S8.6

EVENT-BASED MULTIPLE-RESOLUTION ANALYSIS OF SPEECH SIGNALS

Toomas Altosaar and Matti Karjalainen

Helsinki University of Technology, Acoustics Lab.
Otakaari 5A, 02150 Espoo, Finland

ABSTRACT

Several studies have emphasized the importance of avoiding the use of a single preferred resolution or scale in signal analysis and pattern recognition, see e.g. *scale-space filtering* by Witkin [1]. We have studied the problem from the speech analysis and auditory modeling point of view. In this paper we introduce our methodology of *multiple-resolution analysis* and *event-based representation* of speech signals. The computation of multiple-resolution filtering, event detection and parsing of event structures are described with examples and discussions on auditory modeling aspects. Both the approach and our implementation are entirely based on object-oriented programming which enables a systematic framework for the hierarchical nature of the method.

INTRODUCTION

One of the problems not often fully understood in speech analysis and recognition front ends is the selection of proper scales and resolutions in the time and frequency domains. To avoid the difficulties as a consequence of using any single preferred resolution *Witkin* [1] proposed a general signal analysis method called *scale-space filtering*. *Lyon* [2] has used the concept in speech recognition using cochleagrams. We have studied the problem from a speech-specific and auditory modeling point of view and emphasize the need for *multiple-resolution analysis* and *event-based representation* of speech signals.

We feel that the application of multiple resolution analysis in the domain of speech processing is especially appropriate since experimental evidence exists which shows that the human auditory system may function in a somewhat similar way. The flexibility of the hearing system to apply context-dependent resolutions in both the frequency and time domain is evident. This together with principles like the 3.5 Bark spectral integration of formant clusters proposed by Chistovich [3], temporal effects such as forward and backward masking and modulation sensitivity with maximum around 4 Hz can be investigated and modeled in the light of multiple-resolution analysis. One truly important idea is that by analyzing with multiple resolutions in parallel we can form a structural representation of a signal that emphasizes the prominent features and the inherent structure without discarding or thresholding away details that may be potentially important in later context-dependent classification and recognition.

In both the frequency and time domain certain structures of signals are especially relevant. We have found that two different types of primitive structures (events) are useful in the auditory domain: *blocks* and *edges*. These correspond to the *crank* and *corner* primitives used in shape analysis and were suggested by Asada and Brady [4] in their *curvature primal sketch*. Under multiple-resolution analysis speech formants in a spectrum appear as block structures while the edges of the formants are described accordingly by edge structures. Analyzing a parametric function like total-loudness in the time domain will reveal block-like structures in areas where voiced phonemes are likely while edges correspond to segment boundaries.

The methodology and implementation of our experimental system are entirely based on an object-oriented approach which allows for high flexibility in both the analysis and the hierarchical representation of event structures . Object-oriented programming is a useful abstraction mechanism that enables the user to define *classes*, create and discard *object instances* which can represent or model real things. The objects have local *instance variables* (ivars) while *method functions* are used to interface them to the computational environment.

Our implementation of the analysis system is based upon an object-oriented signal processing system called QuickSig [5] running on the Symbolics Lisp machine. The QuickSig system has object classes such as signals, windows and filter-banks each with their own set of method functions for signal processing operations. We have based the multiple-resolution analysis part of the system upon existing object classes already defined in QuickSig, but have defined new classes for the events and event structures. Throughout the paper we will refer to the object-oriented implementation and describe new classes when mentioned.

MULTIPLE-RESOLUTION ANALYSIS

Systems that use some preferred scale in the analysis of signals are susceptible to *biasing* the results one way or another. For example, if a phoneme recognition system has been *trained* to expect phonemes of duration 120 ms, it might have problems classifying an elongated one of 200 ms. The human auditory system is able to accept and interpret signals over several resolutions in both the time and frequency domains. *Scale-space filtering*, as proposed by Witkin [1], allows for such an analysis without any preassumed scale, frame-length, duration or frequency resolution. It allows for controlled and well-behaved analysis of a signal from a fine to a coarse scale by filtering the incoming signal with a continuum of filters exhibiting a special and unique property. We believe that the auditory system processes information in similar parallel fashion by performing this operation which we call *multiple-resolution analysis*.

Scale-space filtering adds the extra dimension of *scale* (σ) to a signal. It does this by filtering the input signal $f(x)$ through a filtering kernel $g(x,\sigma)$ to yield a half-plane surface $f(x,\sigma)$. The filtering kernel as used by Witkin is the second-derivative Gaussian and it exhibits a unique property shown by Babaud et al. [6] that no new zero-crossings will occur in $f(x,\sigma)$ as σ is increased. In practice, a discrete sampling of σ in exponentially increasing intervals (e.g. by a factor of $2^{1/2}$) suffices to obtain a faithful surface of $f(x,\sigma)$. Zero-crossings in the second derivative indicate the inflection points of $f(x)$ and are related to the locations where the extrema of the first derivative occur. As is described later, we use a first derivative kernel and obtain both the extrema and zero-crossings of $f(x,\sigma)$ at different scales. Figure 1 shows sets of first and second derivative Gaussian kernels which are represented by *m-signal* (multi-channel signal) objects in the QuickSig system and act as filter banks. Given an input signal, the filter-bank will return a new *m-signal* which contains the sampled scale-space image.

CH2561-9/88/0000-0327 $1.00 © 1988 IEEE

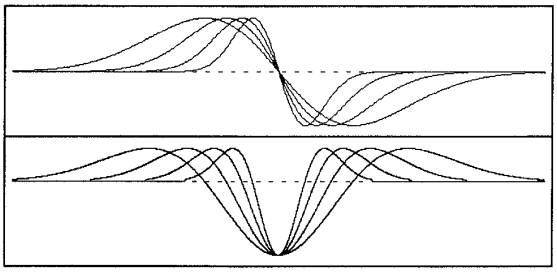

Fig.1. Derivatives of Gaussian kernels used in scale-space filtering: a) first, b) second derivative.

We have studied the effects of using non-Gaussian kernels for efficient implementation reasons. The top waveform of figure 2 shows the form of a non-symmetric piecewise-linear response, while the bottom shows the response of a multi-scale derivative filter suggested by Lyon [2]. Further on in this paper we will show what results are obtained with both of these kernels when used in multiple-resolution analysis and how they differ from the ideal Gaussian case.

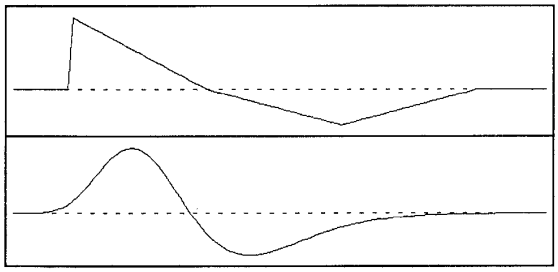

Fig.2. Other types of kernels used in Multiple-Resolution Analysis experiments.

EVENT DETECTION

After the filter-bank has returned its output as a set of signals, a search for *events* is performed. We define an event to be an occurence in a signal with some significance: a local extremum, a zero-crossing, a level-crossing, etc. Several properties related to such a point can be packaged together to constitute an event object.

An instance of *basic-event* is simply an object that has as its local variables (ivars) *point*, *value*, and *type*. *Point* records the location of the event (in time or frequency units), *value* records the amplitude (or slope in the case of a zero-crossing) of the response at *point*, and *type* states what kind of an event it is (zero-crossing, max, min etc.). Higher level classes of events which are based upon *basic-event* contain additional ivars such as pointers (to indicate which signal it came from and in which sequence it is stored), a *resolution* ivar which records the scale of the filter that created the signal the event is from, and link slots that indicate its relationship to other events. As will be seen, links play an important role when the scale-space image is parsed into *event-structures*.

Event objects resulting from a multiple-resolution analysis can be visualized in a two-dimensional *event-map*. Fig. 3 shows an example from the analysis of a speech loudness function where the position of each event in time and resolution scale is plotted with a character corresponding to the type of the event. Four types of events are displayed: + for positive blocks, - for negative blocks, ↑ for rising edges and ↓ for falling edges. Continuous contour lines are formed in the case of Gaussian kernels if the number of resolution scales goes to infinity [1].

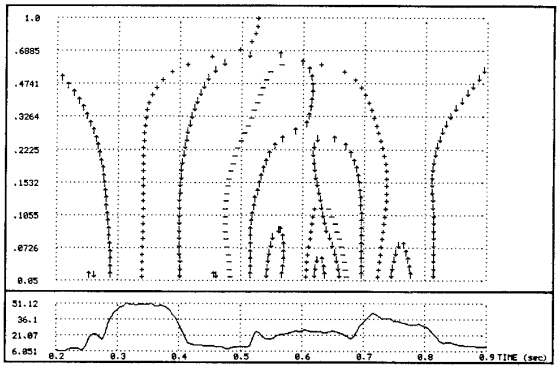

Fig. 3. An event map of a loudness function (word /kaksi/). Dimensions are resolution scale vs. time.

Events have method functions associated with them as well. For the *basic-event* class the function:

(distance event1 event2)

returns the point difference between event1 and event2. For a higher level event which contains a *resolution* ivar the same syntax yields an Euclidean distance measure in scale-space, using the differences in *point* and in *resolution* to determine their distance.

While a signal is being scanned for events the events which are found must be stored. We have chosen to store the events in an *event-sequence*, a class that is based upon *sequence* in QuickSig. An *event-sequence* is similar in structure to a *signal*, having an array to keep the samples which in this case are events. When an event is placed into an *event-sequence*, the event's *owner* ivar is set to the *event-sequence* so it may refer to it if needed. Also, events from a common signal have their *left* and *right* links set to their neighbouring events. An event might require this information while in the parsing stage.

EVENT STRUCTURES

It is natural to assume that events that are closely and systematically grouped are parts of a common larger perceptual object. When using Gaussian filters the connection of events from one resolution to another are well behaved and essentially quite restricted since no new events will appear as scale is increased. This condition facilitates the *parsing* or connecting of events between scales. More practical kernel functions do not behave as well but still some higher-level composite events and event structures can be obtained by proper parsing methods. It is important to note that the composite events don't have to be regular closed contours. Examples of parsing results are given later in this paper.

Event Parsing

Our first approach to generating structures was to take a fine scale event and find its "closest match" in the available scale-space plane. This was done by searching the next coarser scale until a local minimum appeared in the distance function and the event satisfied the parsing rules. These events were then linked together by *up* and *down* links. This method worked well for short off-line analysis but was not very efficient since it was of $O(n^2)$.

A more efficient method which does not require a lengthy search and is more applicable to a real-time application was developed. It uses the *left* and *right* link ivars of the events to gain access into the next scale. It assumes that $event_{i-1,j}$ ($i\text{-}1^{th}$ event in the j^{th} scale) has already been linked to the next larger scale event (e.g. $event_{k,j+1}$). To obtain a good estimate for the correctly matching event in the next larger scale two short searches and one access are made through the links. First travel along the same resolution moving left (back in time through the *left* link) until an event is found that

has an *up* link active, second go up this link (through the *up* link), and finally find the best match in this scale. At best this method takes only three accesses. If no link is found to a higher scale then the event resorts to accessing the next scale through its *owner* link. This link points to its event-sequence, and then proceeds through the event-sequence's *coarser* link (event-sequences are also linked). This occurs at start-up when no events in separate scales are linked.

These linking procedures are local to the events (method functions of class *event*) and have all the necessary matching rules (dependent upon event type) and linking procedures imbedded in them. Separate event classes for blocks (zero-crossing events) and edges (extremum events) with separate method functions could be added for increased structure and clarity. The linked events can now be transformed into higher level *event-structures*.

Line Events

A new object class called *line-event* is used to summarize the information held within a linked set of events and models a trajectory in scale-space. These objects effectively represent perceptually relevant objects like formants or formant clusters and their edges in frequency, or temporal objects like segments, boundaries and transitions in parametric time signals. Line events have their own method functions which allow for classifying line types and determining their relationship to other line-events in the scale-space.

Higher Level Structures

Using the primitives *block* and *edge* higher level structures are constructed. Edges may be joined to form *contours* and used to generate *ternary-trees*, blocks joined to formed *n-branching trees*, etc. Higher level composite structures such as an *edge-block-edge* can be used to describe formants or formant clusters and their quality. Phonetic distance measures can be based on these structures in an attempt to model perception of vowels. More generally, such composite structures can be related to phonetic and linguistic units by fuzzy membership values that are iterated towards a final recognition decision when enough context information is known. Due to the discrete nature of the units they can be interfaced to natural language processing. Rule-based symbolic processing and artificial intelligence methods may be applied.

EXPERIMENTAL RESULTS

In this section we present some experimental results using multiple-resolution analysis. Our signal examples were from the output of an auditory model [7],[8] and represented both temporal and frequency domain information. For the spectral input we used the *auditory spectrum* (loudness density vs. Bark or critical band [9]) of the vowel /i/. In the time domain we used the *total-loudness* function [9] of the Finnish word /kaksi/. This was computed by summing over the entire auditory spectrum every 5 msec in time.

Time Domain Results

The ideal Gaussian analysis is shown in Fig. 4. The bottom window contains the input signal, total-loudness of /kaksi/ as a function of time. The other two windows show resolution scale vs. time. The topmost one shows the edge line-events (extrema of the first derivative) while the middle window contains the block line-events (zero-crossings of the first derivative). The amplitude of the multiple-resolution filtered response at these points is represented by short horizontal lines centred around the line-events. This information is used to indicate the relative prominence a line-event has at each scale. It is different from Witkin's measure of maximum stability.

As can be seen in the middle window, at small scale values the location of the blocks is influenced by the fine structure of the signal while at scales of the highest prominence (max line breadth) the position corresponds to the natural block unit. At larger scales the location depends more upon the local neighbourhood and at very large scale values only one block remains and this represents the complete word. Both types of blocks exist here, local maxima and minima. If we select a scale of 0.25 we can see that only three blocks remain. The first one (a local maximum at 0.35 sec) describes the vowel /a/, the second (a local minimum at 0.5 sec) the stop /k/ between /a/ and /s/, and the final block (a local maximum at 0.72 sec) is the final /i/ in /kaksi/.

Block structures can be related to their neighbouring edge structures at similar resolution scales. From these relations we can obtain a better description of a block by knowing its effective span at some resolution. An example of this is seen at a scale of 0.11 where the peak in loudness for the final /i/ occurs at about 0.73 seconds (indicated by the line-event structure which crosses time 0.73 and resolution 0.11). The corresponding edges for this block can be found from the same scale and occur at 0.7 and 0.81 seconds as seen from the top window. From this *edge-block-edge* structure we can infer that the centre of the block is at 0.73, it has a sharp rising edge, and a slow falling edge. This is because the edge-to-block distance is only 0.03 seconds as compared to the block-to-edge distance of 0.08 seconds.

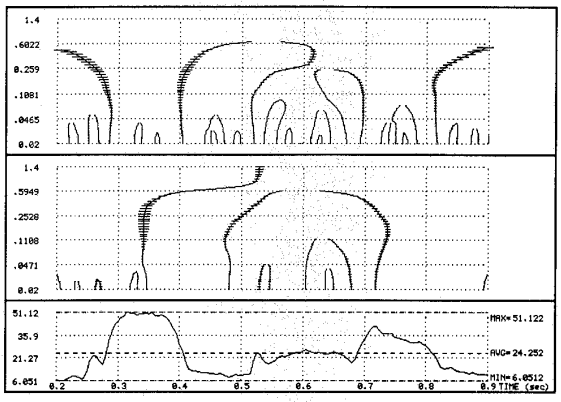

Fig. 4. Event analysis for the loudness signal of /kaksi/ filtered with Gaussian kernel; bottom: input; middle: blocks; top: edges. Horizontal lines around event-lines indicate prominence.

The effect of using kernel functions other than Gaussian was investigated. Figure 5 shows the results obtained when using a multi-scale derivative filter bank proposed by Lyon (see bottom of figure 2) that is known to be computationally very efficient. Here the events have been shifted to compensate for the increasing delay at larger scales introduced by the causal filtering. When compared to the ideal Gaussian case it can be seen that the location of the block and edge structures are quite similar at small scales. At larger scales however differences can be noted especially in which structures exist for the longest periods of scale. For example, the edge structures when linked to form *contours* result in different ones than those obtained with a Gaussian kernel and therefore a different ternary tree will result if constructed. The parsing rules that were developed with a Gaussian kernel worked well with this type of filtering and did not require modifications.

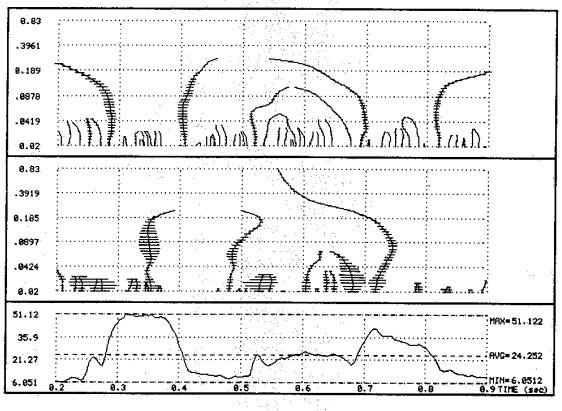

Fig. 5. Event analysis of /kaksi/ filtered with Lyon's Multi-scale derivative filter bank; bottom: input; middle: blocks; top: edges.

The effect of using the piecewise-linear kernel (top of figure 2) and the effects it causes in the analysis are shown in figure 6. The event parser attempted to create continuous structures but often failed since events did not behave systematically. The most priminent areas of line-events can be still recognized in most cases. Closer observation reveals a slight resemblance to figure 5, especially in which structures remain when scale is increased.

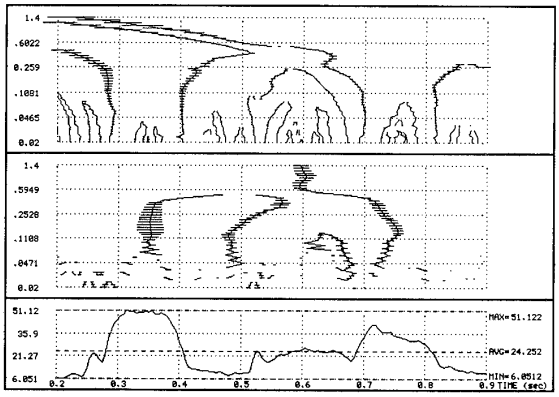

Fig. 6. Event analysis for /kaksi/ when filtered with the piecewise-linear kernel of Fig. 2; bottom: input; middle: blocks; top: edges.

Frequency Domain Results

The results of applying the analysis to the auditory spectrum of /i/ can be seen in figure 7. Blocks (one at 3 Barks, the other at 14 Barks) are shown as solid lines while edges are shown dotted. Negative blocks (spectral valleys) are not shown to avoid overcrowding the figure but they also provide useful information.

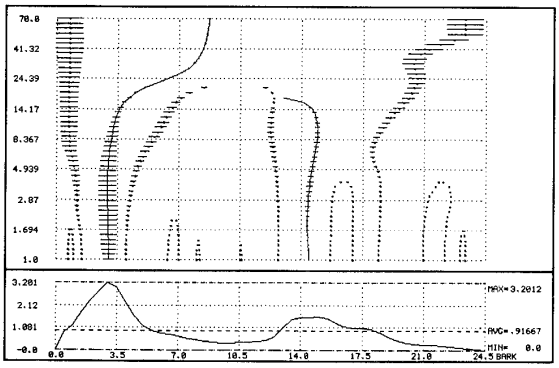

Fig. 7. Event analysis of auditory spectrum of /i/ using Gaussian kernel. Solid lines indicate formants, dotted lines indicate edges.

Each block corresponds to a formant or formant cluster. Formant F1 located at about 3.0 Barks and the related edges (1.5 and 3.5 Barks) can be interpreted as a composite-event (edge-block-edge). Formants F2, F3 and F4 are clustered so that at scale 5.0 their centre-of-gravity (position of the block line-event) is at 14.5 Barks and the main edges are at 12 and 18 Barks. This formant cluster corresponds to the 3.5 Bark formant integration range proposed by Chistovich [3]. The most prominent features of the spectrum can be represented by three main edge-block-edge type composite-events: the F1 range, the spectral valley, and the F2-F4 range. The details of these composite-events are still accessible by the method functions of these objects.

The same input was analyzed with Lyon's filter bank and is shown in figure 8. Formant blocks and their edges are in similar locations as in the Gaussian case although the gross structure of the line-events is different.

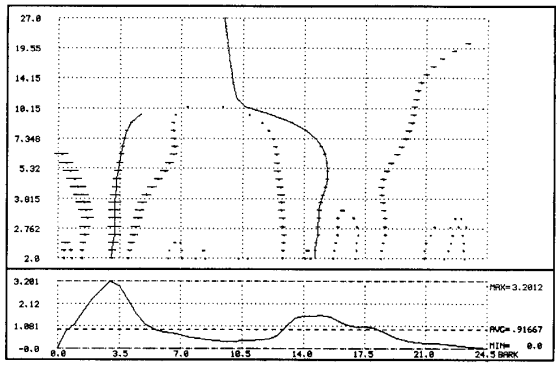

Fig. 8. Event analysis of auditory spectrum of /i/ using Lyon's kernel. Solid lines indicate formants, dotted lines indicate edges.

CONCLUSION

This paper has presented the methods of multiple-resolution analysis (similar to scale-space filtering) and object-oriented event-based representation of speech signals. The methods can be seen as an extension of peripheral auditory modeling and as an interface to phonetic and linguistic processing of speech signals. Thus, speech recognition is a natural and potential application for this methodology.

ACKNOWLEDGEMENTS

The authors are grateful to the Academy of Finland which has financed this research.

REFERENCES

[1] A. P. Witkin, "Scale-space filtering: A new approach to multi-scale description," Proc. of IEEE ICASSP-84, San Diego.

[2] R. F. Lyon, "Speech recognition in scale space," Proc. of IEEE ICASSP-87, Dallas.

[3] L. A. Chistovich, et al., "'Centres of gravity' and spectral peaks as the determinants of vowel quality," *Frontiers in Speech Communication Research*, Academic Press, 1979.

[4] H. Asada and M. Brady, "The curvature primal sketch," IEEE Trans. on Pattern Anal. Mach. Intell., vol. pami-8, no. 1, Jan. 1986, pp. 2-14.

[5] M. Karjalainen, T. Altosaar, and P. Alku, "QuickSig - An object-oriented signal processing environment" (this proceedings).

[6] J. Babaud, A. P. Witkin, M. Baudin, and R. O. Duda, "Uniqueness of the Gaussian kernel for scale-space filtering", IEEE Trans. on Pattern Anal. Mach. Intell., vol. pami-8, Jan. 1986, pp. 26-33.

[7] M. Karjalainen, "A new auditory model for the evaluation of sound quality of audio systems," Proc. of IEEE ICASSP-85, Tampa.

[8] M. Karjalainen, "Auditory models for speech processing," 11th ICPhS, Tallinn, 1987.

[9] E. Zwicker and R. Feldtkeller, *Das Ohr als Nachrictenempfänger*, S.Hirzel Verlag, Stuttgart, 1967.