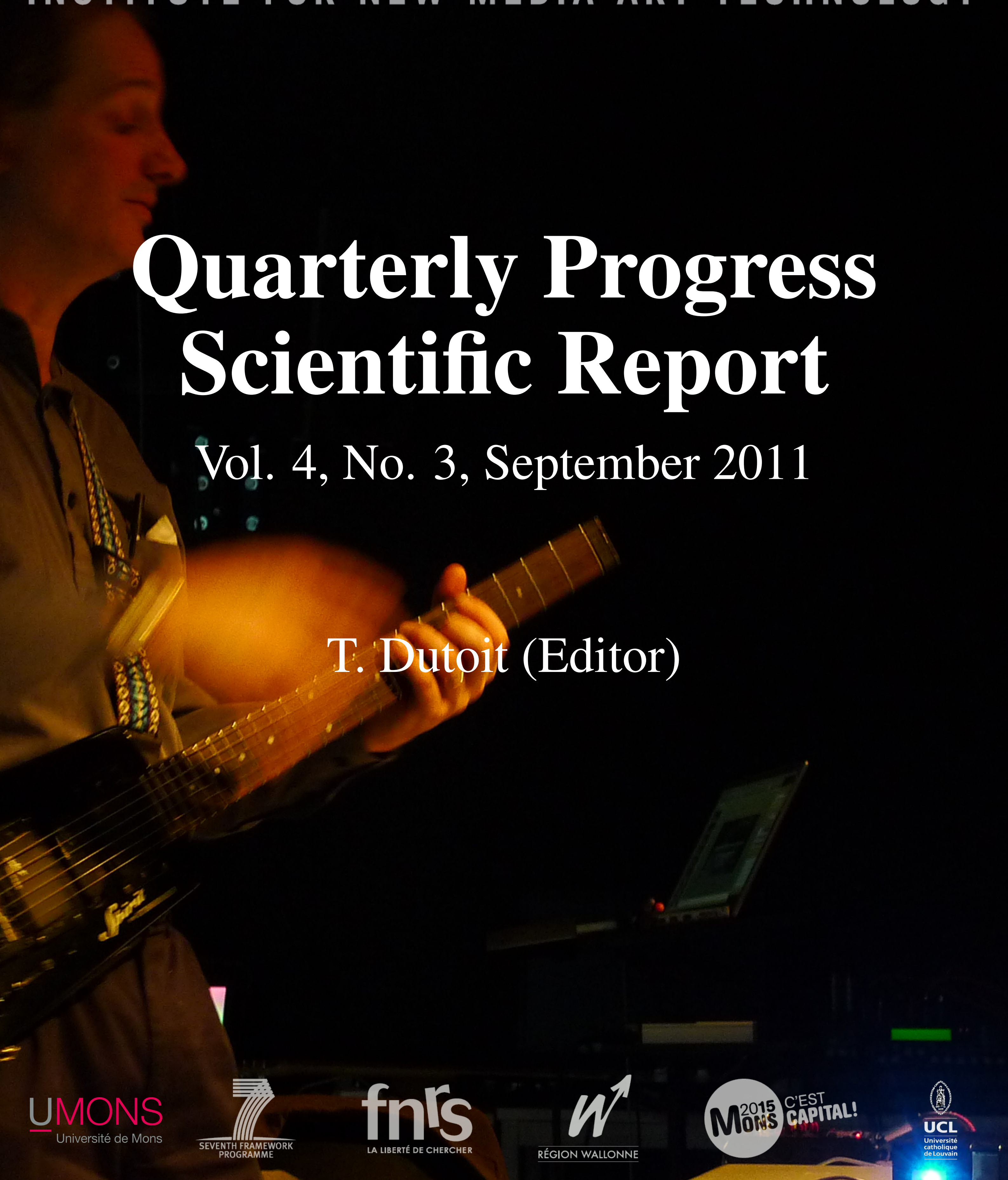

numediart

INSTITUTE FOR NEW MEDIA ART TECHNOLOGY

Quarterly Progress Scientific Report

Vol. 4, No. 3, September 2011

T. Dutoit (Editor)

UMONS Université de Mons

SEVENTH FRAMEWORK PROGRAMME

fnrs LA LIBERTÉ DE CHERCHER

RÉGION WALLONNE

MONS 2015 C'EST CAPITAL!

UCL Université catholique de Louvain

Published online by:
Université de Mons (UMONS)
Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS)
http://tcts.fpms.ac.be

ISSN:
2032-5398 (printed)
2032-538X (online)

Credits:
Editor: Thierry Dutoit (UMONS/TCTS)
LaTeX editor: Christian Frisson (UMONS/TCTS), using LaTeX's `confproc` class (by V. Verfaille)

`numediart` homepage: http://numediart.org
Contact: contact@numediart.org

Preface

The `numediart` Institute for New Media Art Technology was founded in 2010 by the University of Mons (UMONS), as an extension of the eponymous research programme funded by the Walloon Region (1997-2012). Building upon MONS 2015 dynamics (Mons will be the European Capital of Culture in 2015), the Institute shall organize internationally-renowned scientific training and research activities in the area of new media art technology. The topics covered by the Institute are the following: audio, image, video, gesture, and bio-signal processing, for applications in which the interaction between man and machine aims at creating emotions.

The activities of the Institute are coordinated by a strategic board, the `numediart` Consortium, which counts some fifteen members coming from the world of research, arts, entertainment, and industry. The Consortium meets every three months and ensures optimal adequacy between the research projects carried out by the Institute and the regional needs so as to foster scientific, economic and cultural development.

The `numediart` Programme is structured around three R&D themes: HyFORGE, for content-based hypermedia navigation; COMEDIA, for body and media interaction, and COPI, for digital luthery. The Programme consists of a series of short term projects (3-6-months; 3-4 projects running simultaneously), a one week mid-term "hands on" workshop, and a final public presentation. Projects proposals are made by senior researchers who carry them out and who are members of the Follow-up Committee, which meets once a month.

acapela
FISHING CACTUS
Multitel
CREACEED
TRANSCULTURES
UCL Université catholique de Louvain
alterface the interactive company
numediart INSTITUTE FOR NEW MEDIA ART TECHNOLOGY
NEUROTV
i-movix EXTREME SLOW MOTION
virtualis cross-media creativity
LE MANEGE MAUBEUGE MONS
ESAPV
DreamWall
Franco Dragone ENTERTAINMENT GROUP

This fifteenth session of numediart projects was held from July to September 2011.

NUMEDIART took part to the 1-month eNTERFACE'11 Workshop in Plzen, Tchequia, from August 1st to 26th.

UNIVERSITY OF WEST BOHEMIA

eNTERFACE'11

The 7th International Summer Workshop on Multimodal Interfaces

hosted by Department of Cybernetics, Faculty of Applied Sciences

Plzeň (Pilsen) / Czech Republic / 1. 8. - 26. 8. 2011

eNTERFACE'11

We took part to two of the 5 projects, respectively dedicated to collaborative vocal puppetry and to collaborative social gaming using a 3D camera.

Plzeň 2015
Evropské hlavní město kultury

eNTERFACE'11 was hosted by University of West Bohemia.

It was the first occasion for our teams from Mons to collaborate with researchers in Plzen, preparing MONS2015 ad PZLEN2015 (Mons and Plzen will both be EU capitals of culture in 2015). Other projects will follow...

Projects

GUITAR AS CONTROLLER

Loïc Reboursière [1], *Otso Lähdeoja* [1], *Ricardo Chesini Bose* [2],
Thomas Drugman [1], *Stéphane Dupont* [1], *Cécile Picard-Limpens* [1,3], *Nicolas Riche* [1]

[1] Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), Faculté Polytechnique de Mons (FPMs), Belgique
[2] Service Electronique et Microélectronique (SEMI), Faculté Polytechnique de Mons (FPMs), Belgique
[3] Groupe d'Intéraction Musicale (MINT), Haute École de Musique de Genève (HEMGE), Suisse

ABSTRACT

In this project we present a series of algorithms developed to detect the following guitar playing techniques : bend, hammer-on, pull-off, slide, palm muting, harmonic, plucking point. Via the detection of these information, the instrument can be extended as a natural way to control external content (i.e. loops, audio effects, videos, light changes, etc.). The guitar used is a Godin Multiac with an under-saddle RMC hexaphonic piezo pickup (one pickup per string, i.e six mono signals, one per string). Two exhaustive databases have been recorded by two different guitarists with different playing technics (finger style picking and plectrum style picking) to test the different algorithms.

Off-line implementation of the algorithms are showing good results. A global implementation on a dedicated software seems to be the solution to provide a more robust solution to real-time hexaphonic use. Implementation on FPGA have been investigated as well, in order to asses the problem of the big amount of processing power needed for hexaphonic detection.

KEYWORDS

Guitar audio analysis, playing techniques, hexaphonic pickup, controller.

1. INTRODUCTION

The guitar has maintained a close relationship with technological innovation throughout its history, from acoustic to electric and now to virtual [3]. Beyond the guitar, this virtuality has affected the way we play instruments in general. The appearance of the MIDI norm was a main event in the establishment of a communication between computers and traditional instruments. With an increased merging of the two, instruments became more and more interfaces to control computational sound processing or (multi-)media elements. The example of a MIDI keyboard controlling a sampler is one of the many examples that can be found for several years in electronic music.

The term "augmented instrument" is generally used for signifying a traditional (acoustic) instrument with added sonic possibilities. The augmentation can be physical like John Cage's sonic research on prepared pianos, but nowadays the term has acquired a more computational signification: the use of digital audio to enhance the sonic possibilities of a given instrument as well as the use of sensors and/or signal analysis algorithms to give extra and expressive controls to the player [16].

During this project, we focused on the latter kind of augmentation and more particularly on the audio signal analysis algorithms, in order to transform the guitar into a multi-media controller. We concentrated on the guitar's playing techniques as a source for controlling data, so that the guitarist's playing gestures could be used to control or trigger any external type of media or effects.

It has to be noticed here that our system could be compared to existing technologies like the Axon AX 50 [24] or the Roland VG-99 system [19]. These systems retrieve information, like onset detection and pitch-related data fitting the MIDI norm. Playing technics such as harmonics and palm-muted notes are absent as they don't have any translation in the MIDI norm. Based on a feature analysis of the raw audio from the guitar, our approach is broader and more complete than that of the MIDI, including all the major playing techniques used on the guitar.

2. SETUP

In order to work on the algorithms to detect the playing technics of a guitarist, we started out by recording two extensive databases played by two different guitar players. One was playing with a finger picking technique and the other one with a plectrum.

The guitar used for the recording was a Godin Multiac [8] nylon string guitar, equipped with custom RMC hexaphonic piezo pickups. Keith McMillen String Port [14] was used as the interface between the hexaphonic pickup and the computer and set up with a sampling frequency of 44 100Hz and 16 bits of depth. Seven channels of audio were used; one for each string and a summed 6-string mono channel.

Each one of the guitarists recorded twelve sets of data, covering all of the playing techniques detailed in section 3, with three levels of attack amplitude, including all the notes between the open strings and until the sixteenth fret.

Once the recordings were done, the segmentation part was achieved first using Matlab and the MIR Toolbox [15]. However, in the end, the most effective method was to put markers by hand using Sonic Visualizer software [11] and then use the time code of these markers to segment automatically the source files.

3. EXTRACTION OF PLAYING ARTICULATIONS

The articulations used to play the guitar can vary a lot from one guitarist to another as well as from one style to another. Jonathan Norton compiled a list of all the articulations that can be found in classical guitar playing [18] . The articulations we specifically worked on are listed and defined (using Norton's compiled definitions) below :

- **hammer-on**: firmly fret a note with left-hand finger without plucking the string with the right hand
- **pull-off**: after fretting a note with the left-hand finger, the same finger plucks the string with a downward motion

- **bend**: after fretting a note with the left-hand finger, the same finger pushes or pulls the string without lifting
- **slide / glissando**: after fretting a note with left-hand finger, the same finger slides to a new note without lifting
- **harmonic** (natural): Slightly fret a note on a node with the left hand, pluck the string with the right hand and then release
- **palm mute**: pluck the string while lightly touching the string with the side of the right palm near the bridge
- **plucking point**: the point where the right hand pluck the string. This articulation is equivalent to the term brightness used in Norton's list. The description is as follows : plucking near the bridge creates a brighter tone and near the fingerboard a warmer tone. In classical guitar litterature the traditional italian terminology is: *sul tasto* (on the fretboard) and *ponticello* (near the bridge).

Our project made a wide use of pitch detection in order to help characterizing most of the articulations cited above. It has to be pointed out that we did not develop any pitch detector algorithm, as this question is a fairly well-known problem in signal processing. The existing algorithms, may they be based on temporal or frequential techniques, gave good results on guitar signals. Therefore it didn't seem relevant to put our efforts in developing ours.

Our algorithmic approach of the articulation detection of a guitarist is described in figure 1. The diagram emphasizes the discrimination between right-hand and left-hand attack as the first important element leading to the detection of all the articulations. Pitch detection, on its side helps characterizing all the playing technics.

After explaining how we achieve onset detection (section 4) and right-hand / left-hand attack discrimination (section 4.3), the document will focus on detection of the left-hand articulations (section 5) and then on the right-hand ones (section 6). Another approach using the spectral modeling synthesis technics has been investigated; the results are shown in section 8. Real-time implementation using Max MSP software is discussed in section 9 and first steps on hardware implementation are described in section 10.

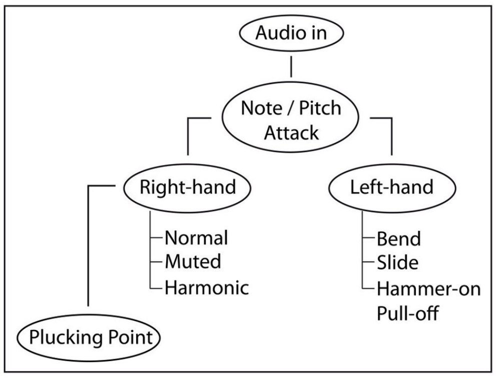

Figure 1: *Diagram of the articulations detection algorithm*

4. ONSET DETECTION

Onset detection is a specific problem within the area of music information retrieval, and can be the first step in a system designed to interpret an audio file in terms of its relevant musical events and its characteristics. Onset detectors can also be useful for triggering audio effects or other events. Due to its importance, it has been the focus of quantitative evaluations benchmarks, in particular through the Music Information Retrieval Evaluation eXchange (MIREX), where it has been present from 2005 onwards until 2011 [17].

As a general scheme for describing onset detection approaches, one can say that they consist in three steps [1]. First, the audio signal is pre-processed in order to accentuate certain aspects important to the detection task, including filtering, separation into frequency bands, or separation between transient and steady-state portions of the signal. Then, the amount of data from the processed signal is reduced in order to obtain a lower sample rate "onset detection function", generally where the onsets manifest themselves as peaks. Finally, as those detection functions are designed to produce high output values during musical onsets or abrupt events, thresholding and/or peak-picking can be applied to isolate the potential onsets. The following section will focus a bit more on the reduction function step. More information on onset detection can be found also in [2].

4.1. Reduction functions

In terms of reduction, a large range of approaches have been proposed, including those based on amplitude or energy envelopes (as well as their time derivatives, in the linear or log domains), short-time Fourier transforms, and probability models as well as machine learning.

Short-time Fourier transforms (STFTs), in particular the amplitude or power spectra, or the phase spectra, or else the whole complex spectra, have been used in multiple papers and systems. A well known approach consists of considering the high-frequency content of the signal by linearly weighting each bin in proportion to its frequency, hence emphasizing the high-frequencies typical to discontinuous portions, and especially onsets [12]. Different weighting functions have been proposed. A more general approach consists of considering changes in the spectrum and formulate detection functions as distances between consecutive STFTs, such as the so-called spectral flux approach when distance is computed between successive power spectra. Variants of these methods make use of different distance measures as well as rectification and smoothing schemes. Discontinuities in the evolution of the phase spectra, and more recently of the complex spectra [28], have also been showed beneficial for onset detection.

Several improvements to this category of approaches have been proposed earlier. Psychoacoustically motivated approaches such as taking into account the equal loudness contours [4] have proved to be of interest. In [5], variants are proposed in terms of frequency bins weighting and rectification. Adaptive whitening of the power spectra has also been tested successfully in [22] as a pre-processing step prior to computing distances between successive STFTs, providing for more robustness to various sound categories. Nonnegative matrix factorization has also been proposed and tested as a pre-processing step [29]. Finally, in [25], magnitude and phase spectra are considered also in combination to the use of pitch deviation.

The use of probability models has also been considered in the literature. These consist in inferring the likelihood of onsets given available observations as well as a model of the signal. Supervised machine learning approaches (including those based on gaussian mixture models, support vector machines, artificial neural networks) can also be used, and have actually been shown to outperform more traditional reduction approaches [10], although they are expected to fail when applied on signal or instrument categories that have not been taken into account during the training of the detector.

4.2. Evaluation

As mentioned above, evaluation and the collection of ground truth data are important issue [23, 13]. Evaluation campaigns with published results have been carried out, and a few publicly available corpora can be found too, for instance in [25], where the evaluation compares algorithms on range of musical material categories, namely bowed string instruments, wind instruments, complex mixed music, and pitched percussive instruments, the later actually including acoustic guitar sequences in those evaluations. Such pitched percussive sounds, similar to percussive instruments, constitute the least difficult category for onset detection, with a published F-Measure of 90% using the best configuration (in [25]), while for complex mixtures and wind instruments, the figure goes down to 80%, and for bowed strings, it drops to a F-Measure of 76%.

Here, we have been comparing a range of approaches on the guitar database collected for this project. Evaluation is performed using the approach proposed in [25] and with a tolerance of plus or minus 50 ms on the timing of the detected onset to still be considered as valid when compared to the reference (hand annotated) onset. Such a large tolerance value is necessary here essentially because the manual annotation used as a reference is not always very accurate in terms of timing. Practically however, the methods we compare here are more accurate with detection timing error generally below half of an analysis frame (hence below 12 ms in our case).

The method we have been comparing cover a range of 18 variants of amplitude-based methods (including in the energy, log-energy domains, and their time derivatives) and short-time Fourier transforms based methods (including spectral flux in different domains, phase-based methods and their variants using amplitude weighting, and complex-based methods). We have not been using methods requiring a pitch estimation.

We used the monophonic recordings of the guitar signals, and optimized the detection threshold for peak F-measure for each detection method individually. Table 1 presents the results for the four best performing methods, and provide the peak F-measure on the whole data set, as well as recall values for 4 categories of attacks: right-hand attack, harmonic, hammer-on, pull-off.

The four best performing methods on this data were:

- A spectral flux approach operating on the amplitude bins of the STFTs
- A phase divergence based approach using weighting according the the STFTs bin amplitudes
- Two methods simply making use of the amplitude of the signal along time

We observe however that if the detection of right hand (including with harmonics) attacks is generally not a problem, the detection of hammer-on and pull-off attacks is better achieved using STFTs-based methods. Indeed, a significant part of those left-hand attacks do not show any amplitude increase at all. Finally, the best performing approach has an F-measure above 96% with a recall close to 100% for right-hand attacks, 99% for hammer-ons, and a moderate 88% for pull-offs. Some further research would hence be necessary to understand how pull-offs can be better detected (without having recourse to pitch information).

4.3. Discrimination between left and right hand attacks

Once an onset has been detected, it is necessary to determine whether the attack was produced by the right or by the left hand. The right hand produces "normal" plucked attacks (finger or pick), (which can be further categorized as muted, bridge, soundhole, and fretboard attacks). The left hand produces two types of "legato" attacks: hammer-ons and pull-offs.

We experimented a method for determining right hand attacks by detecting characteristic signs of finger/pick attacks in the waveform. As the string is plucked, the vibratory regime of the string is momentarily stopped, resulting in a trough in the signal envelope 2. The trough is shorter for pick attack than for finger attacks, but it is still well apparent in the graphical waveform presentation.

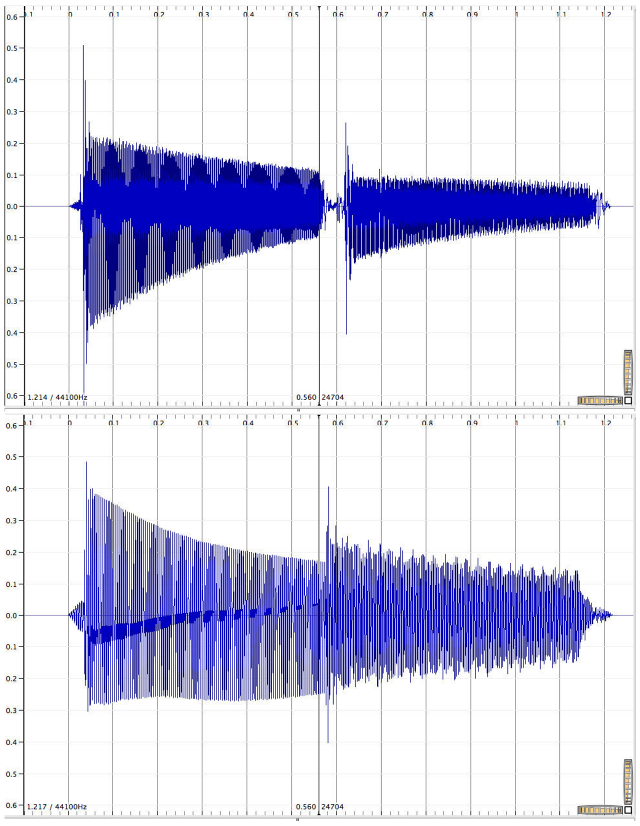

Figure 2: *On the top, two notes played with no left-hand articulations. One can see the trough in the amplitude between the two notes. On the bottom, the first note is played normally and the second one with a hammer-on. No trough is present*

.

The detection of a "gap" in the signal envelope is adjacent to the detection of an onset. The right hand attack criteria is thus double; an onset preceded by a short trough in the signal.

A method of "gap" detection was implemented in Max/MSP, based on a time-domain envelope follower. The application continuously samples 100ms windows of signal envelope data. When an onset is detected, the ongoing window is polled in order to detect a "gap", i.e. series of values below a fixed threshold. The

Method	F-measure	Right-hand Recall	Harmonic Recall	Hammer-on Recall	Pull-off Recall
Spectral Flux	96.2%	99.6%	100%	97.9%	88.0%
Weighted Phase Divergence	95.8%	98.9%	100%	97.4%	87.6%
Amplitude	92.4%	99.6%	100%	92.3%	75.2%
Delta Amplitude	91.8%	99.6%	100%	91.9%	74.4%

Table 1: Results of the onset detection with four different techniques

method is fairly robust, but has the downsides of being prone to noise (accidents in the signal) and non-operational with fast playing (the 100ms window is too large, and smaller windows produce less reliable results).

A reliable detection of right hand attacks opens the door for left hand attack analysis. Once the attack is detected as not being a right hand attack, any derivation in pitch is interpreted as a left hand articulation. It is then necessary to distinguish between the left hand techniques: hammer-on, pull-off, slide and bend.

5. LEFT-HAND ARTICULATIONS

Once the distinction between right-hand and left-hand attack is performed, it is possible to categorize the left-hand articulations by looking into the pitch profile of the note. The left hand works on the string tension and fretting, thus affecting the pitch. Our method of left-hand playing technique detection operates by measuring the pitch derivation in time.

Hammer-on (ascending legato) is characterized by an abrupt change in pitch, as well as its counterpart, the pull-off (descending legato). The bend shows a slower derivation in pitch. The slide has a pitch derivation similar to the hammer-on or pull-off, but with a "staircase" profile corresponding to the frets over which the sliding finger passes.

As a first step, two parameters are investigated: the number of half tones of the note transition, and the maximal relative pitch slope, defined as the maximal difference of pitch between two consecutive frames spaced by 10ms divided by the initial pitch value (at the first note). Figure 3 shows how these two parameters are displayed for notes articulated with a hammer, a pulloff, a bend or a slide. It can be noted from this figure that the great majority of bended notes are easily identified as they present a very low pitch slope. Secondly, it can be observed that for transitions with more than one half tone, a perfect determination of slides versus hammers or pulloffs is achieved. As a consequence, the only remaining ambiguity concerns the distinction of slides with hammers/pulloffs for a transition of *a half tone*.

To address this latter issue, two parameters are extracted: the energy ratio and the spectral center of gravity ratio. Both of them are computed on two 40ms-long frames: one ends at the transition middle, while the other starts at that moment. The distribution of these two parameters is shown in Figures 4 and 5 for one half tone of transitions respectively for hammers vs slides, and pulloffs vs slides.

Based on the aforementioned approaches, a detection of left-hand articulated notes has been proposed simply by using a thresholding. The results of the ensuing classification are presented in Table 2. It can be noticed that all bend effetcs are correctly identified. Slides are determined with an accuracy of 97.61%. Finally, hammers and pulloffs are detected in more than 93%. The main source of errors for these latter effetcs is the ambiguity with slides of one half tone.

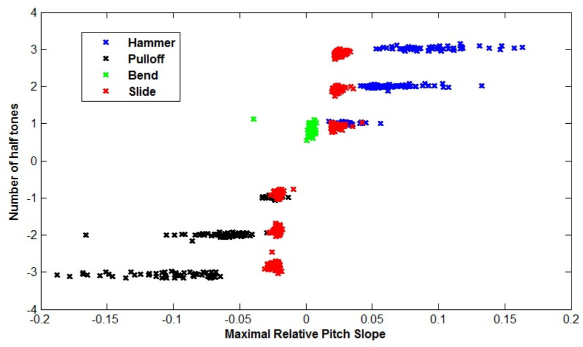

Figure 3: *Distribution of the number of transition half tones with the maximal relative slopes for notes with several effects of left-hand articulation: hammers, pulloffs, bends and slides.*

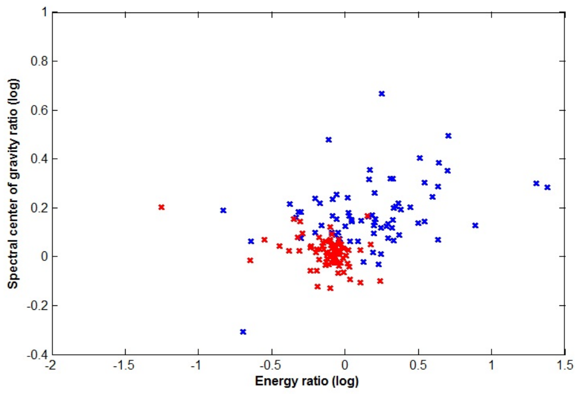

Figure 4: *Distribution of the energy ratio of one half-tone hammer-on and slide notes.*

6. RIGHT-HAND ARTICULATIONS

In this section, two right-hand articulations are studied: palm muting and harmonics notes.

6.1. Palm Muting

A palm muted note stifles the produced sound, which, as a consequence, lacks high frequencies. A frequential approach has then been used to detect this type of articulations.

First, an audio waveform is transformed by using a Discrete Fourier Transform which is a decomposition of the energy of a signal along frequencies. Moreover, we know that human ear, for frequencies lower than 1 kHz, hears tones with a linear scale instead of logarithmic scale for the frequencies higher that 1 kHz. Thus, we decide to use the Mel-frequency scale and separate the

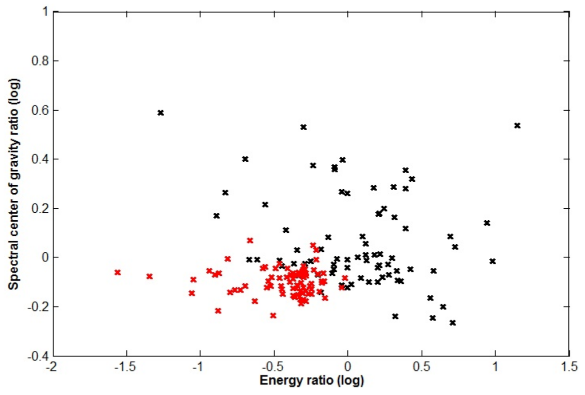

Figure 5: *Distribution of the energy ratio of one half-tone pull-off and slide notes.*

	Hammer-on	**Pull-off**	**Bend**	**Slide**
Hammer-on	93.27%	0.45%	0%	6.28%
Pull-off	0%	93.69%	0%	6.31%
Bend	0%	0%	100%	0%
Slide	1.74%	0.43%	0.21%	97.61%

Table 2: *Confusion matrix for detection of left-hand articulated notes*

frequency into 40 bands. It uses a linear scale below 1 kHz and a logarithmic one above 1 kHz. (Eq. 1). Figure 6 shows the Mel-spectra of a normal note and of a muted note. We calculated the area under the curve as the difference is rather obvious between the two types of notes.

$$mel(f) = 2595 \times \log(1 + \frac{f}{700}) \quad (1)$$

For each string a specific threshold has been set. Results are listed in Tab. 3 and Tab. 4. Most of them are equal or above 87.5%. However, we can observe a more difficult detection on the first string. It also appears that the bridge notes are closer to muted notes than other notes.

Audio Files	Muted Detect String 1 Threshold: 61	Muted Detect String 2 Threshold: 50	Muted Detect String 3 Threshold: 56
Bridge	56%	0%	12.5%
Sound Hole	0%	0%	0%
Neck	0%	0%	0%
Muted	75%	93.75%	100%

Table 3: *Palm muting detection results (string 1 to 3)*

6.2. Harmonics

The goal of this section is to distinguish when notes are played as harmonics. Two techniques are investigated for achieving this detection. One operates in the time domain, while the other focuses on the spectral domain. Both approaches only consider the note attack, which allows for a real-time detection of harmonic notes. It

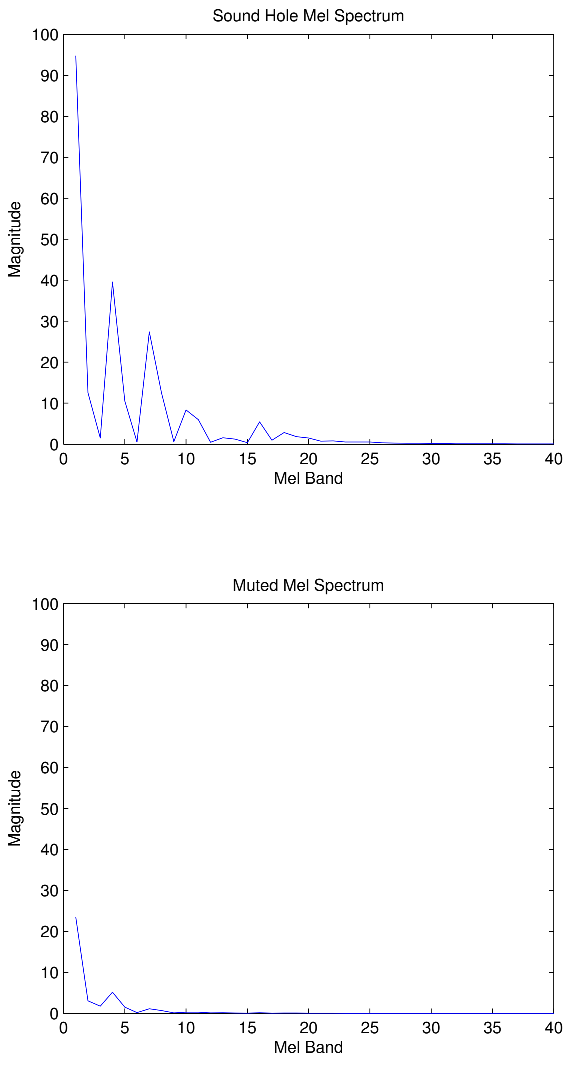

Figure 6: *Comparisons between Mel spectra of a normal note and a muted note*

has been attested that, after the attack, the differentiation between an harmonic and a normal note might become more difficult, especially for the 7^{th} and 12^{th} fret. The two methods are explained in the following.

- **Time-domain approach:** Figures 7 and 8 show the temporal shape of the waveform at the attack respectively for an harmonic and a normal note. From the attack waveform, two parameters are proposed: the attack duration is defined as the timespan during which the attack waveform remains positive; the relative discontinuity during the attack is defined as $Amin/Amax$ where these two latter amplitudes are as displayed in Figure 7.

  Figure 9 exhibits the distribution of these two parameters

Audio Files	Muted Detect String 4 Threshold: 65	Muted Detect String 5 Threshold: 83	Muted Detect String 6 Threshold: 67
Bridge	12.5%	6.25%	0%
Sound Hole	0%	0%	0%
Neck	0%	6.25%	12.5%
Muted	100%	87.5%	87.5%

Table 4: *Palm muting detection results (string 4 to 6)*

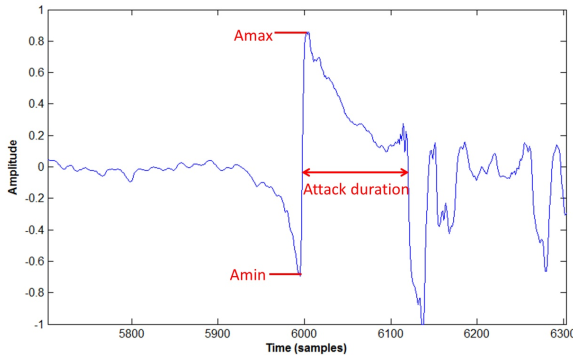

Figure 7: *Attack during the production of an harmonic note. The attack duration and the amplitude parameters for defining the relative discontinuity are also indicated.*

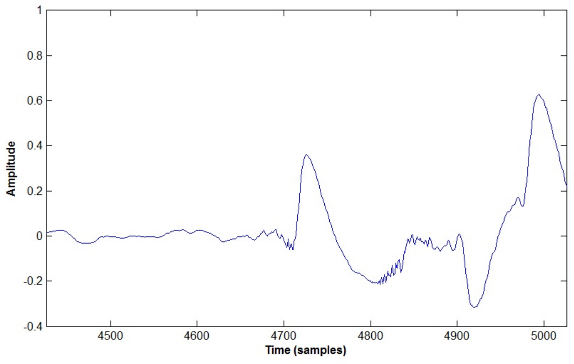

Figure 8: *Attack during the production of a normal note.*

for harmonic notes, as well as for normal notes played at the soundhole, at the bridge, or at the fretboard. As expected from the observation of Figures 7 and 8, an harmonic note is generally characterized by a larger attack duration, and a more important relative discontinuity (in absolute value).

- **Frequency-domain approach:** Figures 10 and 11 show the magnitude spectrum during the attack of respectively an harmonic and a normal note. A 40ms-long Hanning window starting at the attack has been used. On these spectra, a single parameter is extracted: the harmonic-to-subharmonic ratio is defined as the difference in dB between the amplitude of the first harmonic (at $F0$) and the amplitude of the subharmonic at $1.5 \cdot F0$.

Figure 12 exhibits the distribution of the harmonic-to-subharmonic ratio with the pitch value ($F0$) for harmonic notes, as well as for normal notes played at the sound hole, at the bridge, or at the fret board. As expected from the observation of Figures 10 and 11, an harmonic note is generally characterized, for a given pitch value, by a lower harmonic-to-subharmonic ratio.

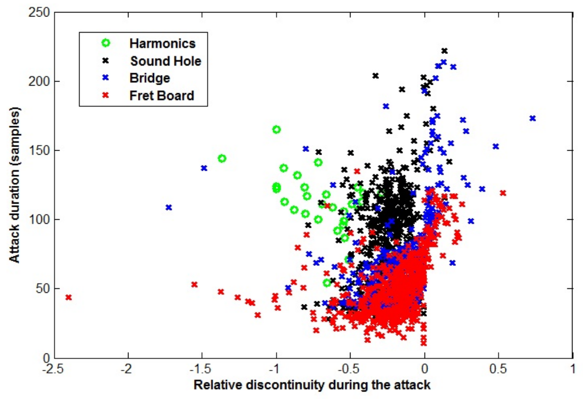

Figure 9: *Distribution of the relative discontinuity during the attack for harmonics compared to normal notes (played at the sound hole, the bridge and the fretboard.*

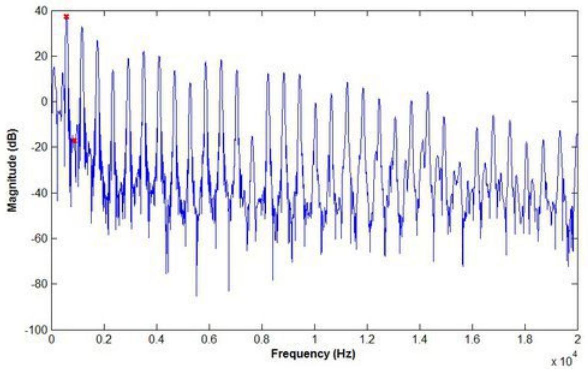

Figure 10: *Magnitude spectrum during the attack of an harmonic note.*

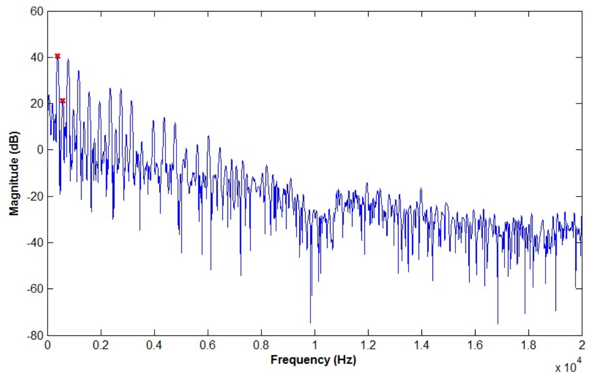

Figure 11: *Magnitude spectrum during the attack of a normal note.*

Based on the two previous approaches, a detection of harmonic notes has been proposed simply by using a thresholding. The results of the ensuing classification are presented in Table

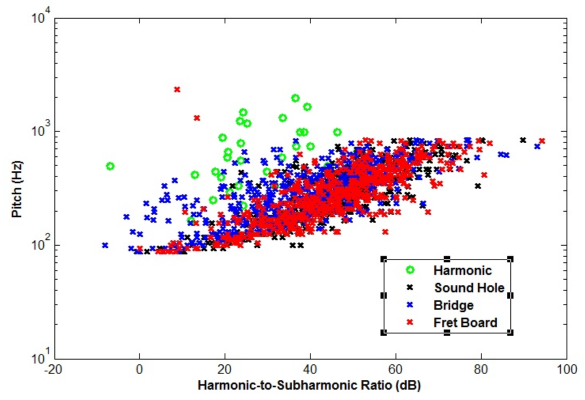

Figure 12: *Distribution of the pitch value and the harmonic-to-subharmonic ratio for harmonics and normal notes played at the sound hole, at the bridge and at the fret board.*

5. It can be noticed that all harmonic notes are correctly identified, while for normal notes, the best results are achieved for notes played on the fretboard (with only 0.52% of misclassification) and the worst ones are obtained for notes played at the bridge (with a bit less of 95% of correct detection).

	Harmonic detection	Normal detection
Harmonic	100%	0%
Fretboard	0.52%	99.48%
Soundhole	2.78%	97.22%
Bridge	5.38%	94.62%

Table 5: *Confusion matrix for detection of harmonics.*

7. PLUCKING POINT

The plucking point is a continuous parameter unlike most of the articulations we focused on. Its value can change during playing while the detection of a hammer-on, e.g, is a triggering value. However, it can as well be used to trigger elements when the string is plucked in pre-defined regions all along the strings.

7.1. State of the art

Two different approaches are present in the literature to asses the question of estimating the plucking point position. These two approaches are based on two main physical facts that happen when a string is plucked:

- The spectrum lacks harmonics that have a node at the plucking point
- Two impulses / waves run through the string in opposite directions

The first physical fact has been used in [26] to approximate the parameters of a comb filter delay as the power spectrum of the signal is similar to such shape. The calculation of the log-autocorrelation is used as a first step approximation. An iterative process in the weighted least-square estimation is then needed to refine the plucking point position. It has to be noticed here that a microphone has been placed in front of the soundhole of the guitar to record this database.

The second physical fact has been used in [9]. Here, as opposed to the previous method, an under-saddle pickup has been used. In this method, the minimum lag of the autocorrelation of the signal right after the attack is used to calculate the plucking position. A time-domain approach has also been used in the Axon software, where the user can defined three zones going from the bridge to the beginning of the fretboard.

7.2. Algorithm

The time difference between the opposite generated impulses has a consequence in the very few ms after the attack. Figure 13 shows typical waveforms (one for each position: bridge, normal, fretboard) of a one and a half period window starting at the attack. Depending on where the string has been plucked, the position of the minimum of the window appears closer to the attack as the string is plucked further from the bridge.

Our algorithm is based on the detection of the minimum of the window and on the calculation of the distance between the period and the minimum position. The process of the algorithm is going through the following steps:

- Pitch detection and calculation of the period
- Attack detection
- Window of 1,5 period from the attack
- Minimum detection in the window
- Distance between minimum and period
- Normalization regarding to the pitch so that the distance is independent of the fretted note

7.3. Evaluation

The algorithm has been tested on all the samples of both databases (finger picking and plectrum). Figure 14 shows the results of the estimation of the plucking point on the plectrum database. The displayed distance is a normalized distance, 0 corresponding to the bridge itself, 0.5 to half of the string length (the twelfth fret). It has to be mentioned here, that results from the finger picking database are nearly in the same order with a bit more overlap between regions.

	$d < 0.1$	$0.1 < d < 0.233$	$d > 0.233$
Bridge	98%	2%	0%
Soundhole	4%	96%	0%
Fretboard	0%	2%	98%

Table 6: *Results of plucking point zones detection.* ***d*** *represents the value of the calculated distance.*

Table 6 shows the results of zones detection when fixing a double threshold : one at 0.1 and the other at 0.233. The clear distinction seen on figure 14 between the fretboard area (d>0.233) and the two others is confirmed by the percentage as none of the bridge or soundhole audio samples has a distance superior to 0.233. However, 2% of the fretboard files are not well categorized.

The discrimination between the bridge and the soundhole areas is good even if figure 14 wasn't that clear about it.

Some of the errors may come from the hand of the guitarist not staying precisely at the same position during the recording as no marked positions were used for this experiment.

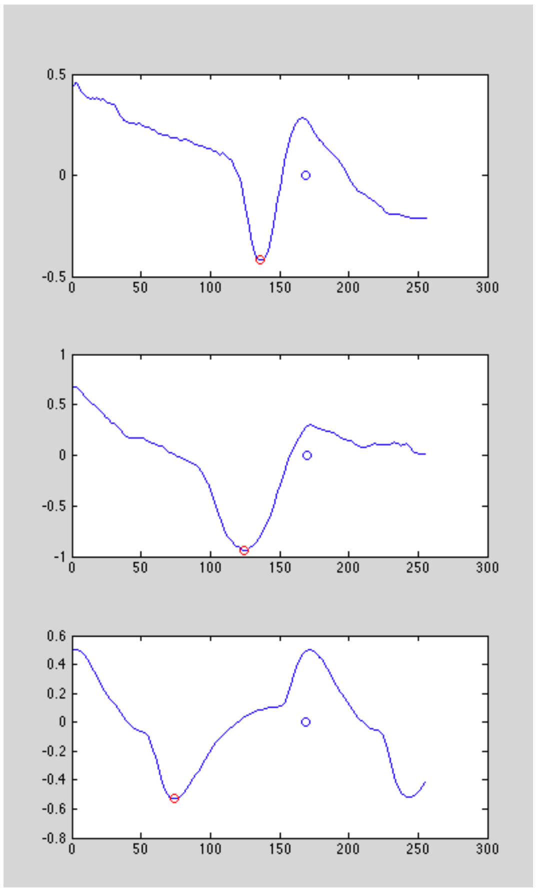

Figure 13: *One and a half period window from the attack for three different pluck positions : bridge (approx. 2cm), soundhole (approx. 10cm), fretboard (approx. 23cm)*

8. SPECTRAL MODELING SYNTHESIS

Another part of the research to build the detection algorithms focused on the Spectral Modeling Synthesis (SMS) method. This method models sounds by their time-varying spectral characteristics. Our approach is based on the original method [20], but also on the recent improvements towards real-time implementation in Pure Data with an on-the-fly analysis [6].

8.1. Method

Spectrum modeling can be used to extract parameters out of real sounds. The Sinusoids (or Deterministic) Plus Noise (or Stochastic) model allows more useful parameters than a simple short-time Fourier analysis. In addition to offering an easy way to modify actual sounds [21, 6], this model is meaningful: sinusoidal, or deterministic, component normally corresponds to the main modes of vibration of the system, whereas residual component corresponds to the energy produced by the excitation mechanism which is not transformed by the system into stationary vibrations. In the case of plucked strings, the stable sinusoids are the result of the main modes of vibration of the strings, whereas the noise is generated by the plucking gesture on the string, but also by other non-linear behavior of the finger-string resonator system. We thus assume that playing mode characteristics emerge in the stochastic part of

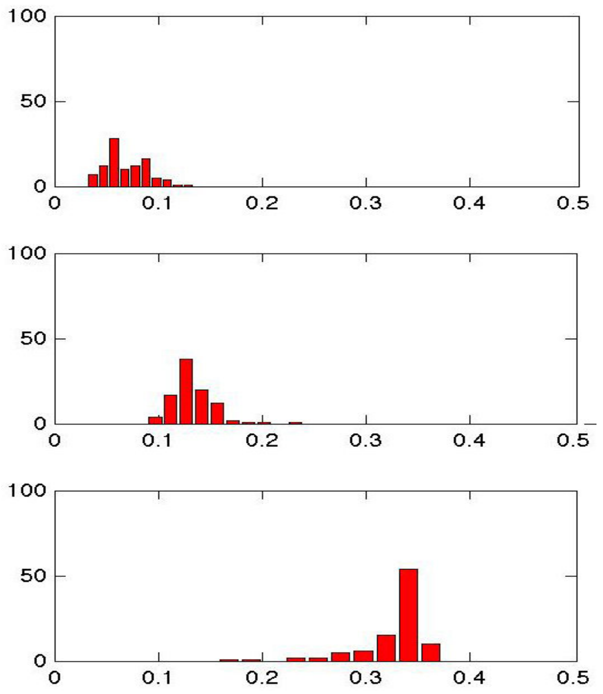

Figure 14: *Histograms of the distance calculation with the plectrum database - Estimation of the plucking point for the three positions: bridge (approx. 2cm), soundhole (approx. 10cm), fretboard (approx. 23cm) - Red circles represent the detected minimum and blue circles place the period.*

the sound.

The analysis starts with the extraction of the prominent peaks from a spectrum frame [20]. This is performed through a peak continuation algorithm that detects the magnitude, frequency and phase of the partials present in the original sound. This results in the deterministic component of the considered sound signal. A pitch detection step can improve the analysis by using the fundamental frequency information in the peak continuation algorithm and in choosing the size of the analysis window (pitch-synchronous analysis). The stochastic component of the current frame is calculated by subtracting from the original waveform in the time domain the deterministic signal obtained with additive synthesis. The stochastic part is then obtained by performing a spectral fitting of the residual signal.

In our approach, we focus on the stochastic part of the signal since we assume it holds the main characteristics of the playing modes. We thus synthesize the stochastic part generating a noise signal with the time-varying spectral shape obtained in the analysis. This signal will be further studied to extract prominent features. The pre-analysis with SMS is also a way to consider the playing modes independently of the played note since the deterministic component has been removed.

8.2. Results

We perform the SMS analysis using the *libsms* [6] on the recorded database. Except for the harmonic mode, all the recordings consist of one note plucked and the second note played in the specific playing technique. Parameters for analysis are the ones used for

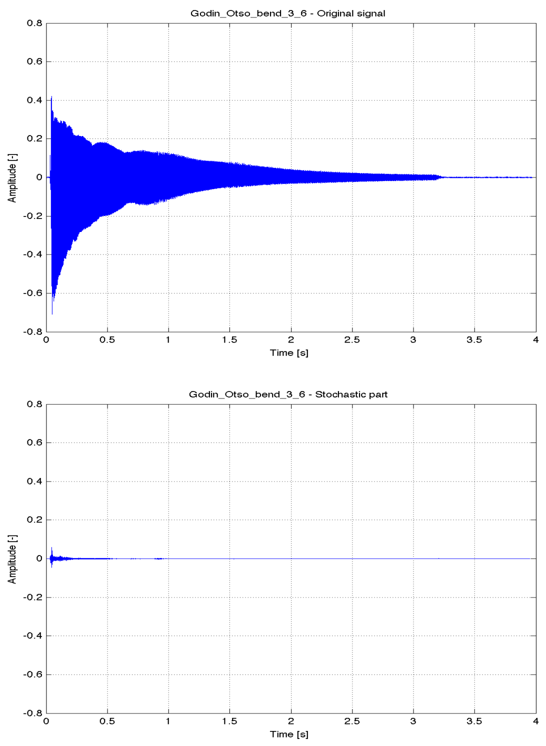

Figure 15: *Example of bend playing mode: string 3, fret 6; original signal (top), stochastic part of the signal extracted with a SMS analysis*

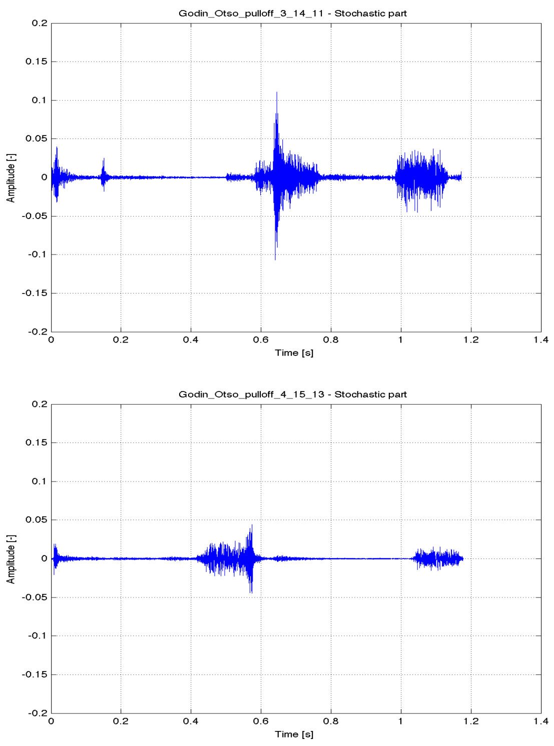

Figure 16: *Two examples of pulloff playing mode: stochastic part of the signal extracted with a SMS analysis*

the example of acoustic guitar chord, proposed by *libsms*. It treats the guitar chord as an inharmonic signal and suggests to track 200 partials with a very short minimum track length. The lowest frequency tracked is set to 85 Hertz and a hamming window is chosen to allow for sub-band frequencies.

To verify our hypothesis that the stochastic part of the recorded signal comprises the main characteristics of the playing techniques, we focused on the bend samples. This playing technique has rather no excitation mechanism. Our assumption is verified in the Figure 15: the only amplitude detected in the residual signal corresponds to the plucked note in the beginning of the recording.

For all extracted stochastic signals, unexpected amplitudes arise at the end of the signals. This must be due to the abrupt end of the recordings and to the fact that the SMS analysis considers it as part of the noisy component. We do no take it into account in the observation.

If we observe the stochastic signals extracted from the pulloff (Figure 16), the slide (Figure 17) and the hammer-on (Figure 18) playing modes , we can notice that the shape of the signals differs according to the playing mode: the pulloff and the hammer-on playing modes show a clear impulse in the signal, which is not the case for the slide playing mode. The hammer-on playing mode demonstrates smaller amplitude values on average, which must be due to the fact that the main part of the recording signal has been considered as sinusoidal component. This is rather consistent since a hammer excitation should give a richer spectrum, thus a larger deterministic amount of signal.

We perform statistics (standard deviation, kurtosis and skewness) directly on the stochastic signals for the four playing modes. We use SVM for classification of playing modes' pairs (see Table 7). The average rate for all the classification is **88.33%** of good results.

8.3. Conclusion and Discussion

The achieved results show interesting trends. However, the average success rate is not strong enough to be considered as significant for playing techniques classification. Even though, we are convinced that extracting the stochastic signal is a promising method. More work is needed to ensure the results.

First, it has to be noticed that some difficulties may arise due to differences in amplitude among recordings. Indeed, stronger the excitation mode is, richer the spectrum may appear, which may give different results among recordings of a playing mode and consequently, different outputs for stochastic signal extraction. Thus, a more homogeneous sound database may improve the results.

As noticed by Serra [20], separation between deterministic and stochastic components of a sound is often not obvious. The deterministic part is assumed to be restricted to sums of quasi-sinusoidal components where each sinusoid models a narrowband component of the original sound and is described by an amplitude and a frequency function. On the other hand, the stochastic part is generally described as filtered white noise, thus it hardly preserves either the instantaneous phase or the exact magnitude details of individual FFT frames. In particular, the model is not adequate when sounds include noisy partials. Moreover, transients, such as onsets, evolve too fast to be precisely represented by the sinusoidal part, even if their phase values are not stochastic. Eakin et al. [6] recommend further methods in this case [27, 7]. Even if the

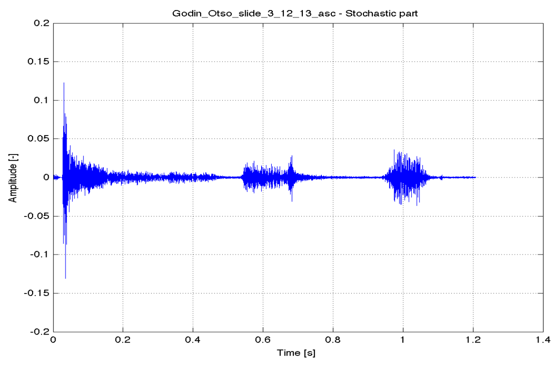
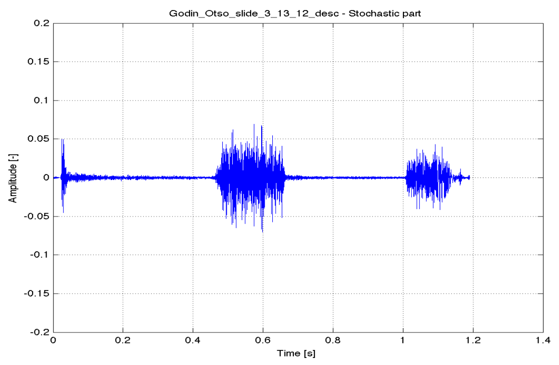

Figure 17: *Two examples of slide playing mode: stochastic part of the signal extracted with a SMS analysis*

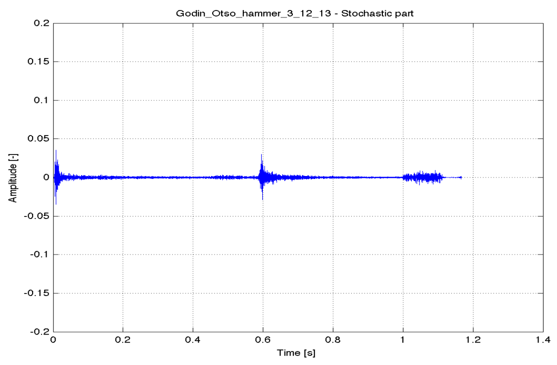
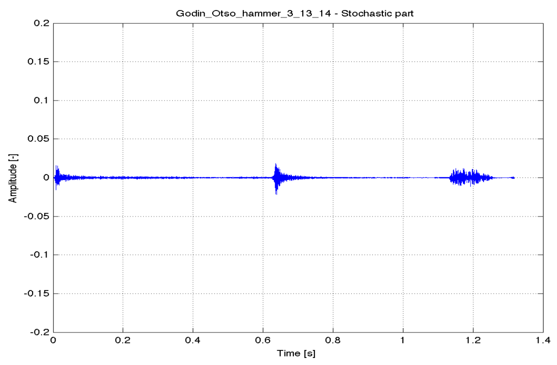

Figure 18: *Two examples of hammer-on playing mode: stochastic part of the signal extracted with a SMS analysis*

SMS method is flexible enough to give the user some controls [6], these parameters are difficult to apply in a real-time process. In our case, more work has to be done to evaluate specific parameters for strings sounds.

9. SOFTWARE IMPLEMENTATION

In order to demonstrate some possible uses of the data acquired via our ”guitar as controller” analysis algorithms, we implemented a series of data-to-sound applications in Max/MSP. The idea was to allow for a sensitive and intuitive control of audio effects by the guitar player, directly via the intonations of the playing itself. We were looking for a system offering a more sensitive control of guitar effects than the traditional pedal-type controllers, where the player would feel more ”in contact” with the audio processing via the minute hand/finger movements of the various techniques.

The articulation-to-sound mappings were the following:

- **Mute -> octaver (-1 octave)**: a detected "muted" note sets an Octaver effect on, producing an electric bass-type sound. The player can switch from "normal" sound to a "bass" sound by simply applying palm muting on the strings.
- **Bend -> wah wah filter Q**: the bend gesture is used to control a wah-wah filter’s frequency, creating a wailing effect on the bent notes. This effect is very sensual and "bluesy", as the expressiveness of the bent notes is accentuated by the filter sweep.
- **Harmonics -> reverse delay**: once a harmonic note is detected, a reverse delay is applied on the signal, creating a "space" effect on the tone.
- **Hammer-on & pull-off -> cross synthesis with a percussive sound**: the percussive gestures of the hammer-on and pull-off techniques are made audible by merging the original guitar sound with a pre-recorded percussive sound, e.g. a bass clarinet "slap" effect. The resulting hybrid tone emphasizes the percussive quality of the notes, creating a new gesture-sound articulation on the guitar.
- **Plucking point -> granular synthesis**: the attack position on the string is mapped into a granular synthesizer . While the "normal" (i.e. soundhole) attack produces no granular effect, the more the attack approaches the bridge, the more high-frequency "grains" (+1 and +2 octaves) are mixed into the sound output. In a similar manner, the attacks on the fingerboard zone produce a sound output mixed with low-frequency (-1 or octaves) grains.

These mappings constitute the first effort of real-time implementation and musical use of our work on guitar articulations detection. The results are encouraging, producing a real feeling of gestural control of the sound processing: the guitarist feels that his/her gestures are closely connected to the produced sound.

However, the real-time implementation highlights numerous problems inherent to the connection of a live musical instrument playing to a computational feature analysis system. The main challenge is the constant variation in the way the articulations are played in a real musical setting. The samples recorded into our database constitute "laboratory" examples of guitar articulations whereas the live rendition of the same techniques may give rise to quite different data sets. Guitar playing in a "real" musical context is a extensively complex environment and it is very difficult to

	BEND	HAMMER	PULLOFF	SLIDE
BEND		100%	100%	100%
HAMMER	100%		70%	85%
PULLOFF	100%	70%		75%
SLIDE	100%	85%	75%	

Table 7: *Percent of good results for playing modes classification using SVM on stochastic signal statistics (standard deviation, kurtosis and skewness) applied on stochastic signals. We used a training group of 48 recordings, 12 by playing modes and a test group of 20 recordings, 5 by playing modes.*

model and articulate with automated feature extraction.

10. HARDWARE

This stage of the project intends to research and test candidate technologies able to provide features on power processing and signal conversion inside the guitar. The hardware requirements are:

- to be installable on a guitar (low power, small, stand-alone),
- to be stand-alone solution able to perform the features processing (no PC dependences),
- to provide high quality hexaphonic to digital conversion,
- to supply a standard data output (e.g, USB),
- to offer easy integration of drivers for general sound processing software.

The distinct elements of the proposed hardware architecture are shown on figure 19. It is based on a field-programmable gate array (FPGA) component combining three general functions: control of digital sound conversion, the extraction of the playing technics and exporting the features data over the USB driver. The main advantage of FPGA over a microprocessor is its native parallel processing capacity. Processing time constrains are found on two functions in particular: digital audio conversion and features processing. They both require parallel processing in order to overcome performance restrictions. The different parts of the system are listed below:

- `Digital Audio Conversion`: the hexaphonic pickup used in the guitar yields six independent analog audio signals, one per string. The conversion of analog signals to digital is performed by four WM8782 Wolfson [1] high performance stereo audio ADCs: SNR 100dB, up to 192khz per channel, up to 24bits of resolution, DC off-set low-pass digital filter, etc. Despite good ADC technical characteristics, the challenge is to drive the eight ADCs at the same time in order to reach high quality of audio conversion. Parallel control is thus required. The choice of FPGA solution allows simultaneous execution of control and communication of each ADC chip in order to achieve the max components' throughput. The ADCs can be seen in the figure 20.
- `FPGA`: the prototype used a DE0-Nano development board from Terasic[2] featuring an Altera[3] Cyclone IV FPGA device, up to 22,320 Logical Elements, SDRAM and EEPROM memory on board, two external GPIO headers (40 digital I/O), board programming by USB mini-AB port. The component has enough processing power to support all processing required for this project. The two control blocks (ADCs and USB) were built and the system is ready to receive the feature processing algorithms when they are implemented. The FPGA component and board are shown in figure 20.

[1]`http://www.wolfsonmicro.com`
[2]`http://www.terasic.com`
[3]`http://www.altera.com`

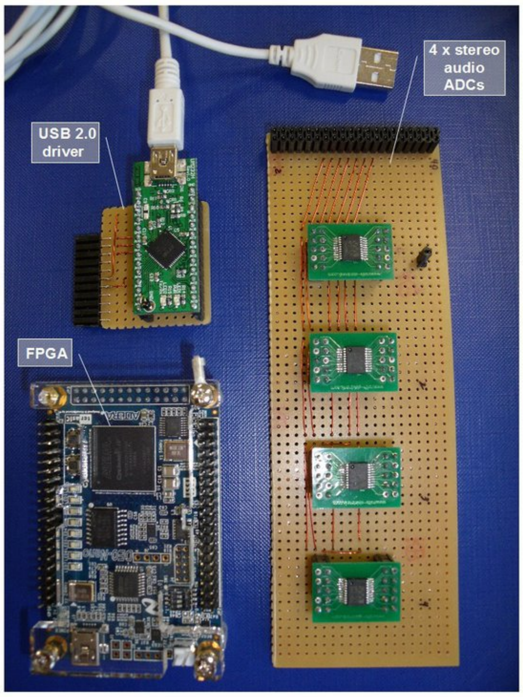

Figure 20: *Electronic parts of developed prototype: ADCs, FPGA and USB driver.*

Concerning the FPGA programming, some issues should be underlined. First of all, FPGAs have a particular programming mode which is completely distinct of computer systems. Many FPGA tools and solutions have been proposed and launched in the market in order to make it comparable to ordinary programming style. Some of them are using SIMULINK-Matworks[4] DSP programming environment as visual interface to program FPGAs with DSP functions, such as: DSPbuilder from Altera and SystemGenerator from Xilinxs[5]. The idea is to extend the use of a well known DSP programming environment with purpose to program the FPGAs with the same style of function-blocks used on SIMULINK. The DSPbuilder was chosen to be experimented in this project, whose part of the "program" developed to control ADC and USB blocks are seen in figure 21.

DSPbuilder has a function-blocks library compliant to SIMULINK model. In this way, these blocks can be simulated and after synthesized to the FPGA. However, DSP-

[4]`http://www.mathworks.nl/products/simulink/index.html`
[5]`http://www.xilinxs.com`

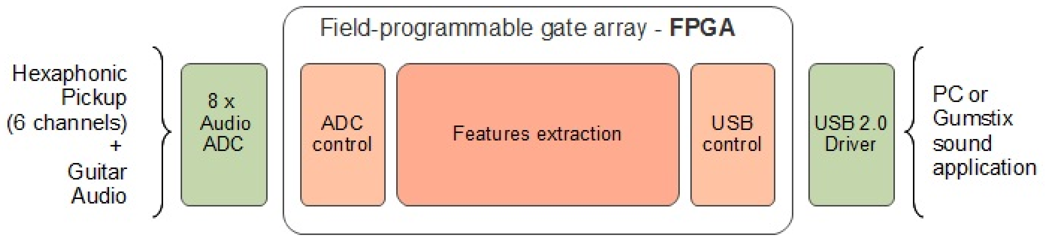
Field-programmable gate array - **FPGA**
Hexaphonic Pickup (6 channels) + Guitar Audio
8 x Audio ADC
ADC control
Features extraction
USB control
USB 2.0 Driver
PC or Gumstix sound application

Figure 19: *Proposed hardware architecture: summary of functional blocks.*

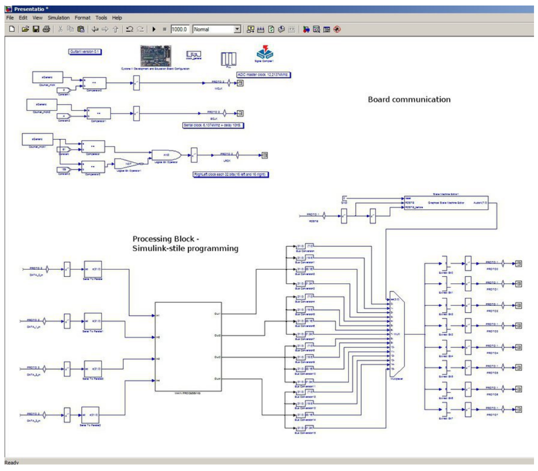

Figure 21: *Control ADC and USB blocks developed in DSPbuilder system.*

builder library doesn't have an equivalent for all SIMULINK block-sets and, further, its function-blocks contain extra constraints not found in the SIMULINK blocks. Consequently, a developer must learn how to use the SIMULINK-like programming even if he knows SIMULINK very well. Because of that, the time used in learning the tool has exceeded a lot the initial expectation, preventing from putting efforts on extraction features functions running on the FPGA.

Despite of learning time, it can also incorporate many benefits such as shorter development time, a proven increase in code efficiency, floating-point development with the fused-datapath tool flow, floating-point vector processing to support linear algebra operations, reduced development and verification efforts with DSP Builder Advanced Blockset, floating-point and fixed-point library functions, etc.

- `USB driver`: data communication between electronic devices using USB is widely adopted in information technology industry since twenty years. This spread technology can be incorporated in systems by USB driver modules, which are engaged on performing all USB protocol layers. The interface of hardware-to-USBdriverChip is implemented by simple serial data communication. And the interface USBdriverChip-to-PC can be developed using a serial port emulator (called VPN virtual COMM port) or calls of communication functions from inside the programming codes (called D2XX direct driver). This last one was used in our development as it is faster than VPN mode: 12Mbaud/sec on VPN versus 480Mbits/s on D2XX mode. Also, it can quickly be integrated on OSC and / or MIDI functions for sound applications codes. The FT232H module from FTDI[6] was exploited in this project as shown in the figure 20.

11. CONCLUSION

The guitar as controller Numediart project focused on features extraction from a hexaphonic guitar signal, in order to detect and recognize all the playing techniques commonly used on this instrument. The methodology was built on the consecutive discretisation of 1) attacks / note onsets, 2) left-hand and right-hand articulations, 3) articulation types: "normal", mute, bend, slide, hammer-on, pull-off, harmonic, palm muting and plucking point.

A set of custom detection algorithms was developed off-line in Matlab and real-time in Max/MSP, showing a general detection success rate of approx. 90%. for all the techniques. These results give a proof of concept for a general guitar playing techniques detection software. However, the project's framework did not allow for the compilation of all the algorithms into one single software, because of power processing issues. In order to propose a functional playing techniques detector, numerous challenges must still be faced.

Regarding the hardware part, the conceived architecture can be used with an ordinary PC or a small embedded platform running a DSP application. The last option would allow a stand-alone system be integrated to the guitar. We have experimented SuperCollider[7] sound processing software running on Linux into a Gumstix[8] board. It has proved to be a quality and reliable solution for embedded sound application. Both approaches must be explored in future development in order to have a true effective application.

The transfer of non-real-time Matlab algorithms into real-time application (e.g. Max/MSP or FPGA implementation) is not yet complete. Moreover, one must proceed towards more comprehensive tests on the algorithms on other types of guitars, as well as other players. Live guitar playing is a complex object for signal analysis, and a prolongated effort of system testing and "tuning" is necessary in order to provide a robust software or hardware implementation.

[6]`http://Ltd.www.ftdi.com`
[7]`http://supercollider.sourceforge.net`
[8]`http://www.gumstix.com`

12. ACKNOWLEDGMENTS

numediart is a long-term research program centered on Digital Media Arts, funded by Région Wallonne, Belgium (grant N°716631).

13. REFERENCES

13.1. Scientific references

[1] Juan Pablo Bello et al. "A Tutorial on Onset Detection in Music Signals". In: *IEEE Transactions on Speech and Audio Processing,* 16 (5) (2005). P.: 42.

[2] Paul M. Brossier. "Automatic Annotation of Musical Audio for Interactive Applications". PhD thesis. Centre for Digital Music Queen Mary, University of London, 2006. P.: 42.

[3] Gavin Carfoot. "Acoustic, Electric and Virtual Noise: The Cultural Identity of the Guitar". In: *Leonardo Music Journal* 16 (2006). Pp. 35–39. P.: 41.

[4] Nick Collins. "A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions". In: *118th Audio Engineering Society Convention*. Barcelona 2005. P.: 42.

[5] Simon Dixon. "Onset Detection Revisited". In: *9th Int. Conference on Digital Audio Effects (DAFx-06)*. Montreal, Canada 2006. P.: 42.

[6] R. T. Eakin and Xavier Serra. "SMSPD, LIBSMS and a Real-Time SMS Instruments". In: *International Conference on Digital Audio Effects*. Cosmo, Italy 2009. Pp.: 48–50.

[7] Kelly Fitz, Lippold Haken, and Paul Chirstensen. "Transient Preservation under Transformation in an Additive Sound Model". In: *Proceedings of the International Computer Music Conference*. Berlin 2000. P.: 49.

[9] Vesa Välimäki Henri Pettinen. "A time-domain approach to estimating the plucking point of guitar tones obtained with an under-saddle pickup". In: *Applied Acoustics*. Vol. 65. 12. 2004. Pp. 1207–1220. P.: 47.

[10] Alexandre Lacoste and Douglas Eck. "A Supervised Classification Algorithm for Note Onset Detection". In: *EURASIP Journal on Advances in Signal Processing* 2007 (2007). P.: 43.

[12] P. Masri. "Computer Modeling of Sound for Transformation and Synthesis of Musical Signal". PhD thesis. Univ. of Bristol, 1996. P.: 42.

[13] "Methodology and Tools for the Evaluation of Automatic Onset Detection Algorithms in Music". In: *Int. Symp. on Music Information Retrieval*. Beijing, China 2004. P.: 43.

[16] Eduardo Reck Miranda and Marcelo Wanderley. "New digital musical instruments: control and interaction beyond the keyboard". In: *Computer music and digital audio series*. Ed. by Wisconsin A-R Editions Middleton. Vol. 21. 2006. URL: http://eleceng.ucd.ie/~pogrady/papers/OGRADY_RICKARD_ISSC09.pdf. P.: 41.

[17] *Music Information Retrieval Evaluation eXchange (MIREX)*. URL: http://www.music-ir.org/mirex/wiki/2011:Main_Page. P.: 42.

[18] Jonathan Carey Norton. "Motion capture to build a foundation for a computer-controlled instrument by study of classical guitar performance". PhD thesis. CCRMA, Dept. of Music, Stanford University, 2008. P.: 41.

[20] X. Serra. "A System for Sound Analysis-Transformation-Synthesis based on a Deterministic plus Stochastic Decomposition". PhD thesis. CCRMA, Dept. of Music, Stanford University, 1989. Pp.: 48, 49.

[21] X. Serra. "Musical Sound Modeling with Sinusoids plus Noise". In: *Musical Signal Processing*. Ed. by Roads et al. Swets & Zeitlinger, 1997. Pp. 91–122. ISBN: 90 265 1482 4. P.: 48.

[22] Dan Stowell and Mark Plumbley. "Adaptive Whitening for Improved Real-Time Audio Onset Detection". In: 2007. P.: 42.

[23] Koen Tanghe et al. "Collecting Ground Truth Annotations for Drum Detection in Polyphonic Music". In: *6th International Conference on Music Information Retrieval*. 2005. P.: 43.

[25] "Three Dimensions of Pitched Instrument Onset Detection". In: *IEEE Transactions on Audio, Speech and Language Processing* 18 (6) (2010). Pp.: 42, 43.

[26] Caroline Traube and Philippe Depalle. "Extraction of the excitation point location on a string using weighted least-square estimation of comb filter delay". In: *Proceedings of the Conference on Digital Audio Effects (DAFx)*. 2003. URL: http://www.elec.qmul.ac.uk/dafx03/proceedings/pdfs/dafx54.pdf. P.: 47.

[27] Tony S. Verma and Teresa H. Y. Meng. "Extending Spectral Modeling Synthesis with Transient Modeling Synthesis". In: *Comput. Music J.* 24 (2 2000). Pp. 47–59. ISSN: 0148-9267. DOI: 10.1162/014892600559317. P.: 49.

[28] Yu Shiu Wan-Chi Lee and C.-C. Jay Kuo. "Musical Onset Detection with Joint Phase and Energy Features". In: *IEEE International Conference on Multimedia and Expo*. Beijing, China 2007. P.: 42.

[29] Wenwu Wang et al. "Note Onset Detection via Nonnegative Factorization of Magnitude Spectrum". In: *EURASIP Journal on Advances in Signal Processing* 2008 (2008). P.: 42.

13.2. Artistic references

[8] Godin. *Godin Multiacs*. 2000. URL: http://www.godinguitars.com/godinacsp.htm. P.: 41.

[11] Queen Mary University of London. *Sonic Visualizer*. URL: http://www.sonicvisualiser.org/. P.: 41.

[14] Keith Mc Millen. *StringPort*. URL: http://www.keithmcmillen.com. P.: 41.

[15] *Mir Toolbox*. URL: https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox. P.: 41.

[19] Roland. *Roland VG-99 pitch to midi converter*. 2000. URL: http://www.rolandus.com/products/productdetails.php?ProductId=849. P.: 41.

[24] Axon Technology. *Axon AX 50 USB pitch to midi converter*. 2000. URL: http://www.axon-technologies.net/. P.: 41.

13.3. Software and technologies

[1] Juan Pablo Bello et al. “A Tutorial on Onset Detection in Music Signals”. In: *IEEE Transactions on Speech and Audio Processing,* 16 (5) (2005). P.: 42.

[2] Paul M. Brossier. “Automatic Annotation of Musical Audio for Interactive Applications”. PhD thesis. Centre for Digital Music Queen Mary, University of London, 2006. P.: 42.

[3] Gavin Carfoot. “Acoustic, Electric and Virtual Noise: The Cultural Identity of the Guitar”. In: *Leonardo Music Journal* 16 (2006). Pp. 35–39. P.: 41.

[4] Nick Collins. “A Comparison of Sound Onset Detection Algorithms with Emphasis on Psychoacoustically Motivated Detection Functions”. In: *118th Audio Engineering Society Convention*. Barcelona 2005. P.: 42.

[5] Simon Dixon. “Onset Detection Revisited”. In: *9th Int. Conference on Digital Audio Effects (DAFx-06)*. Montreal, Canada 2006. P.: 42.

[6] R. T. Eakin and Xavier Serra. “SMSPD, LIBSMS and a Real-Time SMS Instruments”. In: *International Conference on Digital Audio Effects*. Cosmo, Italy 2009. Pp.: 48–50.

[7] Kelly Fitz, Lippold Haken, and Paul Chirstensen. “Transient Preservation under Transformation in an Additive Sound Model”. In: *Proceedings of the International Computer Music Conference*. Berlin 2000. P.: 49.

[8] Godin. *Godin Multiacs*. 2000. URL: http://www.godinguitars.com/godinacsp.htm. P.: 41.

[9] Vesa Välimäki Henri Pettinen. “A time-domain approach to estimating the plucking point of guitar tones obtained with an under-saddle pickup”. In: *Applied Acoustics*. Vol. 65. 12. 2004. Pp. 1207–1220. P.: 47.

[10] Alexandre Lacoste and Douglas Eck. “A Supervised Classification Algorithm for Note Onset Detection”. In: *EURASIP Journal on Advances in Signal Processing* 2007 (2007). P.: 43.

[11] Queen Mary University of London. *Sonic Visualizer*. URL: http://www.sonicvisualiser.org/. P.: 41.

[12] P. Masri. “Computer Modeling of Sound for Transformation and Synthesis of Musical Signal”. PhD thesis. Univ. of Bristol, 1996. P.: 42.

[13] “Methodology and Tools for the Evaluation of Automatic Onset Detection Algorithms in Music”. In: *Int. Symp. on Music Information Retrieval*. Beijing, China 2004. P.: 43.

[14] Keith Mc Millen. *StringPort*. URL: http://www.keithmcmillen.com. P.: 41.

[15] *Mir Toolbox*. URL: https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox. P.: 41.

[16] Eduardo Reck Miranda and Marcelo Wanderley. “New digital musical instruments: control and interaction beyond the keyboard”. In: *Computer music and digital audio series*. Ed. by Wisconsin A-R Editions Middleton. Vol. 21. 2006. URL: http://eleceng.ucd.ie/~pogrady/papers/OGRADY_RICKARD_ISSC09.pdf. P.: 41.

[17] *Music Information Retrieval Evaluation eXchange (MIREX)*. URL: http://www.music-ir.org/mirex/wiki/2011:Main_Page. P.: 42.

[18] Jonathan Carey Norton. “Motion capture to build a foundation for a computer-controlled instrument by study of classical guitar performance”. PhD thesis. CCRMA, Dept. of Music, Stanford University, 2008. P.: 41.

[19] Roland. *Roland VG-99 pitch to midi converter*. 2000. URL: http://www.rolandus.com/products/productdetails.php?ProductId=849. P.: 41.

[20] X. Serra. “A System for Sound Analysis-Transformation-Synthesis based on a Deterministic plus Stochastic Decomposition”. PhD thesis. CCRMA, Dept. of Music, Stanford University, 1989. Pp.: 48, 49.

[21] X. Serra. “Musical Sound Modeling with Sinusoids plus Noise”. In: *Musical Signal Processing*. Ed. by Roads et al. Swets & Zeitlinger, 1997. Pp. 91–122. ISBN: 90 265 1482 4. P.: 48.

[22] Dan Stowell and Mark Plumbley. “Adaptive Whitening for Improved Real-Time Audio Onset Detection”. In: 2007. P.: 42.

[23] Koen Tanghe et al. “Collecting Ground Truth Annotations for Drum Detection in Polyphonic Music”. In: *6th International Conference on Music Information Retrieval*. 2005. P.: 43.

[24] Axon Technology. *Axon AX 50 USB pitch to midi converter*. 2000. URL: http://www.axon-technologies.net/. P.: 41.

[25] “Three Dimensions of Pitched Instrument Onset Detection”. In: *IEEE Transactions on Audio, Speech and Language Processing* 18 (6) (2010). Pp.: 42, 43.

[26] Caroline Traube and Philippe Depalle. “Extraction of the excitation point location on a string using weighted least-square estimation of comb filter delay”. In: *Proceedings of the Conference on Digital Audio Effects (DAFx)*. 2003. URL: http://www.elec.qmul.ac.uk/dafx03/proceedings/pdfs/dafx54.pdf. P.: 47.

[27] Tony S. Verma and Teresa H. Y. Meng. “Extending Spectral Modeling Synthesis with Transient Modeling Synthesis”. In: *Comput. Music J.* 24 (2 2000). Pp. 47–59. ISSN: 0148-9267. DOI: 10.1162/014892600559317. P.: 49.

[28] Yu Shiu Wan-Chi Lee and C.-C. Jay Kuo. “Musical Onset Detection with Joint Phase and Energy Features”. In: *IEEE International Conference on Multimedia and Expo*. Beijing, China 2007. P.: 42.

[29] Wenwu Wang et al. “Note Onset Detection via Nonnegative Factorization of Magnitude Spectrum”. In: *EURASIP Journal on Advances in Signal Processing* 2008 (2008). P.: 42.

RT-MEDIACYCLE : TOWARDS A REAL-TIME USE OF MEDIACYCLE IN PERFORMANCES AND VIDEO INSTALLATIONS

Xavier Siebert [1], *Stéphane Dupont* [2], *Christian Frisson* [2], *Bernard Delcourt* [3]

[1] Laboratoire de Mathématique et Recherche Opérationnelle (MathRo), Université de Mons (UMONS), Belgique
[2] Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), Université de Mons (UMONS), Belgique
[3] Video artist, Brussels, Belgique.

1. ABSTRACT

This project aims at developing tools to extend the use of the `MediaCycle` software to live performances and installations. In particular, this requires real-time recording and analysis of media streams (e.g., video and audio). These tools are developed as patches for the `Puredata (Pd)` software [10] which is widely used by the artistic community. This project is a collaboration with the Belgian video artist Bernard Delcourt [2] who will incorporate `MediaCycle` in his interactive video installations and performances in Spring 2012.

2. INTRODUCTION

The `MediaCycle` software [13] has been developed at `numediart` to classify multimedia databases (sounds, images, videos, texts, ...) based on the similarity among the content of these media. Artistic applications to which this software has contributed (e.g., Bud Blumenthal's *DANCERS!*[1], Thomas Israel's *Méta-Crâne*, ...) implied up to now an analysis of pre-recorded media.

The goal of this project is therefore to develop software tools to display and organize media recorded on-the-fly. To this end, we have developed a set of tools written in `Puredata (Pd)` [10] and intented for the artistic community, to accomplish the following tasks:

- record sounds and videos
- segment videos
- manage a playlist of videos to be displayed
- handle a multi-screen projection system, with options to show and manipulate (mix, fade, loop, ...) videos
- communicate by OSC between `MediaCycle` and the `Pd` patches

These tools will be described in the following section.

3. `Pd` PATCHES

3.1. Audio and Video Recordings

capture are done separately, using the audio rate to control video (the opposite would lead to noisy sound). For video capture, the `Gem` module of `Pd` as well as the `pdp` module have been implemented, `Gem` giving more flexibility but `pdp` being more stable. On Fig. 1 is the `Pd` patch for recording videos. The left-hand side contains options to add effects in the video, such as blur, contrast, hue, saturation, ...

[1]`http://www.dancersproject.com`

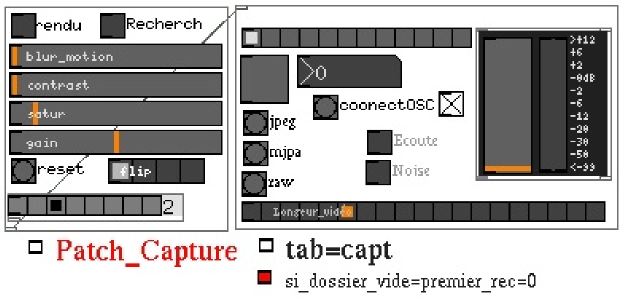

Figure 1: `Pd` component for video capture.

3.2. Segmentation

We implemented video segmentation in `Pd`, based on the pixel-per-pixel difference between successive frames. This method is simpler than other segmentation algorithms used in previous numediart projects (e.g., the Bayesian Information Criterion, BIC), and it was chosen so that it can be implemented in `Pd`.

Fig. 2 shows a typical result obtained for the segmentation on André Hunebelle's movie *Fantômas* (1964). On the right are frames of a video corresponding to two different segments . On the left of Fig. 2 are monitored the difference between successive frames (top graph), as well as peaks indicating a separation between segments (bottom graph). A variable threshold can be changed in real-time (e.g., using a slider) to modulate the number of segments.

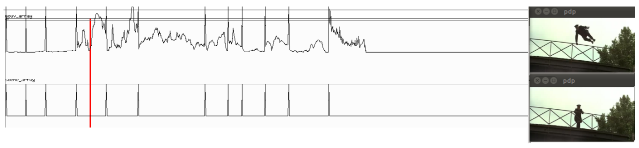

Figure 2: Snapshot of a `Pd` patch for segmenting a video stream.

3.3. Video Playlist

Using the `videogrid` module distributed in `pdvjtools` [9] for `Pd`, we designed a video queuing system (Fig 3). Among other features, it contains a preview of the video to be displayed as well as *Drag and Drop* capabilities to add/remove videos from the list.

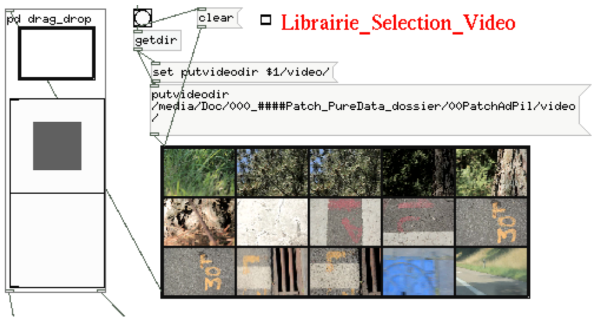
pd drag_drop
clear
Librairie_Selection_Video
getdir
set putvideodir $1/video/
putvideodir /media/Doc/000_####Patch_PureData_dossier/00PatchAdPil/video /

Figure 3: Pd patch for video playlist.

3.4. Multi-screen projection system

In the context of artistic performances/installation, it is often interesting to display selected videos on different screens. Therefore we also developed a patch for multi-screen projection. It has been used for the moment with 2 or 3 screens at a time (see Fig. 4), but can also be used for more screens, depending on the available machine resources.

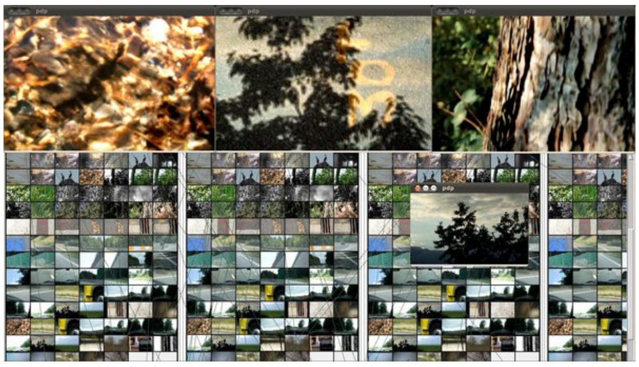

Figure 4: Pd patch for projection on three screens. The three videos to be projected on the screens are in the top, while the library is in the bottom. Also visible on the right is a preview of the next video to be added in the playlist.

3.5. OSC Communication

We updated our implementation of the remote control of MediaCycle through the OSC communication protocol (summarised in Table. 1), following up the partial coverage started in 2010 [3]. It allows the communication with the core engine of MediaCycle through any graphical interface or external controllers.

This OSC namespace is likely to be again further more extended in future numediart projects.

4. OVERVIEW OF THE COMPLETE SYSTEM

Figure 5 represents a schematic of the Pd patches that have been developed, and their interaction with MediaCycle, while Figure 6 shows the current system as it would be used in a performance.

OSC tag	Dir.	Type	Range	Comment
move/xy	in	float	[0, size]	move camera to x . . .
	in	float	[0, size]	. . . and y
hover/xy	in	float	[0, size]	hover on x . . .
	in	float	[0, size]	. . . y
zoom	in	float	> 0	zoom view by factor
rotate	in	float	[0, 360]	rotate view by factor
recenter	in	-	-	reset view
forward	in	-	-	enter in cluster
back	in	-	-	exit cluster
mediaID	out	int	[0, n-1]	Id of the current media
getknn	in	int	[0, n-1]	ask k nearest neighbors
	out	int[k]	[0, n-1]	return k neighbor id's
getkff	in	int	[0, n-1]	first k . . .
	in	int	[0, nfeat]	. . . for a given feature
	out	int[k]	[0, n-1]	return k id's asked
feature/i	in	int	-	toggle feature i on/off
add/s	in	string	-	add media file
remove/i	in	int	[0, n-1]	remove media from id

Table 1: OSC namespace of the MediaCycle browser, with the direction (in: parameter, out: return value), the type and range of the transmitted variables. The double horizontal line separates the previous implementations (top) from the recent (bottom).

5. PERSPECTIVES

For the moment the video streaming is managed by the Pd patches. The video stream is then segmented in Pd (either manually, or by a separate segmentation patch described in section 3.2) and sent to MediaCycle, to be incorporated in its database. Alternatively, streaming could be fully incorporated in MediaCycle, for example using GStreamer [4] which could also synchronize audio and video. For the moment the only media type that can be streamed within MediaCycle is video, using the OpenCV_2.3 library. Other streaming technologies could also be incorporated [1].

5.1. Discovering GStreamer throughout installing scenic for teleperformances

In the framework of Transat/Contamines, the collaborative research program in digital art led by Transcultures (Mons) [14] and La Société des Arts Technologiques (SAT) (Montreal) [5], numediart hosted its first teleperformance on Saturday Dec 3rd, 2011. Two Belgian artists (Valérie Cordy and Laurence Moletta) and two Canadian artists (Alexandre Quessy and Michal Seta) performed a joint transatlantic "digital soup". numediart provided its technological know-how for running scenic [6], a telepresence software developed by SAT. Two video streams and two stereo audio streams were sent from each side of the ocean. For that purpose we set up 2 macbook pro laptops with Ubuntu 10.04 (the current production distribution for scenic). This teleperformance opened the doors for further international real-time collaboration on digital art.

The major advantage of scenic is to provide an user-friendly wrapper over gstreamer tuned for teleperformances, usable from the command-line with its milhouse engine or using the scenic GUI. Gstreamer allows to process audio and video streams through pipelines, processing file and streams seamlessly and similarly.

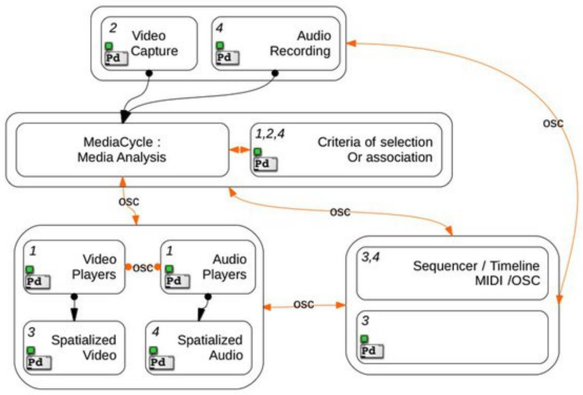

Figure 5: Organisation of the various components of the tool.

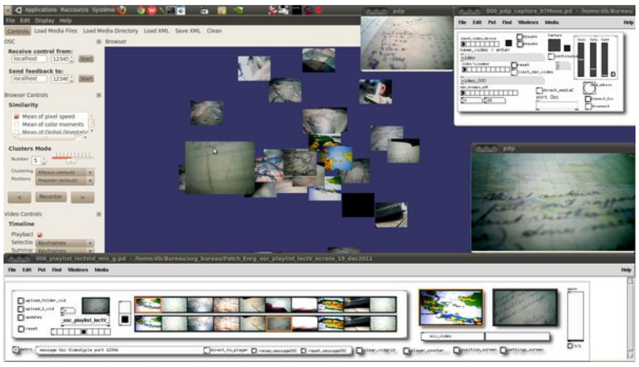

Figure 6: Snapshot of most components used together: video capture, playlist and projection (all three in `Pd`) interacting with `MediaCycle`.

Pipelines can be run using various language wrappers or from a terminal launcher, for example, to visualize the default webcam (source) on a default window (sink) on OSX:

```
gst-launch qtkitvideosrc ! colospace ! ximagesink
```

Various plugins are available for GStreamer, including OpenCV bindings. Among other applications based on GStreamer, the closest to MediaCycle is Mirage [11, 12], a plugin for automatic playlist generation based on music information retrieval for the music collection manager Banshee.

GStreamer is quite a promising candidate for handling streams in MediaCycle, but still suffers from lack of OSX testing and documentation (our main OS for development), the simplest "helloworld" pipeline described above required thorough manual tweaking for installing GStreamer libraries using MacPorts. Moreover, integrating GStreamer into MediaCycle would require a major rework of the architecture, not only converting feature extraction plugins as GStreamer plugins, but basing the whole processing pipeline on GStreamer. Complementarily, the PdGst GStreamer library for PureData [15] might allow to reuse rapidly these GStreamer-based MediaCycle feature extraction plugins into PureData.

5.2. Easing the installation of MediaCycle by setting up an online repository with pre-compiled dependencies for Ubuntu

Bernard Delcourt opted for the most recent Long Term Service version of Ubuntu that was released in April 2010, a safe choice for stability. Since some major dependencies of MediaCycle (Armadillo, FFmpeg, OpenCV, OpenSceneGraph) are still not up-to-date for that Ubuntu release, we opened a numediart LaunchPad team [7] and a first `numediart-deps` Personal Package Archive (PPA) repository [8] so as to package more recent versions of these dependencies notably for Ubuntu 10.04 (and upwards), to make proper MediaCycle packages depending on these dependencies, and to allow seamless installation of the whole. It now just requires adding the repository to the software management system (through Synaptic for instance, installation details on [8]) and double-clicking on MediaCycle packages (that are not yet available on our Launchpad repository since the MediaCycle license is not curently opensource).

6. ACKNOWLEDGMENTS

`numediart` is a long-term research program centered on Digital Media Arts, funded by Région Wallonne, Belgium (grant N°716631).

We would like to thank Michal Seta for our fruitful discussions while fine-tuning the installation of scenic at the numediart lab.

7. REFERENCES

7.1. Scientific references

[1] David Austerberry. *The technology of video and audio streaming*. Focal Press, 2005. ISBN: 0-240-80580-1. P.: 56.

[3] Christian Frisson et al. “DeviceCycle: rapid and reusable prototyping of gestural interfaces, applied to audio browsing by similarity”. In: *Proceedings of the New Interfaces for Musical Expression++ (NIME++)*. 2010. P.: 56.

[12] Dominik Schnitzer. “Mirage: High-Performance Music Similarity Computation and Automatic Playlist Generation”. MA thesis. Vienna University of Technology, 2007. URL: `http://hop.at/mirage/`. P.: 57.

[13] Xavier Siebert et al. “MediaCycle: Browsing and Performing with Sound and Image libraries”. In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit. Vol. 2. 1. numediart Research Program on Digital Art Technologies. 2009. Pp. 19–22. URL: `http://www.numediart.org/docs/numediart_2009_s05_p3_report.pdf`. P.: 55.

[15] IOhannes m zmölnig. “PdGst - GStreamer bindings for Pd”. In: *The International Convention for Puredata (PdCon09)*. 2009. URL: `http://umlaeute.mur.at/Members/zmoelnig/projects/pdgst`. P.: 57.

7.2. Artistic references

[2] Bernard Delcourt. URL: `http://vimeo.com/user2315349`. P.: 55.

[5] La Société des Arts Technologiques (SAT). URL: `http://www.sat.qc.ca`. P.: 56.

[14] Transcultures. URL: `http://www.transcultures.be`. P.: 56.

7.3. Software and technologies

[4] "GStreamer: opensource multimedia framework". URL: http://gstreamer.freedesktop.org/. P.: 56.

[6] La Société des Arts Technologiques (SAT). "scenic: a telepresence application for live performances and installations". source code on https://github.com/sat-metalab/scenic. 2011. URL: http://code.sat.qc.ca/trac/scenic/. P.: 56.

[7] numediart. "Launchpad Team for creating, manipulating and packaging various kinds of software throughout numediart projects". Personal Package Archive (ppa) repositories for Ubuntu. 2011. URL: https://launchpad.net/~numediart. P.: 57.

[8] numediart. "numediart-deps: Useful dependencies for numediart projects". Personal Package Archive (ppa) repository for Ubuntu. 2011. URL: https://launchpad.net/~numediart/+archive/deps. P.: 57.

[9] "pdvjtools". URL: http://artefacte.org/pd/. P.: 55.

[10] "PureData". URL: http://www.puredata.info. P.: 55.

[11] Dominik Schnitzer. "Mirage: An Automatic Playlist Generation Extension for Banshee". 2009. URL: http://hop.at/mirage/. P.: 57.

MAGE / PHTS IN COLLABORATIVE VOCAL PUPPETRY (COVOP) - MULTI-USER PERFORMATIVE HMM-BASED VOICE SYNTHESIS ON DISTRIBUTED PLATFORMS

Maria Astrinaki [1], *Onur Babacan*[1], *Nicolas d'Alessandro* [2], *Thierry Dutoit* [1], *Sidney Fels*[2]

[1] Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), Université de Mons (UMONS), Belgique
[2] Media and Graphics Interdisciplinary Centre (MAGIC), University of British Columbia (UBC), Canada

ABSTRACT

Speech production is a complex phenomenon with many parameters. It is very difficult for one performer to control all aspects of a synthesizer that models this phenomenon. We designed and developed a distributed, multi-user system to tackle this difficulty, where users control different aspects of the synthesizer simultaneously and interactively; treating the complex production process as a social game. HMM-based synthesizers provide flexibility at a high level of naturalness, thus we chose HTS as our synthesizer. However, HTS needs severe architectural modifications to be used reactively, and a major achievement of this work was creating MAGE/pHTS, a library for performative HMM-based speech and singing synthesis. The resulting system provides interactive controls for phonetic content and context, as well as prosody using the previously existing HandSketch controller.

KEYWORDS

MAGE, pHTS, HTS, HMM, speech synthesis, statistical parametric speech synthesis, singing synthesis, real-time, performative, streaming

1. INTRODUCTION

One of the most expressive and ubiquitous instruments of the human body is voice production, and people have always been fascinated for artificial speech and singing production. The vocal behavior and interaction though, is a very complex mechanism that needs computational power for its simulation. Nowadays, with the recent emerging technologies in sensors and computational abilities, we have a broad and complete framework to achieve high-quality, expressive and ubiquitous speech and singing synthesis, with interactive manipulation possibilities, allowing us to see and model voice production as a gestural phenomenon.

1.1. Motivation

The goal of this project is to develop the social aspects of voice production from a different perspective. Developing a new interactive voice synthesizer and a social experience platform, in order to allow a group of users / performers to manipulate various aspects of performative voice synthesis, distributed on their own device (computer or mobile). Implementing a new framework for performative voice synthesis, i.e. a real-time synthesis systems where voice is directly produced by gestural control, with no more reference to textual input. Here we address both phonetical and prosodical issues, with prototyping a first application in speech and singing synthesis. Our aim here is to define a design space for HMM-based voice synthesis as a framework for multi-user participation distributed on different platforms. Indeed we know that the simultaneous management of all the parameters required for speech production is overwhelming for a single user, even in the case of an "expert" performer. Consequently, in this project, we want to explore the idea of collaborative vocal puppetry and see how vocal intelligibility, naturalness and even identity can be addressed as a social game, involving the participation of multiple users on their own device.

1.2. Background

The collaborative vocal puppetry system (CoVoP 1.0) is composed of three main components: an interactive voice synthesis engine, a distributed social experience platform and various kinds of human-computer interaction models distributed on various kinds of devices (computers, cellphones and tablets).

The first and main aspect of this research is to develop a new synthesis engine. In this project, based on the recently implemented system pHTS [4], we developed a library for performative HMM-based speech and singing synthesis, called MAGE / pHTS. The architecture of MAGE / pHTS will be given in details in the following section.

One other significant aspect of the CoVoP project is the distribution of various parts of the synthesis process on several heterogenous devices. More than defining a convenient modularization of the synthesis engine, we need to architecture a distributed peer-to-peer service over the network. Each introduced device; a laptop with sensors, a cellphone or a tablet; will require for some resources (a part of the voice synthesis process) and display an appropriate control space to the user. As a result this platform considers the voice synthesis process as a collection of interconnected resources to be handled by the cloud of devices, with priority and dispatching strategies, as shown in Figure 1. The devices themselves bring information to the whole network, such as screen size, CPU capacity, sound properties or embedded sensors. The whole process of connecting and participating to the collaborative experience has to handle these compatibility issues and particularize the experience for each performer.

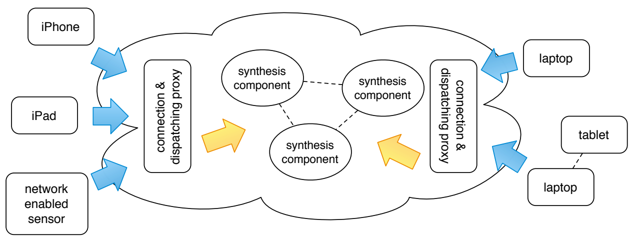

Figure 1: Block diagram of the targeted CoVoP system

We already have investigated several human-computer interaction models for controlling various aspects of voice synthesis. For example the "fan" mapping of the HandSketch [3]. In this work, HandSketch is remapped with several one-handed or two-handed touchless mappings for parameters such as pitch, volume, speed and vocal tract length. However, the introduction of multitouch and portable devices is new in the project, resulting in prototyping new HCI models.

The software implementation of this system is achieved with openFrameworks [8]. The openFrameworks project aims at providing an open-source creative environment for aggregating various software components, initially for computer graphics (OpenGL, image management, Quicktme, etc). One significant advantage of openFrameworks is to be cross-platform (Mac, Linux, Windows, iOS). Moreover the open and free add-on format attracted a lot of third party developers to wrap a large amount of new functions : OpenCV [7], XML, Open Sound Control [6], network protocols, etc. We take advantage of these existing modules for building our real-case prototype.

2. COVOP

For the implementation of the CoVoP 1.0 system, we conducted research in three main modules, which were combined at the end of the project:

- developing a new interactive voice synthesizer,
- mapping Handsketch as a prosodic controller,
- developing a module for reactive contextual control

2.1. Interactive Voice Synthesizer

The most essential part of this project, is the creation of an interactive voice synthesizer, that will enable contextual and prosodic user control. Based on pHTS, we conducted architectural modifications that transformed this static streaming approach to a reactive framework for HMM-based speech and singing synthesis, called MAGE/pHTS. In this design, MAGE is the engine independent and thread safe layer of pHTS, enabling us to handle asynchronous user input to the system in order to have direct on the fly control.

2.1.1. Architecture of pHTS

Coarticulation is known to have forward and backward effects (the latter being the most challenging for the delay issue), in speech and singing synthesis systems. This implies that a given phoneme cannot be produced correctly if some future information is not known. As a matter of fact, the original HTS [12] system synthesizes speech on the basis of complete sentences. Thus, in the offline architecture of HTS, all the contextual information from the full input is used. In other words, the intrinsic delay of this architecture if considered in a real-time perspective is one full sentence, even if the actual computational time needed for the targeted speech synthesis is less.

In direct contrast to the original offline HTS approach and towards to the targeted real-time performative system, we achieved an intermediate step. The streaming version of HTS, pHTS, works on a phoneme-by-phoneme basis, i.e. it produces the samples for a given phoneme each time a new phoneme is sent to it as input. What is more, its delay, i.e. the number of future phonetic labels required for synthesizing the current phoneme, can be constrained.

More specifically, we use an already trained HTS system, in which we change the way the HMM, that models the sentence, is constructed from the phonemes. Consequently the way the sequences of spectrum and excitation parameters are generated. Here we have to clarify, that we still use pre-computed context-dependent phonetic labels that contain all the contextual information of a full text sentence, but all this information is not directly used for the feature synthesis step. Instead, for each given label, a sentence HMM is constructed, and for this single label the sequence of spectrum and excitation parameters are generated. The speech waveform synthesized from these parameters contains the synthetic speech that corresponds only to the given input phonetic label. In this way, the pHTS parameter generation is achieved by computing many locally-optimal paths for the decision tree, which, when combined make a sub-optimal total path, rather than computing one total optimal path for the whole text input sentence. In the current implementation, even though the complete sentence context information is available, the system only has access to information corresponding to a single label at each iteration.

2.1.2. Architecture of MAGE / pHTS

pHTS as a milestone led to the design of MAGE, which encapsulates pHTS in a framework for reactive HHM-based speech and singing synthesis. The architecture of MAGE/pHTS is based on the usage of four basic, different threads. The first and main thread is the so called *MAGE thread*, and runs all the synthesis part, communicates with the other three threads in order to receive the user control and adjusts the synthesis of the targeted output accordingly to the user input. The second thread is the *label thread* that is responsible for the correct communication between the user's context control and the *MAGE thread*, controlling how labels are handled, processed and stored. Then we have the *modification thread*, that is used to manipulate the prosody provided by the model. It allows the user to override, scale or shift the pitch, the speed / duration, the volume and the vocal tract length that are produced by the model on the fly. Finally there is the *audio thread* that carries the load of exchanging the speech samples created from the *MAGE thread* to the system call for audio samples. In other words, all three control threads; *label, modification and audio thread*; are in direct communication with the main synthesis thread. In Figure 2 we illustrate a diagram of the threaded architecture of the MAGE/pHTS.

For the implementation of the *label thread* and of the *audio thread* we needed to use lock-free, cache-efficient shared ring buffers [11], that provide fast data accesses among threads running in multi-threaded systems. In this case, the two ring-buffers allow us to synchronize the asynchronous input of the user and the synchronous demand for data from the synthesis thread, *MAGE thread*.

At this point, we achieved in the MAGE/pHTS framework to have a reactive voice synthesizer where the user can control the prosody of pre-defined context. The user can on the fly scale, shift or override the pitch and duration trajectories produced from the model itself, allowing a variety in prosodic control. Concurrently, it is possible to control the speed of synthesis, the volume of the outputted samples, as well as the vocal tract length parameter used at synthesis.

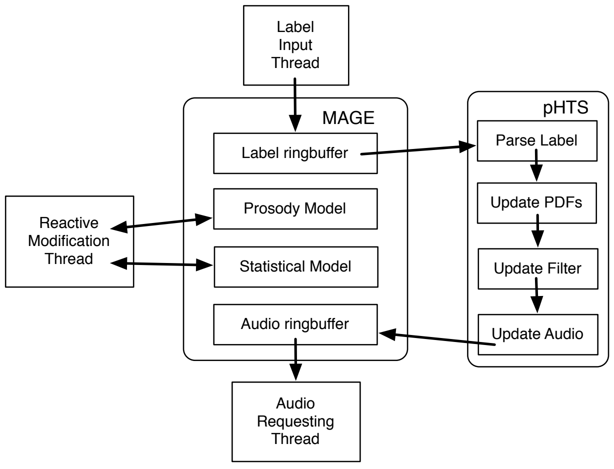

Figure 2: Threaded architecture of MAGE / pHTS

2.2. Prosodic Controller

The next step in this project is to implement an interface for this reactive voice synthesizer. MAGE/pHTS, which currently allows on the fly user prosody control, was integrated in an openFrameworks application. In this application we also integrated OSC messages, so that our reactive synthesizer will interact with its external environment. These messages contain prosodic information passed on the fly from the user from an external controller.

Here, as a controller we decided to use Handsketch, that allows gestural control interpretation. Handsketch was mapped as demonstrated in Figure 3. On the "fan" diagram we have control through the angle and the radius over the pitch trajectory produced by the model and over the speed of the samples synthesis. The pen, through pressure controls the volume, through button "one" controls if the pitch trajectory will be shifted or overridden and through tilt and button "two" controls the vocal tract length parameter.

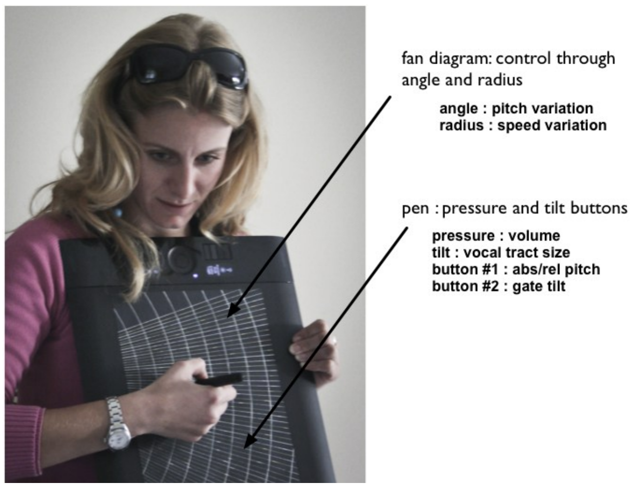

Figure 3: Mapping of pitch, speed, volume and vocal tract length on Handsketch as prosodic controller of MAGE / pHTS

Handsketch is connected to a laptop and via a Max/MSP patch [2] sends OSC messages to the MAGE/pHTS openFrameworks application. These messages contain the control information for pitch, speed, volume and vocal tract length, mapped as shown in Figure 3, are sent to the openFrameworks application that runs MAGE/pHTS, using the four threads as described above. When received, these messages are parsed and accordingly to the user control, the targeted speech samples are synthesized.

The full architecture of this prototype, using Handsketch as a prosody controller in the MAGE/pHTS framework for reactive HMM-based speech and singing synthesis, integrated as an openFrameworks application is demonstrated in Figure 4.

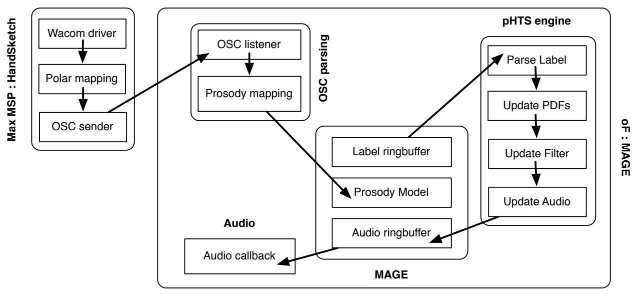

Figure 4: Architecture of the application that combines Handsketch as prosodic controller with MAGE / pHTS in reactive HMM-based speech and singing synthesis

2.3. Reactive Contextual Control

CoVoP 1.0 is a performative system, and by its definition it needs not only prosody user control, but also contextual user control. The problem occurring here, is that the contextual information used in MAGE/pHTS, inherited from the original HTS system, is based on pre-computed phonetic labels. These phonetic labels contain all the possible linguistic information from the targeted sentence.

2.3.1. Minimizing linguistic information

As mentioned, currently our system uses linguistic information extracted from the full textual input. In other words, even if our system runs as a reactive application, the data needed as contextual input are strongly dependent on future information that we do not have access to. This means that still the contextual input of our system introduces delay of one full text sentence, something against the performative definition. For the production of the phonetic labels the contextual factors taken into account are phonemes, syllables, words, phrases and the whole utterance, as listed below.

- phoneme
  - {preceding, current, succeeding} phonemes [1]
  - position of current phoneme in current syllable
- syllable
  - number of phonemes at {preceding, current, succeeding} syllable
  - accent of {preceding, current, succeeding} syllable

[1]The two preceding and the two succeeding phonemes of the current phoneme are taken into account

- stress of {preceding, current, succeeding} syllable
- position of current syllable in current word
- number of {preceding, succeeding} stressed syllables in current phrase
- number of {preceding, succeeding} accented syllables in current phrase
- number of syllables {from previous, to next} stressed syllable
- number of syllables {from previous, to next} accented syllable
- vowel within current syllable

- word
  - guess at part of speech of {preceding, current, succeeding} word
  - number of syllables in {preceding, current, succeeding} word
  - position of current word in current phrase
  - number of {preceding, succeeding} content words in current phrase
  - number of words {from previous, to next} content word
- phrase
  - number of syllables in preceding, current, succeeding phrase
  - position in major phrase
  - ToBI[2] endtone of current phrase
- utterance
  - number of syllables in current utterance

To solve this problem in the MAGE/pHTS approach we decided to train our system with "as little linguistic information as possible". Then creating on the fly phonetic labels, that correspond to user phonetic input, by building a reactive natural language processor (RNLP). Based on the default architectural delay of our system we attempt to define what "as little linguistic information as possible" is. As a matter of fact, we know, that MAGE/pHTS is a reactive system with at least one phoneme delay. This means that we can have access to at least one future phoneme, before its synthesis. The default label look-ahead buffer with one phonetic allows us to have access to two preceding, the current and one succeeding phoneme, plus past syllables. Taking that into account, we use the contextual factors listed below to re-train our model, so to be used at run time.

- phoneme
  - {two preceding, current, one succeeding} phonemes
  - position of current phoneme in current syllable
- syllable
  - number of phonemes at {two preceding, current, one succeeding} syllable
  - accent of {two preceding, current, one succeeding} syllable
  - number of syllables {from previous, to current} accented syllable
  - number of syllables to {next} accented syllables
  - vowel within current syllable
- word - phrase - utterance
  - No information

[2]Tones and Break Indices

As expected, there is quality degradation in the final output, compared to MAGE/pHTS using all the available contextual information, and of course compared to the original HTS system. Various samples of these approaches, can be found in [1]. We have to note here, that we have not assessed any objective or subjective measurements yet. However, currently this is the best possible approach that will enable reactive contextual user input and at the same time give very natural and intelligible output.

2.3.2. RNLP

Once the model was re-trained using the minimum linguistic information, it was time to implement the reactive natural processor module (RNLP). This RNLP module takes as input small chunks of phonemes passed from the user. These chunks are then parsed into alphanumeric phonetic labels, that contain only the linguistic factors mentioned above. All the factors needed for the phonetic label creation are correctly computed based on the parsing of the phonetic chunk input. Only the "number of syllables to the next accented syllable" factor cannot be estimated, and for that reason it is set to a random value between 1 and 5. Finally, these phonetic labels composed on the fly are then passed as input to MAGE/pHTS in order to produce the targeted speech samples.

As a matter of fact, RNLP is a first module that proves that it is possible to some extend to have reactive context control, that preserves the intelligibility and naturalness of the targeted output. Combining now MAGE/pHTS, Handsketch and RNLP, we have a first basic prototype, that enables full reactive control of context and prosody at the same time in the HMM-based speech and singing synthesis framework. Consequently, the prototype of MAGE/pHTS combined with Handsketch as a prosodic controller, described in 2.2, is now augmented so that it includes also the RNLP module.

2.3.3. RNLP & MAGE / pHTS & Handsketch

In order to use the RNLP module in our current system, first, it has to be integrated into MAGE/pHTS, and then to be combined with a context control interface. Initially, we integrated the RNLP module into MAGE/pHTS as a separate thread that controls the parsing of the incoming phonemes. This thread communicates directly with the *label thread* of MAGE/pHTS providing the phonetic label input to be modeled and synthesized.

Since RNLP was successfully integrated, we continued with the interface that controls the phonetic input. In order to control the phonetic content reactively, we developed a phonetic content sequencer. Using the paradigm of musical sequencers, our system has one, looping channel, with user-controlled duration, and whose content is not sound, but groups of phonemes. The software runs on an iPad and was written using openFrameworks.

We named these groups of phonemes "chunks", and they are loosely defined as units consisting of a few phonemes. Our aim in using chunks as the building block was to better facilitate user interaction. Single phonemes are hard for the user to handle in a performance context; and are usually less meaningful than chunks linguistically. We see using chunks as a faster and more intuitive way of creating patterns on-the-fly, if less flexible than single phonemes.

The loop is represented by a circle and a radius line is drawn every frame with its angle representing the temporal position of the loop. Thus we have a "radar-like" display. The available chunks are displayed with color-coded squares, like a palette (See Figure 5). The user sequences chunks by creating slices of desired chunks inside the circle, where the size (hence the angle) of the slice determines the duration of that chunk. The user can resize, reposition, delete, rearrange the order and change the content of the chunks on-the-fly.

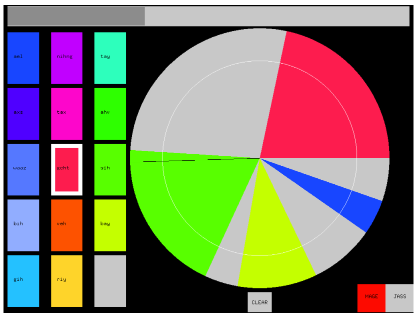
Figure 5: Phonetic Sequencer running on iPad

Whenever the beginning of a new slice is encountered at the current temporal position in the loop, An OSC message with the phonetic content and duration is sent to the RNLP module, at run time, to be parsed into the corresponding phonetic labels, and inputed to MAGE/pHTS for the synthesis of the targeted samples.

This design choice assumes a healthy network connection between the sequencer and the synthesizer, since the timing information is dependent on packet transfer. However, for our purposes, the interactivity gained by not sending the whole sequence at once far outweighs the minimal timing errors that might be caused by network problems.

The system can also receive new chunks during operation, allowing for a dynamic palette. As a proof-of-concept, we implemented a limited phonetic keyboard on an iPhone, which lets the user select from among the displayed phonemes and pack them into chunks, which are then sent to the chunk sequencer where they replace previously existing chunks.

Using this module we were able to augment the prototype described in 2.2 so that it provides both contextual and prosody reactive user control. In details, MAGE/pHTS with RNLP are integrated as an openFrameworks application that supports OSC messages. As described, the phonetic keyboard and sequencer applications running on iOS devices reactively control the context, and simultaneously Handsketch, via a Max/MSP patch sends OSC messages with the control information for pitch, speed, volume and vocal tract length to MAGE/pHTS synthesizer, mapped as shown in Figure 3. These OSC messages are received from an OSC listener, parsed into the expected format so that the prosody and context user control are correctly passed to MAGE/pHTS. Then the targeted speech samples are synthesized by taking the user control into account. Figure 6 demonstrates the architecture of this prototype.

Specifically, in the first version of the CoVoP system, we have one laptop connected on local network, that is running the reactive

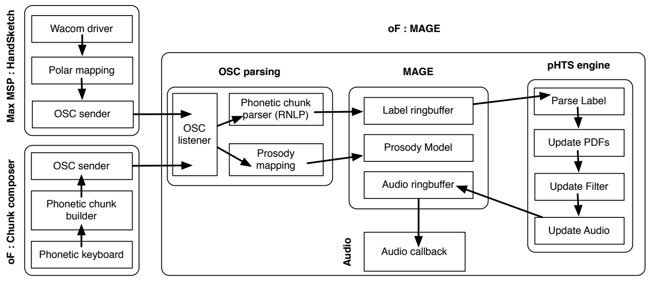

Figure 6: MAGE / pHTS, Handsketch and RNLP architecture that provides contextual and prosodic user control in reactive HMM-based speech and singing synthesis

synthesizer. The device controlling the context, in this case the iPhone, is directly connected to the synthesizer via the network. The device controlling the prosody, in this case the Handsketch, is connected to another laptop, that is then connected to the synthesizer via the network. Each user, through his device is able to send control OSC messages to the synthesizer, manipulating at will the artificial voice that is produced.

3. RESULTS

The architectural modifications of pHTS combined with MAGE bring a new layer providing an engine independent thread safe environment, resulted in the creation of a complete library for reactive HMM-based speech and singing synthesis. This library, MAGE/pHTS, was released as a first beta version on September the 30th, 2011 [5] and now it is an official extension library of the original HTS system [10], [9].

Furthermore, by combining MAGE/pHTS, Handsketch and RNLP, we created CoVop 1.0, a first basic prototype, proving that it is possible to have full reactive control of context and prosody at the same time, in the HMM-based speech and singing synthesis framework. Moreover, this first vocal puppetry system allows to multiple users, using different applications running on different platforms to control a single synthetic voice.

In details, the system consists of two laptops, one iPhone, one iPad and one tablet (Handsketch). One laptop, connected to a local network, is responsible for the voice synthesis and is running the MAGE/pHTS synthesizer as an openFrameworks application, waiting for the contextual and prosodic user input control. Two users operating iOS devices, control the contextual input. The iPhone runs the “phonetic keyboard” openFrameworks application which connects to the iPad running the phonetic sequencer, which sends chunks of phonemes via OSC messages to the synthesizer. The third user is controlling the tablet (Handsketch). Handsketch, is connected to the second laptop, that runs a Max / MSP patch, in order to receive the user control from the tablet and vis OSC messages to send the control also to the synthesizer. The synthesizer, receives and parses the control messages from both users, and accordingly produced the voice samples.

In order to have a complete evaluation of this system, there is need to conduct a user study for the collaboration between users and the final output result. Also it is essential to assess measurements for the intelligibility, naturalness and expressivity of the collaboratively produced artificial voice. Although the user case study and the evaluation of the system is left as future work.

4. CONCLUSIONS

We know, that managing all the parameters needed for synthetic speech production is overwhelming for one single performer. It takes an "expert" user and a lot of training in order to obtain some good results. Here we augmented MAGE / pHTS and developed a framework for multi-user participation for performative voice synthesis and extended its context in collaborative vocal puppetry. Each user has only one control over the synthesized voice, context or prosody. Consequently, with this prototype it is possible to explore this idea and see how vocal intelligibility, naturalness and even identity can be addressed as a social game, involving the participation of multiple users on their own device. As follows we want to conduct user studies and perceptive tests in order to evaluate the impact of such a collaborative and performative approach of voice synthesis on this prototype. Additionally to the system evaluation, we should also investigate the time scale issue. What is the optimal timescale to manipulate, in order to have a real-time system and at the same time providing flexible mapping and meaningful control to the user? As a matter of fact, currently our system is based on phonemes, but phonemes are long enough so that the prosodic control is not accurate enough, and short enough so that is too complicated for the user to obtain meaningful and natural context control. Moreover, the phoneme-based architecture introduces the problem of the constant vowel production. Sustained vowel, used especially in singing synthesis, can be synthesized in the current framework, but they sound very unnatural. Possible solutions to this problem is further architectural, modifications, either on the training level, or on the label input level, or even at the synthesis level.

Although, we believe that the development of a design space for performative speech and singing synthesis is possible in the HMM-based speech synthesis framework. In this work we prove that it is feasible to overcome the restrain of working at the sentence level, brought from the original system and transform HTS into a reactive system that provides on the fly, multi-user, multi-platform phonetic and prosodic control, enabling us to move towards a real-time system by proposing an appropriate time scale that respects the causality of interaction. MAGE / pHTS brings a very advanced and flexible voice synthesizer with built-in interactive properties, for both phonetic and prosodic control. It can bring also tools necessary for analyzing various situations involving voice production models, mapping strategies for interaction prototypes and integrations in heterogenous devices. Additionally it could validate implementations of the first social experience model around voice synthesis. In all cases though, quality evaluation measurements have to be conducted, before exploring all the possible fields of application of such a system.

5. ACKNOWLEDGMENTS

M. Astrinaki and O. Babacan's work is supported by a PhD grant funded by UMONS and ACAPELA-GROUP SA.

`numediart` is a long-term research program centered on Digital Media Arts, funded by Région Wallonne, Belgium (grant N°716631).

6. REFERENCES

6.1. Scientific references

[3] N. d'Alessandro and T. Dutoit. "HandSketch Bi-Manual Controller: Investigation on Expressive Control Issues of an Augmented Tablet". In: *Proceedings of NIME*. 2007. Pp. 78–81. P.: 60.

[4] T. Dutoit et al. *pHTS for Max/MSP: A Streaming Architecture for Statistical Parametric Speech Synthesis*. Tech. rep. volume 4, no. 1, pp. 7-11. QPSR of the Numediart research program, March 2011. P.: 59.

[11] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, 1997. Chap. Digital Signal Processors, pp. 503 –534. P.: 60.

[12] T. Yoshimura et al. "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis". In: *Proceedings of Eurospeech*. 1999. Pp. 2347–2350. P.: 60.

6.2. Software and technologies

[1] M. Astrinaki. *Performative HMM-based Speech Synthesis*. 2011. URL: `http://tcts.fpms.ac.be/~astrinaki/HTS_Results/details.html`. P.: 62.

[2] *Cycling74*. 2011. URL: `http://cycling74.com`. P.: 61.

[5] *MAGE / pHTS*. 2011. URL: `http://www.numediart.org/demos/mage_phts/`. P.: 63.

[6] *Open Sound Control*. 2011. URL: `opensoundcontrol.org/`. P.: 60.

[7] *OpenCV*. 2010. URL: `http://opencv.willowgarage.com/wiki/`. P.: 60.

[8] *openFrameworks*. 2010. URL: `http://www.openframeworks.cc/`. P.: 60.

[9] K. Oura. *HMM-based Speech Synthesis System (HTS) - Extensions*. 2011. URL: `http://hts.sp.nitech.ac.jp/?Extensions`. P.: 63.

[10] K. Oura. *HMM-based Speech Synthesis System (HTS)*. 2010. URL: `http://hts.sp.nitech.ac.jp/`. P.: 63.

KINACT: A SALIENCY-BASED SOCIAL GAME

Matei Mancas [1], *Radhwan Ben Madhkour* [1], *Dominique De Beul* [1],
Julien Leroy [1], *Nicolas Riche* [1], *Yves P. Rybarczyk* [2], *François Zajéga* [1]

[1] Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), Université de Mons (UMONS), Belgique
[2] Departamento de Engenharia Electrotécnica, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Portugal

ABSTRACT

In this paper we discuss the design of an intelligent game system capable of selecting the players who exhibits the most outstanding behavior from groups of people playing on a network. Players can be all in the same place (maximum 3) or in several locations if connected to the web. In that way, the total amount of players can be very large. Tests were made with 3 players in a single location and with 2 different locations of 2 players each. The system uses both static and dynamic features extracted from the upper part of the players' bodies such as symmetry, contraction index, motion index or height. Those features are extracted using the RGB-D Kinect sensor and their relative contrast and time evolution enable an adaptive selection of the most salient or different behavior without any complex rules. First users' feedback and eye tracking tests are shown and applications to social interactions are presented.

KEYWORDS

saliency, computational attention, game, kinect, bottom-up, top-down, behavioral features, eye tracking, social interaction

1. INTRODUCTION

Efficient and low-cost devices as the Kinect sensor opened new highways in gaming community. People detection and skeleton extraction in real life light conditions becomes a reality and more and more complex interactions are possible. Despite those new possibilities, still the interactions are achieved with pre-learnt and pre-programmed gestures. This means that if game scenarios are different from what is expected, the system cannot cope with the novelty.

An impressive human ability is to be able to attend interesting events or interesting behavior even if the situation is completely novel. In this paper, we implement and demonstrate a game which is able to select the active player in an attentive way. The system only uses players' behavior during their observation and a short memory of what they previously did to select in any situation the active player. The game is a network implementation using results from [9].

Beyond the ability to choose the active user, the system will be used to study the social organization of the player team, the influence of the players' localization which can be different between people playing together or through the network [13].

The characteristics we use are mainly related to 3D motion but also social/emotional features as those developed by Camurri et al. [3]. Two of the four features used namely the motion index (also called quantity of motion) and the contraction index come from those developments. The four features used here describe well the kind of gestures that humans would naturally perform in a game/social context like shaking hands, move, sit down...

This paper is organized as follows: section 2 globally discusses the game idea and its play modes. Section 3 details the features extraction while section 4 presents both bottom-up and top-down mechanisms. Section 5 deals with the implementation of our game and especially the network issues. Section 6 provides some results to tests which intend to capture players' interest for this concept and the accuracy of the attention system. Section 7 provides a conclusion and further developments of this game which intends to be a platform towards social behavior analysis.

2. GAME DESCRIPTION

2.1. Idea and Interface Implementation

We based our game on "World of Goo" [2], a puzzle game developed by 2D Boy [1]. The purpose proposed to players in World of Goo is to build a structure made out of balls interconnected by semi-rigid wires. This structure is used to touch a goal located somewhere in the scene.

We implemented a similar game by using the physical interaction Box2D library [4] and QT [10]. Figure 1 shows a screenshot of the game where two players almost touched the goal which is here a top-screen yellow ball.

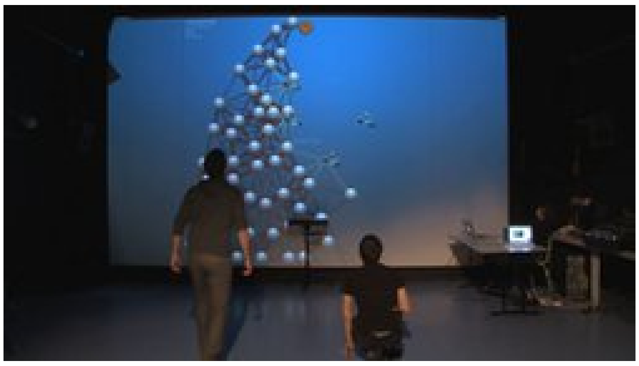

Figure 1: *Two players made a structure of white balls linked by elastic black wires and the purpose is to touch the top yellow ball.*

The difficulty of the proposed game comes from the team management. Each player can add a ball to the structure each 13 seconds. In the meanwhile, the structure evolves. Poor choices made previously destabilize it and can make it collapse. At the game start-up 50 balls are available and the players have to reach the goal with this amount of balls.

Different strategies are possible and each player has to choose between to make the structure higher to approach faster the goal or to consolidate it by adding balls at its basis to avoid the collapse of the structure. The game finishes when there are no more balls left or when the goal is reached.

2.2. Use Players' Body as a Controler

Two games modes were implemented. The first one is a simple sequential mode where each player has 10 seconds to put his own ball, and then the next one becomes active and so on. This is a classical implementation which uses no intelligence in selecting the active user.

The second game mode uses automatic saliency based on players' behavior to attend to a particular player and make him active. There are two distinct steps in each game turn. First, during 5 seconds, the players have to differentiate themselves from the others by adopting unfamiliar or strange poses (to attract the attention of the system). In a second step, as for the sequential mode, the active player has 10 seconds to place his ball on the structure.

While in the first mode at least two players are needed, in the second one at least 3 are needed to be able to find the player who is different from at least two others. Each player is represented graphically as a dashed circle. He controls the position of the cursor by moving in front of the Kinect sensor: the X and Z coordinates of the skeleton's barycenter of each player is mapped on the X and Y position of the cursor. In the second mode, behavioral features are extracted from the upper part of the skeleton of each player.

3. FEATURES EXTRACTION

The first step for an interactive machine is to extract features from the observed people. For that purpose, we use the Kinect sensor for its ability to extract smooth depth maps in complex illumination conditions. Libraries as OpenNI [8] are available to detect human silhouettes and extract anatomical features from skeletons tracking (Figure 2).

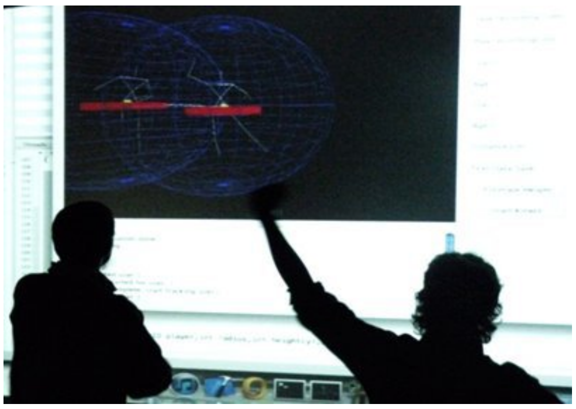

Figure 2: *Real-time skeleton extraction from players using OpenNI library.*

Four features are extracted from the upper body part only as the legs are much less stable in our implementation.

One of the four features is dynamic, namely the **motion index**. It is computed as the mean variation of the same skeleton points between two frames in 3D (on X, Y and Z). The barycenter point variation is extracted from the others (Equation 1) in order to keep only the body relative motion which will describe an excitement

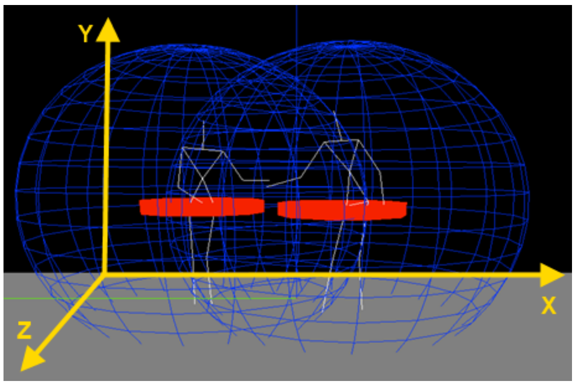

Figure 3: *(X, Y, Z) axes and skeleton extraction. Only the points of the upper body are taken into account.*

degree or movement transition of the body without any assumption on the whole body speed.

$$\begin{cases} D_{mk} = (\sum_{sk} |k_b - k_{sk}|)_t - (\sum_{sk} |k_b - k_{sk}|)_{t-1} \\ where\ k = x, y, z\ ;\ sk = skeletonpoints \\ and\ b = Barycenter \\ MI = \sqrt{Dmx^2 + Dmy^2 + Dmz^2} \end{cases} \tag{1}$$

A second feature extracted from the upper body part is a static feature, namely the **asymmetry index**. This feature is only computed on the X axis by differencing the distances between the barycenter point and the right shoulder, elbow and hand points with the left ones (Equation 2). This index provides information about the symmetry of the upper body.

$$\begin{array}{c} AI = \frac{\sum_{sk} |X_b - X_{sk_{right}}| - \sum_{sk} |X_b - k_{sk_{left}}|}{n_{sk}} \\ where\ n_{sk} = number\ of\ skeleton\ points \end{array} \tag{2}$$

The third extracted feature is the **contraction index**. This index is the ratio between the maximal distance between skeleton points on X axis and the maximal distance on the Y axis (Equation 3). This index tells us if the person is more or less contracted.

$$CI = \frac{|max(X) - min(X)|}{|max(Y) - min(Y)|} \tag{3}$$

The fourth and final feature is the **player height**. That one is simply computed by measuring the player barycenter Y coordinate.

After normalization, those four features provide a quite complete description about the level of excitement, and the upper body configuration of each player.

4. COMPUTATIONAL ATTENTION: WHO IS DIFFERENT?

The aim of computational attention is to provide algorithms to automatically predict human attention. The term of attention refers to the whole attentional process that allows one to focus on some stimuli at the expense of others. Human attention is mainly divided into two main influences: a bottom-up and a top-down one. Bottom-up attention uses signal characteristics to find the most salient or outstanding objects. Top-down attention uses a priori knowledge about the scene or task-oriented knowledge in order to modify (inhibit or enhance) the bottom-up saliency. The relationship and the relative importance between bottom-up and top-down attention is complex and it can vary depending on the situation [8].

Much research has already addressed the computational visual attention, but these works were devoted almost exclusively to 2D image analysis. Up to now little has been done on 3D data [11]. The arrival on the market of low cost 3D cameras opens up new prospects for developing algorithms for visual attention, which should make the move towards more realistic ambient intelligence systems.

4.1. Bottom-up Approach

As stated in [7] a feature does not attract attention by itself: bright and dark, locally contrasted areas or not, red or blue can equally attract human attention depending on their context. In the same way, motion can be as interesting as the lack of motion depending on the context. The main cue, which involves bottom-up attention, is the contrast and rarity of a feature in a given context.

The approach here follows the one in [9] but it has an optimal implementation in C++ instead of several pieces of block-processing software and it is also adapted to games and a network use. In our case, as the group of players can be small, the rarity computation is not relevant; therefore we only use the global contrast. Thus, the first step in this section is to calculate for the i^{th} feature ($f_{i,k}$) a contrast between the different users k.

$$C_{i,k} = \sum_{j=1}^{N} \frac{|f_{i,k} - f_{i,j}|}{N-1} \tag{4}$$

where N is the number of users. Once all the contrasts for a given feature $C_{i,k}$ between each user and the others have been computed, they are ordered in ascending order $C_{i,k,o}$ with o = [1 : N] from the maximum (o = 1) to the minimum (o = N). The difference between the two highest values is compared to a threshold T which decides if the contrast is large enough to be taken into account as in Equation 5.

$$\begin{cases} \alpha = 0 \ \ if \ \ |C_{i,k,1} - C_{i,k,2}| < T \\ \alpha > 0 \ \ if \ \ |C_{i,k,1} - C_{i,k,2}| \geq T \end{cases} \tag{5}$$

Only the features being the largest and passing this threshold T are merged with different weights (Equation 6).

$$C_k = \sum_{i=1}^{H} \frac{|C_{i,k} * W_i * \alpha}{H} \tag{6}$$

Where H is the number of features and α is given in Equation 5. The way the values of the weights W_i is set is described in the next section. This contrast C_k represents the bottom-up saliency for each user k. Saliency will be higher for the people exhibiting the most contrasted features within a given frame. The process of bottom-up attention is summarized on Figure 4 on a three-player scenario example. Each of the three players has its four features computed (in red for the asymmetry index, yellow for the contraction index, violet for the motion index and green for the height). The contrast computation and thresholded (Equations 4 and 5) is displayed in the second column. Finally the contrasted features combination (Equation 6) is explained in the third and fourth columns.

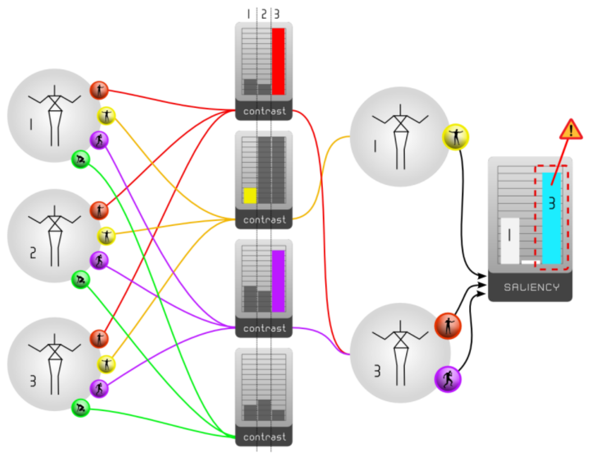

Figure 4: *Example of bottom-up saliency computation for 3 players. For each of the four behavioral features, a contrast is computed between the players. A threshold will eliminate features which are not contrasted enough between players (here the forth feature in green is eliminated). The player having more contrasted features with higher weights will be selected as the most salient (here the third player)*

4.2. Top-down Influence

The weights from Equation 6 can be adapted via the top-down component of attention. Those weights are initially set to be the same for all the 4 features which are used here. Then, the number of times a feature is contrasted enough for a given user ($\alpha > 0$), a counter is increased. The feature weight will be inversely proportional to its counter: if a feature i is often contrasted, its weight will be lower and lower, while a feature which is rarely contrasted enough will see its weight increased. This mechanism ensures a higher weight to novel behavior while too repetitive behavior will be penalized.

As an example, someone who will sit down for the first time (different height feature compared to the others), the height will have the maximum weight. If this person thinks that a different height is enough to attract the system attention, he will try again, but the more he tries again, the more the height feature weight will decrease as this behavior is no longer surprising. This top-down approach allows the system to learn how much a feature is novel and provides higher weights to the most novel ones. While the bottom-up approach will analyze in each frame which person behaves in a different way compared to the others, the top-down approach will also look for people who behave in a novel (or less repetitive) way from the beginning of the observation by taking into account a time memory.

5. IMPLEMENTATION: A DELOCALIZED GAME

While network implementations of games are interesting per se, in the case of an attentive system, the interest is even higher as the more people are involved, the more the attentive selection approach is relevant. We thus implemented our game in a network framework by using a client/server architecture.

5.1. A Client/Server Architecture

The range of each Kinect is quite narrow. After 4.5 meters, the depth map is less consistent and the skeleton extraction out of this depth map becomes very unstable. At 4.5 meters from the Kinect, the filed cover by the camera is about 6 meters large. The number of people who can be involved in a game using only one Kinect is restricted to 3 or 4 maximum. Due to the attention system, the players have to move a lot and perform gestures to gain the focus. This is an additional restriction on the number of users to ensure a good comfort during the game. If players are too close they might hit themselves and the skeleton tracking, in case of people touching their neighbors are much less reliable. That is why we tested a maximum of 3 players per Kinect in this game configuration, which is also the minimum number of users required by an attentional system.

Therefore, we prepared a software architecture that allows the usage of several Kinect cameras concurrently to cover a larger space [9]. But this technique is more complex to set up and needs several sensors, a calibration process and specific software. That is why, another solution was adopted here: the game was prepared to be able to be used in the same time in different locations over the Internet. In this way, the minimum number of 3 players has to be reached in the whole game which means that in one location, there might be only two players and in a second location just one player is enough. We made tests with a configuration of two different locations and 2 players per location. This set up is shown in Figure 5. The two locations are here two different rooms in the same building, but the data passes through Internet and not on the local network. Preliminary tests were also made between the universities of the city of Mons in Belgium and the city of Pilsen in Czech Republic distant about 800 kilometers.

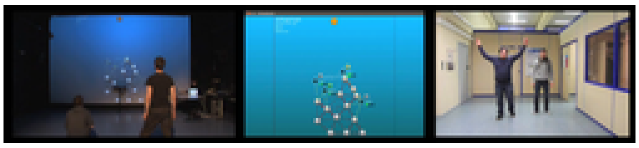

Figure 5: *Test with 4 players in two different locations (left and right image). While the visual feedback is visible for the first location in the left image, the middle image shows the visual feedback from the second location. Both structures are exactly the same.*

In order to implement this network approach, we had to choose between a peer-to-peer and a centralized architecture. The client-server architecture has been chosen for two main reasons.

The first one is because the game elements are managed by a physical engine: Box2D. Running this library on each client always led to inconsistencies. The longer applications were running the more difference appeared in connected clients because even a very small asynchrony leads to differences in the physical engine. In this case, placing the calculation of the graphical elements on a central server and broadcasting the elements to render to each client was the most effective solution.

The second reason is that each element involved in the game has a unique identifier. The management of these identifiers is better achieved by an entity controlling all the connected applications.

In the final version of the game (Figure 6), each client preprocesses the information. The retrieval of the skeletons of the players via OpenNI and the extraction of the behavioral features are executed on each client. The information is then packed in a message and sent to the server.

The server collects these messages and computes the final attention by comparing the behaviors from the different clients. After skeleton extraction on the client-side, each player has a local identifier. To clearly identify each player system-wide, the server distributes global unique identifier to each player. Each client manages the mapping between local and global ID's. In the same time, the server processes a new state of the physical world and manages the unique IDs of the players from all the clients. Eventually, it prepares messages containing the graphical elements and the users to draw on the screen customized for each client and send them back to the clients.

When clients receive the "frame" message, they control that the frame is newer then the one previously received and adapts the graphical display with it.

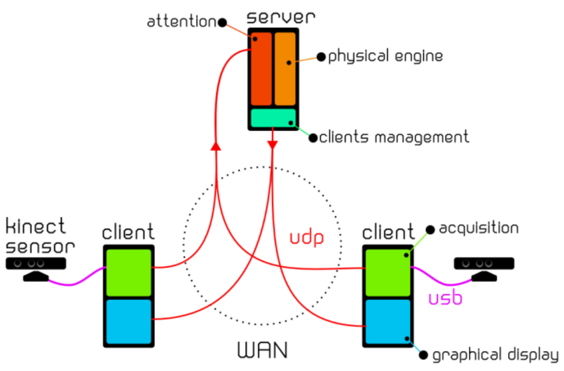

Figure 6: *The client-server architecture of the game. The clients make the signal acquisition and compute the behavioral features of each player. They send the players' IDs, position and features to the server. The server computes the attention and send back the active user and the changes into the display computed by the physical engine. Finally, the clients display the updated visual feedback.*

5.2. Mixed TCP/UDP Communication

Gaming over the Internet implies fast data transmission between all the components of the application. We wanted to guarantee 30 frames per seconds on the client-side so we had to prepare an optimized data transfer using the right protocol, TCP or UDP.

After testing both separately, we built a system using both for what they are good for. A TCP connection is used to start the client-server communication. No data is sent through this connection. It is used only to ensure the presence of both parties. If the connection breaks, the UDP connection will stop instantaneously.

A UDP connection is started directly after the validation of the TCP one. Its speed ensures a good frame rate and responsiveness. This solution is very effective and robust. Not sending any data trough TCP reduces to minimum potential loss of packages [12].

5.3. Players Behavior

The use of this network architecture has a lot of advantages by increasing the number of players without decreasing the game comfort. Nevertheless, as the attention algorithms is based on having a different behavior compared to the other players, any player should be aware about the others. This is obvious when playing in the same location, but much less when the game is delocalized. The network architecture allowed us to send in real-time feature packets, but sending videos is much more complex and this was not in our focus. Therefore we decided to use icons to show the feature which is the most representative of each player. Figure 7 shows the icons which are displayed for each user depending on his maximum feature. Those icons are explicit enough to be immediately understood by the players. If no feature is above a threshold, the gray static icon is displayed.

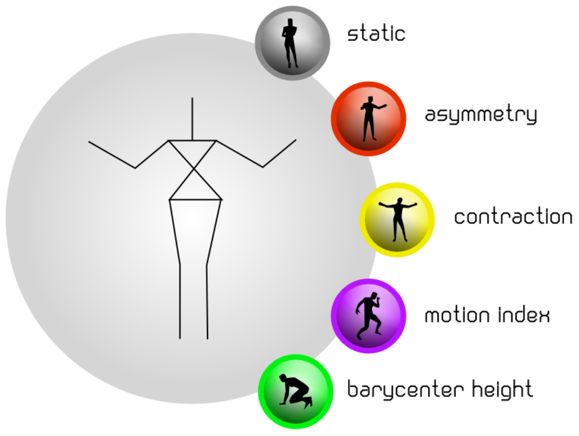

Figure 7: *Icons representing the dominant feature of the players. If there is no specific feature above a threshold, the static gray icon is displayed. Then, the feature which has the maximum intensity for each user is displayed.*

6. FIRST ANALYSIS

After the first tests, some analyses were carried out in the game. In a first step we asked the players to fill a questionnaire and we also used data which was available as output of the game. In a second step we used eye tracking to validate the attentive player selection system.

6.1. Users' Feedback and Behavior Analysis

In order to have a first idea about the players' feelings we set up a questionnaire asking them to score their game experience in terms of 1/Enjoyment, 2/ Isolation compared to the others, 3/ Comfort, 4/ Cooperation, 5/Leadership, 6/ Communication for both versions of the game: the classical sequential one, and the attentive one.

The level of comfort was higher in the sequential mode and the enjoyment comparable (probably also due to the fact that the game was more stable in the sequential mode than the attentive one during those tests). All the other factors were higher for the attentive approach. The players felt the attentive scheme as leading to much higher cooperation, leadership, communication and less isolation which is due to the fact that to attract the attention of the machine, the player has to take care not only about himself but also about the others.

Moreover, after the game is finished, it is possible to save in a XML file data about the features used, the number of balls, the joints between balls, the number of broken joints, the time needed to finish the game. This data showed that there is a faster progression in the learning of the game in the sequential mode than the attentive one which needs more time to be understood and even more to find efficient tactics. Indeed, this mode needs to build a social structure in the team with a leader and specialized players, while the sequential mode needs no specific involvement as all the players have the same role and the only solution to play is to wait your turn. In the sequential mode, people learn the game very fast and in only 3 trials, a team of 3 people went from 460 seconds and 39 balls needed to finish the game to 250 seconds and 23 balls. For the attentive mode, people went from 627 seconds and 36 balls to 501 seconds and 31 balls after a dozen of trials.

6.2. Attention Selection: Eye-tracking Study

To validate our attention-based selection system, we can test what would draw the attention of a human put in the same conditions as the proposed game system. For this purpose, we simulated a typical three-player scenario located in the same place. The RGB stream as viewed by the system was recorded and displayed on a screen where eye tracking [6] was performed using a commercial FaceLab 5 system. The 10 users who took part to the test were people from 25 to 35 years old, 3 females and 7 males. They were not aware about the game principle and they were just asked to watch in a natural way a video where 3 players were already segmented (black background to avoid stimuli coming from the background). No particular task was given to the users, thus their attention was mainly driven by bottom-up features even if top-down influence was also present as one of the three players in the video gazing to another player which also push the eye-tracked users to look in the same direction.

Within the bottom-up features, the static ones like color were used only at the very beginning of the observation, so during most of the video observation, users used bottom-up motion and position features. The results of this test (Figure 8) is surprisingly well correlated to contrasted people behavior, which is also the mechanism used by the proposed system. Of course this test setup uses one contrasted feature at each time and one of the three players is clearly contrasted, while in reality when people play, who is the most contrasted is sometimes not obvious even for humans.

The inter-personal distance is a very interesting feature which has clearly an effect on human attention (Figure 9) that we took into account in [9] but within the game with no network the feature was not easy to apply. The three players do not have enough lateral space to really change the inter-personal distances.

7. DISCUSSION AND CONCLUSION

In this paper we presented an original way of selecting an active user in a network game. Visual attention is implemented into the system which becomes able to choose the active player not by using pre-programmed paradigms but by adaptively finding the most outstanding player in any situation.

This work shows an important improvement potential. The first task will be to make more tests on user perception on the new game which is much more stable and simpler to understand than for the initial tests with the players' questionnaire. A better implementation and deeper analysis of the data saved into the XML file concerning the ball structure construction is also needed. From

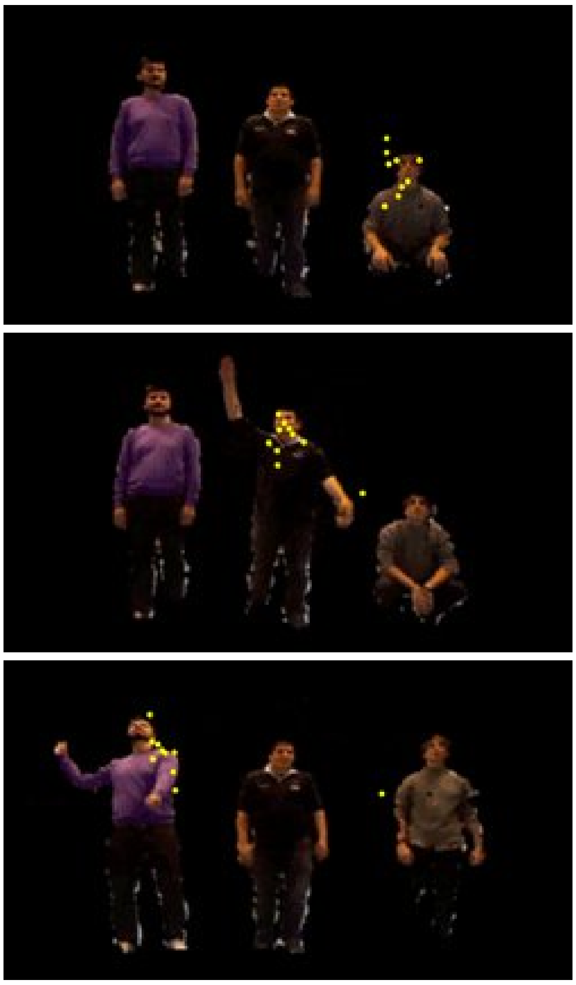

Figure 8: The 10 user eye gaze (yellow points) overlaying on three different frames of the video. For each case the player achieving a different bahvior is gazed by the majority of the 10 users.

the XML file it is for example possible to extract the characteristics of the ball structure shape (Figure 10). A very large basis and a displacement of the balls during its construction may signify that the structure collapsed during the game, so the construction was not very efficient. Colors can be assigned to each user to see chat tactics were used to get information about the tactics used for example.

Within the new network development, it would be interesting to integrate the inter-personal distance feature in the virtual space of the game and monitor people positioning when they are located in the same place or in different locations. Surprising people positioning (one user far from the group for example) can be very attracting from a saliency point of view.

On the game level, we are planning to integrate a team versus team mode. At the application start-up, the players will have to select the number of teams and the type of playground. Two types of playgrounds will be proposed: “shared” or “boxed”.

The type “shared” proposes only one playground and one goal. The gravity will not be linear but radial. The balls will fall towards the borders of the widow and reach the goal which will be located in the playground center. The players will be able to interact with

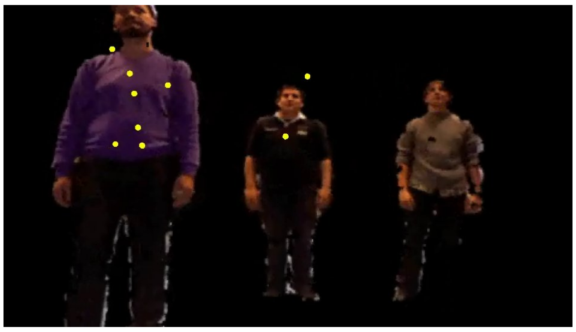

Figure 9: The inter-personal distance seems also to be taken into account during the eye-tracking tests.

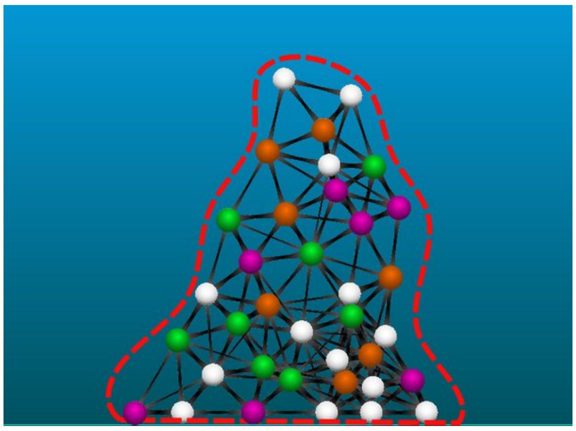

Figure 10: Ball structure properties extraction (area, height/width ratio, density, . . .). Colors correspond to players who dropped the balls.

other teams. The type “boxed” proposes a playground by team. No interaction with other teams will be possible. An example can be seen in Figure 11 with four teams.

The current version of the game focuses on players’ interactions. All the players having a common goal, we can track the reaction of each player against the team. In the next versions the concurrence aspect induced by the existence of several teams with different or the same goal will be interesting to analyze.

By making people play we can test in fact the social interactions between them in several contexts: in the same team with a common goal, in several teams under concurrence. The interesting aspect is that, through a game, it would be possible to observe the social evolution within a team, the emergence of a leader, the emergence of specialized players, the efficiency evolution of the team if several leaders are placed into the same team.

Finally another interesting topic is in the study of the social interactions between players who are co-located or located in different places. The first tests seem indeed to show that even if there is a common goal and the players were told to play together, people who are not located in the same place tend to act like concurrent teams and to see which location participated the most into the final structure. This is an interesting social behavior emergence which was not planned at the beginning.