She Was A Visitor

Recreation of a Composition by Robert Ashley (1930–2014)

Sebastian Arnold
sarnold@mailbox.tu-berlin.de

March 19, 2014

"She Was A Visitor" is a vocal piece written by Robert Ashley in 1967. It is based completely on speech and is performed by one single speaker and number of singers that are divided in chorus groups. This recreation is an attempt to model the chorus digitally using Max, allowing a computer to perform the piece on its own or interactively.

1 Introduction

Looking at the score of "She Was A Visitor", one will instantly recognize the algorithmic form that the piece is given in. The instrumentation consists of a speaker and a vocal chorus, with each performer following a dedicated instruction that does not change for the duration of the performance. The *speaker* repeats the phrase "she was a visitor" periodically and without variation. The chorus (which may include the audience) is divided into *chorus groups* of equal size, each following one *leader*. Each leader periodically chooses a single phoneme that he or she can hear from the speaker's voice (e.g. "sh") and sustains it for one breath. The corresponding chorus group reflects the choice of the leader by sustaining the same phoneme in their individual voice-pitch level. The piece ends when the speaker decides to stop repeating the phrase. A possible configuration of the chorus is visualized in Figure 1. The result of the described process is a "cloud of sound" that seems completely unrelated to the sound of speech.

The 1967 score represents an early idea of what is known today as *granular synthesis*: small pieces of sound (so-called *grains*) are sampled and layered on top of each other, with slight variation of speed, volume, phase and frequency in the playback. The main idea of this recreation is to adopt the score for a computer performance while maintaining the simplistic setting of singers sustaining phonemes. However, the resulting sound may open up new vistas, depending on the parameters that can be controlled by the performer.

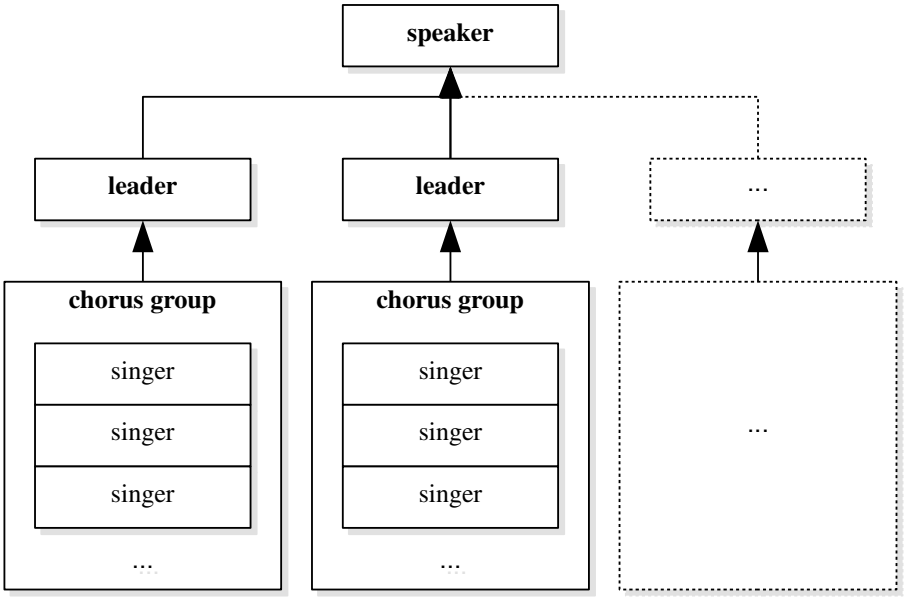

Figure 1: Arrangement of the chorus

2 Modelling the Chorus in Max/Gen

With *Cycling74 Max/MSP*[1] and *Gen*[2], a powerful toolkit is available that allows to model the piece for computer processed performance. The chorus is abstracted in a Max patch that follows the configuration given in Figure 1. The following sections describe each of the elements in more detail.

2.1 Speaker

The speaker is implemented as a sample `buffer~` which, when activated, is being played back in a loop by a `groove~` object. The playback cursor position is available to all choir groups to mirror the auditive communication between the vocalists. The buffer can be loaded with any WAV file from hard disk or the user can record his own voice via the audio input. Files of differing sample rates are resampled on the fly. The output of the speaker can be mapped to any audio output on the interface.

2.2 Chorus Group

Chorus groups are modeled in a `bpatcher` abstraction. That way, it is possible to instantiate multiple groups (up to 8) on demand using the +/− buttons. Each group consists of one leader and an adjustable number of singers (1–16, default 5) utilizing the

[1] http://cycling74.com/products/max/
[2] http://cycling74.com/products/gen/

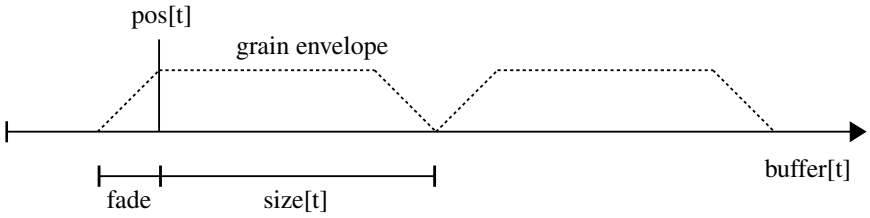

Figure 2: Reading grains from the audio buffer

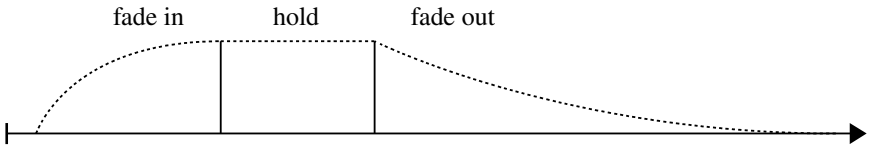

Figure 3: Breath envelope of leaders and singers

`poly~` object. By choosing the corresponding audio outputs, it is possible to distribute the groups over a multi channel playback system. Each group offers interactive controls for the *choose* action, *on/off* switch and a *speed* slider that controls the grain size of the singers. For visual feedback, the audible output of the singers is displayed in a real-time spectrogram.

2.3 Leader

The leader is triggered by a *choose* action. When activated, the current playback cursor position is set as reference point for the granular synthesis of the singers' sustain. The phoneme at this position is then played back for the length of one breath (for implementation of that, see Section 2.4) and the chorus group is activated.

2.4 Singer

The core of this recreation is the singer model. Until now, it is realized in a very basic form, but an extension is discussed in Section 4. Figure 2 shows the process of extracting single grains from the audio buffer. For a given cursor position, an envelope with adjustable size is played back repeatedly. To avoid clicks, the envelope includes a fade length of 10 samples. Note that in the current version, there is no overlap between the periods. The envelope is generated in a `gen~` patcher, which runs at sample rate and accesses the buffer with cubic spline interpolation. Each singer is randomly initialized with different parameters for grain size and playback speed to introduce variation between the voices.

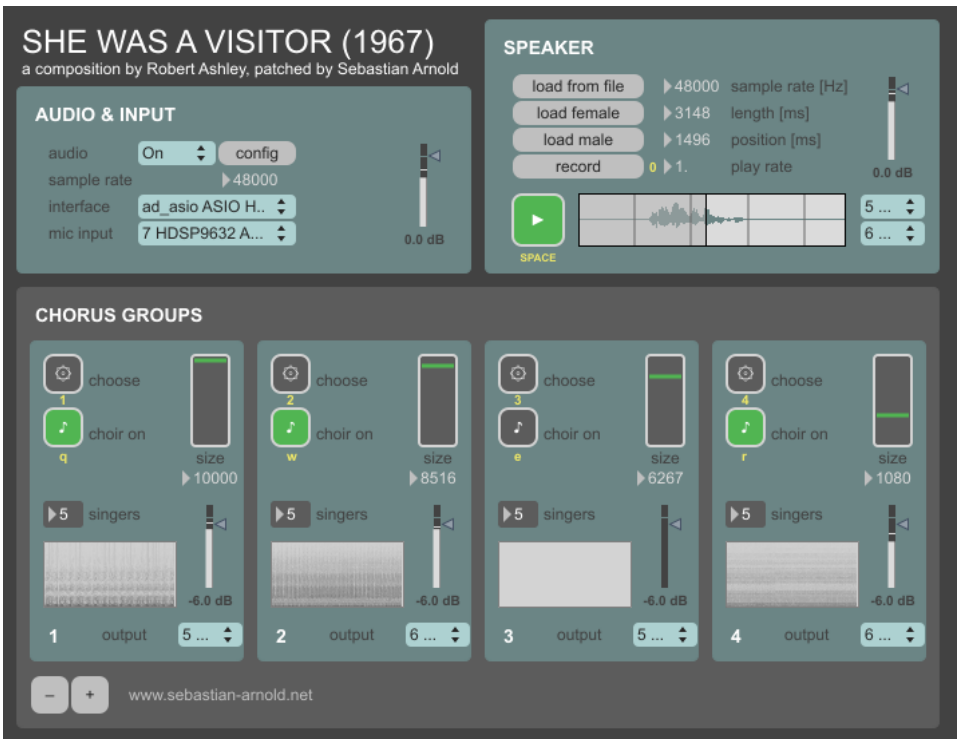

Figure 4: Screenshot of the patch frontend

The breath simulation is implemented using a `curve~` envelope with randomized parameters. Each singer fades in with an exponential and out with a logarithmic volume curve and adopts the position chosen by the leader for the whole period (see Figure 3). Because of differing parameters between the voices, the singers distribute their voices over time and newly chosen phonemes will blend with sounds of the last period.

3 Patch Reference

This section gives an overview on the use of the patch. For a quick start, please see the `README.txt` file enclosed in the package.

3.1 Packaging and Enclosed Content

The patch comes in form of a Max standalone application for Windows and OSX. It should support all the system's audio interfaces and is ready to start without installation. For instant operation, there are two speaker recordings of the sentence "she was a visitor" enclosed[3].

[3]female voice by Tanja Geke, male voice by Elias Emken

3.2 Workflow

The workflow of the patch is laid out from top to bottom (see Figure 4). After configuration, the user starts by loading or recording a speaker sound into the playback box. By hitting *play*, the sound is played back in a loop. Each chorus group can now "grab" a moment of time by clicking *choose* and will slowly start to sustain the chosen position of the speaker loop. Using the *size* fader, the user can tweak the chorus. The parameter values reach from very long grain sizes, where the repeating sounds can be recognized, up into the audio range, where completely new waveforms emerge out of the material. If he or she wants to follow the score, the user can stop the speaker at a given point and turn off the chorus groups. While the speaker will stop after the currently running loop, the singers will decay for a moment until the patch is silent again.

3.3 Remote Control

For live performance, the patch offers basic keyboard and MIDI bindings for playback and chorus control. The corresponding keys are printed in yellow color on the interface. For MIDI control, standard Mackie Control protocol is used. A full list of the bindings can be found in the `README.txt` file.

4 Conclusion and Future Work

While still maintaining the original score, this patch enables a performer to go beyond the limits of a traditional vocal chorus. It turns out that sustaining a single phoneme out of a phrase is not necessarily the same as repeating the waveform on the given position. The role of the leaders is not only to enter on the right moment, but also to anticipate the sound heard and amend the articulation of the phoneme. While this is not feasible using the granular synthesis model, it is now possible to sustain unvoiced sounds such as the "t" and speed up the granular sample repetition up into the audio range.

During the development of this patch, several ideas have arised to expand this rather simple and rough-sounding DSP model to a smoother granular synthesis engine. This includes the introduction of more complex grain envelopes (e.g. with longer fades and overlap), extended playback randomization (e.g. varying starting points and detuning of voices and an advanced breath model) and further sound processing (e.g. the inclusion of equalization, reverberation and/or room simulation). Another feature could be introduced to extend the patch to a self-running generative system: instead of choosing the phonemes manually, a preceding audio analysis could find interesting cursor positions (e.g. by transient, vowel and consonant detection) and offer these to chance-controlled leaders.

The tragic news is that Robert Ashley, the original composer, passed away just weeks before completion of this project. Let this recreation be a memorial for him and his pioneering work.