# REAL-TIME CORPUS-BASED CONCATENATIVE SYNTHESIS WITH CATART

*Diemo Schwarz, Grégory Beller, Bruno Verbrugghe, Sam Britton*

Ircam – Centre Pompidou
1, place Igor-Stravinsky, 75003 Paris, France
http://www.ircam.fr/anasyn/schwarz
schwarz@ircam.fr

## ABSTRACT

The concatenative real-time sound synthesis system *CataRT* plays grains from a large corpus of segmented and descriptor-analysed sounds according to proximity to a target position in the descriptor space. This can be seen as a content-based extension to granular synthesis providing direct access to specific sound characteristics. *CataRT* is implemented as a collection of *Max/MSP* patches using the FTM library and an SQL database. Segmentation and MPEG-7 descriptors are loaded from SDIF files or generated on-the-fly. The object-oriented software architecture follows the model–view–controller design pattern. *CataRT* allows to explore the corpus interactively or via a target sequencer, to resynthesise an audio file or live input with the source sounds, or to experiment with expressive speech synthesis and gestural control.

## 1. INTRODUCTION

Corpus-based concatenative synthesis methods are attracting more and more interest in the musical sound synthesis and content-based processing communities. They use a large database of source sounds, segmented into *units*, and a *unit selection* algorithm that finds the sequence of units that match best the sound or phrase to be synthesised, called the *target*. The selection is performed according to the *descriptors* of the units, which are characteristics extracted from the source sounds, or higher level descriptors attributed to them. The selected units can then be transformed to fully match the target specification, and are concatenated.

These methods allow various applications, such as high level instrument synthesis, resynthesis of audio, also called *mosaicing*, texture and ambience synthesis, artistic speech synthesis, and interactive explorative synthesis in different variants, which is the main application of the *CataRT* synthesis system.

Explorative real-time synthesis from heterogeneous sound databases allows a sound composer to exploit the richness of detail of recorded sound while retaining efficient control of the acoustic result by using perceptually meaningful descriptors to specify a target in the multi-dimensional descriptor space. If the selection happens in real-time, this allows to browse and explore a corpus of sounds interactively.

The *CataRT* system is a collection of patches for *Max/MSP*[1] using the FTM, *Gabor*, and MnM extensions[2]. The sound and descriptor data can be loaded from SDIF files (see section 4.2) containing MPEG-7 descriptors, for instance, or can be calculated on-the-fly. It is then stored in FTM data structures in memory. *CataRT* is released as free open source software under the GNU general public license (GPL)[3].

After an overview of previous and related work in section 2, we present *CataRT*'s underlying model in section 3. The object-oriented software architecture is detailed in section 4, followed by the applications in section 5, and an outlook of future work in section 6 and the conclusion.

## 2. RELATED WORK

Corpus-based concatenative sound synthesis draws on many fields of research, mainly digital signal processing (analysis, synthesis, and matching), computer science (database technology), statistics and machine learning (classification), music information retrieval and modeling, and real-time interaction.

In addition, there are many other topics of research that share methods or objectives, such as speech synthesis, singing voice synthesis, and content-based processing [1].

We could see concatenative synthesis as one of three variants of content-based retrieval, depending on what is queried and how it is used. When just one sound is queried, we are in the realm of descriptor- or similarity-based sound selection. Superposing retrieved sounds to satisfy a certain outcome is the topic of automatic orchestration tools. Finally, sequencing retrieved sound snippets is our topic of concatenative synthesis.

### 2.1. Caterpillar

*Caterpillar*, first proposed in [2, 3] and described in detail in [4], performs non real-time data-driven concatenative musical sound synthesis from large heterogeneous sound databases.

Units are segmented by automatic alignment of music with its score for instrument corpora, and by blind segmentation for free and re-synthesis. The descriptors are based on the MPEG-7 low-level descriptor set, plus descriptors derived from the score and the sound class. The low-level descriptors are condensed to unit descriptors by modeling of their temporal evolution over the unit (mean value, slope, spectrum, etc.) The database is implemented using the relational database management system *PostGreSQL* for added reliability and flexibility.

The unit selection algorithm is a Viterbi path-search algorithm, which finds the globally optimal sequence of database units that best match the given target units using two cost functions: The *target cost* expresses the similarity of a target unit to the database units by weighted Euclidean distance, including a context around

[1] http://www.cycling74.com
[2] http://www.ircam.fr/ftm
[3] http://www.fsf.org

the target. The *concatenation cost* predicts the quality of the join of two database units by join-point continuity of selected descriptors.

Unit corpora of violin sounds, environmental noises, and speech have been built and used for a variety of sound examples of high-level synthesis and resynthesis of audio[4].

### 2.2. Talkapillar

The derived project *Talkapillar* [5] adapted the *Caterpillar* system for artistic text-to-speech synthesis by adding specialised phonetic and phonologic descriptors. The goal here is different from fully automatic text-to-speech synthesis: highest speech quality is needed (concerning both sound and expressiveness), manual refinement is allowed.

The role of *Talkapillar* is to give the highest possible automatic support for human decisions and synthesis control, and to select a number of well matching units in a very large base (obtained by automatic alignment) according to high level linguistic descriptors, which reliably predict the low-level acoustic characteristics of the speech units from their grammatical and prosodic context, and emotional and expressive descriptors.

In a further development, this system now allows hybrid concatenation between music and speech by mixing speech and music target specifications and databases, and is applicable to descriptor-driven or context-sensitive voice effects [6].[5]

### 2.3. Real-Time Concatenative Synthesis Systems

In the last few years, the number of research or development projects in concatenative synthesis exploded, so that a description of each approach would go largely beyond the range of this article. For an in-depth survey comparing and classifying the many different approaches to concatenative synthesis that exist, the kind reader is referred to [7]. We will nevertheless mention some systems with real-time capabilities: The first group of systems match low- or high-level descriptors of the corpus units to target descriptors (*MoSievius* [8], *Synful* [9], *Song Sampler* [10], *Ringomatic* [11]).

The second group performs a spectral match based on FFT-frames (*Input Driven Resynthesis* [12]), on beats in a drum loop (*Granuloop* [13]), or on spectro-temporal "sound lexemes" (*SoundSpotter* [14]).

### 2.4. Granular Synthesis

One source of inspiration of the present work is granular synthesis [15], which takes short snippets (*grains*) out of a sound file, at an arbitrary rate. These grains are played back with a possibly changed pitch, envelope, and volume. The position and length of the snippets are controlled interactively, allowing to scan through the soundfile, in any speed.

Granular synthesis is rudimentarily corpus-based, considering that there is no analysis, the unit size is determined arbitrarily, and the selection is limited to choosing the position in one single sound file. However, its concept of exploring a sound interactively, when combined with a pre-analysis of the data and thus enriched by a targeted selection, results in a precise control over the output sound characteristics, as realised in *CataRT*.

[4] http://www.ircam.fr/anasyn/schwarz/

[5] Examples can be heard on http://www.ircam.fr/anasyn/concat

## 3. MODEL

*CataRT*'s model is a multidimensional space of descriptors, populated by the sound units. The user controls a target point in a lower-dimensional projection of that space with a selection radius around it, and the selection algorithm selects the units closest to the target or within the radius. The actual triggering of the unit is independent of the selection and can happen at any rate.

Because displaying and navigating in a high-dimensional space is not practical, the descriptor space is reduced to a 2-dimensional projection according to two selectable descriptors.

The selection is considering closeness in a geometric sense, i.e. on appropriately scaled dimensions: The generic distance measure is a Euclidean distance on the two chosen descriptors, normalised over the corpus, i.e. a Mahalanobis distance, in order to avoid distortions between different distances because of the different ranges of the values.

No concatenation quality is considered, for the moment, and the only transformations applied are a short crossfade to smooth the concatenation and pitch and loudness changes.

The following data-flow diagrams illustrate *CataRT*'s model, boxes stand for data, circles for processes, and lozenges for real-time signals. Figure 3 shows an overview of real-time corpus-based synthesis with the audio input feeding the corpus, that can be persistently saved and loaded to a database, and synthesis by retrieving data from the corpus.

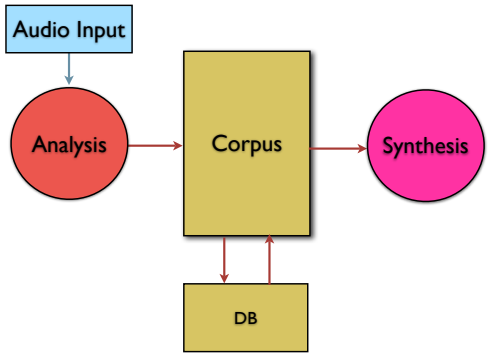


Figure 1: Overview of real-time corpus-based synthesis

The analysis part in figure 3 shows the different possibilities to get data into the corpus: either all data (audio, segment markers, raw descriptors) are loaded from preanalysed files, or the descriptors are analysed in *CataRT* but the segment markers come from a file (or are arbitrarily chosen), or an additional onset detection stage segments the sound files. At this point, all analysis takes place inside *CataRT* in real-time, which means that we could just as well use real-time audio input that is segmented into units and analysed on the fly, to feed the corpus. The audio could come, for example, from a musician on stage, the last several minutes of whose playing constitutes the corpus from which a laptop improviser selects units.

Last, figure 3 shows the use of the corpus for synthesis by selection by user-controlled nearest neighbour search, with subsequent transformation and concatenation of the selected units.

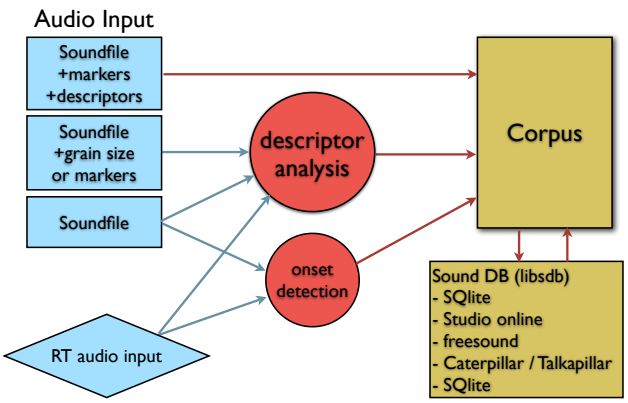


Figure 2: Analysis

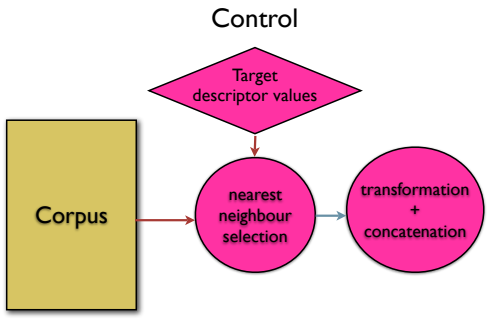


Figure 3: Synthesis

## 4. ARCHITECTURE

The architecture of the collection of *CataRT Max/MSP* patches is given in UML notation in figure 4 and explained in the following. A class corresponds here to a *Max/MSP* abstraction (a subpatch that can be instantiated several times), and an interface is simply a convention of messages that can be sent to a subpatch which is implementing it. The main application-dependent patch that holds everything together is not shown explicitly in the diagram.

### 4.1. Analysis

The segmentation of the source sound files into units can come from external files or be calculated internally. There is also a mode where imported sound files are taken as a whole, which is appropriate for sets of drum and percussion sounds. Markers generated externally can be loaded from SDIF or ASCII files, or be imported from the marker chunks in the AIFF or WAV soundfiles themselves. Internal segmentation calculation is either by arbitrary grain segmentation, by split according to silence (given a threshold), by high-frequency content [16]*, or by the transient analysis stage of the *SuperVP* phase vocoder, ported to *Max/MSP* [17, 18]. In any case, the markers can be viewed and edited by hand, if necessary.

* todo: comparison article

Descriptor analysis either uses precalculated MPEG-7 low-level descriptors or descriptors calculated in the patch. Details for the 230 imported MPEG-7 signal, perceptual, spectral, and harmonic descriptors can be found in [4], following the definitions from [19, 20].

The descriptors calculated in the patch in batch mode, i.e. faster than real-time, thanks to *Gabor*'s event-based signal frame processing, are the fundamental frequency, aperiodicity, and loudness found by the *yin* algorithm [21], and a number of spectral descriptors from [22]: spectral centroid, sharpness, spectral flatness, high frequency energy, mid frequency energy, high frequency content, first order autocorrelation coefficient (that expresses spectral tilt), and energy.

Note that also descriptors describing the units' segments themselves, such as the unit's unique id, the start and end time, its duration, and the soundfile it came from, are stored. Usually, this data is available directly in the data structures and the database tables, but, to make it available for selection, it is convenient to duplicate this information as descriptors.

The time-varying raw descriptors at FFT-frame rate have to be condensed to a fixed number of scalar values to characterise a unit. These *characteristic values* [4] express the general evolution over time of a descriptor with its mean value, variance, slope, curve, min, max, and range, and allow to efficiently query and select units.

### 4.2. Data

Data is kept in the following FTM data structures: A table contains in its rows the descriptor definitions with the name and the specification where to find this descriptor in an SDIF file (the frame and matrix signatures and matrix indices). The loaded soundfiles are kept in a dictionary indexed by file name, containing metadata, a list of dictionaries for the data files, an FTM event sequence with the segmentation marks, and a vector containing the sound data. The unit descriptor data is kept in one big (N, D) matrix with one column per descriptor and one unit per row.

Write access and the actual data storage is situated in catart.data, whereas read access to the data is provided by catart.data.proxy, which references an instance of the former, and which can be duplicated wherever data acess is needed.

For persistent storage of corpora, a layer around the relational database management system *SQLite* keeps track of soundfiles, segments, and unit descriptor data.

The Sound Description Interchange Format (SDIF)[6] is used for well-defined interchange of data with external analysis and segmentation programs.

### 4.3. Selection

Because of the real-time orientation of *CataRT*, we cannot use the globally optimal path-search style unit selection based on a Viterbi algorithm as in *Caterpillar*, neither do we consider concatenation quality, for the moment. Instead, the selection is based on finding the units closest to the current position $x$ in the descriptor space, in a geometric sense, i.e. on appropriately scaled dimensions: A straightforward way of achieving this is to calculate the square Mahalanobis distance $d$ between $x$ and all units with

$$d = \frac{(x - \mu)^2}{\sigma} \quad (1)$$

[6] http://www.ircam.fr/sdif

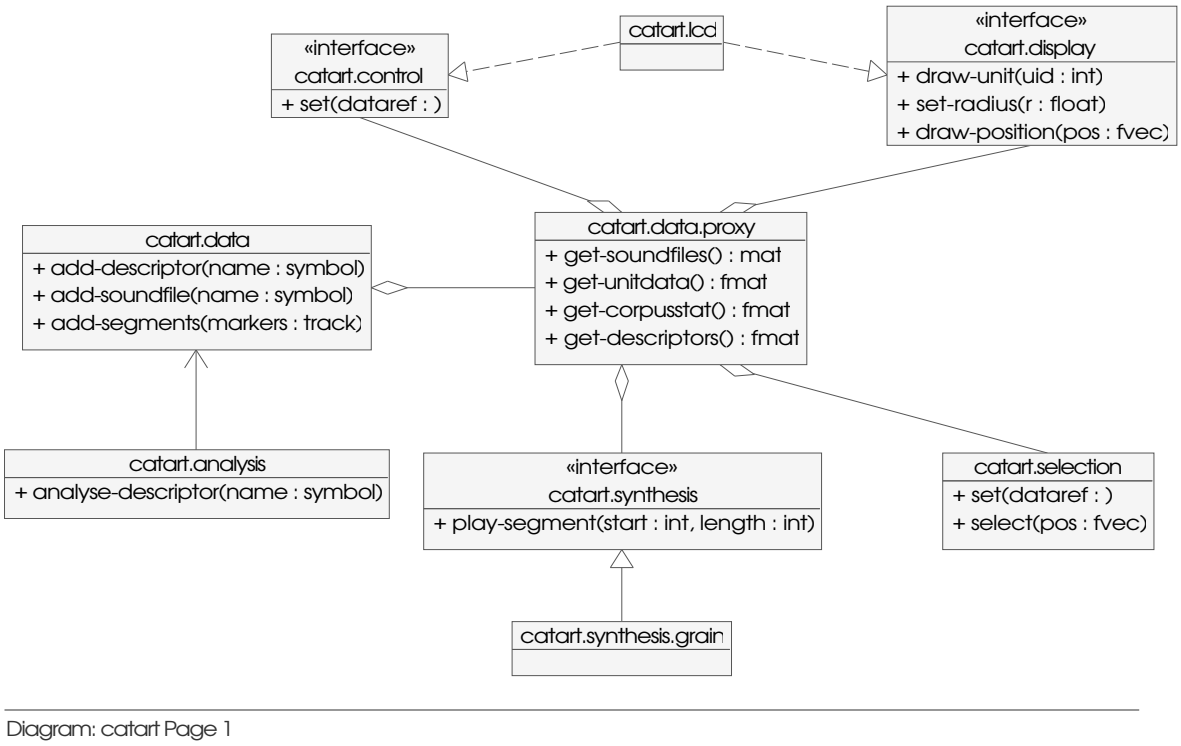


Figure 4: UML Diagram of *CataRT*'s Software Architecture

where $\mu$ is the (N, D) matrix of unit data and $\sigma$ the standard deviation of each descriptor over the corpus. Either the unit with minimal $d$ is selected, or one randomly chosen from the set of units with $d < r^2$, when a selection radius $r$ is specified.

To improve the efficiency of selection, the units in the descriptor space are indexed by an optimised multi-dimensional $k$-nearest neighbour index. The algorithm described in [23] constructs a search tree by splitting up the descriptor space along the hyperplane perpendicular to the principal component vector, and thus achieving a maximal separation of units. This is then repeated for each sub-space until only a few units are left in each leaf node of the resulting tree. The $k$-nearest neighbour search can then, at each step down the tree, eliminate approximately half of the units, by just one distance calculation with the subspace boundary.

### 4.4. Synthesis

*CataRT*'s standard synthesis component catart.synthesis.grain is based on the *Gabor* library's frame-based processing framework: A choosable short fade-in and fade-out is applied to the sound data of a selected unit, which is then pasted into the output delay-line buffer, possibly with a random delay. Other manipulations similar to a granular synthesis engine can be applied: the copied length of the sound data can be arbitrarily changed (de facto falsifying the selection criteria) to achieve granular-style effects or clouds of overlapping grains. Also, changes in pitch by resampling and loudness changes are possible. Note that, because the actual pitch and loudness values of a unit are known in its descriptors, it is possible to specify precise pitch and loudness values that are to be met by the transformation.

However, this granular synthesis component is only one possible realisation of the synthesis interface catart.synthesis in figure 4. Other components might store the sound data in spectral or additive sinusoidal representations for easier transformation and concatenation.

### 4.5. Interface

The user interface for *CataRT* follows the model–view–controller (MVC) design principle, the model being the data, selection, and synthesis components, and the view and controller being defined by the two abstract interfaces catart.display and catart.control. For many applications, these two are implemented by one single component, e.g. when the control takes place on the display, e.g. by moving the mouse. The view (figure 6) plots a 2-dimensional projection of the units in the descriptor space plus a 3rd descriptor being expressed on a colour scale. Note that that the display is dynamic, i.e. multiple views can be instantiated that can connect to the same data component, or one view can be switched between several data instances, i.e. corpora.

Implemented displays use *Max/MSP*'s graphic canvas (*lcd*, see figure 6), a java-controlled display, or an OpenGL display via the *Jitter*[7] graphics library, to plot a 2-dimensional projection of the units in the descriptor space plus a 3rddescriptor begin expressed on a color scale.

In these displays, the mouse serves to move the target point in the descriptor space. Additional control possibilities are MIDI input from fader boxes to set more than two descriptor target values and limit a selectable descriptor range, and advanced input devices for gestural control (see section 5.2).

Independent of the current position, several sources for triggering playing of the currently closest unit exist: an obvious but quite interesting mode plays a unit whenever the mouse moves. De-facto, the friction of the mouse provides an appropriate force-feedback, so that this mode is called *bow*. To avoid the strident repetitions of units, the mode *fence* plays a unit whenever a different unit becomes the closest one (named in homage to clattering a stick along a garden fence). The *beat* mode triggers units regularly via a metronome, and the *chain* mode triggers a new unit whenever the previous unit has finished playing.

*CataRT* incorporates a basic loop-sequencer that allows to automate the target descriptor control (see figure 5). Also the evolution of the weight for a descriptor can be sequenced (second curve in figure 5), such that at the desired times, the target descriptor value is enforced, while at others the selection is less dependent on this descriptor.

## 5. APPLICATIONS

### 5.1. Explorative Granular Synthesis

The principal application of *CataRT* is the interactive explorative synthesis from a sound corpus, based on musically meaningful descriptors. Here, granular synthesis is extended by a targeted selection according to the content of the sound base. One could see this as abolishing the temporal dimension of a sound file, and navigating through it based on content alone.

Usually, the units group around several clusters. With corpora with mixed sources, such as train and other environmental noises, voice, and synthetic sounds, interesting overlaps in the descriptor space occur and can be exploited. Figure 6 shows the *CataRT* main patch with an example of a corpus to explore.

### 5.2. Gesture-Controlled Synthesis

The present model of controlling the exploration of the sound space by mouse position is of course very basic and limited. That's why we're working on the integration of a general mapping layer for gestural control into *CataRT*. This allows more expressive musical interaction and tapping into more than just two dimensions by analysis of higher level gestural parameters from more advanced input devices such as graphics tablets. Clustering, rotation and warping of the descriptor space (by multi-dimensional scaling or magnifying-glass type transformations) maximises the efficiency of the interaction, leading to greater expressivity. Sub-spaces of units can be selected by navigation or according to the temporal evolution of the control gesture (attack, sustain, release phases can have their own sub-space to select units from). Note that each sub-space can have its own projection to the most pertinent descriptors therein.

[7] http://www.cycling74.com/products/jitter

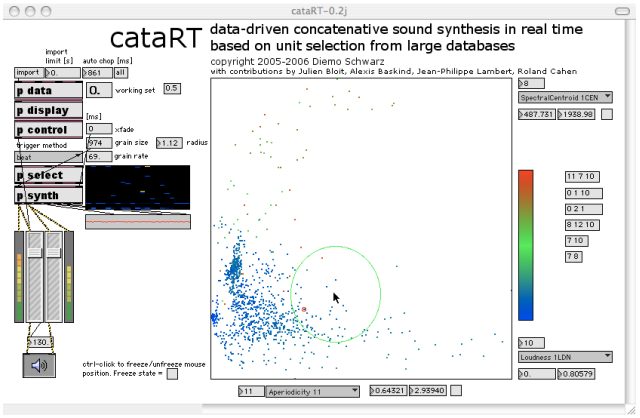


Figure 5: Screen shot of the explorative synthesis interface

### 5.3. Audio-Controlled Synthesis

As all the analysis can take place in *CataRT* itself, it is possible to analyse an incoming audio signal for descriptors and use these to control the synthesis, effectively resynthesising a signal with sounds from the database.

The target descriptor evolution can also be derived from a sound file, by analysing and segmenting it and playing its controlling descriptors from the sequencer.

*CataRT* is used as a compositional and orchestration tool in the context of a piece for banjo and electronics by Sam Britton. The work takes large databases of recorded instrumental improvisations and uses concatenative synthesis to re-sequence and orchestrate these sequences. In this context, the concatenation process acts as a kind of oral catalyst, experimentally re-combining events into harmonic, melodic and timbral structures, simultaneously proposing novel combinations and evolutions of the source material.

### 5.4. Data-Driven Drumbox

A slighly more rigid variation of the *CataRT* sequencer splits the target descriptor sequence into a fixed number of beats, where, for each beat, one sound class can be chosen. The selection within each soundclass, however, is gouverned by a freely editable descriptor curve, or real-time control.

### 5.5. Expressive Speech Synthesis

Research in expressive speech synthesis [24] takes advantage of real-time content-based manipulation and synthesis. Special descriptors are used for the speech units, e.g. acoustic descriptors like formant frequencies and pitch and symbolic descriptors such as the phonemes and the grammar. Here, a sentence is set up as a target sequence of successive segments of speech sounds and their descriptors. We can then change descriptor values of the target sequence to catch other sound units (speech or other) from the database. Figure 5 shows the pitch and the phonemic transcription of a sequence of units corresponding to the utterance "au revoir". It can be seen that we have graphically modified pitch around the temporal index shown by the cursor. All descriptors can be modified in such an easy way to change locally the selection of a unit

within a sequence. Text can also be rewritten using the keyboard as in the example where we have delayed the last "R".

There are two different ways to define temporal boundaries of the speech segments: The first is by segmental units like diphones, phones or semiphones as presented in figure 5, and leads to a small number of units per sentence which allows to use a many files to provide classical TTS with the capability to draw the prosody. The second is by pitch synchronous segmentation in periods of the speech signal. A large number of units per sentence are created which permits a fine grained selection. In this case, modifications of the descriptor sequence lead to a progressive morphing between spoken and singing synthesis, for instance. It is also possible to add jitter (perturbation of the pitch) to provide expressive speech synthesis. In this case, other descriptors describing voice quality are loaded in *CataRT*. Several descriptors will be added to give the possibility to draw the expressivity of the speech.

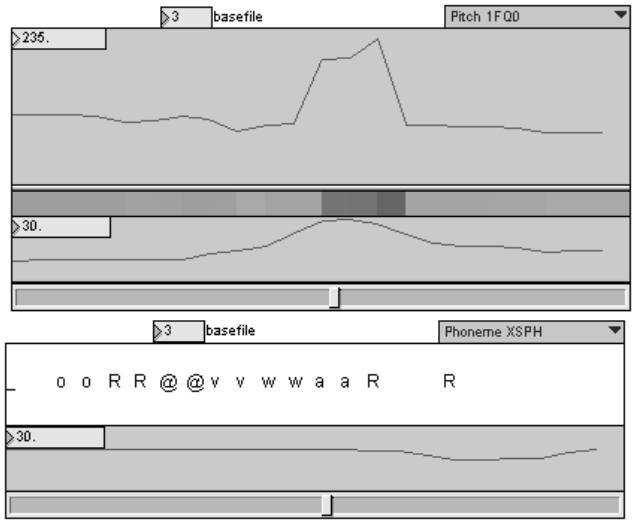


Figure 6: The *CataRT* descriptor sequencer

## 6. FUTURE WORK

Except the obvious work on adding more descriptors and improving the visualisation, interesting questions of control and interaction are raised by the present work.

The used corpora are in general unevenly distributed over the descriptor space. Many units are concentrated in clusters, whereas large parts of the space are relatively empty. This is first a problem of interaction and visualisation, which should allow zooming into a cluster to navigate through the fine differences within. However, the model of navigating through the descriptor space could be refined by a notion of subspaces with links to other subspaces. Note that, within different clusters, possibly different descriptors express best the intra-cluster variation.

*CataRT* should take care of concatenation, at least in a limited way, by considering the transition from the previously selected unit to the next one, not finding the globally optimal sequence as in *Caterpillar*. The concatenation cost could be given by descriptor continuity constraints, spectral distance measures, or by a precalculated distance matrix, which would also allow distances to be applied to symbolic descriptors such as phoneme class. The concatenation distance could be derived from an analysis of the corpus:

It should be possible to exploit the data in the corpus to analyse the natural behaviour of an underlying instrument or sound generation process. By modeling the probabilities to go from one cluster of units to the next, we would favour the typical articulations of the corpus, or, the synthesis left running freely would generate a sequence of units that recreates the texture of the source sounds.

To make more existing sound collections available to *CataRT*, an interface to the *Caterpillar* database, to the *freesound* repository, and other sound databases is planned. The *freesound* project,[8] is a collaboratively built up online database of samples under licensing terms less restrictive than the standard copyright, as provided by the *Creative Commons*[9] family of licenses. A transparent net access from *CataRT* to this sound database, with its 170 unit descriptors already calculated, would give us an endless supply of fresh sound material.

## 7. CONCLUSION

We presented the *CataRT* system that implements a new model of interactive exploration of, and navigation through a sound corpus. The concatenative synthesis approach is a natural extension of granular synthesis, augmented by content-based selection and control, but keeping the richness of the source sounds. By its real-time analysis capabilities, the system can also perform context-based effects. The applications are only beginning to fathom all the possibilities this model allows for interaction modes and visualisation. Because of their object-oriented software architecture, the *CataRT* components can serve as a toolbox to build individual applications.

## 8. REFERENCES

[1] Adam T. Lindsay, Alan P. Parkes, and Rosemary A. Fitzgerald, "Description-driven context-sensitive effects," in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*, London, UK, Sept. 2003.

[2] Diemo Schwarz, "A System for Data-Driven Concatenative Sound Synthesis," in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*, Verona, Italy, Dec. 2000, pp. 97–102.

[3] Diemo Schwarz, "The CATERPILLAR System for Data-Driven Concatenative Sound Synthesis," in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*, London, UK, Sept. 2003, pp. 135–140.

[4] Diemo Schwarz, *Data-Driven Concatenative Sound Synthesis*, Thèse de doctorat, Université Paris 6 – Pierre et Marie Curie, Paris, 2004.

[5] Grégory Beller, "Un synthétiseur vocal par sélection d'unités," Rapport de stage DEA ATIAM, Ircam – Centre Pompidou, Paris, France, 2004.

[6] Grégory Beller, Diemo Schwarz, Thomas Hueber, and Xavier Rodet, "A hybrid concatenative synthesis system on the intersection of music and speech," in *Journées d'Informatique Musicale (JIM)*, MSH Paris Nord, St. Denis, France, June 2005, pp. 41–45.

[8] http://iua-freesound.upf.es

[9] http://creativecommons.org

[7] Diemo Schwarz, "Concatenative sound synthesis: The early years," *Journal of New Music Research*, vol. 35, no. 1, pp. 3–22, Mar. 2006, Special Issue on Audio Mosaicing.

[8] Ari Lazier and Perry Cook, "MOSIEVIUS: Feature driven interactive audio mosaicing," in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*, London, UK, Sept. 2003, pp. 312–317.

[9] Eric Lindemann, "Musical synthesizer capable of expressive phrasing," US Patent 6,316,710, Nov. 2001.

[10] Jean-Julien Aucouturier, Franois Pachet, and Peter Hanappe, "From sound sampling to song sampling," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Barcelona, Spain, Oct. 2004.

[11] Jean-Julien Aucouturier and Franois Pachet, "Ringomatic: A Real-Time Interactive Drummer Using Constraint-Satisfaction and Drum Sound Descriptors," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, London, UK, 2005, pp. 412–419.

[12] Miller Puckette, "Low-Dimensional Parameter Mapping Using Spectral Envelopes," in *Proceedings of the International Computer Music Conference (ICMC)*, Miami, Florida, Nov. 2004, pp. 406–408.

[13] Pei Xiang, "A new scheme for real-time loop music production based on granular similarity and probability control," in *Digital Audio Effects (DAFx)*, Hamburg, Germany, Sept. 2002, pp. 89–92.

[14] Michael Casey, "Acoustic Lexemes for Real-Time Audio Mosaicing," in *Audio Mosaicing: Feature-Driven Audio Editing/Synthesis*, Adam T. Lindsay, Ed. International Computer Music Conference (ICMC) workshop, Barcelona, Spain, Sept. 2005, http://www.icmc2005.org/index.php?selectedPage=120

[15] Curtis Roads, "Introduction to granular synthesis," *Computer Music Journal*, vol. 12, no. 2, pp. 11–13, Summer 1988.

[16] Chris Duxbury, Juan Pablo Bello, Mike Davies, and Mark Sandler, "Complex domain onset detection for musical signals," in *Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx)*, London, UK, Sept. 2003.

[17] Axel Röbel, "A new approach to transient processing in the phase vocoder," in *6th International Conference on Digital Audio Effects (DAFx)*, London, United Kingdom, Septembre 2003, pp. 344–349.

[18] Axel Röbel, "Onset detection in polyphonic signals by means of transient peak classification," in *MIREX Online Proceedings (ISMIR 2005)*, London, Great Britain, Septembre 2005.

[19] Xavier Rodet and Patrice Tisserand, "ECRINS: Calcul des descripteurs bas niveaux," Tech. Rep., Ircam – Centre Pompidou, Paris, France, Oct. 2001.

[20] Geoffroy Peeters, "A large set of audio features for sound description (similarity and classification) in the Cuidado project," Tech. Rep. version 1.0, Ircam – Centre Pompidou, Apr. 2004.

[21] Alain de Cheveigné and Nathalie Henrich, "Fundamental Frequency Estimation of Musical Sounds," *Journal of the Acoustical Society of America (JASA)*, vol. 111, pp. 2416, 2002.

[22] Julien Bloit, "Analyse temps réel de la voix pour le contrôle de synthèse audio," Master-2/SAR ATIAM, UPMC (Paris 6), Paris, 2005.

[23] Wim D'haes, Dirk van Dyck, and Xavier Rodet, "PCA-based branch and bound search algorithms for computing $K$ nearest neighbors," *Pattern Recognition Letters*, vol. 24, no. 9–10, pp. 1437–1451, 2003.

[24] Grégory Beller, Thomas Hueber, Diemo Schwarz, and Xavier Rodet, "Speech rates in french expressive speech," in *Speech Prosody*, Dresden, Germany, 2006, pp. 672–675.