudioAnalysis: An open-source Python library for audio signal analysis. PloS One **10**(12) (2015)
3. Giannakopoulos, T., Petridis, S.: Fisher linear semi-discriminant analysis for speaker diarization. IEEE Transactions On Audio, Speech, And Language Processing **20**(7) (2012)
4. Giannakopoulos, T., Pikrakis, A.: Introduction to Audio Analysis: A MATLAB® Approach. Academic Press (2014)
5. KaewTraKulPong, P., Bowden, R.: An improved adaptive background mixture model for real-time tracking with shadow detection. In: Video-Based Surveillance Systems, pp. 135–144. Springer (2002)
6. Kim, H.G., Moreau, N., Sikora, T.: MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval. John Wiley & Sons (2006)
7. RADIO Project: Deliverable 2.2: Early detection methods and relevant system requirements. Tech. rep. (2015). URL `http://radio-project.eu/deliverables`
8. Sarafianos, N., Giannakopoulos, T., Petridis, S.: Audio-visual speaker diarization using Fisher linear semi-discriminant analysis. Multimedia Tools and Applications pp. 1–16 (2014)
9. Sgouropoulos, D., Giannakopoulos, T., Siantikos, G., Spyrou, E., Perantonis, S.: Detection of clothes change fusing color, texture, edge and depth information. In: E-Business and Telecommunications, pp. 383–392. Springer (2014)
10. Sgouropoulos, D., Spyrou, E., Siantikos, G., Giannakopoulos, T.: Counting and tracking people in a smart room: An IoT approach. In: Proceedings of the 10th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP 2015). IEEE (2015)
11. Siantikos, G., Giannakopoulos, T., Konstantopoulos, S.: A low-cost approach for detecting activities of daily living using audio information: A use case on bathroom activity monitoring. In: Proceedings of the 2nd International Conference on Information and Communication Technologies for Ageing Well and e-Health (ICT4AWE 2016), Rome, Italy (2016)
12. Siantikos, G., Sgouropoulos, D., Giannakopoulos, T., Spyrou, E.: Fusing multiple audio sensors for acoustic event detection. In: Proceedings of the 9th International Symposium on Image and Signal Processing and Analysis (ISPA 2015), pp. 265–269. IEEE (2015)