Smart sensor integration: a framework for multimodal emotion recognition in real-time

Johannes Wagner, Elisabeth André, Frank Jung

Angaben zur Veröffentlichung / Publication details:

Wagner, Johannes, Elisabeth André, and Frank Jung. 2009. "Smart sensor integration: a framework for multimodal emotion recognition in real-time." In *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops, 10-12 September 2009, Amsterdam, Netherlands*, edited by John Cohn, Anton Nijholt, and Maja Pantic, 1–8. Los Alamitos, CA: IEEE. https://doi.org/10.1109/ACII.2009.5349571.

Smart Sensor Integration: A Framework for Multimodal Emotion Recognition in Real-Time

Johannes Wagner, Elisabeth André, Frank Jung
Multimedia Concepts and Applications, University of Augsburg
Eichleitnerstr. 30, D-86159 Augsburg, Germany
johannes.wagner@informatik.uni-augsburg.de

Abstract

Affect sensing by machines has been argued as an essential part of next-generation human-computer interaction (HCI). To this end, in the recent years a large number of studies have been conducted, which report automatic recognition of emotion as a difficult, but feasible task. However, most effort has been put towards offline analysis, whereas to date only few applications exist, which are able to react to a user's emotion in real-time. In response to this deficit we introduce a framework we call Smart Sensor Integration (SSI), which considerably jump-starts the development of multimodal online emotion recognition (OER) systems. In particular SSI supports the pattern recognition pipeline by offering tailored tools for data segmentation, feature extraction, and pattern recognition, as well as, tools to apply them offline (training phase) and online (real-time recognition). Furthermore, it has been designed to handle input from various input modalities and to suit the fusion of multimodal information.

1. Introduction

The modeling and simulation of affect-aware behaviors has been recognized as an essential factor for successful man-machine communication [11, 12]. While a significant amount of effort has been spent on the development of computational models of emotion, such as EMA [10] and ALMA [3], hardly any support is provided for multimodal emotion recognition in real-time which is a necessary prerequisite for the realization of affect-aware user interfaces. In fact, most approaches so far have dealt with offline classification of emotion and the few systems for online emotion recognition (OER) usually concentrate on one modality only, such as speech (EmoVoice [15]) or mimics (SHORE[1] [7] or [5]), while multimodal approaches, such as Gaze-X [9], are based on components that deliver affective output, but do not support their development.

While for the offline analysis of emotional corpora a number of powerful tools are available, such as Anvil[2] for data annotation, Matlab for signal processing and Weka[3] for classification, only little support is given for OER systems. This is due to varied difficulties real-time capability implies, *e.g*. requirement for automatic segmentation and low-cost algorithms. Apart from that it is no longer possible to handle processing in temporally independent steps, which also hampers the use of specialized tools for each task. Instead, a simultaneous execution is required, *i.e*. sensor data must be permanently captured and processed, while at the same time classification has to be invoked on detected segments.

On the other hand, architectures for multimodal processing, such as Pure Data [13], EyesWeb [1] or OpenInterface [14] are not specifically tuned to the recognition of emotions. For instance, they are often specialized on certain modalities, *e.g*. Pure Data on audio and video, and offer no support in respect of other input devices. Yet, in order to exploit multimodal interaction in its full richness, other input devices should be supported, such as physiological sensors. At the same time, limited support is given for the fusion of data from different sensors, which is an important requirement for the full understanding of a user's affective state. Finally, emotion recognition requires the exploration of real data with pattern recognition techniques. However, frameworks, such as OpenInterface, do rather encourage the integration of ready-made components than offering support for the pattern recognition pipeline and providing tools to collect training corpora.

We believe that the additional effort, which is necessary to move from an offline to an online system, and the lack of supporting tools for this task are reasons why to date only

[1] http://www.iis.fraunhofer.de/EN/bf/bv/kognitiv/biom/dd.jsp

[2] Anvil [6] is a video annotation tool offering hierarchical multi-layered annotation driven by user-defined annotation schemes. http://www.anvil-software.de/

[3] Weka is an open source software, which offers a large collection of machine learning algorithms for data mining tasks. http://www.cs.waikato.ac.nz/ml/weka/

few OER systems exist. In response to this deficit we introduce in this paper a framework for multimodal processing in real-time we call Smart Sensor Integration (SSI), which we believe is suited to considerably jump-start the development of multimodal OER systems. Due to a generic design, it is not limited to work with a certain input device, but supports any channel humans use to express their emotional state, including speech, mimic, gesture, pose and physiological signals. In many respects it adopts the structure of EmoVoice and similar approaches, but extends them by the possibility to process input from multiple modalities. In particular SSI supports the pattern recognition pipeline, assembled of tools for data segmentation, feature extraction and classification, and offers a generic GUI for data acquisition and training.

2. Online Emotion Recognition

An OER system falls into two main modules: an offline module for data acquisition and training of the classifier, and an online module, which uses the classifier for real-time tracking of a user's affective state. In this section the different tasks and problems an OER system is concerned with will be analyzed.

2.1. Online Recognition Pipeline

The recognition pipeline of the online module consists of three main parts.

Data segmentation is related to the task of detecting meaningful actions in the signal. In offline systems the units of choice are usually well-defined segments expressed under a certain emotions. For instance – in case of speech – utterances with no change of emotion. For an online system life is not that simple, as start and end have to be detected automatically and emotional changes may happen throughout a term.

Feature extraction is related to the task of extracting meaningful features from the signal. This can involve several steps, covering from pre-processing of the raw signal over the calculation of low-level features to the extraction of high-level statistics. In contrast to offline analysis, online systems are restricted to work with algorithms that deliver an answer in real-time and do not rely on future data. Often, a subset of relevant features is selected in order to remove irrelevant and redundant features. This can significantly improve recognition performance and speeds up the procedure as less features have to be calculated.

Classification is the last step, in which a classifier is used to map the observed feature vectors to discrete categories. While for offline analysis the efficiency of a classifier is of secondary importance, for an online system a trade-off between recognition performance and real-time capability has to be found. Usually, the choice depends on certain properties, such as the dimensionality of the input or the frequency with which classification will be invoked.

2.2. Data Acquisation and Classifier Training

Since the model a classifier uses to map continuous input to discrete categories is initially unknown it has to be learned from training data. This is a function of the offline module and falls into two main parts. The first part, referred to as *data acquisition*, involves the collection of representative training samples. Here, representative means that the picked samples should render the situation of the final system as accurate as possible. The second part concerns the actual *training* of the classifier. Basically, this is related to the problem of training a model that gives a good separation of the training samples, but at the same time is generic enough to achieve good results on unseen data.

Offline classification is usually evaluated on a fixed training and test set collected under similar experimental setting and by tuning the model parameters until they yield optimal recognition results. In contrast, the success of an online system depends on the ability to generalize on future data, which is not available to evaluate the classifier. Hence, special attention has to be paid that the training data is obtained in a situation, which is similar to the one it will be used in. To this end, also non-experts should be given the possibility to record emotional corpora in order to train personalized classifiers, which can be expected to give considerably higher accuracy than a general recognition system.

2.3. Multimodal Approach

So far studies on affect recognition have been largely concentrated on single modalities. In contrast, a multimodal OER system combines information of two or more modalities, *e.g.* from audio and video. Traditionally, three fusion levels are distinguished, namely data, feature, and decision level [11]. Data-level fusion involves merging of raw sensor data, which requires that the observations are of similar type and can be temporally aligned. Feature-level fusion takes place after sensory information has been first analyzed and is accomplished by joining the single feature vectors to a combined vector. Finally, decision-level fusion combines information from different modalities at the end of the analysis, *e.g.* by summing recognition probabilities to derive a final decision.

However, since humans communicate signals in a complementary and redundant manner multimodal signals cannot be considered mutually independent. Certainly it is not sufficient to process the different data streams separately and only combine them at the end. A multimodal approach should rather exploit fusion at all possible stages and mix between them. This, of course, demands a precise synchronization between the different data streams and the possibility to access information at different levels of processing. Since previous OER systems usually focus on one modality

only and do not support simple ways to access intermediate information, they are not suited for this task.

2.4. Smart Sensor Integration

In the following we present our SSI framework for multimodal processing in real-time, which has been designed to support the development of multimodal OER systems. In particular it supports the pattern recognition pipeline in its full length and suits the fusion of multimodal information at different stages. By default SSI handles a number of sensor devices, such as microphone, camera, or haptic and physiological sensors, but also allows the integration of new devices. Likewise, it includes general filter and feature algorithms, such as low/high pass filter, frequency analysis and statistical measurements, but still leaves the freedom to implement new or variant functionality.

In order to train a system on the situation it will be used in, we have developed a GUI for data acquisition and classifier training. Since the graphical interface has been completely decoupled from the signal processing part, it is not restricted to work with a specific sensor. In fact, it allows synchronized recordings from multiple devices. This unifies the task for the user and improves the exchangeability of training corpora. For the mentioned reasons we believe that SSI is qualified to considerably shorten the time necessary to implement and install multimodal OER systems.

3. The Architecture of SSI

3.1. Streams

Since SSI is not restricted to work with a particular type of signal an abstract concept for the treatment of information has been introduced. Information is processed in streams, where a stream is defined as a successive series of samples generated at a certain sample rate. In turn, a sample is characterized by its dimension and the size of a single value in bytes. Due to this abstract specification streams are handled completely independent of origin, content and abstraction. This encourages the integration of new sensor devices and supports the reuse of existing components. It also leaves the necessary freedom for combining different kind of information In the following the term information refers to a stream (or the segment of a stream) of the mentioned properties.

3.2. Layers

SSI has a three-layered architecture (see Figure 1).

On the lowest layer, referred to as *data layer*, synchronized snapshots of all information streams in the current workflow are buffered. In case of a read operation the requested buffer returns a copy of the information, so that the same information can be processed several times. In addition, it is possible to request information from several buffers for the same time slot, *e.g.* one could fetch a certain number of video frames from a buffer and get the according portion of audio from another buffer, which paves the way for fusion of information.

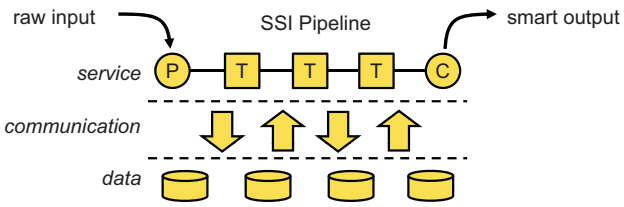

Figure 1. The three layer architecture of SSI.

The middle layer, called *communication layer*, is responsible for handling the exchange of information. It resolves logistical problems caused by simultaneous read and write operations executed on the same buffer. In regular intervals the internal clock of each buffer is compared to a global timer in order to check whether it is still synchronized and if necessary synchronization is restored.

The *service layer* located on the top is responsible for gathering and processing of information, as well as sharing it with the outside world. Different kinds of services are offered, which act as an interface to the data and communication layer and enable a developer to concentrate on the processing of information without having to take care where the information comes from and where it goes to. Services can be connected to pipelines in which the processing of information takes place.

3.3. Services

Three types of services are distinguished.

A *provider* feeds information into the system. It usually wraps a sensor device, which delivers data at a constant sample rate, such as a microphone or a video camera. If the connection to a sensor breaks down, the provider bridges the gap by sending default values. Since a provider decouples reading of information from further processing, any sensor device supported by the framework can be reused without the cause of additional implementation costs. Besides, temporary buffering of the raw measurements allows multiple processing even in cases where the device would only support a single connection.

A *transformer* is a service used to manipulate information. To this end, it constantly reads information from one or more buffer and combines/transforms it to a single stream. Afterwards, the result is written back to a different buffer. Hence, transformers offer an abstract interface for the integration of feature extraction algorithms. By putting multiple transformers in series it is possible to build arbitrary

processing pipelines. The toolkit contains a set of transformers that are frequently used in signal processing, but also allows the integration of new algorithms through an easy-to-use interface.

A *consumer* opens the system to the outside world. SSI contains generic consumers for storing information to disk, visualize it in a graph, or sending it to an external application through a socket connection. This way any information can be easily logged, monitored and shared. However, as we will see later on, consumers can also serve as classifiers in order to map continuous observations to discrete categories. Because a consumer reads information from one or more buffer without writing information back to the system, it is not restricted to work with a constant sample rate. For this purpose, the concept of a *trigger* has been introduced. Trigger control the workflow of one or more consumers by sending events consisting of timestamp and duration. Hence, a trigger breaks a continuous information stream into discrete segments.

3.4. Incorporated Tools

So far, we have explored the core architecture of SSI, which forms the backbone for the integration of the actual signal processing. However, instead of implementing a certain algorithm directly in a transformer's body, it may also include function calls to external sources, *e.g.* some signal processing library. In this way existing tools can be re-used and combined with the functionality of the SSI framework, such as buffering, synchronization and model training. In fact, several wrapper classes to efficient vision toolboxes are already part of the current implementation of SSI. This concerns OpenCV[4], a library for real-time image processing, ARToolKitPlus[5], a library for real-time marker tracking, and SHORE[6], a library for face and object detection and fine analysis. From our lab we have integrated EmoVoice and AuBT[7]. The latter includes tools to analyse physiological signals and has been ported from Matlab to C. Beside the classifiers brought along with EmoVoice, namely a Naïve Bayes and a Support Vector Machine, we have incorporated another three classifiers from the efficient machine-learning library Torch[8], namely a K-Nearest Neighbor, a Gaussian Mixture Model and Hidden Markov Models.

4. Emotion Recognition with SSI

We will now elaborate in what way the architecture of SSI supports the development of OER systems and present a GUI, which allows end-users to collect own training corpora in order to obtain personalized models.

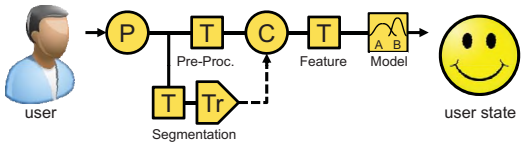

Figure 2. An online recognition system is assembled by connecting the pipelines for segmentation and feature extraction with a classifier.

4.1. Recognition Pipeline

In SSI data segmentation is done by triggers, often in combination with one or more transformers. For instance a voice activity detection (VAD) algorithm decides for each frame if it contains speech or noise by calculating features like zero-crossing or signal-to-noise ratio (SNR) and applying a threshold to them. In SSI it is possible (even though not necessary) to separate the pre-processing from the actual decision problem. Therefore, feature calculation is encapsulated into one or more transformers, which are placed between the input signal and the trigger. Since no more signal processing has to be applied on side of the trigger, a generic solution can be implemented, which compares connected streams to a set of thresholds. Hence, the same trigger can be re-used in other situations, *e.g.* to detect actions in physiological signals. The same pertains for the feature algorithms.

Obviously, feature extraction is delegated to one or more transformers. SSI offers two strategies for placing them in the pipeline. Either before the classifier, which means that the processing is applied continuously and the classifier clips out the segments, or directly attached to the classifier, which means that processing is postponed until a segment is received. The first option is preferred for pre-processing and low-level features, while the second option can be used to extract a final set of high-level features from the whole segment. Thanks to the modular layout, the pipeline, which extracts the features, can be easily extended or modified. This gives developers scope for testing new setups and encourages the re-usability of available algorithms. To apply the same processing during training the feature extraction pipeline can run without modification on offline data.

The final classification task happens with aid of a consumer, which is registered to the trigger and receives its input from one or more transformers. When the consumer is informed by the trigger about a new segment, it requests the according chunks from all connected transformers and forwards them to the classifier. In case of statistical classifiers, *e.g.* a Support Vector Machine classifier, first a set of final features has to be extracted as they require an input

[4] http://sourceforge.net/projects/opencvlibrary/
[5] http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php
[6] http://www.iis.fraunhofer.de/EN/bf/bv/kognitiv/biom/dd.jsp
[7] http://mm-werkstatt.informatik.uni-augsburg.de/aubt
[8] http://www.torch.ch/

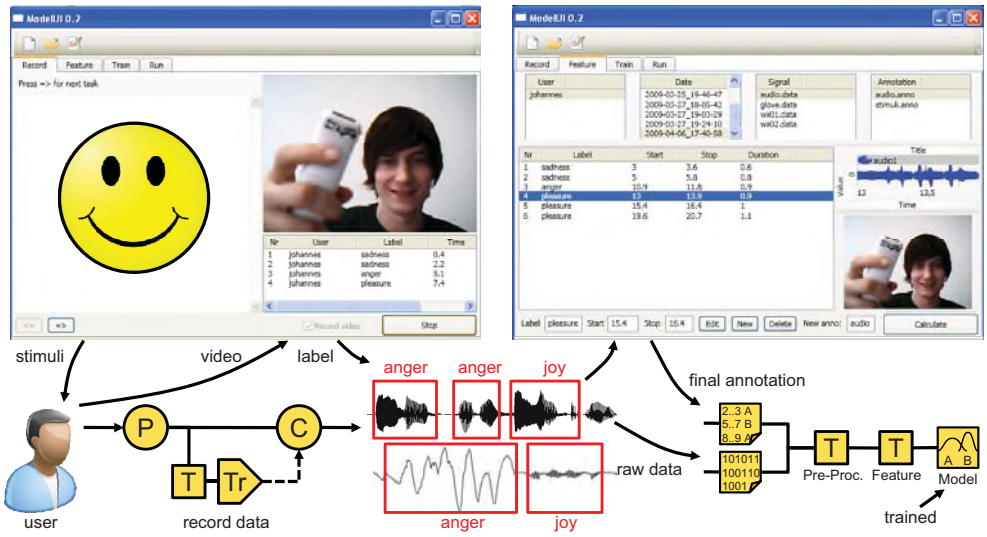

Figure 3. A graphical interface leads the user through the training procedure. The figure shows screenshots of two out of four panels, namely the recording and feature extraction panel. The GUI uses a dll interface to communicate with SSI (see sketches below the screenshots). In this way a complete decoupling of the GUI from the signal processing task is achieved.

vector of fixed length. In contrast, dynamic modeling approaches exist, *e.g.* Hidden Markov Models, which are able to cope with input of dynamic length. SSI supports classifiers of both types. In fact, since segmentation and feature extraction are completely decoupled from the classifier results different learning schemes can be compared similar to a data mining software such as Weka.

Figure 2 illustrates a schematic workflow of the final recognition pipeline. However, it shows only a very basic configuration, which may be expanded by additional processing blocks and input devices.

4.2. Model Training

As mentioned earlier, a model is a function which maps continuous observations to discrete categories. Since the function is unknown it has to be learned from training data. To obtain an adequate model it is indispensable to collect a sufficient number of representative examples for each target class. SSI offers a generic GUI that assists a user to collect own training corpora and use them to obtain personalized models. In this context generic means that the GUI is not restricted to work with a specific sensor. In fact, it even allows synchronized recordings from multiple devices.

To achieve a complete decoupling from the signal processing part the GUI uses a dll interface to communicate with SSI (or basically any other system, which implements the interface). For instance, if the user presses the record button, the GUI gives the command to start a new recording, but is not further involved in the data recording. Actually, it not even knows what types of signals are recorded, except for the video stream, which is made available for display. Since the dll is dynamically loaded at run-time the GUI can be used without modification for the training of classifiers based on different modalities, features and learning schemas.

The screenshot on the left side of Figure 3 shows the recording panel. To elicit the desired behavior a web browser has been placed on the left, which is used to present a stimulus to the user (*e.g.* text instructions, pictures, videos, ...). During recording a SSI tool runs in the background and captures the raw data streams from the connected sensors. The raw sensor data, as well as a video, are synchronized and stored to disk. In addition, the same triggers that are later going to be used in the online module are now applied to yield a pre-segmentation of the signals. Detected segments are labeled with the stimuli that are displayed at that time. This way a preliminary annotation is created.

Finished recordings can be reviewed in the feature extraction panel (see right screenshot of Figure 3). On the left side of the panel the annotation is loaded. If a segment is selected the according video sequence and a graph of the data chunk for this interval are displayed. Now the user has the possibility to adjust the boundaries and change the label of the segment. Of course, it is also possible to delete or add entries. Optionally, import and export functions to Anvil are available in order to further refine the annotation.

Finally, recordings can be split according to a selected annotation. For each segment in the annotation a feature file is created and based on a set of those feature files a classifier can be trained. The performance of a classifier can then be measured using k-fold evaluation and if necessary the user

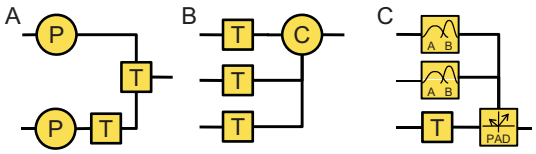

Figure 4. The SSI framework supports fusion of information at different levels, as the examples above illustrate.

can collect additional data until the desired performance is achieved. Of course, new feature extraction algorithms and classifiers can be tested on previous recordings, too. All this is done in the training panel, which is not shown here due to lack of space.

4.3. Fusion

SSI offers several possibilities for the fusion of information. Three of them are exposed in Figure 4.

Example A shows the fusion of a stream of raw sensor data with a low-level feature stream extracted from a second sensor. It is done with a transformer that receives data from both sources and combines it to a single stream.

Example B realizes a classical feature fusion, in which features from different sources are concatenated to a single feature vector. This happens with aid of a consumer, which, for instance, may serve as a classifier that was trained on a combined feature space.

Example C illustrates a case where the affective output of two classifiers is fused with a continuous feature stream. For this purpose, a novel fusion mechanism explored in [4] was integrated. It is based on the PAD model and measures emotional response along three dimensions: pleasure (P), arousal (A) and dominance (D). Hence, the discrete emotion classes are first mapped onto specific PAD values, whereas the continuous feature input can be directly applied given that it produces values in a suitable range and dimension. The outputs of the connected sources are then used to update a global PAD vector representing the final decision.

5. Building concrete OER systems with SSI

In the following, we will show by means of existing applications how different variants of an OER system may be implemented using SSI. For the sake of objectivity, we will not rely on our own applications, but take representative examples created by extern institutes within the EU-funded CALLAS project and IRIS network. We will start with a simple unimodal OER system and stepwise enlarge it to a multimodal system.

The implementation of a unimodal OER system will be illustrated by an IRIS showcase called EmoEmma [2]. EmoEmma is an interactive storytelling system based on Flaubert's novel "Madame Bovary" where users can influence the outcome of the story by acting as one of the characters and their interaction mode is restricted to the emotional tone of their voice.

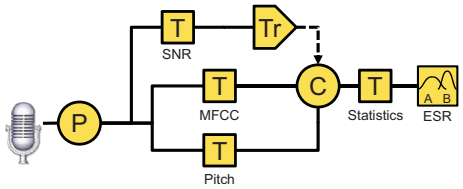

Figure 5. An OER system to detect emotional speech based on MFCC and Pitch features.

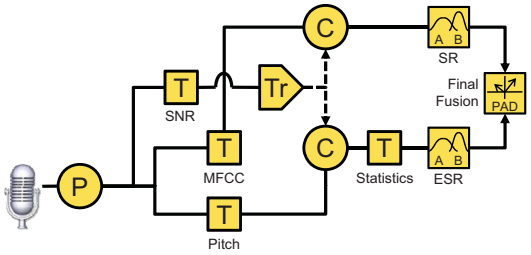

Figure 6. The basic system of Figure 5 enhanced by an speech recognizer in order to combine paralinguistic and semantic information. Both entities are fused in a PAD model.

Figure 5 shows the flow chart of an OER system to detect emotional speech. This system realizes the functionality of real-time vocal emotion recognition systems, such as EmoVoice, which is currently being used in the EmoEmma showcase. It receives input from a microphone, which is connected through a provider and supplies three types of transformer that extract different low-level feature, namely SNR, MFCCs and Pitch. Based on the SNR feature a trigger is used to detect speech events, while MFCCs and Pitch build the input for the emotional speech recognizer (ESR). However, when speech is detected first a number of high-level statistics are extracted. The final feature vector is then analyzed by the ESR.

Now, let us move to a CALLAS showcase with a more complex emotion recognition component, the E-Tree [4], which takes advantage of multimodal input. E-Tree is an Augmented Reality Art installation of a virtual tree that can be made grow or shrink, change colors etc. by affective multimodal input. To integrate the results of the acoustic and lexical recognition components, the PAD model is used. In order to be able to consider lexical information in the E-Tree, we enhance the system by a speech recognizer (SR), which converts spoken words to text in order to combine the paralinguistic analysis with additional semantic information. Fortunately, we can share the speech trigger and the MFCC feature stream, so that the SR module can be directly installed. Since a HMM is used by the SR no further

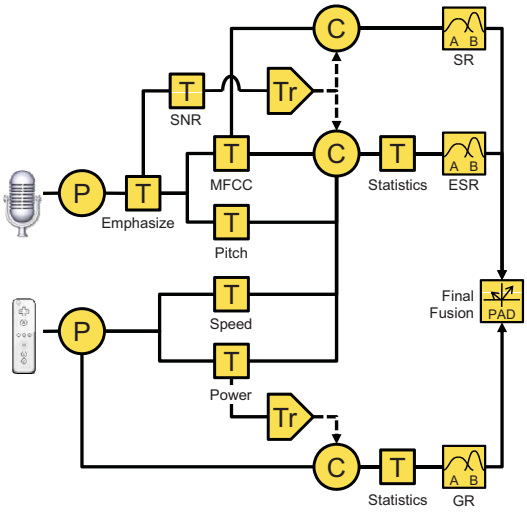

Figure 7. The approach of Figure 6 extended to a multimodal system. The acceleration measured by a wiimote is used in a two-fold way: to recognize gestures (GR) and to extract expressive features as additional input to the ESR module.

processing step has to be applied between the MFCC transformer and the classifier. The predictions returned by the SR and ESR module are then evaluated according to their emotional content and following the approach used in the E-Tree, we map emotion categories onto dimensional values of the PAD model. Of course, SSI is not limited to emotion recognition approaches that provide emotion classes, but can also handle emotion recognition systems that deliver continuous dimensional values. The extended system is shown in Figure 6.

Next, let us upgrade the E-Tree to a multimodal OER system in which the ESR is enhanced by features extracted from an acceleration signal measured by a wiimote.[9] This is reasonable since human beings often accompany speech with expressive gestures. To this end, a wiimote is placed as a second sensor and again connected through a provider. Since both providers are now controlled by the same internal clocking mechanism it is ensured that data from microphone and wiimote is read in a synchronized manner. As a matter of fact, this also guarantees temporal correspondence between derived signals from both modalities. Hence, it will do to connect the ESR consumer with affective features extracted from the wiimote (here speed and power). Apart from that, the wiimote is also connected to a gesture recognizer (GR) that maps changes in acceleration over time to gestures. To recognize the start and end of gesture, we search for peaks in the power of the acceleration signal. The outcome of the GR module is again linked to coordinates in the PAD space. To once again stress the flexibility of the SSI framework, we also place a pre-emphasize filter between the audio provider and the audio feature blocks, which now retrieve the filtered signal instead of raw input. In SSI such changes can be applied with minimal effort. The according flow chart is given in Figure 7.

[9]The current E-Tree does not include a wiimote, but in order to illustrate the easy extensibility of recognition mechanisms developed with SSI, we discuss the example here.

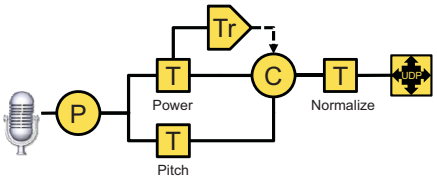

Figure 8. SSI is not restricted to produce high-level emotional information, but may also share continuous low-level feature, *e.g.* via UDP.

In the examples above, the showcases required higher-level emotional information represented as values of a PAD model. SSI provides, however, also support for the design of applications that respond directly to continuous low-level feature input. For illustration, let us have a look at the PuppetWall showcase [8]. In this CALLAS showcase, a user may change the atmosphere of a drawing by the acoustics of his voice. Since in such an environment often a prompt reaction is wanted and since puppeteering encourages users to produce noises apart from speech, such as clapping or singing, one may prefer to directly control the appearance with low-level features, *e.g.* pitch or energy.[10] However, since in SSI information can be accessed at any point in the pipeline, the original OER system of Figure 5 can be quickly re-arranged to fulfill the new requirements. We simply replace the MFCC block by a Power transformer, which we also use as input for the activity trigger. Then we remove the classifier and instead normalize the values to the desired interval, *e.g.* $[0..1]$. Finally we plug an UDP sender at the end of the pipeline in order to stream the values through a socket connection. The new configuration is sketched in Figure 8. Of course, it is still possible to have a classifier running at the same time, which shares the features.

6. SSI Applications

SSI has been successfully employed in a number of HCI projects at our lab, such as an attentive virtual butler which responds to the user's emotional state. Outside our lab, SSI is being used in the EU-funded Metabo project for real-time physiological data analysis of diabetes patients in an automotive environment, as well as, in showcases of CALLAS and IRIS, such as EmoEmma (see Section 5). Furthermore, CALLAS partners developed a game-like application based

[10]In fact, an early version of the PuppetWall was implemented in this way.

on SSI in which users can experiment with their voice and a wiimote to produce interactive art. Beside the extraction of expressivity features as input for the application, here SSI was also used to capture the raw data streams and grab a video of the user interaction. In this way a multimodal emotional corpus was created, which now serves for the further analysis of the experiment. A comprehensive corpus of affective interactions synchronizing speech, mimics, and gestures is currently being recorded in three different countries with SSI. This corpus does not only contain video-based gestures, but also gestures that were performed using two wiimotes or a data glove.

7. Conclusion

We presented SSI, a framework for multimodal processing in real-time. In contrast to other multimedia frameworks SSI has been designed with a strong focus on emotion recognition and offers tools for the development of multimodal OER systems. In particular it supports the pattern recognition pipeline in its full length and allows the fusion of multimodal information at different stages. Furthermore, a GUI for data acquisition and training has been presented, which can be used with all kind of input devices, such as microphone and camera, but also haptic and physiological sensors. This unifies the task for the user and improves the exchangeability of training corpora.

Throughout the paper we discussed the architecture of the SSI framework and explained in what way it suits the recognition and fusion of emotional cues from multimodal input. We outlined various uses of SSI by means of selected showcases from the CALLAS project. The showcases were employing multiple input channels, were fusing information at various levels of abstraction and were driving applications that expected as input emotion classes as well as features. We illustrated the flexibility of SSI in describing and realizing this variety of showcases.

The SSI framework and the training GUI will be publically available in September 2009 under the following address: `http://mm-werkstatt.informatik.uni-augsburg.de/ssi.html`

8. Acknowledgments

The work described in this paper is funded by the EU under research grants CALLAS (IST-34800), IRIS (Reference: 231824) and Metabo (Reference: 216270).

References

[1] A. Camurri, P. Coletta, G. Varni, and S. Ghisio. Developing multimodal interactive systems with eyesweb xmi. In *NIME '07: Proc. of the 7th international conference on New interfaces for musical expression*, pages 305–308, New York, NY, USA, 2007. ACM.

[2] F. Charles, D. Pizzi, M. Cavazza, T. Vogt, and E. André. Emotional input for character-based interactive storytelling. In *The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Budapest, Hungary, 2009.

[3] P. Gebhard. ALMA - a layered model of affect. In *Proc. of the 4th International Joint Conference on Autonomous Agents and MultiAgent Systems*, pages 29–36, 2005.

[4] S. W. Gilroy, M. Cavazza, R. Chaignon, S.-M. Mäkelä, M. Niranen, E. André, T. Vogt, J. Urbain, H. Seichter, M. Billinghurst, and M. Benayoun. An affective model of user experience for interactive art. In *ACE '08: Proc. of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 107–110, New York, NY, USA, 2008. ACM.

[5] R. E. Kaliouby and P. Robinson. Real-time inference of complex mental states from facial expressions and head gestures. *Computer Vision and Pattern Recognition Workshop*, 10:154, 2004.

[6] M. Kipp. Anvil - a generic annotation tool for multimodal dialogue. In *Proc. of the 7th European Conference on Speech Communication and Technology (Eurospeech)*, pages 1367–1370, Aalborg, September 2001.

[7] C. Küblbeck and A. Ernst. Face detection and tracking in video sequences using the modifiedcensus transformation. *Image and Vision Computing*, 24(6):564 – 572, 2006. Face Processing in Video Sequences.

[8] L. A. Liikkanen, G. Jacucci, E. Huvio, T. Laitinen, and E. Andre. Exploring emotions and multimodality in digitally augmented puppeteering. In *AVI '08: Proc. of the working conference on Advanced visual interfaces*, pages 339–342, New York, NY, USA, 2008. ACM.

[9] L. Maat and M. Pantic. Gaze-x: Adaptive, affective, multimodal interface for single-user office scenarios. pages 251–271. 2007.

[10] S. Marsella and J. Gratch. EMA: A computational model of appraisal dynamics. In *European Meeting on Cybernetics and Systems Research*, 2006.

[11] M. Pantic and L. J. M. Rothkrantz. Toward an affect-sensitive multimodal human-computer interaction. In *Proc. of the IEEE*, pages 1370–1390, 2003.

[12] R. Picard. Affective computing. Technical Report 321, MIT Media Laboratory, Perceptual Computing Section, November 1995.

[13] M. Puckette. Pure data: another integrated computer music environment. In *Proc. of the Second Intercollege Computer Music Concerts*, pages 37–41, 1996.

[14] M. Serrano, L. Nigay, J.-Y. L. Lawson, A. Ramsay, R. Murray-Smith, and S. Denef. The openinterface framework: a tool for multimodal interaction. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 3501–3506, New York, NY, USA, 2008. ACM.

[15] T. Vogt, E. André, and N. Bee. Emovoice - a framework for online recognition of emotions from voice. In *Proc. of Workshop on Perception and Interactive Technologies for Speech-Based Systems*, Kloster Irsee, Germany, June 2008. Springer.