INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 643-9153 • FAX (510) 643-7684

Calculation of the psychoacoustic simultaneous masked threshold based on MPEG/Audio Encoder Model One

Kyrill Alexander Fischer
Deutsche Telekom Berkom GmbH*

TR-97-017

September 1997

Abstract

This report describes the implementation of an algorithm to calculate the *psychoacoustic simultaneous masked threshold* for a specific signal. The implementation is derived from the "Psychoacoustic Model One" as defined in MPEG Audio encoding method. In order to make this model more flexible, some changes and additions have been made, thereby defining a new "generalized implementation". The resulting module `PsychoMpegOne` (C language) can handle user-specified sampling frequencies and is able to use a spectral resolution different from the fixed spectral resolution used in MPEG. In addition, the resulting *global masked threshold* is mapped back to the linear frequency axis in a fine resolution and can directly be compared to the signal's power density spectrum.

*This work has been carried out during a stay at ICSI as a research visitor in 1997. Permanent address: Dr. K. A. Fischer, Deutsche Telekom Berkom GmbH, Technologiezentrum Darmstadt, Postfach 10-00-03, D64276 Darmstadt, Germany; e-mail: `fischer@tzd.telekom.de`, `k.fischer@berkom.de`

Contents

1 Introduction

1.1 The effect of simultaneous masking

When analyzing acoustic signals it is sometimes useful to have information about the their auditory perceptual properties. Knowledge about the relation between the measurable properties of an acoustical signal (like energy, spectral and temporal properties) and their corresponding *perceptual* results (eg. perceived loudness or tone) have led to a wide number of attempts to mimic or model key properties of the perceived signal. For an overview, see, for example, Zwicker 1990 ([6]).

One of the basic concepts resulting from these efforts is the attempt to understand and model the so-called *simultaneous masking effect*. This effect refers to the fact that the presence of a tonal-like sound component at a certain frequency can lead to a considerable reduction of the perceptibility of other tones that are close to it on the frequency axis. When the level of the neighboring component falls below a certain threshold (defined by the level and distance on the frequency-domain), it eventually will become inaudible. In this case, this tone is said to be *masked* by the louder tonal-like component.

The analysis of this effect is necessarily based on the measurement of subjective perception of a large number of listeners. The resulting data constitute an *empirical* basis for the description of the underlying perceptual effects.

Nevertheless, since the results indicate a rather small variability between different listeners, this effect can be regarded as being a very basic phenomenon being intrinsically related to the mechanisms involved in the process of human auditory perception in general. It is therefore reasonable to try to find means for *modeling* this effect, i.e. calculating the *average masked threshold* for a given spectrogram rather than measuring it. The result using the rules that define the method of calculation of an *average masking threshold* is necessarily inexact in the sense that the actual masked threshold for a specific listener may (and generally will) be different from the masked threshold predicted by the algorithm. However, it is assumed that the difference will be reasonably small and that the use of the particular algorithm can therefore be justified.

A method for calculating an approximation of the *masking threshold (MT)* can be regarded as a realization of a *psychoacoustic model* of human perception. Therefore, an algorithm of this kind (or the implementation thereof) will be referred to as a "(psycho-acoustical) model" throughout this report.

1.2 Modeling the effect of simultaneous masking

This report describes the implementation of a psychoacoustic model that allows the calculation of a masked threshold (specified in the frequency domain) corresponding to a given signal. This implementation is derived from an implementation of the MPEG Audio-encoding scheme as provided in the MPEG-2 Audio Simulation Software[1]. For general information on the MPEG Audio encoding scheme, see, for example, [2] and [3]. The ISO/IEC MPEG standard for audio encoding itself is referenced in [2] as [1].

The psychoacoustic models provided in the MPEG-Implementation are specifically designed for the MPEG algorithms requirements. The MPEG-Algorithm will be described briefly in section 2.1. In order to define a new, general-purpose and flexible implementation of the underlying psychoacoustic model, it is necessary to change and adapt the implementation.

The *generalized implementation* described in this report has the following properties:

Variable resolution: The resolution of the frequency axis used by the MPEG model is fixed. In the new generalized implementation, the resolution became a variable to be provided by the user.

[1]The software used was a package called `MPEG-dist07.tar.gz`, copied from `ftp://svr-ftp.eng.cam.ac.uk`, directory `/pub/comp.speech/coding`. This package constitutes the MPEG-2 Audio Simulation Software Distribution 07 as published by the ISO MPEG Audio Subgroup Software Simulation Group (1996) covering the ISO 13818-3 MPEG-2 Audio Codec. The package contains source code for the MPEG-2 audio encoder and decoder under development by the MPEG Audio software simulation ad-hoc group. The package includes the multichannel and low sampling frequency extensions in MPEG-2 audio. Throughout this report, the version enabling the new low sampling frequency extensions will be referred to as "MPEG", the "MPEG-Implementation", and the "MPEG-Model".

Variable sampling-frequency: The MPEG algorithm requires the acoustical signal to be sampled at specific sampling rates. The sampling rate can now be chosen independently from MPEG-specific requirements.

Ability to disable usage of hearing threshold in quiet conditions: MPEG uses a fixed scaling for the power spectral density's dB-values. According to this scaling, an Amplitude of 32767 (maximum value of type `short`) corresponds to 96dB SPL. The values calculated according to this scale can then be compared to the *hearing threshold in quiet conditions*, given as a table of dB SPL-values in the implementation. The values of the low energy hearing threshold in quiet are used in several stages of the calculation of the masked threshold. Since, however, for some applications it might be desirable to not consider this hearing threshold at all, its usage can be disabled using a `#define`-switch.

The idea behind the development of the generalized implementation is to provide a scalable solution that returns the original MPEG-results when used with the set of parameters used in MPEG.

The MPEG Psychoacoustic Model One was chosen as a basis for this generalized implementation because MPEG is a widely-used de-facto standard for image and audio-transmission. The psychoacoustic models used in MPEG have proven their ability to reasonably predict the masking properties of the human perception.

However, it should be noted that these models are far from constituting a complete or "final" answer to the question of modeling human perception. These models do approximate the masking threshold within limits of accuracy that are reasonable and sufficient for the MPEG algorithm. The results of the MPEG-Model should themselves be regarded as an approximation Therefore, it does not seem to be reasonable to try to approximate MPEGs results to a higher accuracy than given by the model's original implementation itself. Tolerances within some ± 3 dB are therefore assumed to be acceptable for the generalized implementation.

1.3 Organization of this report

In the next section, a overview of the original MPEG model will be given. This includes a brief outline of the several modes of MPEG encoding and summary of the calculations carried out in the original MPEG Model. This section will be ended by a brief summary of some of the important properties of the MPEG model.

In section 3, the generalized implementation will be described. It will be described how the generalization can be achieved. This involves a detailed description of new methods to define some of the algorithm's values. These values have been provided as table-entries and are read at run-time in MPEG. In order to extend the range of application, the definition of these parameters using formulas and specific algorithms is described.

Section 4 summarizes the properties of the new implementation. First, the accuracy of the approximation implemented in the generalized version is addressed. The values of the "generalized parameters" are compared to the values provided by MPEG's tables. This comparison is necessary in order to validate the generalized model's abilities. It will be shown that the differences between MPEG's tabled values and the values derived from the "generalized methods" are effectively negligible, thereby justifying the algorithms used in the generalized implementation. This section does also contain the description of a sample application, showing the results obtained by the new implementation. Again, these results will be compared to the results obtained by using the original MPEG model.

In the summary (section 7), the most prominent advantages and disadvantages of this generalized implementation will be mentioned.

2 Summary of the psychoacoustic analysis performed in the MPEG Audio Encoder

2.1 Algorithm of MPEG Audio-Encoding

The MPEG Audio encoding standard uses a filter bank that subdivides the frequency range covered by the input signal into 32 bands of equal bandwidth. The encoding mechanism is based on the determination of the short-time Mask-to-Noise-Ratio MNR(i) for each of the $i = 1..32$ bands. The $\text{MNR}(i) := \text{SNR}(i) - \text{SMR}(i)$ is derived from the Signal-to-Mask-Ratio SMR(i) calculated by a psychoacoustic model and the Signal-to-Noise-Ratio SNR(i) (defined by a table).

The encoding algorithm then iteratively assigns additional bits to the channel currently having the smallest MNR(i) (thereby increasing this channel's MNR(i) by about 6 dB) until all bits available for this frame are allocated. This analysis and bit-allocation scheme is repeated for every frame to be encoded[2].

In MPEG, calculation of the SMR(i) can be carried out using either psychoacoustic model "one" or psychoacoustic model "two". Model "one" is intended to be rather simple, while model "two" is more complex. The parameters used for the models depend on the encoding "MPEG-layer", too. The layer (I or II) essentially specifies the number of input samples used for the analysis.

Since model "two" requires the knowledge of the previous two spectra (having been calculated at specific time instants), the generalized implementation described in this report is derived from layer II's psychoacoustic model "one".

2.2 Description of Psychoacoustic Model One of the MPEG Audio (Layer II) encoding algorithm

The MPEG psychoacoustic model one for layer II is implemented as function `II_Psycho_One()` in the module `tonal.c`. The sampling rate of the signal to be analyzed using MPEG is restricted to one of the values of `fs` = 16.0, 22.05, 24.0, 32.0, 44.1, 48.0 kHz[3].

The model internally calculates the *global masked threshold* using an intermediate resolution and then reduces it to a *minimum masked threshold*, covering the 32 bands of the filter bank. The output of the function `II_Psycho_One()` is the SMR for each of the 32 bands. For this calculation, the following steps are performed in the MPEG encoder:

Definition of tables The first time the function `II_Psycho_One()` is called, some initialization is performed. Two tables (provided as text files in the subdirectory ../encoder/tables) are read in order to define the values of the array `cbound[]` and the array `ltg[]`, which has the fields `ltg[].line`, `ltg[].bark` and `ltg[].hear` being defined from the table. (An additional field, `ltg[].x` is used for the calculation.) The table files also define the number of elements in the arrays. The selection of the particular table files depends on the MPEG-layer and the input signal's sampling frequency. Given a sampling frequency of fs=16.0 kHz, the number of elements indicated in the corresponding table files is 21 for `cbound[]` and 133 for `ltg[]`.

Calculate power density spectrum The input signal (1152 samples in layer 2) is centered to cover a 1024-sample wide analysis window. (The $1152 - 1024 = 128$ samples not being covered by this window are supposed to be neglible.) The resulting array of 1024 values is weighted using a Hann window prior to performing the 1024-pt FFT analysis. The resulting power spectrum therefore always has a fixed length of 512 dB-scaled values.

Identify *tonal* components The function `II_tonal_label()` examines all local maxima discovered in the spectrum. Every local maximum that meets some conditions (difference to neighboring components exceeds some limit) is being marked as a "tonal" component.

Identify *noise* components The remaining lines of the spectrum are being treated as "noise". Neighboring lines are replaced by a single line at a corresponding central frequency (calculated using a specific geometric-mean method). The noise contributions are calculated by the sum over a whole bark-range (defined by the corresponding range of `power[]` indexed by `cbound[i]`...`cbound[i+1]`). For this reason, the maximal number of effective noise lines is given by the value of the variable `PMO_crit_band` indicating the number of bark-ranges covered by the power density spectrum `power[]`. (The definition of the variable `PMO_crit_band` and the array `cbound[]` will be described in section 3.4).

[2]Each frame in MPEG layer II has 1152 samples, which corresponds to 72 ms for fs=16 kHz.

[3]The implementation provided in distribution 07 already includes the new "low sampling frequency"-extension. Without this extension, the bit rates allowed are 32.0, 41.1 and 48.0 kHz.

Sub-sampling At this point, the spectrum has already been reduced to a list of "tonal" and "noise"-lines. Now the position of the lines is analyzed and neighboring lines within a certain range (measured on the Bark-scale) are being deleted. This essentially avoids having "tonal" and "noise" -components at nearly the same position. In addition, any lines below the auditory threshold in quiet (as given by `ltg[].hear`) are deleted.

Apply spreading function The array `ltg[].x` is now defined by a copy of the power density spectrum convolved with a signal-dependent *spreading function* using a nonlinear warped frequency axis. The warping is defined by the mapping between the 512 elements of the power density spectrum `power[]` and the 133 values of `ltg[]` using the values `ltg[].line`. Then, considering the values `ltg[].bark`, the warped spectrum is convolved by a spreading function that defines the contribution of every "tonal" and "noise"-component to the global MT being stored in `ltg[].x`. The formulas used for "tonal" and "noise"-like contributions to the global MT are different because noise-like components tend to slightly increase the masking threshold compared to tonal-like components of the same level.

Map to 32 values The approx. 133 values of the global masking threshold `ltg[].x` are now mapped to only 32 values, defining the *minimum masked threshold* `ltmin[k]`. This is done by using the minimum value found in that range of `ltg[].x` that corresponds to the band k (k=1..32).

Calculate SMR The function `II_smr()` replaces the values in `ltmin[k]` by their corresponding SMR-values. The energy of the signal in each band is taken from the array `spike[k]` or `scale[k]` using `max(spike[k], scale[k])`. The values in `spike[k]` are given by the sum of the values of `power[]` in the corresponding range of `power[]`. (`scale[k]` has to be provided as a parameter to `II_Psycho_One()`; it contains the maximum values of the 32 time-domain filter bank outputs.) This means, in order to calculate the Signal-to-Mask-Ratio (SMR), MPEG estimates the "Signal" from the sum(`power[]`) and the "Mask" from the min(`ltg[].x`), each taken over the corresponding range of values. The resulting array `ltmin[]`, containing the SMR-values for the 32 filter bank channels, is returned to the calling function and will be used for the bit allocation procedure.

2.3 Specific properties of the MPEG Audio Psychoacoustic Model One

One of the basic properties of this model is the fact that regions of the signal's spectrum are internally represented by only a *single* spectral line for this whole range. The width of each of this range is determined by the spectral position (ranges at lower frequencies are generally smaller compared to ranges at higher frequencies) and by the type of signal (tonal, non-tonal) most prominent in each range. Since the type of the signal can only be completely *tonal* or completely *non-tonal* no mixtures between these two are possible. This can be regarded as one of the major limitations of the algorithm.

On the other hand, this reduction to only a few spectral representatives severely reduces the numerical effort used for the calculation of the masked threshold. Compared to MPEG's Psycho-acoustical Model Two, the numerical effort in Model One is about `fftlen` times smaller.

The determination of the type of signal is based on the *spectral shape* of the signal. The more pronounced a "peak" at a given frequency is, the more likely it is that this component will be classified as a "tonal" component. The algorithm of MPEG's implementation is based on a fixed spectral representation always using 512 samples. Therefore, changing the size of the spectral representation can give rise to a different classification of tonal/non-tonal components. Of course, all internal parameters used for the detection of "tonal"-like components have been changed to variables that scale according to the number of samples used for the spectrally representation. Nevertheless, since the spectral shape returned by the spectral analysis algorithm (FFT) usually is not independent from the order of analysis used, differences in the classification results still can occur for different spectral representations.

MPEG's Model always assumes the existence of a well-defined "hearing threshold in quiet". This assumption is generally true, since the internal representation defines a signal's power in dB_{SPL} (sound pressure level) to be given by $20\lg(A_{short})$ (A_{short} being the signal's amplitude when represented as a C-language `short`-type value, i.e. in the range -32768 to 32767). The loudest signals possible therefore are assumed to have a level of about

$96dB_{SPL}$. Since the (processed) signal is usually presented at a much smaller level, this assumption of the hearing threshold in quiet lying at about some dB (within the frequency band of speech) is reasonable.

It should be noted that the result returned by this Model in the MPEG Audio algorithm is used in a *very indirect* way: The 32 SMR-Values returned by this model are used to determine the *order* of assigning quantization bits to the 32 frequency channels to be transmitted. Any small deviation of the values returned by this model from the "true" SMR-value will generally not result in a severe or even noticeable degradation of the encoder performance. Therefore, even a deviation in the order of some dB from the "true" value usually can be tolerated within MPEG.

3 The generalized implementation derived from "Psychoacoustic Model One"

The new module is called `PsychoMpegOne`, consisting of a C-language file `PsychoMpegOne.c` and the corresponding `PsychoMpegOne.h`. This module is intended to provide an independent, stand-alone implementation of a generalized version of MPEG's psychoacoustic model one to be included/linked into general signal processing programs. All variables are local to their corresponding functions or are static to this module. Some variables being used in several functions have been made static for the whole module ("global within the module"); these variables are named `PMO_` followed by the unique variable name part. The changes to be described are:

- function call; name of function and I/O-parameters
- handling of input signal (windowing, offsets)
- generalization: replacement of fixed (previously `#defined`) values by variables
- generalization: replacement of parameters provided by tables in MPEG have been replaced by formulas providing "generalized tables".
- generalization: re-mapping of intermediate and resulting arrays
- generalization: compile-time switches

These changes will be described in the following sections. The following sections will describe the new implementation on a very *basic* level, so there will be very detailed, "low-level" information for the remaining part of section 3.

3.1 Function Call

The function performing the whole analysis has been named `PsychoMpegOne()`:

```
long PsychoMpegOne (
short buffer[],           /* in */
long bufferlen,           /* in */
long fs,                  /* in */
long fftsize,             /* in */
double *spectrum,         /* out */
double *masking,          /* out */
double *spectrum_eff,     /* out */
long spectrum_range,      /* in */
double *masking_eff,      /* out */
long masking_range,       /* in */
double *smr_eff           /* out */ )
```

where `buffer[]` is the array containing the (unweighted) `bufferlen` samples of C-type `short` to be analyzed. The sampling rate `fs` in Hertz and the requested resolution `fftsize` have to be specified. `fftsize` is the length of the analysis window prior to calculation of the FFT. (`fftsize` will internally be adapted to the closest corresponding power of two). The resulting power density spectrum always has `fftsize`/2 samples; it is returned in `spectrum[]`. The global making threshold calculated by the function is returned in the array `masking[]`. For the calculation of the SMR, returned in `smr_eff[]`, the values of `spectrum[]` and `masking[]` are represented internally using a reduced resolution. The resolution to be used can be specified using the values `spectrum_range` and `masking_range`. The resulting "effective" values (denoted by `*_eff`), resulting from the reduction of resolution, are returned in `spectrum_eff[]` and `masking_eff[]`, respectively. The SMR returned in `smr_eff[]` is given by the difference of `spectrum_eff[]` and `masking_eff[]`. All arrays have the length of `fftsize/2` elements and have to be allocated prior to calling `PsychoMpegOne()`.

3.2 Handling of input signal

The array `buffer[]` of length `bufferlen` is centered in the analysis window of length `PMO_fft_size`==`fftsize`. If `bufferlen` < `fftsize`, zeroes are padded. If `bufferlen` > `fftsize`, only the central part (`fftsize` samples) of the signal in `bufferlen` will be used for analysis. The signal is windowed by a Hann window prior to FFT analysis.

3.3 Replace important `#define` by variables

The MPEG-Model uses a fixed length of 1024 for the Fourier-analysis window. This length has been made a parameter to be specified by the user. Therefore, the new variable `PMO_fft_size` has been introduced and is used instead of the former `FFT_SIZE`, which used to be `#defined` to 1024. Likewise, the variables `PMO_han_size` (half of `PMO_fft_size`) and `PMO_fs` (sampling frequency in Hertz) have been introduced. `PMO_fft_size` == `fftsize` are used for FFT-analysis (input); the resulting power density spectrum then has `PMO_han_size` := `PMO_fft_size`/2 samples.

3.4 Replace MPEG's parameter tables by formulas

During the process of initialization, the original MPEG-implementation reads some tables that are provided as part of the distribution in order to define some basic parameters. The selection of the particular table files is based on the sampling frequency and the layer being used. The functions reading the tables in MPEG are `read_cbound()` and `read_freq_band()`. For the generalized implementation, the sampling frequency should be allowed to have values different from the values mentioned above. In order to deal with other sampling rates, the reading of table-values has been replaced using specific *formulas*. The formulas being used have to provide values for the two arrays `cbound[]` and `ltg[]`. Therefore, the `read_`-functions have been replaced by new functions called `define_cbound()` and `define_freq_band_v2()`. These functions are supposed to return the same values as given in the corresponding tables when called for the same conditions. In addition, they will provide reasonable values for conditions not covered by MPEG.

3.4.1 Calculation of values for `cbound[]` (function `define_cbound()`)

The table `cbound[]` is used for the definition of bark-scale boundaries in the (linear) frequency scale covered by the `power[]`-array. The array `power[]` has `PMO_han_size` values (in MPEG, `PMO_han_size`=512 always) and it covers the frequency range 0...$fs/2$ Hz. The number of Bark-ranges within this frequency-range itself depends on fs; MPEG uses 21 for fs=16.0 kHz up to 27 for fs=48.0 kHz. The values in `cbound[]` specify the boundaries of adjacent bark ranges as index values within the range 0..`PMO_han_size`.

Several analytical expressions have been proposed for the mapping of frequency measured in Hertz to the frequency measured in Bark. One of these expressions converting the frequency f in Hertz to the corresponding bark value z in Bark has been proposed by Schroeder [4]:

$$f[\text{Hz}] = 650 \sinh(z/7). \tag{1}$$

Another empirical mapping of this kind, which does not require any trigonometric or hyperbolic function, has been proposed by Traunmüller [5]:

$$f[\text{Hz}] = 1960 \frac{0.53 + z}{26.28 - z}, \tag{2}$$

which can be inverted for the corresponding inverse mapping

$$z[\text{Bark}] = \frac{26.81 f}{1960 + f} - 0.53. \tag{3}$$

The equations 2 and 3 have been implemented in the new functions `hz2bark()` and `bark2hz()`. They are used by the functions `define_cbound()` and `define_freq_band_v2()`. First, the number of elements in `cbound[]`, `PMO_crit_band` is defined using `hz2bark(f2/2.)`, and then the boundaries `cbound[]` can be calculated using `bark2hz(z)` (z=1..`PMO_crit_band`), scaled to 1...`PMO_han_size`. The variable `PMO_crit_band` replaces MPEG's variable `crit_band`.

3.4.2 Calculation of values for `ltg[].bark`, `.line` and `.hear` (function `define_freq_bands_v2()`)

The array `ltg[]`, consisting of the fields `ltg[].x`, `ltg[].line`, `ltg[].bark` and `ltg[].hear`, has the number of elements indicated by a variable named `sub_size` in MPEG. This value itself is read from the table defining `ltg[]`. In the generalized implementation, `sub_size` has been replaced by `PMO_sub_size`.

In order to be able to define the table-read values in a similar manner as done by the tables itself, the "rule" behind the tables structure has to be identified and then to be implemented in a generalized form. The generalization method used for each field of the structure `ltg[]` is described in the following paragraphs.

`ltg[].line` The rule common to all tables provided in the MPEG implementation is a piecewise constant increment of the field `ltg[].line`, according to

$$\Delta\texttt{ltg[i].line} = \begin{cases} 1 & : \quad 0 \leq \text{line} \leq 48 \\ 2 & : \quad 48 < \text{line} \leq 96 \\ 4 & : \quad 96 < \text{line} \leq 192 \\ 8 & : \quad 200 \leq \text{line} \end{cases} \tag{4}$$

for covering the range `ltg[].line` $= \{1..$`PMO_han_size`$\}$. This process always starts with `ltg[0].line` $= 0$. From this scheme of constructing the values for `ltg[i].line`, the number of elements (`PMO_sub_size`) follows. In the tables provided in MPEG, the last entry for `ltg[].line` is 480 (not 511). This is due to the fact that in MPEG only 30 out of 32 channels are defined using the masking-model. The uppermost channels are simply approximated by the masking threshold in quiet, as provided by `ltg[].hear`. However, in the generalized implementation the full range of `power[]` will be used. Therefore, the number of elements (`PMO_sub_size`) is 133 for MPEG and 137 for the generalized implementation when using `PMO_han_size` $= 512$.

This rule is used in the function `define_freq_bands_v2()`[4]. The rule has been generalized to become applicable for values of `PMO_han_size` other than 512 by using scaled versions of the index boundaries given in equation (4). In addition, for `PMO_han_size`≤ 128, the rule itself has been changed to

$$\Delta\texttt{ltg[i].line} = \begin{cases} 1 & : \quad \text{line} \leq 48 \\ 2 & : \quad 50 \leq \text{line} \end{cases} \tag{5}$$

in order to provide a reasonable resolution even for small values of `PMO_han_size`.

[4]In a first attempt, another function called `define_freq_band()` has been defined, using a simpler way to define values of `ltg[]`.

ltg[].bark The value of **ltg[].bark** can then be calculated from the frequencies corresponding to the value of **.line** using **hz2bark()**.

ltg[].hear The values of **ltg[].hear** are defined using the new function **hearmask(f)**. This function uses a linear interpolation from values defined in a static table to determine the masked threshold in quiet, valid for the frequency f (in Hertz). The resulting value is returned in dB SPL and will be compared directly to the values given in **power[]**. As for the calculation of **ltg[].bark**, the frequency values corresponding to **ltg[].line** are used in the determination of the value of the argument f in **hearmask(f)**.

3.5 Calculation of SMR and re-mapping to FFT-Analysis' fine spectral resolution

The MPEG-Algorithm defines the *minimum masked threshold* from the *global masked threshold*, thereby reducing the number of elements from 133 (for **ltg[i].x**) to 32 (for **ltmin[]**). In the generalized implementation, the *global masked threshold* is returned after re-mapping the 133 values of **ltg[i].x** to the range of the original power density spectrum **power[]**, i.e. to the range i=1...**PMO_han_size**. This is done in the new function **re-map_mask()**. This result is returned in the array **masking[]**.

As mentioned in section 2.2, MPEG calculates the SMR from the difference of the sum (**sum()**) over the **power[]**-values (in a range corresponding to one of the 32-channels) to the minimum (**min()**) of the global masking threshold **ltg[].x** in the same range. In the generalized implementation, the calculation of the **sum()** has been replaced by calculation of the **mean()**. This generalization can be done easily: Since the number of channels to be summed is always 16 in MPEG, the result of calculating **mean()** can be made equal to using **sum()** by subtracting the factor dB(16)=12.04 dB from the global masked threshold. Now, the signal estimation can be carried out using **mean()** and corresponds directly to the values given in **power[]**, without any offset. In addition, the global masking threshold can be compared directly to the spectrum itself or its local **means()**s (rather than to a signal estimate that lies about 12dB above the spectrum). In the array **spectrum_eff[]**, this "effective spectrum" as calculated using the local **mean()** over **spectrum_range** values, is returned. In MPEG, **spectrum_range**=16 is used. In the generalized implementation, the value of **spectrum_range** can be specified; it should be a multiple of 2 in the vicinity of 16. Increasing the value of **spectrum_range** decreases the spectral resolution of the effective spectrum (returned in **spectrum_eff[]**).

The reduction of the global masked threshold **ltg[].x** to the minimum masked threshold **ltmin[]** using a **min()**-function can not be generalized in a similar manner. Therefore, the usage of **min()** has not been changed for the generalized implementation. The number of neighboring values to be considered when determining the **min()** can be specified by **masking_range**. In MPEG, **masking_range**=16. If **masking_range**=1 is specified, the resulting effective masking function returned in **masking_eff[]** is equal to the global masked threshold **ltg[].x** and has the finest spectral resolution possible. Again, increasing the value of **masking_range** decreases the spectral resolution of the effective masking curve (returned in **masking_eff[]**).

3.6 New compile-time switches (#defines)

In order to define the exact algorithm used in the generalized implementation, some compiler-switches have been used.

PMODEF_CONSIDER_HEAR When defined, the hearing threshold in quiet, as defined in **ltg[].hear**, will be considered for the calculation. The minimum of the masked threshold is directly specified by this hearing threshold in quiet.

If not defined, all comparisons to **ltg[].hear** are skipped over and **ltg[].hear** will never be considered.

PMODEF_USE_TABLE When defined, MPEG's original functions **read_cbound()** and **read_freq_band()** functions will be called and definition of the arrays **cbound[]** and **ltg[]** will be defined using the values read from MPEGs tables.

If not defined, the new functions `define_cbound()` and `define_freq_band_v2()` will be used. Definition of `cbound[]` and `ltg[]` will be based on formula approximation. This is recommended for the generalized implementation.

`PMODEF_USE_MPEG_TABLERANGE` When defined, the tables will be defined for 480 spectral lines in `power[]`. This can be used for comparisons to the original MPEG model. In general, this should not be defined. If it is not defined, the tables will always cover the whole range of `power[]`.

`PMODEF_WRITE*` Various parameters can be written to binary output files. In general, these should not be defined.

`PMODEF_PRINT*` Some [table] values can be `printf()`'d when defined; again, these should not be defined in general.

Some additional macros not mentioned here have been introduced for debugging and developmental purposes or have been provided by the original MPEG implementation.

3.7 Minor Changes

The original MPEG-Implementation (as given in `tonal.c`) was able to process stereo signals. This ability has been removed for the generalized implementation.

The generalized implementation is written in ANSI C.

In order to avoid ANSI-C's ambiguity in the representation of values of type `int`, (nearly) all `int`s have been changed to `long`.

4 Properties of this generalized implementation

4.1 Accuracy of formula-based approximation of MPEG's tables

In order to validate the generalized model, the values calculated from the formulas have to be compared to the values given in the tables provided as part of the MPEG distribution. This will be done in the following sections.

4.1.1 `ltg[]`

`ltg[].line` The array `ltg[]` has the elements `ltg[].line`, `ltg[].bark` and `ltg[].hear` being defined from the appropriate table. Since the method of calculating the entries for `ltg[].line` has been adapted exactly to the way it is provided in the tables, there is no difference at all in the `.line`-entries.

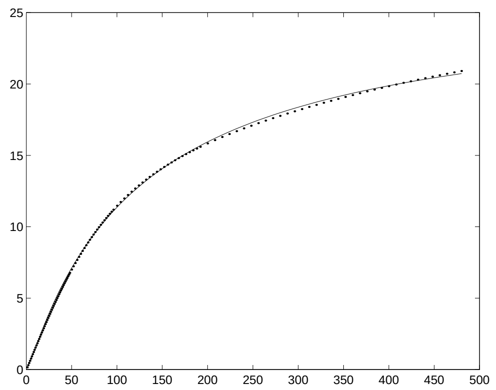
`ltg[].bark` The calculation of the `ltg[].bark`-values is based on the determination of the frequency corresponding to the relative position of `ltg[].line` within 0..`PMO_han_size`-1, which, in turn, represents the frequencies $f = 0...fs/2$ Hz. This frequency value is then converted to Bark using `hz2bark()`.

Figure 1 shows a graphical comparison between the values resulting from reading the table and from using the formula. The differences can hardly be seen in a direct plot of the bark-values itself (Figure 1, right). They can be seen, however, from a direct plot of the the differences (Figure 1, left). The maximum differences are about 0.25 Bark.

`ltg[].hear` The masked threshold in quiet is stored in `ltg[].hear`. The values are calculated using the interpolation function `hearmask()`. Figure 2 shows a comparison of the values given in MPEG's tables and of the values returned by the interpolating function `hearmask()`.

As it can be seen in figure 2, the differences are always smaller than about 3dB. Since the concept of the hearing threshold in quiet itself has a certain variability, a tolerance of about $\pm$ 5dB has to be allowed. The results returned by `hearmask()` are therefore very good.

(The exact value of the hearing threshold in quiet also depends on parameters like the listeners age and attention (see, for example, [6])).



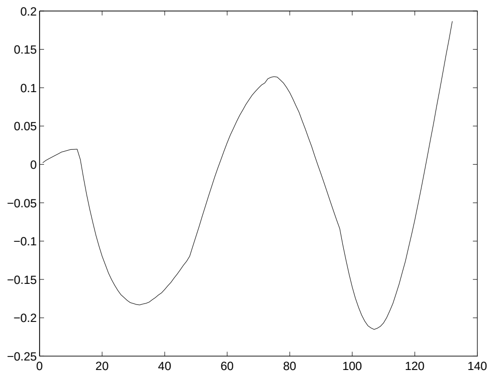


Figure 1: Accuracy of bark-values
left: Plot of `ltg[].bark` as defined in table file (dotted) and as defined by means of equation (3) (line). *right:* Difference of the bark values, calculated as (table)-(formula). (Please note the different scaling of the Bark-axis.)

4.1.2 `cbound[]`

Even though the deviations of bark values shown in Figure 1 seem to be rather small, they lead to some higher differences when scaled to the range covered by `power[]`. The values for `cbound[]` read from table and calculated are:

```
1 6 13 20 27 34 42 50 60 70 80 94 108 124 148 172 208 248 288 344 408 480
1 6 13 19 25 32 40 48 58 69 81 94 110 127 148 172 201 236 280 336 410 479
```

the first row being the values taken from MPEG's table to be used for layer II and fs=16kHz and the second row displaying the values calculated for the same condition according to the method described. The differences are limited to not more than 8 indices in this case, but here they can cause some differences in the result. (A difference of 10 indices corresponds to a Δf = 156 Hz for fs=16kHz and to Δf = 469 Hz for fs=48kHz, when using `PMO_han_size`=512). These values can cause a difference in the resulting masking curve. However, as it will be shown in the next section, the differences introduced by this are usually reasonable small.

4.2 How to calculate the results returned by original MPEG implementation

The generalized function `PsychoMpegOne()` will return the same results as the original MPEG model if and only if the following conditions hold:

- `spectrum_range` = 16,
- `masking_range` = 16,
- `fs` is equal to one of the sampling frequencies allowed in MPEG,
- `PMODEF_CONSIDER_HEAR` and
- `PMODEF_USE_MPEG_TABLERANGE` are the only new switches defined,

The only differences to MPEG's result for the same signal is then due to differerences in the values provided by the formula compared to the values given in MPEG's tables. However, these can be turned off completely by additionally defining

- `PMODEF_USE_TABLE`.

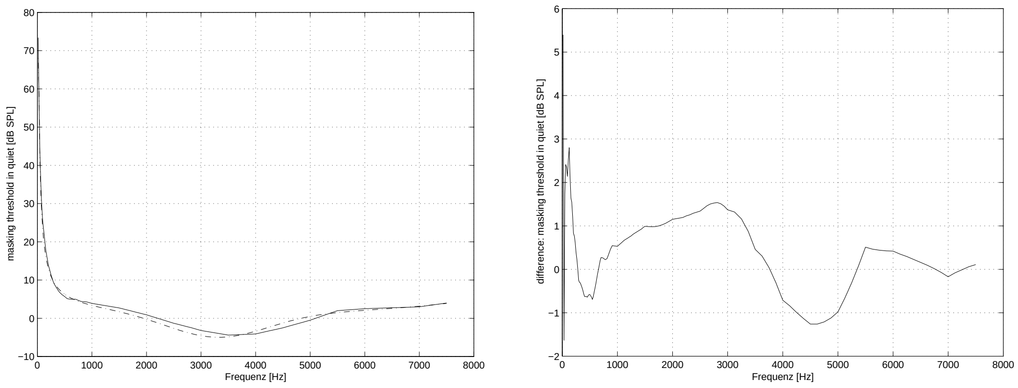


Figure 2: Accuracy of the interpolated hearing threshold in quiet (**`ltg[].hear`**)
left: Plot of **`ltg[].hear`** as defined in table file (dotted) and as defined by means of function **`hearmask()`** (line). *right:* Difference of the hear values, calculated as (MPEG-table)-(interpolated).

4.3 Reasonable values of the `*_range`-parameters

By listening to resyntesised speech that has been reconstructed using the sequence of "diluted" spectra (i.e. all lines being masked according to the model's estimation have been removed), the range of valid or reasonable parameters has been established. Only the combination of parameter values that led to resynthesised signals not showing any audible differences to the original signal are considered valid.

From these "tests" we recommend using an analysis window length of about 20 ms to 40 ms for all sampling frequencies.

4.4 List of general-purpose functions provided in the module PsychoMpegOne

The module described here contains some functions that can be used for more general purposes, too. These functions are listed below.

`double bark2hz (double )` Conversion from Bark to Hertz. Input value is a (positive) Hertz value; the corresponding frequency in Bark will be returned.

`double hz2bark (double )` Conversion from Hertz to Bark. Input value is a (positive) Bark value (within 0 to about 27); the corresponding frequency in Hertz will be returned.

`double hearmask(double )` Returns the hearing threshold in quiet for the frequency specified (in Hertz). The result is returned in dB_{SPL}. The resulting value is determined from a linear interpolation in the (f, dB)-domain.

5 An example application

This section describes an example of an application of this generalized implementation. Input parameters and resulting output values will be described. The results will be compared to the output of the "psychoacoustic model one" of the original MPEG implementation for the same input conditions.

5.1 Input signal and resulting output values

For this particular example, the input signal used had a sampling frequency of fs=16000 Hz. Figure 3 depicts the signal. The length of the signal is 1024 smp (i.e. 64 ms for fs=16000 Hz). The effective length of the analysis window is smaller, since the signal will be Hann-windowed prior to spectral analysis.

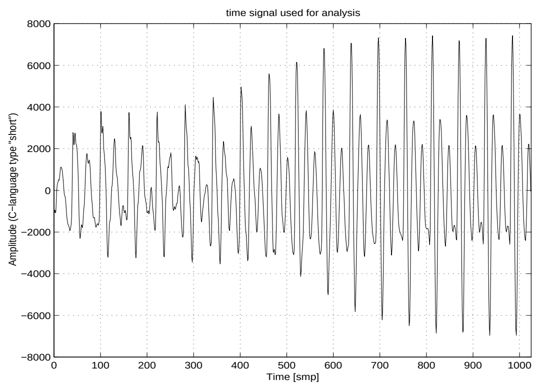


Figure 3: Signal used for the sample application:
Onset of a tonal segment of a speech signal; length: 1024 smp (64 ms; fs=16 kHz); shown prior to Hann-windowing.

Figure 4 shows the power density spectrum of the signal used here. It is the direct output as returned in `spectrum[]`.

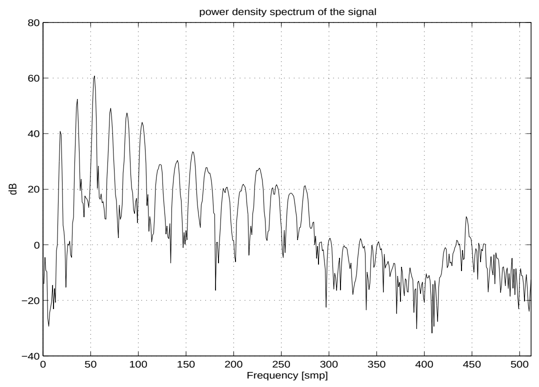


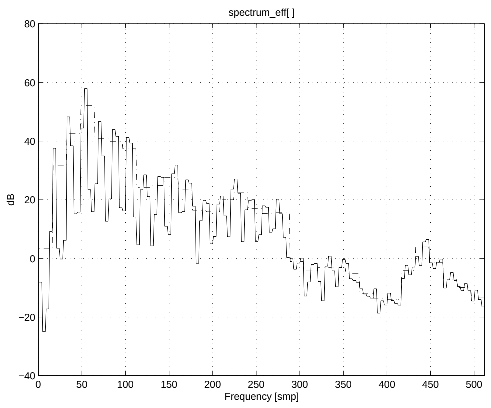
Figure 4: Power spectral density:
Output of a `PMO_fft_size` = 1024-point FFT, resulting in a spectrum with `PMO_han_size` = 512 points. The values are given in dB. This spectrogram is returned in the array `spectrum[]`.

Using this signal, a calculation of the masked threshold has been carried out using the generalized implementation. Since `PMO_fft_size` = `fftsize` = 1024, for each of the arrays for the resulting spectral-like outputs (`spectrum[]`, `spectrum_eff[]`, `masking[]` and `masking_eff[]`) 512 values had to be allocated prior to the function call of `PsychoMpegOne()`.

The parameters `spectrum_range` and `masking_range` have been set to the same value. Two calls of the function `PsychoMpegOne()` have been carried out; the first one using `spectrum_range` = `masking_range` = 16 and the second one using `spectrum_range` = `masking_range` = 4. All other parameters have been left unchanged. So, the first call used the spectral resolution of original MPEG whereas the second call uses a finer spectral resolution.

The left part of Figure 5 depicts the resulting values for the estimation of the effective signal (`spectrum_eff[]`), calculated using the mean() of `spectrum_range` neighboring values. The right part of this figure shows the corresponding values for `masking_eff[]`.

In figure 5 the effects of the `min()`-calculation used for the derivation of `masking_eff[]` can be seen. As mentioned before, this operation can not be "generalized" in a mathematical sense: For values smaller than 16, the estimated masked threshold will always be *above* MPEG's value, and for values larger than 16, it will always be *below*. In this sense, it is different from the calculation of the local `mean()` used in the determination of



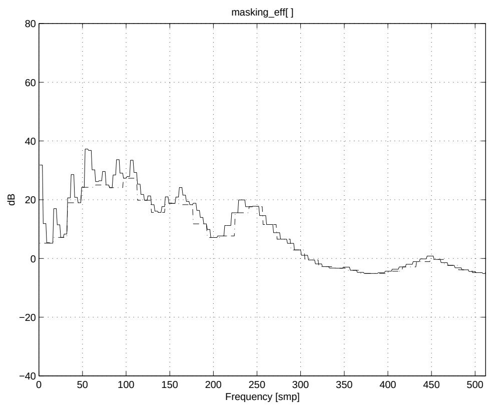


Figure 5: Effective Spectrum and Masking calculated using two different resolutions: *left:* **spectrum_eff[]** for **spectrum_range**=16 (dotted) and for **spectrum_range**=4. *right:* **masking_eff[]** for the same signal and conditions as in the right picture. While the curves of the spectrum are at about the same level, the fine-resolved masking curve is always above the less-resolved curve.

spectrum_eff[]). This operation can be generalized, since the values returned for ***_range**s other than 16 do not tend to be biased towards higher or lower levels.

Therefore, the two curves returned for **spectrum_eff[]** in figure 5 have about the same level. This is not true for the two curves of **masking_eff[]**; the finer-resolution curve always lies *above* the lower resolution curve.

The resulting SMR, as returned in **smr_eff[]**, is presented in figure 6.

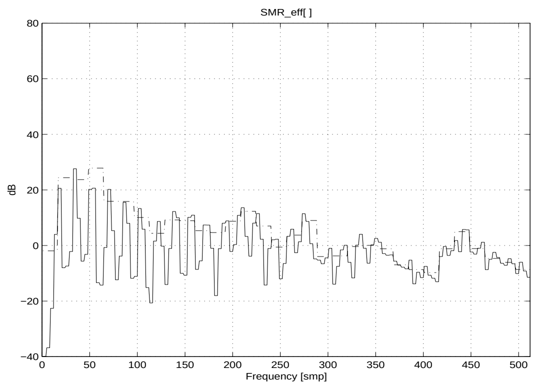


Figure 6: Signal-to-Mask-Ratio: This is the SMR as returned in **smr_eff[]**. Again, this value has been calculated using MPEG's spectral resolution **spectrum_range** = **masking_range** = 16 (dotted) and using an increased spectral resolution **spectrum_range** = **masking_range** = 4 (line).

As it can be seen in figure 6, the estimation of the SMR is influenced by the spectral resolution used. The smaller the value used for ***_range** (i.e. the finer the spectral resolution), the more is the SMR shifted towards smaller levels, resulting in an increase of the number of masked spectral components.

As it has been shown by listening experiments described in section 4.3, even for the finest spectral resolution possible (***_range**=1) the results returned are valid when using the appropriate size of analysis window. Finally, figure 7 summarizes the results by depicting the original spectrum together with the masked threshold **masking_eff[]** as calculated using the two spectral resolutions requested in this example. Again, the masking curve calculated using the higher spectral resolution lies above the lower-resolution curve.

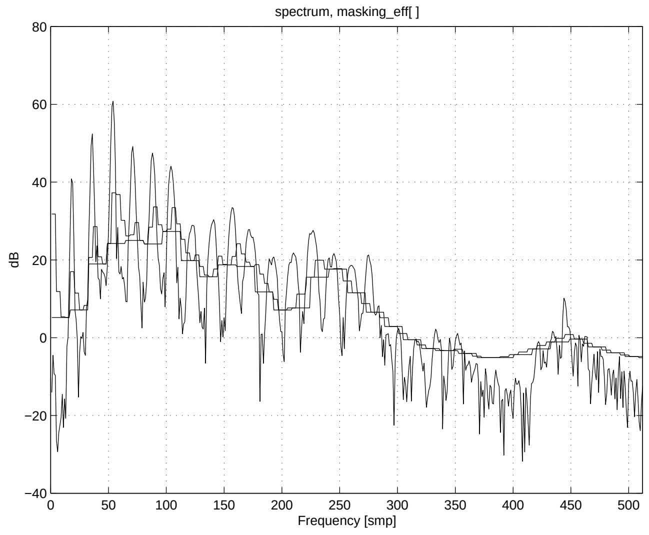


Figure 7: Spectrum and masking threshold **masking_eff[]** at two spectral resolutions: The power density spectrum **power[]** (finest x-resolution curve) and the effective masked threshold (medium and lowest x-resolution curves) are depicted here.

5.2 Comparison to original MPEG's results

In order to determine the impact of the differences caused by replacing the tables by formulas, the original MPEG implementation and the generalized implementation described have been tested. For this comparison, the signal already used in the last paragraph has been used again. The values of **spectrum_range** and **masking_range** have been set to 16.

As already mentioned in section 2.2, the MPEG-Model internally calculates the *global masking threshold* from the input signal's spectrum. This global masking threshold is then reduced to the *minimum masked threshold* using the local **min()**-calculation over 16 neighboring values. It is therefore interesting to compare the resulting global masking threshold as returned by original MPEG to the corresponding result returned by the generalized implementation.

Figures 8 and 9 give a comparison between the results returned by the generalized implementation and original MPEG.

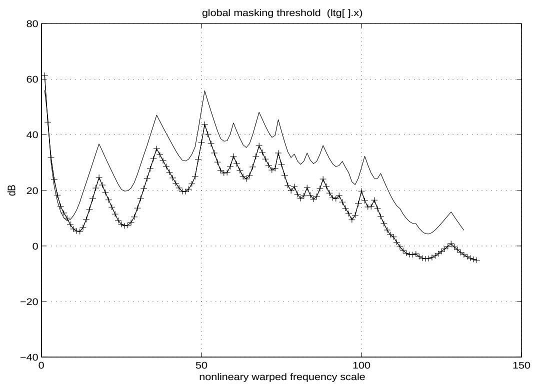


Figure 8: **ltg[].x**: MPEG and generalized implementation
This picture shows the global masking threshold as calculated using the original MPEG implementation (line) and the genaralized implementation described here ('+'-marked line). The values are plotted against the indices of a nonlinear warped frequency scale internally used in both implementations.

As depicted in figure 8, the differences in the calculation of the global masked threshold (calculated using the original MPEG algorithm on the one side and the new implementation on the other) are quite small. The offset of the two curves is given by the 12 dB-offset used to enable `mean()`-based calculation instead of MPEG's `sum()` (see section 3.5): The global masking threshold has been lowered by 12dB in the same way as the calculation of `mean(spectrum)` shifts the values down compared to `sum(spectrum)`.

The first four peaks of the global masked threshold shown in figure 8 correspond to the first tonal peaks in the spectrum (between 20 Hz and about 70 Hz). The following peaks are already defined from a single line representing a range of spectral values. – At the lowest frequencies, the global masked threshold is defined by the hearing threshold in quiet.

Finally, figure 9 shows a comparison of the two SMRs as returned by MPEG and by the new implementation. The differences are limited to about ±3dB.

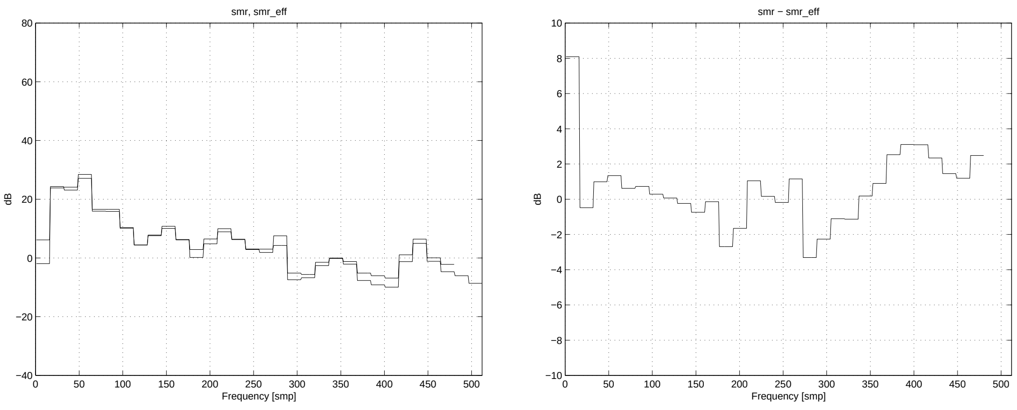


Figure 9: Signal-to Mask-Ratio:
This picture shows the result returned in `SMR_eff[]` and as returned by original MPEG.
left: SMR: Mpeg and generalized implementation *right:* difference between the two SMRs.

6 Availability

The following information is intended to be valid by October 1997:

The source code of the generalized implementation described in this technical report is available via anonymous ftp from ICSI's ftp-server `ftp.icsi.berkeley.edu`. The package is called `PsychoMpegOne.tar.Z`.

7 Summary

This technical report describes the properties of a a new generalized implementation of an algorithm for the calculation of the psychoacoustic simultaneous masked threshold. The generalized implementation was derived from a public-domain implementation of the MPEG-Audio Encoder. The changes and modifications that led to the new implementation have been described.

The new generalized implementation is able to process signals sampled at any user specified sampling frequency. The masked threshold returned by the algorithm is returned in a spectral resolution defined by the user. In addition, the length of the analysis window to be used can be chosen.

This new generalized implementation has been validated using a direct comparison of the results to the corresponding results returned by the original MPEG-Audio algorithm. An informal listening test based on signals modified in the spectral domain has also been carried out to determine the range of reasonable values for the analysis parameters.

The new implementation can be used as a flexible, general-purpose tool for any kind of investigations considering the psychoacoustic simultaneous masked threshold.

8 Acknowledgments

I would like to thank Prof. Nelson Morgan, the head of the Realization Group at ICSI, all members of the Realization Group, and Martin Isenburg for helpful comments and ideas on this report. I would also like to thank Deutsche Telekom Berkom and all my colleagues at Berkom for supporting my stay at ICSI.

References

[1] ISO/IEC International Standard IS 11172-3 “Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbits/s”: Part 3: Audio

[2] Pan, D., “A tutorial on MPEG/Audio compression”, IEEE MultiMedia, Summer 1995, p. 60-74

[3] Shlien, S. “Guide to the MPEG-1 Audio Standard”, IEEE Trans. on Broadcasting, Vol. 40, No. 4, Dec. 1994, p. 206-218

[4] Schroeder, M.R., Atal, B.S. and Hall, J.L. “Optimizing digital speech coders by exploiting masking properties of the human ear”, J. Acoust. Soc. Am. 66(6), Dec. 1979, p. 1647-1652

[5] Traunmüller, H. and Lacerda, F. ”Perceptual Relativity in Identification of Two-Formant Vowels”, Speech Communication, 6, 1987,p. 143-157

[6] Zwicker, E. and Fastl, H., “Psychoacoustics – Facts and Models”, Springer, 1990