A Performer’s Musicological Research in Performing Interactive Computer Music


WUAN-CHIN LI
Peabody Conservatory of Music of The Johns Hopkins University, Baltimore, Maryland
sandra.li@msa.hinet.net

As a result of technology’s fast improvement, performing computer music pieces that use previous technology system became difficult. Though if we see what the musicians learned through discovering Bach, Mozart’s music, we shall consent the value that traditional music pieces gave to us.

How about the computer music works? Is it no way to reproduce or perform while we don’t have the same equipments anymore? How can current musicians see through the innovative concept, or even just a tiny note that a computer music composer processed? Is the value should naturally disappear with the outmode technology?

My research on performing Jean-Claude Risset’s *Duet for one pianist* was then begin after discovering all the computer program (Max patches) that the composer sent to us does not work on the newer computer system.

1. The Need for Historical Musicological Research in Performing Computer Music

I was first attracted to *Duet for One Pianist* after hearing a recording of the composer’s performance of the work. On this recording, the acoustic piano played by the composer is heard primarily on the left channel and the acoustic piano played by an interactive Max patch is heard primarily on the right channel. The elegant interaction of the two acoustic pianos was particularly interesting to me. I became even more interested in performing *Duet for One Pianist* after discovering numerous performance practice problems involved with recreating the piece on contemporary computer systems.

Risset’s *Duet for One Pianist* presents not only a technical challenge, but also a sophisticated musical thought. It was realized in 1989 when the composer was in residence at the Music and Cognition Group of the MIT Media Laboratory.

The computer program for the work was an early Max patch, which was created by Scott Van Duyne. The Max patches “follow” the pianist, and causes additional parts to be played on the same acoustic piano. This process requires a special MIDI-activated acoustic piano and the Max music programming language written by Miller Puckette at IRCAM and MIT (Chadabe 1997, 183-189). According to the composer’s original program note, this is the first piece written for the Yamaha Disklavier (a MIDI-activated acoustic piano) and a computer system.

Eight separate musical sections, or sketches, represent different experiments in interactive performance. Certain sketches also echo musical ideas from other composers such as Chopin, Webern, and Ligeti.

Computer music performers, are sometimes faced with converting even recent compositions such as Risset’s *Duet for One Pianist* to work on current computer systems. This conversion is necessary in part because computer systems become obsolete faster than many other technologies including traditional musical instruments. Also, updating older software allows a more efficient and reliable code for performance to be written. The contemporary programmer has the advantage of being able to study a more mature version of the composition, one which has undergone repeated performances and revisions.

Risset’s score clearly documented his use of the following: an Apple Macintosh computer with at least 2 MB RAM, Max computer music program (version 2.0a52), and the Yamaha Disklavier acoustic MIDI piano. Certainly, this piece can be performed with the exact performance system that Risset used. Unfortunately, I was unable to find the older Macintosh computer to run the original software, and Risset’s patches always caused the new computer systems to crash. I encountered much incompatibility caused by the newer computer systems we employed at the Peabody Conservatory. I tried a variety of combinations of Max and the Mac OS on different Macintosh computers, but always
encountered some compatibility problems. Some of the then ten-year old Max patches could only be opened in Max as text-file format rather than as graphical patches, some can be opened in graphic patch format but not working operatively.

My next approach was to rewrite the Max patches so that they would work with up-to-date technology, also to improve the performance reliability of Duet for One Pianist.

2. Updating the Computer System for Duet for One Pianist.

In order to perform this duet, my first priority was to make sure that I could recreate Risset’s Max patches faithfully to realize the composer’s original musical intentions of real-time interaction. The patches also had to work on the newer computer systems I used in performances.

There were several original sources to compare in recreating the Max patches—the composer’s music manuscripts, articles about the piece (Risset and Van Duyne 1996, Risset 1990) and the composer’s performance of the piece on a CD (Risset, 1994). These sources, however, did not agree with one another in all cases. I chose the composer’s recording to be the standard above the other resources in my research. This is because the score and Max patches apparently underwent testing and many revisions during the creation of the work. It is likely that the recording, performed by the composer five years after the completion of the piece, reflected a more mature version of the musical idea. The revised patches worked with the Newest Macintosh computer and Max software.

2.1 Three Ways of Recreating the Max Patches

I discovered three ways to recreate the Max patches for Duet for One Pianist. I confirmed the accuracy of my patches by comparing them with the composer’s recording.

I The Composer’s Clear and Complete Notation in the Score

In some sketches, Risset notated clear information on the score documenting the exact parameters of the Max patch. This degree of specificity allowed me to rebuild the Max patches easily without having to decipher the original Max patches. For example, the Max patches for “Doubles,” “Extension,” “Mirrors,” and “Stretches” sketches were re-created from the score by simply reading the composer’s notation.

II Using a Combination of the Score and Non-functioning Graphical Max Patches

Some sketches were not notated to give complete or correct information about the Max patches, but I was able open the patches in the Max graphical format. Even though these patches would not run, I was able to see enough information to re-create the patches in Max 3.5. For example, in the beginning of the “Stretches” sketch, the composer notated on the score that the multiplication factor was 1.3. Using a factor of 1.3, however, did not produce results that matched the recording. The multiplication factor in the non-functioning graphical version of the patch, however, was 2.7 and produced the correct musical results when compared to the recording.

III Translating Pre-commercial Max Text File Format to Max 3.5 Graphical File Format

The third solution I used in recreating Risset’s Max patches was opening the original patch not as a graphical file, but rather as a text file. Unfortunately, the text-file format of the pre-commercial version of Max used by Risset was not the same as the text-file format used in Max 3.5. I had to learn to “translate” the earlier format to the more modern version. Once I could read the old text-file format, I could learn enough about Risset’s intentions to create a working Max 3.5 patch. This solution worked for sketches in which two conditions prevailed: the patches were a) incompletely notated in the score and b) not able to be opened in the Max 3.5 graphical format, but can be opened in the text format.

Although Max portrays patches primarily in a graphical format, it can also display patches in a text format. For example, Max 3.5 can display the simple graphical patch (figure 2-1 A) as text (figure 2-1 B). I learned to understand the Max text-file format by comparing the graphical form of progressively more complex patches with the text-file form at in Max 3.5.
A.

![Figure 2-1 A simple Max patch created in Max 3.5.](image)

A— graphical format; B—text format.

B.

![Figure 2-1 A simple Max patch created in Max 3.5.](image)

Max 3.5 text representation of a patch

<table>
<thead>
<tr>
<th>Explanation</th>
<th>I discovered</th>
</tr>
</thead>
<tbody>
<tr>
<td>max v2;</td>
<td></td>
</tr>
<tr>
<td>#N vpatcher</td>
<td></td>
</tr>
<tr>
<td>88 315 276 488;</td>
<td></td>
</tr>
<tr>
<td>#P newex</td>
<td></td>
</tr>
<tr>
<td>83 122 41 196617 noteout;</td>
<td></td>
</tr>
<tr>
<td>“noteout” Max object box; also numbered as “1”;</td>
<td></td>
</tr>
<tr>
<td>#P newex 83 122 41 196617 noteout;</td>
<td></td>
</tr>
<tr>
<td>“83” indicated the “x” position of this object in a patch window;</td>
<td></td>
</tr>
<tr>
<td>#P newex 83 122 41 196617 noteout;</td>
<td></td>
</tr>
<tr>
<td>“122” indicated the “y” position of this object in a patch window;</td>
<td></td>
</tr>
<tr>
<td>#P newex</td>
<td></td>
</tr>
<tr>
<td>27 40 74 196617 notein;</td>
<td></td>
</tr>
<tr>
<td>“notein” Max object box; also numbered as “0”.</td>
<td></td>
</tr>
<tr>
<td>#P connect</td>
<td></td>
</tr>
<tr>
<td>0 1 1 1;</td>
<td></td>
</tr>
<tr>
<td>#P connect</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0;</td>
<td></td>
</tr>
<tr>
<td>#P connect</td>
<td></td>
</tr>
<tr>
<td>0 2 1 2;</td>
<td></td>
</tr>
<tr>
<td>#P pop;</td>
<td></td>
</tr>
</tbody>
</table>

Table 2-1 shows The relationship between Max graphical and text formats in Max3.5. Translation key derived by comparing Max 3.5 graphical patches with their corresponding text form.

Table 2-1
2.2 The Better Design—the Max Objects Used for Improving the Max Patches of Duet for One Pianist

Many new Max Version 3.5 objects were used to help the new Max patches become simpler and also provide better reliability for its performance. Max computer music program has been improved during its development. Part of the reasons to help this improvement was caused by the computer music pieces like Duet for One Pianist. In Duet for One Pianist, Risset and Van Dyne had to use a more complicate programming design in the early version of Max (2.0a52) than in the newer Max 3.5. For example, the “noorder” customized Max object they created in the old Max can be recreated by the new Max object “buddy” (see 2.2. II). Computer musicians’ needs improved the technology. The functionality Max contains today has a better performance than the time when it was just created.

I. Using “gate” Max Objects to Combine the Sub-Max Patches of Each Sketch

All controls are effected from the piano keyboard, except the loading into computer memory of the
Max patches corresponding to the different sections of the piece (Risset and Van Duyne 1996, 67).

The original Max patches still required the pianist or an assistant to open and close patches on the computer when switching between movements. Because of the limited memory on older computers, it was difficult to create a “main” Max patch that contained all the sub-patches for the complete piece in the earlier days. Contemporary computer systems and software have adequate memory to contain all the necessary patches in one main patch. Consequently I was able to design a single, main Max patch that contained and loaded the code of all the sketches simultaneously.

One useful Max object that helped pursuing this intention was the “gate” object. It helped me to easily proceed from sketch to sketch or rehearse any sketch in random order. The “gate” object allows any MIDI data from the right input to be output (Figure 2-5). A non-zero value in the left inlet of “gate” can select the different outlet for each sketch.

The “buddy” object sends its output(s) only after all inputs (MIDI data) have been received without regard to order. The “buddy” object was especially useful in triggering from a block chord or arpeggio in which not all the MIDI bytes arrived at the same time.

One useful Max object that helped pursuing this intention was the “gate” object. It helped me to easily proceed from sketch to sketch or rehearse any sketch in random order. The “gate” object allows any MIDI data from the right input to be output (Figure 2-5). A non-zero value in the left inlet of “gate” can select the different outlet for each sketch.

Risset’s original "noorder" object was created for triggering on incoming note sequences of varying lengths. Using the new Max “buddy” object (not originally available to Risset), the “noorder” object can be represented as in Figure 2-6.

In the Risset’s original patch, some pitches played by the performer were transposed out of the pitch value range of MIDI keyboard (21-108). Risset determined that the Disklavier he used ignored out-of-range notes (Risset and Van Duyne 1996, 71). On other MIDI keyboards, such as the Kurzweil PC88 I used for my performance, out-of-range MIDI notes “wrapped around” to in-range pitch values on other MIDI channels. Therefore it was necessary for me to include key-range limitation code in my revised Max patches (Figure 2-7). In the new Max patches, the “if” object was simply used to limit the output for out-of-range MIDI notes.
2.3 Using a Standard MIDI Keyboard

The MIDI piano called for by Risset was replaced by a standard MIDI keyboard (Kurzweil PC-88) in my research and performance since Peabody Conservatory had no acoustic MIDI piano when I started the project. The use of a MIDI keyboard simplified the MIDI feedback and keystroke-delay skewing problems reported by Risset (Risset and Van Duyne 1996, 64-66). Although it was impossible to perform the “Resonances” sketch on the MIDI keyboard, however, because of the lack of physical strings to resonate.

3. Performance Considerations

3.1 The Need to Balance Prerecorded MIDI Passages with the Performer’s Real-Time Dynamics

The “Double” and “Resonances” sketches require the computer playing back pre-recorded MIDI sequences on the piano. The sequences were already recorded by the composer and included as part of the original Max patch. Since different MIDI keyboards have different velocity sensitivity, an appropriate velocity adjustment for these sequence files becomes necessary.

The performer must match not only the dynamics of the sequence, but also must address the problem of imitating the articulation of the sequence. If the MIDI sequence files had been notated clearly in the score, another performance possibility would have been for each individual performer to re-record the sequence files. In this way, the performer’s unique technique can be presented and heard equally from the live performance and the computer’s interaction.

Although in this case, I decided to perform with the original MIDI sequence files after I pre-adjusted their balance using another music software (Digital Performer 2.2). Because those sequencer files were recorded by the composer himself, the solution preserved the historical value of the sequences and also allowed me to better follow the composer’s intentions.

3.2 Recovering from “Performance Mistakes”

If one player makes a mistake in a traditional duet, the other has to try to adjust. How does the computer react to a mistake from the live performer? Or how must the performer react when there is an interaction problem? There are many possible solutions for different computer music pieces. In the case of performing instrument and tape pieces, the duty of catching-up can only be taken by the live performer. In an interactive computer music performance, a clever and robust program design can allow the computer to adjust in real-time to the performer’s mistakes.

For example, a performance problem I encountered was in moving from one Max sub-patch to the next in progressing from one sketch to another. In the program design, the last note of one sketch is supposed to trigger the next sketch’s sub-patch. In performances, however, a missed or a wrong note might result in the next patch not being triggered. I designed a Max sub-patch to solve this problem (Figure 3-1). In my error-correction code, another MIDI pedal (besides the sustain pedal) can trigger the next sketch’s patch at any time. Another solution provided for in my code was to click on a bang box that selected the next sketch’s patch. This solution was useful in rehearsals while the pedal solution allowed for a fluid musical performance.

![Figure 3-1 The “trigger_section” Max subpatch. Using the other pedal “control-in” to trigger the Max patch for next sketch.](image-url)
4. Performer and Technician–Build Your Own Instrument

Computer technology develops at a rapid pace. Computer systems become outmoded or obsolete more quickly than most other technologies. Computer music, similarly, has the potential of becoming obsolete and outmoded at a faster pace than usual in traditional music.

Music often becomes obsolete when the instrument it was written for becomes obsolete. For example, music for the Sackbutt, Shawm, and Aulos is difficult or impossible to play now unless we find the “correct information” to rebuild these instruments. Even then, there still is a question of how the music actually sounded, since we have no recordings of the original performances.

The problem of the performance of Duet for One Pianist is that the MIDI piano, computer hardware, and software—the instruments—became outmoded, and no longer worked. Has Risset’s music become obsolete too? No, but the instrument became passé.

The existence of the composer’s recording, scores, and original computer programs, make it possible for today’s performers to gain access to the music and perform it on up-to-date instruments. It is also possible to even improve on the design of Risset’s original instruments and allow for more reliable performances.

When music is closely tied to technology, the music stands the same risk of becoming obsolete as the technology. Thus it becomes very import for composers to write all the necessary detail in the score, rather than relying on instrument design and performance practice to convey the musical meaning. In that way, performers have enough information to recreate the instrument for future performances. Good computer music should not disappear simply because the instrument becomes outmoded.

In order to perform computer music proficiently, a performer must also be a good technician. A computer music performer should be able to build one’s own instruments–redesigning computer music software by using existing materials such as composer’s directions, recordings, and others.

Through the experience of researching and performing Risset’s Duet for One Pianist, I have learned how to design computer music instruments for a better performance reliability. Also, I have shared my views of performance reliability as well as the possibilities of interactive computer music performance with composers and assisted in the creation of their interactive compositions.

Reference


Prix Ars Electronica 1999: Jean-Claude Risset.
http://www.aec.at/prix/kunstler/Erisset.html.


Van Duyne, S. 1998. Personal e-mail contact. 27th, March.


Discography


Thanks to Dr. Geoffrey Wright for all the help on this paper.

Copyright 2004 Wuan-Chin Li