HASQUE
-PCM Listening test simulation-

HASQUE DLL VERSION 8.8

FOR WINDOWS OS

FOR THE OBJECTIVE QUALITY EVALUATION

OF AUDIO SYSTEMS

Edition:	**1.3 – 15.02.2021**
HASQUE Software version:	**V 8.8**
Date:	**04.03.2021**

Content

Abbreviations

CTR	Cochlea Transformation
dBov	level below limitation
DLL	Dynamic Link Library
GAL	Gain Alignment
MOS	Mean Opinion Score
SAM	Short Average Magnitude
SPE	Signed Perceptible Error
SPL	Sound pressure level
TAL	Time Alignment

Scope of delivery

HASQUEDll.lib	Import library
HASQUEDll.dll	dynamic link library
HASQUEDLL.h	DLL - header with structures and functions
HASQUEDllApi.h	Header of class HASQUEDllApi
HASQUEDllApi.cpp	class HASQUEDllApi DLL application
HASQUEDLLHelp(English).pdf	This users guide with application examples

User agreements

The usage of above scope of delivery or parts of it (deliverables) is only granted in conjunction with a written agreement with Sound acoustics research and after payment of the license fee and if following rules are accepted and followed:

- This DLL contains solutions which are the intellectual property of Michael Walker-Sound acoustics which must not be misappropriated.
- Any simulation, de-compilation or reverse engineering is prohibited.
- Any changing of the delivered DLL is impermissible.
- The commercial use of this DLL is only allowed with a permanent license, but prohibited for any other license as e.g. developer or test license.
- Any transfer of above mentioned deliverables to third parties who are not mentioned in the contract with sound acoustics is forbidden.
- The grant of usage is not transferable and is only valid for a single user
- The DLL must not be applied for multi user applications nor must be implemented on servers
- Disclaimer of liability: In no event shall Sound acoustics or its suppliers be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this DLL, even if Sound acoustics has been advised of the possibility of such damages.
- Any other liability which belongs not to above disclaimer is limited to 10 Euro.

Functions and interfaces of the HASQUEDllApi

This DLL contains the HASQUE listening test simulator (**H**earing **A**dequate **S**ignal **Qu**ality **E**valuation) and additionally measurement principles for the evaluation of signal properties of audio and telecommunication systems.

This HASQUEDllApi makes easy implementation of the DLL functions in an application possible. Signal interfaces of the DLL are realized as pointers to memory arrays of the application program. This Api allocates the necessary memory and initializes the DLL with a quality scale according to according to ITU-T P.862 and listening test parameterization based on requirements for BDBOS certification tests.

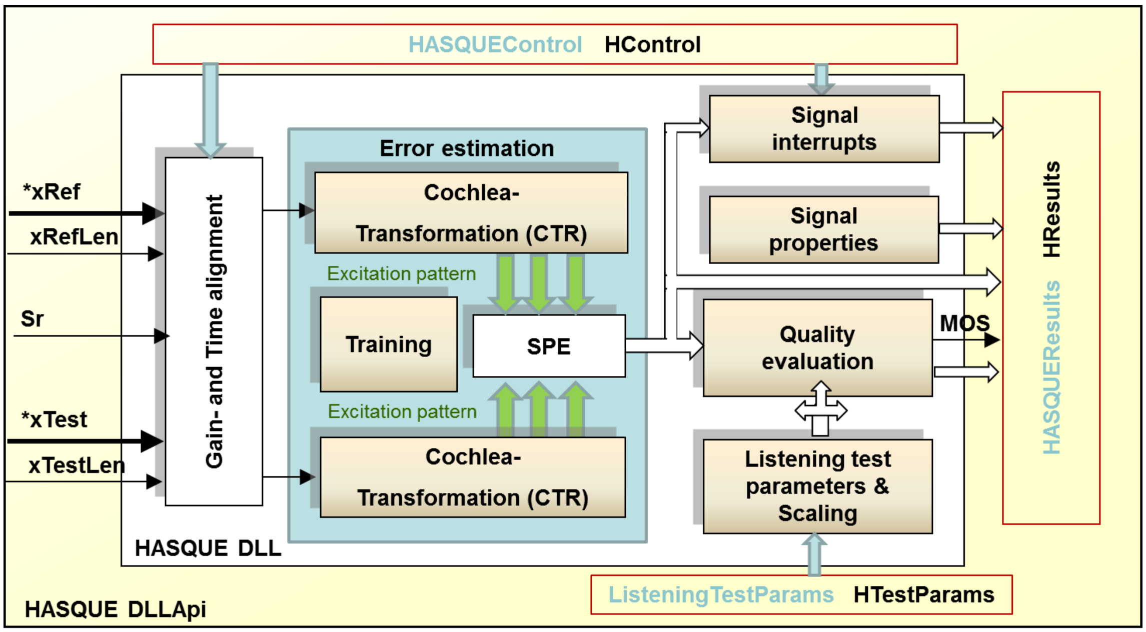

Figure 1: HASQUEDLLApi Interfaces and functions

Figure 1 shows the system arrangement of the HASQUEDLLApi for easy product implementation. With the creation of an object of this class library all necessary DLL functions are initialized and made available.

Error estimation

The simulation of the human hearing system is carried out with the Cochlea transformation (CTR) within the whole available frequency range without band limitation. The CTR crates dependent on the listening test parameter excitation pattern, which are comparable with the psycho acoustic perception of the human hearing system.

The Training is used to compute masking effects with the aid of adaptive algorithms in time and frequency domain in order to receive a hearing adequate masked threshold. The masked threshold is adapted to the background noise of the system under test dependent on the noise properties. Hence constant noise sources as e.g. a motor vehicle with constant speed are differently weighted than variant noise sources as e.g. the rattling of a passing train. The analysis of the noise adapted place coefficients according to the signal properties into different excitation patterns makes a fine approximation to subjective perceived excitations possible and can be seen as a neuronal process

The comparison between clean and processed excitation pattern results in the signed perceptible error (SPE) which indicates the loudness of the distortions in Sone. Negative results correspond with signal attenuation which may be interpreted as signal interrupts from a certain level. Positive SPE results are additional distortions which might result from superposition with strange signal distortions from a certain threshold.

Listening test parameters and scaling

The HASQUE quality evaluation can be adapted to different listening test with the aid of the programmable listening test parameters (Threshold of Acceptance, Bandwidth, and Listening Loudness) based on different quality scales e.g. according to ITU-T P.862 with MOSmax = 4.5 and MOSmin = -0.5 as it is initialized by default during object creation according to any other request as e.g. according to percentage display with MOSmax = 100 and MOSmin =0.

Name	Format	Meaning	Default
SR	int	Sample rate	8000
ThresholdOfAcceptance	float	Threshold of Acceptance in Sone	3.2
UpperFC	float	Upper cutoff frequency – results from SR	4000
LowerFC	float	lower cutoff frequency	100
SystemLevel	float	System level of the listening loudness in dB(SPL)	-13
MOSmax	float	Upper magnitude of the MOS scale (excellent)	4.5
MOSmin	float	lower magnitude of the MOS scale (bad)	-0.5
Compressed	bool	true = compressed (ITU), else natural	true

Settings 1: Listening test parameters and quality scaling with HTestParams

Listening test parameters can be changed to individual tests with the aid of the variable HTestParams based on the indicated ListeningTestParams structure.

Control of DLL functions

The control of DLL functions is carried out with the aid of HControl based on the HASQUEControl structure. Following control is supported:

Gain and Time alignment

Correct error estimation requires a synchronous comparison between reference and test signal with possible same magnitude. To reach this goal, time (TAL) and gain alignment (GAL) functions are initialized by default in the HASQUEDLLApi as indicated in the table below.

Name	Format	Meaning	Default
isGAL	bool	true = GAL on, else fixed gain factor	True
GainCorrDeg	float	Gain factor for gain alignment if isGAL = false	1
isTAL	bool	true = TAL on, else fixed time delay	True
Delay_p2	int	Fixed time delay in samples if isTAL = false	0
TALMax	float	Maximum delay in seconds	1
TALMin	float	Minimum delay in seconds	-0.2
isBlockDComp	bool	true = block compensation on , else off	True
isJitterDComp	bool	true = jitter compensation on, else off	False

Settings 2: Time and gain alignment HControl

GAL may be deactivated, if measurements shall be carried out at a test object with fixed signal delay. This special case is mandatory for automatic parameterization and quality optimization of audio systems in for research and development of new principles. In this case GainCorrDeg must be set to the reciprocal of the desired Gain factor of the test signal amplitude.

With the activation of the GAL function (isGAL=true) any loudness difference between reference and test signal is compensated automatically. The computed Gain factor is indicated in HResult.

The quality evaluation of audio systems with known and fixed latency should be carried out without TAL function (isTAL=false). This concerns for instance quality evaluations at signal processing principles as noise reduction, bandwidth extension or others in the research and development area.

With isTAL = true a delay difference between reference and test signal will be compensated most precisely (accuracy about 1 ms), HResults. Delay_p2 indicates the measured delay in samples.

With activated TAL function it is possible to activate an additional block compensation by setting isBlockDComp = true which might be necessary, if the latency within the same recording changes between reference and test signal. This might occur by cell reselection during radio transmission.

Alternatively to the block compensation a latency jitter can be compensated with isJitterDComp=true. Latency jitter might occur by signal over IP and requires continuous correction of single signal excitations.

The adaptation of the latency limits with TALMAX and TALMIN may become necessary if the range is not covered by the default settings. It is recommended to limit the latency range to the real maximum expected delay of the application in order to save computational power and time and to achieve best possible robustness.

Error classification

Signal interrupts may occur by distorted radio transmission, cell reselection or superposition of distortions. In any case the real signal is not audible if distortions exceed a certain threshold within a perceptible time window. Different error properties as e.g. thresholds and the time window for the detection, the kind of distortions can be controlled in order recognize individual errors.

Name	Format	Meaning	Default
SPIRLOUDTHR1	float	Threshold of loudnes (Sone)	10
SPIRMinIRTime1	float	Minimum excitation time (ms)	10
SPIRLOUDTHR2	float	Not in use	2.5
SPIRMinIRTime2	float	Not in use	0
ThreshOfDistSone	float	Maximum accepted distortion in Sone	15.75
ArtIntervall	float	Interval time of the distortion ms	100
AddIRTimes	bool	If true it adds the times of interrupts during each record	true
ThreshOfAttSone	float	Threshold of attenuations (Sone)	-45
ArtMOSThres	float	Upper threshold of acceptance for the detection	2.5
ArtSpecProperty	float	spectral composition 0 = wide SNR narrow band	0
ArtSpecF1	float	Upper threshold of acceptance for the detection	0
ArtSpecF2	float	Upper threshold of acceptance for the detection	0

Settings 3: Interrupts and programmable errors HControl

Recognition of Speech interrupts

Speech interrupts within one recording are recognized, if the absolute magnitude of SPE_ERROR during speech activity of the reference signal exceeds SPIRLOUDTHR1 for at least SPIRMinIRTime1. If the distortion interval is less than SPIRMinIRTime1, short peaks of SPE_ERROR are neglected in order to avoid miss interpretations.

The control variables SPIRLOUDTHR1 and SPIRMinIRTime1 are set by default to settings for standard listening test simulation corresponding with real perceived interrupts and can be adapted to other application dependent requirements.

Programmable error detection (Error tracer)

The principle of the error recognition is based on the simulation of neuronal processing which uses the statistical probability on the combination of error specific properties. This approach operates with a high recognition rate (typ. > 95%) in most cases due to the number of properties with various individual weightings.

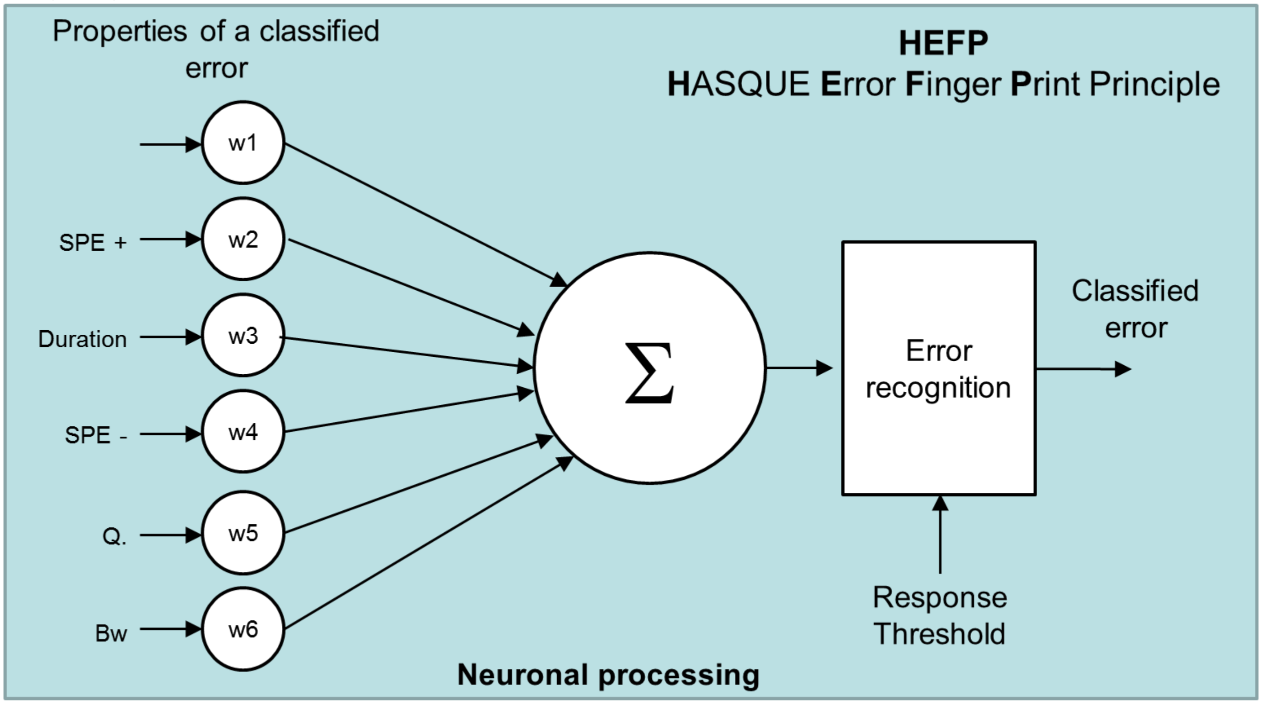

Figure 2: Synthesis of individual properties

The properties of an error are determined in HControl (see Settings 3). These properties can be detected with the aid of each HASQUE measurement system by automated evaluation of scanned recordings.

Examples for classified errors with belonging settings are shown below.

CellReselection

Figure 2 shows an example for distortions, which are produced by cell re-selection. These errors might occur by altering between different cell towers on the road. Hence the error is classified as CellReselection error.

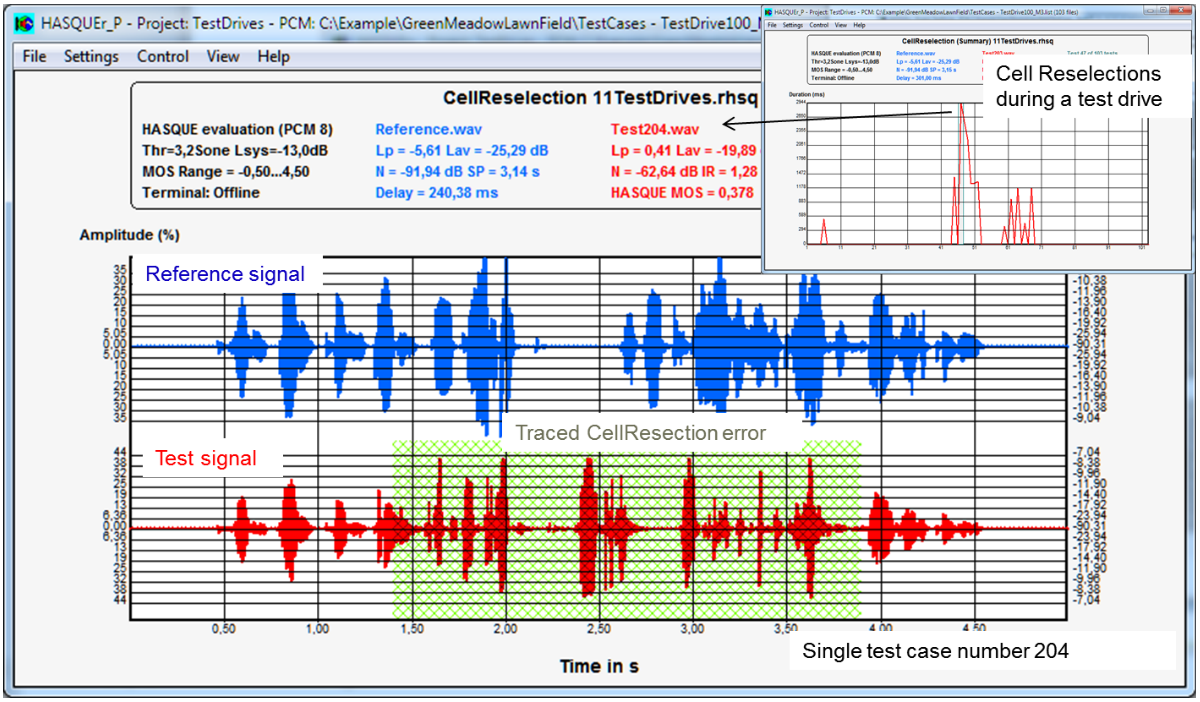

Figure 3: Detection of "CellReselection" errors

The picture in the upper right corner of Figure 3 indicates every occurrence with distortions by cell reselections of a test drive. The red curve points with the magnitude to the time duration of any single recording, which is indicated in the large picture of Figure 3 by "traced CellReselection error".

Following settings of the signal properties are needed to trace Cell Reselection errors:

Signal Property	Value	Unit
HControl.ThreshOfDistSone	15.748	Sone
HControl.ArtIntervall	100	ms
HControl.AddIRTimes	true	Bool
HControl.ThreshOfAttSone	-45	Sone
HControl.MaxCorrelation	79.927	%
HControl.ArtMOSThres	2.64	MOS
HControl.ArtSpecProperty	0	Factor
HControl.ArtSpecF1	0	Hz
HControl.ArtSpecF2	0	Hz

Settings 4: Properties of CellReselection

Upper parameters were detected with the aid of a wizard of the HASQUE measurement system automatically and are used by default.

Martinshorn

Acoustic distortions by a police siren can be introduced during free speaking and handset operation mode of any telecommunication system and are classified as “Martinshorn” distortions.

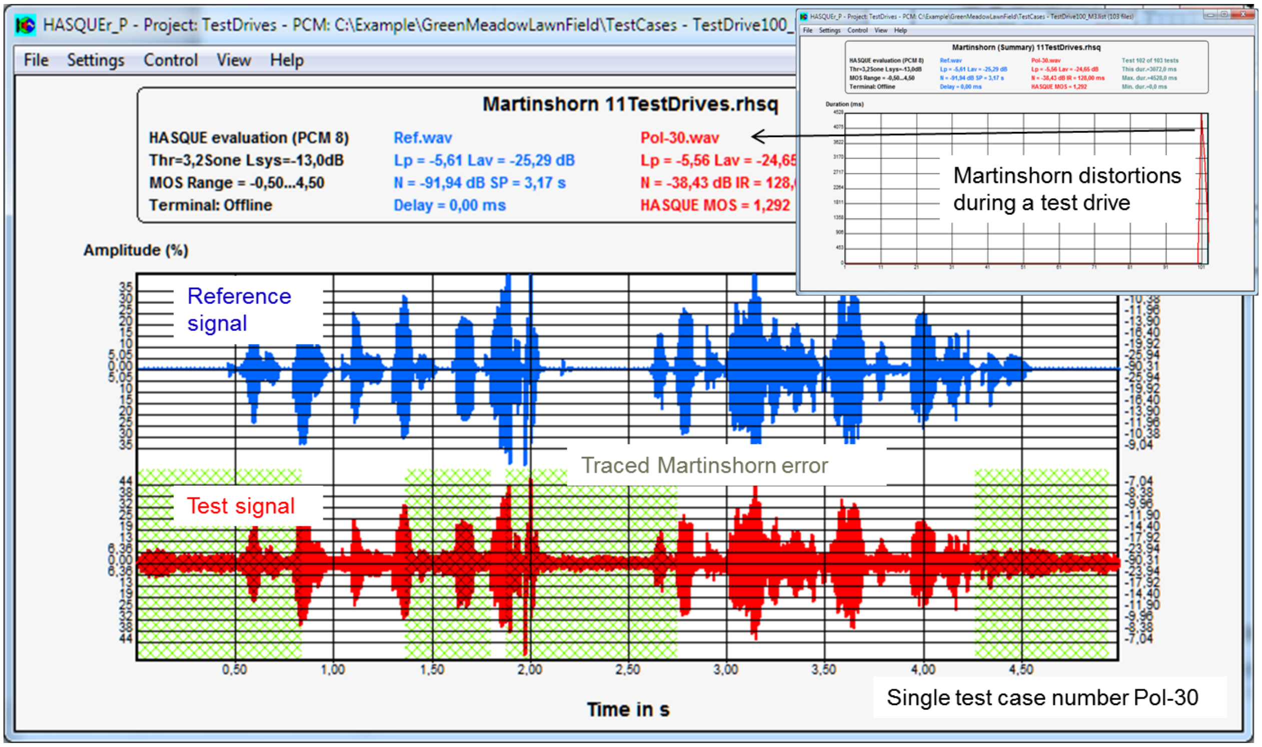

Figure 4: Distortions by police siren

The picture in the upper right corner of Figure 4 indicates every occurrence with distortions by a Martinshorn of a test drive. The red curve points with the magnitude to the time duration of any single recording, as indicated in the large picture of Figure 4 by “traced Martinshorn error”. In contrast to the former classified error, the Martinshorn distortions occurred during the test drive only during the last few test cases. The following settings are needed to trace distortions by police siren:

Signal Property	Value	Unit
HControl.ThreshOfDistSone	13.835	Sone
HControl.ArtIntervall	100	ms
HControl.AddIRTimes	true	Bool
HControl.ThreshOfAttSone	-0.7767	Sone
HControl.MaxCorrelation	91.883	%
HControl.ArtMOSThres	2.76	MOS
HControl.ArtSpecProperty	53.82	Factor
HControl.ArtSpecF1	453	Hz
HControl.ArtSpecF2	609	Hz

Settings 5: Properties of Martinshorn

Upper settings must be carried out before initialization and evaluation. See Software control.

FunkHoles

Audible signal interrupts might be introduced during a test drive by obstacles between sender and receiver. This error type was classified as FunkHoles.

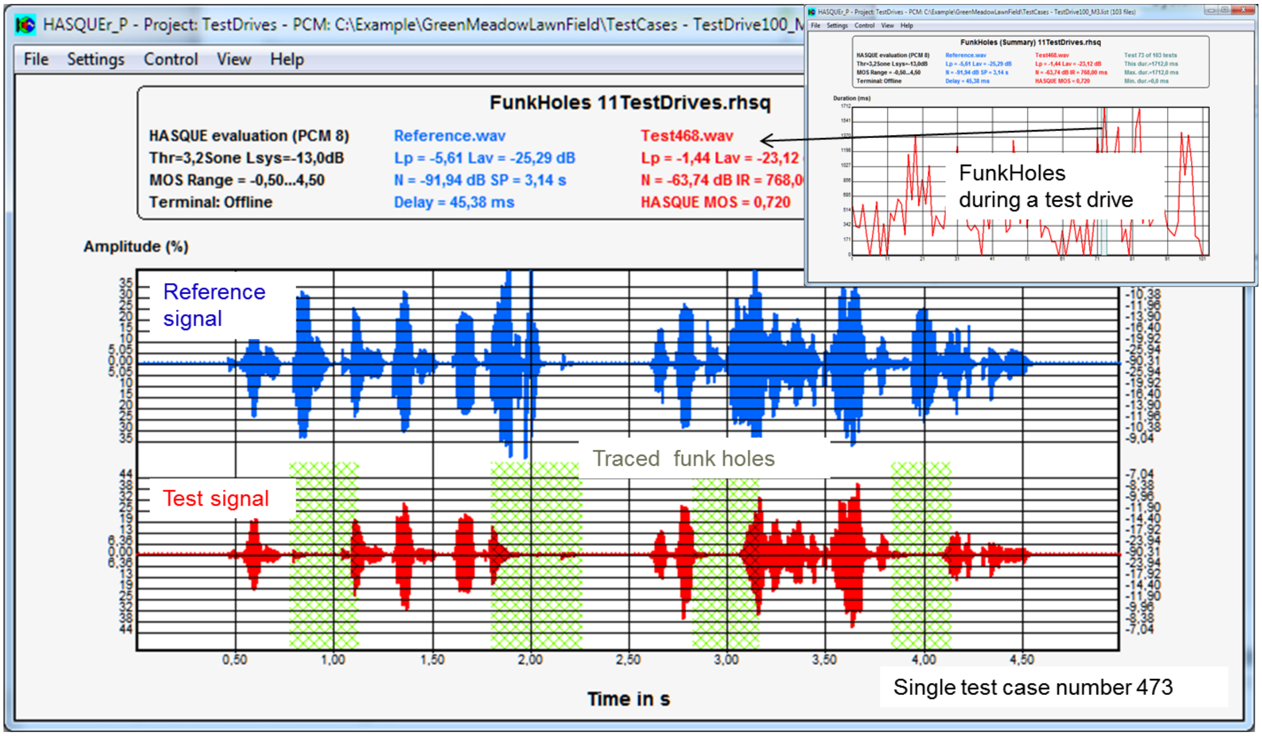

Figure 5: Signal interrupts by funk holes

The picture in the upper right corner of Figure 5 indicates every occurrence with distortions by funk holes. The red curve points with the magnitude to the time duration of any single recording, as indicated in the large picture of Figure 5 by “traced funk holes”.

The following property settings are indicated for the detection of funk holes:

Signal Property	Value	Unit
HControl.ThreshOfDistSone	0	Sone
HControl.ArtIntervall	100	ms
HControl.AddIRTimes	true	Bool
HControl.ThreshOfAttSone	-45	Sone
HControl.MaxCorrelation	86	%
HControl.ArtMOSThres	2.58	MOS
HControl.ArtSpecProperty	0	Factor
HControl.ArtSpecF1	0	Hz
HControl.ArtSpecF2	0	Hz

Settings 6: Properties of FunkHoles

Upper settings must be carried out before initialization and evaluation. See Software control.

Results

Results are combined in HASQUEResuts. Vectors are handed over with Pointers, whereas corresponding reference sizes as e.g. length, time and frequency are indicated with additional variables which can be assigned by its variable name.

Name	Format	Meaning
MOS	float	Mean opinion score of the listening test simulation
SpeechDist	float	Speech distortion in dB(SPL)
PauseDist	float	Pause distortion in dB(SPL)
NVarDist	float	Noise variance in dB(SPL) (roughness)
SPE_Error	*float	Vector of signed perceptible distortions(SPE) in Sone
SPE_ErrorLen	int	Length of SPE vector
SPE_Delay	int	Delay of SPE to the reference signal (number of sub samples)
SamplesPerSPEFrame	int	Number of samples per SPE sample
SPEFrameTime	float	Time interval between SPE samples (ms)

Results 1: Quality measures and errors

Indication of signed perceptible errors (SPE)

The SPE_ERROR can be indicated as shown in Figure 6 in a cartesian system with Y = ordinate indicating SPE and X = abscissa indicating the time axis or in a program with Y=SPE_Error[i] and X=i* SPEFrameTime for i=0, i< SPE_ErrorLen.

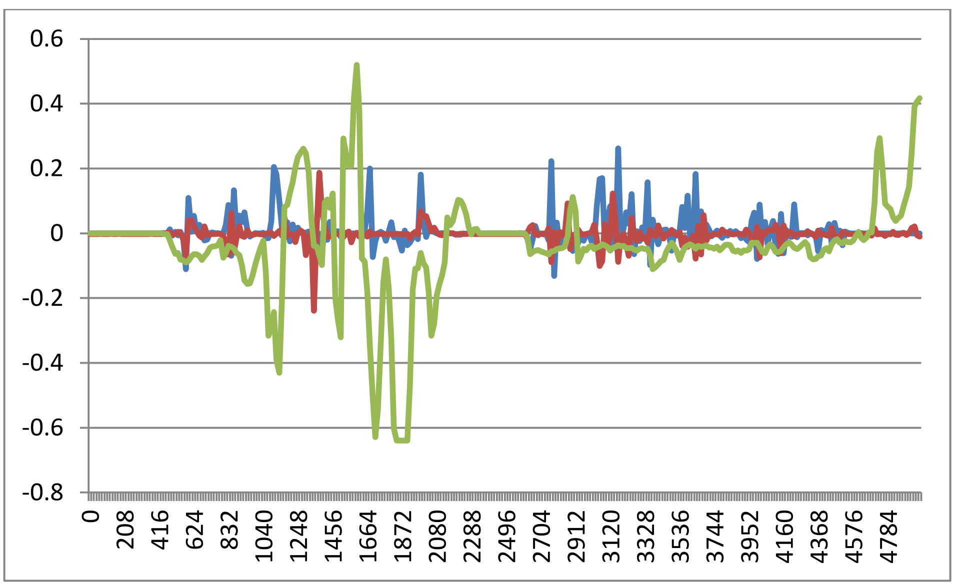

Figure 6: Representation of the SPE_ERROR Vectors (green) with Reference- und Test signal

If the reference and test signal shall be indicated in addition, the time delay between reference signal and SPE must be taken into account with SPE_Delay.

Signal properties

Various signal properties are computed with the aid of high performance measurement functions. These signal properties are provided in HResults as follows:

Name	Format	Meaning
SAMLevelR	float	Peak level in dBov of the reference signal
StdLevelR	float	RMS level in dBov of the reference signal
NoiseFloorR	float	Minimum level in dBov of the reference signal
RefSpectrum	*float	Absolute spectral magnitude of the reference signal
RefSpectrumLen	int	Number of frequency bins
SAMLevelT	float	Peak level in dBov of the test signal
StdLevelT	float	RMS level in dBov of the test signal
NoiseFloorT	float	Minimum level in dBov of the test signal
TestSpectrum	*float	Absolute spectral magnitude of the test signal
TestSpectrumLen	int	Number of frequency bins
Frequency	*float	Frequency vector oft he spectra
TAL_Sample	int	Delay between test and reference signal in samples
TAL_Time	float	Delay between test and reference signal in seconds

Results 2: Signal properties

Signal properties are available after quality evaluation in HResults.

Signal interrupts and individual errors (Classified Error)

HResults provide also the results of speech interrupts and individual programmable distortions or Classified Error as indicated in the table below.

Name	Format	Meaning
SpeechInterrupts	long	Number of samples with speech interrupts
SpeechActivity	long	Number of samples with speech activity
SpeechInterruptsT	float	Time in seconds of speech interrupts
SpeechActivityT	float	Time in seconds of speech activity within the reference signal
ClassErr	*float	Vector of the Classified Error
isClassErr	bool	true if Classified Error occurred during observed record, else false

Results 3: Signal interrupts and Classified Error

The indicated results are useful for statistics and further evaluations. Hence it is easy to indicate the interrupts in percentage related to the reference signal or to indicate the critical times of Classified Error.

As well single statements about the existence of Classified Error within a record is possible with the aid of isClassErr, as indication of the permanently altering signal propertis is possible with the artefact vector.

Individual programmed errors can be indicated as shown in Figure 4 in a Cartesian system with Y = ordinate indicating Classified Error and X = abscissa indicating the time axis or in a program with Y= ClassErrr [i] and X=i* SPEFrameTime for i=0, i< SPE_ErrorLen.

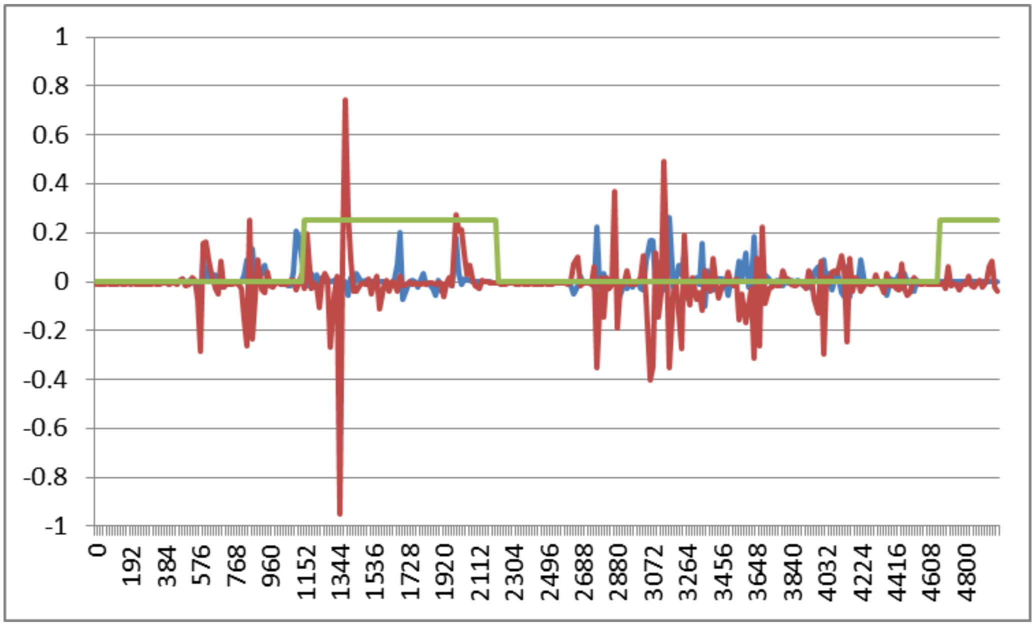

Figure 7: Representation of Classified Error (green) with Referenz- und Testsignal

If the reference and test signal shall be indicated in addition, the time delay between reference signal and SPE must be taken into account with SPE_Delay.

Implementation

The DLL implementation is based on the handover of signal pointers to the reference and test signal streams as follows.

1. Copy the import library HASQUEDLL.lib in the existing project add it to the linker: Project Properties→Linker/Input/Additional dependencies
2. Copy the HASQUEDll.dll to the output directory where YourApplicatio.exe is created or into any common directory which belongs to PATH.
3. Copy the source codes HASQUEDLL.h, HASQUEDllApi.h and HASQUEDllApi.cpp in your source directory and add the sources to your project: Project→Add existing elements
4. Create a new object within your application as indicated in example 1.

Easy application with only two programming lines

With the creation of the HASQUEDllApi class the necessary memory is allocated and the default settings for standard quality evaluation (8kHz Sample rate, ITU-T P.862 MOS scaling, Speech IR weighting …) are initialized. This class applies all available functions of the DLL.

Hence quality evaluation can be carried out with only two programming lines as it is indicated in example 1.

```
#include "HASQUEDllApi.h"
HASQUEDllApi   hasqued;

hasqued.Init();              // Initialization
hasqued.RunHASQUE(xRef, xRefLen, xTest, xTestLen, xSR); // Evaluation

//Results are available in hasqued.HResults
```

Example 1 : Create and apply the new object hasqued

The blue passing arguments xRef und xTest of the hasqued.RunHASQUE() function are floating point buffers with the signal streaming of the reference and test cases which must be provided from the main application.

Extended Application

Software control

Any individual settings or deviations from standard listening test simulation must be set before the evaluation is carried out. Example 2 demonstrates how the latency range can be changed to extended requirements.

```
//Software control: Example extension of the latency range
hasqued.HControl.TALMax = 2; //set the maximum time alignment to 2 seconds
hasqued.HControl.TALMin = -1; //set the minimum time alignment to -1 seconds

//Evaluation
hasqued.Init();             // Initialization
hasqued.RunHASQUE(xRef, xRefLen, xTest, xTestLen, xSR); // Evaluierung
```

Example 2 : Settings must be carried out before evaluation

Parameterization of the listening test simulation

The parameterization of the listening test simulation is carried out by the constructors during creation of a new object per default with standard conditions. Hence the quality scale is set to ITU-T P.862 and the sampling rate as well the threshold of acceptance is set to parameters for certification tests according to the BDBOS. Another test conditions can be determined with HTestParams after initialization and before evaluation as indicated in example 3.

```
hasqued.Init();             // Initialisation

hasqued.HTestParams.BandwidthL = 300;     //lower cutoff frequency
hasqued.HTestParams.ThresholdOfAcceptance = (float)4.3;
hasqued.HTestParams.SystemLevel = 3;
hasqued.HTestParams.MOSmax = 5;           //Scaling max, MOS
hasqued.HTestParams.MOSmin = 1;           //Scaling min MOS
hasqued.HTestParams.Compressed = false;   //natural loudness dependent

hasqued.RunHASQUE(xRef, xRefLen, xTest, xTestLen, xSR); //start evaluation
```

Example 3 : Listening test parameters are set after initialization

Representation of vectors

The Cartesian representation of vectors occurs by two dimensional mapping functions in time and frequency domain. The following examples demonstrate how available vectors of the DLL can be applied. Blue indicated variables are to be provided from the application with Y as ordinate and X as abscissa.

Representation of frequency response

```
for (int i = 0; i<hasqued.HResults.RefSpectrumLen; i++)
       {
       X[i] = hasqued.HResults.Frequency[i];
       YRef[i] = hasqued.HResults.RefSpectrum[i];
       YTest[i] = hasqued.HResults.TestSpectrum[i]);
       }
```

Example 4: Point assignment of Reference- und test signal spectra

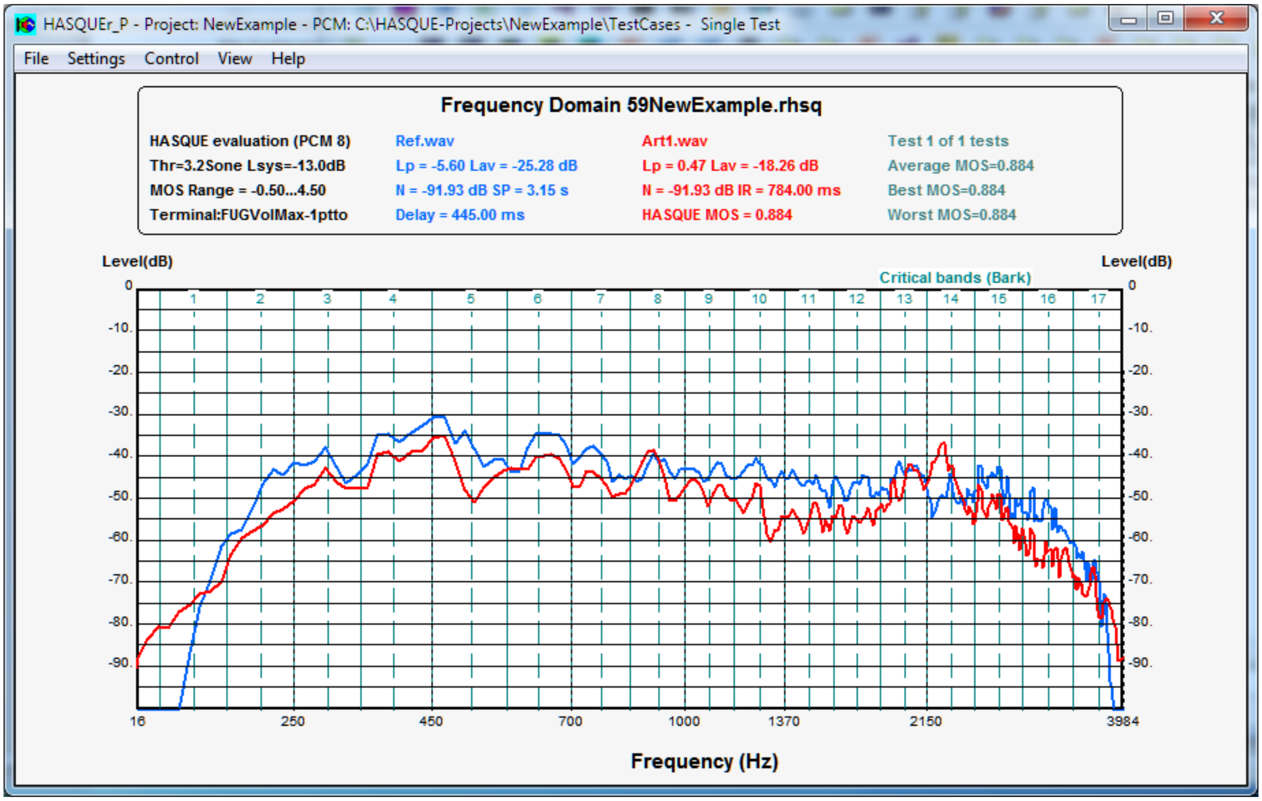

Figure 8: Spectral representation of reference and test cases

Representation of audible errors

The representation of audible errors and the corresponding test and reference signals over the time axis are indicated in **Fehler! Verweisquelle konnte nicht gefunden werden.**. Example 5 clarifies the context between PCM and SPE.

```
long k=0;
long l=0;
int j=0;
float SPENorm = 1/(float)64;
if(hasqued.HResults.TAL>0)
{
        l = hasqued.HResults.TAL;
}
else
{
        k = -hasqued.HResults.TAL;
}

for (int i = hasqued.HResults.SPE_Delay; i<hasqued.HResults. SPE_ErrorLen; i++)
        {
        X[i] = hasqued.HResults. SPEFrameTime *j; j++;
        YSPE[i] = hasqued.HResults. SPE_Error[i]*SPENorm;
        YRef[i] = xRef[k]);
        YTest[i] = xTest [l]* hasqued.HResults.GAL);
        k += hasqued.HResults.SamplesPerSPEFrame;
        l += hasqued.HResults.SamplesPerSPEFrame;
        }
```

Example 5 : Representation of audible errors and the corresponding signals

The time alignment between test and reference signal occurs with the aid of the time shifted index counters l and k dependent on the measured latency TAL. As the SPE_ERROR must be subsampled due to the necessary time resolution, the index counter of the real sampled PCM signals (k,l) must be increased with the whole number of samples per SPE_ERROR sample or per i with SamplesPerSPEFrame respectively. The TAL for the test signal corresponds with the SPE_Delay of the SPE_ERROR and hence is applied as start position of i in order to achieve subsampled time alignment.

Gain alignment between test and reference signal is achieved with the GAL variable. SPENorm is used to normalize the error signal to 1 as it is valid for the PCM signal values.

Representation of Classified Error

The temporal connections between Artefact samples and test signal samples are clarified in Example 6. The resolution of the ordinate X in ms is determined by the sampling time of the sub sampled Classified Error which can be covered with the real sampled test signal by the application of the subsampled SPE_Delay and the block alignment of the test signal with SamplesPerSPEFrame (**Fehler! Verweisquelle konnte nicht gefunden werden.**).

```
long k=0;
long l=0;
int j=0;
if(hasqued.HResults.TAL>0)
{
       l = hasqued.HResults.TAL;
}
else
{
       k = -hasqued.HResults.TAL;
}

for (int i = hasqued.HResults.SPE_Delay; i<hasqued.HResults. SPE_ErrorLen; i++)
       {
       X[i] = hasqued.HResults. SPEFrameTime *j; j++;
       YArt[i] = hasqued.HResults. ClassErr[i];
       YRef[i] = xRef[k]);
       YTest[i] = xTest [l]* hasqued.HResults.GAL);
       k += hasqued.HResults.SamplesPerSPEFrame;
       l += hasqued.HResults.SamplesPerSPEFrame;
       }
```

Example 6: Representation of Classified Error

The time alignment between test and reference signal occurs with the aid of the time shifted index counters l and k dependent on the measured latency TAL. As the Classified Error must be subsampled due to the necessary time resolution, the index counter of the real sampled PCM signals (k,l) must be increased with the whole number of samples per Classified Error sample or per i with SamplesPerSPEFrame respectively. The TAL for the test signal corresponds with the SPE_Delay of the Classified Error and hence is applied as start position of i in order to achieve subsampled time alignment.

Gain alignment between test and reference signal is achieved with the GAL variable. The samples of the Artefact vector are set to 0.25 if Classified Error are recognized, else 0.

Specifications

General

DLL 32 Bit for Windows Operating systems
Sample rate: programmable – default 8kHz
Quality scale: programmable – default for 8kHz samples according to ITU-T P.862
Listening test parameters: programmable - Default according to BDBOS

Signal delay compensation (Latency)

Maximum delay 1000 ms (programmable)
minimum delay -200 ms (programmapble)
Time variance within each record: ≤ 50 ms

Speech samples (reference signal)

According to ITUT-P.862 following guide values:

Speech level: L(peak) typ -6 dBov, L(avarage) typ. -30 dBov
First speech utterance >500 ms (>expected latency) after record start
Last speech utterance >500 ms (>expected latency) before record stop
Speech activity >40 - <80 % oft he record
Record length: 5-10 Seconds
SNR : > 50 dB
Noise floor: >90dBov at 32 Bit >75dBov at 16 Bit – Samples shall not include sequences of zeroes.

Handover arguments signal buffer

Reference nd test signals	: 32 Bit float Pointer
	: Signal amplitude ± (1-x) = maximum peak values (0dBov)
	: 1- x=1- 1/(2^{31})=0.9999…;
Number of reference samples	: long
Number of test samples	: long
Sample rate	: int

DLL Control

The DLL is initialized by default with standard values. The DLL control can be changed if necessary with the aid of the control HASUEControl structure as indicated in the table below

```
typedef struct
{
//Gain compensation by gain alignment (GAL)
	bool	isGAL;		//adaptive GAL if true, else fixed GAL
	float	GainCorrDeg;	//gain factor for fixed GAL
//Latency compensation by time alignment (TAL)
	bool	isTAL;		//adaptive TAL if true, else fixed TAL
	int		Delay_p2;	//Number of samples for fixed TAL
	float	TALMax;		//maximum TAL in seconds
	float	TALMin;		//minimum TAL in seconds
	bool	isBlockDComp;	//extended TAL with block compensation
	bool	isJitterDComp;	//optional extended TAL with latency jitter
//Signal interrupts
	float	SPIRLOUDTHR1;	//Threshold in Sone interpreted as SPIR
	float	SPIRMinTime1;	//Minimum time in ms interpreted as SPIR
	float	SPIRLOUDTHR2;	//Threshold 2 not applied yet for DLL
	float	SPIRMinTime2;	//Minimum time 2 not applied yet for DLL
//Properties of the dignal distortions for a certain error clasification

	float	ThreshOfDistSone;		//Threshold of additionally disturbance
(Sone)
	float	ArtIntervall;	//Interval time
	bool	AddIRTimes;		//include times with interrupts belonging to
artefacts
	float	ThreshOfAttSone;	//Threshold of attenuations (Sone)
	float	MaxCorrelation;		//maximum expected correlation with the original
	float	ArtMOSThres;	//MOS Threshold for proper artifical signal extensions
	float	ArtSpecProperty;	//spectral property AKF - higeh value == narrowband
	int		ArtSpecF1;		//first base frequency in Hz
	int		ArtSpecF2;		//second base frequqncy in Hz

//Others
	bool	SkipLicenseWarning;	//skips warning message before license time has
	//finished if true
}HASQUEControl; //Control HASQUE functions
```

Structures 1: HASQUEControl

Parameterization of the listening test parameters

The DLL is set by default for 8kHz samples according to ITU-T P.862 according to requirements of the BDBOS

```
typedef struct
{
	int SR;
	float ThresholdOfAcceptance; //threshold of max. accepted audible error in sone
	float UpperFC;		//upper cutoff frequency - not applied
	float LowerFC;		//lower cutoff frequency
	float SystemLevel; //listening loudness level 0dB = nominal ...speech = -13dB...
	float MOSmax; //maximum MOS - excellent
	float MOSmin; //minimum MOS - bad
	bool Compressed;	//error weighting true = acc. to ITU-T P.862 - false = 
				natural loudness
}ListeningTestParams; //Listening test conditions and MOS scaling
```

Structures 2: ListeningTestParams

Results

The access to results occurs by the result variable with HASQUEResults structure.

```
typedef struct
{
	float	SAMLevelR;	//peak level of the reference signal in dBov
	float	StdLevelR;	//RMS level of the reference signal in dBov
	float	NoiseFloorR;	//minimum level of the reference signal in dBov
	float	*RefSpectrum;		//discrete spectral magnitudes of the reference spectrum
	int	RefSpectrumLen;		//number of frequency bins of the reference spectrum

	float	SAMLevelT;			//peak level of the test signal in dBov
	float	StdLevelT;			//RMS level of the test signal in dBov
	float	NoiseFloorT;		//minimum level of the test signal in dBov
	float	*TestSpectrum;			//discrete spectral magnitudes of the test spectrum
	int	TestSpectrumLen;	//number of frequency bins of the test spectrum

	float	*Frequency;	//discrete frequencies[0...N] belonging to the spectra[0...N]
	float	GAL;	//gain alignment factor
	int	TAL_Samples;	//time alignment in samples
	float	TAL_Time;	//time alignment in seconds

	float	Mos;	//estimated mean opinion score
	float	SpeechDist;	//total distortions during speech activity in dB(SPL)
	float	PauseDist;	//total distortions during speech pause in dB(SPL)
	float	NVarDist;	//total noise variant distortions in dB(SPL)
	float	*SPE_Error;	//subsampled signed audible errors (Sone)
	int	SPE_ErrorLen;	//total number of signed audible error samples
	int	SPE_Delay;	//delay of the error samples related to the reference signal
	int	SamplesPerSPEFrame;			//number of samples per audible error sample
	float	SPEFrameTime;			//time per audible error sample in ms

	long	SampleRate;			//samples per second
	long	SpeechInterrupts;	//number of samples interrupted
	long	SpeechActivity;			//Number of Samples within speech activity
	float	SpeechInterruptsT;	//Speech interrupt time in seconds
	float	SpeechActivityT;		//Speech activity time in seconds

	float	*ClassErr;	//indication of classified error with SPE_ErrorLen SPEFrameTime
	bool	isClassErr;	//true if artefact detected, else false

	char	ReferenceNumber[MAX_PATH];	//registration number
	int	LicenseNofDays;		//remaining license in days
	char	EndOfLicense[MAX_PATH];		//The license is valid until this date
} HASQUEResults; //Results
```

Structures 3: HASQUEResults

Literature

1. E. Zwicker: „Psychoakustik“ Springerverlag 1982, ISBN 3-540-11401-7.

2. E. Zwicker, H. Fastl : „Psycho acoustics“, Springerverlag 1999, ISBN 3-540-65063-6 .

3. ITU-T P.862 Recommendation: “Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow band telephone networks and speech codecs”, 02/2001

4. ITU-T P.862.1 Recommendation: “Mapping function for transforming P.862 raw result scores to MOS-LQO“, 11/2003

5. ITU-T P.862.3 Recommendation: “Application guide for objective quality measurement based on Recommendations P.862, P.862.1 and P.862.2”, 11/2007

6. E. Terhardt: “Fourier Transformation of Time Signals”, Acustica, Vol. 57, 1985.

7. Rolf Kapust: „Qualitätsbeurteilung codierter Audiosignale mittels einer BARK – Transformation. Dissertation“, Technische Fakultät der Universität Erlangen, 1993.

8. Michael Walker: „Gehöradäquate digitale Sprachsignalverarbeitung“, Funkschau 04/2003, ISSN 0016-2841. Seite 57-58.

9. Michael Walker: HASQUE “Vorrichtung und Verfahren für eine gehöradäquate objektive Qualitätsschätzung von Audiosignalen” DE 10 2005 019 903 A1 2006.11.02, 29.4.2005

Relations:

Figures

Settings

Examples

Results

Data structures