PYORACLE IMPROVISER DOCUMENTATION

GREG SURGES

1. PyOracle Improviser

PyOracle Improviser is a software improvising partner which learns elements of a performer's style and generates a real-time audio accompaniment. PyOracle uses the Audio Oracle algorithm to determine the repetition structure of an improvisation in an online fashion. This repetition structure, called an Oracle, is then used to recombine the original musical material into new, yet related material.

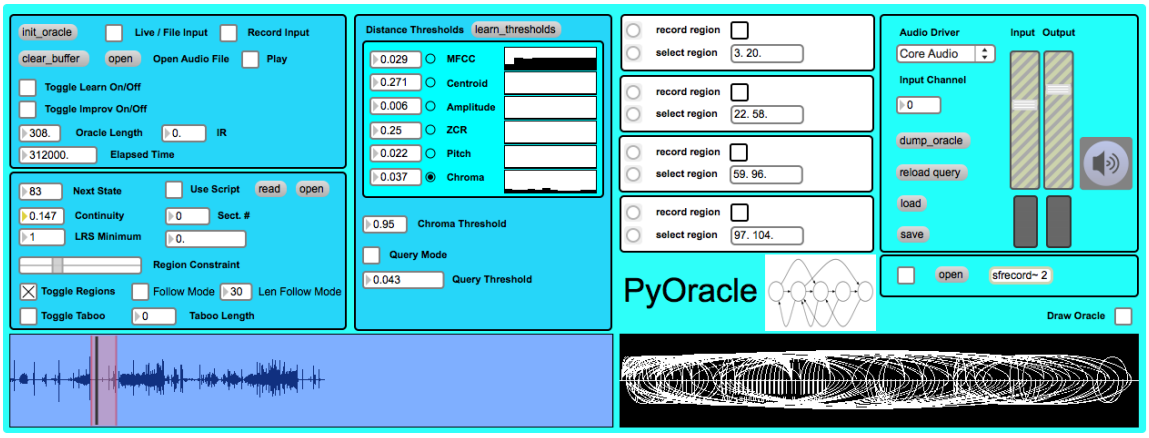

FIGURE 1. The PyOracle Improviser interface.

2. Usage

See Section 6 for installation instructions. The PyOracle Improviser interface is shown in Figure 1. The steps of an example session are as follows:

(1) **Turn on audio processing**: The top rightmost panel of the interface contains the controls of audio I/O. Select the proper audio driver for your system and the audio input you will be using for live input. Turn audio processing on by clicking the speaker icon. The amplitude of **Input** (passthrough of live input) and **Output** (the generated machine improvisation) can be adjusted with the two faders.

Date: 05.03.2013.

(2) **Initialize Audio Oracle**: Click the `init_oracle` and `clear_buffer` buttons in the upper left panel. This discards any previously learned Oracle, and prepares a new one. It also erases any audio which was previously recorded. It is generally desirable to use these functions together, to make sure the Oracle and the captured audio are synchronized. In the same panel, there is a toggle button to choose between live and pre-recorded (file) input. Generally, PyOracle Improviser will be used with live input, but it might be helpful to test or experiment with pre-recorded input. If file input is chosen, click `open` to locate and read an audio file from disk.
(3) **Training**: Depending on whether live or file input is chosen, click the `Record Input` or `Play` toggle button. This begins capturing the audio of the input improvisation. Simultaneously the `Toggle Learn On/Off` toggle box will be automatically activated, beginning the learning of an Audio Oracle structure from the input audio. Since this is a training step, it is best to provide the Oracle with a range of materials, which are representative of the materials which will be used in the full improvisation. After a short period (30" - 1'00"), switch off learning with `Toggle Learn On/Off`, stop recording / playback with `Play` or `Record Input`, and click `learn_thresholds`, near the upper center of the screen. This will start the training process, which may take a short time. When training has finished, the number boxes below `learn_thresholds` will be updated to reflect the ideal thresholds.
(4) **Improvisation**: Reinitialize the Oracle and erase the recorded audio by clicking the `init_oracle` and `clear_buffer` buttons again. Now the Oracle is ready for a full improvisation. Select a signal feature to follow, from the list to the right of the threshold number boxes. Each feature corresponds to some unique aspect of the input audio, allowing for performance styles which might emphasize one musical parameter, i.e. timbre, over another. When ready to begin, toggle either `Record Input` or `Play`, depending on whether you are using live input or a recording, and Oracle learning will immediately begin. At this point you can play freely while the Oracle learns from your playing. To bring in the Oracle-generated improvisation, use `Toggle Improv On/Off`. See Section 3 for an explanation of the parameters (Continuity, LRS Minimum, and Regions) which control Oracle navigation. The controls for these parameters are found on the left center panel.

3. Key Parameters

During an improvisation, the Oracle structure is automatically navigated and used to index the recorded input from the human performer. There are several key parameters which affect the way in which the Oracle is navigated.

- **Continuity**: This is a floating-point number that determines, at each step in the Oracle navigation, the probability of the recorded material being either heard 'as-is' or in a recombined fashion. Another way to think about this parameter is that it controls the amount of fragmentation / granulation of the machine improviser output. The range of this parameter is 0 - 1, with 0 causing complete recombination, and 1 causing a presentation of the original material with no jumps or

recombinations. An example Audio Oracle segment is shown in Figure 2. Each circle represents an Oracle frame, with time advancing from left to right. The Continuity parameter sets the probability that Oracle navigation will move linearly from left to right, following the same path as the original input audio (indicated by the horizontal links from state to state). As the amount of Continuity is reduced, it becomes more likely that the Oracle navigation will advance by forward or backward jumps (indicated by upper or lower arcs on the figure). These jumps connect similar material, but in ways which were not originally played.

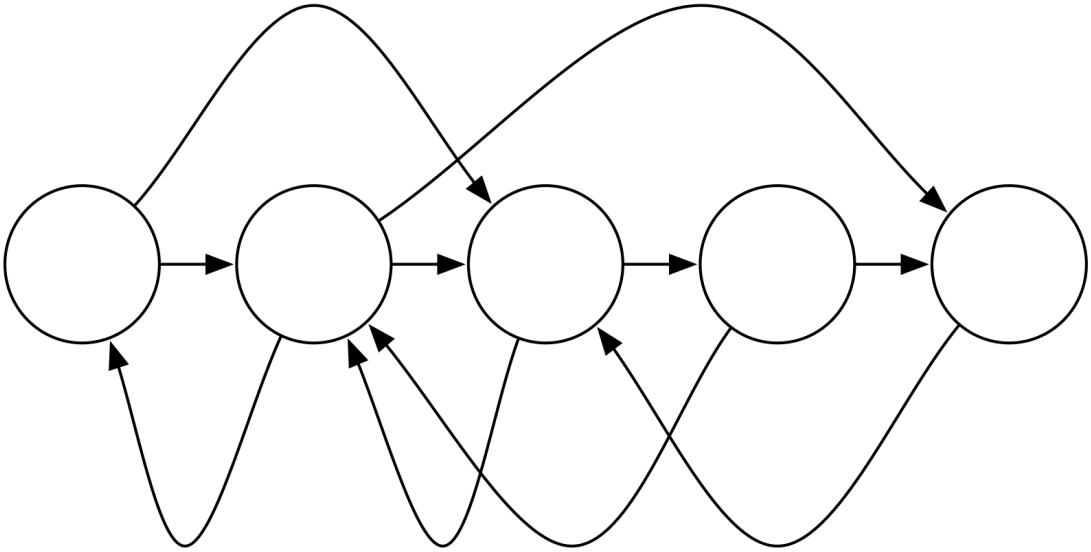

FIGURE 2. An example Audio Oracle structure.

- ***LRS* Minimum**: Controls the amount of shared musical context which must be present between two points before a jump / recombination can occur. Restricting LRS to higher values will produce smoother musical transitions when jumps occur, but will limit the number of possible jumps.
- **Region Constraints**: Limit the navigation to a specific region of the Oracle. This can be used to limit the machine improviser to only specific musical materials at particular times, and can be used as a compositional / structural parameter.
- **Taboo List**: The taboo list can be used to make sure that the Oracle navigation does not return to the same audio frame within too short an amount of time. This can prevent Oracle improvisations from becoming too repetitive.

4. Special Oracle Improvisation Modes

PyOracle Improviser includes several improvisation modes:

- Standard - navigate Oracle according to user-specified parameters.
- Query - Oracle navigation guided by pitch content of user input. In this mode, the Oracle attempts to navigate toward similar material to that being played. Query mode has an independent distance threshold, which allows the required amount of similarity to be adjusted. The Query mode parameters are found in the middle panel, below the audio feature display.
- Follow - Limit Oracle navigation to only the n most recent Oracle states, where n is set by the user. Useful for canonic and imitative effects.

These modes can all be used freely in the same improvisation / composition, to create different musical effects.

5. Scripting Oracle Improvisation

For a fully autonomous performance, it is possible to write a script which modifies Oracle navigation parameters automatically. Using a script, musicians are able to create more structured improvisational structures, which can be repeated and shared with others. A script is stored in a text file, with one instruction or parameter change per line. To use a script file, click the `Use Script` toggle box, and the `read` button. Locate the script file you want to use. To edit the script file, you can click `open` to open a text editor window. The script will begin to execute when `Toggle Improv On/Off` is switched On. It is often best to group parameters into sections, where the section corresponds to a duration.

An example section is shown below:

```
25000 region_handler 0 14;
oracle_gain 0;
section_number 5;
15000 region_handler 15 30;
```

The number in the first line indicates that the Oracle should wait 25,000 ms. before advancing to the next instruction. This means that the parameters set for the previous section will last for 25,000 ms. After 25,000 ms., the Oracle region constraints will be updated to a region lasting from time 0 to time 14 (measured in seconds). The Oracle improvisation output volume can be controlled with `oracle_gain` n, where n is a value between 0 and 127. The `section_number` parameter allows the composer to number sections, and this number is shown on the interface to allow the performer to follow the structure of the piece. Finally, the last line indicates that this new set of parameters should last for 15,000 ms. before changing the Oracle region constraint.

The parameters which can be scripted are listed in the table below, along with their argument value ranges:

Scripting Name	Description	Range
section_number	Display a section number	int
oracle_gain	Oracle improvisation output gain	0 - 127
region_toggle	Toggle on/off region constraints	0/1
region_handler	Set Oracle region constraints	int int
toggle_query	Toggle on/off query mode	0/1
s_query_thresh	Set Oracle query distance threshold	float
p	Oracle navigation continuity	0.0 - 1.0
lrs	Oracle *LRS* minimum	int
s_toggle_learn	Toggle Oracle learning on/off	0/1
s_toggle_improv	Toggle Oracle improvisation on/off	0/1
s_toggle_follow	Toggle Follow mode on/off	0/1
s_follow_len	Set Follow mode buffer length (in Oracle states)	int
s_toggle_taboo	Toggle Taboo mode on/off	0/1
s_taboo_len	Set Taboo mode buffer length (in Oracle states)	int
s_set_feature	Set current feature Oracle	0 - 5

6. Installation Notes

PyOracle Improviser is currently known to run on OS X 10.6 (and above). Windows is probably possible, but is currently unsupported.

(1) If you don't have Max 6, install the lastest Max/MSP Runtime from `http://cycling74.com/downloads/runtime`
(2) PyOracle Improviser requires Python 2.7 (not 3.x!) from Python.org. Get the package here: `http://python.org/download`. The currently recommended package is `Python 2.7.4 Mac OS X 64-bit/32-bit x86-64/i386 Installer`. Open the .dmg and install Python.
(3) After Python installation is complete, run `/Applications/Python2.7/Update Shell Profile.command`. This will update your system to reference the newly installed and up-to-date Python.
(4) Install NumPy. The most current binary is available at `http://sourceforge.net/projects/numpy/files/NumPy/1.7.1/numpy-1.7.1-py2.7-python.org-macosx10.6.dmg/download`
(5) Install SciPy. The most current binary is available at `http://sourceforge.net/projects/scipy/files/scipy/0.12.0/scipy-0.12.0-py2.7-python.org-macosx10.6.dmg/download`
(6) Install Matplotlib. The most current binary is available at `https://downloads.sourceforge.net/project/matplotlib/matplotlib/matplotlib-1.2.1/matplotlib-1.2.1-py2.7-python.org-macosx10.6.dmg`
(7) If you'd like to test your Python install, open Python from the Terminal by typing `python`. If you can `import numpy`, `import scipy`, and `import matplotlib`, things are working as they should. An example session is shown below.

```
unknownf0b4791ee44f:~ Greg$ python
Python 2.7.4 |Anaconda 1.4.0 (x86_64)| (default, Apr 26 2013, 16:48:58)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>> import scipy
>>> import matplotlib
>>>
```