Amélioration d'un codeur paramétrique

de ***Massimo Gregorio MUZZI'***

DEA ATIAM 2002 – 2003

Acoustique, Traitement du signal et Informatique Appliqués à la Musique

(Paris 6, ENST, Aix Marseille II, UJF Grenoble I)

Philips Digital Systems Laboratories,
Eindhoven, Pays Bas

Responsable du stage : M. Werner OOMEN

PHILIPS

Table of contents

Table of figures

Index of tables

1. Introduction

A lot of applications nowadays are related with speech and music, or audio in general. For example, the telephone network, television, radio, CDs, DVDs, spoken books, public announcements in big places such as railway stations, airports… The possibility of managing all these applications with a digital and unique architecture is very important. This is true for several reasons, such as standardisations, portability, diffusion, improvement of the final quality, storing…

Audio coding is the process of digitally encoding/decoding audio signals. The goal of audio coding is to obtain a representation of the audio signal that is as compact as possible. After decoding the signal should be as close as possible to the original, from a perceptive point of vew. Scalability is a good property of such a codec, because it is expected that more bits give better quality and the band of the channel can vary during the transmission.

Mainly thanks to the Internet, everybody has heard (or used) something about audio coding, in particular the MPEG-1 Layer III standard, better known as MP3. This compression scheme delivers high quality audio at compression gains of over a factor of 10. Since the standardization of MPEG-1 many new and innovative ideas have been brought forward. In practice, this led to state-of-the-art MPEG4-AAC coders that provide compression-factors of around 15 while still maintaining the same high quality level of the MPEG-1 Layer III. The current opinion in the audio coding community is that no further improvement in compression gains is expected for waveform type of coders. There is a general belief that, in order to achieve even higher compression gains, audio should be coded parametrically.

In 1998 Philips Research started a feasibility study to parametric audio coding of general broadband audio. In order to give a parametric representation of an audio segment, this one was divided into three objects: transients, sinusoids and noise. These objects will be presented with details further in this report. This feasibility study indicated that, using these three objects, high quality audio coding should be possible at bit-rates of around 40kbit/s for stereo signals, i.e. a compression-gain of about $44{,}100 \cdot 2 \cdot 16 / 40{,}000 \approx 35$! With this bit-rate an hour of music can be encoded into 16 MB of RAM.

An implementation has been developed, mainly for mono-signals at a goal of 24 kbit/s. This implementation however does not deliver the desired high quality at the intended bit-rate. Furthermore, from a certain point on, the spending of extra bits does not give an extra increase in quality.

The goal of the graduation project, described in this report, is first of all to assess what exactly causes this lack of quality. Secondly, for the problems that are recognised solutions within the current framework are investigated and evaluated. In the first chapter a general introduction to audio coding is given. In the next chapter SSC, the parametric audio coder developed by Philips Research, is briefly described. Next, the experiments run are described. In the following chapters some solutions are described. Then performance results, in quality and in bit-rate, are described. Finally, we end with conclusions and recommendations.

2. Résumé en français

Dans le cadre du DEA ATIAM, j'ai décidé de faire mon stage de DEA chez les laboratoires de systèmes numériques de Philips en Hollande. Cette possibilité en fait était la plus adaptée à ma formation « internationale » d'ingénieur, et le centre de recherche de Philips en Eindhoven est parmi les plus modernes et importants d'Europe. En plus, le domaine d'application, qui a été le traitement numérique du signal, coïncidait exactement avec mes options du DEA et mes envies.

Philips travaille depuis quatre ans sur une nouvelle technique de compression audio. Cette technique a pour objectif d'atteindre le 24 kbit/s pour des signaux audio stéréo génériques avec une bonne qualité finale. Le temps réel est aussi recherché. Ceci permettra de mettre une heure de musique dans 16 Mbyte RAM. Parmi les applications de ce nouveau standard, appelé SSC (SinuSoïdal Coder), il y a Internet radio, le livre parlé, le DVD, le tempo and pitch scaling rapide, des applications de type MP3 etc… Le principe qui est à la base de ce codeur est qu'il est possible de représenter un signal audio quelconque avec la superposition d'un certain nombre d'objets paramétrables, c'est à dire, des objets qui peuvent être entièrement décrits avec un nombre fini et compact de paramètres. Apres une étude initiale sur la faisabilité du projet, quatre objets ont été retenus : l'objet transitoire, l'objet sinusoïdal, l'objet bruit et l'objet stéréo. Les objets sont calculés sur la base d'une segmentation du fichier audio en subframes de longueur constante, et reconstruit selon la technique appelée « *OverLap and Add* ».

L'**objet transitoire** a pour objectif de représenter les changements rapides dans l'évolution temporelle du signal, dans l'objectif de rendre le signal plus stationnaire à la sortie de l'analyse / synthèse. En particulier, deux types d'enveloppe sont considérés, une première forme à échelon ou une deuxième qui ressemble à une cloche asymétrique, décrite par la fonction de Meixner, voir Figure 13.

L'**objet sinusoïdal** représente toutes les parties harmoniques du signal. Pour gagner en compression les paramètres de cet objet sont transmis avec quantization et avec différenciation avec les précédents. En outre les sinusoïdes sont tracées (tracking), pour avoir une idée meilleure de leur correcte estimation.

Ces deux premiers objets, après leur détection, sont soustraits au signal original. L'idée est que tout ce qui reste, appelé signal résiduel, doit être stationnaire et ne pas contenir de parties harmoniques, donc très proche du bruit.

L'**objet bruit** modélise la partie bruit du signal, en particulier, dans ce cas, les paramètres décrivent l'enveloppe temporale et spectrale du signal.

L'**objet stéréo** modélise la représentation des deux canaux en se basant sur un seul canal, qui peut etre l'issue de une combination des deux canaux gauche et droite, auqueul on ajoute des informations supplémentaires pour reconstruire l'image stéréo. En particulier, la différence interaurale d'intensité et de localisation spatiale et le degré de corrélation sont codés. Pendant toute la durée de mon stage je

n'ai jamais consideré l'objet stéréo, car je tout de suite supposé qu'il n'était pas le responsable d'une perte de qualité. En effet, les problèmes existaient déjà avec des signaux mono.

A la date de mon arrivée, le système existait déjà sous forme d'un long (et lent) programme en Matlab. L'architecture de ce programme est décrite, par exemple, dans la Figure 32. En particulier, celui crée par Philips constitue l'état de l'art en ce domaine, et il est de loin le meilleur codeur à 24 kbit/s sur le marché. Le signal SSC sonne en générale un peu métallique, un peu bruité, mais la parole est toujours compréhensible et la musique sonne en tout cas très dynamique.

Cependant, aux bitrates plus élevés, ce primat est perdu, et en particulier la qualité finale n'améliore pas en augmentant le bitrate, au contraire de ce qui en général est toujours souhaité.

Le problème donc qui constitue le sujet de mon DEA est exactement celui-ci : comprendre quels sont les facteurs theoriques et pratiques qui limitent la qualité du codeur, et donc en trouver et impementer des solutions.

J'ai passé les premiers deux mois à comprendre le code et l'architecture du système, à encoder et écouter des signaux audio avec SSC pour m'habituer à ses caractéristiques, et à trouver et mettre en place des tests et des expériences auditives pour trouver les défauts du système. On ne voulait pas se lancer dans la résolution du premier problème rencontré, mais plutôt avoir une idée de toutes les limitations, et donc essayer de résoudre les plus relevants. En plus, il était important aussi de ne pas changer le format du bitstream, car il est en phase de standardisation avancée par le Moving Picture Expert Group MPEG.

J'ai ainsi testé les performances du module de détection des transitoires et des sinusoïdes, la robustesse du système vis-à-vis du bruit, la cohérence du module psycho acoustique, l'importance de la phase originale, etc. Toutes ces expériences sont détaillées dans le chapitre 6.5.

Les conclusions que j'ai tirées de cette série d'expériences a été tout d'abord que l'algorithme qui contrôlait le bitrate était à modifier : il est clair que les bits ne sont pas donnés aux modules corrects, en particulier on n'obtient pas une meilleure qualité en codant plus de sinusoïdes ou en donnant plus de bits à la quantization des amplitudes. Ensuite, il a été clair aussi que SSC est un système qui sature très rapidement, car il a été optimisé pour 24 kbit/s. Donc, en donnant plus de bits aux trois objets, la qualité elle-même n'améliore pas trop. En particulier la partie « bruit » est modélisée toujours par un bruit blanc filtré et avec une enveloppe temporelle similaire à celle réelle, ce qui ne peut pas approcher que de loin tous les sons de type bruit de la nature. Beaucoup de transitoires, en général, ne sont pas détectés, seulement les plus étroits. Celui-ci n'est pas un grand problème en réalité car le transitoire est le seul objet qui est localisé dans le temps. Il conditionne donc la qualité finale seulement localement.

La détection des sinusoïdes est en générale très bonne et correct, mais pauvre. Cependant, dans un signal de type bruit blanc, où on s'attend à ne pas avoir de sinusoïdes détectées, on obtient au contraire également un grand nombre de sinusoïdes détectées et tracées. Ce fait cause une importante perte un terme de bits.

Pourtant, j'ai considère primaire essayer d'améliorer l'estimation des sinusoïdes.

En particulier, j'ai proposé d'utiliser une version plus riche de l'objet sinusoïdal, et c'est à dire la suivante équation 1:

$$y = \left(A + Bt + Ct^2\right) e^{j\left(D + Et + Ft^2 + Gt^3\right)}$$

Equation 1 - Le nouveau objet sinusoïdal

Les deux polynômes du second ordre donnent à la sinusoïde une adaptabilité majeure aux partiels en détection. En particulier, le polynôme en facteur correspond à des piques avec des queues plus larges, le polynôme en exposant corresponde à des piques avec des pointes plus larges. En outre, l'information sur la dérivée de la fréquence instantanée permet de calculer des traces plus fiables et correctes. Aussi la largeur de la fenêtre d'analyse peut être augmentée, grâce à ces estimations plus riches. Les piques avec une dérivée trop élevée seront considérées comme dérivant du bruit, et donc en générale, pas tracés.

J'ai donc cherché dans la littérature un algorithme qui estimait en même temps tous ces paramètres, et je l'ai implémenté en Matlab. L'algorithme brièvement cherche le minimum local d'une fonction de coût déterminée, le minimum le plus proche à une estimation grossière du pique, qui lui est donnée pendant la phase d'initialisation.

Les résultats ont été très encourageants mais les performances en terme de rapidité du codeur ont chutées, car l'algorithme est très lourd en terme de calcul. En particulier, beaucoup moins de sinusoïdes sont estimées dans un signal type bruit blanc, ce qui permet de gagner des bits, et des traces incorrectes sont éliminées. En outre, ces détections étant plus fiables, il est possible d'agrandir les fenêtres d'analyse. Pour des signaux très harmoniques, cette fenêtre peut être agrandie même d'un tiers.

Dans l'idée encore d'améliorer les traces, j'ai développe un nouveau module, qui cherche à en améliorer l'évolution, en particulier en remplissant des éventuels petits trous. Ces trous sont en générale crées autour des transitoires ou par le module psycho acoustique. En outre, il est maintenant possible de choisir l'intervalle de fréquences pour lequel la phase originale sera transmise. En effet, il est probable que la phase des basses fréquences ne change pas trop rapidement et que nous n'étions pas trop sensibles à la phase des hautes fréquences. Un module qui valide les sinusoïdes estimées a été ajoute aussi. L'idée est que une erreur en détection se transforme dans une double erreur après : cette fausse sinusoide sera en fait presente dans le signal synthetisé et dans le signal residuel, changée de signe.

Apres toutes ces modifications, le résultat est un signal qui sonne beaucoup moins métallique, mais malheureusement plus bruité, surtout dans les basses fréquences. Ceci est dû au fait que la partie

harmonique est maintenant en général mieux représentée, mais moins de sinusoïdes sont tracées. Le signal résiduel qui entre dans le module bruit est donc plus fort en général.

La fin du stage est prévue pour la fin d'août 2003. Pendant ce dernier mois, principalement la direction de recherche sera l'intégration d'un système de segmentation qui puisse donner des informations supplémentaires au module de détection sur la nature du segment (très harmonique ou très bruité etc…).

En outre, le sujet de mon mémoire a été accepté pour être publié dans un article de l'Audio Engineering Society. Dans ce dernier mois, encore, je m'occuperai de sa rédaction.

3. Audio coding

In this chapter the audio coding principles are presented.

3.1 Introduction

Audio coding is the process of encoding / decoding digitised audio signals. This is done in such a manner that after encoding the data rate is kept as low as possible while maintaining as much as possible of the original quality after decoding. In this chapter a brief introduction to audio coding is given.

3.2 Audio coding in general

When referring to the term 'audio', this is meant in a very broad sense. Basically everything that can be perceived by the human auditory system can be described by the term 'audio'. So this includes music, speech but also non-coherent sounds. The only restriction to the term 'audio' in audio coding is the fact that this audio must be (made) available in a digitised form, preferably in Pulse Code Modulated (PCM) format. PCM is very easy to describe: the audio amplitudes are recorded at a fixed sampling rate using a linear quantization on a fix number of bits. The most common form is the CD-standard 16-bits PCM sampled at 44.1kHz in stereo. A basic diagram of an audio encoder/decoder is given in Figure 1.

Figure 1 - General audio encoding / decoding process

As can be seen in Figure 1, the digital PCM waveform is encoded into a so-called bit-stream. The amount of compression, also indicated as coding gain, compared to the original PCM waveform data can be derived from the bit-rate of the bit-stream. The bit-rate indicates the average amount of bits per time-segment (usually seconds) needed to decode one such time-segment of the signal.

The main goal of audio coding can be described in two (symmetric) ways: achieve the highest possible perceptual audio quality for a given bit-rate *or* achieve the lowest possible bit-rate for a given perceptual audio quality level. Although these descriptions seem straightforward it's hard to give a quantitative measure of perceptual audio quality because of its subjectivity. In the Philips Digital Systems laboratories listening tests are often run with at least ten persons and on different excerpts, in order to have an average estimation of the quality of the codec tested.

There are basically two steps in audio encoding. The first step consists of the removal of irrelevancy. Irrelevancy is the part of a message that can be eliminated, without affecting its comprehension. *In ths sntnc I hv elmnt sme irelvant infrmtn.* From the original signal only the relevant parts, the parts that can

be perceived, are kept for further encoding. This implies that the removal of irrelevancy is not lossless: it is not always possible to obtain again the removed part. In the sentence that I have proposed as an example there might have been some other words that could have been removed entirely. The second step consists of the removal of redundancy. The removal of redundancy can affect the comprehension of the ciphered message, but it is always possible to draw back the original signal. So, in contrary to the removal of irrelevancy, it is a so-called lossless process. Coding schemes like e.g. Huffman or Tunstall coding exploit the redundant information in order to come to a more efficient representation.

There are many applications in which audio coding can be used advantageously. Some examples are mobile telephony, Internet radio, solid-state audio players, etc. Basically everywhere where digital audio has to be stored, in case of storage restrictions, or has to be transmitted, in case of channel restrictions, audio coding can be used to a certain advantage.

Of course audio coding also has some drawbacks. These drawbacks are mostly described in terms of some important properties:

- **Audio quality**: the perceived audio quality with respect to the original signal;
- **Complexity**: the amount of processing power/time needed for encoding or decoding;
- **Delay**: the amount of delay, usually expressed in samples, caused by the encoder/decoder system;
- **Bitrate**: the average amount of bits needed to decode a time-unit of the signal, usually expressed in kilobits per second.

Furthermore the cascading of audio coding systems will further degrade the audio quality.

Not every application has the same requirements, for e.g. mobile telephony it is very important that real-time encoding with very low encoding/decoding delay is possible.

3.3 Waveform coding

One of the more classic ways of audio coding is referred to as waveform or transform coding. A schematic description of waveform coding is given in Figure 2.

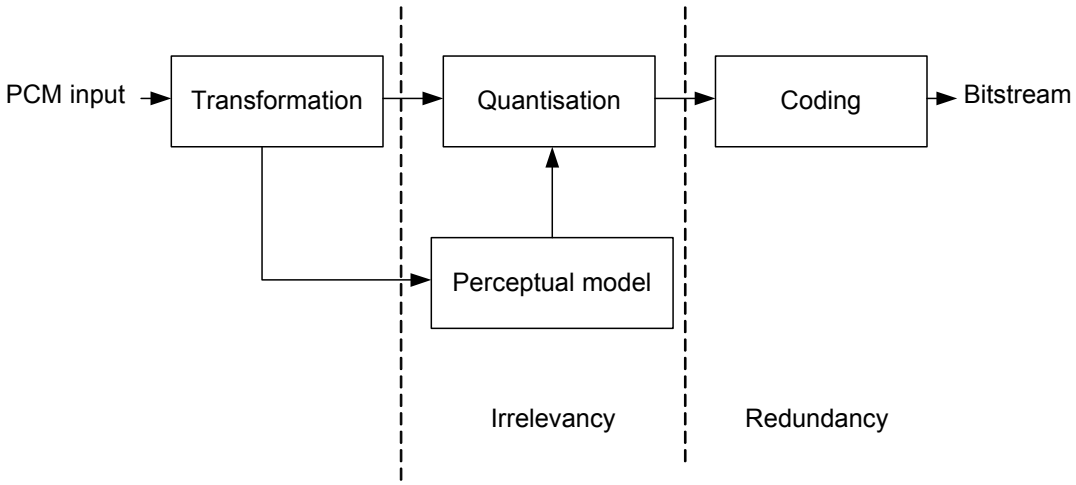

Figure 2 - Waveform coding

A waveform coder is characterised by the transformation stage. First the PCM input signal gets transformed to the frequency domain by means of a filterbank like e.g. an FFT, a polyphase filterbank or a Modified Discrete Cosine Transform. The PCM input signal is also fed into a perceptual model. This model determines which frequency components can be perceived and which cannot by means of a so-called masking model. This information, together with the transformed PCM data is given to the quantization module. In this module the frequency components are quantized, which inevitably results in quantization noise. Inside this module, the best possible quantization for each frequency component is determined given the masking information from the perceptual model and the amount of bits that may be spent. Finally the quantized values are efficiently coded by removing the redundancy that's still present in the quantized frequency components.

A lot of research on waveform coding has already been done. This makes the theory behind such coders well established. However progress in the amount of data-rate reduction achieved by waveform coders almost seems to be saturated. In order to come to an even lower bit-rate for the same high quality audio, it seems sensible to look for another description of audio.

3.4 Parametric coding

When the input signals are restricted (e.g. to speech only) specific features of the input signal can be exploited to further improve the coding gain. One way of implementing these features in an audio coder is to use a parametric coding scheme. The main difference with a waveform coder is the explicit usage of a source model, for example a speech model could be based on the human vocal tract. For a parametric model of a piano the hammers, the snares and the cabinet could be considered. Figure 3 depicts a schematic description of a parametric audio coder.

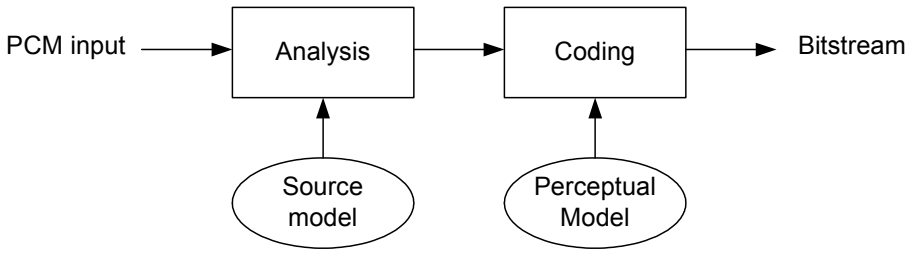

Figure 3 - Parametric audio encoder

The main advantage of parametric audio coding is the exploitation of both the sender-end (source of sound) as well as the receiver-end (human auditory system), where the waveform coder exploits the receiver-end only. If the input signal however does not appropriately fit the source model this might lead to unpredictable results. This lack of robustness then again is the main disadvantage of parametric audio coding.

Another advantage of parametric audio coding is the perceptual model. This model can be adapted fully to the source model. If e.g. one object of the source model is defined as a set of harmonics, perceptual experiments using harmonics could be performed in order to come to a psycho-acoustic model for harmonics only.

Parametric models are also often called object-oriented models. Objects can be a bit abstract like for speech: 'tonal elements' and 'non-tonal elements' but also concrete like 'snare generated harmonics' of a guitar model. A description in terms of different objects automatically implies that within the encoder, decisions will have to be made about what part of the signal must be assigned to what object. The more such decisions have to be made the worse the robustness will probably be.

All in all, parametric coding has more potential than waveform coding as far as bit-rate versus perceived audio quality is concerned. But, especially when designing a parametric model for a broader type of input signals, one has to consider robustness versus coding efficiency.

4. Sinusoidal Coding

In this chapter the sinusoidal audio coder, developed by Philips Research is presented.

4.1 Introduction

In 1998 Philips Research started the 'Sinusoidal Audio Coding' project in collaboration with IPO, Delft Technical University and the Royal Institute of Technology. This project aims at developing a low bit-rate audio codec that gives high quality output at bit-rates substantially lower than defined in the MPEG1 and MPEG2 standard. Within Philips, this project is also referred to as 'Sinusoidal Coding of Audio and Speech' (SiCAS) or 'Sinusoidal Coding' (SSC).

In this chapter a description of SSC is given with the status at the beginning of the graduation project.

4.2 SSC project goals

In the first stage of the SSC project some qualitative goals, requirements and conditions have been determined [den Brinker and Oomen 1998]. Quantitative goals are hard to give because there exists no reliable method for measuring perceived audio quality. The main goals, requirements and conditions are:

- **Input**: bandwidths from 4kHz for narrow band speech to 24kHz for high quality audio;
- **Quality**: multiple quality levels, from medium quality to comparable to MPEG1 Layer III (mp3) at 128 kbit/s to transparent quality;
- **Bit-rates**: multiple bit-rates ranging from 2 kbit/s for intelligible speech up to 40 kbit/s for stereo CD-like quality audio;
- **Bit-rate scalability**: the bit-stream can be seen as a layered collection of streams each representing a quality improvement over the other;
- **Low delay**: useful for point-to-point streaming applications;
- **Graceful degradation**: quality must gradually decrease with decreasing bit-rate.

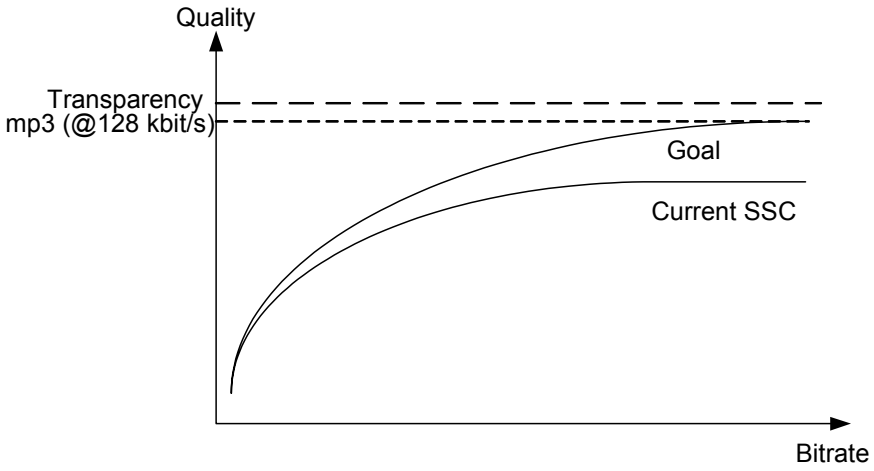

Figure 4 - Quality scalability of the SSC coder

Later on in the project, after the first generation encoders and decoders were produced, one of the main problems with SSC proved to be the quality scalability. From a certain bit-rate on the quality does not increase anymore. This is shown graphically in Figure 4.

The goal of the graduation project described in this report is to address this problem. More specifically the goal is to identify and analyse the current problems and investigate possible solutions.

At the start of the graduation project it was only possible to encode mono PCM signals sampled at 44.1kHz. The quality of the encoded material was however mediocre. These conditions have been taken as the point of departure for the graduation project. Furthermore all the encoder and decoder source code was written using Matlab.

4.3 SSC description

The SSC coder is a parametric audio coder that aims at coding of audio in a broad sense, i.e. unrestricted to a classified source like e.g. speech. Therefore a source model must be developed that is both simple and effective. To do so, one first has to identify the different auditory occurrences that are present within audio signals. To illustrate these occurrences Figure 5 and Figure 6 show spectrograms of respectively a harpsichord signal and a castanets signal. A spectrogram is basically a time-frequency plot with on the x-axis the time and in the y-axis the frequency. In both figures a lighter grey value indicates a higher signal power.

When looking at both figures one can easily recognise three phenomena that are quite common for audio signals in general.

1. **Horizontal lines**: these appear to be deterministic by nature. In Figure 5 these lines represent individual harmonic lines generated by the strings of a harpsichord. They are mostly characterised by their instantaneous frequency.

2. **Vertical lines**: these also appear to be deterministic by nature but are characterised mostly by their placement in time. Such lines indicate transient phenomena like attacks and steps in the audio signal. Figure 6 and to a less degree Figure 5 clearly show such phenomena.

3. **Noise-like areas**: there also seem to be areas that do not have a specific structure but are more stochastic by nature. Figure 6 clearly shows such areas. These areas are characterised as being noise-like.

Unknowingly, we have now classified audio into three objects, namely sinusoids, transients and noise. Such a description seems to be both simple as well as complete.

Another reason for choosing these objects as the base for a parametric audio model is the apparent relation to psychoacoustics. When e.g. calculating the masking curve in most perceptual models of waveform coders a distinction is made between tonal elements (sinusoids) and non-tonal elements (noise). The masking effect of sinusoidal tones is different from the masking effect of noise.

Also, when looking at state-of-the-art audio codecs (encoder/decoder) like AAC or MPEG-1 Layer III (MP3) [Brandenburg and Stoll 1992, Pan 1995] a distinction between transients and non-transients is being made by means of window switching. Window switching can lead to a locally increased time-resolution. This is needed for describing transient-phenomena because of the characterisation in the time-domain.

Furthermore, most experiments that are performed in the field of psychoacoustics are done using simple tones (sinusoids) and narrow-band or broadband noise. Basically psycho-acoustic models can be fit quite well to a description using those objects.

A third and final reason for choosing the objects is the assumption that such a description of audio will be most effective.

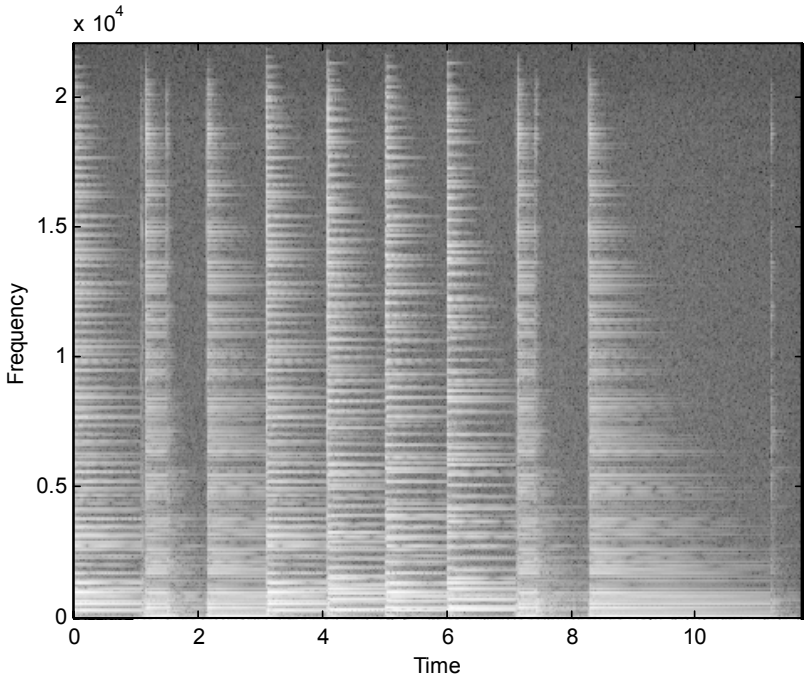

Figure 5 - Spectrogram of harpsichord signal

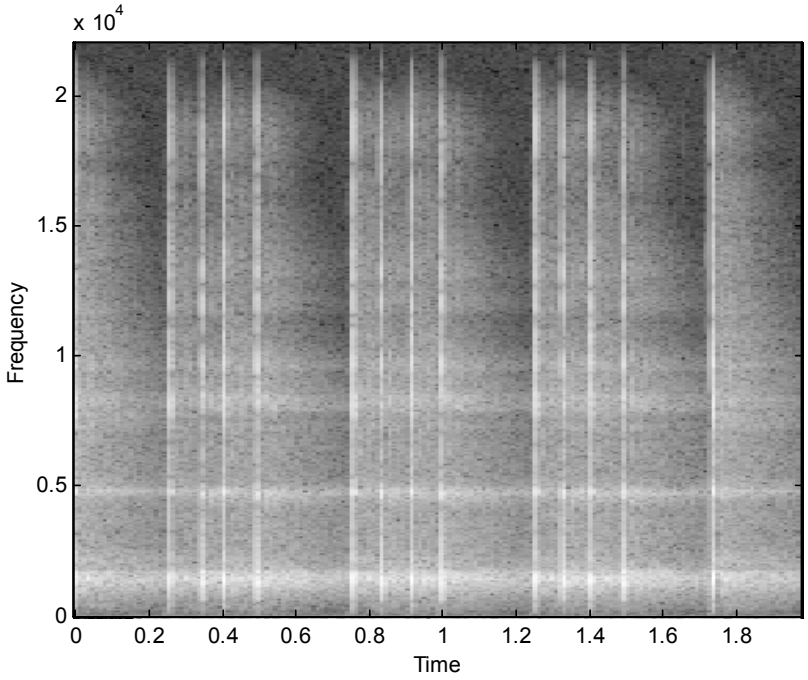

Figure 6 - Spectrogram of castanets signal

The SSC codec is based on the three objects described above, namely: transients, sinusoids and noise. A block-diagram of the SSC-encoder is shown in Figure 7.

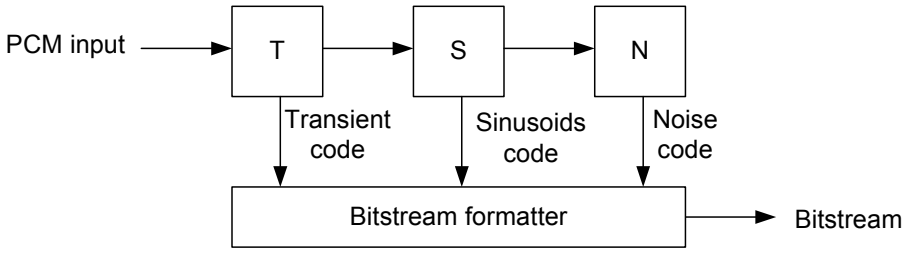

Figure 7 - Block diagram of the SSC encoder

One of the most crucial steps in parametric audio coding is the subdivision of different parts of the signal to different objects. For both the analysis of sinusoidal components as well as for analysing noise, quasi-stationarity is a prerequisite. So if transient phenomena could be removed first the residual signal will be more stationary and thus easier to analyse. This is the main reason why the transients module (T) has been placed in front of the other two. The sinusoidal module (S) is placed before the noise module (N) because it is much harder to analyse and remove the noise from the residual signal of the transient module than it is the other way around.

4.4 Transients

It's hard to give a definition of transients when referring to audio signals. However, a transient mostly has one (or both) of the following properties:

- The signal characteristics in the time or in the frequency domains just before and just after the transient are different. In this way a transient can be seen as a transition between auditory events.
- The transient itself consists of a short burst of signal energy.

A large class of signals falls under the above properties, e.g. attacks, steps, clicks, snaps, etc. This already indicates that a common model for transients is hard to give. However, within the SSC project an attempt has been made to provide such a model.

The block-diagram of the transient module is described in Figure 8. It consists of three blocks: transient analysis (TA), transient synthesis (TS) and transient quantization (TQ).

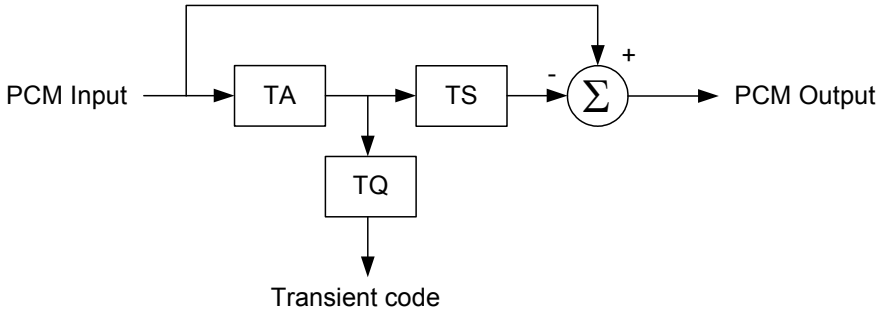

Figure 8 - Block-diagram of the transient module

In the transient analysis block, transients are detected and analysed into a set of parameters. In the transient synthesis block the parameters that have been found by the transient analysis block are used to synthesise the transients. The signal generated by the transient synthesis block is subtracted from the original signal to form a residual signal, which is fed to the sinusoidal module.

Finally the transient quantization block performs both coding and quantization of the parameters that have been extracted by the transient analysis block.

The transient analysis module also performs a first stage of segmentation of the input signal. In Figure 9 a PCM waveform is shown that clearly contains transients. The transient analysis module detects these transients and divides the original PCM waveform into smaller segments that are processed more or less individually at a later stage. This segmentation is in correspondence with the first transient property, namely that before and after the start of the transient characteristics of the signal can be different. It is therefore more efficient to divide the original signal into segments that are more or less quasi-stationary and code each of these segments individually.

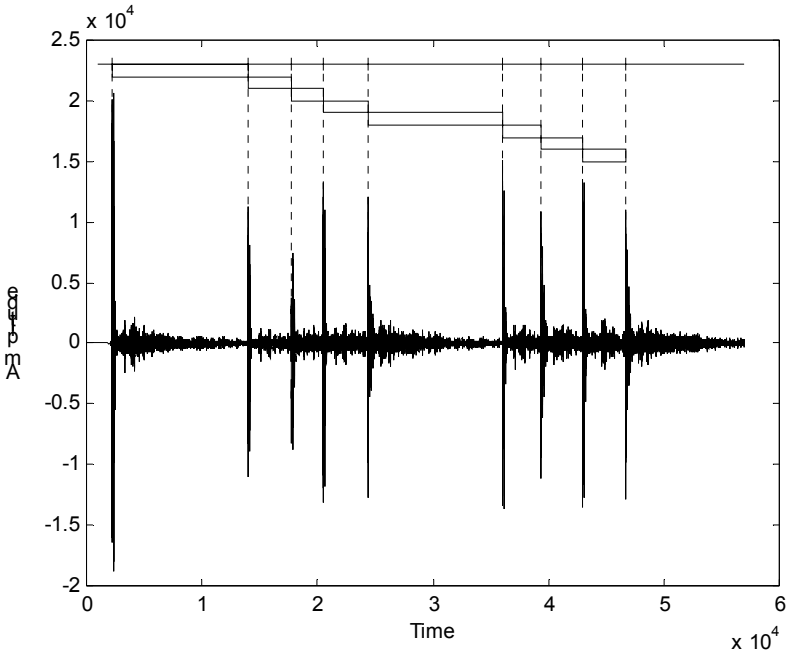

Figure 9 - First stage of segmentation for PCM waveform of castanets

4.4.1 Transient analysis

Because of the analysis-by-synthesis structure of the SSC encoder it is important that transient positions and parameters are extracted properly. The segmentation and stationarity of the residual signal are of great importance for the performance of the sinusoidal and noise module behind the transient module. A block diagram of the transient analysis block is given in Figure 10:

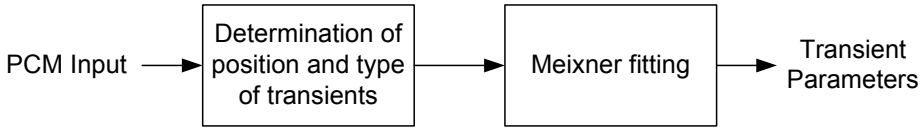

Figure 10 - Block diagram of transient analysis block

First of all the position and type of the transients is determined from the PCM input signal. A choice is made between either a step-transient or a Meixner-transient [den Brinker and Oomen 1998], a transient of which the envelope is described by a Meixner function. A step-transient is described solely by its position. This type of transient corresponds to the first transient property of being a transition between two more or less quasi-stationary segments.

The Meixner-transient corresponds mostly to the second property of being a short signal-energy burst. This type of transient is therefore not only characterised by its position but also by a few parameters to describe the short burst.

The determination of positions and types of transients is described in Figure 11.

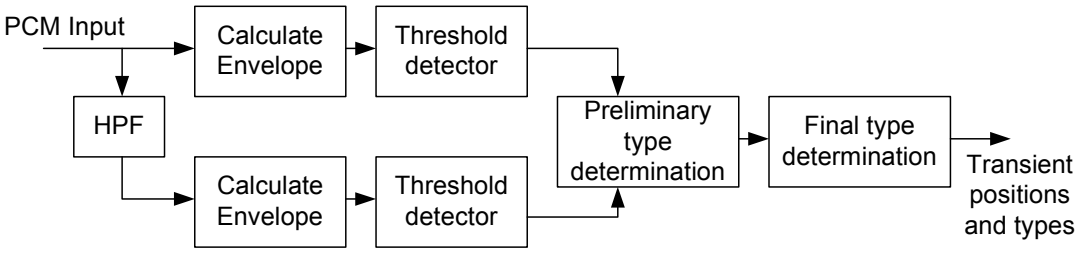

Figure 11 - Block diagram of determination of position and type

For both the original PCM signal as well as a high-pass filtered version of the original signal an envelope is calculated. This envelope is calculated as (Equation 2.1):

$$e[n] = \max\left\{|x[n]|, \tau e[n-1]\right\}, \qquad 2.1$$

where $e[n]$ describes the envelope function and $x[n]$ the input signal, both as a function of time n, and τ is a time-constant between zero and one. As an example the envelope for the castanets excerpt from Figure 6 is given (see Figure 12).

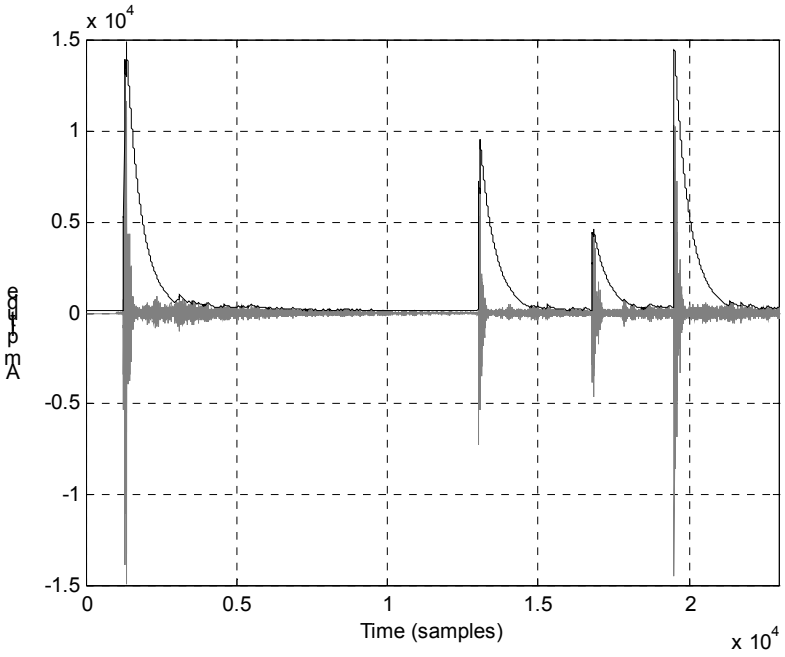

Figure 12 - Envelope for castanets excerpt (black line)

In this envelope, function jumps that are greater than a certain predefined threshold value are sought. The positions of these jumps are indicated as candidate transient positions.

This same process is also applied for the high-pass version of the original signal because of signal energy that may be present in the lower frequency areas. Low frequency components that are perceptually relevant in practice typically have high amplitude. Calculating the envelope of signals that have much signal energy at the lower frequencies can easily mask transients present in the higher frequency area.

The candidate positions of both the original and the filtered signal are then used to determine the exact positions as well as a preliminary type-description of the transients. Transient positions that have only been found in the filtered version are always characterised as being a step-transient at this point. For all other positions the type has still to be determined.

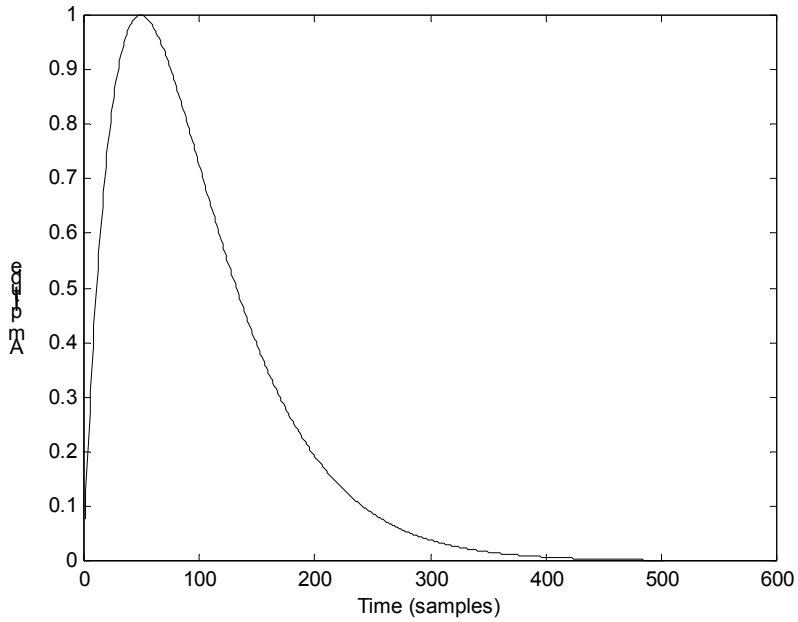

Figure 13 - Typical example of a Meixner-envelope

Whether a transient can be classified as being either a Meixner-transient or a step-transient is determined first of all by the possibility to fit a Meixner-envelope on the input signal at the transient-position. If such a fit can be made to the input signal a transient is preliminary classified as being a Meixner-transient. A typical example of a Meixner-envelope is given in Figure 13.

A description of the Meixner-fitting process is given in Figure 14.

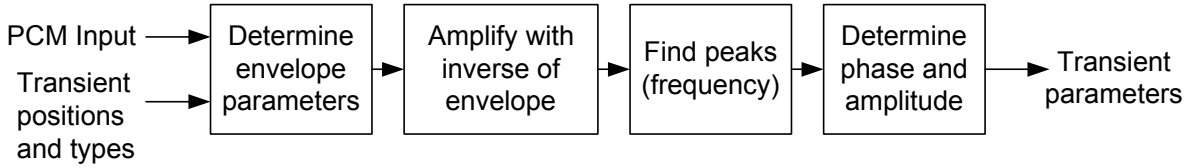

Figure 14 - Block diagram of Meixner-fitting

If the envelope can be fit to the data-segment this segment is amplified with the inverse of the envelope in order to make the transient as stationary as possible. By means of an FFT and interpolation techniques the frequencies of the highest peaks of the frequency spectrum are estimated. Finally for those frequencies found a sinusoidal fit is being made. If the amount of energy reduction achieved with such a fit however is below a certain threshold the type is still finally set to a step-transient.

4.4.2 Transient synthesis

The synthesis of the transients is the inverse process of the analysis section. For step-transients no signal has to be generated. For Meixner transients first of all the sinusoids are regenerated from the extracted parameters and summed together. Thereafter, the Meixner-envelope gets regenerated and multiplied with the summed sinusoids.

4.4.3 Transient coding

All in all, the following data are extracted per transient:

1. Transient position
2. Transient type
3. Meixner envelope parameters (2 parameters, if applicable)
4. Sinusoidal parameters: frequency, amplitude and phase (maximally 8 sinusoids, if applicable)

The sinusoidal parameters are coded differentially where possible in order to decrease bit-rate. In order to do so first the frequencies, amplitudes and phases of the sinusoids are converted to representation levels. These representation levels represent quantized values of the input variables. For frequencies, those quantization levels are related to the ERB-scale. This is a scale that closely matches the way the human auditory system perceives frequency. This is explained in more detail in paragraph 2.5.2. Amplitudes are quantized logarithmically which also corresponds to the sensitivity in the auditory system. Finally, the phase values are quantized uniformly to five bits, an experimentally determined value. When all representation levels are determined, the sinusoids are sorted from low to high frequency. For the first sinusoid the absolute values of the representation levels are coded in the bit-stream. For all following sinusoids the amplitude, frequency and phase representation levels are coded differentially with respect to the previous sinusoid in the bit-stream.

4.5 Sinusoids

The SSC project is based on the assumption that any digital audio signal can be described adequately by (Equation 2.2):

$$x[n] = \sum \text{transients} + \sum \text{sinusoids} + \sum \text{noise}. \qquad 2.2$$

It is however not clearly defined what a sinusoid is. Within the SSC project elements that are classified as being a sinusoid can be described by (Equation 2.3):

$$s[n] = \sum_p A_p(n)\cos\left(\omega_p(n)n + \varphi_p\right), \qquad 2.3$$

where $A_p(n)$ is the slowly varying amplitude, $\omega_p(n)$ is the slowly varying frequency and φ_p the phase of the p^{th} sinusoid. This representation was first used by McAulay and Quatieri for describing speech signals [McAulay and Quatieri 1986].

It is neither efficient nor feasible because of complexity to extract $A_p(n)$, $\omega_p(n)$ and φ_p on a sample-by-sample basis. A more feasible method would be to extract these parameters on a frame-to-frame basis. So for a single frame the sinusoids could be described by:

$$s_k[n] = \sum_p A_{p,k}\cos\left(\omega_{p,k}n + \varphi_{p,k}\right), \qquad 2.4$$

where k indexes the frame and p the p^{th} sinusoid.

In order to come to such a description another segmentation takes place on the PCM input signal. At this segmentation stage the space between two transient-positions is divided into overlapping frames of 720 samples. This number has been determined experimentally as a balance between stationarity and efficient coding of parameters. The segmentation is illustrated in Figure 15. The upper line describes the transient positions extracted by the transients module. The small blocks at the lower end show how the sinusoids are segmented between these transient positions. It is noted that the last frame of a segment is always placed in such a way that it ends just before another segment starts. This is also because of stationarity as described before by the first transient property.

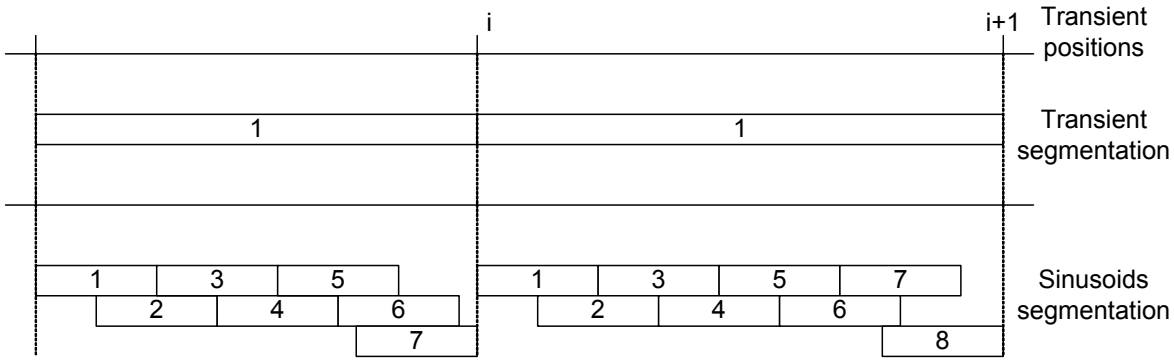

Figure 15 - Segmentation of transients and sinusoidal module

The sinusoidal module can now be described as in Figure 16. It first of all consists of a sinusoidal analysis block (SA). In this block the sinusoidal parameters are estimated on a frame-to-frame basis. The sinusoidal synthesis block (SS) synthesises the sinusoidal signal from these parameters. Finally this signal is subtracted from the residual signal of the transient module to create a presumably noisy signal for the noise module.

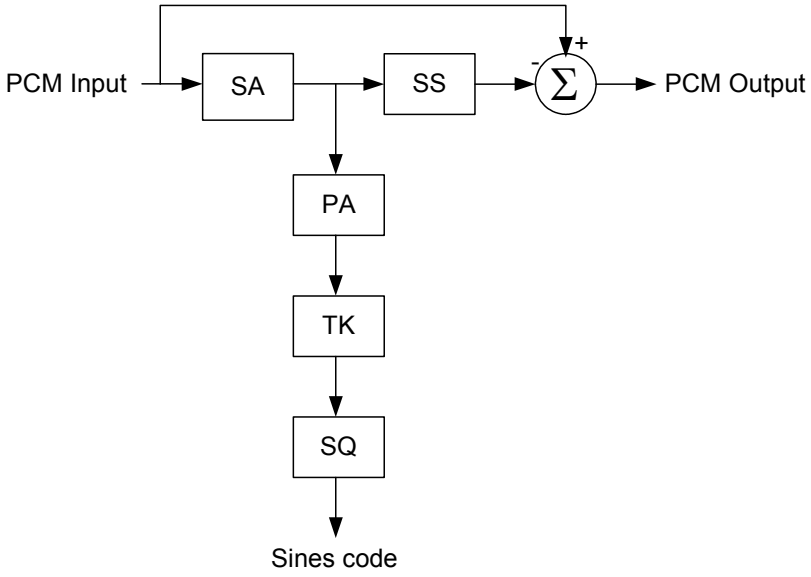

Figure 16 - Block-diagram of sinusoidal module

Of course not all sinusoidal components that have been extracted are perceptually relevant. Inclusion of such components in the bit-stream is superfluous. Therefore a psycho-acoustic model (PA) is used to remove sinusoidal components that fall well below the masking threshold. The reason that the psycho-acoustic model is not used very tightly lies within the tracking block (TK).

To come to an efficient representation of all the individual sinusoidal components found over all the analysed frames a tracking algorithm (TK) is used. The main idea behind this algorithm is that sinusoids in general last longer than only a single frame and can thus form tracks. Differential encoding of e.g. amplitude and frequency can then prove to be efficient. This differential coding is the reason that the restraints of the psycho-acoustic model are set loosely. It is more efficient to code a long track differentially, even though it can't be perceived during the whole length of the track, than only encode the short relevant parts.

Even more coding gain can be achieved by applying phase less reconstruction. This means that instead of updating the phase of a track at each frame only the phase of the birth of a track is encoded. For all following frames of the track the phase is calculated based on the presumption that the instantaneous frequency is a smooth and slowly-varying function in time.

Finally the sinusoidal quantization (SQ) block codes the processed parameters to further increase coding gain. It does so by quantization and coding of the parameters extracted for frequency, amplitude and phase.

4.5.1 Sinusoidal Analysis

The sinusoidal analysis block consists of an iterative algorithm for extracting the sinusoidal parameters [den Brinker and Oomen 1999]. The block diagram is depicted in Figure 17.

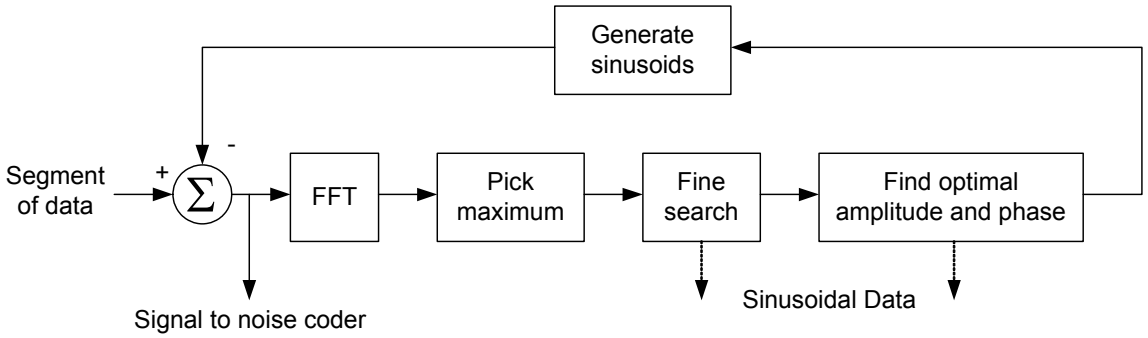

Figure 17 - Extraction of sinusoidal parameters

During the first iteration a segment of data (frame) is presented to the sinusoidal extraction block. The FFT is determined after which the maximum amplitude of the FFT is sought. A rectangular window is used because such a window has the smallest main lobe width. However the main disadvantage of a rectangular window is the side lobe attenuation which causes spectral smearing. It is because of this smearing that the extraction process has to be done in an iterative way.

For the maximum found in the FFT a fine search by means of interpolation is made in order to precisely extract the frequency of the underlying sinusoid. When the frequency has been determined the optimal amplitude and phase can be determined by use of linear regression. Finally the sinusoid is generated using the extracted parameters and subtracted from the original segment. This process is repeated so that finally fifty frequencies with accompanying amplitude and phase are extracted per frame. It is noted that this method of extraction does not consider whether a spectral peak is actually the result of a sinusoid.

The use of only a short segment length of 720 samples implies that the lower frequencies can't be estimated with high precision. Therefore a multi-scale sinusoidal extraction mechanism has been developed (see Figure 19) [den Brinker and Oomen 1999]. The basic principle of this mechanism is as follows.

First the PCM input signal is fed to an anti-aliasing filter (AAF) after which the signal is downsampled by a factor three (DS3). Now the structure of Figure 17 is applied to 720 samples in the downsampled domain (SE). These samples correspond to three times 720 samples in the original domain. An appropriate segment of downsampled PCM samples that corresponds to the samples in the original domain will therefore have to be selected. This is done in the sinusoidal segmentation unit (SU). This segmentation is shown graphically for all three scales in Figure 18. Note that every segment on the third scale corresponds to multiple segments on the second scale. Likewise every segment on the second scale corresponds to multiple segments on the first scale. Also note that the segments on the second

and third scale are always placed within two transient positions just like the segments (frames) on the first scale.

On the third scale three sinusoids are extracted, on the second scale seven sinusoids and on the first scale forty. When less than three sinusoids are found in the third scale, the second scale may extract seven sinusoids plus what has been left by the third scale. The same applies to the second scale and the first scale.

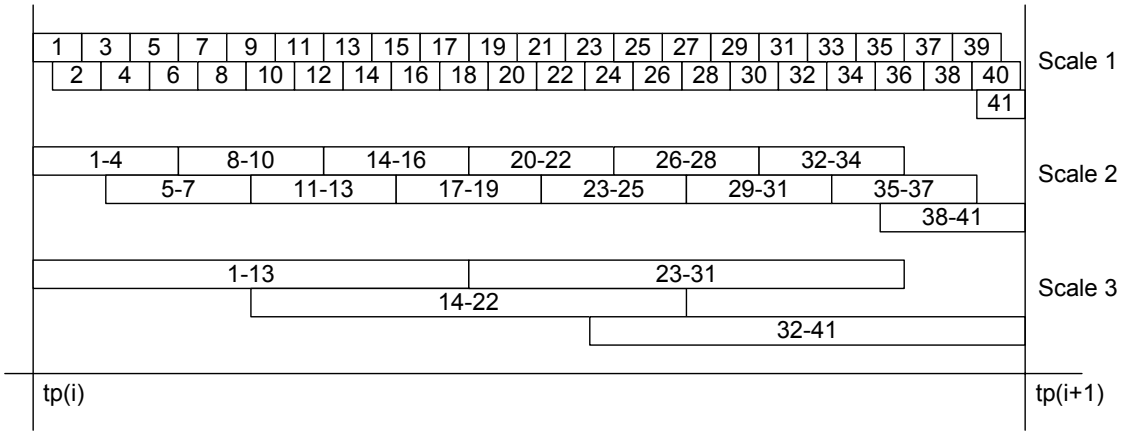

Figure 18 - Multi-scale segmentation

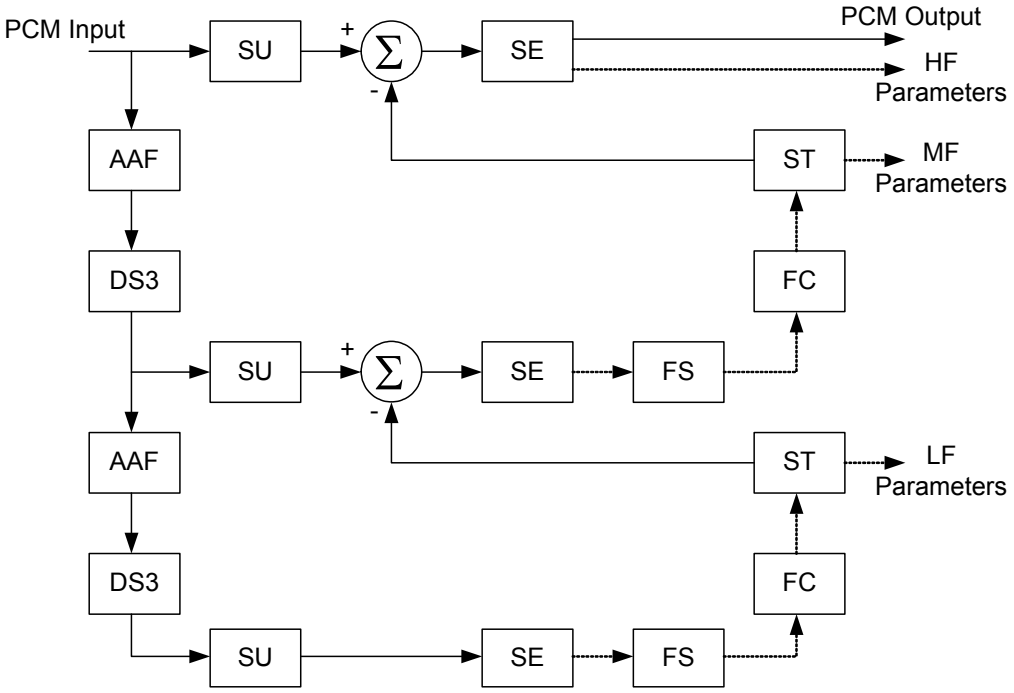

Figure 19 - Multi-scale sinusoidal analysis

Because of complexity reasons in both encoder and decoder a description on a single scale is preferable. To come to a single-scale description the estimated parameters must be converted back to the original non-downsampled domain. This is done in a few steps. First of all every scale is band-limited and thus only a limited amount of frequencies may be included. This selection is done in the frequency selection module (FS). For sinusoids that are kept, compensation in amplitude and phase is made for the gain and delay caused by the anti-aliasing filter (FC). Finally the sinusoidal components are converted to the non-downsampled domain by mapping the parameters to the segments (frames) of the previous scale (ST).

4.5.2 Psycho-acoustic model

The key element in removing irrelevancy in audio coding is the psycho-acoustic model. This model tries to describe, given a certain input-signal what parts of that signal can and cannot be perceived by the human auditory system. Psycho-acoustic models can be very comprehensive. They can describe time masking, the masking of parts of the signal over time, frequency masking, the masking of frequency components over each other, stereo masking/unmasking, the masking or unmasking caused by the use of stereo, etc.

The psycho-acoustic model currently used in the SSC encoder only includes a frequency masking model. Every sinusoid can be seen as a frequency component with its own masking ability determined by its power and frequency. All these components together form a so-called masking curve. This is a curve in the frequency domain that describes the total masking ability of all components present. As an example the masking curve of three sinusoids with frequencies of 1000, 2000 and 4000Hz has been calculated (see Figure 21). Note that the individual masking curves get broader as the frequency gets higher. This effect, as many other laws in psycho-acoustics, are related to the critical band concept [Zwicker 1961]. This concept basically states that the human auditory system can be seen as a bank of band pass filters with different bandwidths. This concept is described by the Equivalent Rectangular Bandwidth scale (ERB) defined as Equation 2.5 [Moore 1989]:

$$e_f = 21.4 \log_{10}\left(\frac{4.37 f}{1000} + 1\right), \qquad 2.5$$

where the frequency f is in Hertz and e_f the frequency in erb. The ERB scale is depicted in Figure 20.

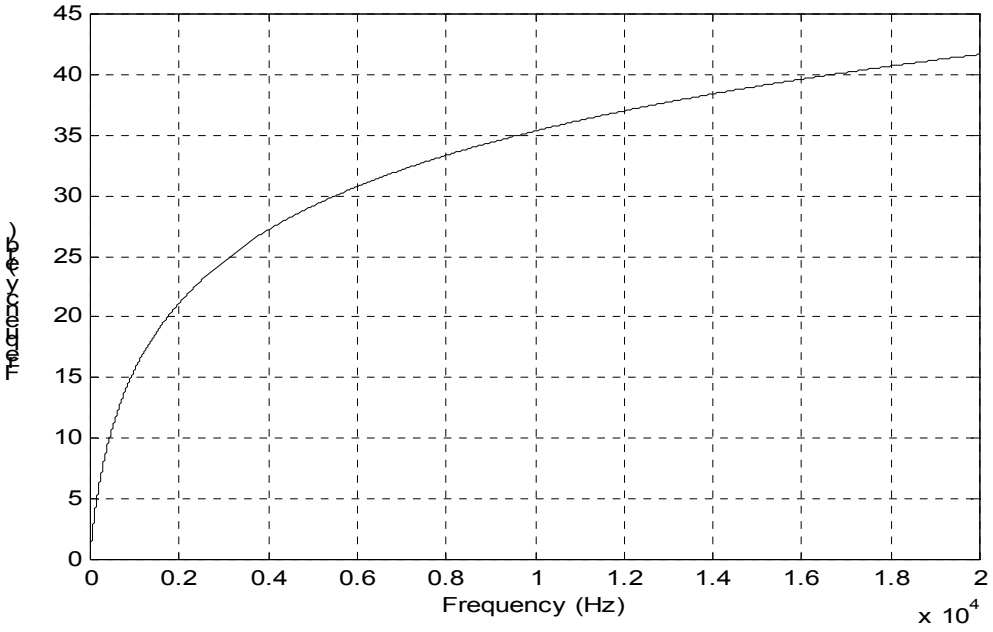

Figure 20 - ERB scale as a function of frequency

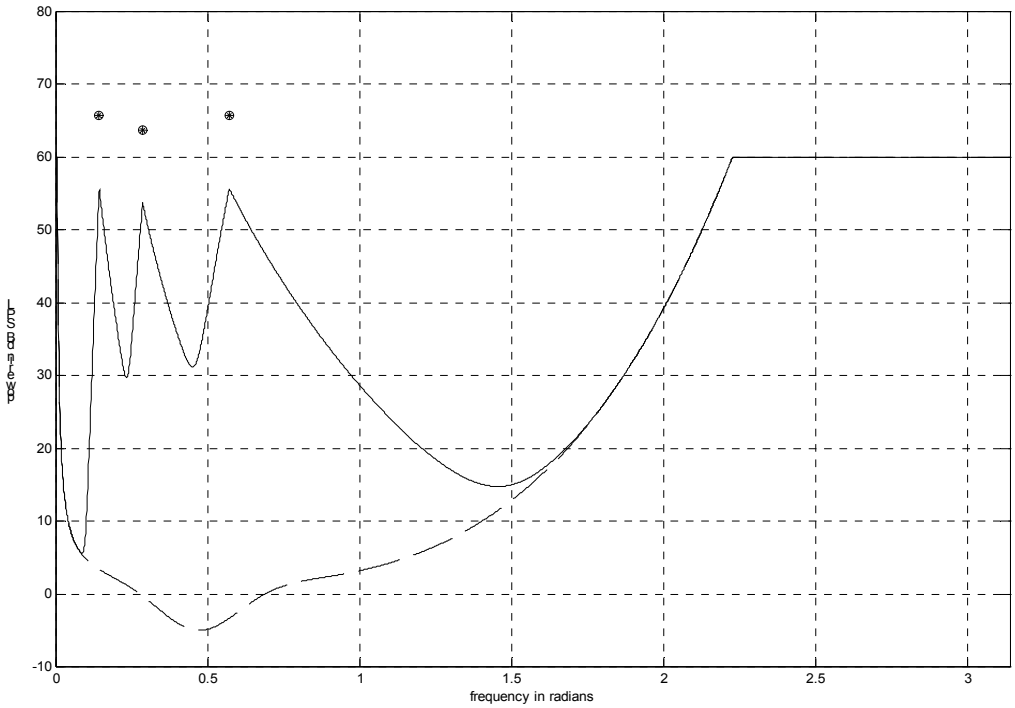

Figure 21 - Example of frequency masking of sinusoids

Apart from the individual masking curves the total masking curve is also partially determined by the hearing threshold in quiet. This is a threshold describing how much power a single component should contain in order to be perceived in an absolute sense. The hearing threshold in quiet is shown as an interrupted line, the total masking curve is shown as a solid line in Figure 21.

4.5.3 Tracking

The main function of the tracking algorithm is to improve coding efficiency. The tracking algorithm consists of three steps:

1. **Apply tracking**: linking of sinusoidal components in time.
2. **Phase less reconstruction**: for tracks that have been found only the initial phase has to be transmitted.
3. **Removal of non-tracks**: tracks that are very short can't be perceived as being a sinusoidal component, they are therefore removed.

The first step of the algorithm tries to link the sinusoidal components that have been found on a frame-to-frame basis [Edler et al 1996, den Brinker and Oomen 1999]. This process is shown graphically in Figure 22. A sinusoidal component will become one of the next four possibilities:

1. **Birth**: part of a track with only a successor.
2. **Continuation**: part of a track with predecessor and successor.
3. **Death**: part of a track with only a predecessor.
4. **Non-track**: a 'track' consisting of a single frame.

Now the problem arises how to link the sinusoidal components that have been extracted. It is assumed that a track is a slowly varying function of both amplitude and frequency (see Equation 2.3). Therefore separate cost-functions for both amplitude and frequency have been developed. For the frequency the cost-function is based on the ERB-scale. The reason to do so is that for complex stimuli the relevant events are more or less separated according to the ERB-scale. The cost-function for frequency then becomes (Equation 2.6):

$$Q_{p,q}^{f} = \begin{cases} 0 & for \quad \left| e(f_{p,k}) - e(f_{q,k-1}) \right| \geq e_{\max} \\ 1 - \dfrac{\left| e(f_{p,k}) - e(f_{q,k-1}) \right|}{e_{\max}} & for \quad \left| e(f_{p,k}) - e(f_{q,k-1}) \right| < e_{\max}, \end{cases} \qquad 2.6$$

where $e(f_{p,k})$ denotes the frequency in erb of the p^{th} component in the k^{th} frame and $e_{\max}$ the maximally allowed deviation expressed in erb.

For the amplitudes a similar cost-function is used (Equation 2.7):

$$Q_{p,q}^{a} = \begin{cases} 0 & for \quad \left| A_{p,k} - A_{q,k-1} \right| \geq A_{\max} \\ 1 - \dfrac{\left| A_{p,k} - A_{q,k-1} \right|}{A_{\max}} & for \quad \left| A_{p,k} - A_{q,k-1} \right| < A_{\max}, \end{cases} \qquad 2.7$$

where $A_{p,k}$ denotes the amplitude expressed in decibels of the p^{th} sinusoidal component in the k^{th} frame and $A_{\max}$ the maximally allowed deviation. The total cost-function now becomes (Equation 2.8):

$$Q_{p,q} = Q_{p,q}^{f} Q_{p,q}^{a}. \qquad 2.8$$

When for a certain sinusoid *p* there exists no $Q_{p,q}$ greater than zero it is marked as being the end of a track. If for a certain sinusoid *p* there exist more than one $Q_{p,q}$ greater than zero sinusoid *q* is chosen with the largest value of $Q_{p,q}$.

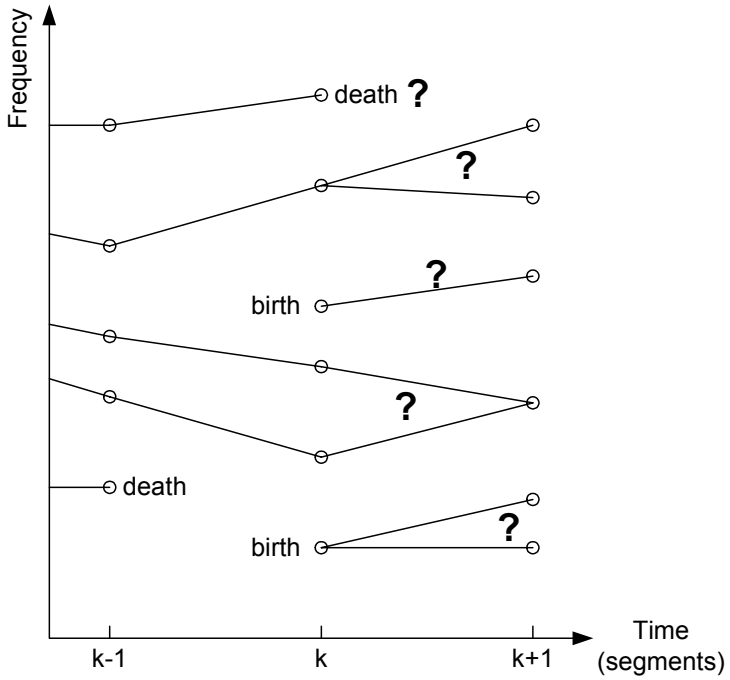

Figure 22 - Tracking of sinusoidal components

The second step of the algorithm consists of phase less reconstruction. Phase less reconstruction is based on the assumption that the instantaneous phase of a track is a smooth function of time. For two consecutive frames of a track the instantaneous phase is equated as shown below. Assume that the sinusoid in frame *k-1* and frame *k* are described as (Equation 2.9 and 2.10):

$$s_{p,k-1}[n] = A_{p,k-1}\cos\left(\omega_{p,k-1}n + \varphi_{p,k-1}\right), \qquad 2.9$$

$$s_{q,k}[n] = A_{q,k}\cos\left(\omega_{q,k}n + \varphi_{q,k}\right). \qquad 2.10$$

The equation of the instantaneous phase can be best performed at the middle of the overlapping segments. This is because the synthesis windows are (symmetric) Hanning windows. The overlap is shown graphically in Figure 23 for a segment length of *N=40*. Note that n is defined symmetrically around zero for both Equation 2.9 as well as 2.10; shifted time-axes are thus used.

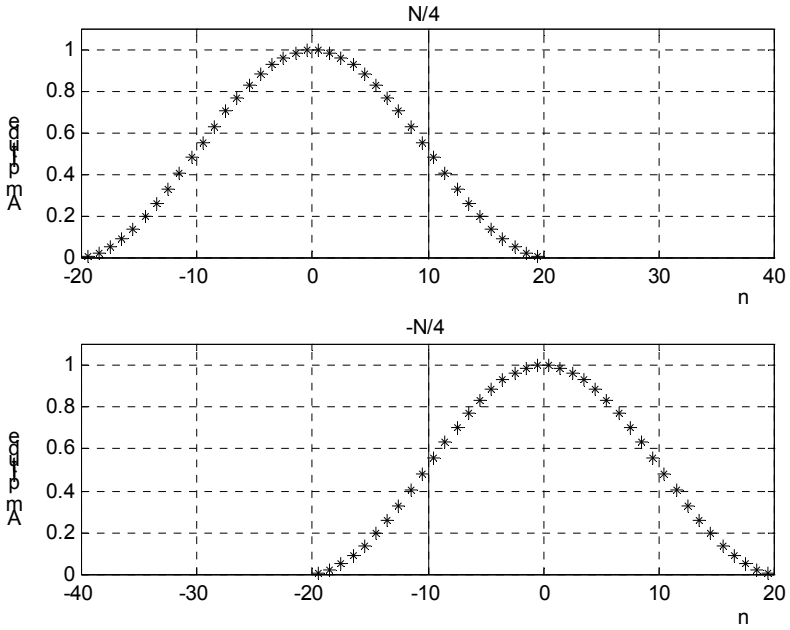

Figure 23 - Equation of instantaneous phase at overlapping frames

Equating the instantaneous phase of Equation 2.9 and 2.10 at the middle of the overlap of a segment with length *N* then gives (Equation 2.11):

$$\omega_{p,k-1}\frac{N}{4}+\varphi_{p,k-1}=\omega_{q,k}\frac{-N}{4}+\varphi_{q,k}, \qquad 2.11$$

which results in (Equation 2.12):

$$\varphi_{q,k}=\left(\omega_{p,k-1}+\omega_{q,k}\right)\frac{N}{4}+\varphi_{p,k-1}. \qquad 2.12$$

Equation 2.12 already indicates that the first step of the tracking algorithm, the linking procedure, has great influence on quality. Erroneous linking of tracks can seriously distort phase relations between tracks.

The final step of the tracking algorithm consists of the removal of short tracks. Tracks that are shorter than five sinusoidal periods cannot be perceived by the human auditory system as being a tonal component. Such tracks are therefore deleted.

4.5.4 Coding of sinusoidal components

The coding of the sinusoidal components consists of:

1. Quantization of parameters.
2. Sort data in births and continuations.
3. Sort births in frequency.
4. Sort continuations in frequency (where deaths are also seen as continuations).
5. Apply absolute and differential coding.

The quantization of the parameters is done according to the same rules as the sinusoids in the transient code (see paragraph 2.4.3).

In order to efficiently code the parameters and the tracking information the matrices containing the sinusoidal components' frequency, amplitude and phase are sorted. This is shown graphically in Figure 24. The left matrix shows how the information is stored before sorting; the right matrix shows how the information is stored after sorting. Note that for both matrices the only information needed to pick the right index of the next set of parameters belonging to a track is whether or not a sinusoidal component is continued.

For the first birth in a frame the amplitude and frequency are coded absolutely. The amplitude and frequency for all other births within a frame are coded differentially to their predecessor in the frequency domain (vertically in Figure 24). For continuations the amplitude and frequency is coded differentially in the time domain (horizontally in Figure 24).

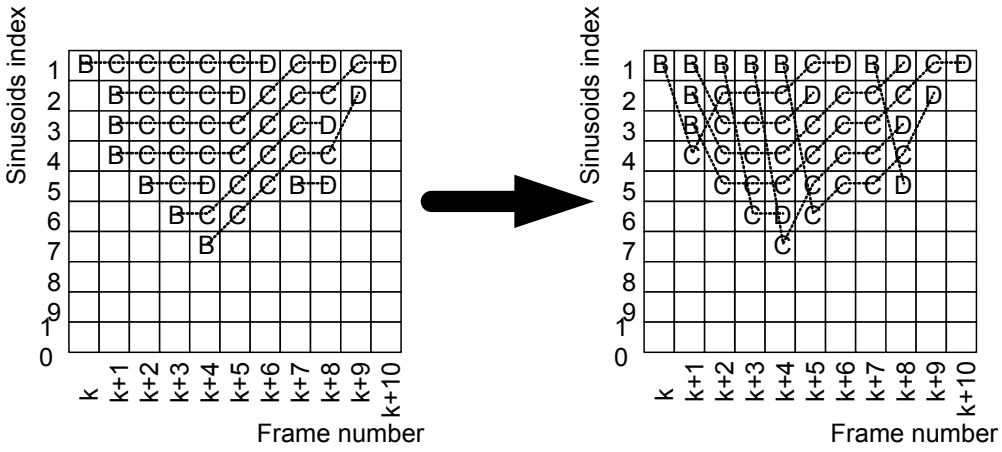

Figure 24 - Sorting of births and continuations

4.5.5 Sinusoidal Synthesis

The synthesis of the sinusoidal components differs from the analysis only in the windowing that is used. In the analysis section overlapping frames were analysed as (Equation 2.13):

$$s_k[n] = \sum_p A_{p,k} \cos\left(\omega_{p,k} n + \varphi_{p,k}\right), \qquad 2.13$$

where p denotes sinusoid index and k the frame index.

In the synthesis section the overlapping frames are synthesised as (Equation 2.14):

$$s_k[n] = h[n] * \sum_p A_{p,k} \cos\left(\omega_{p,k} n + \varphi_{p,k}\right), \qquad 2.14$$

where $h[n]$ denotes the window function and * denotes convolution. The windows that are used are amplitude complementary. Three types of windows are used during synthesis:

1. Normal window, defined as a Hanning window (Equation 2.15):

$$h[n] = \frac{1}{2} + \frac{1}{2}\cos\left(\frac{2\pi}{N}\left(n - \frac{N-1}{2}\right)\right) \quad for \quad 0 \le n \le N-1, \qquad 2.15$$

where N denotes the frame length.

2. Start window, defined as half rectangular and half a Hanning window (Eq. 2.16):

$$h[n] = \begin{cases} 1 & for \quad 0 \le n \le \dfrac{N-1}{2} \\ \dfrac{1}{2} + \dfrac{1}{2}\cos\left(\dfrac{2\pi}{N}\left(n - \dfrac{N-1}{2}\right)\right) & for \quad \dfrac{N-1}{2} < n \le N-1. \end{cases} \qquad 2.16$$

3. Stop window, defined as half a Hanning window and for the rest a rectangular window (Equation 2.17):

$$h[n] = \begin{cases} \dfrac{1}{2} + \dfrac{1}{2}\cos\left(\dfrac{2\pi}{N}\left(n - \dfrac{N-1}{2}\right)\right) & for \quad 0 \le n \le \dfrac{N-1}{2} \\ 1 & for \quad \dfrac{N-1}{2} < n \le T - N - 1, \end{cases} \qquad 2.17$$

where T denotes the length of the frame needed.

Only the frames at the edges of transients use start- and stop-windows (frames 1 and 41 in Figure 18), all others use normal windows (frames 2 until 40 in Figure 18). A typical windowing sequence is depicted in Figure 25.

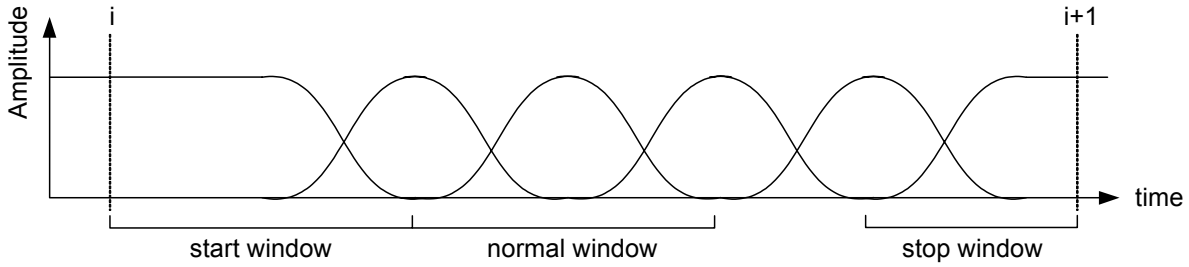

Figure 25 - Typical synthesis windowing sequence

4.6 Noise

Finally the residual signal from the sinusoidal module gets fed to the noise module. An important precondition for this module is the amount of tonality of the PCM input signal. In order for this module to work properly the tonality of the PCM input signal must be kept to a minimum. In effect this means that the transient and synthesis modules must remove as much as possible of the tonal components present in the original signal. The noise module is depicted in Figure 26.

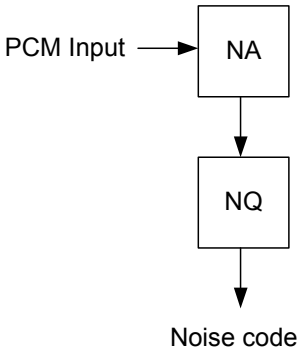

Figure 26 - Block-diagram of noise module

In the noise analysis module the noise gets analysed into a set of parameters per frame. The segmentation is equal to that of the sinusoidal module. In the noise quantization block the noise parameters get quantized and coded for use in the bitstream. Unlike the transients and the sinusoidal module the noise module does not contain a noise synthesis block because during encoding there is no need to do so. Some important notes on the synthesis block are however made.

4.6.1 Noise Analysis

The transients module and the sinusoidal module both tried to match the original waveform as closely as possible. For the noise module such an approach however will not lead to an efficient representation. From a perceptual point of view this is also not an efficient representation. From that viewpoint the amount of signal (noise) power per critical band is the only relevant information. Therefore the noise module tries to spectrally model the input signal per segment. It does so by means of an auto-regressive moving-average (ARMA) model [den Brinker and Oomen 2000]. This ARMA model tries to describe the power spectral density function (psdf) as well as possible by means of poles (auto-regressive model) and zeros (moving-average model). The transfer function of an ARMA model can be described by (Equation 2.18):

$$H = \frac{H_n}{H_d} = G \frac{\prod_{k=1}^{K}\left(1 - z^{-1} p_k\right)}{\prod_{l=1}^{L}\left(1 - z^{-1} q_l\right)}, \qquad 2.18$$

where H_d is the denominator polynomial, H_n the numerator polynomial, G the gain factor, K the number of zeros and L the number of poles. Within SSC, $K = 1$ and $L = 6$.

The block-diagram of the noise analysis module is given in Figure 27.

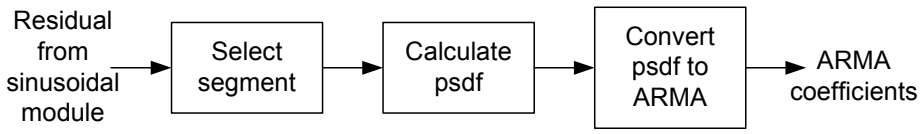

Figure 27 - Block-diagram of noise-analysis module

First of all from the input signal the right segment is selected. From this segment the psdf is calculated by means of a smoothed FFT. Finally from the psdf the ARMA coefficients are estimated. This is depicted in Figure 28.

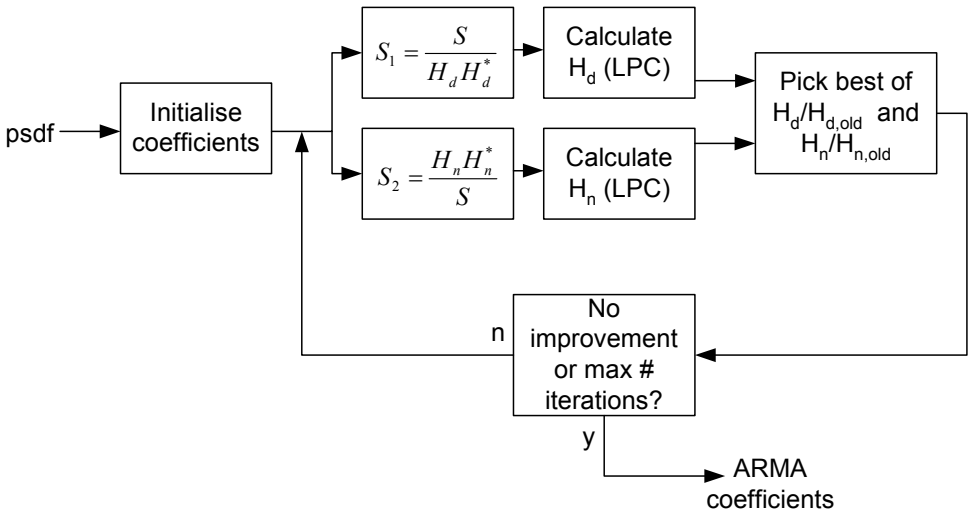

Figure 28 - ARMA estimation from psdf

The process of estimating the psdf by means of an ARMA model is iterative. Mostly ARMA models are not used because of their computational complexity. The method as described in Figure 28 however tries to split the psdf into a part that can be modelled adequately by the zeros, denoted S_1, and a part that can be modelled adequately by the poles, denoted S_2. When this split has been made new estimates of the denominator and numerator polynomials are made with the use of linear prediction. The best combination of the polynomials found in the previous iteration and the current iteration is picked based on a squared error criterion on a logarithmic scale. When no more improvement can be obtained or if the maximum number of iterations is exceeded, the ARMA coefficients are passed on for further processing.

As an example Figure 29 shows a psdf, shown as an interrupted line, estimated by the ARMA model as described above, shown as a solid line.

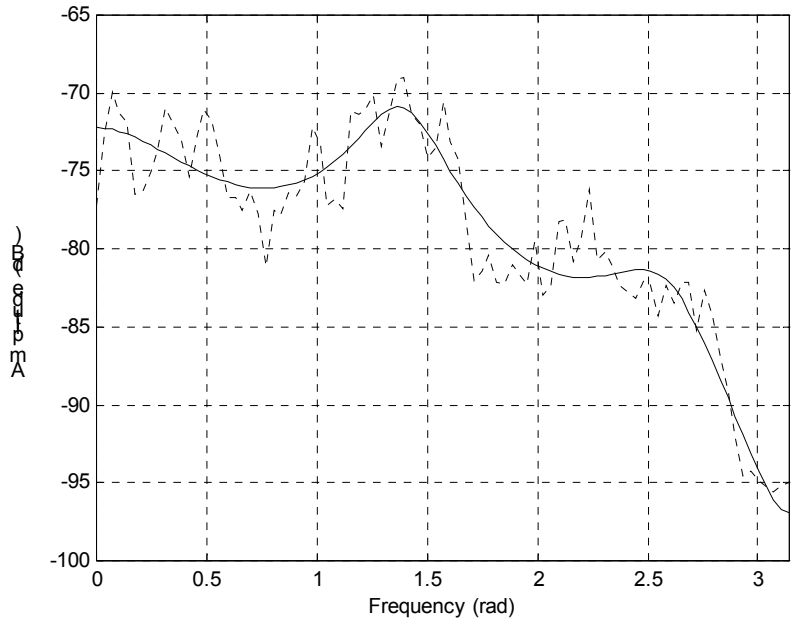

Figure 29 - Example of ARMA estimate of psdf

4.6.2 Noise synthesis

The synthesis of the noise modelled by the ARMA model is fairly simple. The block-diagram is shown in Figure 30.

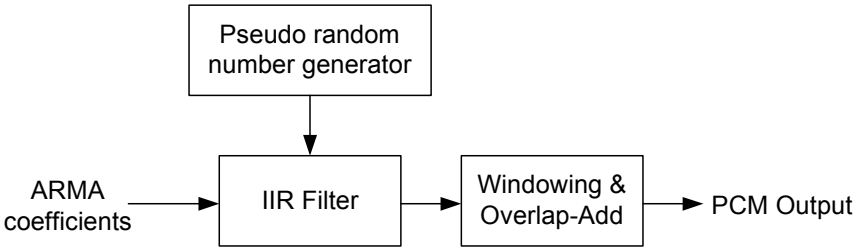

Figure 30 - Block-diagram of noise synthesis

First of all white noise is generated using a pseudo random generator. This noise is fed through the IIR filter. For every frame a segment of white noise is filtered by the ARMA coefficients belonging to that frame. For every such segment windowing is applied. Instead of an amplitude-complementary window in the sinusoidal module a power-complementary window is used. Such a window is defined as (Equation 2.19):

$$h[n] = \sin\left(\pi \cdot \frac{n+0.5}{N}\right) \quad for \quad 0 \le n \le N-1, \qquad 2.19$$

where N is the segment length.

Finally an overlap-add method is used to create the PCM output signal.

4.6.3 Noise quantization

In order to efficiently code and quantise the ARMA coefficients these coefficients are converted to the Log Area Ratio (LAR). In order to do so the coefficients of both denominator and numerator polynomial are first converted to reflection coefficients as used in lattice filters. Thereafter the reflection coefficients are converted to LAR coefficients by (Equation 2.20):

$$g = \frac{1}{2}\log\left(\frac{1+k}{1-k}\right), \qquad 2.20$$

where k is the reflection coefficient and g the LAR coefficient. The advantage of the LAR representation is the fact that linear quantization can be applied. This is currently applied with an accuracy of one decibel. The gain factor G (see Equation 2.18) is also quantized with an accuracy of one decibel.

So finally, per frame eight parameters are passed on to the bit-stream formatter: a gain factor, one LAR coefficient for the numerator polynomial and six LAR coefficients for the denominator polynomial.

4.7 Quantization and coding

Until now the focus was mainly on the extraction and processing of the parameters towards an efficient representation. It was also shown how irrelevancy is exploited within the SSC coder. To further improve coding efficiency not only irrelevancy should be exploited but also the redundancy.

To efficiently encode all the extracted and processed parameters into a bit-stream three steps are required:

1. Pre-processing (sorting),
2. Quantization,
3. Entropy coding.

First of all the parameters need to be pre-processed. This is mainly a sorting process in which the different parameters are grouped together to form sets that can each be represented efficiently. For example the sorting of births and continuations as described in paragraph 2.5.4 belongs to the pre-processing stage, but also the sorting of sinusoidal components by frequency in the transient module.

In the second stage, quantization is applied to the sorted parameters. The parameters must be quantized in such a manner that after decoding *just* no differences can be perceived before and after quantization. For e.g. the amplitudes of the sinusoids logarithmic quantization can be applied.

Finally entropy coding is applied to the representation levels, which represent the quantized levels. The most common form of entropy coding is Huffman coding. This is a constant to variable wordlength entropy coding technique. The main advantage of Huffman coding over other entropy coders is the low complexity: both encoding and decoding can be performed by table look-up. Furthermore these tables are easily constructed.

A general diagram for coding audio parameters using Huffman coding is depicted in Figure 31.

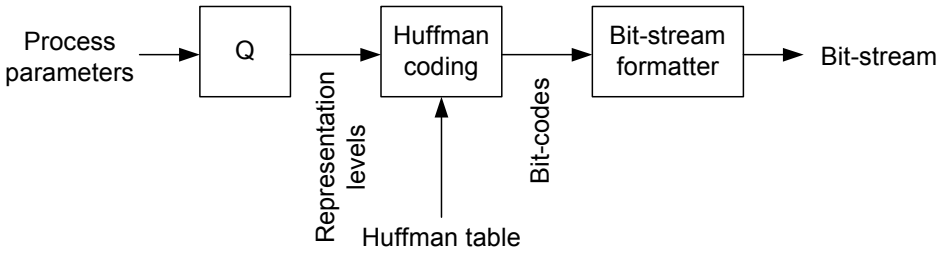

Figure 31 - General block-diagram for efficiently encoding of parameters

As an example the following process *x* is given (Table 1):

Parameter value *x*:	**Representation level:**	**Quantized representation $\hat{x}$:**
$0.00 \le x < 0.52$	0 (00)	0.32
$0.52 \le x < 0.82$	1 (01)	0.74
$0.82 \le x < 0.95$	2 (10)	0.89
$0.95 \le x < 1.00$	3 (11)	0.97

Table 1 - Example of parameters quantization

Now state that these representation levels have the following distribution:

Representation level:	**Probability of occurrence:**	**Huffman code:**
0 (00)	0.5	'0'
1 (01)	0.3	'01'
2 (10)	0.1	'001'

Representation level:	Probability of occurrence:	Huffman code:
3 (11)	0.1	'000'

Table 2 - Distribution of the representation levels

The Huffman table is built up in such a way that less probable representations get longer codewords. For this particular code the mean word-length (representing a single representation) is 0.5·1+0.3·2+0.1·3+0.1·3=1.7 bits per representation, which is a coding gain of 2/1.7=1.17.

One disadvantage of Huffman coding is that the look-up tables might become rather large. In case a large number of representations only occur with a small probability escape codes can be used advantageously. Table 3 shows an example of the usage of an escape code. Using escape-codes in Table 3 only four codes have to be stored, while in the case of no escape-code all sixteen codes have to be stored. The code with escape will on average generate 1.9 generate per symbol (representation) while the 'pure' code will generate 1.89 bits per symbol (representation).

Code	Probability:	Code with escape	Code without escape
0000	0.5	0	0
0001	0.3	10	10
0010	0.15	110	110
0011	0.0038	111 0011	111101
0100	0.0038	111 0100	111110
0101	0.0038	111 0101	111111
0110	0.0038	111 0110	1110000
0111	0.0038	111 0111	1110001
1000	0.0038	111 1000	1110010
1001	0.0038	111 1001	1110011
1010	0.0038	111 1010	1110100
1011	0.0038	111 1011	1110101
1100	0.0038	111 1100	1110110
1101	0.0038	111 1101	1110111
1110	0.0038	111 1110	1111000
1111	0.0038	111 1111	1111001

Table 3 - Huffman table with and without escape code

5. Quality assessment experiments

5.1 Introduction

The first objective of the graduation project was to identify the problems that were within the SSC encoder. In order to do so numerous experiments have been performed. In this chapter the most important experiments are described and conclusions of these experiments are given.

5.2 General quality assessment

The first step in analysing the quality limitations of the SSC coder focused on listening and evaluating the decoded versions of extracted parameters and to residual signals at every step of the encoder. This is shown graphically in Figure 32. For this purpose a set of thirteen excerpts were used that are generally known to be critical for audio coding. Throughout the whole project this set has been used for evaluation purposes.

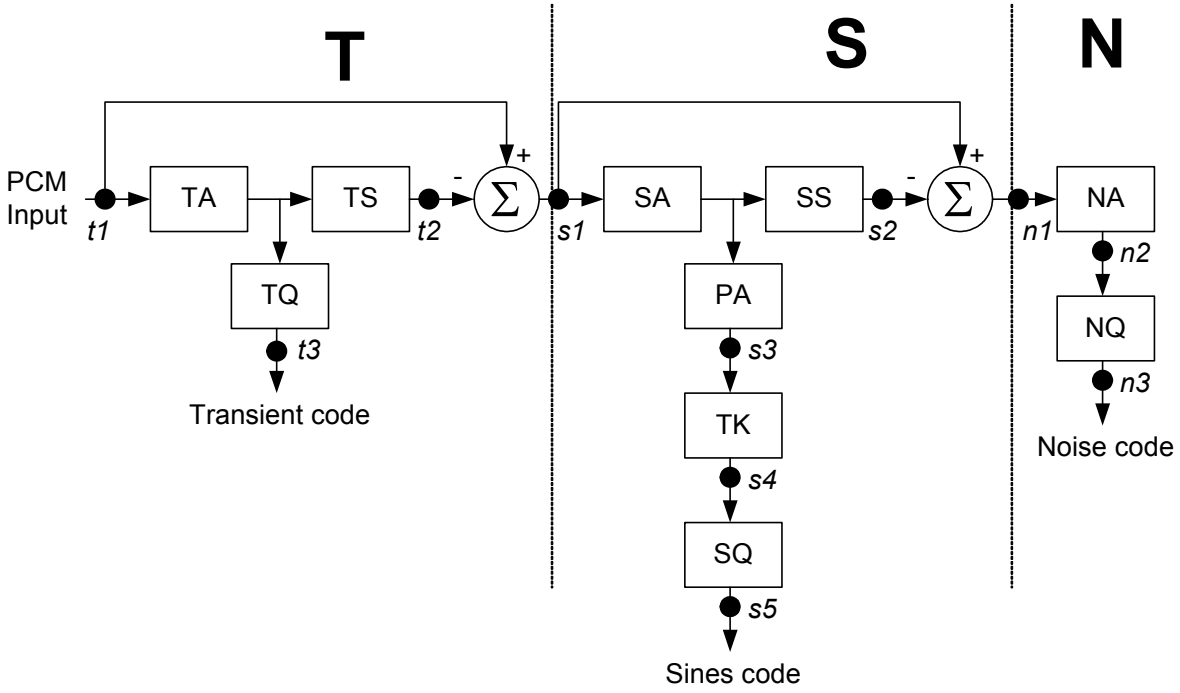

Figure 32 - Evaluation points for quality assessment

First of all when comparing the original signal (*t1*) with the quantized transients signal (*t3*) plus the transients residual signal (*s1*) no performance loss seems to occur. However, when listening to the extracted transients only (*t2*), it seems that only a small part of what's perceived as a transient is actually removed. This is confirmed by listening to the residual signal of the transients module (*s1*).

Especially when listening to signals, which have clear transients like e.g. castanets, still some residue of the transients seem to be present.

What's first of all noticeable in the sinusoids extraction module is the fact that the residual signal of this module (*n1*) still contains a lot of tonal elements. Ideally this signal should be perceived as being noisy. These tonal elements could be attributed to shortcomings in both the transients as well as the sinusoidal module. When listening to the sinusoids only (*s2*) in comparison to the residual signal from the transients module (*s1*) it is noticed that the sinusoids (*s2*) have less sharpness or transient-like behaviour.

When comparing the sinusoids that have been extracted before (*s2*) and after the psycho-acoustic model (*s3*) no noticeable differences can be perceived. When however comparing the sinusoids after the psycho-acoustic model (*s3*) with the sinusoids after tracking (*s4*) a big quality loss is noticed. Further listening within this block showed two reasons:

1. The use of phase less reconstruction seems to cause a certain echo-like effect. This effect is also described as 'metallic' sound.
2. The removal of non-tracks removes a lot of sharpness of the signal, especially on positions where transients have been detected.

Quantization of the parameters (*s5*) even further decreased the quality of the sinusoids. This mainly increases the 'metallic' sound of signals.

When comparing the noise signal generated from the extracted parameters (*n2*) with the residual signal of the sinusoidal module (*n1*) a big quality loss is observed. This is however not caused with certainty by the performance of the noise module. It could be caused by the condition of the residual signal. This signal still contains a lot of tonal elements that show up as peaks in the frequency domain. Because of the limited amount of ARMA coefficients the noise coder would profit from a smooth frequency spectrum.

Finally the noise signal synthesised from the quantized parameters (*n3*) was compared with the noise signal before quantization (*n2*). This comparison showed that quantization of the parameters caused no noticeable loss of quality.

All in all the following conclusions can be made:

- The transients module performs reasonably well. Quantization of parameters leads to no perceptual loss. It is noted however that the residual signal from this module still contains transient-like information.
- The sinusoidal module is not able to fully remove all tonal components from its input signal. Furthermore phase less reconstruction, the removal of short tracks and quantization further decrease the quality.
- It is unsure how the noise module performs because of the condition of its input signal. Quantization however does not seem to lead to an additional decrease of quality.

6. On the SSC encoder scheme

I have dedicated the first week of my internship in understanding how the SSC encoder works, and in figuring out what are the main features of a SSC encoded signal and its problems. I eventually tried some simple algorithm in order to improve the encoder. In this chapter, these first impressions are reported.

6.1 Negative points of the SSC

SSC is a good coder and I was impressed when I first listened to the excerpts, thinking that they had been encoded at 24 kbit/s.

The biggest problems that I could find during this first phase of listening were:

1. For certain excerpts it sounds very natural, but for certain other it sounds more artificial (even metallic). Maybe this effect is due to quantization of the amplitudes or to some hole in the tracking trajectories. It is not excellent in the high frequencies, and the sound results "closed".
2. The transient detection is very poor; most of the transients are skipped or refused. Moreover the transient reconstruction is not perfect.
3. The sinusoids detection is quite good, but a lot of components in the high frequency domain are detected (and tracked!!) when they do not exist at all. I have used some synthetic signals for that. This is because the window is sometime too short (the longer the window, better resolution in the frequency domain). The problem is that the S analyzer is blind, I mean, it does not have any idea of how many sinusoids it should detect and this is not depending on the criterion used for the selection of the sinusoids.
4. The tracking algorithm should be improved, or maybe just smoothed a little bit more.
5. The noise signal has a lot of transient components and even sinusoids components, but in general it is not too bad, I mean, at least it sounds like a noise. A type of noise that is usually present in music is the reverberation, artificial or natural. Maybe we should consider also its features.
6. Fixed size of subframes (versus an adaptative sizing)

The idea of parametric coding is based on the fact that the sound can be separated in 3 components: T, S, and N. After each of the analysis phases, a synthesis is made. Therefore, it is clear that the encoded signal cannot be better than the original.

T is estimated first, as it is important to have an as stationary as possible signal in the following analysis steps.

Therefore, the transient detection must be as perfect as possible, and it is, nowadays, very poor. From the other side, the improvement of the transient module will improve the quality of the signal only locally as in general the number of relevant transients is less than 10 events per second.

6.2 Passing from 24 to 32 kbit/s

The algorithm used in order to increase the bitrate is the following:

First of all, the signal is encoded at 24 kbits, then some parameters are changed, and the signal re-encoded at each time, until the target bitrate (32 kbits) is reached.

The parameters that are changed are: the frequency and amplitude quantization grids and the relevance threshold of the psychoacoustic model.

Changing the first parameter, gives the sinusoids a smoother behaviour, lowering the relevance threshold implies that more sinusoids will be detected and tracked.

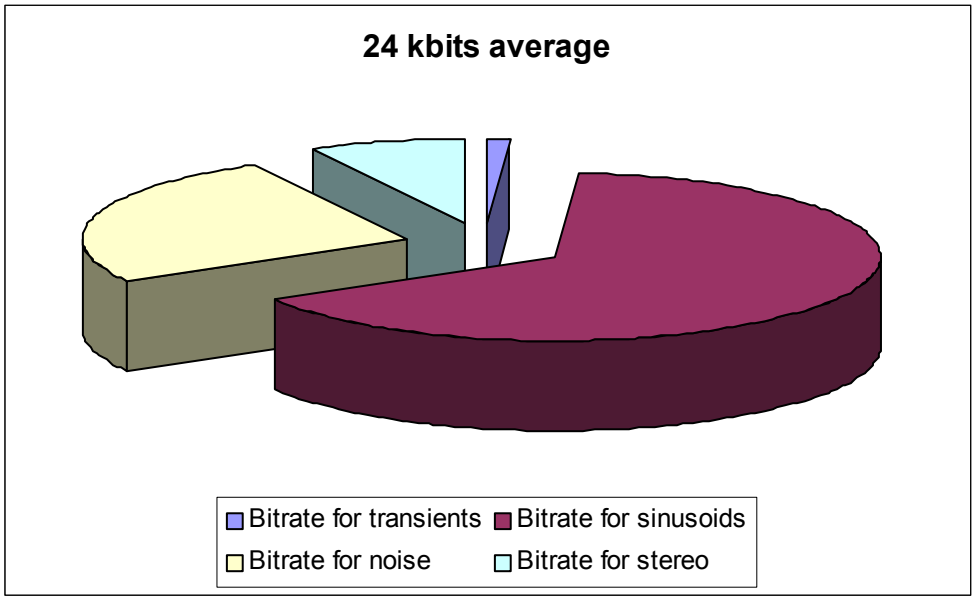

Figure 33 - 24kbits bitrate distribution

In the Figure 33 and Figure 34, it is easy to understand why the quality of the excerpts does not improve passing from 24 to 32kbits, at least in the stereo case. In this case indeed the percentage of the bitrate dedicated to the stereo parameters increases, while the percentage dedicated to noise decreases and the percentage of noise and sinusoids remains constant. In my opinion we are more

sensitive to mono encoding quality than to the stereo encoding one, so that less bit should be given to stereo.

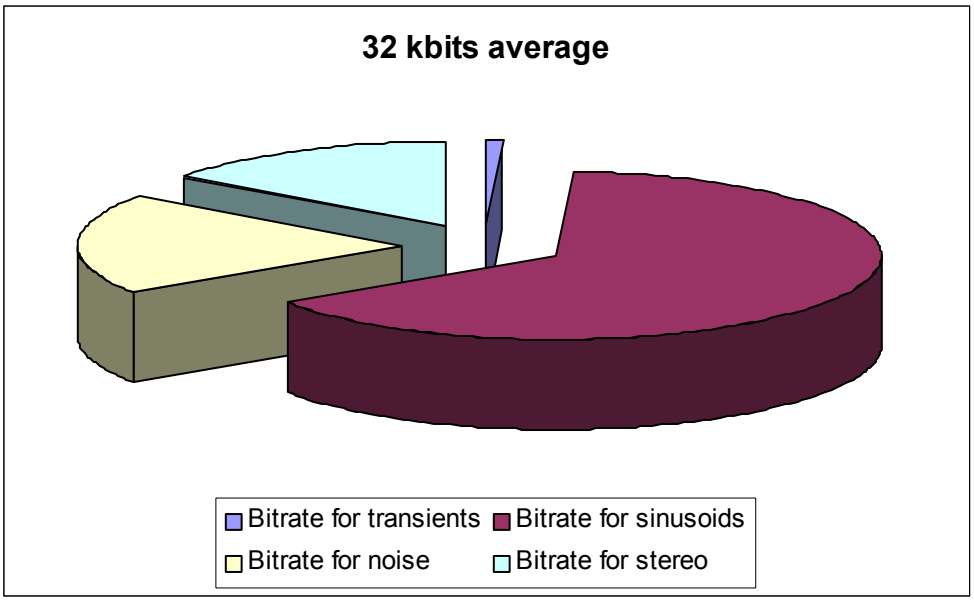

Figure 34 - 32kbits bitrate distribution

6.3 A new transient detection algorithm

The old algorithm basically works like this: on a frame basis, the signal is filtered with this high pass filter: [1 –1]. This filter is an approximation of a gradient filter at low frequencies. As a result we have an approximation of the gradient of the signal. Then, the superior envelope of the gradient is computed and each sample of the envelope is compared with its previous. If this ratio is bigger than a certain threshold (min_ratio_near) and also this ratio is bigger than another ratio (min_ratio_far) some samples later, than a transient is detected in the frame. A fine detection is computed in the next stages.

Within this algorithm it is assumed that at most one transient can be detected in a subframe and that two consecutive subframes cannot both contain a transient, which means a minimal distance between transient of 360 samples. Moreover, a Meixner envelope cannot last more than 4 subframes, id est. almost 1200 samples. One bit is dedicate in each frame to tell whether the transient is present or not. This is equivalent to 122.5 bits (= 44100 / 360) while the average amount of bits is 230 for transients, both at 24 kbit/s to 32 kbit/s.

My personal opinion is that this algorithm has several problems in detecting step transients and it is too static (not adaptative) in detecting the transients. To solve this problem I propose a new algorithm:

The signal is divided in frame of about one second each. The signal is normalized to 1 and divided in subframes of for instance 360 samples each. For each subframe the power is computed, then a low pass filter smoothes the vector of powers and this vector is normalized to the maximum power of a subframe, id est the subframe length. The criterion will then be the following: if in a certain subframe the power is bigger than the average of the last 5 subframes multiplied by a coefficient, then this subframe is a candidate for a fine detection of the transient.

The coefficient is assigned according to the following table that I have estimated with some experiments with the excerpts. 0 means that the transient is refused, one that is accepted.

Average Power Level vs. Actual Power Level

	0 – 10	**10 – 20**	**20 – 30**	**30 – 40**	**40 – 50**	**50 – 60**	**60 – 70**	**70 – 80**	**80 – 90**	**90 – 100**
0 – 10	100	0	0	0	0	0	0	0	0	0
10 – 20	20	0	0	0	0	0	0	0	0	0
20 – 30	5	2.5	0	0	0	0	0	0	0	0
30 – 40	1.2	1.7	1.9	0	0	0	0	0	0	0
40 – 50	1	1.2	2.1	0	0	0	0	0	0	0
50 – 60	1	1	1.8	1.5	0	0	0	0	0	0
60 – 70	1	1	1.2	2	1.6	0	0	0	0	0
70 – 80	1	1	1	1.2	1.4	0	0	0	0	0
80 – 90	1	1	1	1	1.2	1.4	0	0	0	0
90 – 100	1	1	1	1	1	1.2	1.4	0	0	0

Table 4 - Table of coefficients for the detection threshold

More transients are detected, and also step transients.

6.4 Development of the TSN model

The idea that I have in mind is that we should change the concept of transient, and more precisely work on the idea of segmentation of the wave file.

If the signal is segmented in such a way that in each segment at least one feature of the signal is constant, than S and N estimation can be more precise and the distribution of bits among S, T and N optimized. Even the subframe could be taken larger. We could propose a classification of the segments.

I have thought of these types of segments:

0. silence
1. noise segment
2. Meixner transient
3. step transient up
4. step transient down
5. triangular transient (like fade in, fade out), maybe even second and third order type
6. sudden changes in spectral features (this is what happens when a new note is played for example or a new instrumentalist joins in)

Furthermore, each type of segment could be encoded differently, I mean, for the silence, just the length is enough, for the noise, no need to dedicate bits to T or S, etc...

Negative points of this approach are that of course a real time implementation is not possible, it is more complicated algorithmically and I do not have any idea of how much the quality will eventually improve on behalf of the bit rate, but I think is worthy to try it. I have not considered stereo parameters for the moment.

It seems from the experiments, see next section, that the quality is not going to improve, but the bitrate gain can vary from 0 to 20%.

6.5 Experiments in order to detect the problems of SSC

I have made different tests. Each test has a precise target and different settings. The context and the results are summarized for each experiment.

6.5.1 Experiment number one: Harmonic sound in noise

Filename: La_noise_s44.wav

Description: 8 seconds signal, constant white noise power with harmonic sinusoids at low frequencies (110, 220, 440, 880 Hz), step transients;

Conclusion: A lot of sinusoids are tracked in the noise. If I reduce the maximum number of sinusoids to be detected from 60 to 10, then a lot of tracks are reduced, and the remaining are no more audible. The quality therefore increases, but bits are used to code something not audible. Step transients are not detected, so that in the noise signal in correspondence of the steps there is a small burst of energy, which was absent in the original signal.

6.5.2 Experiment number two: Sinusoid with frequency linearly increasing

Filename: cos_up_s44.wav

Description: Only one sinusoid is present. Its frequency goes linearly and slowly from 1000 Hz to 2000 Hz.

Conclusion: No transient detected, no noise synthesized. Perfect reconstruction.

6.5.3 Experiment number three: Two sinusoids with frequency crossing

Filename: cross_s44.wav

Description: Two sinusoids of the same amplitude, one from 1000 to 5000 Hz and the other from 2000 to 0 Hz, crosses at a certain point of the time. No noise added. 10 sinusoids used in the analysis.

Conclusion: No transient detected, no noise synthesized. Perfect reconstruction. The tracking algorithm showed a problem in the crossing position. A +3db amplitude is detected and the tracks lose their original slope. The frequency are not correctly estimated, anyway the crossing masks completely these errors.

6.5.4 Experiment number four: Two sinusoids crossing with noise

Filename: cross_noise_s44.wav

Description: It is the cross_s44.wav with a white noise added. 60 sinusoids are used in the analysis.

Conclusion: Sinusoids are well detected, but worse than in the previous case in the low frequency domain. Lots of short sinusoids are tracked, but far from the two main sinusoids. A characteristic of these small tracks is that they are not correlated among them and are very short (from 3 to 6 segments). Longer tracks are present in low frequencies (up to 30 segments) and they are also more regular in their movement. A high level algorithm of decision is missing. What about hidden markov chains?

6.5.5 Experiment number five: Sinusoids with sinusoid frequency changing

Filename: sinsin_1100_s44.wav

Description: A sinusoid is changing its frequency as a sinusoid with increasing amplitude from 1000 to 8000 Hz. No noise added.

Conclusion: The tracking algorithm is weak and also the frequency detection is poor. This fact causes a noise estimation.

6.5.6 Experiment number six: Sinusoids with slow sinusoid frequency changing

Filename: coscos_slow_s44.wav

Description: A sinusoid is changing its frequency as a sinusoid with increasing amplitude from 500 to 1100 Hz. No noise added.

Conclusion: The tracking algorithm is weak. This fact causes a noise estimation. Also some long tracks are present at the frequency 0.

6.5.7 Experiment number seven: White noise

Filename: white_noise_s44.wav

Description: Stereo white noise, with fade in and fade out

Conclusion: There should be no sinusoids, but a lot are detected and tracked. There is a very interesting phenomenon: below 500 Hz, tracks are long (max 60 segments) and horizontal; above 500 Hz, tracks are very short and randomly distributed. The synthetic sound appears voiced. Furthermore, the noise component sounds “liquid”.

6.5.8 Experiment number eight: Rain noise plus piano

Filename: Rain_piano_s44.wav

Description: Rain is coming, after some seconds a piano is starting to play.

Conclusion: Transients should be detected all over the excerpts. Almost no sinusoids should be estimated in the first part. Transients are not detected and sinusoids tracked all over the excerpt. Noise sounds too filtered.

6.5.9 Experiment number nine: Quantization versus non quantization

Filename: sc03_s44.wav

Description: Noise part is not analyzed and not synthesized. 4 output files are produced: with (with not) quantization of the parameters and with original or continuous phase.

Conclusion: The quantization is completely irrelevant for signals re-synthesized with original phase. With continuous phase, even if there is a difference on the waveform, this is almost not audible. The quality of the two signals is equivalent.

6.5.10 Experiment number ten: reduction of the number of the sinusoids

Filename: Rain_piano_low.wav; Rain_piano_verylow.wav

Description: The max number of sinusoids to use in the coarse and fine analysis is reduced from 60 (10) to 16 (8) and 10 (5).

Conclusion: Each time the residual signal that reaches the noise module had more power, so that the reconstruction of the noise was in general better. The quality in the piano part was, as expected, decreasing. No transient have been detected.

6.5.11 Experiment number eleven: only noise analysis + only sinusoids analysis

Filename: Rain_piano_test.wav

Description: The sinusoids are estimated as usual, quantization, continuous phase. The noise is estimated without subtracting the synthesized sinusoids. Finally the two signals are added.

Conclusion: Improved reconstruction of the noise. Bad quality in the piano part. The analysis of the sinusoids before the noise is therefore necessary, but it should be controlled.

6.5.12 Experiment number twelve: on the noise filtering

Filename: white_noise_s44.wav

Description: A white noise is analyzed and synthesized only by the noise module.

Conclusion: A white noise signal is still obtained, but it sounds slightly different from the original. For instance, less power is given to the low frequencies. This is not due to the high pass filtering of the beginning of the encoding process. The analyze-synthesis process is therefore audible. The white noise is closer to the original for lower orders of the Laguerre filter. Changing the pole of the Laguerre filter does not change the result qualitatively.

6.5.13 Experiment number thirteen: scalability of the noise filtering

Filename: wave_s44.wav

Description: It is the sound of a sea wave crushing on the shore. The excerpt is synthesized using different orders for the Laguerre filter: 2, 4, 8, 16, 20, 24, and 40.

Conclusion: Quality increases very fast at the beginning, but then it is almost stable over 20. Also the computation time increases with the order, passing from 5 seconds to 30 seconds. Still the resulting noise reconstruction sounds poor in the low frequencies domain. Anyway, the noise is scalable.

6.5.14 Experiment number fourteen: scalability of sinusoids

Filename: expo_75

Description: A sinusoid at 100 Hz with 74 harmonics. The weight of each harmonic is constant in the first experiment, exponentially decaying in the second.

Conclusion: As 60 sinusoids are tracked as a maximum, 60 sinusoids are tracked in the first experiment, and 43 in the second. In the second case these were the first 43 harmonics, in the first there were some gaps and for instance the highest harmonic was tracked. In both cases some harmonic were tracked with slightly variation of the frequency, which was absent in the original signal.

6.5.15 Experiment number fifteen: scalability of sinusoids (2)

Filename: Orchestra_s44.wav

Description: An orchestra is playing. This excerpt can be considered as if there were only sinusoidal components. Sinusoids are estimated with different maximal number of sinusoids in the coarse and fine analysis: 10, 12; 10, 20; 10, 40; 10, 50; 10, 60.

Conclusion: Quality increases at the beginning. Nevertheless, some spurious sinusoids are tracked. There is a saturation point due to the fact that SSC does not ever consider useful to track more than 30 sinusoids in this excerpt. In this sense the sinusoidal module appear to be not scalable.

6.5.16 Experiment number sixteen: speech signals

Filename: es02_s44.wav

Description: A male German voice is speaking. In the waveform, voiced parts and unvoiced parts are easily separable. Manually, voiced parts are copied from an analysis with lots of sinusoids, and unvoiced without sinusoids analysis.

Conclusion: The quality improves a little, but still the signal sounds metallic in the voiced parts.

6.5.17 Experiment number seventeen: separation of high and low frequencies

Filename: Kelly_s44.wav

Description: A woman is singing with a keyboard. The signal presents very peculiar characteristics: the keyboard and the voice are harmonic and concentrate their energy below 6000 Hz. Above 6000 Hz the spectrum looks more like an irregular noise. These two parts (low and high) are separated and processed separately. Low frequencies with the whole system, high frequencies only with the noise module.

Conclusion: The quality is of the same level, but the bit consumption should be lower.

6.5.18 Experiment number eighteen: scalability of noise temporal envelope

Filename: Rain and wave files

Description: These natural sounds are encoded using different temporal envelope order: 5, 10, 15, 20, 25, and 50.

Conclusion: There is a very small quality improvement passing from 5 to 50. Over 50 the encoding process is not correct.

6.5.19 Experiment number nineteen: scalability of the subframe size

Filename: Kelly_If_I_could_s44.wav

Description: Changing the size of the subframe of analysis should change a lot the quality

Conclusion: The quality decreases very fast passing from 360 to 600, 1200, 2400 samples per subframe. Using smaller size like 300 and 240 does not seem to increase the quality.

6.5.20 Experiment number twenty: encoding of synthetic speech

Filename: Synthetic_speech.wav

Description: Synthetic speech excerpts are encoded.

Conclusion: Synthetic speech is encoded better than natural speech, the formants are tracked with bigger precision, although a lot of errors are present, especially when partials crosses and at the beginning of phonemes.

6.5.21 Experiment number twenty-one: Bell sounds

Filename: Bell_s44.wav

Description: Synthetic bells with big frequency variations

Conclusion: Most of the harmonics are correctly tracked. The resulting encoded sound is very close to the original, it sounds just a little bit less loud in the high frequencies. With the second

bell, partials are not correctly tracked and as a result a noise is synthesized while there were no noise in the original.

6.5.22 Experiment number twenty-two: Phone tones in noise

Filename: Phone_s44.wav

Description: Some phone tones in white noise, then a chord at 220 Hz goes toward 440 Hz. The experiment is repeated with 60 and 20 as max_nr_of_sins.

Conclusion: The detection of the sinusoids is correct and the noise sounds better with a small number of sinusoids.

6.5.23 Experiment number twenty-three: Echo effect

Filename: es01_echo_s44.wav

Description: An echo is added to the voice

Conclusion: The echo is correctly tracked, but the quality of the main voice is slightly worse. Maybe the echoes should be eliminated first (and rebuilt after), to have a more stationary signal.

6.5.24 Experiment number twenty-four: Subframe size scalability

Filename: all excerpts

Description: All excerpts have been coded with 3 different subframe sizes: 240, 360 and 420. The noise is analyzed with an order 40 and with a temporal envelope of order 20.

Conclusion: There is a slight improvement in quality, but not everywhere in the excerpts. It is more audible in speech signals and near transients. This fact suggests that variable size windows should be used.

6.5.25 Experiment number twenty-five: Changing the update rate

Filename: es02, castanets

Description: The update rate will vary from 1 to 16.

Conclusion: The update rate does not seem to be used or to influence the quality. The same phenomenon is valid for the refresh rate.

6.5.26 Experiment number twenty-six: Removing the temporal envelope

Filename: Britney_Spears.wav

Description: The temporal envelope is estimated as a function that contains the signal, then the signal is divided by the temporal envelope such that a more constant signal is obtained.

This signal is encoded with the SSC and eventually the encoded signal is multiplied by the original envelope.

Conclusion: The quality obtained is generally worse than in the normal process. This is due to the fact that multiplying a waveform by an envelope means calculate the convolution of all the peaks with the envelope transform, which typically is a large impulse in the low frequencies.

6.5.27 Experiment number twenty-seven: Separation of noise like region from sinusoidal like noise region

Filename: speech

Description: Manually (with the help of Matlab) the spectrogram of the signal is divided into rectangular regions of two types: first types are considered noise like, and therefore are coded with only the noise analysis; second types are considered sinusoidal like and therefore are coded with the normal process. Considering, that we should gain bits in the regions that are coded with only noise parameters, the sinusoids are synthesized using the original phase.

Conclusion: The quality improves clearly, but an estimation of the increased bitrate is not available, as it is not possible to command this way the SSC bit stream formatter.

6.5.28 Experiment number twenty-eight: Larger band sinusoids

Filename: z.wav

Description: It is a sinusoid at 2000 Hz, who has been convoluted with a gaussian impulse

Conclusion: More than one sinusoid is estimated, and the tracking is ambiguous. Anyway, the reconstruction is not that incorrect. The fact is that some noise is estimated, where there was no noise.

6.5.29 Experiment number twenty-nine: Tracking algorithm and subframe size

Filename: crazy_s44.wav

Description: It is a synthetic signal, made of sinusoids that vary sinusoidically and linearly their frequencies. The subframe size varies from 180 to 360 samples.

Conclusion: The tracking improves with shorter subframes.

6.5.30 Experiment number thirty: On the use of the residual signal

Filename: All excerpts

Description: The residual after sinusoidal analysis and synthesis is computed. Then the continuous quantized, the continuous unquantized, the original quantized, the original unquantized and the polynomial are added to this signal.

Conclusion: Each time we add a detail an improvement is audible, but most of the time very little. This is the ranking:

1. Original
2. Polynomial quality
3. Original phase, unquantized parameters;
4. Original phase, quantized parameters;
5. Continuous phase, unquantized parameters;
6. Continuous phase, quantized parameters;
7. Continuous phase, quantized parameters with noise synthesis.

Using quantized or not quantized parameters does not have a big effect.
The biggest improvements are given by using the original residual, especially for unvoiced signals, and the original phase, especially for voiced signals.
The polynomial is almost equal to the original; the only difference is given by the bad computation of transients and some residual in the low frequencies (below 1000 Hz).
The residual has a lot of picks and sometimes sounds still voiced.

6.5.31 Experiment number thirty-one: Maximal quality of encoding of the SSC

Filename: Extreme_Hole_Hearted_s44.wav
Extreme_Get_the_funk_out_s44.wav

Description: The excerpts are 3-minute songs. They are encoded with 240 samples subframe length and polynomial quality, which means original phase, unquantized parameters and second order description for sinusoid amplitudes. They are compared with the normal SSC encoding.

Conclusion: Quality improves a lot, very close to the original. Only the noise parts do not sound exactly like the originals. In the residual signal we can listen to some errors in the low frequencies, this is due to the fact that the smaller window causes some errors in their detection.

6.5.32 Experiment number thirty-two: Differences between continuous, original and polynomial phase

Filename: 012_s44.wav

Description: This is a synthetic sound with a frequency and amplitude-modulated sinusoid.

Conclusion: The estimation of the parameters is almost perfect, as a consequence the residual signal is almost zero everywhere. With the continuous phase reconstruction the steps in the approximation of the sinusoid are not only visible but also audible. The same is valid for the original phase reconstruction where this artefact is less audible. Perfect reconstruction is achieved with the polynomial phase.

6.5.33 Experiment number thirty-three: On the RPE additional encoding

Filename: aiff files provided by NatLab

Description: For each original excerpts, 6 files are generated, according to the following table:

Bit Rates for SSC + RPE configurations		
Original PCM	**Bit Rate**	705.6 kbit/s
	Filename	*_org.aiff
	Remarks	This is the mid channel derived from the stereo signals.
SSC	**Bit Rate**	24 kbit/s
	Filename	*_cod_ssc.aiff
	Remarks	These files are generated by the Natlab prototype.
SSC + RPE D8	**Bit Rate**	34 kbit/s
	Filename	*_rpe_d8_pc.aiff
	Remarks	Added to the SSC bit stream is a RPE decimation 8 layer, which is build up as follows: sub-frame size = 720/3 = 240 samples offset 3 bits pulses 30 pulses * 3 levels = 48 bits (6 * 8 bits) gains 5 bits (noise/RPE) _____ 56 bits/240 samples = 10.29 kbit/s
SSC + RPE D8 and phase information	**Bit Rate**	40 kbit/s (estimated)
	Filename	*_rpe_d8_pq.aiff
	Remarks	Added to the SSC bit stream is a RPE decimation 8 layer, and the phase information. The current bit rate of the phase information is

Bit Rates for SSC + RPE configurations		
		around 10 kbit/s, but it is expected that this bit rate can be lowered to 6 kbit/s while maintaining the same quality. This can be done by transmitting the phase for a certain frequency range (see TN Burak). If this does not work, an additional 4 kbit/s should be added to the estimated bit rate.
SSC + RPE D4	**Bit Rate**	43 kbit/s
	Filename	*_rpe_d4_pc.aiff
	Remarks	Added to the SSC bit stream is a RPE decimation 4 layer, which is build up as follows: sub-frame size = 720/3 = 240 samples offset 2 bits pulses 60 pulses * 3 levels = 96 bits (12 * 8 bits) gains 5 bits (noise/RPE) _____ 103 bits/240 samples = 18.93 kbit/s
SSC + RPE D4 and phase information	**Bit Rate**	50 kbit/s (estimated)
	Filename	*_rpe_d4_pq.aiff
	Remarks	Added to the SSC bit stream is a RPE decimation 4 layer, and the phase information.
SSC + RPE D2	**Bit Rate**	66.5 kbit/s (estimated)
	Filename	*_rpe_d2_pq_res2.aiff
	Remarks	Added to the SSC bit stream is a RPE decimation 4 layer, which is build up as follows: sub-frame size = 720/3 = 240 samples offset 1 bits pulses 120 pulses * 3 levels = 192 bits (24 * 8 bits) gains 5 bits (noise/RPE) _____ 198 bits/240 samples = 36.38 kbit/s Furthermore, the phase information is added (circa 6 kbit/s). The RPE sequence is generated using the following residual: *res = input - trans_q – sine_q* This means that quantization and estimation errors of the transients and sinusoids are incorporated in the residual signal. This is a different residual signal that is used in the noise coder. The other RPE configurations use the same residual as is used in the noise coder.

Table 5 – Description of the tests with a waveform parametric module codec

Conclusion: first of all, there is an improvement in the sound quality each time a better codec is used. This effect was expected as our previous experiments showed us that it is possible to have a perfect reconstruction using the original residual and the voiced signal re-synthesised. Comparing the best one (SSC+RPE D2) with the original, still some differences are audible, specially for speech and the castanets excerpts, but they are very small. 42 kbit/s more than in the SSC normal codec are used, which is a lot.

6.5.34 Experiment number thirty-three: Refresh rate for the original phase

Filename: es02_s44.wav

Description: Previous experiments showed that the use of the original phase improves audio quality, especially for speech excerpts. Next step is the comprehension of which is the frequency range where we are more sensitive to the original phase. Also the optimal refresh rate in the time domain is researched.

Conclusion: The algorithm used is the following: first of all the time refresh rate is inspected. The signal with 0 (continuous phase), 1 (original phase), 2, 3 … up to 100 refresh rate is computed. Then the threshold where the update rate was no more audible is inspected with informal listening. This threshold seems to be around 4. For the frequency range, before the upper limit is computed with a refresh rate of 1, then the lower limit is inspected. The results indicated that a band from 300Hz to 3500Hz, which is approximately the band of a telephone line, is enough.

6.6 Conclusions on the experiments

The SSC encoder scheme is a bottom-up system that saturates very fast: as a result, passing from 24 kbits to 32 kbits, the quality does not improve in a relevant way. Bottom-up systems start with a poor quality and a low bitrate and try to improve it, while top-down systems start from a good quality but a very high bitrate and try to reduce the latter, without losing too much in quality.

The transient module detects very poorly the transient positions. The idea of estimating the temporal envelope does not seem a good one as different frequencies follow in general different envelopes.

The noise module with a Laguerre filter with an order bigger than 20 does not improve its quality a lot. This noise part appears to be too much static, compared to the number of possible noise-like signals. It will always sound as a filtered white noise, even if the temporal enveloped is taken into account. Also increasing the resolution of the temporal envelope does not improve sensitively the output quality. The problem is that the noise part cannot sound differently than a filtered white noise and this in general is not the best choice to approximate the noise part of a signal.

The sinusoidal module “refuses” to estimate more sinusoids than necessary in general, even though if in the high frequency part of the spectrum lots of sinusoids are overestimated.

The quantization does not seem to affect a lot the quality, except for speech signals were its contribution is more relevant.

As a consequence, the first module I have decided to change was the original phase module (see experiments 6.5.34), then I worked on the improvement of the tracks behaviour.

7. Improvements

As a result of the previous experiments, the tracking algorithm should be improved.

In order to do so, I have found two solutions: improvement of the tracks, by filling the gaps that sometimes were found and the introduction of a new detection algorithm, developed by Francois Myburg in the Philips National Laboratories of Research. Also I have slightly changed the tracking process algorithm, added the possibility of changing the subframe size and the selection of the relevant range for the original phase.

7.1 Francois Myburg's algorithm

In this subsection, Francois Myburg's algorithm, which computes sinusoids poynomial parameters, is presented by a theoretical point of view.

7.1.1 The SPE module

The initial frequency estimates $\hat{\theta}_{k,2}$ must be provided as an input to the algorithm. Their estimation can be coarse. If the frequency $\hat{\theta}_{k,2}$ of sinusoid *k* in the current frame corresponds to the frequency of a sinusoid in the previous frame, the initial estimate of the frequency chirp can be taken equal to that of the corresponding sinusoid in the previous frame. Otherwise, $\hat{\theta}_{k,3}$ can be taken zero. Denote the model parameters in vector notation by

$$\theta = \left\lfloor \theta_{1,2}\theta_{1,3}\cdots\theta_{N_c,2}\theta_{N_c,3} \right\rfloor$$

At the start of an iteration, the model of the individual sinusoids is

$$s_{is}\left(\hat{\theta};\hat{A};n\right) = \sum_{k=1}^{N_c} A_k e^{j\left(\hat{\theta}_{k,2}n+\hat{\theta}_{k,3}n^2\right)}$$

where the optimal complex-valued amplitudes are found by solving the standard least-squares problem

$$\min_A \left\| s - s_{is}\left(\hat{\theta},\hat{A}\right)\right\|_w^2 .$$

The linear version of s_{is} around the model-parameter estimates is used, where regularization of the resulting system matrix is applied as required by the Levenberg-Marquardt method. The estimation errors are obtained, after which the model-parameter estimates are improved:

$$\hat{\theta}_{k,2} = \hat{\theta}_{k,2} + \hat{\Delta}_{k,2}$$

$$\hat{\theta}_{k,3} = \hat{\theta}_{k,3} + \hat{\Delta}_{k,3}$$

and the iterative process is repeated, until one of the following three condition is satisfied :

1. The improvement have become too small; this is an indication of the fact that we are very close to a local minimum;
2. Two sinusoids are very close in the frequency domain: only the one that has a better matching with the signal will continue, the other is removed;
3. If, for a specific sinusoid k, the following condition is satisfied $\frac{\left|\hat{\theta}_{k,3}\right|}{\hat{\theta}_{k,2}} N_s \geq MaxChirp$ where MaxChirp is a constant, whose value is the maximum growth in percentage of the frequency in a subframe, this sinusoid is removed and the number of sinusoids is decreased Nc = Nc - 1. We consider this occurrence to be an indication that the phase behaviour of the underlying partial cannot be modelled by a quadratic phase polynomial.

Two factors were found that improve the conditioning of the system matrix at each iteration.

1. As in the HPE module, using normalized vectors $p_{k;2;n}$ and $p_{k;3;n}$ significantly lower the condition number of the system matrix.
2. By ensuring that the partials are well spaced in frequency: $\min_{k,l}\left|\hat{\theta}_{k,2} - \hat{\theta}_{l,2}\right| \geq 2B_s$, where B_s denotes the DFT bin size, results in a low condition number of the system matrix.

One interesting phenomenon observed when applying this algorithm to real-world signals is that sinusoidal frequencies can converge to the same value. This usually happens when the underlying non-stationary partial exhibits more than one spectral peak. Given this observation, two sinusoids k and l are merged when $\left|\hat{\theta}_{k,2} - \hat{\theta}_{l,2}\right| \leq \delta_{\min}$, where $\delta_{\min}$ will be specified in Section 3.5, and the number of individual sinusoids is decreased Nc = Nc - 1.

7.1.1.1 Levenberg-Marquardt optimisation

The main difference between Gauss-Newton optimization and Levenberg-Marquardt optimization is that regularization of the system matrix H in the form of

$$(H + \lambda I)\hat{\Delta}_x = P$$

is applied. Regularization has two key advantages. Firstly, in the case that H is ill conditioned, regularization can improve its condition. However, it is worthwhile to keep the condition number of H in mind, and to find ways of lowering it to avoid unnecessary regularization. Secondly, for a properly chosen λ at each iteration step, the method becomes a descent method, which guarantees convergence.

Thus, the Gauss-Newton algorithm described in the previous section is extended as follows. Denote the model-parameter estimates from the previous iteration i - 1 by $\hat{x}^{(i-1)}$. The cost function at the end of iteration step *i - 1* is

$$c^{(i-1)} = \left\| s - s_{\mathrm{mod}\,el}\left(\hat{x}^{(i-1)}\right)\right\|_w^2$$

During iteration step *i*, the parameters are improved by solving the normal equations

$$\left(G^{(i)} + \lambda I\right)\hat{\Delta}_x^{(i)} = P^{(i)}$$

where a suitable λ has to be chosen. For each λ, the cost function associated with the updated parameter estimates is

$$c_i(\lambda) = \left\| s - s_{\mathrm{mod}\,el}\left(x^{(i-1)} + \hat{\Delta}_x^{(i)}\right)\right\|_w^2$$

The aim is to find the smallest λ in iteration *i*, denoted by $\lambda^{(i)}$, for which $c_i\left(\lambda^{(i)}\right) \le c^{(i-1)}$. To this end,

the strategy proposed by Marquardt is employed. Compute $c_i(\lambda)$ for both $\lambda = \lambda^{(i-1)} / \upsilon$ and $\lambda = \lambda^{(i-1)}$, where $\lambda^{(i-1)}$ is the choice of λ from the previous iteration and υ a constant. If

1. $c_i(\lambda^{(i-1)}/\upsilon) \le c^{(i-1)}$, let $\lambda^{(i)} = \lambda^{(i-1)}/\upsilon$;
2. $c_i(\lambda^{(i-1)}/\upsilon) > c^{(i-1)}$ and $c_i(\lambda^{(i-1)}) \le c^{(i-1)}$, let $\lambda^{(i)} = \lambda^{(i-1)}$;
3. $c_i(\lambda^{(i-1)}/\upsilon) > c^{(i-1)}$ and $c_i(\lambda^{(i-1)}) > c^{(i-1)}$, multiply $\lambda^{(i-1)}$ by powers of υ until, for some smallest m, $c_i(\lambda^{(i-1)}\upsilon^m) \le c^{(i-1)}$. Let $\lambda^{(i)} = \lambda^{(i-1)}\upsilon^m$.

To make this algorithm feasible for practical implementation, it is worthwhile to limit λ by $\lambda_{min} < \lambda < \lambda_{max}$. When the parameter estimates $\hat{x}^{(i-1)}$ are very close to a minimum of the cost function $c^{(i-1)}$, Step 3 of the algorithm will be repeated many times without any real practical benefit. Hence the upper bound λ_{max} on λ. The lower bound λ_{min} on λ should be chosen such that a small condition number of the system matrix *H* is guaranteed. A value of υ = 10 and an initial value of $\lambda = 10^{-2}$ are suggested by Marquardt. The iterations have converged when

$$\max_k\left(\frac{\hat{\Delta}_{x_k}^{(i)}}{\hat{x}_k^{(i)}}\right) \le \Delta_{\min}$$

A suitable value for Δmin is 10^{-5}.

The HPE and SPE modules, described in the following two sections, utilize the Levenberg-Marquardt method, where the strategy given above for choosing $\lambda^{(i)}$ is used. Furthermore, while the stopping criterion used in the SPE module is based on this last equation, the HPE module utilizes an alternative-stopping criterion.

7.1.2 The CAP module

To describe how the amplitude parameters are obtained, we define $s_{\sin}[n]$ as

$$s_{\sin}(\hat{x};A;n)=\sum_{k=1}^{N}(a_{k,1}+a_{k,2}n)\cos(\theta_{k,1}+\theta_{k,2}n+\theta_{k,3}n^2)$$

The optimal amplitude and constant phase parameters are found by solving the linear least-squares problem:

$$\min_A \|s-s_{\sin}(\hat{x};A)\|_w^2$$

where the optimal **A** contains the desired parameters.

The amplitude sweep $\hat{a}_{k,2}$ is restricted by

$$\hat{a}_{k,1}+\hat{a}_{k,2}n>0$$

to ensure that the instantaneous amplitude is always positive for the duration of the frame.

7.2 Changing the subframe size

One of the first modification that I have made to the original code of the SSC is the possibility of changing the size of the subframe. The value of 360 was somewhere hard coded in the program, so that some values and coefficients needed to be computed again. The value of the subframe size must be anyway multiple of 12, as the signal is downsampled by three in the coarse scale and depending on the transient position, four possible windows are considered.

In fact, from the experiments it was clear that using higher order in the polynomials of the sinusoids, required (and also let) longer subframe sizes.

In particular for harmonique excerpts it is possible to use subframes of 480 samples, and also have a better output quality (less noisy).

7.3 Range of relevance of the original phase

Another module that I have implemented allows the choice of the range of frequencies where the original phase will be used. It is likely that humans are no more sensitive to the original phase in the high frequencies, and that in the low frequencies this changes so slowly that the continuous phase could be used at its place. From the experiments, this range was fixed from 300 to 4000 Hz. This range can be chosen by an option on the command line of the encoder.

7.4 The link tracks module

I have noticed that sometimes some holes or gaps were left between tracks, especially near transients, where the analysis is more difficult. The causes of that fact are a bad estimation of the fact that the frequency has been removed by the psycho acoustic module.

In order to solve this problem, I have added to the SSC scheme a new module that fills in gaps up to 2 subframes. 3 is in fact the length of the shortest track allowed.

The algorithm is the following:

First of all, 0 gaps are filled. With this expression I mean the tracks that have a sort of break along their path. Then 1 and 2 gaps are filled. The idea is always the same: tracks are extended using the polynomial estimations, and if the extremities are enough close, then the tracks are linked. The thresholds are different depending on the hole. For largest hole, the threshold is lower, so that a very precise matching is required.

This algorithm is applied only to frequencies below 8 KHz, as it is not considered useful to link tracks above this frequency.

In the Figure 35 and Figure 36, around subframe 75 (horizontal axis), it is possible to see two gaps filled by this algorithm.

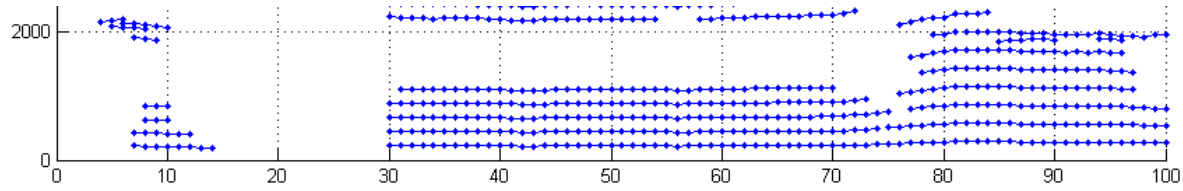

Figure 35 – An harmonic excerpt, without the link tracks module

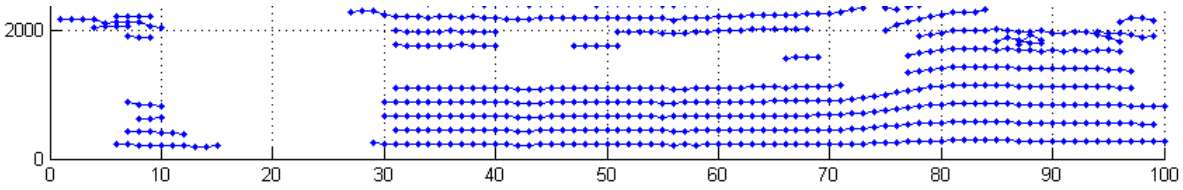

Figure 36 – The same excerpt of the previous figure, with the link module

7.5 The tracking process module

In this paragraph, the modifications made to the tracking process algorithm will be explained.

First of all, the old algorithm is presented, then the new method is briefly discussed.

Consider two consecutive subframes, n and n+1. Let f(n) and f(n+1) be the set of frequencies that have been estimated for each subframes.

Three quantities are taken into account: frequency, amplitude and phase difference.

A matrix is computed as a cost function. The element j,k of this matrix is the product of the probability that the j^{th} frequency of subframe n is "different" from the kth element of subframe n+1. This difference is measured on frequency distance, ratio of the amplitudes and phase difference at the interface of the two subframes. The elements with the highest probability are linked, only one track can come out of a frequency and only one track can arrive to a frequency.

Also some thresholds are fixed, so that if one of those thresholds is passed, the probability is set to zero for that link.

This algorithm is very clever, but it has a defect. Distant values in time are compared and the same importance is given to amplitude and frequency differences. Of course, the track process is a matter of more the frequency domain than of the time domain.

In this sense I have slightly modified it. Less importance is given to amplitude differences and the phase difference is not computed anymore, as meaningless. Frequencies and amplitudes are compared along the frontier between two subframes, extending the estimates with their polynomial parameters.

To have an idea of this improvement, I have synthesized a critical signal, with fast frequency variations. Results are reported in Figure 37 and Figure 38.

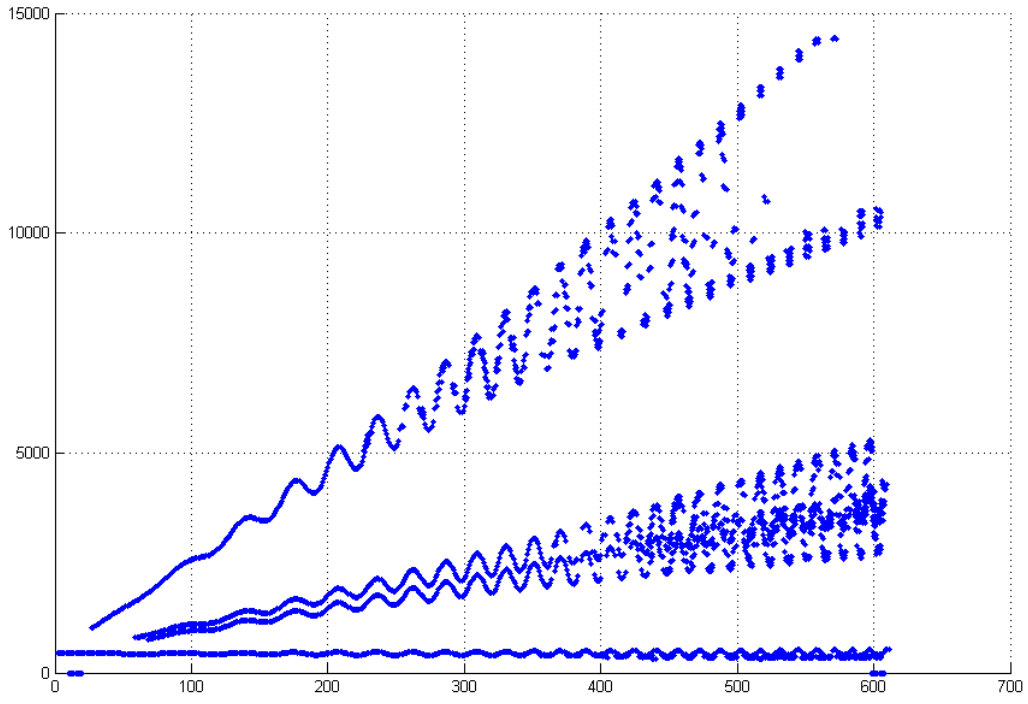

Figure 37 - A critical signal tracks with the old SSC coder

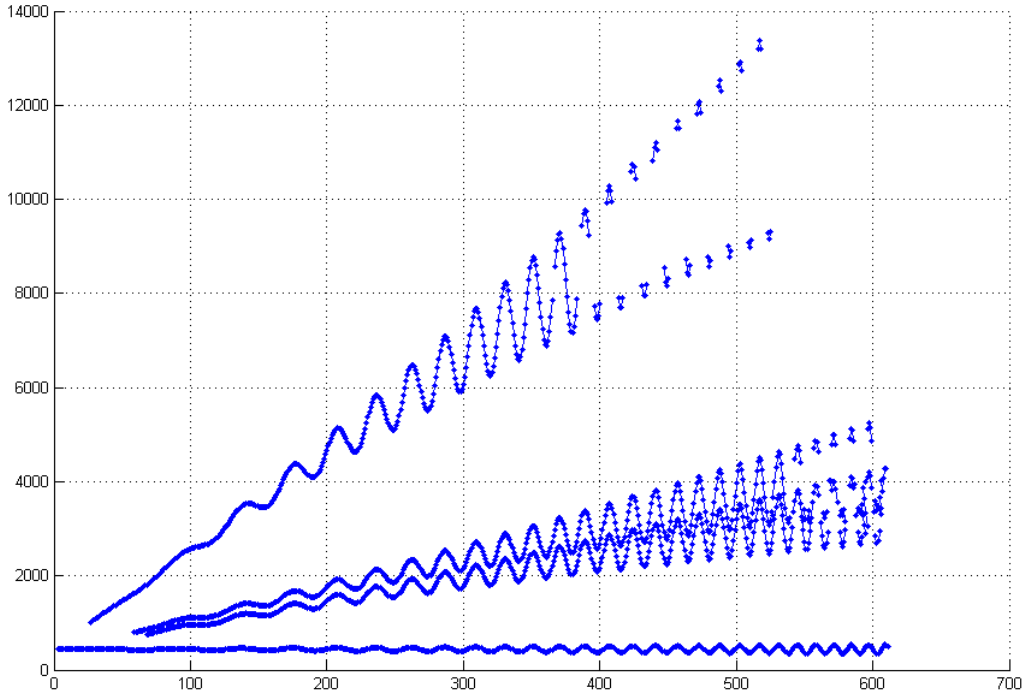

Figure 38 – Tracks improve with the new algorithm

8. Conclusions and recommendations

In this chapter conclusions on the work and recommendations for future developments are presented.

8.1 Conclusions

During this internship, I had the possibility of looking deeply inside at the way an audio codec is programmed, and to actually see what of all the problems we have studied during our academic year at the IRCAM, are taken into account in its engineering. The code was huge but its structure was so regular that eventually it was not difficult to understand its behaviour. Also the use of the debug feature of Matlab allows following the code during its execution, so that after the first run, the big steps were already clear.

Then I planned the first serie of tests, and after that, the characteristics and the weak points of the SSC were also clear. The second serie of tests showed that the system was almost saturated, so that it would not be easy to improve its quality.

I then decided to find some solutions, and implemented them. This took more time than expected because the system performances became slower by a factor of 1000 and I had to work also at the optimization of the code.

The tracking of sinusoids, thanks to the extended sinusoid model, improved a lot.

As a conclusion, a small improvement of the quality and also of the bitrate has been achieved. The coded excerpts sound now less metallic, but a little bit more noisy. This is tipically true for speech excerpts, while for instruments the improvement is clear. Less sinusoids are coded, which lowers the bit-rate, and a distinction between sinusoids and noise can be partially made using the chirp estimations of the detected sinusoids. Finally, the possibility of using larger subframe size (upto 30 %) gives a big gain in the bitrate. This gain can be used for coding a finer quantization grid, or the original phase, or all the polynomials coefficients that are present in the new sinusoidal object.

8.2 Recommendations

At the end of the graduation project still a lot of open ends remain. Both on the subject of complexity reduction and quality improvement work remains.

1. **Insufficiently accurate transient representation**

The transient object should probably be completely revised, probably substituted by a segmentation object. When comparing the transient to a sinusoidal track in a spectrogram it seems logical that a transient should be coded in the time-domain (vertical line) rather than in the frequency domain (horizontal line), as is the case now.

2. **Quality loss because of continuation of phase**

The linking of sinusoidal components can probably be improved even more by implementation of frequency dependent linking criteria. The instantaneous phase of a low-frequency sinusoid e.g. will rotate much slower than that of a high-frequency sinusoid. Over time the estimate of this phase will thus be more precise. Linking can thus probably be stricter than in the case of high frequencies.
Currently, the frequency and the phase together describe the instantaneous phase function of a sinusoid. It is questionable whether such a representation is optimal for the purpose of audio coding. Other representations could be investigated.

3. **Quality loss caused by quantisation of sinusoidal parameters**

The rules for the quantisation of sinusoidal parameters should be improved. Experiments showed that the original phase is very important. Non-tracks e.g. do probably not need to be coded as precisely as a long track.

Apart from the recommendations mentioned above some more general recommendations are given that might be considered to further improve the codec:

Quality improvement:

To further improve the quality of the encoder, a better distinction between noise and sinusoids should be made. I would also suggest the introduction of a segmentation module, that informs the sinusoidal analyser of the type of segment it will have to process. Often sinusoids get coded as noise and noise gets coded as sinusoids. This is however a fundamental problem and will probably not be solved very easily.
In this report no mention has been made about bit-rate control. In order to use SSC in practical applications bit-rate control is necessary. This is a further constraint to the bit-rate/quality ratio. It should be investigated how bit-rate control can be incorporated within the codec. I would for instance suggest the use of a bit-reservoir. When the signal is easy to code, for instance just because it is a silence, bits could be saved for the future.

Complexity reduction:

An efficient extraction method for sinusoids is absolutely necessary to achieve real-time encoding. The iterative extraction of sinusoids as described withing Francois Myburg algorithm should be further investigated. After the introduction of this algorithm, performances of the SSC encoder went down of a factor of 100.
The linear regression process also demands a lot of calculations per sinusoid. Basically for every sinusoid, every time-sample gets multiplied six times, once for every pattern function, with a complex

value. These multiplications probably do not need to be performed on every sample. It could be investigated whether sub-sampling could be used to decrease the number of multiplications.

9. Bibliography

[1] Zwicker, E.
SUBDIVISION OF THE AUDIBLE FREQUENCY RANGE INTO CRITICAL BANDS (FREQUENZGRUPPEN)
Journal Acoustical Society of America, 1961, Vol. 33, p. 248.

[2] Moore, B.C.J.
AN INTRODUCTION TO THE PSYCHOLOGY OF HEARING
3rd ed., Harcourt Brace Jovanovich, Publishers, London, Academic Press, 1989.

[3] Oomen, A.W.J., A.G. Kohlrausch and R.J. van der Vleuten
A PSYCHO-ACOUSTIC MASKING MODEL FOR A SUB-BAND CODER - STATUS QUO
Nat.Lab. Technical Note 346/98, ASALE Report AR6-875020WO C7 S4, Philips Research Laboratories, 1999.

[4] Pan, D.
A TUTORIAL ON MPEG/AUDIO COMPRESSION
IEEE Multimedia Journal, Summer 1995 issue, 1995.

[5] Brandenburg, K. and G. Stoll
THE ISO/MPEG-AUDIO CODEC: A GENERIC STANDARD FOR CODING OF HIGH QUALITY DIGITAL AUDIO
92nd AES-convention, 1992, preprint 3336, Vienna, Austria.

[6] McAulay, R. and T. Quartieri
SPEECH ANALYSIS/SYNTHESIS BASED ON SINUSOIDAL REPRESENTATION.
IEEE Transactions on Acoustics, Speech and Signal Processing 1986, Vol. 34, Iss. 4, p. 744-754.

[7] den Brinker, A.C. and A.W.J. Oomen
FAST ARMA MODELLING OF POWER SPECTRAL DENSITY FUNCTIONS
Proceedings EUSIPCO 2000, Volume II, 5-8 september 2000, Tampere, Finland, p. 1229-1232.

[8] den Brinker, A.C. and A.W.J. Oomen
SINUSOIDAL CODING OF AUDIO AND SPEECH (SiCAS) - A LOW BITRATE UNIVERSAL AUDIO AND SPEECH CODER
Nat.Lab. Technical Note 335/98, Philips Research Laboratories, 1998.

[9] den Brinker, A.C. and A.W.J. Oomen
A FEASIBILITY STUDY ON SINUSOIDAL CODING FOR AUDIO AND SPEECH - A LOW BITRATE UNIVERSAL AUDIO AND SPEECH CODER
Nat.Lab. Technical Note 157/99, Philips Research Laboratories, 1999.

[10] den Brinker, A.C. and A.W.J. Oomen
DESCRIPTION OF PAREMETERIZATION FOR AUDIO AND SPEECH SIGNALS BASED ON A SINUSOIDAL APPROACH (SiCAS)
Nat.Lab. Technical Note 331/98, Philips Research Laboratories, 1998.

[11] Edler, B., H. Purnhagen and C. Ferekidis
TECHNICAL DESCRIPTION OF THE MPEG-4 AUDIO-CODING PROPOSAL FROM UNIVERSITY OF HANNOVER AND DEUTSCHE BUNDESPOST TELEKOM AG (REVISED)

Technical Note MPEG95/0414r, Int. Organisation for Standardisation ISO/IEC JTC1/SC29/WG11, 1996.

[12] Desainte-Catherine, M. and S. Marchand
HIGH-PRECISION FOURIER ANALYSIS OF SOUNDS USING SIGNAL DERIVATIVES
Journal Audio Engineering Society, 2000, Vol. 48, No. 7/8, July/August.

[13] ITU-R
SUBJECTIVE ASSESSMENT OF SOUND QUALITY
BS Recommendation 562-3, International Telecommunication Union, 1994.

[14] Schuijers, E.G.P.
METHODICAL LITERATURE SEARCH PROJECT – QUALITY SCALABILITY OF A PARAMETRIC AUDIO CODER
Technical Report, University of Technology Eindhoven (EUT), 2000.

[15] Francois Myburg
SINUSOIDAL ANALYSIS OF AUDIO WITH POLYNOMIAL PHASE AND AMPLITUDE
Philips Electronics Research Report, 2001

[16] Daniel Pressnitzer, Michele Castellengo
DEA ATIAM – NOTES ET SUPPORT DE COURS
Annee universitaire 2002 – 2003

[17] Richard Heusdens, Steven Van de Par
RATE-DISTORTION OPTIMAL SINUSOIDAL MODELING OF AUDIO AND SPEECH USING PSYCHO ACOUSTICAL MATCHING PURSUITS

[18] Markus Erne, G.S. Moschytz
AUDIO CODING BASED ON OPTIMIZATION TECHNIQUES
AES 17th International conference on High Quality Audio Coding

[19] Paolo Prandoni, Michael Goodwin, Martin Vetterli
OPTIMAL TIME SEGMENTATION FOR SIGNAL MODELING AND COMPRESSION
1997 IEEE

[20] Shlomo Dubnov, Naftali Tishby
HEARING BEYOND THE SPECTRUM
Department of Musicology, Hebrew University, Jerusalem 91904, Israel, article

[21] Steven van de Par, Armin Kohlrausch, Ghassan Charestan and Richard Heusdens
A NEW PSYCHOACOUSTICAL MASKING MODEL FOR AUDIO CODING APPLICATIONS
Philips and Delft University of Technology