Master Project:

Evolutionary Algorithms Music Toolbox (EAMT)

The aim of the project is to create a toolbox for interactive music programming. Interactive music is a form of computer music, in which the response of a computer to a musical event is complex, rather than singular (as with conventional musical instruments). Conventionally, as musicians we want a musical instrument to play exactly what we command it by touching its keys, bowing its strings or blowing into it. In interactive music, the response to the musician from the music computer is intelligent. The musical context is analyzed and is influencing/altering the musical result. Interactive music principals can be used in improvisational robots, as designed by composers such as George E. Lewis, Daniel Scheidt and Luc Houtkamp. But the same algorithms can also be used in many other software applications, such as the recognition of harmony, rhythm, musical form and structure, etc.

An interactive model will look like this:

| Input MIDI-signal | Low level features | Artificial interpretation | Response generation | Output MIDI-signal |

In this project we want to use the principles of interactive evolutionary algorithms in interactive music applications. Using mutation and selection allows the user to explore new tools for composition, improvisation and sound design. Thus we will create a toolbox of routines, written in MAX/MSP, a graphic programming environment, widely used in computer music. MAX/MSP objects can be written in C, can be programmed is MAX/MSP itself, or a combination of both.

In the first phase of the project we will explore the possibilities of MIDI in and output. Later on we will concentrate on signal processing and work with the composition of sound.

In the project there will be a co-operation with two professional music composers, Wim de Ruiter and Luc Houtkamp. Both have experience with writing interactive and algorithmic composition software. They will support to the master student by defining the needs that interactive music composers have, and guide him in the search for solutions. The newly designed objects can be first tested in practice by implementing them in the composer’s software compositions.

Besides interest in artificial intelligence, and music, the candidate student should also be willing to learn a music specific graphical programming language (MAX/MSP).

Contact: Michael Emmerich, emmerich@liacs.nl
Johannes Kruisselbrink: jkruisse@liacs.nl
Edwin van der Heide: evdheide@liacs.nl
Links:

MAX/MSP: http://www.cycling74.com/
MAX/MSP writing in C:
CNMAT, Berkeley: http://cnmat.berkeley.edu/
Article Computer Music Journal:
http://muse.jhu.edu/journals/computer_music_journal/v025/25.1garnett02.html
Article by George E. Lewis:
http://muse.jhu.edu/journals/leonardo_music_journal/v010/10.1lewis.html