Chapter 5

From audio to content

Giovanni De Poli - Luca Mion

5.1 Sound Analysis

Models for the *representation* the information from sound are necessary for the description of the information, from the perceptive and operative point of view. Beyond the models, analysis methods are needed to discover the parameters which allow sound description, possibly lossless from the physical and perceptual properties description.

Audio features extraction. When aiming to the *extraction* of information for sound, we need to discard every feature which is non relevant. This process of feature extraction consists of various steps, starting from pre-processing the sound, then windowing, extraction, and post-processing procedures. An audio signal classification system can be generally represented as represented in Figure 5.1.

Pre-processing: The pre-processing consists of noise reduction, equalization, low-pass filtering. In speech processing (voice has a low-pass behavior) a pre-emphasis is applied by high-pass filtering the signal to smooth the spectrum, to achieve an uniform energy distribution spectrum.

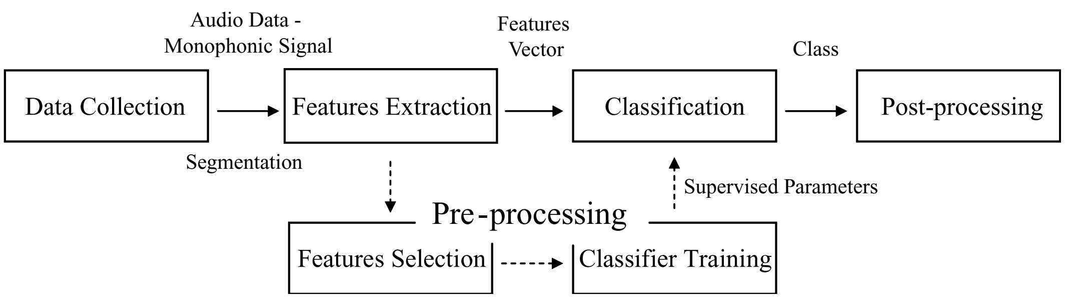


Figure 5.1: Scheme for supervised classification.

Windowing: Second step is to create frames from a continuous waveform signal. Frames are partially overlapping, as discuss in section 5.1.1

Extraction: Third step consists in obtaining a vector of features for each frame.

Post-processing: In the post-processing phase, the most relevant cues of the features vector are selected. For instance, earlier mel-cepstral coefficients can be lightly weighted when having low frequency noise.

The origins of audio features extraction derive from the research related to speech processing and representation. Initially, they proposed their representation for compressing data (i.e. for telephone applications), and later for speech recognition and synthesis. Features-based representation for non-speech audio was introduced at the end of the 90s, exploring different directions of research from the typical Short Time Fourier Transform; for example, representations based on Cochlear Models, Wavelets, and the MPEG audio compression filterbank. Audio features differ not only in how effective they are but also in their performance requirements, for instance FFT-based features might be more appropriate for a real time applications than a computationally intensive but perceptually more accurate features based on a cochlear filterbank.

Many features have been proposed by different communities in previous studies, e.g. from the musical instrument classification or psycho-acoustical studies. Features can be grouped according to various aspects; *steadiness*, because the features can represent either a value derived from signal, or a parameter from a model of signal behavior (e.g. mean, standard deviation); *time extent* of the description provided by the feature (global or instantaneous descriptors); *abstractness* of the feature, that is how the feature represents (e.g. spectral envelope can be represented via cepstrum or LPC on two different levels of abstraction); *extraction process*, which groups features according if features extracted from waveform, features extracted from transformed signal, features related to models, features based on auditory models.

Audio framework in MPEG-7 Mpeg7 became ISO/IEC 15398 standard in Fall 2001. It is the standard for describing multimedia content that provides the richest multimedia content description tools for applications ranging from content management, organization, navigation, and automated processing. The MPEG-7 standard defines a large library of core description tools, and a set of system tools provides the means for deploying the description in specific storage and transport environments. MPEG-7 addresses many different applications in many different environments, which means it needs to provide a flexible framework for describing multimedia data, including extensibility (using the Description Definition Language) and restrictibility (via the MPEG-7 Profiles under specification). The Mpeg7 descriptors for audio documents content consist of low-level and high-level descriptors. The task of supplying suitable descriptors for the extraction of significant characteristic is very interesting and it is also pretty hard. Audio signals can belong to different musical genres, played with various instruments, and can refer to speech-non speech sound, environmental sounds, mechanical sounds etc.

According to the features description proposed for the MPEG-7 standard, the extraction process can be summarized as depicted in Fig. 5.2. In the following, we will present many audio cues that are useful; some of them are also used as descriptors for standard MPEG-7 files, in that cases the Mpeg7 descriptor name will be indicated along with the feature description.

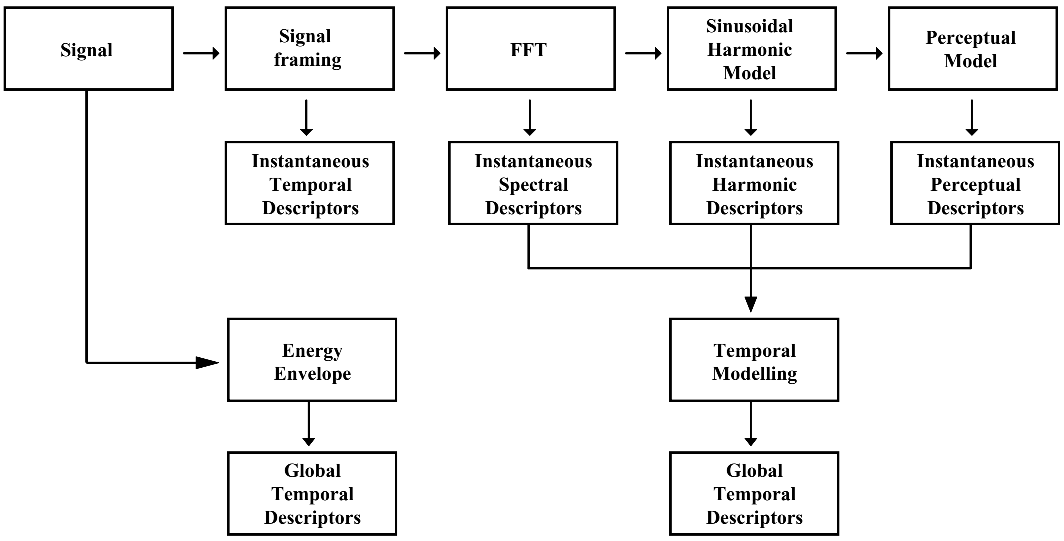

Signal
Signal framing
FFT
Sinusoidal Harmonic Model
Perceptual Model
Instantaneous Temporal Descriptors
Instantaneous Spectral Descriptors
Instantaneous Harmonic Descriptors
Instantaneous Perceptual Descriptors
Energy Envelope
Temporal Modelling
Global Temporal Descriptors
Global Temporal Descriptors

Figure 5.2: Features and extraction processes as proposed by Cuidado Project for MPEG-7 sound descriptors.

5.1.1 Time domain: Short-time analysis

Time domain methods divide according to the time extent of the description provided by the feature. Short and long time analysis techniques apply to different parts of the object (i.e. a sounding note) to describe the attack of the sound (short extent) or the loudness of a note (long extent).

In the case of short-time analysis, an hypothesis of stationarity of sound is now needed, with respect to the sampling rate. That is, we assume the signal to be slowly time-varying over intervals of a few milliseconds to manage the parameters as they were constants within a single frame. Considering the vocal signal for instance, this assumption can be justified by the fact that the words generation is affected by both the vocal chords and the entire phonation apparatus (larynx, mouth, tongue) with modifications not much quick, such to be considered constants within 100-200 ms. Therefore, the signal is divided into frames of e.g. 10 ms. The number of frames computed per second is called frame rate. A tapered window function (e.g. a Gaussian or Hanning window) is applied to each frame to minimize the discontinuities at the beginning and at the end. Consecutive frames are usually considered with some overlap for smoother analysis, and the duration between two fames defines the temporal accuracy, which is chosen according to the target accuracy. This analysis step is called *hop-size* H. On short time analysis, the signal is multiplied by a function $w[k]$ on $l = 0, \ldots, N-1$ which is null out the temporal window.

Windowing Temporal window defines the frame duration. The choice of the window depends on three factors: *(1)* it has to be short enough to be under the stationarity assumption; *(2)* it has to be long enough to comfortably allow the parameter computation and to reduce noise if affecting the signal; *(3)* windows have to entirely cover the parameter, that is the *frame rate* of the parameter has to be at least the inverse of the window duration.

The simplest window is the rectangular window, which gives the poorest quality result of all standard windows and requires a very large number of coefficients to yield a result equivalent to a higher quality window:

$$r[n] = \begin{cases} 1 & \text{for } 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Many applications use longer windows to satisfy the hypothesis of stationarity, but also changing the shape to emphasize the central samples (see Fig. 5.3). An alternative to the rectangular window 5.1 is the Hamming window, which is formulated from cosines and the like and follow sinusoidal curves:

$$h[n] = \begin{cases} 0.54 - 0.46\cos(\frac{2\pi n}{N-1}) & \text{for } 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

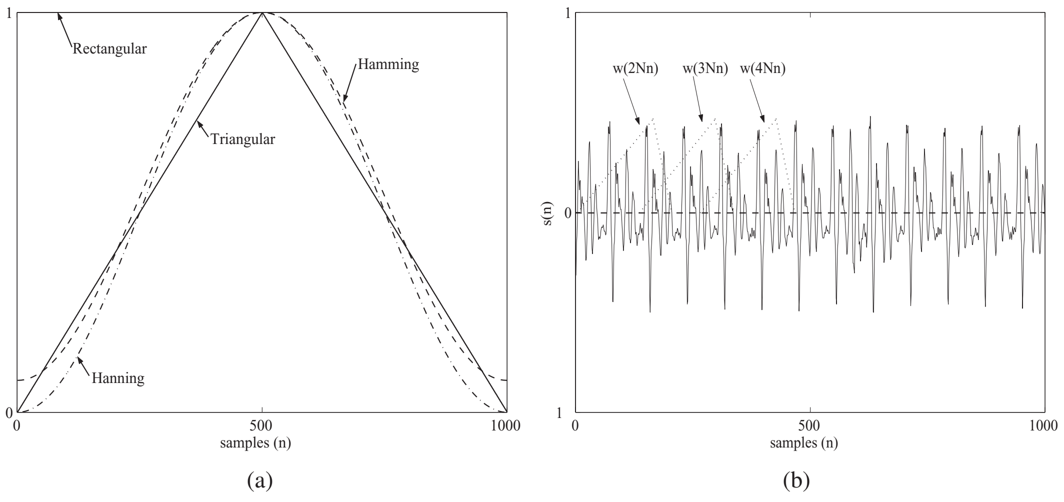


Figure 5.3: Various windows on the left; on the right, three windows over the signal$s[n]$, shifted from origin by $2N$, $3N$ e $4N$ samples

Preprocessing Some parameters in the time domain can also be represented by following formulation:

$$Q[n] = \sum_{m=-\infty}^{\infty} T[s[m]]w[n-m] = T[s] * w[n] \tag{5.3}$$

where $T[\cdot]$ is a (even non-linear) transformation weighted by a window $w[n]$. Before being processed, signal can be filtered to select the correct frequency band. In eq. 5.3, $w[n]$ can be a finite impulse response filter (FIR), which allows us to decrease the frame rate (less computational load), or alternatively an IIR filter; an example of IIR window is:

$$w[n] = \begin{cases} a^n & \text{for } n \ge 0 \\ 0 & \text{for } n < 0 \end{cases} \tag{5.4}$$

where $0 < a < 1$;

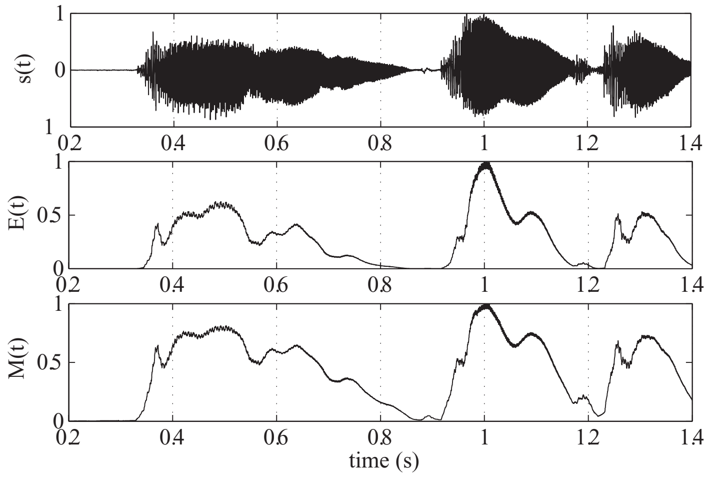


Figure 5.4: On the top: a short excerpt from violin performance of Handel's Flute Sonata in E minor. Other diagrams represent normalized *Short-Time Average Energy* and *Short-Time Average Magnitude*, computed using rectangular windows with N=100 samples and *frame rate* equal to signal sampling rate (8kHz).

5.1.1.1 Short-Time Average Energy and Magnitude

For a discrete signal, the *Short-Time Average Energy* is defined as follows:

$$E[n] = \frac{1}{N} \sum_{i=n-N+1}^{n} s[i]^2 \tag{5.5}$$

thus is equivalent to $Q[n]$ in equation 5.3 if $T[\cdot] = (\cdot)^2$.

A drawback of *Short-Time Average Energy* is to be affected when large signals occur. To face this problem, one solution is to define *Short-Time Average Magnitude* as follows:

$$M[n] = \frac{1}{N} \sum_{i=n-N+1}^{n} |s[i]| \tag{5.6}$$

which is equivalent to 5.3 when $T[\cdot] = |\cdot|$.

M-5.1

Write two MATLAB functions to compute Short-Time Average Energy e Magnitude.

M-5.1 Solution

```
Nframe=100;       %   numero di campioni per frame
Ns=max(size(s)); %   numero di campioni del segnale

for n=1:Ns;       %   calcola la Short-Time Average Energy
   E(n,1)=sum(s(max(1,n-Nframe+1):n).*...
      s(max(1,n-Nframe+1):n))/Nframe;
end;

for n=1:Ns;      %   calcola la Short-Time Average Magnitude
    M(n,1)=sum(abs(s(max(1,n-Nframe+1):n)))/Nframe;
end;

      %    disegna E(t) e M(t)
E=E/max(E);      %   normalizza E(t)
tempi = (1/fS)*[1:max(size(E))]; subplot(2,1,1);
plot(tempi,E); xlabel('time (s)'); ylabel('E(t)');

M=M/max(M);      %   normalizza M(t)
tempi = (1/fS)*[1:max(size(M))]; subplot(2,1,2);
plot(tempi,M); xlabel('time (s)'); ylabel('M(t)');
```

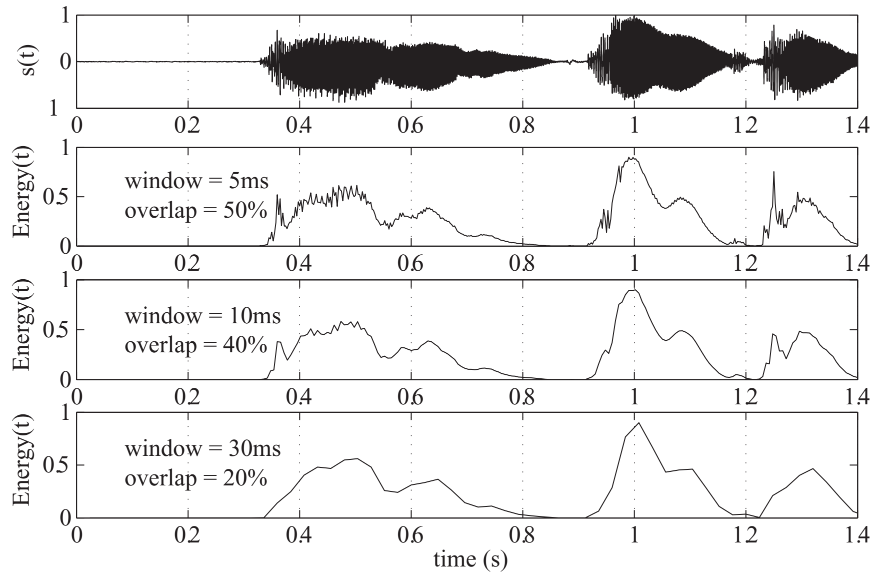


Figure 5.5: On the top: a short excerpt from violin performance of Handel's Flute Sonata in E minor. Other diagrams represent *Short-Time Average Energy* computed with different Hamming windows.

5.1.1.2 Short-Time Average Zero-Crossing Rate

Zero-Crossing Rate (*ZCR*) can yield useful information about the spectrum with low computational costs. *ZCR* represents the number of crossing through the zero signal, and it is mathematically described as the changing of the sign of two following samples. For narrow-band signals (e.g. sinusoids or band-pass filter output), from *ZCR* we obtain the fundamental frequency (F0) of the signal:

$$F0 = \frac{ZCR * F_S}{2} \tag{5.7}$$

where F_S is the signal sampling rate and ZCR is expressed as *zero crossing* for sample. We can have $ZCR = Q[n]$ when in 5.3 we use $T[s[n]] = |\text{sign}(s[n]) - \text{sign}(s[n-1])|/2$, and scaling the window $w[n]$ by a factor $1/N$; thus, we have

$$Z[n] = \frac{1}{N} \sum_{m=n-N+1}^{n} \frac{|\text{sign}(s[m]) - \text{sign}(s[m-1])|}{2} w[n-m] \tag{5.8}$$

where the sign of $s[n]$ is defined as follows:

$$\text{sign}(s[n]) = \begin{cases} 1 & \text{for } s[n] \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{5.9}$$

In Figure 5.6 the Zero crossing Rate of the word /sono/ is shown. Notice the high ZCR values at the beginning in correspondence of the unvoiced /S/ and low values for the voiced part. This properties can be exploited to distinguish voiced (periodic) from unvoiced sounds.

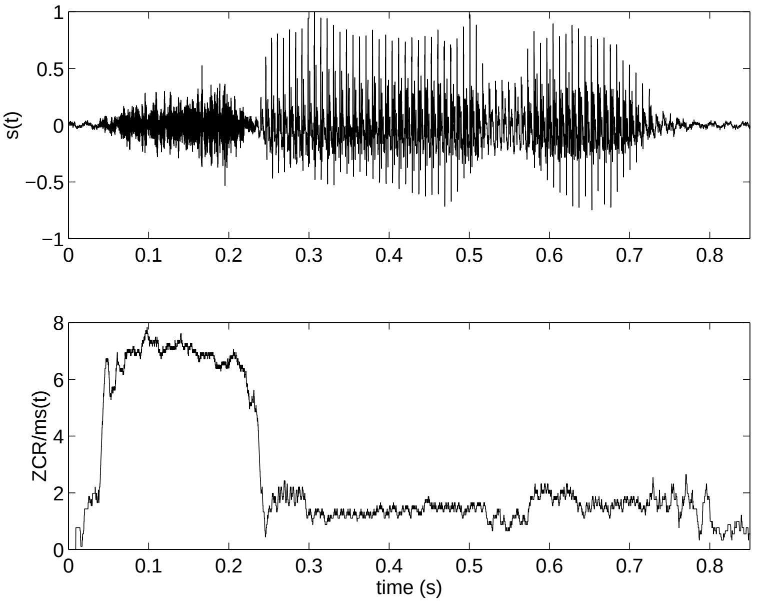


Figure 5.6: Zero-Crossing Rate of the word /SONO/.

M-5.2

Write a MATLAB function for Zero Crossing Rate computation.

M-5.2 Solution

```
Nframe = 100;          %  numero di campioni per frame
Ns = max(size(s));

for n = 1+Nframe:Ns;  %  calcola la Short-Time Average ZCR
   Z(n,1) = sum(abs(sign(s(n-Nframe+1:n))- ...
      sign(s(n-Nframe:n-1)))/2)/Nframe;
end;

Z=Z*fS/1000;           %  Zero-Crossing per ms

     %  disegna Z(t):
t = (1/fS)*[1:max(size(Z))];
plot(t,Z); xlabel('time (s)'); ylabel('ZCR/ms(t)');
```

5.1.1.3 Short-Time Autocorrelation Function

Signal *autocorrelation* is the inverse Fourier transform of the spectral density of the signal energy $C_s(f)$. It is formulated as follows:

$$\mathcal{F}[\phi[k]] = C_s(f) = |S(f)|^2 \tag{5.10}$$

For a discrete signal, it is defined as

$$\phi[k] = \sum_{m=-\infty}^{\infty} s[m]s[m+k] \tag{5.11}$$

Autocorrelation preserves the information related to harmonics, formants amplitude and their frequencies. Equation 5.11 shows that $\phi[k]$ is somehow representing the signal likeness to its shifted version. So, it will assume higher values when occurring delays k such that $s[m]$ and $s[m+k]$ have similar waveforms.

Some important properties of $\phi[k]$ are the followings:

1. it is an even function: $\phi[k] = \phi[-k]$
2. when $k = 0$ $\phi[k]$ takes its maximum value, $\phi[0] \geq |\phi[k]| \;\; \forall k$
3. $\phi[0]$ corresponds to the signal energy (or to the average power if the signal is periodic or non-deterministic)
4. if the signal is periodic with period P, the autocorrelation is periodic with the same period of the analyzed signal: $\phi[k] = \phi[k+P]$ (this is an important property when the signal periodicity has to be estimated). In fact it has maximal values at time lag P, $2P$, and so on. If the sound is quasi-periodic, we will have local maxima at lag P, $2P$, and so on (see Fig. 5.7).

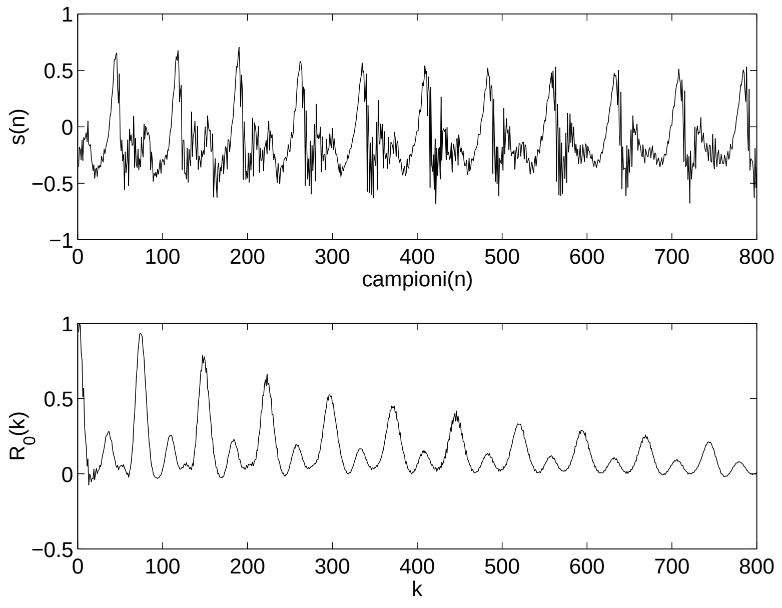


Figure 5.7: Autocorrelation of a voiced sound..

M-5.3

Write a MATLAB function for computing the *Short-Time Autocorrelation Function*.

M-5.3 Solution

```
Ns = max(size(s));   %no. of samples
window = ones(Ns,1); %rectangular window
s_w = s.*window;

for k = 1:Ns-1;    %compute ST autocorrelation
  R0(k) = sum(s_w(1:Ns-k) .* s_w(k+1:Ns));
end;
%plots
R0=R0/max(abs(R0));   %normalize R0
plot(1:max(size(R0)),R0); xlabel('k'); ylabel('R_0(k)');
```

Short-Time Autocorrelation Function (STAF) is applied for pitch extraction applications and speech-non speech signal discriminations. It is yielded by Eq. 5.11 after filtering the signal with windows $w[n]$:

$$R_n[k] = \sum_{m=-\infty}^{\infty} s[m]w[n-m]s[m+k]w[n-k-m] \tag{5.12}$$

This equation can be seen in the form:

$$R_n[k] = \sum_{m=-\infty}^{\infty} [s[n+m]w'[m]] \cdot [s[n+m+k]w'[k+m]] \tag{5.13}$$

where $w'[n] = w(-n)$; if we assume that $w'[n]$ has finite duration N we obtain:

$$R_n[k] = \sum_{m=0}^{N-1-k} [s[n+m]w'[m]] \cdot [s[n+m+k]w'[k+m]] \tag{5.14}$$

5.1.1.4 Short-Time Average Magnitude Difference Function

Beyond the *Short-Time Autocorrelation Function*, the detection of F0 can also be faced by means of the *Short-time Average Magnitude Difference Function* (AMDF). For a periodic signal with period P, succession $d[n] = s[n] - s[n-k]$ is zero when $k = 0, \pm P, \pm 2P, \ldots$, so we can consider the absolute value of the difference of $s[m]$ and $s[m-k]$ instead of their product:

$$\gamma_n[k] = \sum_{m=-\infty}^{\infty} |s[n+m]w[m] - s[n+m-k]w[m-k]| \tag{5.15}$$

We can have a simpler formulation when $w[n]$ is rectangular, with duration N:

$$AMDF[k] = \sum_{m=k}^{N-1} |s[m] - s[m-k]| \tag{5.16}$$

The AMDF of the signal of Fig. 5.7 is shown in fig. 5.8.

M-5.4

Write a MATLAB function for *Short-time Average Magnitude Difference Function* computing.

M-5.4 Solution

```
Ns=max(size(s));     %  numero di campioni

window=ones(ceil(Ns/2)+1,1);    %  finestra rettangolare

for k=1:floor(Ns/2)-1;  %  calcola la Short-Time AMDF
   STAMDF(k) = sum(abs(s(floor(Ns/2):Ns).* window - ...
      s(floor(Ns/2)-k:Ns-k).* window));
end;

     %  disegna STAMDF(t):
STAMDF=STAMDF/max(STAMDF);      %  normalizza STAMDF(t)
plot(1:max(size(STAMDF)),STAMDF); xlabel('k'); ylabel('AMDF(k)');
```

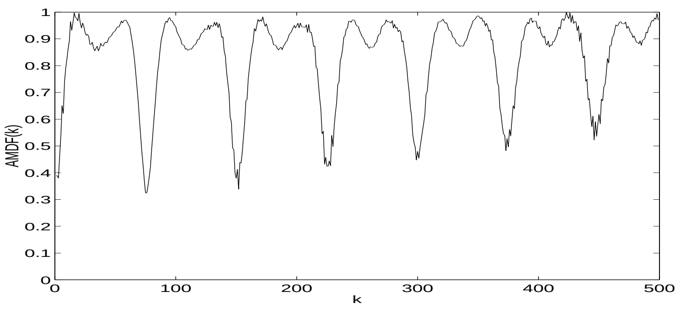


Figure 5.8: Short time AMDF of the voiced sound of fig. 5.7.

5.1.2 Audio segment temporal features

Descriptors computed after long-time localization of sound events (segmentation). These descriptors are extracted from the whole signal.

5.1.2.1 Temporal envelope estimation

Amplitude envelope can be defined as the variation of the amplitude of a note while sounding. This is often described through four phases called *attack, decay, sustain release* (ADSR). The basic idea to extract the temporal envelope starts with a low-pass filtering (i.e. 10 - 30 Hz), in order to extract the slow time changing components. One method to detect the envelope is based on *Short-Time Average Energy*, where window acts as a low-pass filter:

$$env[n] = \sqrt{E[n]}$$

5.1.2.2 ADSR envelope modelling

Modelling Attack, Decay, Sustain and Release of a note comes from the time representation of energy envelope. This feature is not always easily achievable because of overlapping notes with noise or with other notes. The estimation of attack can be achieved by means of either fixed or adaptive thresholds techniques, often empirically tuned. Fixed threshold methods consists of taking into account the possible presence of noise setting a threshold (e.g. to 20%) on energy envelope. Also, in order to take into account the possibility that the maximum of the envelope does not occur at the end of the attack, another threshold is set (e.g. to 90%). Adaptive techniques are based on the behavior of the signal during the attack; the best threshold is chosen along multiple notes by repeated tunings, according to the slope of the threshold crossing. I will discuss some onset detection techniques in Sec. 5.3. This set of features consists of:

Log-Attack Time [In Mpeg7 is LogAttackTime]"

$$LAT = \log_{10}(attack\_time)$$

where $attack\_time$ is the time duration of note attack. This feature has been proven to be strongly related to perceptual description of timbres.

Temporal Centroid [In Mpeg7 is TemporalCentroid]"

$$TC = \frac{\sum_t env(t) \cdot t}{\sum_t env(t)}$$

where $env(t)$ is the temporal envelope. This is a useful parameter to distinguish percussive sounds from sustained sounds.

Effective Duration is a measure of the time the signal is perceptually meaningful. It is approximately given by the time the energy is above a given threshold (e.g. 40%).

5.1.2.3 Pitch detection ($F0$) by time domain methods

The general problem of fundamental frequency estimation is to take a portion of signal and to find the dominant frequency of repetition.

Many applications are based on the detection of pitch, i.e. the dominat perceived frequency of a sound. The basic problem is to extract from a sound signal the fundamental frequency $F0$, which is the lowest sinusoidal component, or partial, which relates well to most of the other partials.

Difficulties arise from: (i) Not all signals are periodic; (ii) Those that are periodic may be changing in fundamental frequency over the time of interest; (iii) Signals may be contaminated with noise, even with periodic signals of other fundamental frequencies; (iv) Signals that are periodic with interval T are also periodic with interval $2T$, $3T$ etc, so we need to find the smallest periodic interval or the highest fundamental frequency; (v) Even signals of constant fundamental frequency may be changing in other ways over the interval of interest.

In a pitched signal, most partials are harmonically related, meaning that the frequency of most of the partials are related to the frequency of the lowest partial by a small whole-number ratio. The frequency of this lowest partial is $F0$ of the signal. The simplest approach to periodicity evaluation is based on the investigation of the time domain waveform. We may test each $F0$ hypothesis by testing how well the signal will resemble a delayed version of itself. An evaluation criteria can be either the correlation or the difference between the signal and its delayed version.

From the Short-Time Autocorrelation Function we obtain the information on the signal periodicity (fig. 5.9) by means of k_M, that is the first maximum after the one related to $k = 0$:

$$F0 = \frac{F_S}{k_M} \tag{5.17}$$

where F_S is the signal sampling rate. On the other side, if we use the Short-Time AMDF we have to consider the first minimum k_m after the one related to $k = 0$ (fig. 5.10). However sometimes we get a harmonic or sub-harmonic, depending on the shape of the spectrum. Whitening the signal by center-clipping is effective in minimizing this problem.

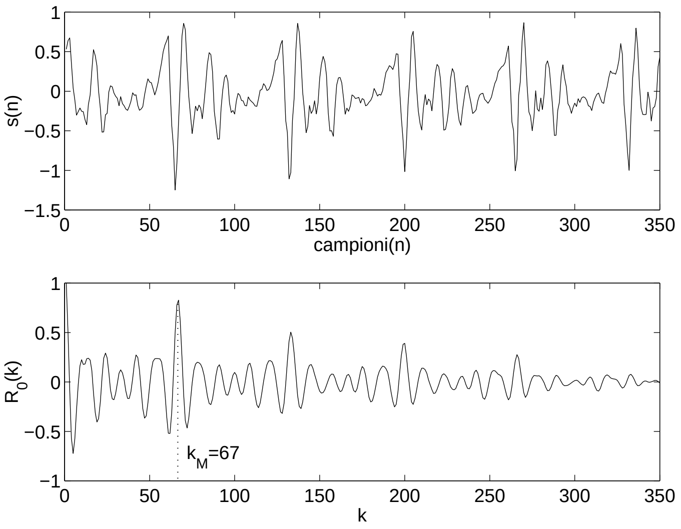


Figure 5.9: Frame of the phoneme /OH/ and its Short-Time Autocorrelation Function. The position of the second maximum at k_M indicates the pitch period.

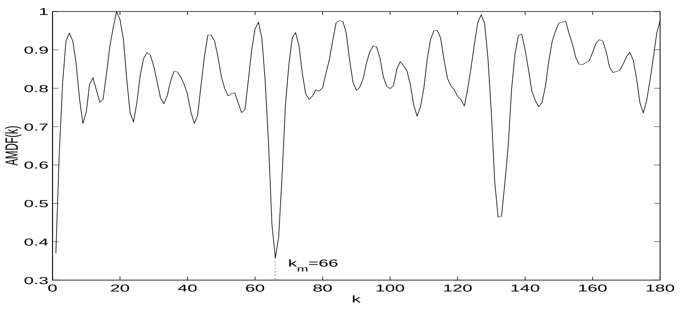


Figure 5.10: AMDF of the frame of the phoneme /OH/ of fig. 5.9. The position of the second minimum at k_m indicates the pitch period.

A pitch estimator normally proceeds in three steps:

- pre-processing: the signal is filtered to remove high frequency components;
- pitch extraction;
- post-processing to correct possible errors.

M-5.5

Compute the pitch with *Short-Time Autocorrelation Function*.

M-5.5 Solution

```
inizio=floor(fS*0.001);   %  salta il primo massimo
[massimo,kM] = max(R0(inizio:max(size(R0))));
kM=kM + inizio -1;
F0=fS/kM;
```

M-5.6

Compute the pitch with *Short-time Average Magnitude Difference Function*.

M-5.6 Solution

```
inizio=floor(fS*0.001);   %  salta il primo minimo
[minimo,km] = min(STAMDF(inizio:max(size(STAMDF))));
km=km + inizio -1;
F0=fS/km;
```

5.1.3 Frequency domain analysis

We have already seen in the signal based sound modeling chapter how a sound can be represented in the frequency domain. Moreover we presented analysis methods which allow the estimation of the spectral representation. In the following sections other methods for sound analysis in the frequency domain will be presented.

5.1.3.1 Energy features

Sinusoid plus noise representation Let's suppose $x[n]$ to be a sound, which is constituted by two components $x_S[n]$ (sinusoidal), and $x_R[n]$ (residual): $x[n] = x_S[n] + x_N[n]$, where $x_S[n] = \sum_{i=1}^{I} a_i cos(n2\pi f_i[n]/F_S + \phi_i[n])$.

In this form, a_i represents the amplitude (in linear scale) and f_i the frequency of i-th partial. These are time-changing parameters, which often are considered constant during a frame. The most used features, derived from this representation, are:

Total amplitude of the sinusoidal component, resulting by sum of the partials (dB expressed) within the frame:

$$AS_{tot} = 20 \log_{10} \left(\sum_{i=1}^{I} a_i \right)$$

where a_i is the amplitude of i-th partial;
Total amplitude of the residual component, resulting by sum of absolute values of the residual within the frame:

$$AR_{tot} = 20 \log_{10} \left(\sum_{n=0}^{M-1} |x_R[n]| \right) = 20 \log_{10} \left(\sum_{k=0}^{N-1} |X_R[k]| \right)$$

Total amplitude of the sound

$$\begin{aligned} A_{tot} &= 20\log_{10}\left(\sum_{n=0}^{M-1}|x[n]|\right) = 20\log_{10}\left(\sum_{k=0}^{N-1}|X[k]|\right) \\ &= 20\log_{10}\left(\sum_{i=1}^{I}a_i + \sum_{k=0}^{N-1}|X_R[k]|\right) \end{aligned}$$

Total Energy [In Mpeg7 is AudioPower]" estimates the signal power at a given time. It is estimated directly from the signal frame.

Noise to Harmonic Ratio - NHR is defined as the ratio between the energy of noise and the energy of harmonic part. For sustained sounds, energy modulation and fundamental frequency modulation are taken into account.

5.1.3.2 Spectral shape features

Spectral centroid [In Mpeg7 is AudioSpectrumCentroid]" is the barycenter of the spectrum. It is computed considering the spectrum as a distribution which values are the frequencies and the probabilities to observe there are the normalized amplitude.

$$BR = \frac{\sum_{k=0}^{N-1} k|X[k]|}{\sum_{k=0}^{N-1}|X[k]|} \cdot \frac{F_S}{N}$$

In the case of harmonic sounds, brightness is related to the fundamental frequency $F0$:

$$BR_{F0} = \frac{\sum_{i=1}^{I} i\ a_i}{\sum_{i=1}^{I} a_i} = \sum_{i=1}^{I} i\ w_i$$

Bandwidth is the difference between the lowest and highest frequency components of a signal:

$$BW = \frac{\sum_{k=0}^{N-1}|X[k]|\cdot|f_k - BR|}{\sum_{k=0}^{N-1}|X[k]|}$$

Spectral spread [In Mpeg7 is AudioSpectrumSpread]" is the spread of the spectrum around its mean value, i.e. the variance of the spectral distribution.

Spectral skewness gives a measure of the asymmetry of a distribution around its mean value.

Spectral slope represents the amount of decreasing of the spectral amplitude. It is computed by linear regression of the spectral amplitude.

$$Stilt = \frac{1}{\sum_{i=1}^{I} t_i^2} \cdot \sum_{i=1}^{I} \frac{t_i a_i}{w_i}$$

where

$$t_i = \frac{1}{w_i} \left(f_i - \frac{\sum_{i=1}^{I} f_i/w_i^2}{\sum_{i=1}^{I} 1/w_i^2} \right)$$

Spectral decrease still represents the decreasing of spectral amplitude, and it is supposed to be more correlated to human perception.

$$SD = \frac{1}{\sum_{i=2}^{I} a[i]} \sum_{i=2}^{I} \frac{a[i] - a[1]}{i-1}$$

Spectral roll-off is the frequency R_s so that 85% of the amplitude distribution is below this frequency. It is correlated to harmonic/noise cutting frequency.

$$\sum_{k=1}^{R_s} |X[k]| = 0.85 \cdot \sum_{k=1}^{N-1} |X[k]|$$

Spectral Flux is defined as the Euclidean distance between two amplitude spectrums of two close frames.

$$SF = \sum_{k=1}^{N-1} [N_t[k] - N_{t-1}[k]]^2$$

where $N_t[k]$ and $N_{t-1}[k]$ are respectively the spectral amplitude of frame FFT at instants t and $t-1$.

5.1.3.3 Harmonic features

Fundamental frequency [In Mpeg7 is AudioFundamentalFrequency]" is the frequency that a periodic waveform repeats itself. There are many methods in the literature to detect the fundamental frequency $F0$. For the Mpeg7 descriptor it is computed using the maximum likelihood algorithm. The method used in our experiments will be described in section **??**.

Noisiness [In Mpeg7 is AudioHarmonicity]" is the ratio between the energy of the noise an the total energy. It is close to zero for purely harmonic sounds.

$$Noisiness = \frac{\sum_{n=0}^{M-1} |x_R[n]|}{\sum_{n=0}^{M-1} |x[n]|}$$

Inharmonicity represents the divergence of the signal spectral components from a purely harmonic signal. It is computed as an energy weighted divergence of the spectral components. The range is [0,1].

$$HD = \sum_{i=1}^{I} |f_i - iF_o| \cdot w_i$$

Harmonic Spectral Deviation [In Mpeg7 is HarmonicSpectralDeviation]" is the deviation of the amplitude harmonic peaks from the global envelope.

$$HDEV = \frac{1}{I}\sum_{i=1}^{I}[\ a_i - spec\_env(f_i)\ ]$$

where $spec\_env(f_i)$ is the smoothed spectral envelope computed at frequency f_i of i-th harmonic.

Even-odds energy is useful to distinguish sounds like clarinet sound, which has low energy on even harmonics, differing from the trumpet sound which has similar behavior for both kinds of harmonics.

$$OER = \frac{\sum\limits_{i=even} a_i^2}{\sum\limits_{i=odd} a_i^2}$$

5.1.3.4 Pitch detection from the spectrum

For an harmonic signal F0 is the frequency so that its integer multiple explain the content of the digital spectrum. In fact for a periodic sound, all the partials are multiple of the fundamental frequency, that is $f_i = iF0$. In real sounds this is not completely true, and F0 can be calculated as the weighted sum of frequencies, normalized over all the harmonics.

$$F0 = \sum_{i=1}^{I} \frac{f_i}{i} \cdot w_i \tag{5.18}$$

where

$$w_i = \frac{a_i}{\sum_{i=1}^{l} a_i} \tag{5.19}$$

is the weight of i-th harmonic, with reference to the total sinusoidal component, and a_i is its amplitude. For signals with a more distributed spectrum, cepstrum analysis (sect. 5.2.2) is the form more conventionally used to make the analysis of pitch.

5.1.4 Perceptual features

Specific Loudness is associated to each Bark band z, see **?** for precise definitions. It can be simply defined as

$$N'(z) = E(z)^{0.23}$$

where $E(z)$ is the energy in the z-th bark-band.

Total Loudness is the sum of individual loudness.

$$N = \sum_{z=1}^{band} N'(z)$$

Sharpness is the perceptual equivalent to the spectral centroid, computed through the specific loudness of he Bark bands.

$$Sh = 0.11 \cdot \frac{\sum_{z=1}^{band} z \cdot g(z) \cdot N'(z)}{N}$$

where z is the index of the band and g(z) is a function defined as follows:

$$g(z) = \begin{cases} 1 & \text{for } z < 15 \\ 0.066 \exp(0.171z) & \text{for } z \geq 15 \end{cases} \tag{5.20}$$

5.2 Spectral Envelope estimation

Families of musical instruments can often be described by typical spectral envelope. When processing sound, operations that preserves the spectral envelope are roughly expressed as *pith shifting* operations with timbre preservations. Various techniques are used to represent the shape of a stationary spectrum. In sect. 2.4.7.2 we already presented the Linear Predictive (LPC) analysis.

M-5.7

In LPC analysis, the position of formants (resonances) is related to the poles of the estimated transfer function. Factorize the denominator of the transfer function and estimate the frequency of the formants. Note that if θ_k is the argument of z_k complex conjugate zero of the denominator, then its corresponding resonant frequency f_k derives from $\theta_k = 2\pi f_k/F_s$; the formant bandwidth B_k is related to the zero modulus by $|z_k| = \exp(-\pi B/F_s)$.

5.2.1 Filterbank

Filter-bank is a classical spectral analysis technique which consists in representing the signal spectrum by the log-energies at the output of a filter-bank, where the filters are overlapping band-pass filters spread along the frequency axis. This representation gives a rough approximation of the signal spectral shape while smoothing out the harmonic structure if any. When using variable resolution analysis, the central frequencies of the filters are determined so as to be evenly spread on the warped axis and all filters share the same bandwidth on the warped axis.

M-5.8

Write a MATLAB function for the spectral envelope computing, with the filterbank approach. Try a filterbank of frequency linearly spaced filters and logarithmic spaced filters (e.g. third octave filters).

M-5.9

Write a MATLAB function for the spectral envelope computing, with the gamma tone filterbank approach. Look in the literature or on the web for gammatone filter definition. gamma tone filters simulate the behaviour of the cochlea.

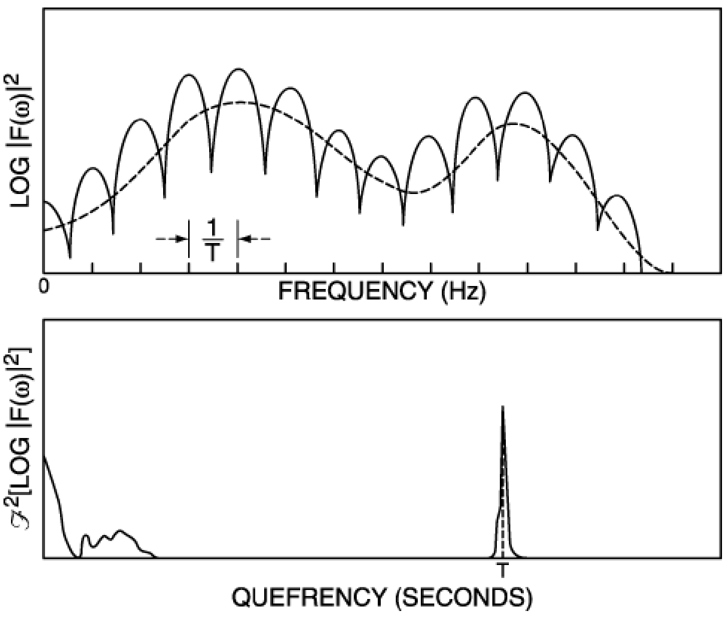


Figure 5.11: Example of cepstrum: on the top $\log|Y[k]|^2$; below, the related cepstrum $c[n] = \text{DFT}^{-1}(\log|Y[k]|)$

5.2.2 Spectral Envelope and Pitch estimation via Cepstrum

Families of musical instruments can often be described by typical spectral envelope. When processing sound, operations that preserves the spectral envelope are roughly expressed as *pith shifting* operations with timbre preservations. Various techniques are used to represent the shape of a stationary spectrum. Cepstrum method allows the separation of a signal $y[n] = x[n] * h[n]$, (source-filter model), where the source $x[n]$ passes through a filtered described by impulse response $h[n]$. Signal spectrum $y[n]$ results $Y[k] = X[k] \cdot H[k]$, which is the product of two spectrums (k is the discrete-frequencies index). The former is related to the source spectrum, and the latter to the filter spectrum. It's pretty difficult to separate these two spectrums, thus what is usually done is to extract the envelope (real) of the filter, and making the phase related to the source only. Cesptrum idea is based on the properties of logarithms : $\log(a \cdot b) = \log(a) + \log(b)$. Taking into account the logarithm of the absolute value of spectrum $Y[k]$, we get:

$$\log|Y[k]| = \log(|X[k] \cdot H[k]|) = \log|X[k]| + \log|H[k]| \tag{5.21}$$

If we consider the diagram for $\log|Y[k]|$ as a time-domain signal, we can distinguish two components: a quick oscillation, due to harmonic structure (rows), and a slower behavior related to the filter resonances (spectral envelope). We can separate the components by low/high-pass filtering signal $\log|Y[k]|$ (see Fig. 5.11, top). For components separation, one method is based on IFFT:

$$\text{DFT}^{-1}(\log|Y[k]|) = \text{DFT}^{-1}(\log|X[k]|) + \text{DFT}^{-1}(\log|H[k]|) \tag{5.22}$$

The part of $\text{DFT}^{-1}(\log|Y[k]|)$ towards the origin describes the spectral envelope, far from excitation. There is a sort of line in correspondence with $\log|Y[k]|$ periodicity, and thus to sound periodicity (see Fig. 5.11, below). At this point the cepstrum name origin comes out. Cepstrum word corresponds to spectrum when backward reading the former (ceps) part. The pitch can be estimated by the following procedure:

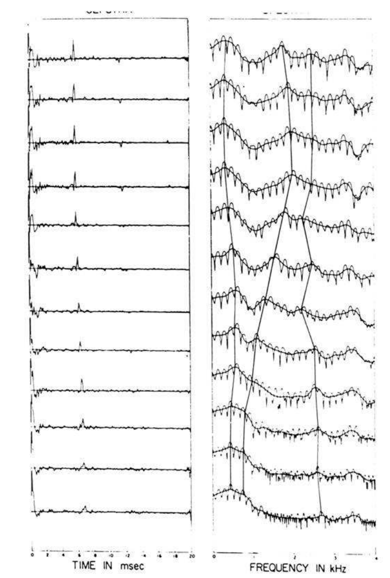


Fig. 7.15 Automatic formant estimation from cepstrally smoothed log spectra. (After Schafer and Rabiner [11].)

Figure 5.12: Automatically formant estimation from cepstrally smoothed log Spectra [from Schaefer Rabiner].

```
compute cepstrum every 10-20 msec
search for periodicity peak in expected range of $n_p$
**if** found and above threshold
    sound is periodic
    pitch=location of cepstral peak
**else** sound is not periodic
```

A drawback of the cepstral coefficients is the linear frequency scale. Perceptually, the frequency ranges 100-200Hz and 10kHz -20kHz should be approximately equally important, and standard cepstral coefficients do not take this into account.

We can notice that the maxima of spectral envelope corresponds to resonances (formats) which are very important to differentiate the vowels. In Fig. 5.12 it is shown how to individuate the formants from the spectral envelope.

M-5.10

Estimate the formants of a voice in a song and plot their position on the spectrogram.

5.2.3 Analysis via mel-cepstrum

Psychoacoustic studies have shown that human perception of frequencies goes with a logarithmic-like scale. Each tone of f (Hz), corresponds to a subjective value of pitch measured on the *m*el scale. As reference on the mel scale, 1000 Hz match 1000 mel. To obtain a value in the mel scale, a non-linear transformation on frequency scale is applied (see Fig. 5.13(a)), computed as follows:

$$\mathrm{mel}(f) = \begin{cases} f & \text{if } f \leq 1 \text{ kHz} \\ 2595 \log_{10}\left(1 + \frac{f}{700}\right) & \text{if } f > 1 \text{ kHz} \end{cases} \tag{5.23}$$

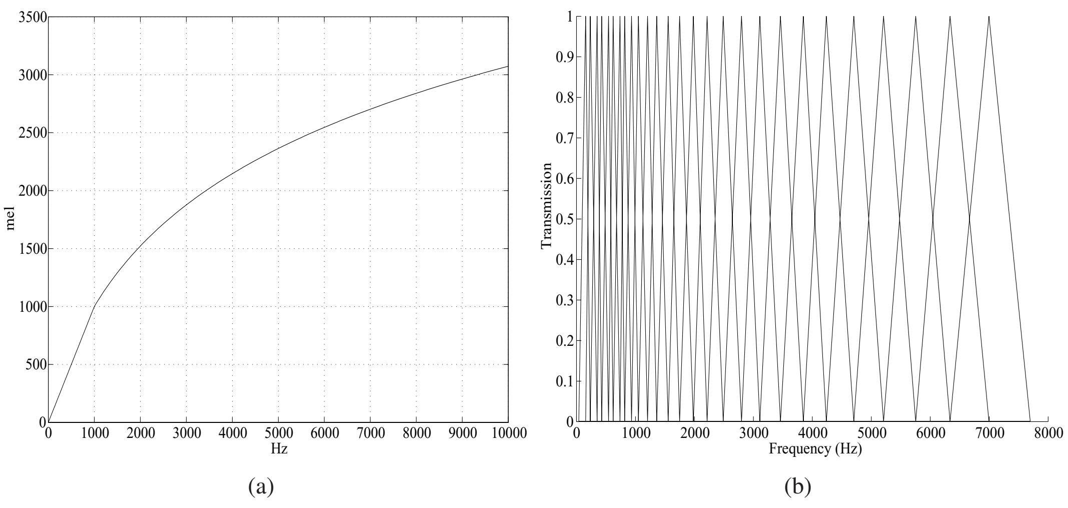


Figure 5.13: (a) Transformation from Hz to mel. (b) Mel-scale filterbank.

To apply the mel scale to cepstrum, triangular band-pass filterbanks are used, with central frequency in K mel values (see Fig. 5.13(b)). Each filter has bandwidth equal to the distance to previous filter central frequency, multiplied by two. First filter starts from 0. Thus, the bandwidth of filters below 1000 Hz is 200 Hz; then it will raise exponentially. Mel-cesptrum aim to estimate the spectral envelope of this filterbank output.

When Y_n is the logarithm of energy exiting from channel n, we can use the discrete time cosine transform DCT to obtain the mel-cepstral coefficients MFCC (mel frequency cepstral coefficients) by means of following equation:

$$c_k = \sum_{n=1}^{N} Y_n \cos\left[k\left(n - \frac{1}{2}\right)\frac{\pi}{N}\right] \qquad k = 0, \ldots, K \tag{5.24}$$

We can use the first K_m (with $K_m < K$) coefficients to draw a simplified spectral envelope $\tilde{C}(mel)$, similar to what we have seen for cepstrum:

$$\tilde{C}(mel) = \sum_{k=1}^{K_m} c_k \cos(2\pi k \frac{mel}{B_m}) \tag{5.25}$$

where $B_m =$ is the bandwidth, expressed in mel. A typical value for K_m in music classification is $K_m = 20$. We can notice that the coefficient c_0 is the mean value (in dB) of the energy in the channel of the filterbank. Thus it is related to the signal energy and often is not considered when we want to compare two spectral envelopes.

M-5.11

Write a MATLAB function for the spectral envelope computing, with the mel-cepstral approach and experiment it for different kinds of sounds. Compare the results obtained with the different spectral envelope algorithms.

In fig. 5.14 an example of mel-cesptrum analysis of a clarinet tone is shown. Spectra in dB, represented on a logarithmic frequency scale are compared: tone spectrum (high left); spectral envelope reconstructed with first 6 mel cepstral coefficients (low right), spectral envelope rebuilt from LPC analysis (low left); spectral envelope estimated with all mel cepstrum coefficients (low right).

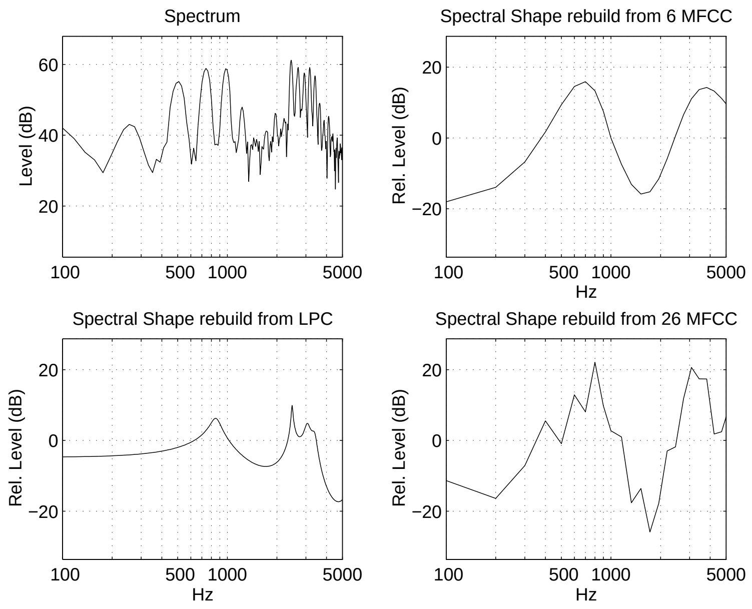


Figure 5.14: Example of mel-cesptrum analysis of a clarinet tone: tone spectrum (high left); spectral envelope reconstructed with first 6 mel cepstral coefficients (low right), spectral envelope rebuilt from LPC analysis (low left); spectral envelope estimated with all mel cepstrum coefficients (low right).

5.3 Onset Detection

This section is dedicated to the problem of the segmentation of the audio signal, through the onset detection. The concept of onset can be defined as the instant in which a new event starts. Event, in this context, can be defined as an auditory phenomenon that shows continuity inside the normal limits of perception. These auditory phenomenons can be expressive features (legato, vibrato, etc), timbre or notes. Note onset detection and localization is useful in a number of analysis and indexing techniques for musical signals. In most cases, onset will coincide with the start of the transient of the note, or the earliest time at which the transient can be reliably detected.

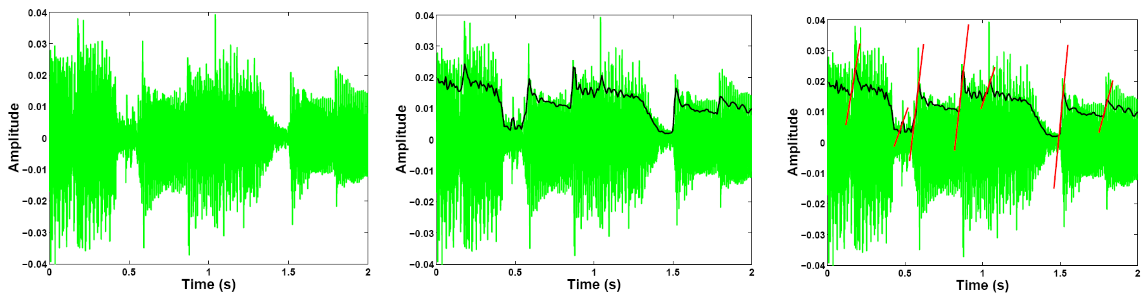
First two methods widely used for the onset detection, based in frequency domain and local energy detection, will be presented. Then in section 5.3.3 a more complete method will be presented.

5.3.1 Onset detection in frequency domain

In the spectral domain, energy increases linked to transients tend to appear as a broadband event. Since the energy of the signal is usually concentrated at low frequencies, changes due to transients are more noticeable at high frequencies This attack transient noise is particularly noticeable at high frequency locations, since at low frequencies, high concentrations of energy (in the bins corresponding to the first few harmonics of the played note) mask this effect. The High Frequency Content (HFC) function, is defined, for the jth frame, by:

$$D_H[j] = \sum_k k|X_j[k]|$$

where $|X_j(.)|$ is the spectral magnitude of the jth frame. Aim of this function is to emphasize the high frequency content of the sound and it works well for identifying percussive sounds. If compared with energy, this HFC function has greater amplitude during the transient/attack time. The HFC function



(a) (b) (c)

Figure 5.15: Example of onset detector based on local energy: time-domain audio signal (a), 40ms windowed RMS smoothing with 75% overlap (b), peaks in slope of envelope (c).

produces sharp peaks during attack transients and is notably successful when faced with percussive onsets, where transients are well modeled as bursts of white noise.

5.3.2 Onset detection from Local Energy

In order to find onsets, a detection function is cimpuyted, i.e. an intermediate signal that reflects, in a simplified form, the local structure of the suond and manifests the occurrence of transients in the original signal.

Despite the number of variants, practically all time domain methods are based on the calculation of a first order difference function of the signal amplitude envelopes and taking the maximum rising slope as an onset or an onset component. The envelope of the signal is computed as explained in sect. 5.1.2.1.

A common approach is to use as detection function $D(t)$ the time derivative of the energy

$$D(t) = \frac{dE(t)}{dt}$$

(or rather the first difference for discrete-time signals) so that sudden rises in energy are transformed into narrow peaks in the derivative. An example is the algorithm based on the surfboard method of **?**, which involves smoothing the signal to produce an amplitude envelope and finding peaks in its slope using linear regression.

In Fig. 5.15 the effect of a simple onset detector based on Local energy is shown. In Fig. 5.15(a) the time-domain audio signal; in Fig. 5.15(b) its smoothed amplitude envelope drawn in bold over it, computed by a 40ms windowed RMS smoothing with 75% overlap and in Fig. 5.15(c) peaks in slope shown by dotted lines tangential to the envelope. This method is lossy, in that it fails to detect the onsets of many notes which are masked by simultaneously sounding notes. Occasional false onsets are detected, such as those caused by amplitude modulation in the signal.

When we take into account perceptual aspects, we may notice that psychoacoustics indicates that loudness is perceived logarithmically. This means that changes in loudness are judged relative to the overall loudness, since, for a continuous time signal,

$$D_r(t) = \frac{d(\log E(t)}{dt} = \frac{1}{E(t)}\frac{dE(t)}{dt}$$

Hence, computing the first-difference of $\log(E[n]$ roughly simulates the ears perception of loudness. The relative difference function $D_r(t)$ as detection function effectively solves the problems of low

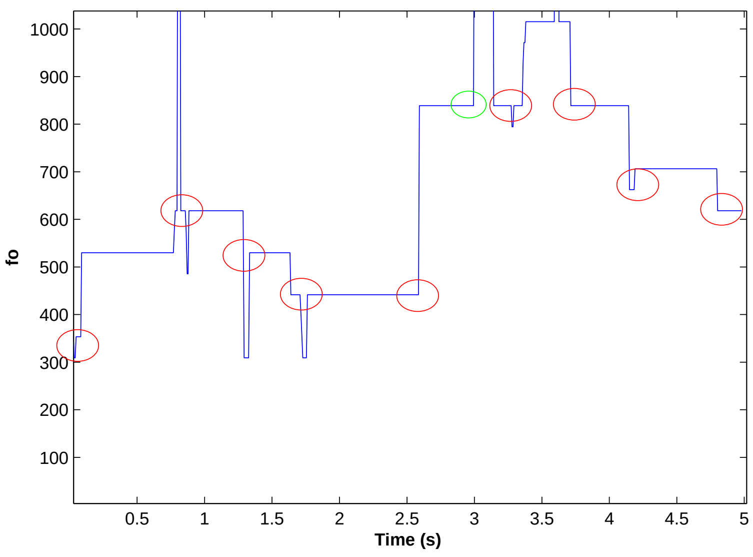


Figure 5.16: $F0$ temporal variation (with a window and one step with the values above considered) calculated by standard spectrum for the 5 first seconds of a considered signal. In this type of study it is verified the variation of the spectral shape in respect to the first maximum of the FFT.

sound, where the amplitude grows slowly, by detecting the onset times ealier and, more importantly, by handling complicated onset tracks, since oscillations in the onset track of a sound do not matter in relative terms after its amplitude has started rising.

5.3.3 Combining pitch and local energy information

In this section a case study on how to combine pitch and energy information for onset detection is presented. The analysis of pitch variation can help in finding tone onsets. Figure 5.16 shows the progression of pitch along five seconds of a signal (Handel's Flute Sonata in E minor (Adagio) - "hard" performed): The red and the green circles show the considered spectral variation zones for the onsets detection. The red circles indicate the zones of effective occurrence of onsets, while that the circles the green indicate the zones where onsets had not occurred.

As we will see in the following section, this type of analysis can be complemented with an analysis of the signal envelope (that it equivalent to a study of the variation of the energy). It can be said that main objectives for that are:

a) The elimination of false detections (circumscribed zones from the green circles), when the spectral disturbances are not follow for a minimum variation of energy (given by a minimum considered limit the note attack)

b) Add great variations of energy, still that are not verified significant spectral disturbances. This election is also made with base in a threshold of variation of energy in the note attack.

c) If there is a set of possible onsets in an interval of maximum distance, it is only considered the onset that corresponds to the lowest energy. All the others possibilities are discard.

Analysis of the temporal envelope In fig. 5.17 we see the temporal variation of the RMS envelope along the first 5 seconds of the considered signal. Instead of using a constant threshold that detected

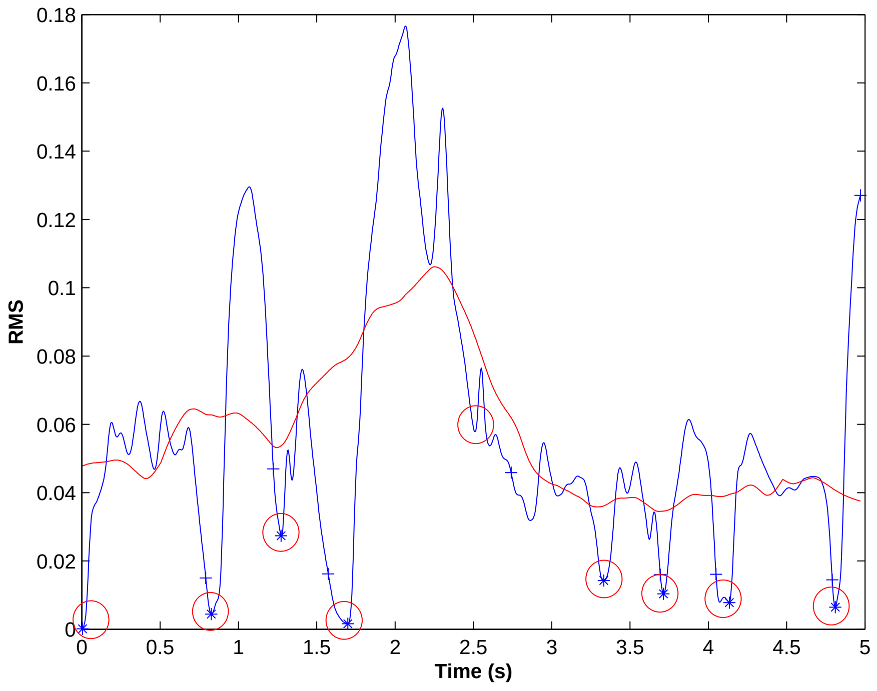


Figure 5.17: envelope of a 5 second window of signal.The blue line represents the RMS temporal variation, while that the red line represents the dynamic threshold that follows the RMS of the signal. The crosses on signal RMS represent the detected onsets. The circles represent the zones of effective onsets.

the brusque variations of the derivative of the RMS signal, we arrive at the conclusion that one adaptative threshold that follow the signal according to a more global dynamic, through the calculation of the average in a certain neighborhood of each point of the RMS, results much more efficient. The average was calculated between the 200 samples that surround each instant (1 second, for a step=0.005 s).The process consists in searching the minimum of RMS between two consecutive values detected by the threshold, since these values define a valley between two peaks.

Hybrid approach RMS-Frequency The analysis of the behavior of pitch and RMS envelope can be combined in the following steps.

1) We calculate all the onsets that result of the envelope analysis

2) In the case of the occurrence of onsets detected too much closed in the time, inside a given limit, is considered only in this interval the onset that has a lesser value of RMS. This limit consists of 0.25 of the mean of the measured distances between all detected onsets

3) We have considered that would be false detections (false positives) onsets that don't have a note attack in the energy domain greater than a given limit. We have eliminated to the list of onsets, that we have until this moment, the set of onsets that are below of this threshold. This threshold is calculated in relation to the average of the attacks of onsets considered until 2). Is considered the attack in the positive temporal direction and in the negative temporal direction. That is, the attack is defined as the energy jump that occurs between an onset and the next peak, and is defined also as the energy interval

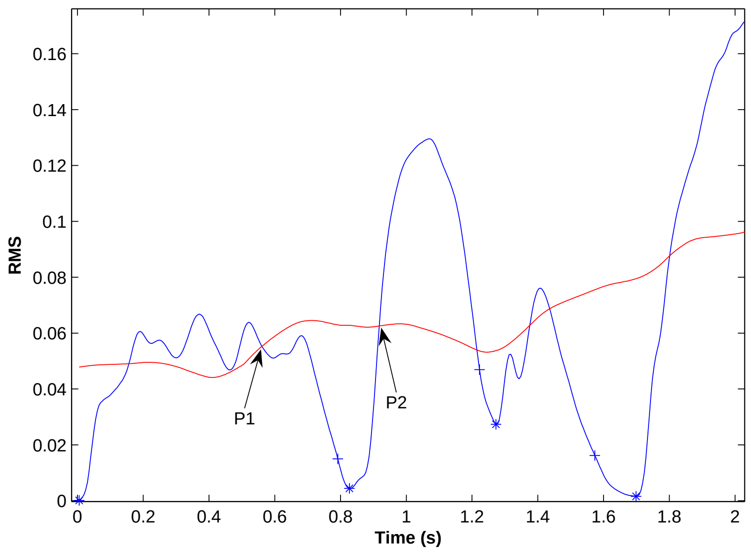


Figure 5.18: The points P1 and P2 are two values of the RMS signal detected by threshold. These values define a valley between two peaks. The value for the onset is the minimum that can be find between these two points.

between an onset and the immediately previous peak. This threshold consists of 0.3 of the average of the considered attacks.

4) We calculate all onsets that result of spectral disturbances, that in our case can be seen as disturbances in the temporal progression of pitch.

5) As in 2), in the case of the occurrence of onsets too much closed in time, inside of a given threshold, is considered only the onset that has a lesser value of RMS. This threshold is the same that was considered in 2).

6) Are considered here as valid onsets the onsets considered until 3) (valid onsets extracted by RMS) that are inside a given threshold of proximity in relation to the more closed onset in the set of onsets considered in 5) (onsets valid extracted by disturbances of pitch). This threshold of proximity corresponds to a maximum absolute distance of 0.1 seconds.

7) Of the onsets calculated by RMS analysis considered valid - before the onsets elimination according to the previous criterion (i.e. the set considered in 3)) we add to the list the onsets that had a note attack superior than a given threshold in the energy domain. The attack is here defined as the difference between the RMS value for the instant of the onset and the value of the next peak (that can be found between this onsets and the consecutive one). This limit is calculated relatively to the mean of all the considered attacks of onsets considered until 6). Numerically corresponds to 0,5 of the mean of these attacks.

5.4 Feature Selection

In many applications, reducing the dimensionality of the data by selecting a subset of the original variables may be advantageous for reasons including the expense of making, storing and processing measurements. The art of machine learning starts with the design of appropriate data representations.

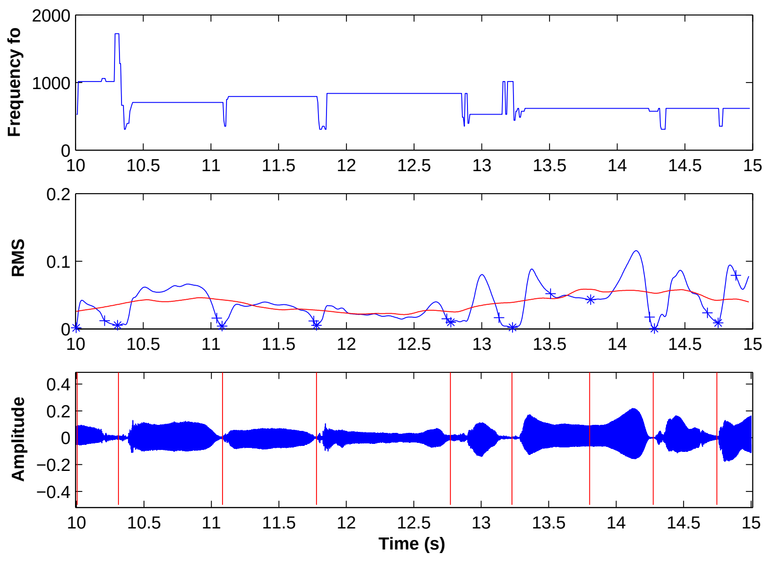


Figure 5.19: Detection of onsets for a five seconds window: the last figure represents the onset detection, where red lines indicate the instants for each detected onset.

Better performance is often achieved using features derived from the original input. The reasons for reducing the number of features to a sufficient minimum are that: (i) less features implies simpler models and then less training data needed; (ii) the higher the ratio of the number of training patterns N to the number of free classifier parameters, the better the generalisation properties of the resulting classifier; (iii) computational complexity, correlating (redundant) features.

Good features result in large between-class distance and small within-class variance; to perform this, the approaches divide according to: (i) examine features individually and discard those with little discrimination ability; (ii) to examine the features in combinations; (iii) linear (or nonlinear) transformation of the feature vector. Conventional statistical approaches such as comparing mean correlations within and between subgroups, principal components analysis (PCA), analysis of variance (ANOVA), and visualization techniques can be appropriately applied to determine which features to select. These techniques allow us to discover which data is redundant, which features have a relatively high variation between the subjects, and how the original feature set can be reduced without a big loss in the variance explained and recognition rate.

Variable subset selection procedure consists on measuring the relevance of a subset of input variables, and an optimization algorithm for searching for the optimal or a near-optimal subset with respect to the subset of variables. Procedures for variable subset selection can be classified into two groups: filter procedures and wrapper procedures. In case of *filter* procedures, the relevance measure is defined independently from the learning algorithm. The subset selection procedure in this case can be seen as a preprocessing step. In case of *wrapper* procedures, the relevance measure is directly defined from the learning algorithm, for example, in terms of the cost of the learning and the precision achieved by classification algorithm. On the other side, wrapper procedures need the number of possible parameters to be as low as possible, so that the algorithm should be highly computationally efficient.

5.4.1 One-way ANOVA

The analysis of variance (ANOVA) helps to identify the features which highlight differences between groups (subjects).

Let a database contain records of g subjects (number of groups) and n_l sessions for each subject. Let X_{l_j}, where l = 1, 2, ..., g and j = 1, 2, ..., n_l, be a random sample of size n_l from a population with mean μ_l, l = 1, 2, ..., g. Anova is used to investigate whether the population mean vectors are the same, i.e. the null hypothesis of equality of means could be formulated as $H0$: $\mu_1 = \mu_2 = ... = \mu_g$, and if not, which implies components differ significantly. The F-test rejects $H0$ at level α if:

$$F = \frac{SSA/(g-1)}{SSW/(\sum_{l=1}^{g} n_l - g)} > F_{g-1,\sum n_l - g}(\alpha) \tag{5.26}$$

where $F_{g-1,\sum n_l-g}(\alpha)$ is the upper (100α)th percentile of the F-distribution with $g-1$ and $\sum n_l - g$ degrees of freedom; $SSA = \sum_{l=1}^{g} n_l(\overline{x_l} - \overline{x})^2$ is the sum of squares among groups, where $\overline{x_l}$ and $\overline{x}$ are estimates of group and overall sample means respectively; $SSW = \sum_{l=1}^{g}\sum_{j=1}^{n_l}(x_{l_j} - \overline{x_l})^2$ is the total within-group sum of squares, where x_{l_j} is an observation of the feature from subject l, session j.

5.4.2 Principal Component Analysis (PCA)

Principal component analysis (PCA) is concerned with explaining the variance-covariance structure of a set of variables through a few linear combinations of these variables. This means that the original feature space is transformed by applying e.g. a linear transformation via a PCA. PCA is a linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on. PCA can be used for dimensionality reduction in a dataset while retaining those characteristics of the dataset that contribute most to its variance, by keeping lower-order principal components and ignoring higher-order ones. Such low-order components often contain the "most important" aspects of the data.

Given a set of n observations on p observed variables, the purpose of PCA is to determine r new variables, where r is small relative to p. The r new variables, called principal components, must together account for most of the variation in the p original variables. Principal component analysis operates by replacing the original data matrix $\mathbf{X}$ by an estimate composed of the product of two matrices. In matrix notation the approximation $\widehat{\mathbf{X}}$ of the matrix $\mathbf{X}$ is given by $\widehat{\mathbf{X}} = \mathbf{Z}\mathbf{V}'$, where $\mathbf{Z}$ is the $(n \times r)$ matrix of observations on the first r principal components, and $\mathbf{V}(p \times r)$ is the matrix whose columns are the first r eigenvectors of $\mathbf{X}'\mathbf{X}$. The matrix $\mathbf{Z}'\mathbf{Z} = \Lambda$, where Λ is the diagonal matrix of r eigenvalues $\lambda_k, k = 1, 2, ..., r$. The sum of squares and cross products matrix for the principal components is therefore a diagonal matrix with diagonal elements λ_k, that decline in magnitude. The eigenvalues λ_k are given by:

$$\sum_{i=1}^{n} z_{ik}^2 = \lambda_k, k = 1, 2, ..., r$$

The sum of squares for each principal component is therefore given by the corresponding eigenvalue. The quantity to be minimised is given by:

$$tr(\mathbf{X} - \widehat{\mathbf{X}})'(\mathbf{X} - \widehat{\mathbf{X}}) = tr\mathbf{X}'\mathbf{X} - \sum_{k=1}^{r} \lambda_k$$

and hence if $r = p$, then this expression has the value zero. Each of the eigenvectors generates a portion of the total variation (or sum of squares) in $\mathbf{X}$ as measured by $tr(\mathbf{X}'\mathbf{X})$. The contribution to $\sum_{k=1}^{p} \lambda_k$ provided by $\mathbf{z_l} = \mathbf{Z}\mathbf{v}_l'$ is $\mathbf{z}_l'\mathbf{z_l} = \lambda_l$. The proportion of the total variance measured by $tr(\mathbf{X}'\mathbf{X})$, accounted for by the component $\mathbf{z_l}$ is given by $\lambda_l / \sum_{k=1}^{p} \lambda_k$. The number of components actually used for the approximation of $\mathbf{X}$ can be guided by the measure $\sum_{k=1}^{l} \lambda_k / \sum_{k=1}^{p} \lambda_k$, where $l \leq p$.

In other words, the first principal component is the axis passing through the centroid of the feature vectors that has the maximum variance therefore explains a large part of the underlying feature structure. The next principal component tries to maximize the variance not explained by the first. In this manner, consecutive orthogonal components are extracted. The principal components depend solely on the covariance or correlation matrix of the data.

5.4.3 Further feature subset selection

To find the optimal subset of features, we should form all possible combinations of *M* features out of the *D* originally available; the best combination is then selected according to any desired class separability measure *J*. In practice, it is not possible to evaluate all the possible feature combinations. Thus the search is for a satisfactory set of features instead of an optimal set and greedy approaches are used.

Sequential backward selection (SBS) consists of choosing a class separability criterion *J*, and calculate its value for the feature vector which consists of all available features (length = *D*). Eliminate one feature, and for each possible resulting combinations (of length *D-1*) compute *J*. Select the best, and continue this for the remaining features, and stop when you have obtained the desired dimension *M*. This is a suboptimal search procedure, since nobody can guarantee that the optimal *r-1* dimensional vector has to originate from the optimal *r*-dimensional one. This method is good for discarding a few worst features.

Sequential forward selection (SFS) consists of computing the criterion *J* for all individual features and select the best. Form all possible two-dimensional vectors that contain the winner from the previous step, calculate the criterion for each vector and select the best; continue adding features one at time, taking always the one that results in the largest value of the criterion *J*, and stop when the desired vector dimension *M* is reached. This method is particularly suitable for finding a few good features.

Both SBS and SFS suffer from the nesting effect: once a feature is discarded in SBS (selected in SFS), it cannot be reconsidered again (discarded in SFS).

5.5 Auditory model for sound analysis

5.5.1 Auditory modeling motivations

Every sound classification and recognition task is preceded by an acoustic analysis front-end, aiming to extract significant parameters from the time signal. Normally, this analysis is based on a model of the signal or of the production mechanism. Short-Time Fourier Transform (STFT), Cepstrum, and other related schemes were all developed strictly considering physical phenomena that characterise the speech waveform and are based on the quasi-periodic model of the signal. On the other hand LPC technique and all its variants were developed directly by modelling the human speech production mechanism. Even the most simple physical models of musical instruments are highly non linear; thus they are not suitable to be used for analysis purpose. In music research and speech recognition the focus is on perceived sound rather than on physical properties of the signal or of the production mechanism. To this purpose, lately, almost all these analysis schemes have been modified by incorporating, at least at a very general stage, various perceptual-related phenomena. Linear prediction on a warped frequency scale STFT-derived auditory models, perceptually based linear predictive analysis, are a few simple examples of how human auditory perceptual behaviour is now taken into account while designing new signal representation algorithms. Furthermore, the most significant example of attempting to improve acoustic front-end with perceptual related knowledge, is given by the Mel-frequency cepstrum analysis of speech, which transforms the linear frequency domain into a logarithmic one resembling that of human auditory sensation of tone height. In fact, Mel Frequency Cepstrum Coefficients (MFCC) are almost universally used in the speech community to build acoustic front-end for Automatic Speech Recognition (ASR) systems.

All these sound processing schemes make use of the “short-time” analysis framework. Short segments of sounds are isolated and processed as if they were short segments from a sustained sound with fixed properties. In order to better track dynamical changes of sound properties, these short segments which are called analysis frames, overlap one another. This framework is based on the underlying assumption that, due to the mechanical characteristics of the generator, the properties of the signal change relatively slowly with time. Even if overlapped analysis windows are used, important fine dynamic characteristics of the signal are discarded. Just for that reason, but without solving completely the problem of correctly taking into account the dynamic properties of speech, “velocity”-type parameters (simple differences among parameters of successive frames) and “acceleration”-type parameters (differences of differences) have been recently included in acoustic front end of almost all commercialized ASR systems. The use of these temporal changes in speech spectral representation –i.e. ΔMFCC, Δ Δ MFCC– has given rise to one of the greatest improvements in ASR systems.

Moreover, in order to overcome the resolution limitation of the STFT (due to the fact that once the analysis window has been chosen, the time frequency resolution is fixed over the entire time-frequency plane, since the same window is used at all frequencies), Wavelet Transform (WT), characterized by the capability of implementing multiresolution analysis, is being used. With this processing scheme, if the analysis is viewed as a filter bank, the time resolution increases with the central frequency of the analysis filters. In other words, different analysis windows are simultaneously considered in order to more closely simulate the frequency response of the human cochlea. As with the preceding processing schemes, this auditory-based technique, even if it is surely more adequate than STFT analysis to represent a model of human auditory processing, it is still based on a mathematical framework built around a transformation of the signal, from which it tries directly to extrapolate a more realistic perceptual behaviour.

Cochlear transformations of acoustic signals result in an auditory neural firing pattern significantly different from the spectral pattern obtained from the waveform by using one of the above mentioned techniques. In other words, spectral representations such as the spectrogram, a popular time-frequency-energy representation of speech, or either the wavelet spectrogram, or scalogram, obtained using the above described multiresolution analysis technique are quite different from the true neurogram. In recent years, basilar membrane, inner cell and nerve fiber behaviour have been extensively studied by auditory physiologists and neurophysiologists and knowledge about the human auditory pathway has become more accurate. A number of studies have been accomplished and a considerable amount of data has been gathered in order to characterize the responses of nerve fibers in the eighth nerve of the mammalian auditory system using tone, tone complexes and synthetic speech stimuli. Phonetic features probably correspond in a rather straightforward manner to the neural discharge pattern with which speech is coded by the auditory nerve.

Various auditory models which try to physiologically reproduce the human auditory system have been developed in the past, and, even if they must be considered only as an approximation of physical reality, they appear to be a suitable system for identifying those aspects of the acoustic signal that are relevant for automatic speech analysis and recognition. Furthermore, with these models of auditory processing, perceptual properties can be re-discovered starting not from the sound pressure wave, but from a more internal representation which is intended to represent the true information available at the eighth acoustic nerve of the human auditory system.

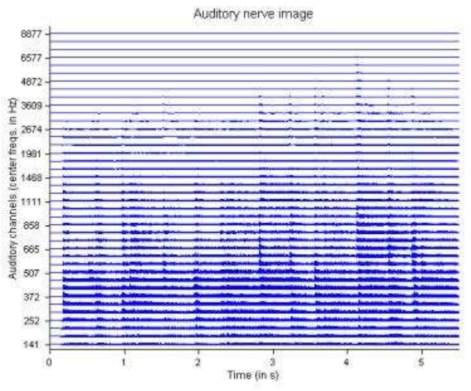
Auditory Modelling (AM) techniques not only include "perception-based" criteria instead of "production-based" ones, but also overcome "short-term" analysis limitations, because they implicitly retain dynamic and non-linear sound characteristics. For example, the dynamics of the response to non-steady-state signals, as also "forward masking" phenomena, which occur when the response to a particular sound is diminished as a consequence of a preceding, usually considerably more intense signal, are important aspects captured by efficient auditory models. Various evidences can be found in the literature suggesting the use of AM techniques, instead of the more classical ones, in building speech analysis and recognition systems. Especially when speech is greatly corrupted by noise, the effective power of AM techniques seems much more evident than that of classical digital signal processing schemes.

Auditory based cues A more complete approach to features extraction includes conventional techniques such as the short-time-Fourier-Transform (STFT or Spectrogram), and cochlear models that estimate auditory nerve firing probabilities as a function of time. The reasons for using both approaches come from different considerations. *Ecological*: the human information processing system and the musical environment are considered as a global unity in which musical content is an emerging outcome of an interactive process. *Computational*: the human auditory front-end can extract noise-robust features of the speech, whereas the recognition performances are seriously degraded with general feature extraction methods such as LPC and MFCC in noisy environments. Moreover, auditory models need huge computational load, and it is needed to make proper trade off between the performances and the computational load. For this reason, many features were extracted with typical techniques. *Purposes*: the motivation for using perception-based analysis is also due to the purposes of research; when facing the musical gestures analysis with a multi-level approach, the score representation is a drawback to the generalization of expression modelling. Also, thinking about different cultures of the world and different epoches in history, scores might not be used at all. Moreover, if a score representation is available, it still represents but weakly what musical communication really is about.

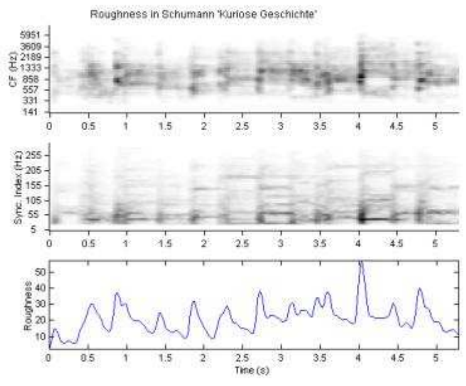
We can argue that the new tools should be developed in a way that allows a fully integrated approach, since human faculties of perception and processing of expressive communication are the both vehicle for understanding and reason for being of large social and cultural phenomenons. Expression in sound goes beyond the score knowledge, and the information derived from studies can be mapped in the physical world, embedding expression on many everyday sounding sources (i.e. for domotics, alarms design, HCI in general).

5.5.2 Auditory analysis: IPEM model

In this section the auditory model developed by Marc leman at IPEM at University of Ghent **?** will be presented and how relevant auditory features can be derived. The model is implemented as a toolbox composed by a collection of MATLAB functions for perception-based music analysis. The basic component is the Auditory Peripheral Module (APM), that takes as input a sound and gives as output the auditory primary image which is a kind of physiological justified representation of the auditory



(a)



(b)

Figure 5.20: IPEM auditory model analysis of a short excerpt (first four measures) of Schumanns Kuriose Geschichte: Auditory Nerve Image (a); Roughness : The top panel show the energy as distributed over the auditory channels, the middle panel shows the energy as distributed over the beating frequencies, the lower panel shows the roughness (b). [from Leman 2001]

information stream along the VIIIth cranial nerve. The musical signal is decomposed in different sub-bands and represented as neural patterns. The patterns are rate-codes, which means that they provide the probability of neuronal firing during a short interval of time. Auditory images, or images in short, reflect features of the sound as internal representations, in other words as brain activations so to speak. Inferences, on the other hand, provide derived information that can be compared to human behavior. Hence comes the dual validation model: images and associated processes are compared with human physiology, while inferences are compared with human behavioral responses. Figure 5.20(a) shows the auditory nerve image obtained as the result of processing a short excerpt (first four measures) of Schumanns Kuriose Geschichte.

An interesting aspect is underlined by the researchers on this auditory model. The image has two aspects: (i) its content represents features related to the musical signal, and (ii) it is assumed to be carried by an array of neurons. From the point of view of computer modelling, an image is an ordered array of numbers (= a vector) which values represent neural activation. The neural activation is expressed in terms of firing rate-code, that is, the probability of neuronal spiking during a certain time interval. A distinction will be made between different types of images (such as primary images, pitch images, spectral images, etc.).

Signal processing description The auditory peripheral module simulates the cochlear mechanical filtering using an array of overlapping band-pass filters. The basic steps can be summarized as follows.

- The outer and inner ear filtering is implemented as a second-order low-pass filter (LPF) with a resonance frequency of 4 kHz. This accounts for the overall frequency response of the ear, a coarse approximation to the Fletcher-Munson curves.

- The filtering in the cochlea is implemented by an array of band-pass filters (BPF). Forty channels are used with center frequencies ranging from 141 to 8877 Hz. The filters have a 3 dB bandwidth of one critical band; a low-frequency slope of about 10 dB per critical band unit and a high-frequency slope of about 20 dB per critical band unit.

- The mechanical to neural transduction is performed by a hair cell model (HCM), which is assumed identical in all channels. The HCM is a forward-driven gain controlled amplifier that incorporates half-wave rectification and dynamic range compression. The HCM introduces distortion products that reinforce the low frequencies that correspond to the frequency of the beats.

- A low-pass filter at 1250 Hz does an envelope extraction of the patterns in each channel. This low-pass filter accommodates for the loss of synchronization observed in the primary auditory nerve.

Features from Cochlear model From this auditory model audio cues can be derived.

Roughness The *roughness* (R) is the amplitude after a high-pass filter on the filter-bank output amplitude. Roughness is considered to be a sensory process highly related to texture perception. The estimation should be considered an inference, but the module offers more than just an inference. The calculation method of this module is based on Leman's Synchronization Index Model **?**, where roughness is defined as the energy provided by the neuronal synchronization to relevant beating frequencies in the auditory channels. This model is based on phase locking to frequencies that are present in the neural patterns. It assumes that neurons somehow extract the energy of the beating frequencies and form internal images on which the inference is based. The concept of synchronization index refers to the amount of neural activation that is synchronized to the timing of the amplitudes of the beating frequencies in the stimulus. Figure 5.20(b) shows the results of calculating roughness of the excerpt from Schumanns Kuriose Geschichte.

Loudness The *loudness* (A) extractor is based on a low-pass filter on the amplitude in each auditory filter band, and then summed over all bands: The gammatone filterbank is scaled according to one equal-loudness curve (of 50 phon). The listening level is unknown by the software, this is taken as a rough guide. The instantaneous amplitude on each channel is converted to a dB scale over a range of 70 dB. Silence is 0 dB on this scale. The instantaneous amplitude is smoothed by a first-order recursive LP filter with a cut-off frequency equal to half the framerate. The instantaneous amplitudes of all channel is summed and returned as loudness.

Centroid The computation of the cochlear filter-bank *centroid* (C) takes into account the non-linear distribution of the cochlear filter-bank: $C = \sum_i (f_i A_i) / \sum_i A_i$, where A_i and f_i are respectively the loudness and central frequency of the i-th band.

Sound Level The peak sound level $PSL = \max_i(A_i)$ and the *sound level range* $SLR = \max_i(A_i) - \min_j(A_j)$ are computed directly from the loudness profile.

5.5.3 Auditory analysis: Seneff's model

In this section a computational scheme for modelling the human auditory system will be presented. It refers essentially to the joint Synchrony/Mean-Rate (S/M-R) model of Auditory Speech Processing (ASP), proposed by S. Seneff **?**, resulting from her important studies on this matter. The overall system structure, whose block diagram is illustrated in Fig. 5.21, includes three stages: the first two deal with peripheral transformations occurring in the early stages of the hearing process while the third one attempts to extract information relevant to perception. The first two blocks represent the periphery of the auditory system. They are designed using knowledge of the rather well known responses of the corresponding human auditory stages. The third unit attempts to apply an effective processing strategy for the extraction of important speech properties like an efficient representation for locating transitions between phonemes useful for speech segmentation, or spectral lines related to formants useful for phoneme identification.

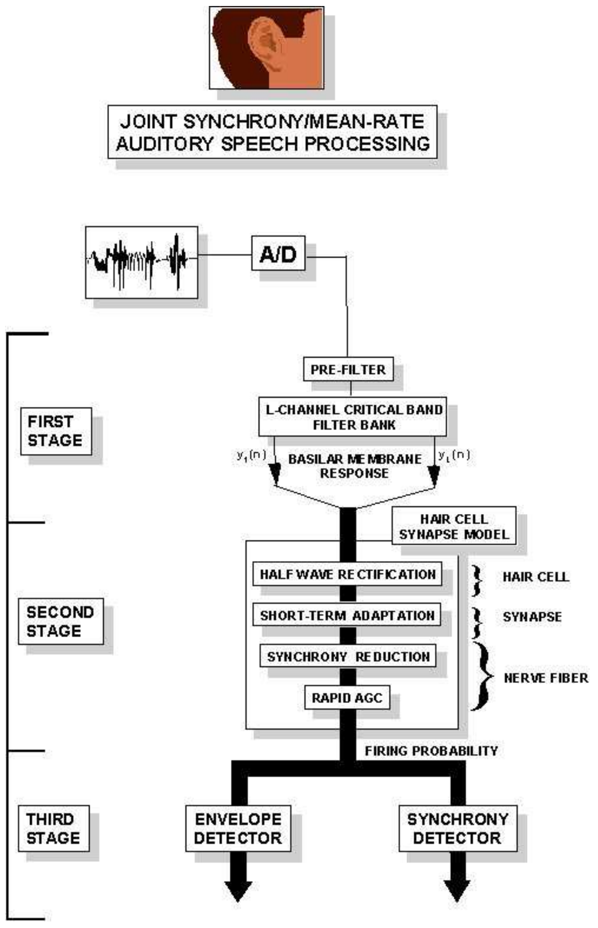
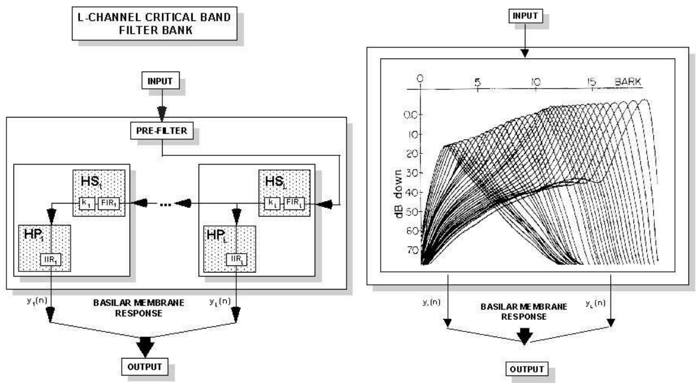


Figure 5.21: Block diagram of the joint Synchrony/Mean-Rate model of Auditory Speech Processing.



(a) (b)

Figure 5.22: Block diagram (a) and frequency response (b) of the 40-channel critical-band linear filter bank

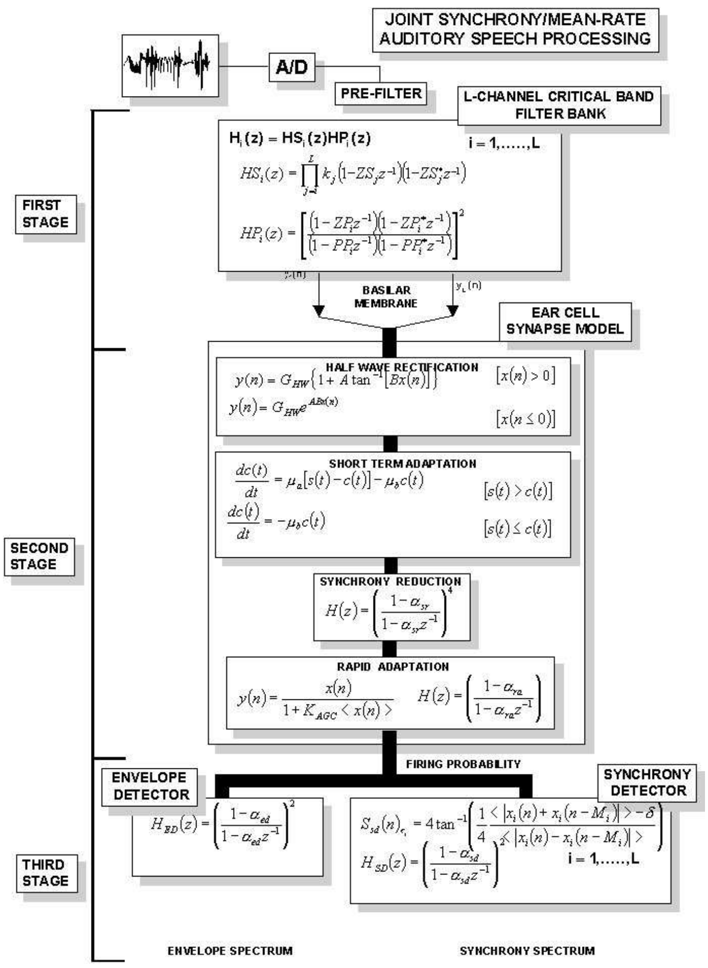

JOINT SYNCHRONY/MEAN-RATE AUDITORY SPEECH PROCESSING
A/D
PRE-FILTER
L-CHANNEL CRITICAL BAND FILTER BANK
FIRST STAGE
$H_i(z) = HS_i(z)HP_i(z)$
i = 1,.....,L
$HS_i(z) = \prod_{j=1}^{L} k_j(1-ZS_jz^{-1})(1-ZS_j^*z^{-1})$
$HP_i(z) = \left[\frac{(1-ZP_iz^{-1})(1-ZP_i^*z^{-1})}{(1-PP_iz^{-1})(1-PP_i^*z^{-1})}\right]^2$
$y_1(n)$
$y_L(n)$
BASILAR MEMBRANE
EAR CELL SYNAPSE MODEL
SECOND STAGE
HALF WAVE RECTIFICATION
$y(n) = G_{HW}\{1 + A\tan^{-1}[Bx(n)]\}$ $[x(n) > 0]$
$y(n) = G_{HW}e^{ABx(n)}$ $[x(n \le 0)]$
SHORT TERM ADAPTATION
$\frac{dc(t)}{dt} = \mu_a[s(t) - c(t)] - \mu_b c(t)$ $[s(t) > c(t)]$
$\frac{dc(t)}{dt} = -\mu_b c(t)$ $[s(t) \le c(t)]$
SYNCHRONY REDUCTION
$H(z) = \left(\frac{1-\alpha_{sr}}{1-\alpha_{sr}z^{-1}}\right)^4$
RAPID ADAPTATION
$y(n) = \frac{x(n)}{1 + K_{AGC}\langle x(n)\rangle}$
$H(z) = \left(\frac{1-\alpha_{ra}}{1-\alpha_{ra}z^{-1}}\right)$
FIRING PROBABILITY
ENVELOPE DETECTOR
SYNCHRONY DETECTOR
THIRD STAGE
$H_{ED}(z) = \left(\frac{1-\alpha_{ed}}{1-\alpha_{ed}z^{-1}}\right)^2$
$S_{sd}(n)_{c_i} = 4\tan^{-1}\left(\frac{1}{4}\frac{\langle |x_i(n) + x_i(n-M_i)|\rangle - \delta}{\langle |x_i(n) - x_i(n-M_i)|\rangle}\right)^2$
$H_{SD}(z) = \left(\frac{1-\alpha_{sd}}{1-\alpha_{sd}z^{-1}}\right)$
i = 1,.....,L
ENVELOPE SPECTRUM
SYNCHRONY SPECTRUM

Figure 5.23: Mathematical framework of the joint Synchrony/Mean-Rate model of Auditory Speech Processing.

The signal, band-limited and sampled at 16 kHz, is first pre-filtered through a set of four complex zero pairs to eliminate the very high and very low frequency components. The signal is then analyzed by the first block, a 40-channel critical-band linear filter bank. Fig. 5.22(a) shows the block diagram of the filter bank which was implemented as a cascade of complex high frequency zero pairs with taps after each zero pair to individual tuned resonators. Filter resonators consist of a double complex pole pair corresponding to the filter center frequency (CF) and a double complex zero pair at half its CF. Although a larger number of channels would provide superior spatial resolution of the cochlear output, the amount of computation time required would be increased significantly. The bandwidth of the channels is approximately 0.5 Bark, which corresponds to the width of one critical band, that is, a unit of frequency resolution and energy integration derived from psychophysical experiments. Filters, whose transfer functions are illustrated in Fig. 5.22(b), are designed in order to optimally fit physiological data. As for the mathematical implementation of the 40-channel critical-band filter bank, it is described on the top of Fig. 5.23, where serial (FIR) and parallel (IIR) branches are illustrated in detail.

The second stage of the model is called the hair cell synapse model (see Fig. 5.21). It is non-linear and is intended to capture prominent features of the transformation from basilar membrane vibration,

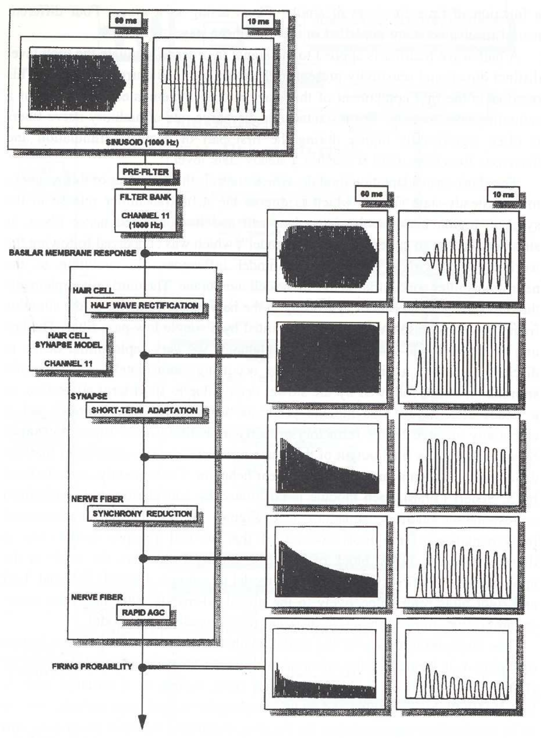


Figure 5.24: Result of the application of the four modules implementing the hair-cell synapse model to a simple 1000 Hz sinusoid. Left and right plots refer to the global 60 ms stimulus and to its corresponding first 10 ms window, in different positions along the model.

represented by the outputs of the filter bank, to probabilistic response properties of auditory nerve fibers. The outputs of this stage represent the probability of similar fibers acting as a group. Four different neural mechanisms are modelled in this non-linear stage. A half-wave rectifier is applied to the signal in order to simulate the high level distinct directional sensitivity present in the inner hair cell current response. This rectifier is the first component of this stage and is implemented by the use of a saturating non linearity. The instantaneous discharge rate of auditory-nerve fibers is often significantly higher during the first part of acoustic stimulation and decreases thereafter, until it reaches a steady-state level. The short-term adaptation module, which controls the dynamics of this response to non steady-state signals which is due to the neurotransmitter release in the synaptic region between the inner hair cell and its connected nerve fibers, is simulated by a "membrane model". This model influences the evolution of the neurotransmitter concentration inside the cell membrane. The third unit implements the observed gradual loss of synchrony in the nerve fiber behaviour as the stimulus frequency is increased, and it is implemented by a simple low-pass filter. The last unit is called Rapid Adaptation and implements the very rapid initial decay in discharge rate of auditory nerve-fibers occurring immediately after acoustic stimulation onset, followed by the slower decay, due to short-term adaptation, to a steady state level. This module performs "Automatic Gain Control" and is essentially inspired by the refractory property of auditory nerve fibers. The final output of this stage is affected by the ordering of the four different components due to their non-linear behaviour. Consequently each module is positioned by considering its hypothesized corresponding auditory apparatus (see Fig. 5.21). As for the mathematical implementation of the four modules of the hair-cell synapse model, this is illustrated in the central block of Fig. 5.23. Fig. 5.24 describes the result of the application of the model to a simple 1000 Hz sinusoid. Left and right plots refer respectively to the global 60 ms stimulus and to its corresponding first 10 ms window in different positions along the model.

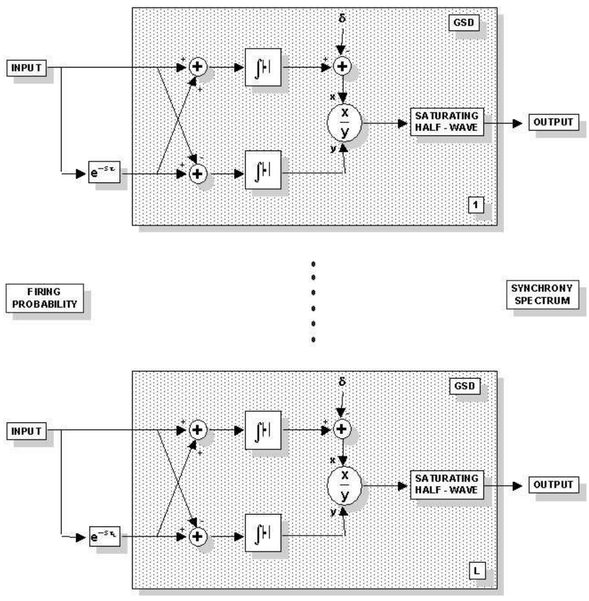
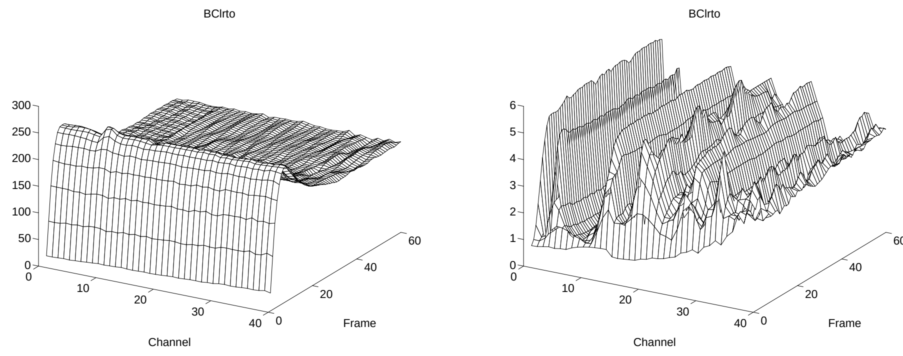


Figure 5.25: Block diagram of the Generalized Synchrony Detector (GSD) module.

The third and last stage of the model, mathematically described on the bottom of Fig. 5.23, is formed by the union of two parallel blocks: the Envelope Detector (ED), implemented by a simple low-pass filter, which by smoothing and down sampling the second stage outputs, appears to be an excellent representation for locating transitions between phonemes, thus providing an adequate basis for phonetic segmentation, and the Synchrony Detector (SD), whose block diagram as applied to each channel is shown in Figure 5.25, which implements the known "phase locking" property of the nerve fibers. This block enhances spectral peaks due to vocal tract resonances. In fact, auditory nerve fibers tend to fire in a "phase-locked" way responding to low frequency periodic stimuli, which means that the intervals between nerve fibers tend to be integral multiples of the stimulus period. Consequently, if there is a "dominant periodicity" (a prominent peak in the frequency domain) in the signal, with the so called Generalized Synchrony Detector (GSD) processing technique, only those channels whose central frequencies are closest to that periodicity will have a more prominent response.

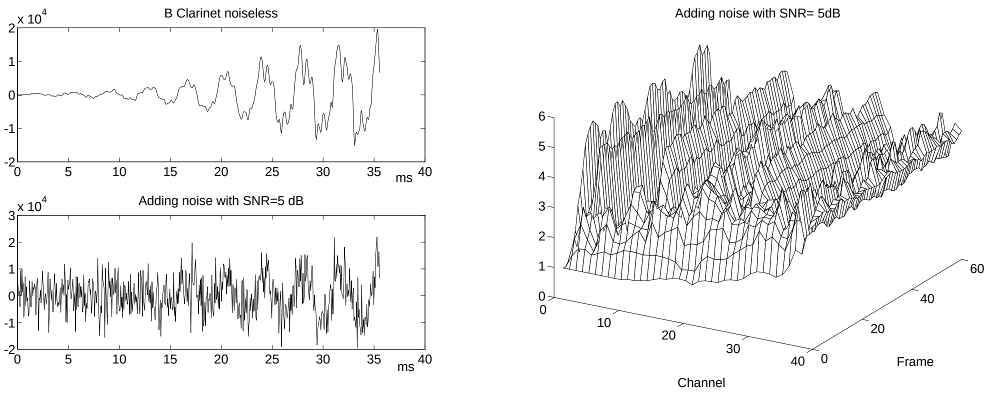
In Fig. 5.26, an example of the output of the model, as applied to a clean BClarinet sound is illustrated for the envelope (a) and the synchrony (b) detector module respectively. The use of the GSD parameters (Fig. 5.26b) allowed to produce spectra with a limited number of well defined spectral lines and this represents a good use of sound knowledge according to which harmonics are sound parameters with low variance. Due to the high level of overlapping of filter responses, envelope parameters (Fig. 5.26a) seem less important for classification purposes but maintain their usefulness in capturing very rapid changes in the signal. Thus they should be more significant considering transient sounds instead of sustained one

In order to prove the robustness of auditory parameters, the same B Clarinet sound with gaussian random noise superimposed at a level of 5 dB S/N ratio was analyzed. It is evident, from a comparison between Figures 5.26(b) and 5.27(b) that the harmonic structure is well preserved by the GSD parameters, even if the sound is greatly corrupted by quite a relevant noise. Figure 5.27(a) shows, in



(a) (b)

Figure 5.26: Output of the model, as applied to a clean B Clarinet sound: (a) envelope, (b) synchrony.



(a) (b)

Figure 5.27: (a) Time domain representation of a portion of the B Clarinet tone in (upper plot) clean and (lower plot) noisy conditions (5 db SNR). (b) Synchrony parameter output of the analysis of the same B Clarinet of Fig 5.26, superimposed with a gaussian random noise at a level of 5 db S/N ratio.

time domain, the great difference of a portion of the signal in the clean and noisy conditions.

5.6 Object description: MPEG-4

Adapted from MPEG-4 specifications

5.6.1 Scope and features of the MPEG-4 standard

The MPEG-4 standard provides a set of technologies to satisfy the needs of authors, service providers and end users alike.

- For authors, MPEG-4 enables the production of content that has far greater reusability, has greater flexibility than is possible today with individual technologies such as digital television, animated graphics, World Wide Web (WWW) pages and their extensions. Also, it is now possible to better manage and protect content owner rights.

- For network service providers MPEG-4 offers transparent information, which can be interpreted and translated into the appropriate native signaling messages of each network with the help of relevant standards bodies.
- For end users, MPEG-4 brings higher levels of interaction with content, within the limits set by the author. It also brings multimedia to new networks, including those employing relatively low bitrate, and mobile ones.

For all parties involved, MPEG seeks to avoid a multitude of proprietary, non-interworking formats and players.

MPEG-4 achieves these goals by providing standardized ways to support:

Coding representing units of aural, visual or audiovisual content, called *media objects* These media objects can be of natural or synthetic origin; this means they could be recorded with a camera or microphone, or generated with a computer;

Composition describing the composition of these objects to create compound media objects that form audiovisual scenes;

Multiplex multiplexing and synchronizing the data associated with media objects, so that they can be transported over network channels providing a QoS appropriate for the nature of the specific media objects;

Interaction interacting with the audiovisual scene generated at the receiver end or, via a back channel, at the transmitter's end.

The standard explores every possibility of the digital environment. Recorded images and sounds co-exist with their computer-generated counterparts; a new language for sound promises compact-disk quality at extremely low data rates; and the multimedia content could even adjust itself to suit the transmission rate and quality.

Possibly the greatest of the advances made by MPEG-4 is that viewers and listeners need no longer be passive. The height of "interactivity" in audiovisual systems today is the user's ability merely to stop or start a video in progress. MPEG-4 is completely different: it allows the user to interact with objects within the scene, whether they derive from so-called real sources, such as moving video, or from synthetic sources, such as computer-aided design output or computer-generated cartoons. Authors of content can give users the power to modify scenes by deleting, adding, or repositioning objects, or to alter the behavior of the objects; for example, a click on a box could set it spinning.

5.6.2 The utility of objects

At the atomic level, to use a chemical analogy, the audio and video components of MPEG-4 are known as objects. These can exist independently, or multiple ones can be grouped together to form higher-level audiovisual bonds, to coin a phrase. The grouping is called composition, and the result is an MPEG-4 scene [Fig. 1]. The strength of this so-called object-oriented approach is that the audio and video can be easily manipulated.

Visual objects in a scene are described mathematically and given a position in a two- or three-dimensional space. Similarly, audio objects are placed in a sound space. When placed in 3-D space,

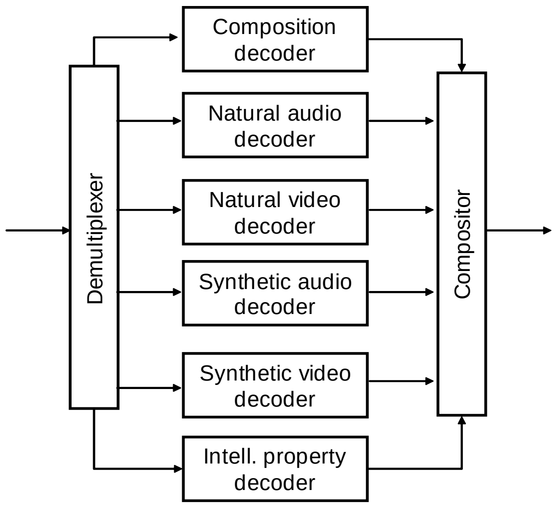


Figure 5.28: Mpeg-4 high level system architecture.

the video or audio object need only be defined once; the viewer can change his vantage point, and the calculations to update the screen and sound are done locally, at the user's terminal. This is a critical feature if the response is to be fast and the available bit-rate is limited, or when no return channel is available, as in broadcast situations.

Figure 5.28 shows a high-level diagram of an MPEG-4 system's components. It serves as a reference for the terminology used in the system's design and specification: the demultiplexer, the elementary media decoders (natural audio, natural video, synthetic audio, and synthetic video), the specialized decoders for the composition information, and the specialized decoders for the protection information,

The following sections illustrate the MPEG-4 functionalities described above, using the audiovisual scene depicted in Figure 5.29.

5.6.3 Coded representation of media objects

MPEG-4 audiovisual scenes are composed of several media objects, organized in a hierarchical fashion. At the leaves of the hierarchy, we find primitive media objects, such as:

- Still images (e.g. as a fixed background);
- Video objects (e.g. a talking person - without the background;
- Audio objects (e.g. the voice associated with that person, background music).

MPEG-4 standardizes a number of such primitive media objects, capable of representing both natural and synthetic content types, which can be either 2- or 3-dimensional. In addition to the media objects mentioned above and shown in Figure 5.29, MPEG-4 defines the coded representation of objects such as:

- Text and graphics;
- Talking synthetic heads and associated text used to synthesize the speech and animate the head; animated bodies to go with the faces;

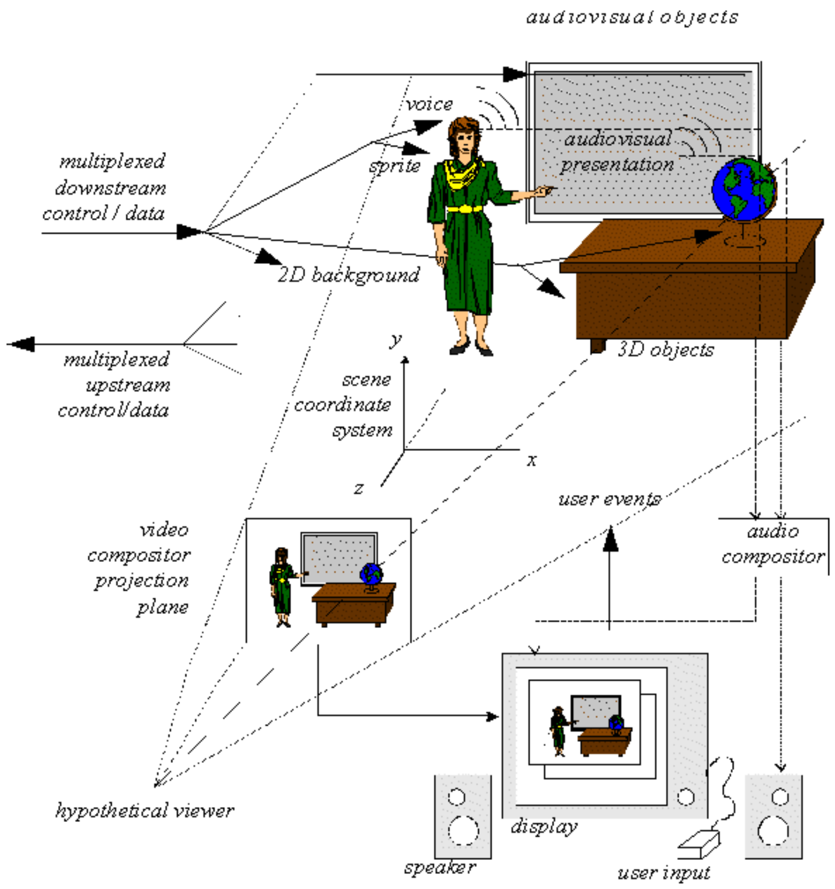


Figure 5.29: An example of an MPEG-4 scene.

- Synthetic sound.

A media object in its coded form consists of descriptive elements that allow handling the object in an audiovisual scene as well as of associated streaming data, if needed. It is important to note that in its coded form, each media object can be represented independent of its surroundings or background. The coded representation of media objects is as efficient as possible while taking into account the desired functionalities. Examples of such functionalities are error robustness, easy extraction and editing of an object, or having an object available in a scalable form.

5.6.3.1 Composition of media objects

The MPEG-4 standard deals with frames of audio and video (vectors of samples and matrices of pixels). Further, it deals with the objects that make up the audiovisual scene. Thus, a given scene has a number of video objects, of possibly differing shapes, plus a number of audio objects, possibly associated to video objects, to he combined before presentation to the user. Composition encompasses the task of combining all of the separate entities that make up the scene.

Figure 5.29 explains the way in which an audiovisual scene in MPEG-4 is described as composed of individual objects. The figure contains compound media objects that group primitive media objects together. Primitive media objects correspond to leaves in the descriptive tree while compound media objects encompass entire sub-trees. As an example: the visual object corresponding to the talking person and the corresponding voice are tied together to form a new compound media object, containing both the aural and visual components of that talking person.

Such grouping allows authors to construct complex scenes, and enables consumers to manipulate meaningful (sets of) objects.

More generally, MPEG-4 provides a standardized way to describe a scene, allowing for example to:

- Place media objects anywhere in a given coordinate system;
- Apply transforms to change the geometrical or acoustical appearance of a media object;
- Group primitive media objects in order to form compound media objects;
- Apply streamed data to media objects, in order to modify their attributes (e.g. a sound, a moving texture belonging to an object; animation parameters driving a synthetic face);
- Change, interactively, the user is viewing and listening points anywhere in the scene.

Composition information consists of the representation of the hierarchical structure of the scene. A graph describes the relationship among elementary media objects comprising the scene. The scene description builds on several concepts from the Virtual Reality Modeling language (VRML) in terms of both its structure and the functionality of object composition nodes and extends it to fully enable the aforementioned features.

The resulting specification addresses issues specific to an MPEG-4 system:

- description of objects representing natural audio and video with streams attached,
- description of objects representing synthetic audio and video (2D) and 3D material) with streams attached (such as streaming text or streaming parameters for animation of a facial model).

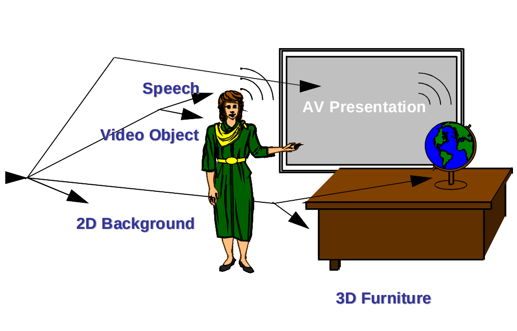
The scene description represents complex scenes populated by synthetic and natural audiovisual objects with their associated spatiotemporal transformations.

MPEG-4's language for describing and dynamically changing the scene is named the Binary Format for Scenes (BIFS). BIFS commands are available not only to add objects to or delete them from the scene, but also to change visual or acoustic properties of an object without changing the object in itself; thus the color alone of a 3-D sphere might be varied.

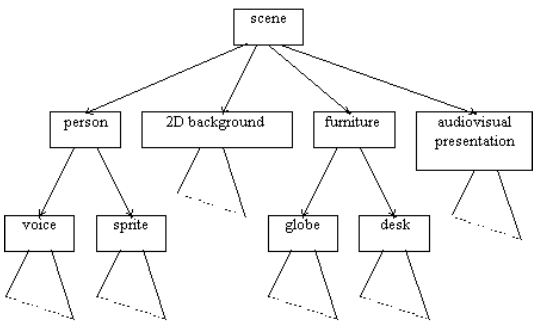
BIFS can be used to animate objects just by sending a BIFS command and to define their behavior in response to user input at the decoder. Again, this is a nice way to build interactive applications. In principle, BIFS could even be used to put an application screen (such as a Web browser's) as a "texture" in the scene.

BIFS borrows many concepts from the Virtual Reality Modeling Language (VRML), which is the method used most widely on the Internet to describe 3-D objects and users' interaction with them. BIFS and VRML can be seen as different representations of the same data. In VRML, the objects and their actions are described in text, as in any other high-level language. But BIFS code is binary, and thus is shorter for the same content–typically 10 to 15 times. More important, unlike VRML, MPEG-4 uses BIFS for real-time streaming, that is, a scene does not need to be downloaded in full before it can be played, but can be built up on the fly.

The author can generate this description in textual format, possibly through an authoring tool. The scene description then conforms to the VRML syntax with extensions. For efficiency, the standard



(a)



(b)

Figure 5.30: Objects in a scene (a) and the corresponding BInary Format for Scene (BIFS) representation (b).

defines a way to encode the scene description in a binary representation–Binary Format for Scene Description (BIFS). Multimedia scenes are conceived as hierarchical structures represented as a graph. Each leaf of the graph represents a media object (audio; video; synthetic audio like a Musical Instrument Digital Interface, or MIDI, stream; synthetic video like a face model). The graph structure isn't necessarily static, as the relationships can evolve over time as nodes or subgraphs are added or deleted, All the parameters describing these relationships are part of the scene description sent to the decoder.

The initial snapshot of the scene is sent or retrieved on a dedicated stream. It is then parsed, and the whole scene structure is reconstructed (in an internal representation) at the receiver terminal. All the nodes and graph leaves that require streaming support to retrieve media contents or ancillary data (video stream, audio stream, facial animation parameters) are logically connected to the decoding pipelines.

An update of the scene structure may be sent at any time. These updates can access any field of any updatable node in the scene. An updatable node is one that received a unique node identifier in the scene structure. The user can also interact locally with the scenes, which may change the scene structure or the value of any field of any updatable node.

Composition information (information about the initial scene composition and the scene updates during the sequence evolution) is, like other streaming data, delivered in one elementary stream. The composition stream is treated differently from others because it provides the information required by the terminal to set up the scene structure and map all other elementary streams to the respective media objects.

How objects are grouped together An MPEG-4 scene follows a hierarchical structure, which can be represented as a directed acyclic graph. Each node of the graph is a media object, as illustrated in Figure 5.31(b) (note that this tree refers back to Figure 5.29 and Figure 5.31(a)). The tree structure is not necessarily static; node attributes (e.g., positioning parameters) can be changed while nodes can be added, replaced, or removed.

How objects are positioned in space and time: In the MPEG-4 model, audiovisual objects have both a spatial and a temporal extent. Each media object has a local coordinate system. A local coordinate system for an object is one in which the object has a fixed spatio-temporal location and

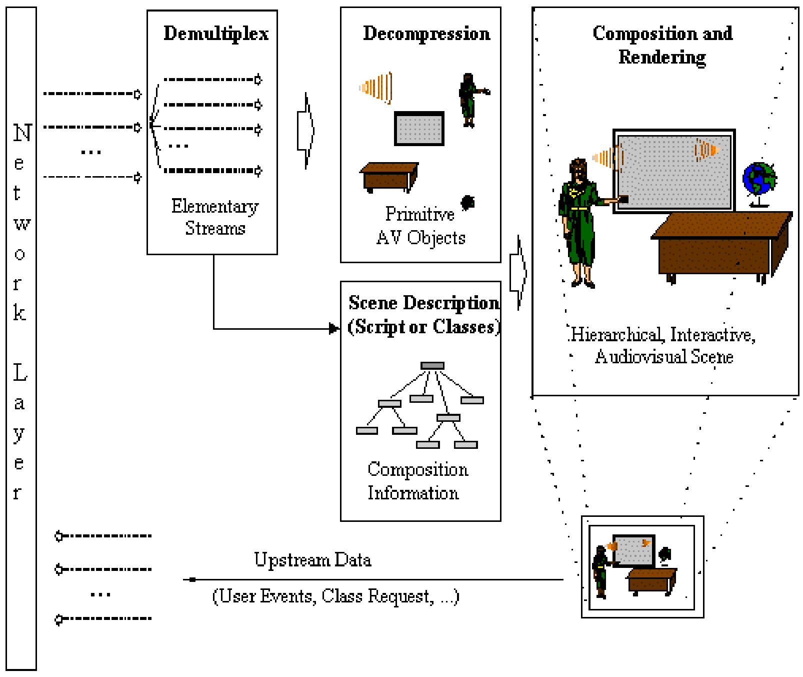

Network Layer
Demultiplex
Elementary Streams
Decompression
Primitive AV Objects
Composition and Rendering
Hierarchical, Interactive, Audiovisual Scene
Scene Description (Script or Classes)
Composition Information
Upstream Data
(User Events, Class Request, ...)

Figure 5.31: Major components of an MPEG-4 terminal (receiver side).

scale. The local coordinate system serves as a handle for manipulating the media object in space and time. Media objects are positioned in a scene by specifying a coordinate transformation from the object local coordinate system into a global coordinate system defined by one more parent scene description nodes in the tree.

Wrapping the data Just as MPEG-4's representation of multimedia content is new and versatile, so is its scheme for preparing that content for transportation or storage (and, logically, for decoding) [Fig 2]. Here, objects are placed in so-called elementary streams (ESs). Some objects, such a sound track or a video, will have a single such stream. Others objects may have two or more. For instance, a scalable object would have an ES for basic-quality information plus one or more enhancement layers, each of which would have its own ES for improved quality, such as video with finer detail or faster motion.

Higher-level data describing the scene–the BIFS data defining, updating and positioning the media objects–is conveyed in its own ES. Here again the virtue of the hierarchical, object-based conceptions in MPEG-4 can be seen: it is easier to reuse objects in the production of new multimedia content, or (to say it another way) the new production is easier to modify without changing an encoded object itself. If parts of a scene are to be delivered only under certain conditions, say, when it is determined that enough bandwidth is available, multiple scene description ESs for the different circumstances may be used to describe the same scene.

To inform the system which elementary streams belong to a certain object, MPEG-4 uses the novel, critical concept of an object descriptor (OD). Object descriptors in their turn contain elementary stream descriptors (ESDs) to tell the system what decoders are needed to decode a stream. With another field, optional textual information about the object can be supplied. Object descriptors are sent in their own, special elementary stream, which allows them to be added or deleted dynamically as the scene changes.

The play-out of the multiple MPEG-4 objects is coordinated at a layer devoted solely to synchronization. Here, elementary streams are split into packets, and timing information is added to the payload of these packets. These packets are then ready to be passed on to the transport layer.

Streams Timing information for the decoder consists of the speed of the encoder clock and the time stamps of the incoming streams, which are relative to that clock. Two kinds of time stamps exist: one says when a piece of information must be decoded, the other says when the information must be ready for presentation.

The distinction between the types of stamp is important. In many video compression schemes, some frames are calculated as an interpolation between previous and following frames. Thus, before such a frame can be decoded and presented, the one after it must be decoded (and held in a buffer). For predictable decoder behavior, a buffer model in the standard augments the timing specification.

In terms of the ISO seven-layer communications model, no specific transport mechanism is defined in MPEG-4. Existing transport formats and their multiplex formats suffice, including the MPEG-2 transport stream, asynchronous transfer mode (ATM), and real-time transport protocol (RTP) on the Internet. Incidentally, the fact that the MPEG-2 transport stream is used by digital TV has the important consequence of allowing co-broadcast modes.

A separate transport channel could be set up for each data stream, but there can be many of these for a single MPEG-4 scene, and as a result the process could be unwieldy and waste bits. To remedy matters, a small tool in MPEG-4, FlexMux, was designed to act as an intermediate step to any suitable form of transport. In addition, another interface defined in MPEG-4 lets the application ask for connections with a certain quality of service, in terms of parameters like bandwidth, error rate, or delay.

From the application's point of view, this interface is the same for broadcast channels, interactive sessions, and local storage media. Application designers can therefore write their code without having to worry about the underlying delivery mechanisms. Further, the next release of the standard will allow differing channels to be used at either end of a transmission/receive network, say, an Internet protocol channel on one end and an ATM one on the other.

Another important addition in Version 2 is a file format known as mp4, which can be used for exchange of content and which is easily converted. MPEG-1 and MPEG-2 did not include such a specification, but the intended use of MPEG-4 in Internet and personal computer environments makes it a necessity. It will be the only reliable way for users to exchange complete files of MPEG-4 content

5.6.3.2 Description and synchronization of streaming data for media objects

Media objects may need streaming data, which is conveyed in one or more elementary streams. An object descriptor identifies all streams associated to one media object. This allows handling hierarchically encoded data as well as the association of meta-information about the content (called object content information) and the intellectual property rights associated with it.

Each stream itself is characterized by a set of descriptors for configuration information, e.g., to determine the required decoder resources and the precision of encoded timing information. Furthermore the descriptors may carry hints to the Quality of Service (QoS) it requests for transmission (e.g., maximum bit rate, bit error rate, priority, etc.)

Synchronization of elementary streams is achieved through time stamping of individual access units within elementary streams. The synchronization layer manages the identification of such access units and the time stamping. Independent of the media type, this layer allows identification of the type of access unit (e.g., video or audio frames, scene description commands) in elementary streams, recovery of the media objector scene description time base, and it enables synchronization among them. The syntax of this layer is configurable in a large number of ways, allowing use in a broad spectrum of systems.

5.6.4 MPEG-4 visual objects

Classical, "rectangular" video, as the type that comes from a camera may be called, is of course one of the visual objects defined in the standard. In addition, objects with arbitrary shapes can be encoded apart from their background and then placed before other video types.

In fact, MPEG-4 includes two ways of describing arbitrary shapes, each appropriate to a different environment. In the first, known as binary shape, an encoded pixel (of a certain color, brightness, and so on) either is or is not part of the object in question. A simple but fairly crude technique, it is useful in low bit-rate environments, but can be annoying–the edges of pixels are sometimes visible, and curves have little jagged steps, known as aliasing or "the jaggies."

For higher-quality content, a shape description known as gray scale, or alpha shape, is used. Here, each pixel belonging to a shape is not merely on or off, but is assigned a value for its transparency. With this additional feature, transparency can differ from pixel to pixel of an object, and objects can be smoothly blended, either into a background or with other visual objects.

One instance of smooth blending can be seen in most television weather reports. The weatherman's image seems to be standing in front of a map, which in fact is generated elsewhere. Not surprisingly, then, manufacturers of television studio equipment have expressed an interest in the capabilities for arbitrary shape objects and scene description since, conceptually, they closely match the way things are already done in a studio. In fact, MPEG video has started working on bit-rates and quality levels well beyond the high-definition television (HDTV) level that can already be achieved.

Note that MPEG does not specify how shapes are to be extracted. Doing this automatically–video segmentation, as it is known–is still a matter of intensive research. Current methods still have limitations but there are ways to get the job done: the best way of obtaining shaped objects such as the weatherman is recording them with a blue or green background, colors that can easily be filtered out.

Actually, MPEG-4, like its predecessors, specifies only the decoding process. Encoding processes, including any improvements, are left to the marketplace. Even today, improvements in MPEG-2 compression quality sometimes show up, even though that standard was cast in stone several years ago.

On the other end of the bit-rate spectrum, a great deal of effort has gone into making moving video possible at very low bit-rates, notably for mobile devices. MPEG-4 has been found usable for streaming wireless video transmission (making use of GSM/Global System for Mobile Communications) at 10 kb/s–the data rate in GSM currently used for voice communications.

5.6.5 MPEG-4 audio

MPEG-4 coding of audio objects provides tools for both representing natural sounds (such as speech and music) and for synthesizing sounds based on structured descriptions. The representation for synthesized sound can be derived from text data or so-called instrument descriptions and by coding parameters to provide effects, such as reverberation and spatialization. The representations provide compression and other functionalities, such as scalability and effects processing.

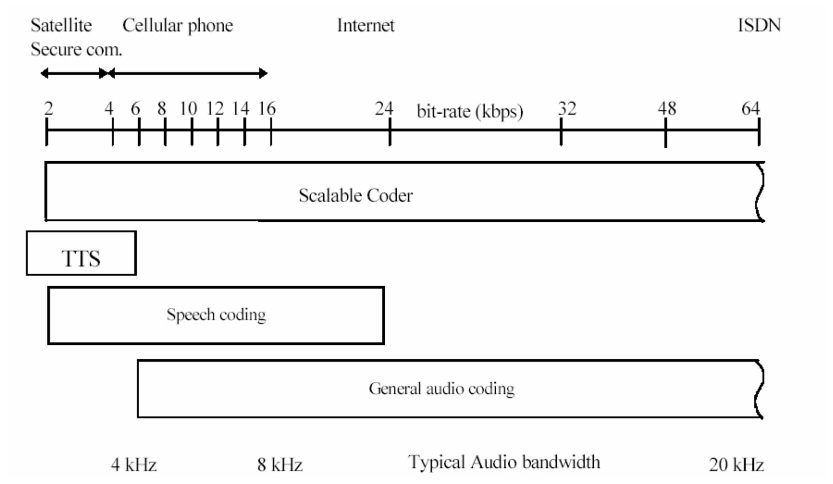


Figure 5.32: Major components of an MPEG-4 terminal (receiver side).

To achieve the highest audio quality within the full range of hit rates and at the same time provide extra functionality's, the MPEG-4 Audio standard includes six types of coding techniques:

- Parametric coding modules
- Linear predictive coding (LPC) modules
- Time/frequency (T/F) coding modules
- Synthetic/natural hybrid coding (SNHC) integration modules
- Text-to-speech (US) integration modules
- Main integration modules, which combine the first three modules to a scalable encoder

While the first three parts describe real coding schemes for the low-bit-rate representation of natural audio sources, the SNHC and TTS parts only standardize the interfaces to general SNHC and TTS systems. Because of this, already established synthetic coding standards, such as MIDI, can he integrated into the MPEG-4 Audio system. The TTS interfaces permit plugging TTS modules optimized for a special language into the general framework. The main module contains global modules, such as the speed change functionality of MPEG-4 and the scalability add-ons necessary to implement large-step scalability by combining different coding schemes. To allow optimum coverage of the bitrates and to allow for bitrate and bandwidth scalability, a general framework has been defined. This is illustrated in Figure 5.32.

5.6.5.1 Natural audio

Starting with a coder operating at a low bitrate, by adding enhancements to a general audio coder, both the coding quality as well as the audio bandwidth can be improved.

Bitrate scalability, often also referred to as embedded coding, allows a bitstream to be parsed into a bitstream of lower bitrate that can still be decoded into a meaningful signal. The bitstream parsing can occur either during transmission or in the decoder. Bandwidth scalability is a particular case of bitrate scalability whereby part of a bitstream representing a part of the frequency spectrum can be discarded during transmission or decoding.

Encoder complexity scalability allows encoders of different complexity to generate valid and meaningful bitstreams. The decoder complexity scalability allows a given bitstream to be decoded by decoders of different levels of complexity. The audio quality, in general, is related to the complexity of the encoder and decoder used.

The MPEG-4 systems layer allows codecs according to existing (MPEG) standards, e.g. MPEG-2 AAC, to be used. Each of the MPEG-4 coders is designed to operate in a stand-alone mode with its own bitstream syntax. Additional functionalities are realized both within individual coders, and by means of additional tools around the coders. An example of such a functionality within an individual coder is speed or pitch change within HVXC.

5.6.5.2 Synthesized audio

MPEG-4 defines decoders for generating sound based on several kinds of ?structured? inputs. Text input is converted to speech in the Text-To-Speech (TTS) decoder, while more general sounds including music may be normatively synthesized. Synthetic music may be delivered at extremely low bitrates while still describing an exact sound signal.

Text To Speech. TTS coders bitrates range from 200 bit/s to 1.2 Kbit/s, which allows a text or a text with prosodic parameters (pitch contour, phoneme duration, and so on) as its inputs to generate intelligible synthetic speech. It supports the generation of parameters that can be used to allow synchronization to associated face animation, international languages for text and international symbols for phonemes. Additional markups are used to convey control information within texts, which is forwarded to other components in synchronization with the synthesized text. Note that MPEG-4 provides a standardized interface for the operation of a Text To Speech coder (TTSI = Text To Speech Interface), but not a normative TTS synthesizer itself.

An itemized overview:

- Speech synthesis using the prosody of the original speech
- Lip synchronization control with phoneme information.
- Trick mode functionality: pause, resume, jump forward/backward.
- International language and dialect support for text. (i.e. it can be signaled in the bitstream which language and dialect should be used)
- International symbol support for phonemes.
- support for specifying age, gender, speech rate of the speaker support for conveying facial animation parameter(FAP) bookmarks.

Score Driven Synthesis: Structured Audio The Structured Audio tools decode input data and produce output sounds. This decoding is driven by a special synthesis language called SAOL (Structured Audio Orchestra Language) standardized as a part of MPEG-4. This language is used to define an orchestra made up of instruments (downloaded in the bitstream, not fixed in the terminal) which create and process control data. An instrument is a small network of signal processing primitives that might emulate some specific sounds such as those of a natural acoustic instrument. The signal-processing

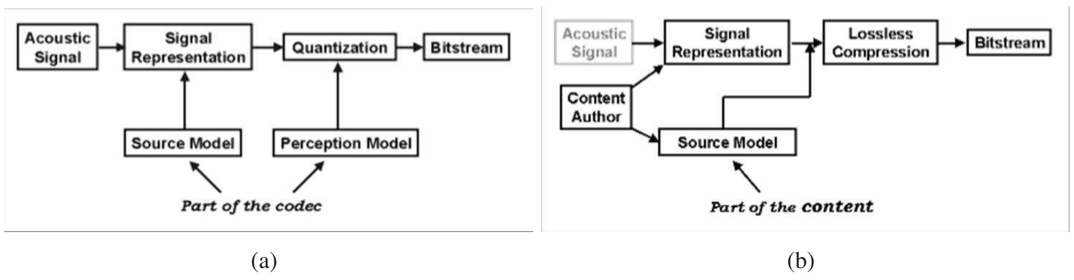


Figure 5.33: In a traditional audio coder, the source model and perception model are defined outside of the transmission (for example, in a standards document). The codec designers do the best job they can at designing these models, but then they are fixed for all content (a). In Structured Audio, the source model is part of the content. It is transmitted in the bitstream and used to give different semantics to the signal representation for each piece of content. There can be a different source model, or multiple source models, for each different piece of content (b).

network may be implemented in hardware or software and include both generation and processing of sounds and manipulation of pre-stored sounds.

MPEG-4 does not standardize a single method of synthesis, but rather a way to describe methods of synthesis. Any current or future sound-synthesis method can be described in SAOL, including wavetable, FM, additive, physical-modeling, and granular synthesis, as well as non-parametric hybrids of these methods.

Control of the synthesis is accomplished by downloading scores or scripts in the bitstream. A score is a time-sequenced set of commands that invokes various instruments at specific times to contribute their output to an overall music performance or generation of sound effects. The score description, downloaded in a language called SASL (Structured Audio Score Language), can be used to create new sounds, and also include additional control information for modifying existing sound. This allows the composer finer control over the final synthesized sound. For synthesis processes that do not require such fine control, the established MIDI protocol may also be used to control the orchestra.

Careful control in conjunction with customized instrument definition, allows the generation of sounds ranging from simple audio effects, such as footsteps or door closures, to the simulation of natural sounds such as rainfall or music played on conventional instruments to fully synthetic sounds for complex audio effects or futuristic music.

An important result of the description framework is that the synthetic audio fed to each terminal is identical. Thus, barring the vagaries of physical equipment that one user might have compared to another, the output is guaranteed to sound the same from terminal to terminal.

For terminals with less functionality, and for applications which do not require such sophisticated synthesis, a wavetable bank format is also standardized. Using this format, sound samples for use in wavetable synthesis may be downloaded, as well as simple processing, such as filters, reverbs, and chorus effects. In this case, the computational complexity of the required decoding process may be exactly determined from inspection of the bitstream, which is not possible when using SAOL.

Structured audio and traditional coding Structured audio coding differs from traditional audio coding in that the sound model is not fixed in the protocol (Fig. 5.33a), but dynamically described as part of the transmission stream, where it may vary from signal to signal (Fig. 5.33b). That is, where a traditional audio coder makes use of a fixed model such as a vocal-tract approximation (for LPC coding) or a psychoacoustic making model (for wideband techniques such as MPEG-AAC or Dolby AC-3), a structured audio coder transmits sound in two parts: a description of a model and a set of parameters making use of that model.

The fact that we have great flexibility to encode different sound models in the bitstream means that, in theory, SA coding can subsume all other audio coding techniques. For example, if we wish to transmit speech in CELP-coded format, but only have the SA decoding system available, we can still use CELP: we write the CELP-decoder in SAOL, transmit it in the bitstream header, and then send frames of data optimized for that CELP model as the bitstream data. This bitstream will be nearly the same size as the CELP bitstream; it only requires a fixed constant-size data block to transmit the orchestra containing the decoder, and then the rest of the bitstream is the same size.

SAOL example SAOL is a "C-like" language. The syntactic framework of SAOL is familiar to anyone who programs in C, although the fundamental elements of the language are still signal variables, unit generators, instruments, and so forth, as in other synthesis languages. The program below shows a simple SAOL instrument that creates a simple beep by applying an envelope to the output of a single sinusoidal oscillator.

```
// This is a simple SAOL instrument that makes a short tone,
// using an oscillator over a stored function table.

instr tone(pitch,amp) {
  table wave(harm,2048,1); // sinusoidal wave function
  asig sound;              // 'asig' denotes audio signal
  ksig env;                // 'ksig' denotes control signal

  env = kline(0,0.1,1,dur-0.1,0); //make envelope
  sound = oscil(wave, pitch) * amp * env;
                 // create sound by enveloping an oscillator
  output(sound); // play that sound
}
```

A number of features are immediately apparent in this instrument. The instrument name (`tone`), parameters (or "p-fields": `pitch` and `amp`), stored-function table (`wave`), and table generator (`harm`) all have names rather than numbers. All of the signal variables (`sound` and `env`) are explicitly declared with their rates (`asig` for audio rate and `ksig` for control rate), rather than being automatically assigned rates based on their names. There is a fully recursive expression grammar, so that unit generators like `kline` and `oscil` may be freely combined with arithmetic operators. The stored-function tables may be encapsulated in instruments or in the orchestra when this is desirable; they may also be provided in the score, in the manner of Music V (Csound also allows both options). The unit generators `kline` and `oscil` are built into the language; so is the wavetable generator `harm`.

The control signal `dur` is a standard name, which is a variable automatically declared in every instrument, with semantics given in the standard. There is a set of about 20 standard names defined in SAOL; `dur` always contains the duration of the note that invoked the instrument.

This SASL file plays a melody on `tone`:

```
0.5  tone 0.75 164.6
1.5  tone 0.75 329.6
2.5  tone 0.5  311.1
3    tone 0.25 246.9
3.25 tone 0.25 277.2
```

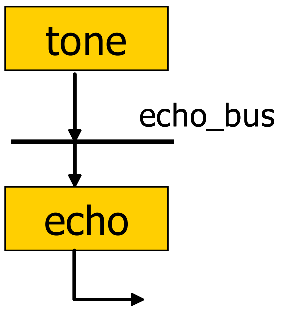


Figure 5.34: In SAOL, a metaphor of bus routing is employed that allows the concise description of complex networks. The output of the `tone` instrument is placed on the bus called `echo_bus`; this bus is sent to the instrument called `echo` for further processing.

```
3.5   tone 0.5   311.1
4     tone 0.5   329.6
5     end
```

An aspect of modularity in SAOL involves its flow-of-control processing model. In SAOL, a metaphor of bus routing is employed that allows the concise description of complex networks. Its use is shown in the following example:

```
// This is a complete SAOL orchestra that demonstrates the
// use of buses and routing in order to do effects processing.
// The output of the 'tone' instrument is placed on the bus
// called 'echo_bus'; this bus is sent to the instrument called
// 'echo' for further processing.

global {
  srate 32000; krate 500;

  send(echo; 0.2; echo_bus};
  // use 'echo' to process the bus 'echo_bus'
  route(echo_bus, beep);
  // put the output of 'beep' on 'echo_bus'
}

instr tone(pitch, amp) {
  // as above
}

instr echo(dtime) {
  // a simple digital-delay echo. 'dtime' is the
  // cycle time.
  asig x;

  x = delay(x/2 + input[0],dtime);
  output(x);
}
```

In this orchestra, a global block is used to describe global parameters and control. The `srate` and `krate` tags specify the sampling rate and control (LFO) rate of the orchestra. The send instruction creates a new bus called `echo_bus`, and specifies that this bus is sent to the effects processing instrument called echo. The `route` instruction specifies that the samples produced by the instrument beep are not turned directly into sound output, but instead are "routed onto" the bus `echo_bus` for further processing (see Fig. 5.34).

The instrument `echo` implements a simple exponentially decaying digital-echo sound using the `delay` core opcode. The `dtime` p-field specifies the cycle time of the digital delay. Like `dur` in Figure 1, `input` is a standard name; input always contains the values of the input to the instrument, which in this case is the contents of the bus `echo_bus`. Note that echo is not a user-defined opcode that implements a new unit generator, but an effects-processing instrument.

This bus-routing model is modular with regard to the instruments `tone` and `echo`. The `tone` sound-generation instrument does not "know" that its sound will be modified, and the instrument itself does not have to be modified to enable this. Similarly, the `echo` instrument does not "know" that its input is coming from the beep instrument; it is easy to add other sounds to this bus without modification to `echo`. The bus-routing mechanism in SAOL allows easy reusability of effects-processing algorithms. There are also facilities that allow instruments to manipulate busses directly, if such modularity is not desirable in a particular composition.

5.6.5.3 Sound spatialization

Although less self-evident than with images, audio is also represented in the form of objects. An audio object can be a monaural speech channel or a multichannel, high-quality sound object. The composition process is in fact far more strictly prescribed for audio than for video. With the audio available as objects in the scene graph, different mixes from input channels (objects) to output channels (speakers) can be defined for different listening situations.

Another advantage of having audio as objects is that they then can have effects selectively applied to them. For example, if a soundtrack includes one object for speech and one for background audio, an artificial reverberation can be applied to the speech as distinct from the background music. If a user moves a video object in the scene, the audio can move along with it, and the user could also change how audio objects are mixed and combined.

Like video objects, audio objects may be given a location in a 3-D sound space, by instructing the terminal to spatially position sounds at certain spots. This is useful in an audio conference with many people, or in interactive applications where images as well as audio are manipulated.

A related feature known as environmental spatialization will be included in MPEG-4 Version 2. This feature can make how a sound object is heard depend on the room definition sent to the decoder, while the sound object itself need not be touched. In other words, the spatializations work locally, at the terminal, so again virtually no bit-transmission overhead is incurred.

Imagine spatialization when a person walks through a virtual house: when a new room of different shape and size is entered, the sound of the voice object changes accordingly without the object itself having to be changed.

5.6.5.4 Audio BIFS

The AudioBIFS system [part of the MPEG-4 Binary Format for Scene Description (BIFS)] allows multiple sounds to be transmitted using different coders and then mixed, equalized, and post-processed once they are decoded. This format is structurally based on the Virtual Reality Modeling Language (VRML) 2.0 syntax for scene description, but contains more powerful sound-description features.

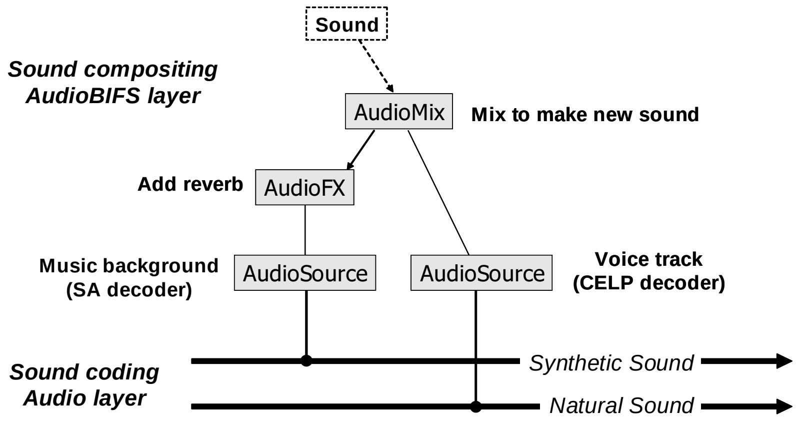


Figure 5.35: Two sound streams are processed and mixed using the AudioBIFS scene graph. A musical background is transmitted with the MPEG-4 Structured Audio system and reverb is added with an AudioFX node. Then the result is mixed with a speech sound transmitted with MPEG-4 CELP.

Using MPEG-4 AudioBIFS, each part of a soundtrack may be coded in the format that best suits it. For example, suppose that the transmission of voice over with background music is desired in the style of a radio advertisement. It is difficult to describe high-quality spoken voice with sound-synthesis techniques, and so Structured Audio alone cannot be used; but speech coders are not adequate to code speech with background music, and so MPEG-4 CELP alone cannot be used. In MPEG-4 with AudioBIFS, the speech is coded using the CELP coder, and the background music is coded in Structured Audio. Then, at the decoding terminal, the two "streams" of sound are decoded individually and mixed together. The AudioBIFS part of the MPEG-4 standard describes the synchronization provided by this mixing process.

AudioBIFS is built from a set of nodes that link together into a tree structure, or scene graph. Each of these nodes represents a signal-processing manipulation on one or more audio streams. In this, AudioBIFS is somewhat itself like a sound-process-ing language, but it is much simpler (there are only seven types of nodes, and only "filters," no "generators"). The scene-graph structure is used because it is a familiar and tested mechanism for computer-graphics description, and AudioBIFS is only a small part of the overall BIFS framework. The functions performed by AudioBIFS nodes allow sounds to be switched (as in a multiple-language soundtrack), mixed, delayed, "clipped" for interactive presentation, and gain-controlled.

More-advanced effects are possible by embedding SAOL code into the AudioBIFS scene graph with the AudioFX node. Using AudioFX, any audio effect may be described in SAOL and applied to the output of a natural or synthetic audio decoding process (see Figure 5.35).

For example, if we want to transmit speech with reverberated background music, we code the speech with MPEG-4 CELP. As above, we code the background music using MPEG-4 Structured Audio and provide an AudioFX node containing SAOL code that implements the desired reverberator. When the transmission is received, the synthetic music stream is decoded and the reverberation processing is performed; the result is added to the decoded speech stream. Only the resulting sound is played back to the listener. Just as MPEG-4 Structured Audio allows exact, terminal-independent control of sound synthesis for the composer, MPEG-4 AudioBIFS allows exact, terminal-independent control of audio-effects processing for the sound designer and producer.

The combination of synthetic and natural sound in the same sound scene with downloaded mixing and effects processing is termed synthetic/natural hybrid coding, or SNHC audio coding, in MPEG-4. Other features of AudioBIFS include 3-D audio spatialization for sound presentation in virtual-reality applications, and the creation of sound scenes that render differently on different terminals, depending (for example) on sampling rate, speaker configuration, or listening conditions.

Finally, the AudioBIFS component of MPEG-4 is part of the overall BIFS system, which provides sophisticated functionality for visual presentation of streaming video, 3-D graphics, virtual-reality scenes, and the handling of interactive events. The audio objects described with AudioBIFS and the various audio decoders may be synchronized with video or computer-graphics objects, and altered in response to user interaction.

5.7 Multimedia Content Description: Mpeg-7

Adapted from MPEG-7 specifications

5.7.1 Introduction

The MPEG-7 standard also known as “Multimedia Content Description Interface” aims at providing standardized core technologies allowing description of audiovisual data content in multimedia environments. It supports some degree of interpretation of the informations meaning, which can be passed onto, or accessed by, a device or a computer code. MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardises shall support as broad a range of applications as possible.

Accessing audio and video used to be a simple matter - simple because of the simplicity of the access mechanisms and because of the poverty of the sources. An incommensurable amount of audiovisual information is becoming available in digital form, in digital archives, on the World Wide Web, in broadcast data streams and in personal and professional databases, and this amount is only growing. The value of information often depends on how easy it can be found, retrieved, accessed and filtered and managed.

The transition between two millennia abounds with new ways to produce, offer, filter, search, and manage digitized multimedia information. Broadband is being offered with increasing audio and video quality and speed of access. The trend is clear. In the next few years, users will be confronted with such a large number of contents provided by multiple sources that efficient and accurate access to this almost infinite amount of content will seem to be unimaginable. In spite of the fact that users have increasing access to these resources, identifying and managing them efficiently is becoming more difficult, because of the sheer volume. This applies to professional as well as end users. The question of identifying and managing content is not just restricted to database retrieval applications such as digital libraries, but extends to areas like broadcast channel selection, multimedia editing, and multimedia directory services.

This challenging situation demands a timely solution to the problem. MPEG-7 is the answer to this need.

5.7.1.1 Context of MPEG-7

Audiovisual information plays an important role in our society, be it recorded in such media as film or magnetic tape or originating, in real time, from some audio or visual sensors and be it analogue or, increasingly, digital. Everyday, more and more audiovisual information is available from many sources around the world and represented in various forms (modalities) of media, such as still pictures, graphics, 3D models, audio, speech, video, and various formats. While audio and visual information used to be consumed directly by the human being, there is an increasing number of cases where the audiovisual information is created, exchanged, retrieved, and re-used by computational systems. This may be the case for such scenarios as image understanding (surveillance, intelligent vision, smart cameras, etc.) and media conversion (speech to text, picture to speech, speech to picture, etc.). Other scenarios are information retrieval (quickly and efficiently searching for various types of multimedia documents of interest to the user) and filtering in a stream of audiovisual content description (to receive only those multimedia data items which satisfy the user preferences). For example, a code in a television program triggers a suitably programmed PVR (Personal Video Recorder) to record that program, or an image sensor triggers an alarm when a certain visual event happens. Automatic transcoding may be performed from a string of characters to audible information or a search may be performed in a stream of audio or video data. In all these examples, the audiovisual information has been suitably "encoded" to enable a device or a computer code to take some action.

Audiovisual sources will play an increasingly pervasive role in our lives, and there will be a growing need to have these sources processed further. This makes it necessary to develop forms of audiovisual information representation that go beyond the simple waveform or sample-based, compression-based (such as MPEG-1 and MPEG-2) or even objects-based (such as MPEG-4) representations. Forms of representation that allow some degree of interpretation of the informations meaning are necessary. These forms can be passed onto, or accessed by, a device or a computer code. In the examples given above an image sensor may produce visual data not in the form of PCM samples (pixels values) but in the form of objects with associated physical measures and time information. These could then be stored and processed to verify if certain programmed conditions are met. A video recording device could receive descriptions of the audiovisual information associated to a program that would enable it to record, for example, only news with the exclusion of sport. Products from a company could be described in such a way that a machine could respond to unstructured queries from customers making inquiries.

MPEG-7 is a standard for describing the multimedia content data that support these operational requirements. The requirements apply, in principle, to both real-time and non real-time as well as push and pull applications. MPEG-7 does not standardize or evaluate applications, although in the development of the MPEG-7 standard applications have been used for understanding the requirements and evaluation of technology. It must be made clear that the requirements are derived from analyzing a wide range of potential applications that could use MPEG-7 tools. MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as broad a range of applications as possible.

5.7.1.2 MPEG-7 objectives

In October 1996, MPEG started a new work item to provide a solution to the questions described above. The new member of the MPEG family, named "Multimedia Content Description Interface" (in short MPEG-7), provides standardized core technologies allowing the description of audiovisual data content in multimedia environments. It extends the limited capabilities of proprietary solutions in identifying content that exist today, notably by including more data types.

Audiovisual data content that has MPEG-7 descriptions associated with it, may include: still pictures, graphics, 3D models, audio, speech, video, and composition information about how these elements are combined in a multimedia presentation (scenarios). A special case of these general data types is facial characteristics. MPEG-7 descriptions do, however, not depend on the ways the described content is coded or stored. It is possible to create an MPEG-7 description of an analogue movie or of a picture that is printed on paper, in the same way as of digitized content.

MPEG-7 allows different granularity in its descriptions, offering the possibility to have different levels of discrimination. Even though the MPEG-7 description does not depend on the (coded) representation of the material, MPEG-7 can exploit the advantages provided by MPEG-4 coded content. If the material is encoded using MPEG-4, which provides the means to encode audio-visual material as objects having certain relations in time (synchronization) and space (on the screen for video, or in the room for audio), it will be possible to attach descriptions to elements (objects) within the scene, such as audio and visual objects.

Because the descriptive features must be meaningful in the context of the application, they will be different for different user domains and different applications. This implies that the same material can be described using different types of features, tuned to the area of application. To take the example of visual material: a lower abstraction level would be a description of e.g. shape, size, texture, color, movement (trajectory) and position (where in the scene can the object be found); and for audio: key, mood, tempo, tempo changes, position in sound space. The highest level would give semantic information: "This is a scene with a barking brown dog on the left and a blue ball that falls down on the right, with the sound of passing cars in the background". Intermediate levels of abstraction may also exist.

The level of abstraction is related to the way the features can be extracted: many low-level features can be extracted in fully automatic ways, whereas high level features need (much) more human interaction.

Next to having a description of what is depicted in the content, it is also required to include other types of information about the multimedia data:

The form - An example of the form is the coding format used (e.g. JPEG, MPEG-2), or the overall data size. This information helps determining whether the material can be read by the user terminal;

Conditions for accessing the material - This includes links to a registry with intellectual property rights information, and price;

Classification - This includes parental rating, and content classification into a number of pre-defined categories;

Links to other relevant material - The information may help the user speeding up the search;

The context - In the case of recorded non-fiction content, it is very important to know the occasion of the recording (e.g. Olympic Games 1996, final of 200 meter hurdles, men).

The main elements of the MPEG-7 standard are:

Descriptions Tools comprising

- **Descriptors (D),** that define the syntax and the semantics of each feature (metadata element);
- **Description Schemes (DS),** that specify the structure and semantics of the relationships between their components, that may be both Descriptors and Description Schemes;

A Description Definition Language (DDL) to define the syntax of the MPEG-7 Description Tools and to allow the creation of new Description Schemes and, possibly, Descriptors and to allow the extension and modification of existing Description Schemes;

System tools to support binary coded representation for efficient storage and transmission, transmission mechanisms (both for textual and binary formats), multiplexing of descriptions, synchronization of descriptions with content, management and protection of intellectual property in MPEG-7 descriptions, etc.

Therefore, MPEG-7 Description Tools allows to create descriptions (i.e., a set of instantiated Description Schemes and their corresponding Descriptors at the users will), to incorporate application specific extensions using the DDL and to deploy the descriptions using System tools

The MPEG-7 descriptions of content that may include:

- Information describing the creation and production processes of the content (director, title, short feature movie).
- Information related to the usage of the content (copyright pointers, usage history, broadcast schedule).
- Information of the storage features of the content (storage format, encoding).
- Structural information on spatial, temporal or spatio-temporal components of the content (scene cuts, segmentation in regions, region motion tracking).
- Information about low level features in the content (colors, textures, sound timbres, melody description).
- Conceptual information of the reality captured by the content (objects and events, interactions among objects).
- Information about how to browse the content in an efficient way (summaries, variations, spatial and frequency subbands,).
- Information about collections of objects.
- Information about the interaction of the user with the content (user preferences, usage history).

All these descriptions are of course coded in an efficient way for searching, filtering, etc.

To accommodate this variety of complementary content descriptions, MPEG-7 approaches the description of content from several viewpoints. The sets of Description Tools developed on those viewpoints are presented here as separate entities. However, they are interrelated and can be combined in many ways. Depending on the application, some will present and others can be absent or only partly present.

A description generated using MPEG-7 Description Tools will be associated with the content itself, to allow fast and efficient searching for, and filtering of material that is of interest to the user.

MPEG-7 data may be physically located with the associated AV material, in the same data stream or on the same storage system, but the descriptions could also live somewhere else on the globe. When the content and its descriptions are not co-located, mechanisms that link the multimedia material and their MPEG-7 descriptions are needed; these links will have to work in both directions.

MPEG-7 addresses many different applications in many different environments, which means that it needs to provide a flexible and extensible framework for describing audiovisual data. Therefore, MPEG-7 does not define a monolithic system for content description but rather a set of methods and tools for the different viewpoints of the description of audiovisual content.

5.7.2 MPEG-7 terminology

This section presents the terminology used by MPEG-7. This terminology plays a major role in the understanding of the MPEG-7 process.

Data. Data is audio-visual information that will be described using MPEG-7, regardless of storage, coding, display, transmission, medium, or technology. This definition is intended to be sufficiently broad to encompass graphics, still images, video, film, music, speech, sounds, text and any other relevant AV medium. Examples for MPEG-7 data are an MPEG-4 stream, a video tape, a CD containing music, sound or speech, a picture printed on paper, and an interactive multimedia installation on the web.

Feature. A Feature is a distinctive characteristic of the data which signifies something to somebody. Features themselves cannot be compared without a meaningful feature representation (descriptor) and its instantiation (descriptor value) for a given data set. Some examples are: color of an image, pitch of a speech segment, rhythm of an audio segment, camera motion in a video, style of a video, the title of a movie, the actors in a movie, etc.

Descriptor. A Descriptor (D) is a representation of a Feature. A Descriptor defines the syntax and the semantics of the Feature representation. A Descriptor allows an evaluation of the corresponding feature via the descriptor value. It is possible to have several descriptors representing a single feature, i.e. to address different relevant requirements. Possible descriptors are: the color histogram, the average of the frequency components, the motion field, the text of the title, etc.

Descriptor Value. A Descriptor Value is an instantiation of a Descriptor for a given data set (or subset thereof).

Description Scheme. A Description Scheme (DS) specifies the structure and semantics of the relationships between its components, which may be both Descriptors and Description Schemes. A Description Scheme corresponds to an entity or relationship at the level of the MPEG-7 Audio-visual Conceptual Model. A Description Scheme shall have descriptive information and may participate in many-to-one relationships with other description elements. Examples: A movie, temporally structured as scenes and shots, including some textual descriptors at the scene level, and color, motion and some audio descriptors at the shot level. The distinction between a Description scheme and a Descriptor is that a Descriptor is concerned with the representation of a Feature, whereas the Description Scheme deals with the structure of a Description.

Description. A Description consists of a DS (structure) and the set of Descriptor Values (instantiations) that describe the Data. Depending on the completeness of the set of Descriptor Values, the DS may be fully or partially instantiated.

Coded Description. A Coded Description is a Description that has been encoded to fulfil relevant requirements such as compression efficiency, error resilience, random access, etc.

5.7.3 Scope of the Standard

MPEG-7 addresses applications that can be stored (on-line or off-line) or streamed (e.g. broadcast, push models on the Internet), and can operate in both real-time and non real-time environments. A real-time environment in this context means that the description is generated while the content is being captured

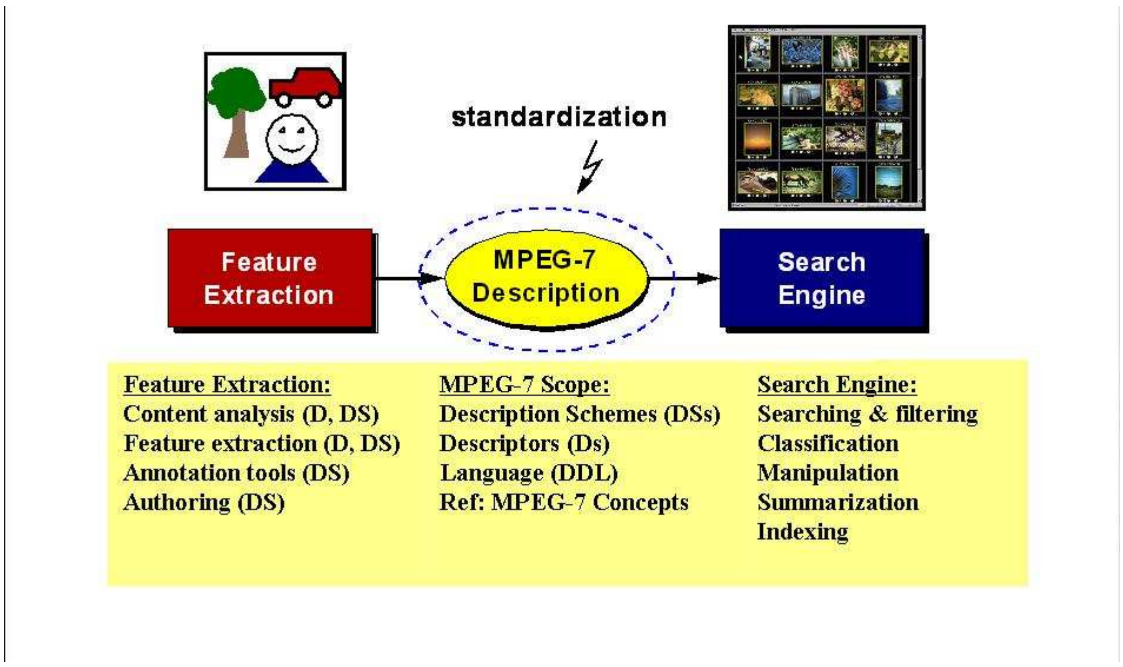


Figure 5.36: Scope of MPEG-7.

Figure 5.36 shows a highly abstract block diagram of a possible MPEG 7 processing chain, included here to explain the scope of the MPEG-7 standard. This chain includes feature extraction (analysis), the description itself, and the search engine (application). To fully exploit the possibilities of MPEG-7 descriptions, automatic extraction of features (or descriptors) are extremely useful. It is also clear that automatic extraction is not always possible, however. As was noted above, the higher the level of abstraction, the more difficult automatic extraction is, and interactive extraction tools will be of good use. However useful they are, neither automatic nor semi-automatic feature extraction algorithms is inside the scope of the standard. The main reason is that their standardisation is not required to allow interoperability, while leaving space for industry competition. Another reason not to standardise analysis is to allow making good use of the expected improvements in these technical areas.

Also the search engines, filter agents, or any other program that can make use of the description, are not be specified within the scope of MPEG-7; again this is not necessary, and here too, competition will produce the best results.

Figure 5.37 shows the relationship among the different MPEG-7 elements introduced above. The DDL allows the definition of the MPEG-7 description tools, both Descriptors and Description Schemes, providing the means for structuring the Ds into DSs. The DDL also allows the extension for specific applications of particular DSs. The description tools are instantiated as descriptions in textual format (XML) thanks to the DDL (based on XML Schema). Binary format of descriptions is obtained by means of the BiM defined in the Systems part.

Figure 5.38 explains a hypothetical MPEG-7 chain in practice [There can be other streams from content to user; these are not depicted here. Furthermore, it is understood that the MPEG-7 Coded Description may be textual or binary, as there might be cases where a binary efficient representation of the description is not needed, and a textual representation would suffice.] . From the multimedia content an Audiovisual description is obtained via manual or semi-automatic extraction. The AV

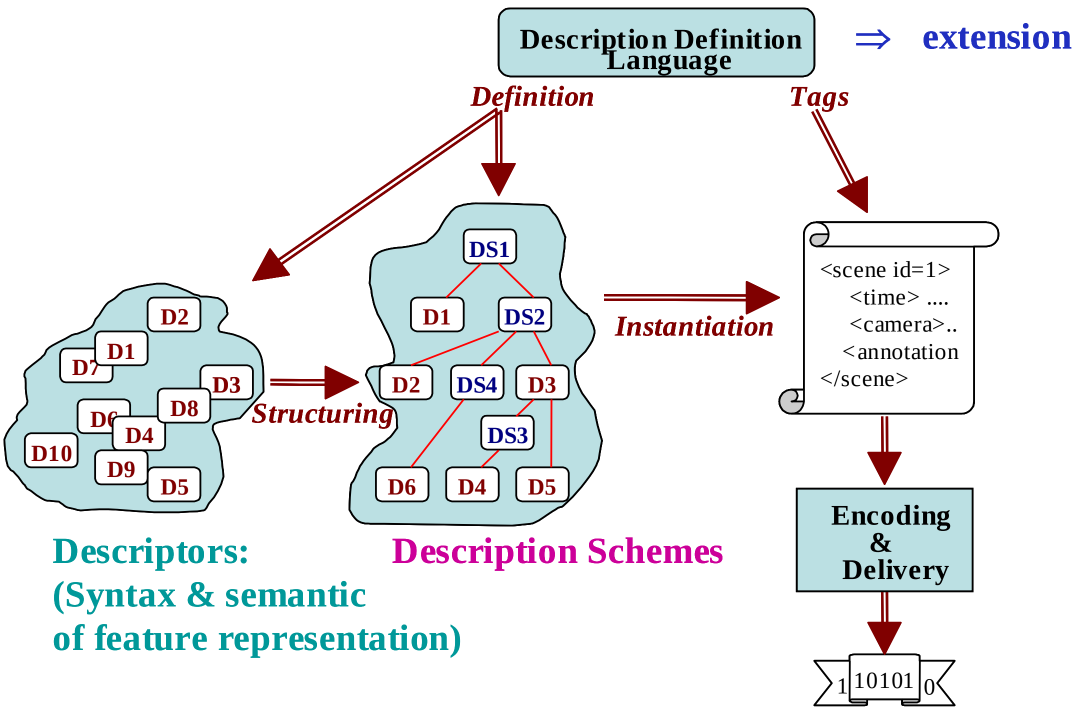


Figure 5.37: MPEG-7 main elements.

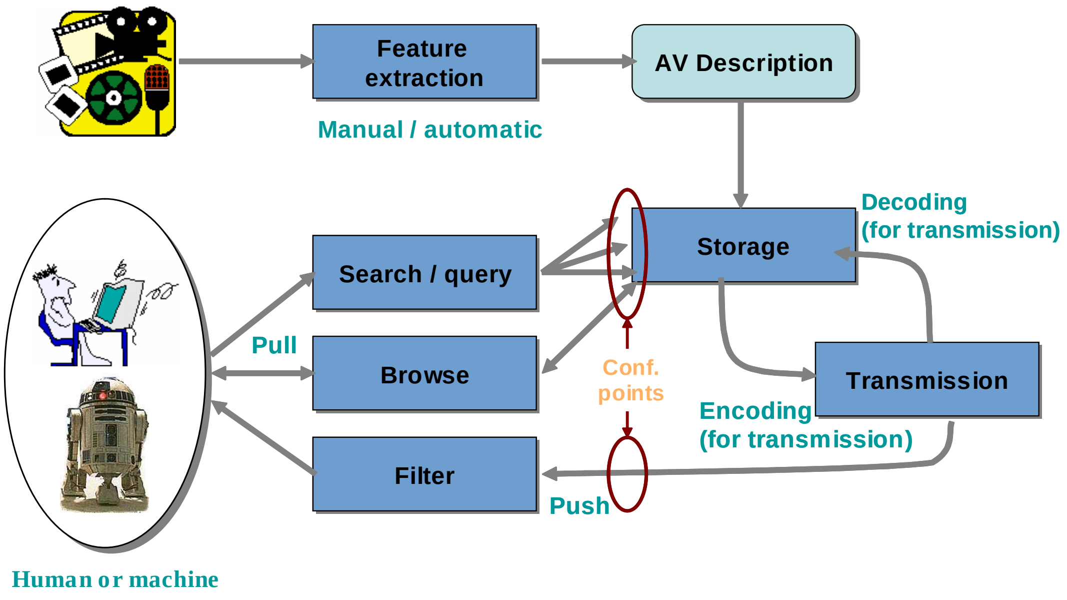


Figure 5.38: Abstract representation of possible applications using MPEG-7.

description may be stored (as depicted in the figure) or streamed directly. If we consider a pull

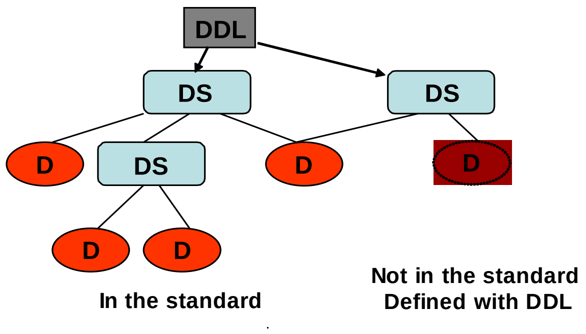


Figure 5.39: Abstract representation of possible relation between Descriptors and Description Schemes.

scenario, client applications will submit queries to the descriptions repository and will receive a set of descriptions matching the query for browsing (just for inspecting the description, for manipulating it, for retrieving the described content, etc.). In a push scenario a filter (e.g., an intelligent agent) will select descriptions from the available ones and perform the programmed actions afterwards (e.g., switching a broadcast channel or recording the described stream). In both scenarios, all the modules may handle descriptions coded in MPEG-7 formats (either textual or binary), but only at the indicated conformance points it is required to be MPEG-7 conformant (as they show the interfaces between an application acting as information server and information consumer).The emphasis of MPEG-7 is the provision of novel solutions for audio-visual content description. Thus, addressing text-only documents was not among the goals of MPEG-7. However, audio-visual content may include or refer to text in addition to its audio-visual information. MPEG-7 therefore has standardized different Description Tools for textual annotation and controlled vocabularies, taking into account existing standards and practices.

To provide a better understanding of the terminology introduced above (i.e. Descriptor, Description Scheme, and DDL), please refer to Figure 5.39 and Figure 5.39.

Figure 5.39 shows possible relation between Descriptors and Description Schemes. The arrows from DDL to Description Schemes are generated using the DDL. Furthermore, it indicates that the DDL provides the mechanism to build a Description Scheme which in turn forms the basis for the generation of a Description (see also Figure 5.40).

Figure 5.40 explains a hypothetical MPEG-7 chain in practice . The circular boxes depict tools that are doing things, such as encoding or decoding, whereas the square boxes represent static elements, such as a description. The dotted boxes in the figure encompass the normative elements of the MPEG-7 standard.

The emphasis of MPEG-7 is the provision of novel solutions for audio-visual content description. Thus, addressing text-only documents is be among the goals of MPEG-7. However, audio-visual content may include or refer to text in addition to its audio-visual information. MPEG-7 therefore considers existing solutions developed by other standardisation organisations for text only documents and support them as appropriate.

Besides the descriptors themselves, the database structure plays a crucial role in the final retrievals performance. To allow the desired fast judgement about whether the material is of interest, the indexing information will have to be structured, e.g. in a hierarchical or associative way.

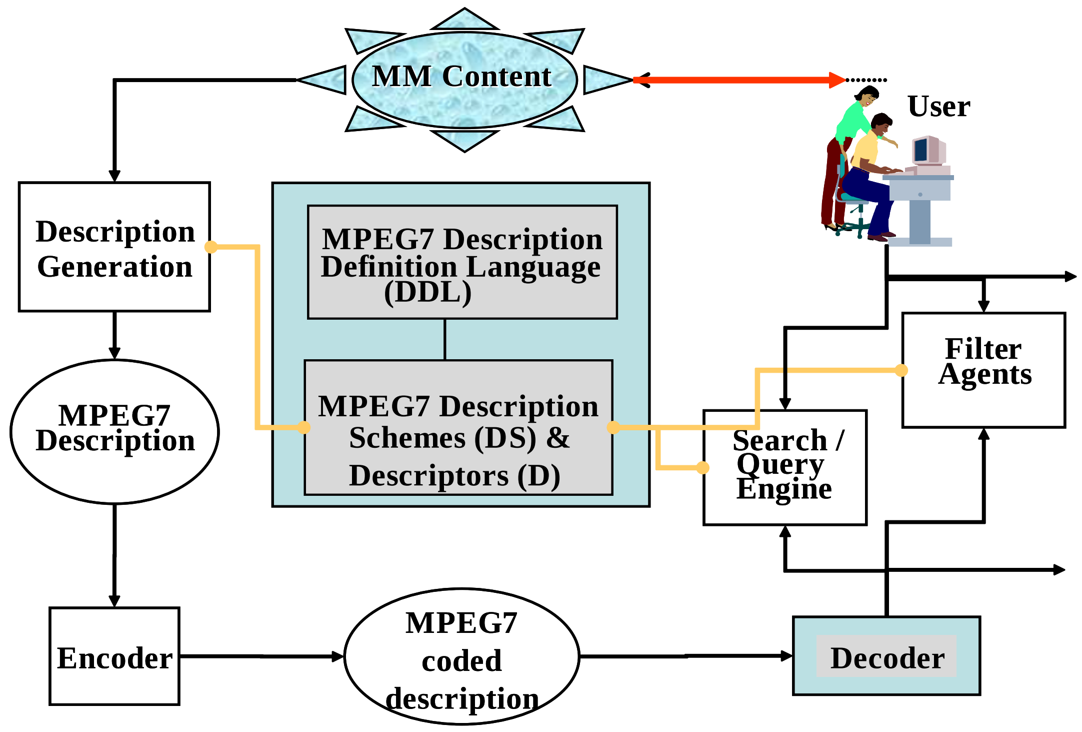


Figure 5.40: Abstract representation of possible applications using MPEG-7.

5.7.4 MPEG-7 Applications Areas

The elements that MPEG-7 standardizes supports a broad range of applications (for example, multimedia digital libraries, broadcast media selection, multimedia editing, home entertainment devices, etc.). MPEG-7 will also make the web as searchable for multimedia content as it is searchable for text today. This would apply especially to large content archives, which are being made accessible to the public, as well as to multimedia catalogues enabling people to identify content for purchase. The information used for content retrieval may also be used by agents, for the selection and filtering of broadcasted "push" material or for personalized advertising. Additionally, MPEG-7 descriptions allows fast and cost-effective usage of the underlying data, by enabling semi-automatic multimedia presentation and editing.

All applications domains making use of multimedia benefits from MPEG-7. Considering that at present day it is hard to find one not using multimedia, please extend the list of the examples below using your imagination:

- Architecture, real estate, and interior design (e.g., searching for ideas)
- Broadcast media selection (e.g., radio channel, TV channel)
- Cultural services (history museums, art galleries, etc.)
- Digital libraries (e.g., image catalogue, musical dictionary, bio-medical imaging catalogues, film, video and radio archives)
- E-Commerce (e.g., personalised advertising, on-line catalogues, directories of e-shops)

- Education (e.g., repositories of multimedia courses, multimedia search for support material)
- Home Entertainment (e.g., systems for the management of personal multimedia collections, including manipulation of content, e.g. home video editing, searching a game, karaoke)
- Investigation services (e.g., human characteristics recognition, forensics)
- Journalism (e.g. searching speeches of a certain politician using his name, his voice or his face)
- Multimedia directory services (e.g. yellow pages, Tourist information, Geographical information systems)
- Multimedia editing (e.g., personalised electronic news service, media authoring)
- Remote sensing (e.g., cartography, ecology, natural resources management)
- Shopping (e.g., searching for clothes that you like)
- Social (e.g. dating services)
- Surveillance (e.g., traffic control, surface transportation, non-destructive testing in hostile environments)

The way MPEG-7 data is used to answer user queries is outside the scope of the standard. In principle, any type of AV material may be retrieved by means of any type of query material. This means, for example, that video material may be queried using video, music, speech, etc. It is to the search engine to match the query data and the MPEG-7 AV description. A few query examples are:

- Play a few notes on a keyboard and retrieve a list of musical pieces similar to the required tune, or images matching the notes in a certain way, e.g. in terms of emotions.
- Draw a few lines on a screen and find a set of images containing similar graphics, logos, ideograms,...
- Define objects, including colour patches or textures and retrieve examples among which you select the interesting objects to compose your design.
- On a given set of multimedia objects, describe movements and relations between objects and so search for animations fulfilling the described temporal and spatial relations.
- Describe actions and get a list of scenarios containing such actions.
- Using an excerpt of Pavarotti's voice, obtaining a list of Pavarotti's records, video clips where Pavarotti is singing and photographic material portraying Pavarotti.

5.7.4.1 Making audio-visual material as searchable as text

MPEG-7 began as a scheme for making audiovisual material as searchable as text is today. Indeed, it is conceivable that the structure and discipline to even minimally describe multimedia may exceed the current state of textual information retrieval. Although the proposed multimedia content descriptions now serve as much more than search applications, they remain the primary applications for MPEG-7. These retrieval applications involve databases, audio-visual archives, and the Web-based Internet paradigm (a client requests material from a server). TV and film archives represent a typical application in this domain. They store vast amounts of multimedia material in several different formats (digital or analog tapes, film, CD-ROM, and so on) along with precise descriptive information (metadata) that may or may not be precisely timecoded. This metadata is stored in databases with proprietary formats. An enormous potential interest exists in an international standard format for the storage and exchange of descriptions that could ensure

- interoperability between video archive operators,
- perennial relevance of the metadata, and
- a wider diffusion of the data to the professional and general public.

To support these goals, MPEG-7 must accommodate visual and other searches of such existing multimedia databases. In addition, a vast amount of the older, analog audio-visual material will be digitized in years to come. This creates a tremendous opportunity to include content-based indexing features (extractable during the digitization/compression process5) into those existing databases.

For new audio-visual material, the ability to associate descriptive information within video streams at various stages of video production can dramatically improve the quality and productivity of manual, controlled vocabulary annotation of video data in a video archive. For example, preproduction and postproduction scripts, information captured or annotated during shooting, and postproduction edit lists would be very useful in the retrieval and reuse of archival material.

MPEG-7 specific requirements for such applications include

- Full-text descriptions as well as structured fields (database descriptions).
- A mechanism by which different MPEG-7 descriptions can support the ability to interoperate between different content-description semantics (such as different database schemas, different thesauri, and so on).
- A robust linking mechanism that allows referencing audio-visual objects or object instances and time references (including descriptions with incomplete or missing time references) even in an analog format.
- A structure to handle multiple versions of the same document at several stages in the production process and descriptions that apply to multiple copies of the same material.

For audio databases we face a similar situation. The consumer music industry is currently struggling with how to reach consumers with increasingly fragmented tastes. Music, as with all broadcast media artifacts, is undergoing the same Internet-flavored transformation as cable TV. An ideal way of presenting consumers with available music is to let them search effortlessly for what they want. Searchers may hum approximate renditions of the song they seek from a kiosk or from the comfort of their own home.7 Alternately, they may seek out music with features (musicians, style, tempo, year of creation) similar to those they already know. From there, they can listen to an appropriate sample (and perhaps view associated information such as lyrics or a video) and buy the music on the spot. The requirements for such types of audio-oriented applications on MPEG-7 include

- A mechanism that supports melody and other musical features that allow for reasonable errors by the indexer to accommodate queryby- humming.
- A mechanism that supports descriptors based on information associated with the data (such as textual data).

- Support description schemes that contain descriptors of visual, audio, and/or other features, and support links between the different media (cross-modal).

Other interesting applications related to audio include sound effects libraries, historical speech databases, and movie scene retrieval by memorable auditory events.

5.7.4.2 Supporting push and pull information acquisition methods

Filtering is essentially the converse of search. Search involves the "pull" of information, while filtering implies information "push." Search requests the inclusion of information, while filtering excludes data. Both pursuits benefit strongly from the same sort of meta-information. Typical domains for such applications include broadcasting and the emerging Webcasting. These domains have very distinct requirements, generally dealing with streamed descriptions rather than static descriptions stored on databases.

User-agent-driven media selection and filtering in a broadcasting environment has particular interest for MPEG-7. This approach lets users select information more appropriate to their uses and desires from a broadcast stream of 500 channels, using the same meta-information as that used in search. Moreover, this application gives rise to several subtypes, primarily divided among types of users. A consumer-oriented selection leads to personalized audio-visual programs, for example. This can go much farther than typical video-on-demand in collecting personally relevant news programs, for example. A content-producer-oriented selection made on the segment or shot level is a way of collecting raw material from archives. The requirements for such types of applications on MPEG-7 include

- Support for descriptors and description schemes that allow multiple languages.
- A mechanism by which a media object may be represented by a set of concepts that may depend on locality or language.
- Support efficient interactive response times.

However, new ways of automating and streamlining the presentation of that data also requires selecting and filtering. A system that combines knowledge about the context, user, application, and design principles with knowledge about the information to be displayed can accomplish this.8 Through clever application of that knowledge, you can have an intelligent multimedia presentation system. For MPEG, this requires a mechanism by which to

- encode contextual information and
- represent temporal relationships.

Finally, selecting and filtering facilitates accessibility to information for all users, especially those who suffer from one or several disabilities such as visual, auditory, motor, or cognitive disabilities. Providing active information representations might help overcome such problems. The key issue is to allow multimodal communication to present information optimized for individual users' abilities. Consider, for example, a search agent that does not exclude images as an information resource for the blind, but rather makes the MPEG-7 meta-data available. Aided by that metadata, sonification (auditory display) or haptic display becomes possible. Similarity of metadata helps provide a set of information in different modalities, in case the user can't access the particular information. Thus, MPEG-7 must support descriptions that contain descriptors of visual, audio, and/or other features.

5.7.4.3 Enabling nontraditional control of information

The following potential MPEG-7 applications don't limit themselves to traditional, media-oriented, multimedia content, but are functional within the metacontent representation in development under MPEG-7. Interestingly, they are neither push nor pull, but reject a certain amount of control over information through metadata. These applications reach into such diverse, but data-intensive domains as medicine and remote sensing. Such applications can only increase the usefulness and reach of this proposed international standard.

One of the specific applications is semi-automated video editing. Assuming that sufficient information exists about the content and structure of a multimedia object (see the previous section), a smart multimedia clip could start to edit itself in a manner appropriate to its neighboring multimedia. For example, a piece of music and a video clip from different sources could combine in such a way that the music stretches and contracts to synchronize with specific hit points in the video, creating an appropriate customized soundtrack.

This could be a new paradigm for multimedia, adding a method layer on top of MPEG-7 representation layer. (We by no means suggest that such methods for interaction be standardized in MPEG-7. As with many other advanced capabilities building on the standard, it is an issue for implementers to address.) Making multimedia aware to an extent opens access to novice users and increases productivity for experts. Such hidden intelligence on the part of the data itself shifts multimedia editing from direct manipulation to loose management of data.

Semi-automated multimedia editing encompasses a broad category of applications. It can aid home users as well as experts in studios through varying amounts of guidance or assistance through the process. In its simpler version, assisted editing can consist of an MPEG-7-enabled browser for selecting video shots, using a suitable shot description language. In an intermediate version, assisted editing could include planning, proposing shot selections and edit points, thereby satisfying a scenario expressed in a sequence description language.

The education domain relates closely to semi-automated editing. The challenge of using multimedia in educational software lies in exploiting the intrinsic information as much as possible to support different pedagogical approaches such as summarization, question answering, or detection of and reaction to misunderstanding or nonunderstanding. By providing direct access to short video sequences within a large database, MPEG-7 can promote the use of audio, video, and film archive material in higher education in many areas, including history, performing art, film music.

5.7.5 Mpeg-7 description tools

In this section, more details on the MPEG-7 description tools, which comprise all of MPEG-7 predefined descriptors and description schemes, will be presented. We can also define additional description tools for specific applications using the MPEG-7 Description Definition Language (DDL), an extension of the XML Schema.

We can group these description tools in different classes according to their functionality (see Figure 5.41). These description tools are standardized by three parts: Visual for descriptors related to visual features that apply to images and/or videos; Audio for the description tools related to audio features, covering areas from speech to music; and Multimedia Description Schemes for description tools related to features applying to audio, visual, and audio-visual content.

Figure 5.41 provides an overview of the organization of MPEG-7 Multimedia DSs into the following areas: Basic Elements, Content Description, Content Management, Content Description, Content Organization, Navigation and Access, and User Interaction.

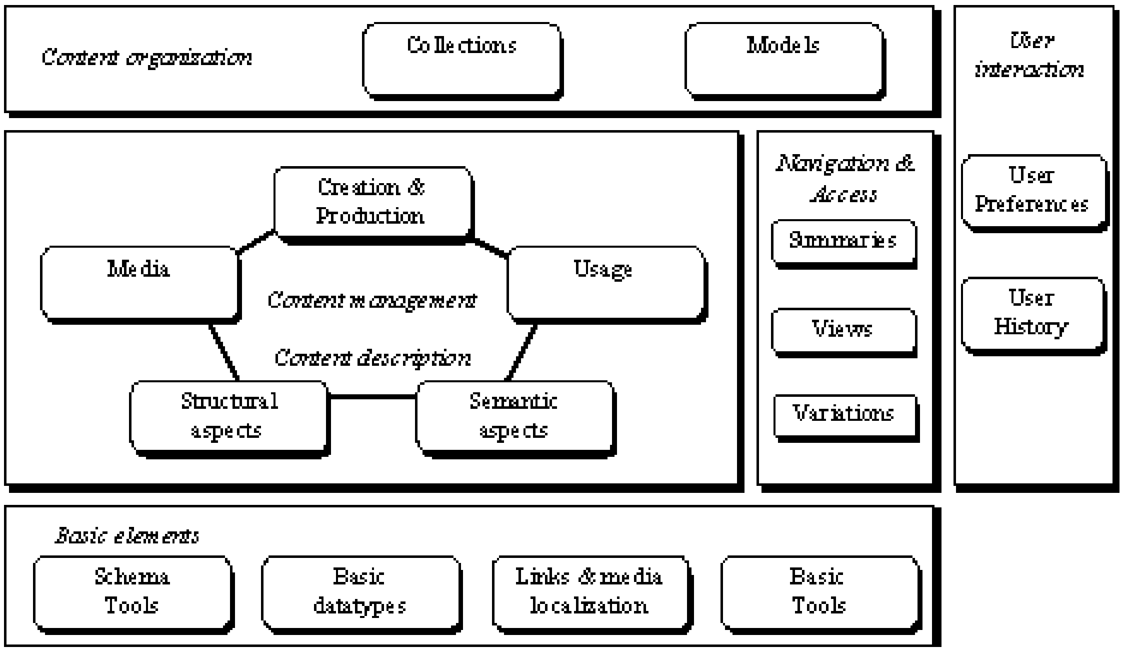


Figure 5.41: Overview of the MPEG-7 Multimedia DSs.

Basic elements MPEG-7 provides a number of Schema Tools that assist in the formation, packaging, and annotation of MPEG-7 descriptions. A number of basic elements are used throughout the MDS specification as fundamental constructs in defining the MPEG-7 DSs. The basic data types provide a set of extended data types and mathematical structures such as vectors and matrices, which are needed by the DSs for describing AV content. The basic elements include also constructs for linking media files, localizing pieces of content, and describing time, places, persons, individuals, groups, organizations, and other textual annotation.

Content Description Content Description Tools represent perceptible information, including structural aspects (structure description tools), audio and visual features, and conceptual aspects (semantic description tools).

Structure description tools let us describe content in terms of spatio-temporal segments organized in a hierarchical structure, for example, letting us define a table of contents or an index. We can attach audio, visual, annotation, and content management description tools to the segments to describe them in detail.

Visual description tools include the visual basic structures (such as description tools for grid layout, time series, and spatial coordinates) and visual description tools that let us describe color, texture, shape, motion, localization, and faces.

Audio description tools comprise the audio description framework (including the scale tree for creating a scalable series of audio samples, low-level audio descriptors, and the silence descriptor) and high-level audio description tools that let us describe musical instrument timbre, sound recognition, spoken content, and melody.

Semantic description tools let us describe the content with real-world semantics and conceptual notions: objects, events, abstract concepts, and relationships. We can cross-link the semantic and structure description tools with a set of links.

Content Management The Content Management Tools let us specify information about media features, creation, and usage of multimedia content.

Media description tools let us describe the storage media, coding format, quality, and transcoding hints for adapting content to different networks and terminals.

Creation description tools let us describe the creation process (for example, title, agents, materials, places, and dates), classification (for example, genre, subject, parental rating, and languages), and related materials. This information may very likely be subject to change during the lifetime of the audio-visual (AV) content.

Usage description tools let us describe the conditions for use (for example, rights and availability) and the history of use (for example, financial results and audience).

Content Organization MPEG-7 provides also DSs for organizing and modeling collections of audio-visual content and of descriptions. We can describe each collection as a whole by their attribute values characterized by models and statistics.

Navigation and Access Navigation and Access Tools let us specify summaries, partitions and decompositions, and variations of multimedia content for facilitating browsing and retrieval.

Summary description tools provide both hierarchical and sequential navigation modes to provide efficient preview access to the multimedia material.

Partitions and decompositions description tools allow multiresolution and progressive access in time, space, and frequency.

Variations description tools let us describe preexisting views of multimedia content: summaries, different media modalities, (for example, image and text), scaled versions, and so on.

User Interaction The User Interaction tools describe user preferences and usage history pertaining to the consumption of the multimedia material. This allows, for example, matching between user preferences and MPEG-7 content descriptions in order to facilitate personalization of audio-visual content access, presentation and consumption.

5.7.6 MPEG-7 Audio

MPEG-7 Audio comprises five technologies: the audio description framework (which includes scalable series, low-level descriptors, and the uniform silence segment), musical instrument timbre description tools, sound recognition tools, spoken content description tools, and melody description tools.

5.7.6.1 MPEG-7 Audio Description Framework

The Audio Framework contains low-level tools designed to provide a basis for the construction of higher level audio applications. By providing a common platform for the structure of descriptions and the basic semantics for commonly regarded audio features, MPEG-7 Audio establishes a platform for interoperability across all applications that might be built on the framework.

There are essentially two ways of describing low-level audio features. One may sample values at regular intervals or one may use Segments (see the MDS description) to demark regions of similarity and dissimilarity within the sound. Both of these possibilities are embodied in two low-level descriptor types (one for scalar values, such as power or fundamental frequency, and one for vector types, such as spectra), which create a consistent interface. Any descriptor inheriting from these types can be instantiated, describing a segment with a single summary value or a series of sampled values, as the application requires.

The sampled values themselves may be further manipulated through another unified interface: they can form a Scalable Series. The Scalable Series allows one to progressively down-sample the data contained in a series, as the application, bandwidth, or storage required. The scale tree may also store various summary values along the way, such as minimum, maximum, and variance of the descriptor values.

The low-level audio descriptors are of general importance in describing audio. There are seventeen temporal and spectral descriptors that may be used in a variety of applications. They can be roughly divided into the following groups:

Basic: The two basic audio Descriptors are temporally sampled scalar values for general use, applicable to all kinds of signals. The AudioWaveform Descriptor describes the audio waveform envelope (minimum and maximum), typically for display purposes. The AudioPower Descriptor describes the temporally-smoothed instantaneous power, which is useful as a quick summary of a signal, and in conjunction with the power spectrum, below.

Basic Spectral. The four basic spectral audio Descriptors all share a common basis, all deriving from a single time-frequency analysis of an audio signal. They are all informed by the first Descriptor, the AudioSpectrumEnvelope Descriptor.

AudioSpectrumEnvelope descriptor which is a logarithmic-frequency spectrum, spaced by a power-of-two divisor or multiple of an octave. This AudioSpectrumEnvelope is a vector that describes the short-term power spectrum of an audio signal. It may be used to display a spectrogram, to synthesize a crude "auralization" of the data, or as a general-purpose descriptor for search and comparison.

AudioSpectrumCentroid descriptor describes the center of gravity of the log-frequency power spectrum. This Descriptor is an economical description of the shape of the power spectrum, indicating whether the spectral content of a signal is dominated by high or low frequencies.

AudioSpectrumSpread descriptor complements the previous Descriptor by describing the second moment of the log-frequency power spectrum, indicating whether the power spectrum is centered near the spectral centroid, or spread out over the spectrum. This may help distinguish between pure-tone and noise-like sounds.

AudioSpectrumFlatness descriptor describes the flatness properties of the spectrum of an audio signal for each of a number of frequency bands. When this vector indicates a high deviation from a flat spectral shape for a given band, it may signal the presence of tonal components.

Signal parameters The two signal parameter Descriptors apply chiefly to periodic or quasi-periodic signals.

AudioFundamentalFrequency descriptor describes the fundamental frequency of an audio signal. The representation of this descriptor allows for a confidence measure in recognition of the fact that the various extraction methods, commonly called "pitch-tracking," are not perfectly accurate, and in recognition of the fact that there may be sections of a signal (e.g., noise) for which no fundamental frequency may be extracted.

AudioHarmonicity descriptor represents the harmonicity of a signal, allowing distinction between sounds with a harmonic spectrum (e.g., musical tones or voiced speech [e.g., vowels]), sounds with an inharmonic spectrum (e.g., metallic or bell-like sounds) and sounds with a non-harmonic spectrum (e.g., noise, unvoiced speech [e.g., fricatives like /f/], or dense mixtures of instruments).

Timbral Temporal The two timbral temporal descriptors describe temporal characteristics of segments of sounds, and are especially useful for the description of musical timbre (characteristic tone quality independent of pitch and loudness). Because a single scalar value is used to represent the evolution of a sound or an audio segment in time, these Descriptors are not applicable for use with the Scalable Series.

LogAttackTime descriptor characterizes the "attack" of a sound, the time it takes for the signal to rise from silence to the maximum amplitude. This feature signifies the difference between a sudden and a smooth sound.

TemporalCentroid descriptor also characterizes the signal envelope, representing where in time the energy of a signal is focused. This Descriptor may, for example, distinguish between a decaying piano note and a sustained organ note, when the lengths and the attacks of the two notes are identical.

Timbral Spectral The five timbral spectral Descriptors are spectral features in a linear-frequency space especially applicable to the perception of musical timbre.

SpectralCentroid descriptor is the power-weighted average of the frequency of the bins in the linear power spectrum. As such, it is very similar to the AudioSpectrumCentroid Descriptor, but specialized for use in distinguishing musical instrument timbres. It is has a high correlation with the perceptual feature of the "sharpness" of a sound.

The four remaining timbral spectral Descriptors operate on the harmonic regularly-spaced components of signals. For this reason, the descriptors are computed in linear-frequency space.

HarmonicSpectralCentroid is the amplitude-weighted mean of the harmonic peaks of the spectrum. It has a similar semantic to the other centroid Descriptors, but applies only to the harmonic (non-noise) parts of the musical tone. T

HarmonicSpectralDeviation descriptor indicates the spectral deviation of log-amplitude components from a global spectral envelope.

HarmonicSpectralSpread describes the amplitude-weighted standard deviation of the harmonic peaks of the spectrum, normalized by the instantaneous HarmonicSpectralCentroid.

HarmonicSpectralVariation descriptor is the normalized correlation between the amplitude of the harmonic peaks between two subsequent time-slices of the signal.

Spectral Basis The two spectral basis Descriptors represent low-dimensional projections of a high-dimensional spectral space to aid compactness and recognition. These descriptors are used primarily with the Sound Classification and Indexing Description Tools, but may be of use with other types of applications as well.

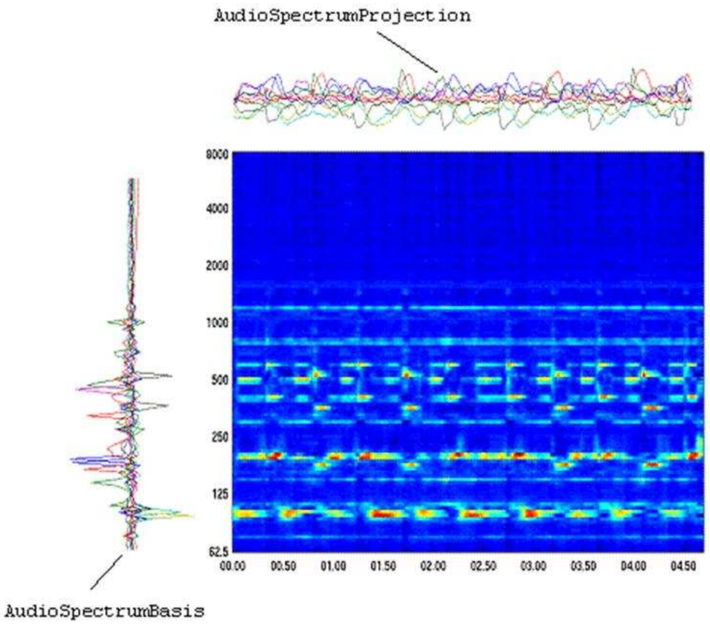


Figure 5.42: AudioSpectrumEnvelope description of a pop song. The required data storage is NM values where N is the number of spectrum bins and M is the number of time points.

AudioSpectrumBasis descriptor is a series of (potentially time-varying and/or statistically independent) basis functions that are derived from the singular value decomposition of a normalized power spectrum.

AudioSpectrumProjection descriptor is used together with the AudioSpectrumBasis Descriptor, and represents low-dimensional features of a spectrum after projection upon a reduced rank basis.

Together, the descriptors may be used to view and to represent compactly the independent subspaces of a spectrogram. Often these independent subspaces (or groups thereof) correlate strongly with different sound sources. Thus one gets more salience and structure out of a spectrogram while using less space. For example, in Figure 5.42, a pop song is represented by an AudioSpectrumEnvelope Descriptor, and visualized using a spectrogram. The same song has been data-reduced in Figure 5.43, and yet the individual instruments become more salient in this representation.

Each of these can be used to describe a segment with a summary value that applies to the entire segment or with a series of sampled values. The Timbral Temporal group is an exception, as their values only apply to segments as a whole.

While low-level audio descriptors in general can serve many conceivable applications, the spectral flatness descriptor specifically supports the functionality of robust matching of audio signals. Applications include audio fingerprinting, identification of audio based on a database of known works and, thus, locating metadata for legacy audio content without metadata annotation.

Additionally, a very simple but useful tool is the

Silence descriptor. It attaches the simple semantic of "silence" (i.e. no significant sound) to an Audio

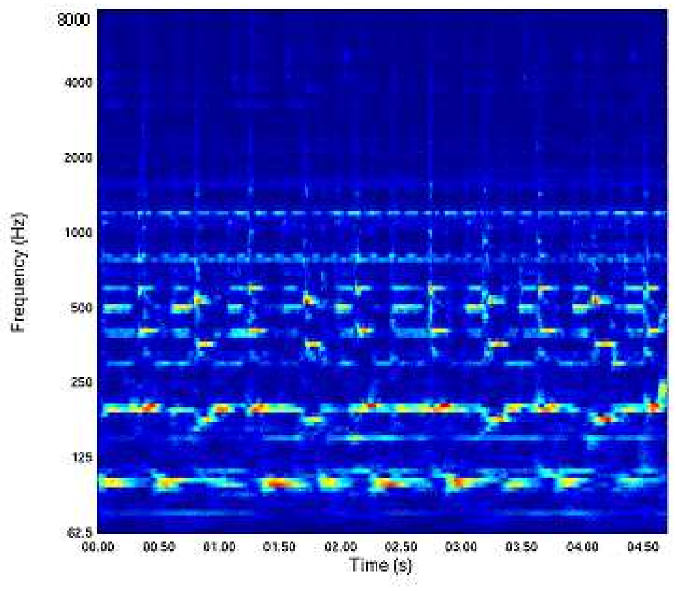


Figure 5.43: A 10-basis component reconstruction showing most of the detail of the original spectrogram including guitar, bass guitar, hi-hat and organ notes. The left vectors are an AudioSpectrumBasis Descriptor and the top vectors are the corresponding AudioSpectrumProjection Descriptor. The required data storage is $10(M + N)$ values.

Segment. It may be used to aid further segmentation of the audio stream, or as a hint not to process a segment.

5.7.6.2 High-level audio description tools (Ds and DSs)

Because there is a smaller set of audio features (as compared to visual features) that may canonically represent a sound without domain-specific knowledge, MPEG-7 Audio includes a set of specialized high-level tools that exchange some degree of generality for descriptive richness. The five sets of audio Description Tools that roughly correspond to application areas are integrated in the standard: audio signature, musical instrument timbre, melody description, general sound recognition and indexing, and spoken content. The latter two are excellent examples of how the Audio Framework and MDS Description Tools may be integrated to support other applications.

Audio Signature Description Scheme While low-level audio Descriptors in general can serve many conceivable applications, the spectral flatness Descriptor specifically supports the functionality of robust matching of audio signals. The Descriptor is statistically summarized in the AudioSignature Description Scheme as a condensed representation of an audio signal designed to provide a unique content identifier for the purpose of robust automatic identification of audio signals. Applications include audio fingerprinting, identification of audio based on a database of known works and, thus, locating metadata for legacy audio content without metadata annotation

Musical Instrument Timbre description tools Timbre descriptors aim at describing perceptual features of instrument sounds. Timbre is currently defined in the literature as the perceptual features that make two sounds having the same pitch and loudness sound different. The aim of the Timbre description tools is to describe these perceptual features with a reduced set of descriptors. The descriptors relate to notions such as “attack”, “brightness” or “richness” of a sound. Within four detailed possible classes of musical instrument sounds, two classes are well detailed, and had been the subject of core experiment development. At this point, harmonic, coherent sustained sounds, and non-sustained, percussive sounds are represented in the standard. The timbre descriptor for sustained harmonic sounds combines the above Timbral Spectral low-level descriptors with a log attack time descriptor. The percussive instrument descriptor combines the Timbral Temporal low-level descriptors with a spectral centroid descriptor. Comparisons between descriptions using either set of descriptors are done with an experimentally-derived scaled distance metric.

Melody Description Tools The melody Description Tools include a rich representation for monophonic melodic information to facilitate efficient, robust, and expressive melodic similarity matching. The Melody Description Scheme includes a MelodyContour Description Scheme for extremely terse, efficient melody contour representation, and a MelodySequence Description Scheme for a more verbose, complete, expressive melody representation. Both tools support matching between melodies, and can support optional supporting information about the melody that may further aid content-based search, including query-by-humming.

MelodyContour Description Scheme uses a 5-step contour (representing the interval difference between adjacent notes), in which intervals are quantized into large or small intervals, up, down, or the same. The Melody Contour DS also represents basic rhythmic information by storing the number of the nearest whole beat of each note, which can dramatically increase the accuracy of matches to a query.

MelodySequence. For applications requiring greater descriptive precision or reconstruction of a given melody, the MelodySequence Description Scheme supports an expanded descriptor set and high precision of interval encoding. Rather than quantizing to one of five levels, the precise pitch interval (to cent or greater precision) between notes is kept. Precise rhythmic information is kept by encoding the logarithmic ratio of differences between the onsets of notes in a manner similar to the pitch interval. Arrayed about these core Descriptors are a series of optional support Descriptors such as lyrics, key, meter, and starting note, to be used as desired by an application.

Sound recognition tools The sound recognition descriptors and description schemes are a collection of tools for indexing and categorization of general sounds, with immediate application to sound effects. Support for automatic sound identification and indexing is included as well as tools for specifying a taxonomy of sound classes and tools for specifying an ontology of sound recognizers. Such recognizers may be used to automatically index and segment sound tracks.

The recognition tools use the low-level Spectral Basis descriptors as their foundation. These basis functions are then further segmented into a series of states that comprise a statistical model, such as a hidden Markov or Gaussian mixture model, which is then trained on the likely transitions between these states. This model may stand on its own as a representation, have a label associated with it according to the semantics of the original sound, and/or associated with other models in order to categorize novel sounds input into a recognition system.

Spoken Content description tools The Spoken Content description tools allow detailed description of words spoken within an audio stream. In recognition of the fact that current Automatic

Speech Recognition (ASR) technologies have their limits, and that one will always encounter out-of-vocabulary utterances, the Spoken Content description tools sacrifice some compactness for robustness of search. To accomplish this, the tools represent the output and what might normally be seen as intermediate results of Automatic Speech Recognition (ASR). The tools can be used for two broad classes of retrieval scenario: indexing into and retrieval of an audio stream, and indexing of multimedia objects annotated with speech.

The Spoken Content description tools are divided into two broad functional units: the lattice, which represents the actual decoding produced by an ASR engine, and the header, which contains information about the speakers being recognized and the recognizer itself. The lattice consists of combined word and phone lattices for each speaker in an audio stream. By combining these lattices, the problem of out-of-vocabulary words is greatly alleviated and retrieval may still be carried out when the original word recognition was in error.

References

References

Contents