Deliverable 5.3.1

Project Title	Next-Generation Hybrid Broadcast Broadband
Project Acronym	HBB-NEXT
Call Identifier	FP7-ICT-2011-7
Starting Date	01.10.2011
End Date	31.03.2014
Contract no.	287848
Deliverable no.	5.3.1
Deliverable Name	Design and Protocol: Intermediate Multimodal Interface and Context Aware Recommendation Engine
Work package	5
Nature	Report
Dissemination	Public
Author	Mark Gülbahar, Michael Probst (IRT); Sebastian Schumann (ST); Gregor Rozinaj, Ivan Minarik, Renata Rybárová, Miloš Oravec (STUBA)
Contributors	Jarmila Pavlovičová, Marek Vančo, Matúš Vasek, Juraj Kačur, Matej Féder (STUBA)
Due Date	31.05.13
Actual Delivery Date	31.05.13

Table of Contents

Executive Summary

This document is the second deliverable for Task 5.2 of the HBB-NEXT project. It is one of two Milestone 7 deliverables created by WP5 and thus presents Milestone 7 for the Multimodal Interface Enabler part of WP5: "MS7: The intermediate version of the software components of WP3, 4 and 5 is in place". The document presents designs and first prototype implementations related to personalisation of next-generation hybrid-broadcast-broadband television. It provides a current (May 2013) view of the status of the designs, prototypes, integrations and demonstrations towards a central use case.

Use case: The central use case of WP5 is described as follows. One person watches television and retrieves a list of content recommendations. Then a second person enters the room, and he is also identified by the system. Subsequently, the recommendation list changes, tailored to the taste of the two persons together. The whole scenario is based on the personalization engine.

Design: A preliminary system design is presented, based on the following components: multi-modal and other solutions for identifying the users, e.g. multi-user speaker identification, multi-face recognition, voice command recognition, speech synthesis; a personalisation framework to collect and use personalisation information; a notification framework to support asynchronous interactions between the different components; and Java and RESTful APIs specifying the communication messages between the components.

Prototypes: The following initial prototypes have been realised: a voice-recognition-based system for identifying users; a QR-code-based user identification system; multi-face recognition; static gesture recognition; and a graphical user interface for the user to receive the recommendations, to watch the metadata and to select the content.

The consequent part of the prototype will be developed within D5.4.1 [27]: a novel and highly efficient collaborative filtering algorithm that calculates the "preference neighbourhood" of a user based on genre preference; a metadata enrichment system that aggregates metadata from various internet sources and outputs it in TV-Anytime format; the coordinating recommendation engine framework (PREF), including a first multi-user algorithm based on the "least misery" strategy.

1. Introduction and role in the system

One of the major objectives of HBB-NEXT is to define an open framework for the next generation of hybrid broadcast internet (HBI or HBB) services. Building on top of deployed standards like HbbTV, HBB-NEXT defines a set of extensions and open APIs to enable more advanced services and business cases. On business models please refer to D2.4 [26] for more details.

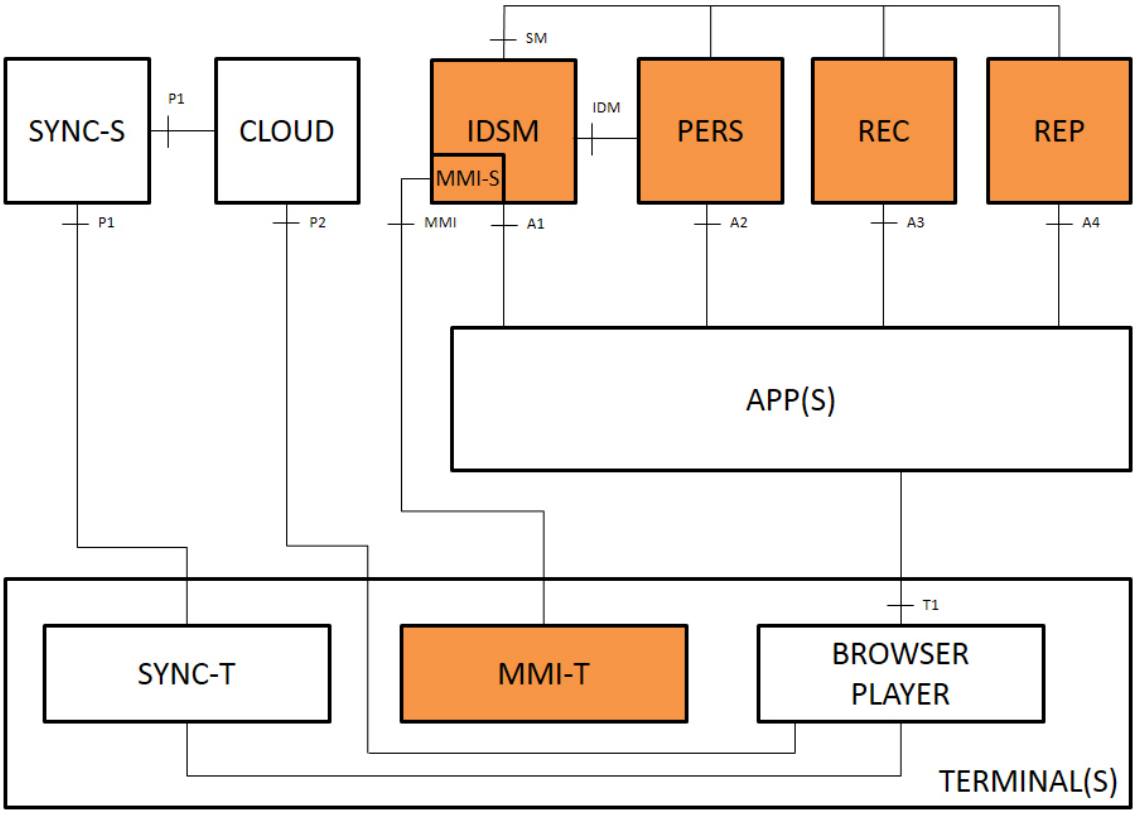

Figure 1: WP5 modules in the HBB-NEXT system architecture

Figure 1 shows the system architecture of the HBB-NEXT platform as developed in WP6. The diagram consists of enablers provided by WP3/4/5, which either reside on the internet (cloud) or on the terminal, and public interfaces to the HBB-NEXT applications (APPS). Interfaces in the diagram can be distinguished into APIs and protocols. APIs of internet based enablers are potentially exposed by one or multiple service provider and named A1, A2 etc. The API that is exposed to applications on the terminal is named T1. HBB-NEXT also defines open protocols for the advanced synchronization features.

These protocol interfaces are named P1 and P2. APIs between enablers are labelled with the name of the sub module that exposes it, MMI, IDM and SM.

WP5 contributes to the multimodal interface with focus to the multi-user identification, navigation the whole system (gesture recognition, speech recognition and speech synthesis) and using Identity and Security Manager to the recommendation engine.

This deliverable provides a first system design of a context aware group recommendation engine with a multimodal interface as designed and implemented in work package 5 of the HBB Next project. The "Multi-modal interface for Multi-user Service Personalisation Engine" enabler consists of different software modules. As shown in

Figure 2, these modules must work together to realize the use case that is currently central to WP5: One person watches television and retrieves a list of content recommendations. When a second person enters the room and is identified by the system, the recommendation list changes, tailored to the taste of the two persons together.

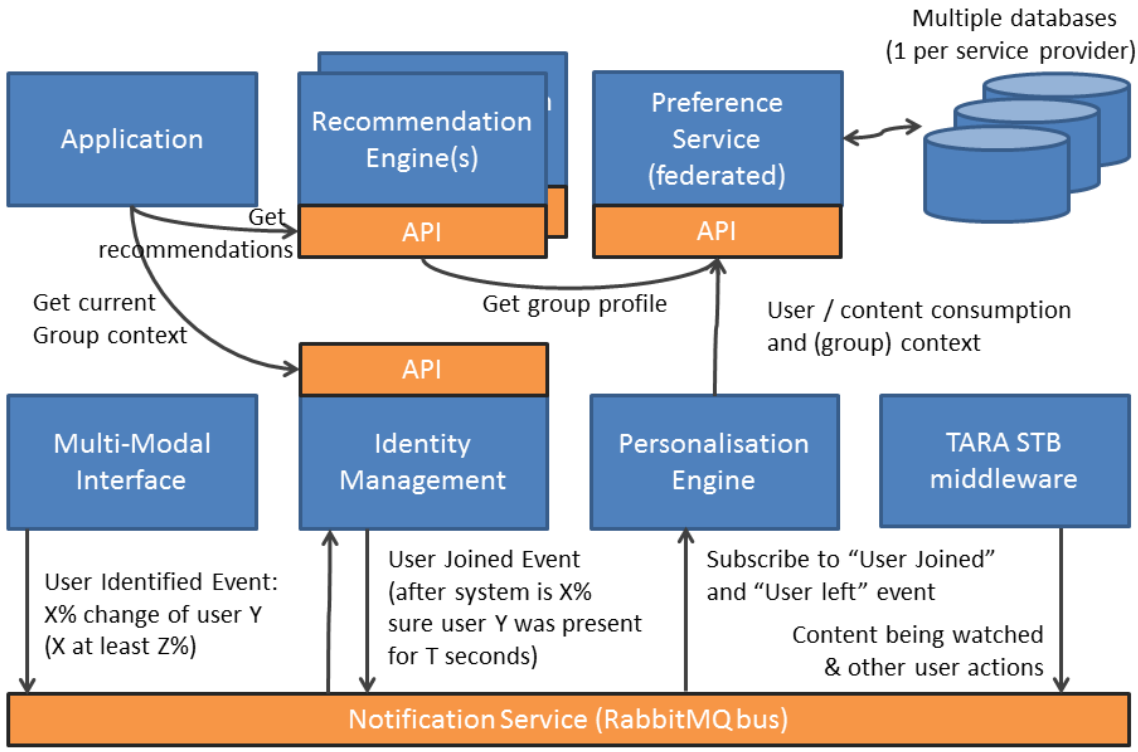

Figure 2: Relationships between WP5 modules

The recommendations that are generated by the Recommendation Engine are provided to the user(s) through the Application. The identification of the users that are in front of the TV and their interactions with the Application are handled by the Multi modal interface module. Information on the presence of specific users and the content they are consuming is combined by the Personalisation Engine and translated into preferences that are stored by the Preference Service. The Preference Service is the source of user and group preferences on which the Recommendation Engine is basing its recommendations. These modules communicate with each other through the Notification Service. Note that the Identity Management modules as well as the TARA set-top box middleware are not part of WP5 and are out of this document's scope. Of course, we use e.g. Identity Manager for user identification in multi-user environment to provide relevant information for Recommendation engine.

The remainder of this document is structured with respect to the modules presented in Figure 2 and their tasks. Chapter 2 describes the modules related to Task 5.2: the personalisation engine (Section 2.1), notification service (Section 2.2) and multi-modal interface (Section 2.3).

2. Multimodal Interface for User/Group-Aware Personalisation in a Multi-User Environment

This chapter describes the personalisation engine (Section 2.1), notification service (Section 2.2), and multi-modal interface (Section 2.3).

2.1. Personalisation Engine

This chapter describes the personalization engine (PE), an enabler that allows applications to easily provide context-specific group data on the basis of single user profile information.

The enabler expects applications to request information for certain contexts, e.g., the number of persons in a room, their age etc. The application can determine the context through the Identity Management (IdM) enabler (via user/device relation) if it does not know the context ID itself. It will interact with the IdM enabler to retrieve the active users in this context. Through the Profile Management (PM), the personalization engine retrieves the profiles of the active users. In its first version, the PE will provide aggregated information for basic profile parameters. In the future, the API may be extended with the ability to receive more complex algorithms directly in the request and provide personalized information using them. This allows application requests to pass the personalization logic via the API (per request or pre-provision).

2.1.1. Design

According to the brief description in Deliverable D6.1.1 [20] the PE shall perform the following functions:

- Retrieve and modify single user profile in the system (via PM)
- Get and set group profile for the active context

The function “Retrieve and modify single user profile in the system (via PM)” is handled via the API of the PM module, which is described in D3.2 [21]. Section 2.1.2 provides the description of the API for managing the group profile.

The group context is an object provided by the personalization engine, including:

- A context ID (for easy reference in subsequent requests) that contains parameterized information about the group, potential aliases
- All users of a context

The group context data model is shown in Table 1. .

Level 1	Level 2	Level 3	JSON Type	JSON Object	Obligation	Ocurrences
				(preliminary)		
Identifier (unique)			object		M	1
	ID		string	id	M	1
	Alias		array	alias	O	n
Users			array	users	M	1
Parameters					O	1
	Amount of users		integer	persons	O	1
	Average Age		number	avg_age	O	1
	Type of group		string	tog	O	1
API version:	1					
Date:	8.9.2012					

Table 1. Group context data model

The data model may be extended for future API versions. In the initial version the parameters showcase only the aspired capability of the enabler. In the future, sample services (incl. service profiles and user parameters) will be created and the group profile will be extended with service specific personalization.

2.1.2. Protocol

This subsection describes the API of the personalization engine and also provides sample sequence diagrams.

Function (Ref. D6.1.1):	Get and set group profile for the active context		
Function:	Get group profile		
Method:	GET	**Resource:**	*/contexts/{contextid}*
Parameters:			

contextid	The id of the group context for which the personalized profile that shall be retrieved.

Function (Ref. D6.1.1):	Get and set group profile for the active context		
Function:	Set group profile		
Method:	PUT	**Resource:**	*/contexts/{contextid}*
Parameters:			
contextid	The id of the group context for which the personalized profile that shall be set.		

Note: Setting the group profile makes only sense once the API is extended. The values in the profile are currently provided per-user. The function makes sense if group profiles are used to set certain values for all users. Instead of setting the values per user, the personalization engine receives the group feedback and provisions individual user values back to their single user profiles.

Function (Ref. D6.1.1):	n/a		
Function:	Retrieve users in a group		
Method:	GET	**Resource:**	*/contexts/{contextid}/users*
Parameters:			
contextid	The id of the group context for which the active users shall be retrieved.		

The response shall contain a body with the users in the group.

The final API description will be made in line with the IdM/PM API described in deliverable D3.2 [21] from WP3.

Note: With regard to context, it shall be clearly understood that the personalization engine in WP5 handles the user profiles in the active context. The IdM from WP3 has a context element as well; it is used however to manage the usual and actual context of a user. While the context assignment in WP3 is rather static and only its activity changes, the context assignment in scope of this WP is dynamic. Further study will clarify this and provide implemented samples to showcase the difference.

Deliverable D3.2 [21] contains a sequence diagram that covers also the personalization engine from this WP.

2.2. Notification Service

The notification service is based on a message queue server to send and subscribe for events of the HBB-NEXT framework. Events in the HBB-NEXT context are regard as information to describe changes in the context of both system (e.g. “the HBB-NEXT device has tuned to channel RBB”) and users (e.g. “Anna has taken seat in front of the TV”).

2.2.1. Design

Since software implementations of messaging systems exist for a long time, there is no need to design a new / HBB-NEXT specific solution. Instead, an analysis of these existing solutions was performed with respect to usability within the scope of HBB-NEXT use cases and requirements. The most well-known solutions are XMPP [16] and RabbitMQ [17]. Since the latter is also used within the Openstack IaaS Cloud Systems [19] and also Javascript libraries for RabbitMQ exist [18], the choice fell onto RabbitMQ.

For first system integration, a RabbitMQ Server [17] was installed on a PC within the home network (e.g. the DLNA Home Gateway).All clients (HBB-NEXT STB, 2nd screen devices etc.) will connect to this Server in order to subscribe to events and to receive publications of events. In later stages of the project, the deployment of a RabbitMQ Server on the TARA STB will be investigated.

2.2.2. Protocol

The protocol used by RabbitMQ is AMQP, an open and flexible implementation. RabbitMQ extends the AQMP specification, providing several protocol extensions. A detailed description of the RabbitMQ protocol can be found on the official website [17].

2.3. Multi Modal Interface

Multimodal interface (MMI) consists of several topics and modules which serve for natural and user-friendly communication with the system. Altogether, these modules represent the functionality of of MMI. This section describes the four modules that are currently part of the multi-modal interface:

- Multi voice identification
- Speech and command recognition
- Speech & voice command recognition
- Gesture Navigation
- Eye Navigation
- Speech synthesis

The following subsection explains their algorithms and the approaches. Several other modules e.g. Multi face recognition are the part of the MMI, as well, which are described in D3.3.1 [25].

2.3.1. Multi voice identification

Multi-speaker identification aims to identify possibly more speakers based on a recorded signal that may contain utterances of more individuals. This general task can be separated into several categories based on additional refinements [26]. If the speakers who may appear in given conversation are known in prior, i.e. they were present in some sort of training phase, the task resembles a single speaker identification problem, even though additional algorithms must be applied, tuned and enhanced. However, when the set of possible speakers is unknown, the techniques of speaker segmentation and clustering (diarization) must be used.

In doing so it is expected that there is no serious overlap of utterances of different speakers. In this scenario and setting the detection of changing points for different speakers is the main focus instead of knowing their identity (which can't be done as no prior information exist).

The aim of the designed application is to continuously run and "listen" to an incoming stream of PCM samples (sound waves), detect voice activity (VAD), silence and background noises, and if substantially long voice period is caught then identify the speaker with certain confidence measure. Optionally, system detecting overlapped speech can be employed as well. As it can be seen the main focus here is not in speaker clustering and segmentation, thus additional and extended algorithms for speaker identification (multi-user) will be outlined in the following.

The two main parts of speaker identification system are speech features methods and classification algorithms. As they are in this scenario similar to the single speaker identification, please see the basic principle of MFCC construction and KNN classification in a single user system description; deliverable D3.3.1 [25]. Here the modifications to improve the speed and accuracy in the case of having less training samples are given.

2.3.1.1. K-D trees and K Nearest Neighbour

KNN is a simple but in certain conditions very accurate method. However, its main drawback is that its recognition phase takes a huge computational load. To alleviate this, data is stored in k dimensional binary trees. Each node has a key (dimension upon which it performs a space division) and a dividing value that should be the most separating for a given level and dimension. Even though to find a proper element takes log2 searches, the situation is more complicated as more individuals must be found (k neighbours). In addition, different leaves are usually needed to be searched as well. To do so, two functions must be implemented: within_bonds and overlaped_region. The first one checks if the worst neighbour can be outperformed by another vector stored in remaining leafs. The second one determines whether it is necessary to search also "other" leaf, not the original which the tested vector belongs to. Using the above mentioned function the tree is recursively searched from top to bottom and when all possible neighbours are found the algorithm stops.

2.3.1.2. GMM

GMM is the parameter based classification [27] which presents the feature space by a number of Gaussian mixtures. It is beneficial if there is less data and is computationally faster than KNN in the recognition phase; however it requires more advanced training phase.

It is assumed that a linear combination of Gaussian densities can approximate with arbitrary small error any continuous statistical distribution. A multidimensional GMM is thus given as:

$$pdf(\mathbf{x}) = \sum_{i=1}^{N} \pi_i \frac{1}{\sqrt{(2\pi)^d \det(\mathbf{\Sigma}_i)}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)\mathbf{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)^T}$$

where x is the observation vector, μ_i is the mean of i-th mixture, Σ_i is the covariance matrix of i-th mixture, π_i is the weight of i-th mixture, and N is the number of mixtures. Thus the main problem is to estimate the mean vectors, covariance matrices and weights of all Gaussian distribution. It is done by the well-known EM algorithm that is a maximum likelihood (ML) estimator. Again only a local maximum of ML is guaranteed. Then the recognition may follow the optimal Bayes classification rule where each speaker (its distribution) is represented by GMM. Using such models required likelihoods $P(x/class_i)$ can be easily calculated.

2.3.1.3. VAD

Voice activity detection [27] is usually done based on simple features like energy, intensity, log energy, zero-crossings, periodicity, etc. Its main aim is to separate voices and background noises. In the presence of high SNR the task is rather simple and purely energy-threshold based. However in most situations this is not the case and various combinations of different types of features must be used and adapted in real time. In addition to energy pitch period or voicing measure can be used in some time interval and further evaluated. Pitch can be estimated via xcorr or amdf functions [27] as follows:

$$AMDF(k) = \sum_{n=0}^{n<N} \left| x(n) - x(n+k) \right| \quad k \in < T_{0\min}, T_{0\max} >$$

In the case of k is a period or its multiplication, amdf exhibits low values (in ideal case zeros). Finding potential pitch a voicing feature can be simply estimated as the measure of resemblance between the original and shifted signal introducing some kind of normalization.

2.3.1.4. Overlapped speech (two or more speakers speak at the same time)

Overlapped speech can be detected directly using acoustical features e.g. pitch period which in the case of two competing speeches is not unique and usually its trace is rather weak. It can be detected indirectly using classification algorithms where in the intervals of competed speeches rather unstable identifications with low confidences are observed. Even in such a situation usable speech for classification can be derived from voiced/unvoiced intervals where speeches have different energy levels.

2.3.1.5. Theoretical Basis –research activities and achievements

As all the designed modifications, improvements and optimizations are related to the feature extraction methods and classification techniques that are basically the same for a single and multi-user system, please see deliverable D3.3.1 [25] for more details.

2.3.1.6. Algorithms description

For the sake of simplicity, maintenance and evolution, the application is divided into 4 parts, namely: recording of training samples, parameterization of samples, constructions of KD-trees, and finally the identification module. All these standalone programs are controlled and invoked by a user interface application, as shown in Figure 3.

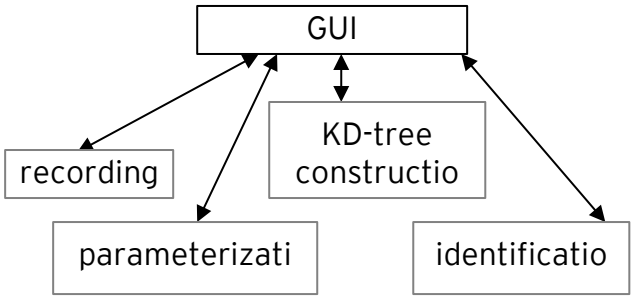

Figure 3: A block scheme of the identification system

GUI

The graphical user interface offers several options: recording a new user, parameterization of recorded speech, and construction of KD trees. These operations are enclosed in the training option and are executed sequentially for each user. If they are finished successfully the user is added into the database. Then there is an identification options that runs an identification process in real time and displays one or more identified user each time interval (700ms). Finally listing of recorded users and user deleting options are also available, see Figure 10.

User recording application

Its main aim is to record a new user. Based on the entered name through the GUI application the application starts a recording process. The process constantly checks the incoming data stream if an eligible speech is in the input. It does so by the voice activity detection (VAD) algorithm. When a sufficiently large portion of voice inactive part is detected the recording automatically stops. Afterwards the algorithm checks if there is a usable speech. If so, the raw data is saved under proper name in a directory. The block scheme is shown in Figure 4. If there is insufficient data the new user is not saved and the GUI application is warned about this situation.

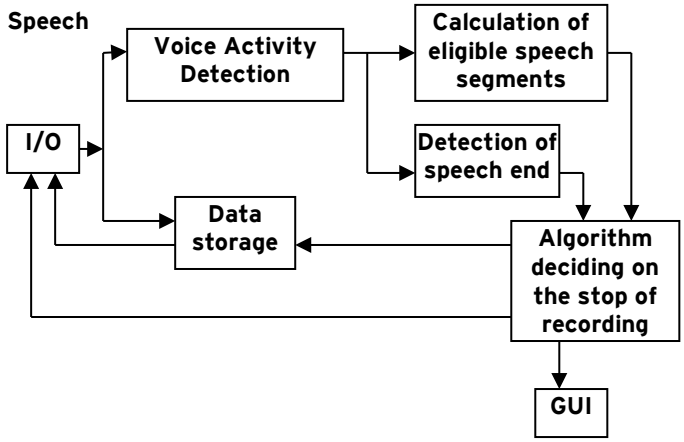

Figure 4: A block scheme of the user recording application

Speech feature extraction application

The basic algorithm of MFCC is enhanced by voice activity detection so that only eligible speech segments that contain identification information are processed and stored into the training database. If a signal segment is classified as a speech segment it is further processed as follows: segmented into frames (20ms long), high pass filtered, windowed by the Hamming window, subjected to FFT transform, warped into Mel spectra, and energies in each band are calculated. Finally logarithm and DCT transform are applied to derive MFCC features (see Figure 5).

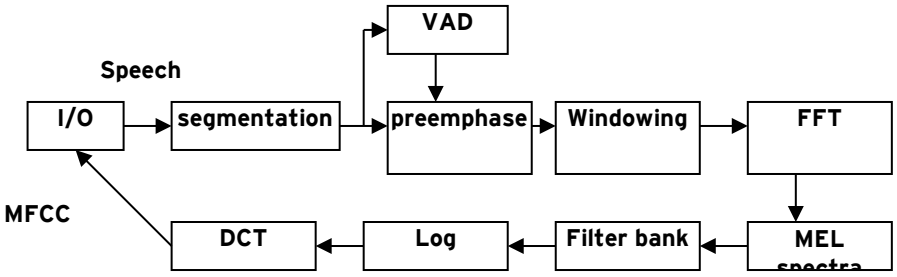

Figure 5:A block scheme of the feature extraction application

Construction of KD tree

In order to speed up the basic KNN algorithm training data is stored in a KD tree. This is performed by the KD tree construction application, that takes all speech features from all users, and in iterations finds the dimension with highest dispersions that it splits in half (in its median, so two nods are created). Each newly created nod is checked for further splitting (key- dimension, and the splitting value). The process goes till the prescribed size of a tree is achieved, see Figure 6.

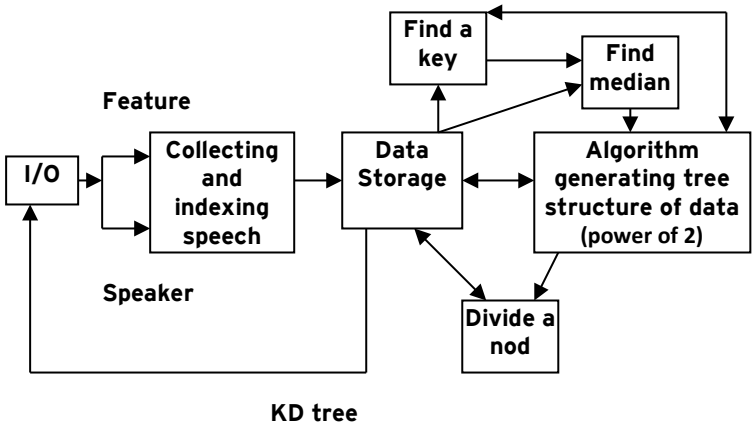

Figure 6: A block scheme of the KD tree building application

Real time speaker identification application

The application is fully configurable via 2 config files. It loads a KD tree, opens an input speech channel using WIN API 32, and starts to process the incoming wav stream. First each frame is checked if it contains speech or background noise. If speech is detected by VAD algorithm then it is transformed into MFCC features. These MFCC vectors are searched through the KD tree in order that the nearest neighbours are found. Based on the neighbours and their distances a local winner is found and its confidence is calculated. The final decision is taken using longer time period (3s) and the most probable user is found and sent to the GUI application with the confidence measure. If more hypotheses are needed more users are output ordered in a descending way (see Figure 7).

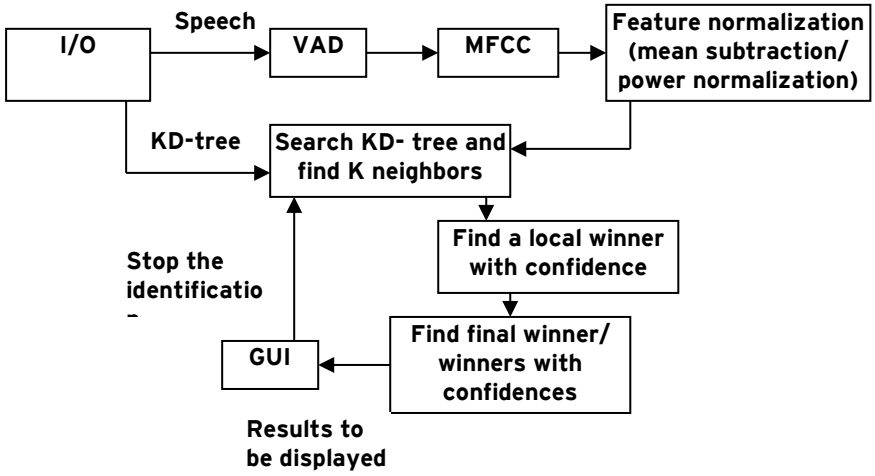

Figure 7: A block scheme of the speaker identification application

2.3.1.7. Input and Output Description

The whole GUI speaker identification application is a standalone one operating on Windows platforms (NT and higher). As an input it takes 2 config files “confg_param.txt” and “config_knn.txt”. These files basically control the behaviour of the system. However the fine tuning should be kept for a trained person. Then there is a keyboard and mouse interface to control the GUI application and finally there must be a microphone input which is accessed via calling WIN API 32 functions that read signals into the created buffers. The prime output of the application is a display that shows the currently recognized users with their confidences.

2.3.1.8. Installation Guide

Speaker identification system is an application written in several programming languages (C/C++ and Java). Because there are several system dependent libraries used in the application, the system works properly only under Windows operating system. For other operating systems it has to be recompiled and system dependent things have to be tackled. As GUI is programmed in Java, (multi-platform language), it works properly for Windows, Linux and OS X. This advantage brings also one disadvantage. Prior to the deployment of the application JDK(Java Development Kit) (if changes to the GUI are to be made) and JRE(Java Runtime Environment) have to installed. JRE and its part JVM (Java Virtual Machine) enable to run Java applications. JRE and JDK are freely available and installation of these components is very easy and one doesn't have to be skilled in this field. When installation of required components is done, speaker identification system can be installed. The application doesn't have any automatic installation. It's available only in the form of a structure of directories, which has to be copied into the file system. An example of a files structure is shown in Figure 8.

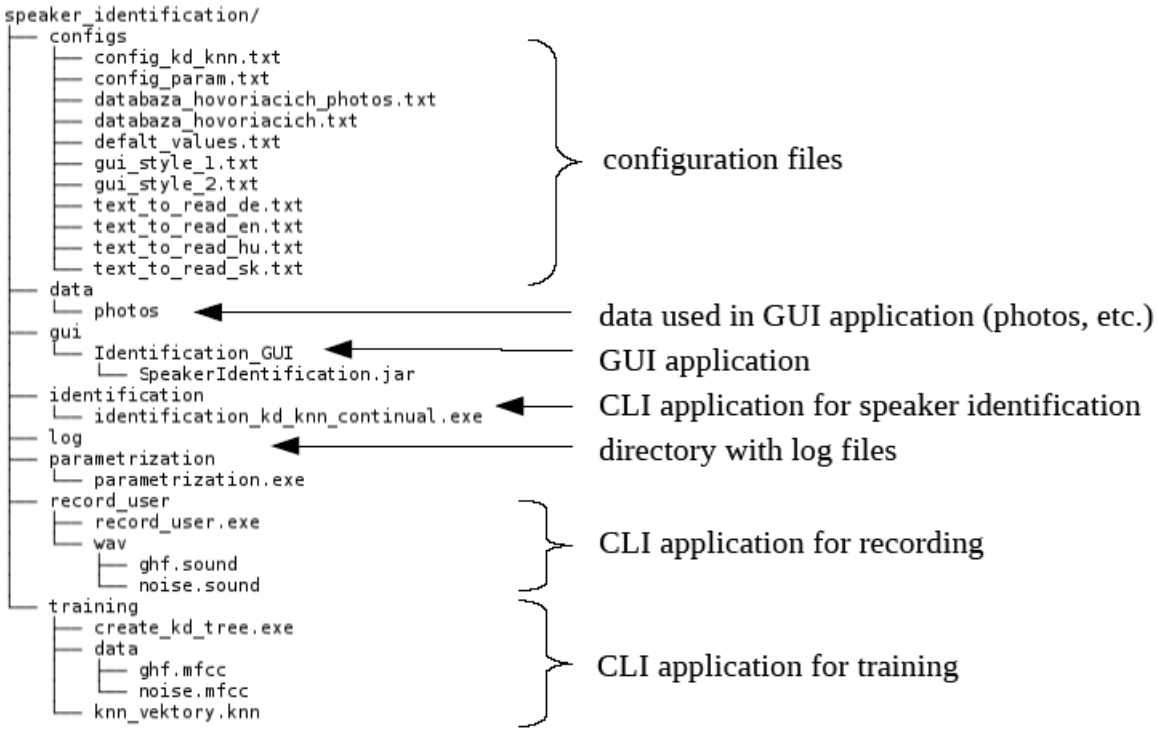

Figure 8: A File structure for the identification application.

"Configs" directory contains several configuration files. For example configuration files for speaker identification and training, configuration files mapping database of trained users, a configuration file with default values for GUI, configuration files with colour schemes for GUI and texts which must be read during the training stage. Some config files have to be manually modified when the directory structure is changed or when the processing settings are to be changed, in the case of different recording conditions. However the others are mainly modified automatically using partial CLI programs or GUI. Data directory contains images required by GUI, shadows for GUI, etc. There is also a Log directory that contains log files so that when the application stops working, problems can be easily detected. Other directories as record_user, parameterization, identification and training contain files and programs for speaker identification as well as temporal data e.g. records, parameterization files etc. It is not needed to describe them in details because their names are self-explanatory.

2.3.1.9. User Guide

When the installation of the speaker identification application is successfully done, it can finally be started be by simply clicking on an icon (jar file). When application is started initialization window is displayed (see Figure 9). During initialization process, it must be specified where configuration data are stored and where log files can be saved. When initialization process is done, main GUI is displayed (see Figure 10).

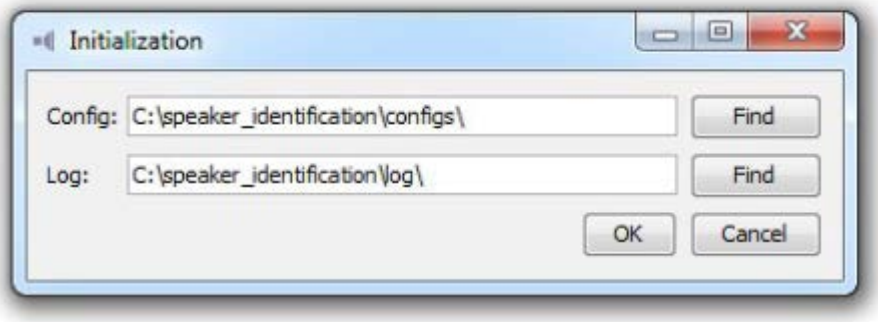

Figure 9: The initialization of the application.

Now as the application is started the training or identification process can be invoked.

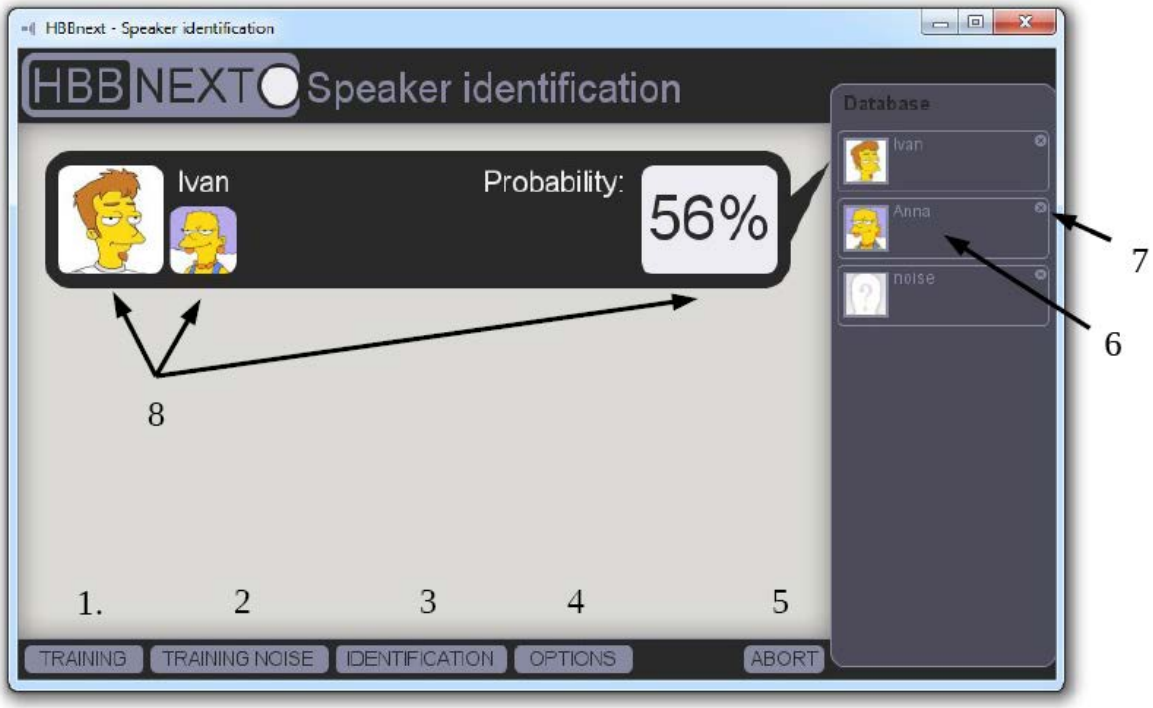

Figure 10: A graphic user interface for speaker identification.

Descriptions of options (buttons)

1. Training Button. When this option is chosen, the user is asked to enter his/her name and photo (not required). After that the training process is started (see Figure 11). As it can be seen, when the training procedure is started the text which has to be read is displayed. There are also other functions, such as changing language and increasing or decreasing the size of displayed text.

a) With English text example

b) With Slovak text example

Figure 11: The training window for a user.

2. Training Noise. It is very similar to the previous option. The only difference is that, this option is designed for a noise training. It means that except of the classical training, voice activity detection procedure is omitted. It is designed for improving detection, mainly in the noisy environment.

3. When the training is done (at least two users are in database) the identification procedure is started by this button.

4. Option button is intended to specify the location of partial applications (recording, training, parameterization and identification).

5. This button is intended to abort running operation (training or identification process).

6. On the right side of the GUI there is a list of trained users. The current view (Figure 10) shows two users and a noise sample.

7. Every single user can be simply removed using the button in the upper right corner.

8. Image, name and probability of the users are displayed during the identification process. As it can be seen, there are displayed images, where the size of the image reflects the probability of identified users. The bigger one the more likely the user is identified correctly. The number of identified and displayed users can be configured in a separate configuration file.

2.3.1.10. Conclusions

Both acoustical and prosodic features (pitch period, voicing, etc.) will be used to improve speaker recognition and VAD accuracy. Further modification of a VAD algorithm will be designed and implemented to classify speech and non-speech event with a high accuracy and low computational load. New decision taking algorithms (probably a classification fusion technique) will be suggested to meet multi speaker particularities with a possibility of detecting unknown speaker (not being recorded in the database) by employing some confidence criteria or by creating general speaker model.

2.3.2. Speech & voice command recognition

Speech signal is produced by human speech organs and is originally represented by air waves. It contains among other information lexical part that is crucial for speech recognition (speaker specific information, additional and convolutional noises as well). Every language contains a huge vocabulary, usually of several hundreds of thousands of words. To meet different settings many approaches have evolved [26] and thus some basic classifications are used: small, medium or large vocabulary systems, speaker dependent or speaker independent system, phoneme or word based system (sub-phoneme or phrases are also possible), continuous or isolated word (dictation) systems, etc.

In the case of command recognition, the systems belonging to the group of isolated word recognition would be an option. The most successful and used ones are those based on HMM statistical speech modelling, especially those using tight context dependent phonemes as a basic modelling unit. In case of a fixed set of commands and abundance of test samples, whole word models can be used for achieving potentially higher accuracies (better capturing of co- articulation effects).

2.3.2.1. Hidden Markov Model

The Hidden Markov Model [26] is a statistical modelling method for speech and more precisely for its parts (words, syllables, phonemes, sub phonemes, etc.). It is based on the concept of Markov chain that makes it computationally very effective even though not reflecting time evolution of genuine speech. Thus each model must be estimated using usually very large set of training examples that contain multiple recordings of the same word (its different realizations). HMM is defined by a priory probability vector (π) of being in particular states at the begging, transition matrix (transition probabilities between states, a_{ij}) and probability distributions (likelihoods) of generating observation vectors in a given state, $P(x/s_i)$. These distributions are not known in prior but the most widely used ones are mixtures of multidimensional normal distributions (GMM).

Then the probability of observing string of feature vectors at the model λ is given by:

$$P(\mathbf{x}_1 ... \mathbf{x}_T \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \text{ where } \alpha_t(j) = [\sum_{i=1}^{N} \alpha_{t-1}(i) * a_{ij}] * P(\mathbf{x}_t / S_j) \quad j = 1,...N$$

$$\text{and } \alpha_1(j) = \pi_j P(\mathbf{x}_1 / S_j)$$

The recognition is done be choosing the HMM model (λ) with the highest probability. A great advantage of HMM models is the possibility of concatenating several models into a string thus utterance of any length can be built up using set of basic models. In practical situation only the path with maximal probability is calculated e.g. $P(x_1,St_1,x_2,St_2,\ldots x_T St_T)$, and via backtracking the sequence of hidden states is determined (states can be related to sequence of words) using Viterbi algorithm.

The main problem with HMM is the estimation of their parameters. Usually it is done by maximizing the maximum likelihood P(x1,x2,…xT/ λ)which leads to well-known method called Baum-Welche algorithm. However newer strategies like maximal mutual information (MMI), large margin HMM, and corrective training may be beneficial in some cases, especially when the HMM model constrains do not fit the underlying physical model. The MMI model estimation criterion is defined as follows:

$$\log(P(\mathbf{x}_1,\ldots\mathbf{x}_T / \lambda_i)) - \sum_{j=1}^{P} \log(P(\mathbf{x}_1,\ldots\mathbf{x}_T / \lambda_j))$$

where λ_i is the correct HMM model according to the known transcription of the observation ($x_1,x_2,\ldots x_T$) and λ_j are all available models. This criterion aims to increase the overall separation gap among models.

Except training strategies the structure of HMM models is also important. Usually for speech units left - right models are used and the most common is probably the Bakis structure. It is commonly accepted that each acoustical state is modelled by 3 HMM states in order to capture its beginning, middle and ending part. A basic HMM model can model phonemes, syllables, words or even whole phrases. It is a trade-off between accuracy of modelling and computational feasibility and availability of the training data. Some frequent and usually short words can be modelled by a single HMM model. A good balance is achieved by so called tied triphones models which are used by most of the employed systems. Whatever training criterion is used there are no close formulas for optimal parameters thus several iteration cycles must be applied in a controlled way. Unfortunately only local maxima are guaranteed to be found. Following the classification theory HMM models are also prone to be over-fitted, thus a validation set must be used to detect and prevent this phenomenon to take place.

The model complexity, modelled speech units and training strategies depend on the particular application, and are mutually adjusted in order to achieve the best trade-off between the accuracy and robustness.

Except speech modelling there is still the task of speech extraction. The extracted features should capture the lexical information while suppressing the other ones (noises and supports inter and intra speaker variability). The most successful techniques are MFCC and PLP and their modifications, but still some experimental ones are emerging. As MFCC was briefly described in the single speaker identification module please refer to it. Additional features may include signal dynamic observed via delta and acceleration coefficients constructed over acoustic features showing their time evolution as well. They are defined as:

$$\Delta(n) = m\sum_{k=-L}^{L} kc(n+k) \quad and \quad \Delta\Delta(n) = m\sum_{k=-L}^{L} k\Delta(n+k)$$

2.3.2.2. Theoretical Basis- research activities and achievements

The focus of the research in the area of speech/ command recognition has been in two directions. The first is in the HMM model training phase where new modifications of the training process and used speech features have been suggested and evaluated. The second is to optimize the recognition phase (speech decoding process) i.e. to speed it up and to preserve a good accuracy figure.

We tested MFCC and PLP methods [26] for deriving eligible speech features. Based on these tests we observed slightly better results in the case of PLP method. In addition, cepstral mean normalization was also beneficial with differential energy information. To distinguish between voiced and unvoiced phonemes a simple voicing feature was also incorporated into a final feature vector. It was defined as a ratio of the averaged AMDF function to its minima found within the range of possible pitch periods. This has been a beneficial step that recorded substantial improvements in the tests scenarios (application words, digits), see Figure 12.

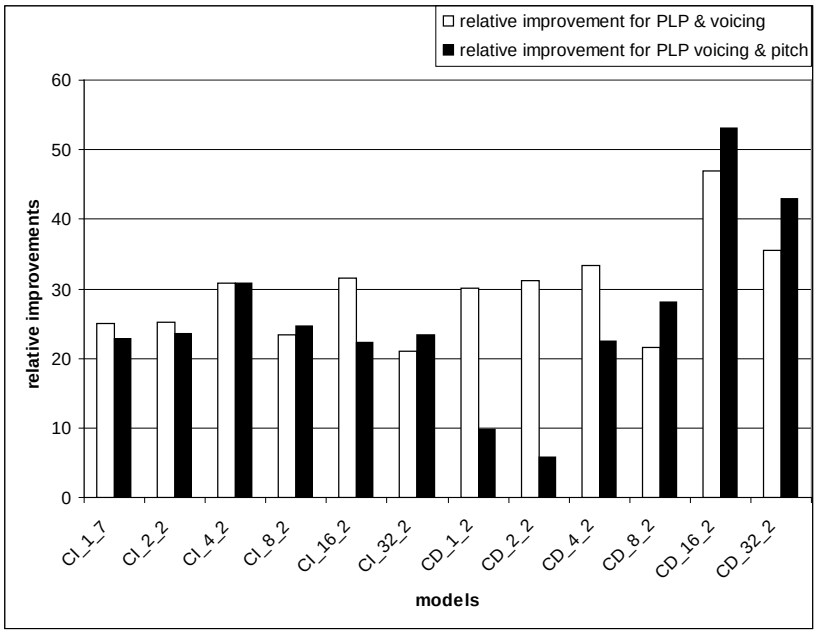

Figure 12: Relative improvements of word error rates after adding voicing and pitch information to the original PLP

Further, in the training phase a well know multilingual reference recognition system – MASPER [27] was used. Some of its free parameters have been subject to the optimization for the Slovak speech database. The most conspicuous change has been the introduction of a HMM model for garbled speech, so called BH model. In doing so more recordings can be involved in the training process that turns out to provide more reliable training as well as more accurate ones and more context dependent phonemes exist (see Figure 12).

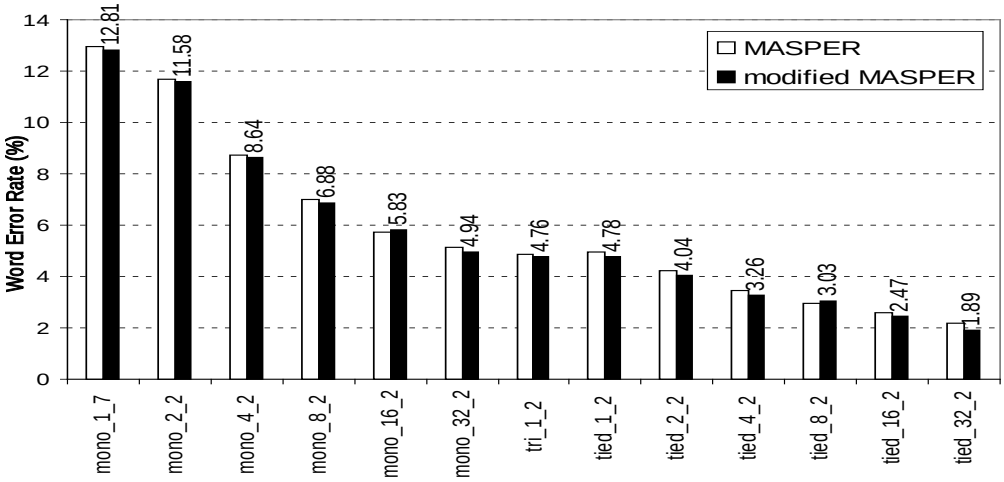

Figure 13: The word error rates for the original and modified MASPER training procedure, for SVWL test and tied CD models of phonemes

Other improvements are related to the changes in the dictionary transcriptions- the existence of more pronunciations.

Finally we optimized a simple command (frame grammar) real time speech recognition application based on ATK system. That involved approximately 10 of its vital free parameters, like: speech detection settings, insertion penalties, tree pruning during the Viterbi search algorithm, etc. by using methods of evolutionary strategies, see Figure 14.

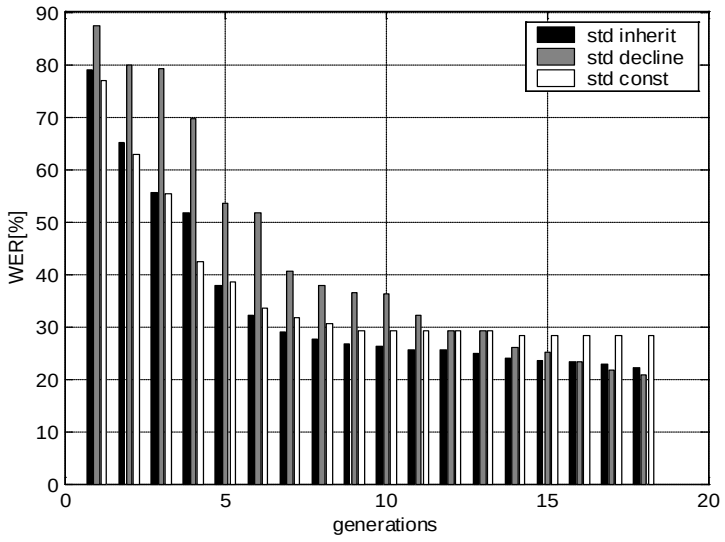

Figure 14: The mean word error rates per generations for gradually declining, inherited and constant standard deviation

By doing so we were able to come up to better results than those archived when parameters were set by experts; more than 5% improvement in word error rates.

2.3.2.3. Algorithms, Input and Output description, Installation Guide, User Guide

No demo will be provided for this application, so there is no deployment of an exact algorithm, thus there is neither User nor Installation Guide.

2.3.2.4. Conclusions

For a given voice command based application HMM technology with context dependent triphones will be used with an option for modelling very frequent and short words as a whole. Then the task will be to use proper training strategy (ML, MMI, MCE, etc.) as well as additional transforms to better model full covariance matrices.

Another course of research would be represented by selecting and modifying extracted features that would fit the environment as well as the HMM models and training strategies.

2.3.3. Gesture Navigation

Gesture recognition can be conducted in two manners. Either a data glove is used which transforms the body flexions into movement information, or vision-based approach is applied where a camera serves as a human eye to record body positions which are then extracted using image processing. It is clear that while the first method might bring precise results, it is rather uncomfortable in terms of user convenience.

Also, equipment needed to employ the method would be unacceptably costly for most of standard customers making it only suitable for special use. The latter approach, on the other hand, has no other equipment requirements for the end user (except for the camera), making it suitable for general applications. The drawback, however, lays in algorithmic complexity where considerable amount of time and computing power is required to extract body movements.

There are various algorithms available which focus on different aspects of the gesturing person (and take different assumptions). Generally, they can be divided into two categories, appearance- and 3D model-based approaches. The 3D model-based approach compares the input parameters of a limb with 2D projection of a 3D limb model. The appearance-based approach uses image features to model the visual appearance of a limb and compares it with extracted image features from the video input.

When focusing on the latter approach, the results depend on the capabilities of the capturing device. If an RGB camera is used the methods focus on tracking the skin colour or shape of the gesturing body part. The approach, however, depends highly on the lighting conditions, as well as the stability of foreground and background of tracked subjects. Also, no other skin-coloured or limb-shaped objects can appear in the examined area as they would trick the algorithm. An infra-red light depth camera uses its own IR light emitter and is thus much more resilient to lighting conditions of the scene. Moreover, the camera is capable of providing a depth map, a pseudo 3D image of the scene which can be very useful when tracking gesturing body parts, i.e. hands.

2.3.3.1. Algorithms

The Kinect device was introduced to the market in November 2010 by Microsoft. Primarily, it was designed for the XBOX360 gaming console where it serves as a contactless controller. Later Microsoft responded to the needs of the market and launched a second version of the device, the Kinect for Windows, in February 2012, which is primarily designed for commercial purposes.

Kinect sensor returns the distance as depth data for each captured pixel with the depth camera resolution of 640x480 pixels.

Figure 15 shows the depth data rendered as an RGB image after normalization and conversion of distances to RGB elements of the moving scene. Character shooting with depth camera has considerable advantages over using RGB camera, because the depth camera does not depend on light conditions. The only limiting factor for the shooting by depth camera is direct sunlight because it affects the IR sensor by interference with IR light from the sensor's emitter. When using an RGB camera it is necessary to modify the brightness, contrast and other parameters that affect image quality. Using the depth camera these corrections are not needed.

Figure 15: Depth camera image

2.3.3.1.1. First Algorithm

At first we decided to use a method that uses convexity defects. Convexity defects can be applied in whole image processing. Naturally, lonely methods are not designed for finger detection but we will modify this method for our purpose. The algorithm is described in steps below.

Finding the contours of the hand

For faster processing of data, we cannot process the whole image area. After we automatically detect the hand we allocate the area around the hand which is then only processed. When we convert obtained distances into the RGB image, we get the contour of our hand on a black background (Figure 16). This aspect also should be taken into consideration for the best calculating. It is mainly due to the fact that only the used area is processed, rather than the whole picture.

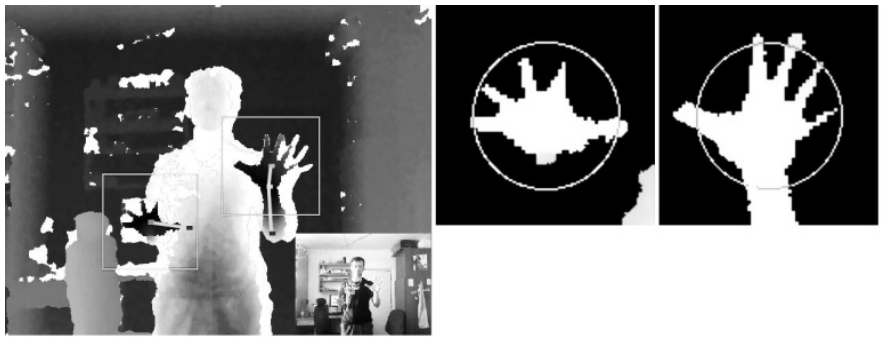

Figure 16: Detected hands are segmented from the main picture in order to minimize image processing complexity

Finding the middle point of the hand

Finding the coordinates of the hand using implemented algorithm in Kinect is relatively inaccurate because large differences occur between the opening and closing palm. Therefore, our own method was used for finding the middle point of the palm. We found the hand contour in the previous step. Now, this knowledge will help us find the centre contours - which are ultimately the centre of the palm, no matter whether it is an open or closed hand.

Calculating convexity defects

To find and calculate the convexity defects we used the GetConvexityDefacts method of emguCV. Figure 17 shows convexity defects (white colour). There are two methods to find convexity defects: clockwise method and counter clockwise method. They return the coordinates of three points (START_POINT, DEPTH_POINT and END_POINT, representing the starting point, the deepest point and end point) where the deepest point is understood as maximum distance between the hull and hand contour. Figure 17 shows one of the convexity defects, however, the method computes all defects.

Figure 17: START_POINT (left), DEPTH_POINT (bottom) and END_POINT (top)

Convexity defects method is not primarily designed for recognition of the number of fingers on hand but it can be applied for any object, if the method is applied properly. Therefore it is necessary to adapt its functionality so that it recognizes only the fingers and it does not consider the whole area of hand. When we use the method without modifications, the algorithm finds many defects and cannot recognize if the defect represents a finger or not. Additionally, this raw method finds more defects on the hand with closed palm than when the palm is opened.

Finding the point on the finger that is the farthest from the centre of hand

For finding the point on the finger that is the farthest from the centre of hand we reverse the order of the whole contour points since the contour points are placed in a list. The algorithm starts in point START_POINT and the MAX_POINT has the same value as START_POINT at the beginning. Then we sequentially move to the next coordinate until it finds a coordinate that is the farthest from the centre of hand. The distance from middle point of hand to MAX_POINT is compared at each step, while there is farther point from the centre point than in previous point.

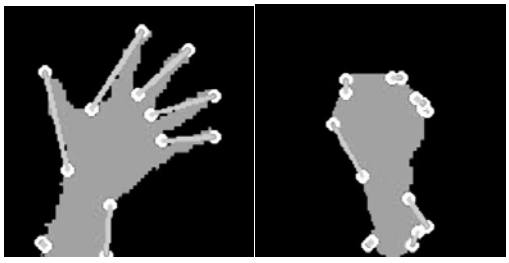

Figure 18: Finding the maximum distances between contour and depth points

Removal of defects whose triangle height is less than the specified

Since we found many defects on the hands, including those which do not represent fingers and thus are not necessary, we must remove all these defects. The first step is to remove all defects whose height is less than a specified value. This value is dynamically changing according to the size of area where the hand is detected.

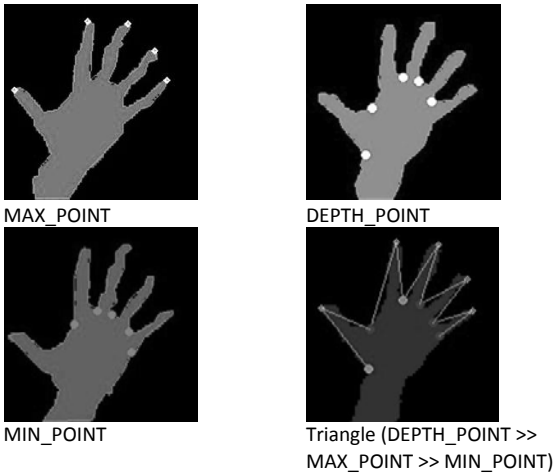

Figure 19: To illustrate the logic behind using triangles to eliminate unnecessary defects

Removing of defects that have the distance MAX_POINT – START_POINT longer than the specified

The second step of filtering is to remove the defects that have a distance between START_POINT and MAX_POINT more than a specified value, which is also dynamically changing (if two points are too far, it means that they cannot represent a finger).

Removing of defects that have the distance START_POINT – DEPTH_POINT less than the specified

The third step of filtering is to remove also the defects that have a distance between START_POINT and DEPTH_POINT and less than a specified value. This value changes dynamically according to the size of the region where the hand is detected.

Remove points that are below the wrist

The last step is to remove all the defects that occur below the wrist. We will find a way to join together two points, HAND and WRIST (provided by the Kinect API). This straight line is rotated 90 degrees left or right, eventually. All defects that arise on the side which does not contain coordinate of HAND will be removed.

2.3.3.2. Second Algorithm

First algorithm had relatively good results but sometime it has got some problems with various rotations. This method loses tracking of fingers when the resolution of hand area is not sufficient. We found a new method that looks to be better and more robust.

This method works with hand contour where template matching algorithm is applied. The algorithm tries to find the least angle between two points of each finger. If there exist three points with minimum angle we will consider middle point as a fingertip. It is very possible that more than one fingertip will be founded. Then we applied a filter to choose only the best one point of fingertip. The filter chooses the point with the greater distance from the centre of palm.

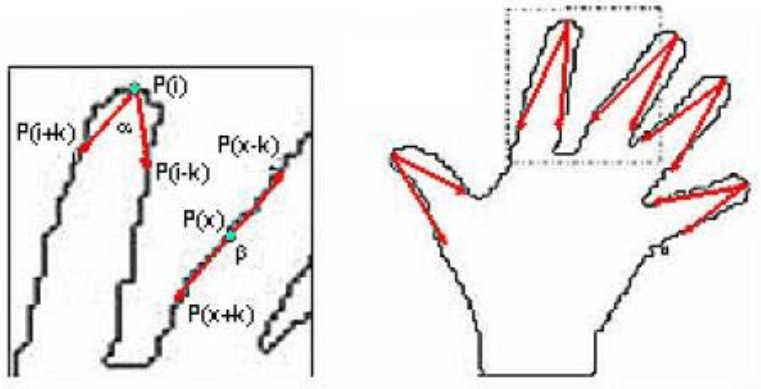

Figure 20 Template matching algorithm

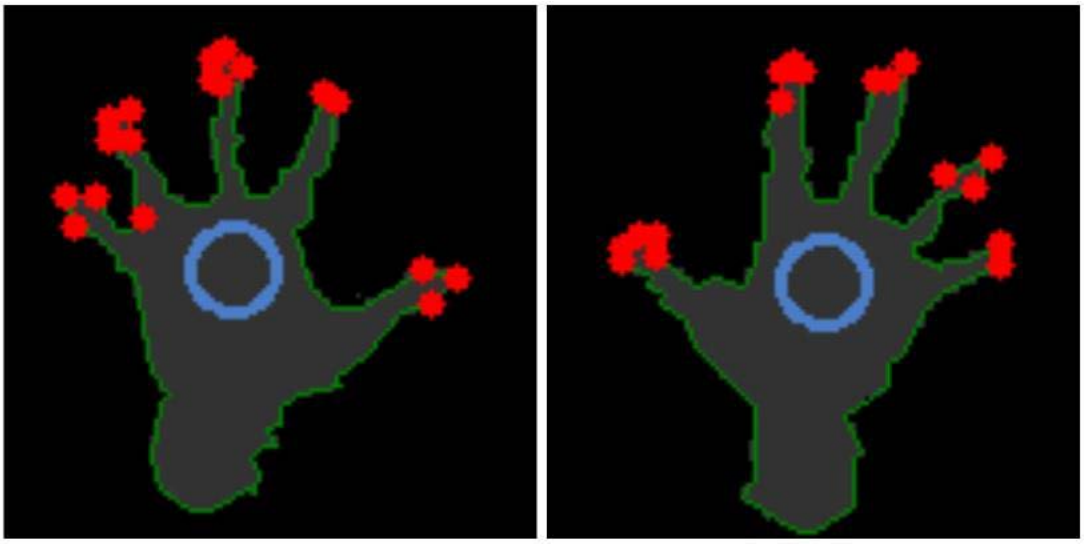

Figure 21 Possible tip before using the filter

The filter causes that each finger has just one tip marker. This method has better results and is more robust than first method used in past.

2.3.3.3. Inputs and outputs

Inputs

Input is raw data from the Kinect sensor. This application uses only depth data from sensor, which are represented by depth stream.

Outputs

Application provides XML output, where are stored data about count of detected fingers of right or left hand. The XML file structure is below.

```
<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<application>VOICE_RECOGNITION</application>
    <users>
        <user>
            <name></name>
            <probability></probability>
            <command>left_hand:4;right_hand:3</command>
            <presence></presence>
        </user>
    </users>
```

where number next to "left_hand:" and "right_hand:" means number of fingers.

2.3.3.4. System requirements

Hardware Requirements

- CPU: Intel DualCore (optimal Intel i3,i5 series)
- RAM: 4GB(optimal)
- VGA: 128MB (optimal)
- ETH: Ethernet connection
- Sens: Kinect Sensor

Software Requirements

- WIN 7 pro (optimal)
- .NET Framework v.4 and higher
- Microsoft Kinect SDK v1.6 and higher

2.3.3.5. Installation steps

1. Be sure that .NET Framework and Kinect SDK are installed.
2. Copy files of Gesture Recognition onto disk.
3. Open exe file.

2.3.3.6. User guide

This application sends data about hand to specific IP address. Data are represented as XML file containing this structure:

```
<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<application>VOICE_RECOGNITION</application>
    <users>
        <user>
            <name></name>
            <probability></probability>
            <command>left_hand:4;right_hand:3</command>
            <presence></presence>
        </user>
    </users>
```

where number next to "left_hand:" and "right_hand:" means number of fingers.

The file is sent every time when the number of fingers is changed. Specific IP address can be changed in block of source code.

2.3.3.7. Conclusion

In this part of HBB-NEXT project we have focused on improving a finger detection algorithm. The old algorithm has a problem with noise and tracking of identified fingers and therefore we have made the new algorithm more resistant to noise and more accurate to calculate number of fingers than the old algorithm.

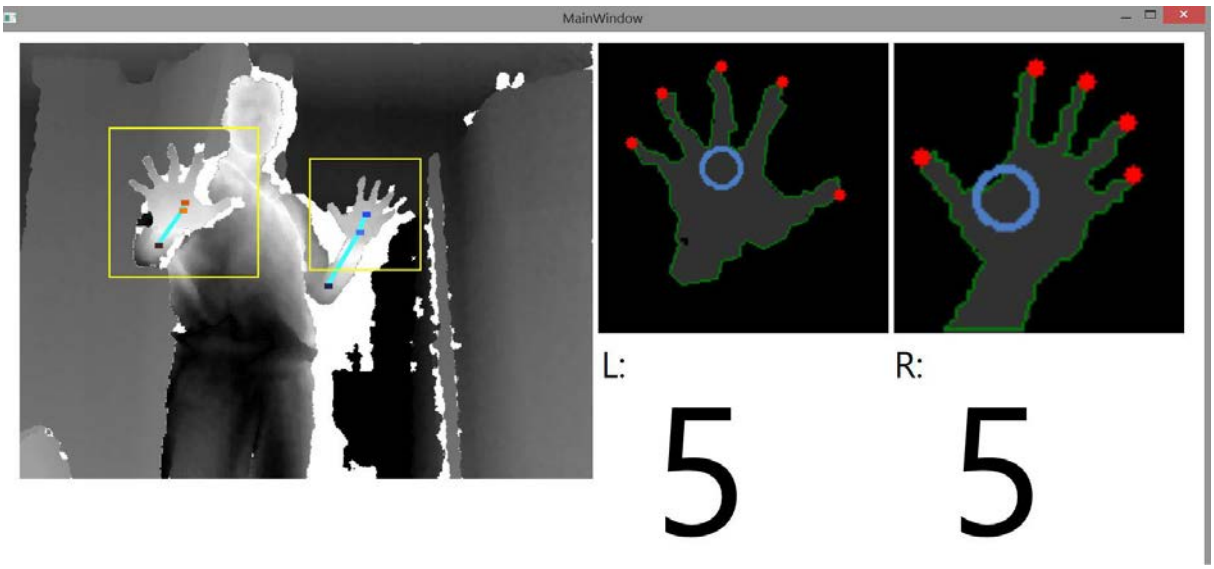

Figure 22: User interface of the new tested application

The recognition works well up to the distance of 2 metres from sensor. Greater distances provide an image of hand with limited pixel count due to native (VGA) resolution. The application is currently able to recognize the number of shown fingers in real time, 30 frames per second, while processing both hands.

Methods based on RGB camera image processing depend highly on lighting conditions and background colour and dynamics. Thus, most of the solutions only work as long as the specified conditions are met almost exactly. This is, however, impossible to achieve in a normal user environment. Our method has proved its independence from background and illumination - only strong direct sunlight has forced the sensor not to respond. Moreover, gaining from the Kinect capabilities, the hand can be rotated in any angle as long as it faces towards the Kinect sensor.

Already we have started working on a dynamic gesture, where we would like to make the application recognize dynamic gestures and record some new gestures defined by user. Nowadays we started to work on testing some algorithms because we would like to connect a touchless idea with well-known touching algorithms together. User will draw his own gesture by hand onto an imaginary canvas and the application will recognize this trajectory as gesture.

There is an on-going research yielding to implementation of a gesture recognition system based on a pre-defined database. We plan to build a more sophisticated system which will recognize not only static gestures but also dynamic gestures and their combination as well. Eventually, the system will decide the meaning of a gesture also from movement characteristics and starting and ending position of the hand.

2.3.4. Eye Navigation

Currently, there are few methods how eye controlling can be implemented. We choose the simplest and the most natural way. This system has only one part. It is the static RGB camera, which is available commonly in every laptop. The cameras in laptops usually have different image resolutions. It cannot be said that higher resolution means better camera. Picture quality depends on the size of camera lens and sensor. From our experience, laptop web cameras don't have very good picture quality, but with the help of special image filters it can be useful for our application. In our case, better camera definitely means better and more accurate results.

2.3.4.1. Algorithm

As a middleware for Kinect device we have used both Open NI and the official MS Kinect SDK. For image processing we use the Egmu CV library. This library is in basic wrapper for the popular Open CV library which provides compatibility with .NET framework.

Pupil detection with CDF analysis

The region of interest (ROI) consists of those areas that contain the eyes. If we find this region, we can reduce the computational cost by reducing the search space. We can find the region of interest by dividing the face region vertically into top and bottom parts. Then the top part is partitioned horizontally into left and right segments. At this stage we have two regions where each one has an eye.

The centre of pupil is detected by an adaptive approach. After finding the ROIs, that each one contains an eye, the following approach is performed on both regions. At the first step we filter each ROI by the cumulative distributed function (CDF) of that region.

We find the CDF by integrating the histogram of each of the ROIas follows:

$$CDF(r) = \sum_{w=0}^{r} P(w)$$

where P(w) is the histogram representing the probability of occurrence of gray level w and 0 ≤ r ≤ 255. It was found that the pixels of eyelid and pupil have CDF ≤ 0.05, so ROI is filtered by:

$$I'(x,y) = \begin{cases} 255; CDF(I(x,y)) \le 0.05 \\ 0; \quad \text{otherwise} \end{cases}$$

where I(x,y) is the original eye region and I‘(x,y) is the image that only contains the pixels of pupil and upper eyelidand those parts with a CDF less than 0.05. We can see the result of this section in Figure 23a.

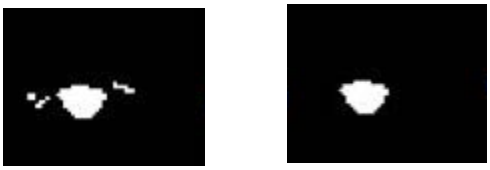

Figure 23: a) Image after CDF b) Image a) after erosion morf. oper.

The result of the first part contains regions rather than pupil. We perform morphology to remove these parts. The erosion morphological operation by 2×2 structure element is a suitable candidate for this purpose. Figure 23b) illustrates the outcome of this step which is approximately the pupil.

In the other approach we use a Kinect RGB camera. The person sitting in the front of the kinect (monitor) will be asked to look with its eyes to the highlighted points on the screen with the still head. Simultaneously the application by kinect depth camera measures the head distance from the kinect (monitor). For the purpose of the triangle calculation we will determine the dimensions of the screen. Application will calculate the variance of the pupil movement exactly the same like in section calibration and also it will calculate the angles by which the pupil must be diverted from straight position to see the edge of the monitor. After knowing the head distance from monitor we can recalculate the variance and angles when the distance is changed to ensure the accuracy of the controlling.

2.3.4.2. Conclusion

Our experiments have shown that the methods are highly dependent on the quality of illumination of the environment. The algorithm works properly only under certain lighting conditions which are difficult to maintain for longer periods of time. Additionally, sensors used in the system are primarily constructed with insufficient image resolution in comparison to size and distance of human eyes from the sensor.

2.3.5. Speech Synthesis

Speech synthesis is an important area of multimodal interface development. Duplex speech communication creates a user friendly environment. Speech commands are usually used by users to control features of their television. Display opportunities should be extended with speech reaction. Speech is mainly required when the user is not in eye contact with display. Various actions can cause this situation. Other scenarios are those, where video stream cannot be interrupted but HbbTV needs to bring some message to the user, e. g.: “You have one new mail” or “Go out for a walk, you need fresh air and movement”.

Speech synthesizer is a device that provides voice communication. From wide range of speech synthesis kinds e.g.: formant, sinusoidal, HMM, etc. we use concatenative speech synthesis. We use a database with units cut from recorded human speech. Speech units differ from each other in length e. g.: phonemes, diphones and triphones. We use diphones. The larger units provide better quality but the disadvantage is that the database is larger.

Speech synthesis has indispensable role in the field of information technology development. Its importance lies in opening new possibilities for human communication with the information system. This is related to its openness to a wide range of people, no age limits, education or social status. Intuitive user interface is very important. Speech synthesis has already achieved many good results in various applications. Qualitative improvements were supported by development in related areas.

2.3.5.1. Algorithm

Speech synthesis for Slovak language is built on a modular architecture. This solution has many advantages, e. g.: depending on the demand it can be chosen which module is used. Inter-modular data exchange of partial results is based on XML RPC protocol. Data are transmitted and encapsulated in packets mainly in our internal XML standard. The whole process is controlled by a supervision node in a hub topology. This solution is currently limited to the Slovak language. However, the solution can be applied to other language speech synthesis. Modular speech synthesizer is developed as a stand-alone demo. This application can be integrated to the HbbTV. A unique interface is required for this purpose.

2.3.5.2. Phonetic Transcription

Phonetic transcription is a key speech synthesizer functionality, which is responsible for the transcription of text to phonetic alphabet. As one of the basic characteristics for the classification of artificial speech, correct reading of the text depends precisely on qualities of this functionality. The correct interpretation of phonetic transcription of words is an extremely complex fact and it was introduced in previous paragraph. There are not linear relations between object, its name and its phonetic form. It should be noted that the speech synthesizer is usually applied in a dynamic environment. Tools of phonetic transcription have to be flexible and well optimized. Phonetic transcriptions are usually written in the International Phonetic Alphabet (IPA), in which each English sound has its own symbol.

2.3.5.3. LTS Rules

Rulesthat we use for text transcription into phonetic alphabet SAMPA are called LTS (Letter-To-Sound) rules. The rules work with letters. Text in general contains entities such as sentences, phrases, words, syllabus, and letters. It is important to consider three entities for rule-based transcript: text, word, and letter. LTS rules are represented by classification trees. Classification trees are part of CART (Classification And Regression Trees). CART represents a combination of rule-based knowledge and is specified based on statistical methods.

Rules for our speech synthesizer are written in Lisp and stored in file. Classification-type problems are generally those where we attempt to predict values of a categorical dependent variable (class, group, membership, etc.) from one or more categorical predictor variables. In most general terms, the purpose of the analyses via tree-building algorithms is to determine a set of if-then logical (split) conditions that permit accurate prediction or classification of cases [23].

Systematic relation is between written form (orthographic) and pronunciation (orthoepic) only in some languages [24]. There may be some cases, when rules can be simply handwritten. Of course, there is an opportunity to create set of rules automatically. This way is often the most effective. The choice depends on a language and its orthographical character set and phonemes character set. In the case of Slovak language, simple handwritten rules cannot cover a large number of languages. However, handwritten dictionary of exceptions from phonetic view has great importance for our research and development. Phonetician determines them according to research. Abel Kráľ and Sach Daržagín have done such research for Slovak language.

Creating rules automatically starts with creation of database. The database is composed of training vectors. Each vector has information about how some letter is read. We can create as many training vectors from one word, as many letters this word has. It is because one training vector has information about one SAMPA character and each SAMPA character has to belong to exactly one grapheme. This balance is reached in grapheme to phoneme alignment (G2P). One vector belongs to each letter. Finally, information about pronunciation is encoded into the sequence of phonemes. New LTS rules are built automatically and depend on vectors in database.

2.3.5.4. LTS rules and Dictionary in Classification role

The advantage of dictionary is mainly its scalability. Words can be added or removed freely according to the need from the dictionary. These adjustments have no impact on the other words in the dictionary. In contrast, LTS rules cannot be partially modified, so they cannot be raised or reduced. Modifications of the structure of binary classification trees are technically possible, but could easily lead to irrelevant or incorrect changes that could bring systematic errors in the transcription process.

The rules are generated by taking into account any linguistic phenomenon, with statistical sampling only those which meet the conditions. Our LTS rules in its final form contain no longer information about how many sample based vectors contributed specific node, or the likelihood of the decision for the respective SAMPA character. We cannot Change LTS rules ones they are finished, because we don't regard the original corpus information. A correct approach is to modify corpus and then regenerate whole LTS rules.

2.3.5.5. Conclusion

Basic functionality of speech synthesis can be implemented as an entity of words and phrases or voice commands. This way it is possible to ensure support for foreign languages in relatively short time. The main advantage of this solution is naturalness of speech.

2.3.5.6. Future work

The question is, in which scenario artificial speech is needed. As it was afore mentioned, it is in time, when user should be noticed. User should be noticed either according to his own personalized settings e. g.: five minutes before news, or after too long period watching TV, or before some special scenes that are very funny or scary, violent, obscene or inappropriate for children. TV can attract attention the chance of scoring in soccer. In the case, when children watch TV immediately after they come home from school, TV can ask them, if their homework is finished.

3. References

[1] DublinCore: http://dublincore.org

[2] ETSI TR 102033: Architectural framework for the delivery of DVB-services over IP-based networks: http://www.etsi.org/deliver/etsi_tr/102000_102099/102033/01.01.01_60/tr_102033v010101p.pdf

[3] ETSI TS 102034: Transport of MPEG-2 TS Based DVB Services over IP Based Networks: http://www.etsi.org/deliver/etsi_ts/102000_102099/102034/01.04.01_60/ts_102034v010401p.pdf

[4] ETSI TS 102814: Ethernet Home Network Segment: http://www.etsi.org/deliver/etsi_ts/102800_102899/102814/01.01.01_60/ts_102814v010101p.pdf

[5] ETSI TS 102822: Broadcast and On-line Services: Search, select, and rightful use of content on personal storage systems ("TV-Anytime"); Part 3: Metadata; Sub-part 1: Phase 1 - Metadata schemas: http://www.etsi.org/deliver/etsi_ts/102800_102899/1028220301/01.04.01_60/ts_1028220301v010401p.pdf

[6] YouTube: www.youtube.com

[7] YouTubeGData: YouTube GData API, http://code.google.com/apis/youtube/2.0/reference.html and http://code.google.com/apis/youtube/developers_guide_protocol.html

[8] TVAnytime Forum: http://tech.ebu.ch/tvanytime/

[9] MPEG, MPEG-7 Overview, Oct 2004, http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm

[10] MPEG-21 DIDL Format: http://xml.coverpages.org/MPEG21-WG-11-N3971-200103.pdf

[11] DBPedia: http://www.dbpedia.org

[12] iMDB: http://www.imdb.com

[13] FreeBase: http://www.freebase.com

[14] Uberall2012: Phd Thesis - Christian Überall – A Dynamic Multi-Algorithm Collaborative-Filtering System – 2012

[15] ZXing: http://code.google.com/p/zxing/

[16] XMPP: http://xmpp.org

[17] RabbitMQ: http://www.rabbitmq.com

[18] RabbitMQ javascript: http://www.rabbitmq.com/blog/tag/javascript

[19] OpenStack: http://www.openstack.org

[20] Deliverable D6.1.1 - Initial Version of the HBB-NEXT System Architecture : http://www.hbb-next.eu/index.php/documents

[21] Deliverable D3.2 - DESIGN AND PROTOCOL (High Level Architecture): User ID, Profile, Application Reputation Framework: http://www.hbb-next.eu/index.php/documents

[22] Restlet: http://www.restlet.org

[23] StatSoft, Inc. (2010): Electronic Statistics Textbook, Tulsa, OK: StatSoft. Accessible: http://www.statsoft.com/textbook/

[24] Black, A., W., and Lenzo, K., A.: Building Synthetic Voices, Language Technologies Institute, Carnegie Mellon University, Sunday, 21st January, 2007, Lexicons. Accessible: http://festvox.org/bsv/x1469.html

[25] Deliverable D3.3.1 - DESIGN AND PROTOCOL: Intermediate User ID, Profile, Application Reputation Framework: http://www.hbb-next.eu/index.php/documents

[26] Deliverable D2.4 - Description of the Selected Business Mode, http://www.hbb-next.eu/index.php/documents

[27] Deliverable D5.4.1 - EVALUATION: Intermediate Context-Aware Personalised Multi-User Content Recommendation Engine, http://www.hbb-next.eu/index.php/documents

4. Annexes

A. GroupContext Server API specification

The methods that are provided by the Group Context Server are specified below.

I. Get registered applications

Get all applications that registered themselves at the Group Context Server.

URL: /GroupContextService/<version>/apps

Method: GET

II. Unregister all applications

Unregister all applications that are registered at the Group Context Server.

URL: /GroupContextService/<version>/apps

Method: DELETE

III. Register application

Register an application with a specific application ID.

URL: /GroupContextService/<version>/apps/<appId>

Method: PUT

IV. Unregister a specific application

Unregister an application with a specific application ID.

URL: /GroupContextService/<version>/apps/<appId>

Method: DELETE

V. Register user

Register a user with a specific user ID at the specified application.

URL: /GroupContextService/<version>/apps/<appId>/users/<userId>

Method: PUT

VI. Unregister user

Unregister a user with a specific user ID from the specified application.

URL: /GroupContextService/<version>/apps/<appId>/users/<userId>

Method: DELETE

VII. Get registerd users

Get a list of all users that are currentlu registered at the specified application.

URL: /GroupContextService/<version>/apps/<appId>/users

Method: GET

VIII. Wait for group context change

Issue a long polling request, waiting for a group context change event.

URL: /GroupContextService/<version>/apps/<appId>/context

Method: GET

B. Recommendation Engine API specification

The methods that are provided by the Recommendation Engine are specified below.

I. Get recommendations

URL: /RecEngine/<version>/<party>/recommendations

Method: GET

Parameters

g	Group, specified by their foreignUserIds (comma separated)
s	Size, the maximum number of items to return
rc	RecommendationContext paremeters, specified as “<tag>:<value>” (comma separated)

II. Get content-content recommendations

URL: /RecEngine/<version>/<party>/recommendations/similarto/<foreignItemId>

Method: GET

Parameters

g	Group, specified by their foreignUserIds (comma separated)
s	Size, the maximum number of items to return
rc	RecommendationContext paremeters, specified as “<tag>:<value>” (comma separated)

III. Get similar content

URL: /RecEngine/<version>/<party>/similarto/<foreignItemId>

Method: GET

Parameters

g	Group, specified by their foreignUserIds (comma separated)
s	Size, the maximum number of items to return
rc	RecommendationContext paremeters, specified as “<tag>:<value>” (comma separated)

C. Preference Service API specification

The methods that are provided by the Preference Service are specified below.

I. Get (selected) items

URL: /PrefService/<version>/<party>/items

Method: GET

Parameters

q	Query used to filter the items, represented as “<tag>:<value>”, where <tag> and/or <value> can be *
ps	PageSize, the maximum number of results to return
p	Page, the page to return
e	Expand, the properties to expand (comma separated)

II. ***Get specific item***

URL: /PrefService/<version>/<party>/items/<foreignItemId>

Method: GET

Parameters

e Expand, the properties to expand (comma separated)

III. ***Create items***

URL: /PrefService/<version>/<party>/items

Method: PUT

Body

```
<items>
     <item>…</item>
     <item>…</item>
     …
</items>
```

IV. ***Create single item***

URL: /PrefService/<version>/<party>/items/<foreignItemId>

Method: PUT

V. ***Delete items***

URL: /PrefService/<version>/<party>/items

Method: DELETE

Body

```
<items>
     <item>…</item>
     <item>…</item>
     …
</items>
```

Delete single item

URL: /PrefService/<version>/<party>/items/<foreignItemId>

Method: DELETE

VI. Get item characteristics

URL: /PrefService/<version>/<party>/items/<foreignItemId>/characteristics

Method: GET

Parameters

q Query used to filter the item characteristics, represented as "<tag>:<value>", where <tag> and/or <value> can be *

ps PageSize, the maximum number of results to return

p Page, the page to return

VII. Add/Update item characteristics

URL: /PrefService/<version>/<party>/items/<foreignItemId>/characteristics

Method: PUT/POST

Body

```
<characteristics>
    <characteristic>
        <tag>…</tag>
        <value>…</value>
    </characteristic>
    …
</characteristics>
```

VIII. Delete item characteristics

URL: /PrefService/<version>/<party>/items/<foreignItemId>/characteristics

Method: DELETE

Parameters

t Tags, the characteristics with these tags should be deleted (comma separated)

IX. Get item aliases

URL: /PrefService/<version>/<party>/items/<foreignItemId>/aliases

Method: GET

X. Add/Update item aliases

URL: /PrefService/<version>/<party>/items/<foreignItemId>/aliases

Method: PUT/POST

Body

```
<aliases>
     <alias>
          <foreignid>…</foreignid>
          <otherparty>…</otherparty>
          <otherforeignid>…</otherforeignid>
     </alias>
     …
</aliases>
```

XI. Delete item aliases

URL: /PrefService/<version>/<party>/items/<foreignItemId>/aliases

Method: DELETE

Parameters

p Parties, the partyId's for which the aliases must be deleted (comma separated)

XII. Create users

URL: /PrefService/<version>/<party>/users

Method: PUT

Body

```
<users>
     <user>…</user>
     <user>…</user>
     …
</users>
```

XIII. Create single user

URL: /PrefService/<version>/<party>/users/<foreignUserId>

Method: PUT

XIV. Delete users

URL: /PrefService/<version>/<party>/users

Method: DELETE

Body

```
<users>
     <user>…</user>
     <user>…</user>
     …
</users>
```

XV. Delete single user

URL: /PrefService/<version>/<party>/users/<foreigUserId>

Method: DELETE

XVI. Get user aliases

URL: /PrefService/<version>/<party>/users/<foreignUserId>/aliases

Method: GET

XVII. Add/Update user aliases

URL: /PrefService/<version>/<party>/users/<foreignUserId>/aliases

Method: PUT/POST

Body

```
<aliases>
    <alias>
        <otherparty>…</otherparty>
        <otherforeignid>…</otherforeignid>
    </alias>
    …
</aliases>
```

XVIII. Delete user aliases

URL: /PrefService/<version>/<party>/users/<foreignUserId>/aliases

Method: DELETE

Parameters

p Parties, the partyId's for which the aliases must be deleted (comma separated)

XIX. Get item ratings

URL: /PrefService/<version>/<party>/users/<foreignUserId>/ratings

Method: GET

Parameters

i Items, the itemId's for which the ratings must be retrieved (optional, comma separated)

q Query used to filter the items for which the user's rating must be returned, represented as "<tag>:<value>", where <tag> and/or <value> can be *

ps PageSize, the maximum number of results to return

p Page, the page to return

Rate items

URL: /PrefService/<version>/<party>/users/<foreignUserId>/ratings

Method: PUT

Body

```
<ratings>
     <rating>
          <foreignitemid>…</foreignitemid>
          <groupcontext>
               <foreignuserid>…</foreignuserid>
               <foreignuserid>…</foreignuserid>
               …
          </groupcontext>
          <utility>…</utility>
          <confidence>…</confidence>
     </rating>
     …
</ratings>
```

XX. Delete item ratings

URL: /PrefService/<version>/<party>/users/<foreignUserId>/ratings

Method: DELETE

Parameters

i Items, the itemId's for which the ratings must be deleted (comma separated)

XXI. Get recommendation list ratings

URL: /PrefService/<version>/<party>/listratings

Method: GET

Parameters

l Lists, the listId's for which the ratings must be retrieved (comma separated)

XXII. Rate recommendation list

URL: /PrefService/<version>/<party>/listratings

Method: PUT

Body

```
<ratings>
     <rating>
          <listid>…</listid>
          <utility>…</utility>
          <confidence>…</confidence>
     </rating>
     …
</ratings>
```

XXIII. Delete recommendation list ratings

URL: /PrefService/<version>/<party>/listratings

Method: DELETE

Parameters

l Lists, the listId's for which the ratings must be deleted (comma separated)