VOCABLE SYNTHESIS

Alex McLean and Geraint Wiggins
Center for Cognition, Computation and Culture, Department of Computing
Goldsmiths College

ABSTRACT

In many cultures musicians describe the sounds made by their instruments by speaking, singing or transcribing them as words. This paper proposes applying this approach to synthesised sounds, where these *vocable* words control physical modelling synthesis systems. We aim to produce instruments played by typing words, allowing a rich range of musical expression. A working prototype of a system for music improvisation is presented, and future work and applications are considered.

1. VOCABLES

A *vocable* word as one considered only for its musical meaning. That is, a word used to describe a sound made by a musical instrument in a manner analogous to onomatopoeia. Vocable words are used in pedagogy, for example by music tutors to describe a sound to a student, as well as in musical performance itself. Systems of vocables have developed within many musical cultures. In Indian tabla, *bol* syllables are used, each describing a particular articulation. For example *ṭe* represents a non-resonating stroke with the 1st finger on the centre of the dāhinā right hand tabla drum. Similarly, Scottish pipers use *cantaireachd* to describe articulations of the bagpipe, for example *hiaradalla* represents an echo of the D note in the McArthur canntaireachd dialect [3]. Other examples may be found in Chassidic Jewish tunes, Japanese *Kakegoe*, Northern American *Scat singing* and Gaelic *Diddling*.

Chambers [3] classifies vocables in two different ways. Firstly, an *improvisatory* vocable is one created by an individual, albeit from an inventory of accepted sounds, and a *jelled* vocable is one prescribed by convention. There is a grey area between these absolutes, where prescribed vocables may be improvised with to a degree. The second classification defines the relationship between a vocable and the sound it represents. Accordingly, an *imitative* vocable is one structurally similar to the represented sound and an *associative* vocable one is assigned to an instrumental sound or gesture without sharing significant structural or aural features.

1.1. Motor effects

In practice the distinction between imitative and associative vocables is difficult to make, as Chambers [3], p. 13 reports: “Occasionally a piper will say that a vocable is imitative (indigenous evaluation) when analysis seems to indicate that it is actually associative (analytic evaluation) because he has connected the vocable with the specific musical detail for so long that he can no longer divorce the two in his mind.” We may observe the same effect in onomatopoeia, for example an Englishman may hear a hen say “Cluck”, while his German neighbour will likely perceive the same sound as “Tock” [7].

Such incongruities may be explained by the *motor theory of speech perception* [17], which posits that speech is perceived not as sound, but as articulation. We are therefore able to perceive a cat’s meow as imitative articulations of our own vocal apparatus. Here the perception of speech is set in competition with perception of the sound that carries it – we tend towards perceiving either one or the other. This presents interesting challenges for composers working on the boundary between vocal and instrumental sound, as we see in the following section.

1.2. Speech-like timbre in synthesised music

Jones [13, 14] discusses what he terms *speech like timbre*, equivalent to our use of the word *vocable*. In his own pieces “Still life dancing” and “Scritto for computer tape” he uses the CHANT software [20] to apply vowel like quality to instrumental sounds to allow the listener to perceptually relate them to speech like sounds. The listener is then able to focus on the other timbral qualities of speech and appreciate them as part of the music.

Chris Jeffs, who composes and performs music as Cylob, has developed a speech synthesis system which features heavily on his “Formant Potaton” album, released in 2007. His speech synthesis features just two low pass and one high pass filter, with each phoneme assigned parameters found from spectral analysis of Jeffs’ own voice. This speech synthesis is one component of the Cylob Music System [12], developed in Supercollider since 2001, with which Jeffs now composes all his music. While it is possible to identify a song title where it is repeated as lyrical motif, only occasional lyrical fragments are otherwise discernible. This system may therefore be classified as producing vocable speech-like timbre rather than synthesised speech. Perhaps because the speech synthesis is from the same hand as the rest of the synthesis within the music, the vocable sounds blend well with the ‘instrumental’ sounds.

1.3. Voice driven synthesis

Voice driven synthesis is a technique developed by Janer and Maestre [11] to control concatenative synthesis with vocables. This work is inspired by *scat singing* of vocables in vocal jazz improvisation. The vocal audio signal of a musician is segmented and classified according to low-level analysis based on phonetics. Pre-recorded samples are then selected and concatenated with particular note-to-note transformations matched from the vocal articulations. Evaluation found best results required user and instrument dependent training of the system. Perhaps this is a result of the focus on improvisatory scat singing. It would be interesting to see if user dependence is weaker where test subjects are chosen from experts in jelled vocables such as bol syllables.

This model takes an exciting direction in musical interface, with the promise of direct control of an synthesiser from the human voice. The connection with the existing tradition of scat singing is clear [10], suggesting very real musical applications.

2. TEXT DRIVEN VOCABLE SYNTHESIS

Here a system is presented for controlling synthesis with text. Vocables are typed directly as text rather than spoken, and control physical modelling synthesis. This process is analogous to articulatory speech synthesis [6], but with the aim of producing musical sounds and not necessarily speech or speech-like sounds.

The prototype system described below has been built using Haskell SuperCollider [8]. Source code is available at `http://doc.gold.ac.uk/~ma503am/`, along with video screencasts. The system is already usable for music improvisation, and while no part of it has yet been formally evaluated, qualitative feedback from users has been encouraging [18].

For this first prototype we chose to control a straightforward implementation of percussive Karplus-Strong synthesis [15]. This simple model provides us with two parameters; the length of a delay unit in samples and the probability of the signal polarity being switched. The former controls the resonant frequency or pitch, and the latter the aperiodicity or percussive nature of the sound. As is usual with physical models the result is a coherent, continuous parameter space.

Each consonant in the English alphabet is assigned a value for each of the two synthesis parameters. Values were chosen to produce a timbre with some similarity to the spoken consonant, however the greatest effort was placed in producing a broad range of interesting sounds. Vowels were mapped to parameters of a formant filter effecting the Karplus-Strong synthesis output. Formant values were taken from the *FormantTable* Supercollider library [19], giving a vowel-like quality to the sound.

Using these mappings, a word may straightforwardly be parsed into a sequence of parameter values. The synthesis is then controlled by linear interpolation between these values, resulting in *diphones*. That is, the dominant aural features do not arise from individual letters but from articulations between pairs of letters. These articulations are triggered at a fixed rate, giving a striated rhythm to a continuous, speech-like articulation.

2.1. Polymetric language

The vocable words are arranged into polymetric rhythms using a syntax adapted from the Bol Processor version 2 [1]. A monophonic sequence of vocables is specified separated by whitespace, with pauses denoted with an underscore. A simple example would be

```
boom _ tish takka
```

which translates directly to the tabular form:

boom	_	tish	takka

Polymetric sets of sequences are surrounded in braces, the different parts separated by commas. For example

```
{one two three, four five}
```

gives the following structure:

one	_	two	_	three	_
four	_	_	five	_	_

This polymetric rhythm runs from left to right, where sounds in the same column are played at the same time. Notice that pauses are automatically inserted in order to arrange two sequences of different lengths together. This is done using the lowest common multiple of their lengths. If square brackets are used rather than braces, then the parts are repeated rather than padded with pauses. So if we change the above to:

```
[one two three, four five]
```

we then get:

one	two	three	one	two	three
four	five	four	five	four	five

Polymetric constructs may be nested, so that:

```
{boom chakka, bip {bap, bipbap bop}
```

is expanded to:

boom	_	_	chakka	_	_
bip	_	bap	_	_	_
		bipbap	_	bop	_

3. FUTURE WORK AND APPLICATIONS

Wider musical application will require a more involved physical model, and work is in progress exploring vocable control of a drum head model based on a waveguide mesh [16]. To aide evaluation and further development, we then intend to map the perceptual space allowed by such synthesis control methods, using statistical analysis over the results of listening tests.

3.1. Live Coding

Live coding language environments allow computer programmers to improvise music, where a dynamic interpreter takes on code changes without loss of state [5, 22, 23]. Live coders often perform before an audience, writing software to generate their live music. This presents a problem of latency, where it may take some time to bring a musical idea to implementation [2]. This latency may be reduced through regular practice [4], parsimonious descriptions of musical form, and the use of interactive development environments designed for live coding [21]. Vocable words could reduce this latency further, allowing fast expression of instrumental articulations. Vocable synthesis could be made available in libraries, and where possible with domain specific syntax. Techniques could be developed not only for specifying vocables as text, but manipulating them as such; for example using regular expressions.

3.2. Computational Creativity

Imitative vocables in text form have properties potentially useful in the domain of Computational Creativity. We hope to build on earlier work [18] and develop a similarity metric between vocables with well tested perceptual significance. This would allow us to not only build symbolic and probabilistic models of vocable rhythms, but also geometric models of the perceptual space they act within, drawing on research into conceptual spaces [9]. From there we see opportunities for artificial creative agents which share a musical playing field with human agents.

4. CONCLUSION

The use of vocable words in written form to describe instrumental sounds is common in musical culture, and the system prototype described here demonstrates how they may be used to control sound synthesis. We have seen how they might be gainfully used to represent sounds within the burgeoning live coding tradition, and also inform development of artificial creative agents. Further work in these areas is underway.

References

[1] Bernard Bel. Rationalizing musical time: syntactic and symbolic-numeric approaches. In Clarence Barlow, editor, *The Ratio Book*, pages 86–101. Feedback papers, 2001.

[2] Alan Blackwell and Nick Collins. The programming language as a musical instrument. In *Proceedings of PPIG05*. University of Sussex, 2005.

[3] Christine K. Chambers. *Non-lexical vocables in Scottish traditional music.* PhD thesis, University of Edinburgh, 1980.

[4] Nick Collins. Live coding practice. In *NIME*, 2007.

[5] Nick Collins, Alex McLean, Julian Rohrhuber, and Adrian Ward. Live coding techniques for laptop performance. *Organised Sound*, 8(3):321–330, 2003.

[6] Perry R. Cook. *Identification of control parameters in an articulatory vocal tract model, with applications to the synthesis of singing*. PhD thesis, Stanford University, Stanford, CA, USA, 1991.

[7] Victoria de Rijke, Adrian Ward, Karlheinz Stockhausen, John Drever, and Hattice Abdullah. Quack project cd cover notes, 2003.

[8] Rohan Drape. Haskell supercollider, a tutorial. `http://www.slavepianos.org/rd/sw/hsc3/`, 2007.

[9] Peter Gärdenfors. *Conceptual Spaces: The Geometry of Thought*. The MIT Press, March 2000. ISBN 0262071991. URL `http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20\&amp;path=ASIN/0262071991`.

[10] J. Janer and A. Loscos. Morphing techniques for enhanced scat singing. In *Proceedings of 8th Intl. Conference on Digital Audio Effects*, Madrid, Spain, 2005.

[11] J. Janer and E. Maestre. Phonetic-based mappings in voice-driven sound synthesis. In *Proceedings of International Conference on Signal Processing and Multimedia Applications*, Barcelona, Spain, 2007.

[12] Chris Jeffs. Cylob music system. `http://durftal.com/cms/cylobmusicsystem.html`, 2007.

[13] David E. Jones. Compositional control of phonetic/nonphonetic perception. *Perspectives of New Music*, 25(1-2):138–155, 1987.

[14] David E. Jones. Speech extrapolated. *Perspectives of New Music*, 28(1):112–142, 1990.

[15] Kevin Karplus and Alex Strong. Digital synthesis of plucked string and drum timbres. *Computer Music Journal*, 7(2):43–55, 1983.

[16] Joel A. Laird. *The Physical Modelling of Drums using Digital Waveguides*. PhD thesis, University of Bristol, November 2001.

[17] Alvin M. Liberman and Ignatius G. Mattingly. The motor theory of speech perception revised. *Cognition*, 21(1):1–36, October 1985. doi: 10.1016/0010-0277(85)90021-6. URL `http://dx.doi.org/10.1016/0010-0277(85)90021-6`.

[18] Alex Mclean. Improvising with synthesised vocables, with analysis towards computational creativity. Master's thesis, Goldsmiths College, University of London, 2007.

[19] Lance Putnam. Supercollider extensions – formanttable. `http://www.uweb.ucsb.edu/~ljputnam/sc3.html`, 2006.

[20] Xavier Rodet, Yves Potard, and Jean-Baptiste Barriere. The chant project: From the synthesis of the singing voice to synthesis in general. *Computer Music Journal*, 8(3):pp. 15–31, 1984.

[21] Andrew Sorensen and Andrew Brown. Aa-cell in practice: an approach to musical live coding. *Proceedings of the International Computer Music Conference*, 2007.

[22] Adrian Ward, Julian Rohrhuber, Fredrik Olofsson, Alex McLean, Dave Griffiths, Nick Collins, and Amy Alexander. Live algorithm programming and a temporary organisation for its promotion. In Olga Goriunova and Alexei Shulgin, editors, *read_me — Software Art and Cultures*, 2004.

[23] Iohannes Zmölnig and Gerhard Eckel. Live coding: An overview. *Proceedings of the International Computer Music Conference*, 2007.