Is This Guitar Talking or What !?

M. Astrinaki, L. Reboursiere, A. Moinet, R. Graham, N. d'Alessandro, T. Dutoit

Circuit Theory and Signal Processing Lab, University of Mons, Belgium

{maria.astrinaki, loic.reboursiere, alexis.moinet, nicolas.dalessandro
nicolas.dalessandro, thierry.dutoit}@umons.ac.be, info@rickygraham.net

Abstract

In this project, we explore the possibility for an augmented guitar to be used as a controller for the expressive manipulation of reactive synthetic speech. This idea comes at the intersection of two research frameworks. On the one hand, we aim at augmenting the electric guitar by extracting guitar playing techniques directly from the guitar sound, through an hexaphonic pickup (one microphone per string). On the other hand, we develop the MAGE software, a unique set of tools for generating high-quality HMM-based speech synthesis in a reactive way. Bringing these two technologies together allows us to explore various mappings between the controller and the speech synthesis, and propose expressive solutions.

Index Terms: HMMs, speech synthesis, reactive control

1. Introduction

Speech is one of the richest and most ubiquitous modalities of communication used by human beings. Vocal expression involves complex production and perception mechanisms. Conversation is a highly interactive process, with complex timings and wide-ranging variations of quality. It is known that speech production properties have a deep impact on perceived identity and social cues [1]. This critical role of speech production in our life makes anybody an expert listener. The synthesis of artificial speech has been explored for decades to use in many applications, from the purely functional level to artistic exploration. However, human's natural expertise in listening to spoken content makes speech synthesis a really complex problem. Recent synthesizers have made great progress in terms of intelligibility and naturalness but they are still not providing a completely convincing vocal experience to users, neither an expressive tool for artists. In this Section, we describe the various research problems that lead to this situation, as an introduction to our concept of *tangible speech synthesis* and the new speech synthesis system that we present.

1.0.1. From Speech Production to Social Cues

Understanding voice production requires an interdisciplinary approach. It can be seen bio-mechanically as pulmonary pressure being applied on tensed vocal folds and the manner of placing the various articulators in the vocal tract, such as tongue, jaw or lips [2]. The acoustics of this phenomenon suggest that the volume velocity waveform generated by the vocal folds vibration propagates through pharyngeal, oral and nasal cavities with time-varying resonance frequencies, called *formants* [3]. Linguists are interested in how these formants vary over time and their relation with vocal tract postures that we continuously browse when we speak, called *phonemes* [3]. They also study, what is called *prosody* [3], how fundamental frequency and amplitude of vocal folds vibration vary over time, as well as phoneme durations. Phonetical and neurological studies show that upcoming speech fragments are planned ahead by the brain, and then corrected on-the-fly by continuously evaluating acoustical and sensorial distances from the plan [2].

This active research community has been making outstanding progresses over the last decades, but it seems that some aspects of speech production remain misunderstood. For example, we do not have an exhaustive model for vocal folds vibration, because observations *in vivo* are nearly impossible. There is also a big debate in how speech production is influenced by the context, such as speaker's emotional state, listener's reaction or other surrounding stimuli, because real-life measurements are intrusive. These issues result in an elusive mapping between parameters of existing production models and real social impacts of speech, such as intents or emotions, as illustrated in Figure 1.

1.0.2. Text-To-Speech and User Interaction

Early research in speech synthesis was firstly trying to model the physiology of speech production, then manipulating the models according to what was observed in the vocal tract or on the speech spectrum, such as the well-known source-filter model [4]. The generalization of computing in speech synthesis research progressively benefited to a new approach, detached from physiological roots, and focused on the systematic conversion from text to speech. New algorithms from the early 1990s, based on waveform segmentation and concatenation [5], made a remarkable leap forward in term of intelligibility and

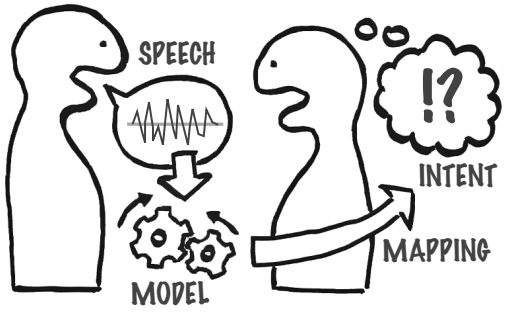

Figure 1: Major obstacles remain in order to accurately map between parameters of speech production models and perceived intents or emotions.

naturalness, shifting the physiological trend to focus on speech simulation.

Text-To-Speech (TTS) systems have a common architecture and work in two steps. First, text is converted into the *narrow phonetic transcription* by the Natural Language Processor (NLP), containing phonemes and prosody. Then this information is converted into speech sound by the Digital Signal Processor (DSP) [3], as illustrated in Figure 2.

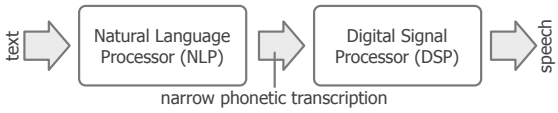

Figure 2: Text-To-Speech: NLP converts text into the narrow phonetic transcription, then DSP converts this transcription into speech sound.

At the time TTS became the main trend, it was not evident that computing would go mobile so massively. Retrospectively, we understand how the design choices underlying TTS – text input and black-boxed generation of resulting speech – have anchored its use to reading text on desktop computers. However, mobile computing relies on ubiquitous sensing of user's context, and user interaction tends to become more natural. Therefore, high-quality speech synthesis seems to have major issues in being used "in the wild". Nowadays there are two main application types that are prevented to expand because of these limitations:

1. *Context-reactive speech synthesis*: our modern life is encountering an increasing amount of virtual agents, on the phone, in the car or in public spaces. Ubiquitous computing brings these systems to gather a lot of information about our context: location, light/noise conditions, movements, social connections, etc. Most of this information is dynamically changing. However, embedded TTS makes few sense of these context changes, because the speech production properties can barely be altered, even less on-the-fly.

2. *Performative speech synthesis*: artificial speech can be generated from gestural performance, rather than pre-typed text. This approach has many applications, such as silent speech communication [6], speech production replacement for voice-impaired users, etc. This technique is also interesting for artistic purposes, as speech is a common medium used in various disciplines. These situations require a major breakthrough in speech synthesis techniques in order to create speech sounds from non-textual fast-changing inputs.

1.1. Guitar as the Controller

One essential aspect involved in developing a system for performative speech synthesis is the design of the controlling device. One approach to this research is to consider that the purpose is to design a new instrument for musical expression, or NIME, here applied to the speech signal. A common concern in NIME research is the lack of human practice associated to new instruments. This issue can trap the design process in a loop where the lack of a good device prevents good practice to appear and the lack of good practice prevents a good device to emerge. In order to avoid such a deadlock situation, many people have started their NIME design from an existing instrument. Indeed the existing practice can be reused and then extended for the new purpose. Due to its wide availability and contemporary history, the electric guitar has been a good candidate for such a strategy [7, 8].

In this project, we have decided to use the electric guitar as the input device for controlling speech synthesis. It is motivated by the above-mentioned intent to reuse and extend the existing guitar playing techniques for our new purpose, but also because we wanted to further explore the Guitar As Controller hardware/software platform that we have built in the lab for the last 2-3 years.

1.2. Outline of the Report

In this report, we start describing the speech synthesis technique we use in this project, called HMM-based synthesis in Section 2. Then we describe the main modifications that we have applied to state-of-the-art HMM-based speech synthesis in order to create a fully reactive and controllable sound synthesis system, called MAGE, in Section 3. Section 4 explains the modifications applied on the existing Guitar As Controller toolbox and new mapping strategies that have been developed specially for controlling speech. Finally we discuss the prototype that we could assemble and test during the workshop in Section 5.

2. HMM-Based Speech Synthesis

Nowadays, the most common approach for achieving high quality natural speech synthesis is the corpus-based unit selection technique. In principle, this method relies on runtime selection and concatenation of speech units from a large speech database using explicit matching criteria [5]. In direct contrast to the dominance of corpus-based unit selection, there is an increased interest for statistical parametric speech synthesis [9]. Statistical parametric speech synthesis is based on an model-based parametric framework, where speech is generated by averaging sets of similarly sounding speech segments. Indeed, instead of using real speech samples at runtime, context-dependent HMMs are trained from the databases of natural speech, and then speech waveforms are generated from the HMMs themselves.

2.1. Core architecture of typical system

In a typical statistical parametric speech synthesis system, the pre-recorded database is analysed, various production parameters are extracted spectral envelopes, fundamental frequency and duration of the phonemes - and used to train statistical models. Usually a maximum likelihood (ML) criterion is used to estimate the model parameters [10]. Later these models will generate the speech parameters for a given targeted text input. Speech waveforms are produced from the parametric representations of speech with typical speech synthesis techniques: subtractive synthesis [11] or harmonic plus noise [12]. Any generative model can be used, however most widely used are Hidden Markov Models (HMMs) [13], and this approach is known as HMM-based speech synthesis [10].

In Figure 3 we present an overview of a typical HMM-based speech synthesis system (HTS) [14], which consists of a training and a synthesis part.

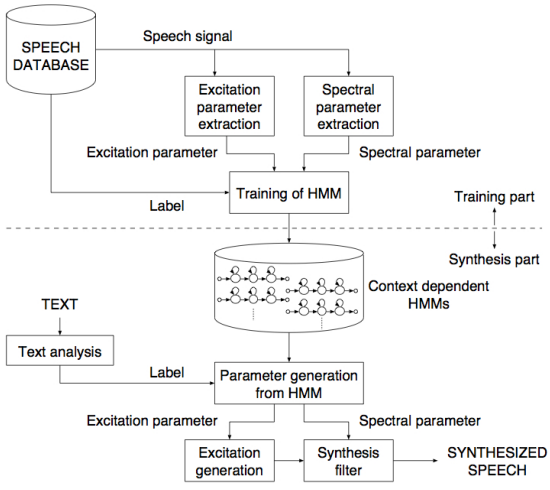

Figure 3: *Overview of a typical HMM-based speech synthesis system (HTS) [14].*

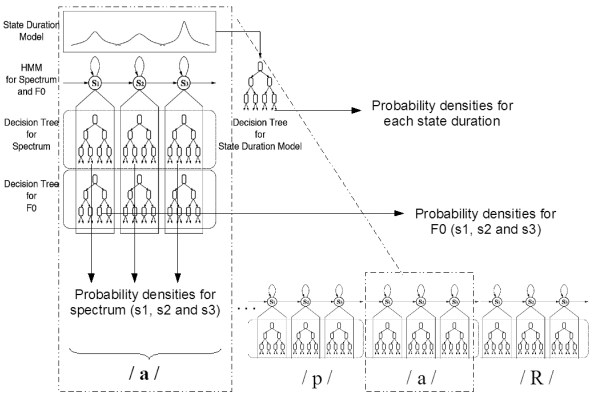

Figure 4: *Decision trees for context clustering [18].*

During the training part both spectrum (mel-cepstral coefficients [15], and their dynamic features) and excitation (logarithmic fundamental frequency (logF0) and its dynamic features) parameters are extracted from a natural speech database. These parameters are then modeled by context-dependent HMMs, taking into account phonetic, linguistic and prosodic contexts. Multi-space probability distributions (MSD) [16] are used to properly model the excitation parameters which is a variable dimensional parameter sequence with non continuous pitch values in unvoiced regions. In order to model speech temporally, HMMs model the state duration densities by using multivariate Gaussian distributions [17]. So to handle all the contextual factors, such as phone identity and stress or accent related factors that affect the targeted synthetic speech output, decision-trees based on context clustering techniques [18] are used, as shown in Figure 4. Magnitude spectrum, fundamental frequency and duration are modeled independently, therefore there is a different phonetic decision tree for each of these features [19].

At the synthesis part, the input text is analyzed and converted to a context-dependent phoneme sequence. Then by concatenating the context-dependent HMMs according to the generated context-dependent phoneme sequence an HMM utterance is constructed. This HMM utterance is used to generate the sequences of spectral and excitation parameters [20], and by using excitation generation and a speech synthesis filter (e.g., mel log spectrum approximation (MLSA) filter [21]) the final speech waveform is reconstructed.

2.2. Advantages

Compared to the unit-selection synthesis, statistical parametric synthesis offers significant advantages not only in controlling the synthesis procedure but also in being much more flexible due to the well defined statistical modeling process. One of its main advantages is the flexibility in changing its voice characteristics and speaking

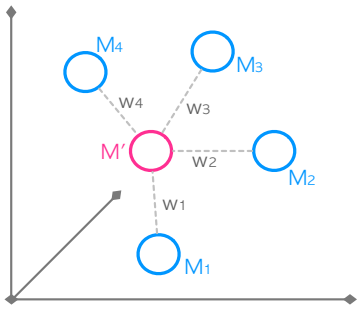

Figure 5: Speakers individuality modeled by HMMs, Mi, where Wi denotes the interpolation weight between the existing models in order to produce the untrained voice characteristics, M'.

styles by simply transforming the model parameters. This is possible through :

- adaptation, mimicking voices by means of maximum a posteriori (MAP) estimation [22] and maximum likelihood linear regression (MLLR) [23].
- interpolation, mixing voices and synthesize speech with untrained voice characteristics [24] as shown in Figure 5.
- eigenvoice, producing new voices, [25].
- multiple regression, to control voice characteristics intuitively [26].

Other advantage is that HTS can easily be adapted in different languages, contexts and applications [14]. Compared with unit-selection synthesis, statistical parametric synthesis has a very small footprint, just a few MBytes [27], since there are no real speech samples used, only the statistics of acoustic models are stored and fewer tuning parameters since both modeling and synthesis processes are based on mathematically well-defined statistical principles. However the main drawbacks of statistical parametric synthesis is the quality of the final synthesized waveform. The reason for this quality degradation seems to be the vocoders used, the acoustic modeling accuracy and finally the over-smoothing [28].

3. Reactive HMM-Based Speech and Singing Synthesis

As the new trends in understanding expressivity in speech are being explored, and the need for real world speech and singing synthesis applications such as entertainment and gaming applications, silent speech communication and performing arts application as well as assistive applications for speech impaired people. However one might notice that a real solid platform for performative speech and singing synthesis is missing. The challenges of such a platform though are on the one hand the reactive production of expressive speech and the adaptability and latency of the speech synthesis and on the other hand how to provide a meaningful gestural control mechanism.

In traditional HMM-based speech synthesis, as described in Section 2, a certain amount of text is required in advance to be processed and converted into speech as a whole target but during this text to speech conversion process any external influence is rather limited. This limitation prevents to adapt to any external solicitation within the sentence as it is being synthesized. Thus, we decided to build MAGE, which proposes the generation of speech parameters within a smaller look-ahead window rather than the whole available text. This approach allows to infer on speech outputs at various production levels and time scales. However, such a system has totally different requirements than the original one; it needs to have a reactive programming architecture, and to be both listener-specific and context-aware. To our current knowledge, MAGE is the first platform for reactive programming of speech and singing synthesis able to address these issues, allowing reactive prosodic and contextual user control.

3.1. Short-Term Speech Parameter Trajectories

As inherited form the original system HTS; MAGE also has a training and a synthesis part. For both systems the training part is identical, as described in [28], but their fundamental difference lays in the synthesis part. During the synthesis time of HTS, the input text is analyzed and converted to a context-dependent phoneme sequence, then according to this sequence, context-dependent HMMs are concatenated, constructing an HMM utterance. Then this HMM utterance is used to generate the sequences of spectral and excitation parameters by maximising the probability of the speech parameter sequence [28]. Consequently, in HTS the smallest accessible time scale is the complete targeted word sequence. Finally, the targeted speech output is reconstructed by using excitation generation and a speech synthesis filter, here Mel Log Spectrum Approximation (MLSA) filter [21] with pulse-train or white-noise excitation.

In direct contrast to HTS synthesis part, in MAGE the observation window is reduced, from all the available phoneme sequence to just a sliding window of two phonemes. More specifically, as the available phonemes are being streamed as input to MAGE, only the new phoneme and the previous phoneme are used to concatenate the context-dependent HMMs and construct an HMM utterance. Then this HMM utterance, consisting only of the context-dependent HMMs of two phonemes is used to generate the corresponding sequences of spectral and excitation parameters. Then, by using the maximisation in Equation 2, as described in [28] the speech parameter trajectories are generated. A result of the reduced ob-

servation window approach is that the generated speech parameter trajectories do not correspond to the overall maximum of probability (HTS), but only the concatenation of locally-maximized speech parameters (MAGE).

$$q^\star = \underset{q}{\operatorname{argmax}}\, P(q \mid \boldsymbol{\lambda}^\star, \hat{\boldsymbol{T}}) \tag{1}$$

$$\hat{\boldsymbol{O}} = \underset{O}{\operatorname{argmax}}\, P(O \mid q^\star, \boldsymbol{\lambda}^\star, \hat{\boldsymbol{T}}) \tag{2}$$

where O and q are respectively the sequence of speech parameters and the sequence of states, $q^\star$ and $\lambda^\star$ respectively the estimated sequence of states and the concatenated left-to-right context-dependent HMMs of the 2-phoneme window, $\hat{O}$ the sequence of locally-maximized generated speech parameters, and $\hat{T}$ is the time frame corresponding to the first label of the 2-phoneme window on which $\hat{O}$ is computed.

By using a 2-phonemes window for the speech parameter trajectory generation, MAGE opens the enclosed synthesis loop of HTS, and the initial accessible time scale of the sentence level in now reduced to the phoneme level. As Figure 6 illustrates, when there are two phonemes for the sliding window, the speech parameter trajectories are generated and the corresponding speech samples are synthesized and stored in independently. By providing the needed real-time audio architecture, as it is described in Section 3.2, the audio samples can be synthesized, altered and streamed on the fly, only with one phoneme delay. In other words, it is possible to influence the generation of the speech parameters and change the corresponding speech samples with a delay of only one phoneme. As follows, the spectral envelopes, the phoneme durations as well as the pitch curves can be modified as speech samples are being synthesized and affect the final output with only one phoneme delay.

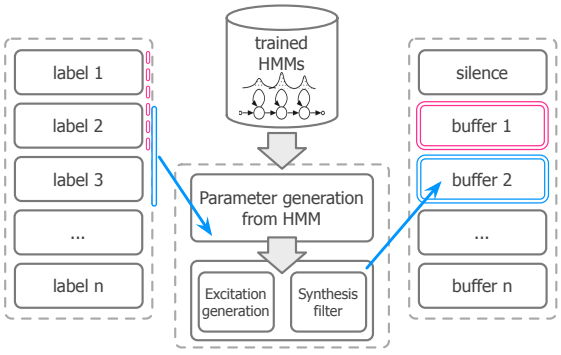

Figure 6: MAGE *synthesis, using a 2-phoneme sliding window to generate the speech parameter trajectories and audio buffers.*

3.2. The MAGE platform

MAGE is a platform for reactive HMM-based speech and singing synthesis. It provides an API for reactive programming in C/C++, aimed at being included in realtime audio softwares. MAGE is thread safe and engine independent. It is the shell that provides to HTS the needed real-time audio architecture so that the targeted speech samples can be reactively manipulated. As illustrated in Figure 7, MAGE consists of the label thread, that control how the inputed phonemes are streamed to be processed, the parameter generation thread, that generates the sequences of spectral and excitation parameters by maximizing the probability of the speech parameter sequence and finally the audio generation thread that generated the targeted speech samples. During runtime MAGE will allow the input of user controls so that the speech samples finally outputted in to the audio thread can be reactively controlled.

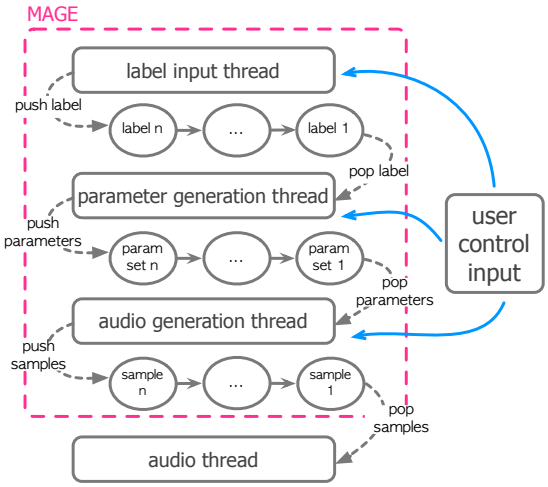

Figure 7: *Multithread architecture of* MAGE*: speech synthesis thread makes the connection between the user control and the audio thread.*

Since MAGE can be easily imported in various frameworks and it can be simply combined with OSC-enabled sensors, it allows fast and easy prototyping. Additionally it provides easy context and prosody controls over the synthesized voice. Contextual control is implemented based on the label thread and how the available phonemes are streamed into MAGE, while prosody control is based on the reactive manipulation of the pitch trajectories, phoneme duration and vocal tract parameter.

MAGE comes as a consequential implementation, following the idea of performative speech synthesis, as a way of looking beyond Text-To-Speech (TTS). It is an interdisciplinary project, addressing problems in the fields of speech processing, linguistics and human-computer interaction (HCI) and it attempts to bring a common platform to address their problems. MAGE is targeted to be used for approaching and understanding longterm ques-

tions in speech production, such as degrees of coarticulation, speech motor control, speech planning, intonation, voice quality, speech time scales, etc. through gestural control and interactive interfaces, mainly through mobile and social computing.

4. The guitar as a controller

4.1. Introduction

The first time time guitars were used as controllers can be correlated to the appearance of MIDI guitars. On those systems, an hexaphonic pickup (1 pickup per string) enables polyphonic pitch and amplitude tracking so as to drive synthesizers or samplers in order to extend the sounding possibilities of the instrument. The guitar became, thus, a MIDI controller.

Later, the augmented instruments term and field appeared pushing further the notion of controller by embracing the more global notion of gesture and more specifically of musical gesture. Anything that gives the user controls on the produced sound is a musical gesture.

In [29], the author defined the three types of musical gestures:

- effective: physical technique employed by the agent to produce sound (picking, fretting, etc.)
- ancillary: accompanying physical body movements
- figurative: note attack, scalar instrumental structures, melodic contour

These three types of musical gestures are as many possible ways to have control on the resulting sound. Regarding the guitar, the three types of musical gesture have been assessed in many different ways expanding each time the controller notion.

Ancillary musical gesture, e.g, have been used and mapped to audio effects in [7] and [8]: in the first example, an inclinometer on the head of the guitar was controlling the volume of the effect. On the second example, pressure sensors have been added to the rear of the guitar to catch information on the movement the guitar does around the belly while the guitarist is playing. This information was then mapped to the different parameters of a wah-wah like effect.

Figurative musical gestures have been used in [30] to create a system which analyses the melodic structure it receives and applies different mappings depending on which pitch contour is detected. Different effective gestures have been analysed and detected in [31] and [32]. In [33], we developed algorithms for most of the major guitar playing techniques, i.e, hammer-on, pull-off, slide, palm muting, bend, natural harmonic notes as well as detection of the plucking point.

During this project, we mainly worked on the first (effective) type of gesture using the algorithms developed in [33] and implemented in Max MSP software. The third type (figurative) of gestures was taken into account, but due to a lack of time this type of gesture was not included in the final mapping.

4.2. Guitar playing techniques detection and optimization

The detection of the playing techniques detection was made possible by the use of an hexaphonic pickup (one pickup per string, i.e six separate signals), e.g [1]Roland GK-3, which implies that six separate analysis had to run at the same time. The Max MSP implementations of the algorithms were first done one by one in order to test separately their real-time efficiency. However, once grouped together and working at the same time, the CPU consumption increased dramatically.

To address this problem of CPU consumption, the first and main step was to gather every detections in one patch to define only the needed FFT. Indeed `fft~` Max MSP object is quite CPU consuming and the first implementations of the algorithms were using nearly 6 FFT for each playing technique. Gathering all the algorithms dropped down the number of FFT at 12, 2 per string: one was from `fft~` Max MSP object and the other one from the `sigmund~` external (third party object) used for the pitch extraction. It has to be noticed that a FFT developed in C language into an external, i.e `sigmund~`, is less consuming than an `fft~` object.

The second element that has been tested was the difference between the use of abstractions (instances of a patch, i.e a C++ object is an instance of a class) and the use of the `poly~` Max MSP object which manages the polyphony and its own DSP consumption. With `poly~` several instances of the same patch can be defined and their processing activity can be totally taken off the DSP chain and CPU consumption with the *mute* message. Muting the different playing techniques algorithm decrease the CPU consumption significantly, however this solution is useful only in case of someone not using all the detections, but not in a complete use case. This functionality remains therefore useful but doesn't fit all cases. Another property of the `poly~` is that the DSP treatment can be specified to be done on all the processors of the comupter by using the *parallel 1* message.

However, despite all these options, the use of six abstractions instead of one `poly~` object with 6 voices remains faster. In his lightest version (no graphical object for output visualization), the detection algorithms used 6% of CPU for the DSP part against 9% with `poly~`. In a more friendly version (use of vumeter to monitor the signal and of other graphical elements to monitor the out-

[1]http://www.roland.com/products/en/GK-3/

puts of the detections), the consumption increases equally until, respectively, 10% and 13%.

In both cases, the main element which dropped down the CPU consumption was the decrease of the number of computed FFT. Indeed, before gathering all the detection, CPU consumption was around 50%, 60%. Depending on the situation the two solutions cited above can be used. If all the detections are used, the solution with the 6 abstractions fits best. In the case of not using all the detections, the solution using the `poly~` fits better.

Figure 8 shows the GUI of the patch used for the guitar playing techniques detections.

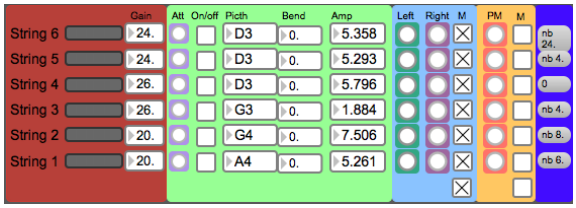

Figure 8: *Patch for guitar playing techniques detection (from left to right): input of each strings with adjustable gain, attack, pitch and note on / note off detection bend and amplitude tracking, left / right-hand attack discrimination, palm muted notes and harmonics detection.*

4.3. Mapping

It has to be noticed, before going into the details, that these mappings have been chosen in terms of the resulting sound they produced. Indeed, they were used in an improvisation framework where guitar and synthesized voice were mixed. Flexibility and sound quality were therefore what led the mappings designs. Moreover, at the time of the mappings' conception, no controls over the structure of the synthesized sentence or text was available, therefore the guitar could not have been used as a moving playhead or any similar type of control.

One of the main element used in the mapping between the guitar and the MAGE synthesizer is a 2D-interpolation tool (`node` Max MSP object). With this tool the states of the interpolation are defined by circles and distances are outputted as functions of the position of the cursor regarding each one of the states. The size of the circles is used as a weight applied to each states in the distances computation: the bigger the size, the bigger the state's influence. In [34], a similar tool has been developed and described. This interpolation tool can be linked to the `pattrstorage` object (store and recall presets) in order to easily use 2D-interpolation to move through defined presets. In our case, several voices with different parameters were defined as presets and the interpolation helped navigating between these voices, creating in-between voices when the cursor is in-between presets.

The second element that we added was a trajectory tool to record and play back movements (motion of the cursor in a lapse of time) into the 2D world of voices. Subjectively interesting (in terms of generated sound) trajectories were then recorded to be used as guitar-controlled materials. Figure 9 shows these 2D-interpolation and trajectory tools.

Three different mappings, controlling the trajectory tool, were defined. On each one of those mappings, volume of the guitar has been mapped to the volume of the synthesized voice, in order not to have sound unless the guitar is played. The three mappings are detailed below:

- first mapping: the note on the 3rd fret of the 4th string (F) loads a defined trajectory and the note on the 3rd fret of the 5th string (C) plays it. Any bends played on the 2nd and 3rd string is mapped to the vocal tract length; the bigger the string is bent, the smaller the vocal tract length is.
- second mapping: any normal attack on the 6th string chooses a trajectory and plays it. If time lapse between two palm muted notes is above a certain threshold the speed of the trajectory is 4 times faster.
- third mapping: an harmonic note on any of the strings overwrites the pitch of the voice. A series of four pitches are defined so that the voice follows a simple melody. Playing the note on the 2nd fret of the first string (F#) changes the series of pitches. As in the second mapping, the amount of bend is controlling the length of the vocal tract.

4.4. Discussion

The mappings used and listed above were a first attempt to give the MAGE synthesizer an intuitive controller.

On the guitar side, it appears that detection information needs to be reduced. Indeed, with this kind of setup (an instrument controlling synthesized or digital sounds) the player needs to keep a certain correlation between what he plays and what the audience sees and hears. In other terms, if a large amount of the detections is used and mapped to different elements of the synthesized voice, correlation between what is played on the guitar and what is heard can become blurry.

Moreover, being too specific about the mapping (i.e, a specific note played with a specific playing technique) can prevent the player from a certain flexibility and playability. In the improvisation context which was ours, these two elements were important to keep. In the third mapping e.g, using all the harmonic notes detection to overwrite the pitch was preferred to using only one specific harmonic note detection.

On the voice side, it appeared that changing the pitch of the voice was not that perceptible when the pitch difference was around a half tone to two tones. Mapping the

pitch of the guitar directly to the pitch of the voice, was therefore, not the best option.

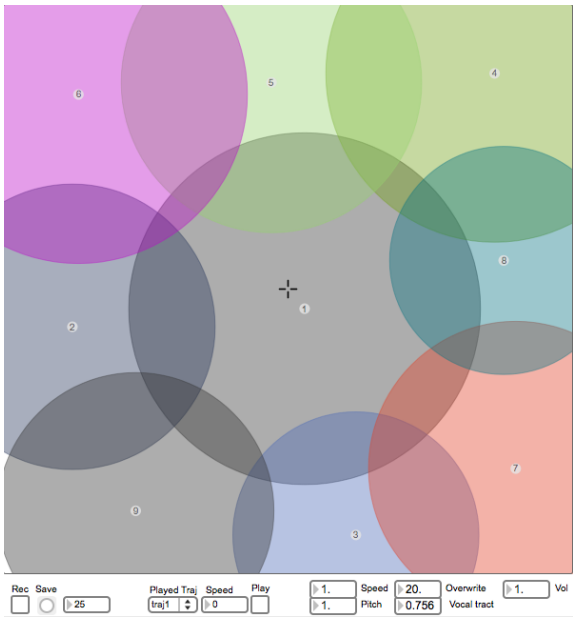

Figure 9: *2D-Interpolation and trajectory tools used to map the guitar detection to the* MAGE *synthesizer.*

5. Results

This project brought us to improve the two frameworks that we were using. On the one hand, we have been able to significantly improve both the computational load and robustness of the algorithms for extracting guitar playing techniques. Various useful expressive gestures such as hammers, pull-offs and harmonics can now be detected reliably on the six strings with a reasonable load on the computer. On the other hand, MAGE has been completely rewritten, leading to MAGE 2.0 being released soon, and this new version clearly leaps forward in term of synthesis reactivity. Due to the previous need in MAGE to preserve compatibility with the sentence-wise stream-based approach, many memory management issues were preventing MAGE to deliver a constant high-quality output with convincing reactivity. MAGE 2.0 makes everything far more usable for performative usages. As a consequence, real mapping strategies could be designed and tested during the workshop, such as using the tonal quality of the guitar to control the intonation of synthetic speech or mapping various fingering techniques to voice types and vocal effects.

6. Acknowledgements

Authors would like to thank the financial and academic supports of the MAGE project: University of Mons (grant 716631) and Acapela Group S.A. Alexis Moinet's work is supported by a public-private partnership between University of Mons and EVS Broadcast Equipment SA, Belgium. Also, we thank G. Wilfart for his contributions.

7. References

[1] B. C. J. Moore, L. K. Tyler, and W. D. Marslen-Wilson, Eds., *The Perception of Speech: From Sound to Meaning*. Oxford University Press, 2009.

[2] F. H. Guenther, S. S. Ghosh, and J. A. Tourville, "Neural Modeling and Imaging of the Cortical Interactions Underlying Syllable Production," *Brain and Language*, vol. 96, pp. 280–301, 2005.

[3] T. Dutoit, *An Introduction to Text-to-Speech Synthesis*. Klumer Academic Publishers, 1997.

[4] G. Fant, *The Acoustic Thenry of Speech Production*. Mouton The Hague, 1960.

[5] A. J. Hunt and A. W. Black, "Unit Selection in a Concatenative Speech Synthesis System Using a Large Speech Database," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing*, 1996, pp. 373–376.

[6] B. Denby *et al.*, "Silent Speech Interfaces," *Speech Communication*, vol. 52, no. 4, pp. 270–287, 2010.

[7] O. Lähdeoja, "An approach to instrument augmentation : the electric guitar," in *Proceedings of the 2008 Conference on New Interfaces for Musical Expression (NIME08)*, 2008.

[8] L. Reboursière, C. Frisson, O. Lähdeoja, J. I. Mills, C. Picard, and T. Todoroff, "Multimodal guitar: A toolbox for augmented guitar performances," in *Proc. of NIME*, 2010.

[9] K. Tokuda, T. Kobayashi, and S. Imai, "Speech Parameter Generation from HMM Using Dynamic Features," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing*, 1995, pp. 660–663.

[10] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous Modeling of Spectrum, Pitch and Duration in HMM-based Speech Synthesis," *IEICE Transactions On Information And Systems*, vol. 83, no. 11, pp. 2347–2350, 1999.

[11] P. Cook, *Real Sound Synthesis for Interactive Applications*. AK Peters, 2002.

[12] I. Stylianou, "Harmonic plus noise models for speech, combined with statistical methods, for speech and speaker modification," Ph.D. dissertation, Ecole Nationale Supérieure des Télécommunications, 1996.

[13] T. Dutoit, *An Introduction to Text-To-Speech Synthesis*. Springer, 1997, vol. 3.

[14] A. Black, H. Zen, and K. Tokuda, "Statistical parametric speech synthesis," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing*, vol. 4. IEEE, 2007, pp. 1229–1232.

[15] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, "An Adaptive Algorithm for Mel-Cepstral Analysis of Speech," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing ICASSP92 1992*, vol. 1, no. 1. IEEE, 1992, pp. 137–140.

[16] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Hidden Markov Models based on Multi-space Probability Distribution for Pitch Pattern Modeling," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing ICASSP99*, vol. 1. IEEE, 1999, pp. 229–232.

[17] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Duration Modeling for HMM-based Speech Synthesis," in *Proc. of ICSLP*, vol. 2, 1998, pp. 29–32.

[18] H. Zen, K. Tokuda, and T. Kitamura, "Decision Tree based Simultaneous Clustering of Phonetic Contexts, Dimensions, and State Positions for Acoustic Modeling," in *Proc. of Eurospeech*, 2003, pp. 3189–3192.

[19] K. Tokuda, H. Zen, and A. W. Black, "An HMM-based Speech Synthesis System Applied to English," in *Proceedings of IEEE Workshop on Speech Synthesis 2002*, vol. 47, no. 1571. IEEE, 2002, pp. 227–230.

[20] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech Parameter Generation Algorithms for HMM-based Speech Synthesis," in *Proc. of the IEEE International Conference on Audio, Speech and Signal Processing*, vol. 3, no. 3. IEEE, 2000, pp. 1315–1318.

[21] S. Imai, K. Sumita, and C. Furuichi, "Mel log Spectrum Approximation (MLSA) Filter for Speech Synthesis," *Electronics and Communications in Japan (Part I: Communications)*, vol. 66, no. 2, pp. 10–18, 1983.

[22] J. L. Gauvain and C.-H. Lee, "Maximum A Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains," *IEEE Transactions On Speech And Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[23] C. J. Leggetter and P. C. Woodland, "Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Models," *Computer Speech and Language*, vol. 9, no. 2, pp. 171–185, 1995.

[24] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Speaker Interpolation in HMM-based Speech Synthesis System," in *Proc. of Eurospeech*, vol. 97, 1997, pp. 2523–2526.

[25] A. Sawabe, T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Eigenvoices for HMM-based Speech Synthesis," in *Proc. of 7th International Conference on Spoken Language Processing ICSLP*, vol. 2, 2002, pp. 1269–1272.

[26] T. Nose, J. Yamagishi, T. Masuko, and T. Kobayashi, "A Style Control Technique for HMM-based Expressive Speech Synthesis," *IEICE - Transactions on Information and Systems*, vol. E90-D, no. 9, pp. 1406–1413, 2002.

[27] H. Zen, T. Toda, M. Nakamura, and T. Tokuda, "Details of the Nitech HMM-based Speech Synthesis System for the Blizzard Challenge 2005," *IEICE Transactions on Information and Systems*, vol. 90, no. 1, pp. 325–333, 2007.

[28] H. Zen, K. Tokuda, and A. W. Black, "Statistical Parametric Speech Synthesis," *Speech Communication*, vol. 51, pp. 1039–1064, 2009.

[29] F. Iazzetta, "Meaning in music gesture," *Trends in gestural Control of Music*, 2010.

[30] R. Graham, "A live performance system in pure data: Pitch contour as figurative gesture," in *Proc.of Pure Data Convention*, Bauhaus-Universität, Weimar, Germany, August 2011.

[31] E. Guaus, T. Ozaslan, E. Palacios, and J. L. Arcos, "A left hand gesture caption system for guitar based on capacitive sensors," in *Proc. of NIME*, 2010.

[32] C. Traube and P. 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. [Online]. Available: http://www.elec.qmul.ac.uk/dafx03/proceedings/pdfs/dafx54.pdf

[33] L. Reboursière, O. Lähdeoja, T. Drugman, S. Dupont, C. Picard, and N. Riche, "Left and right-hand guitar playing techniques detection," in *Proc. of NIME*, 2012.

[34] T. Todoroff and L. Reboursière, "1-d, 2-d and 3-d interpolation tools for max/msp/jitter," in *Proc. of NIME*, 2009.