Tree Music and GAIA: An Interactive Demonstration

Dave Topper
University of Virginia
P O Box 400176
University of Virginia
011-434-924-7355
topper@virginia.edu

Judith Shatin
University of Virginia
P O Box 400176
University of Virginia
011-434-924-6495
shatin@virginia.edu

ABSTRACT

This demonstration shows the functionality of GAIA (Graphical Audio Interface Application) in a live demonstration of *Tree Music*, Judith Shatin's interactive composition, commissioned for an exhibit at the University of Virginia Art Museum. A wireless camera sends motion and occlusion data to the computer, where GAIA uses it to navigate a multi-level compositional architecture. Created by Dave Topper, Technical Director of the Virginia Center for Computer Music at the University of Virginia, GAIA is a Linux-based, open-source program that combines GUI manipulation and Perl scripting in an application that invokes RTcmix for sound synthesis and processing.

Keywords

Composition, new interfaces, interactive systems, open source

1. INTRODUCTION

The composition of *Tree Music* posed a number of issues. Fundamental was the question of what interface to use in creating an interactive sound installation to accompany a museum exhibit. GAIA provided the answer with its combination GUI and Perl scripting options that would access RTcmix for sound processing and generation, as well as its ability to capture, analyze and map video. A wireless camera was installed to track motion and occlusion in the gallery, and this data was used to shape the musical flow. For early testing purposes, sliders were substituted for video capture. Video capture was then activated and tested.

The composition itself was accomplished in several stages. First, the sound world was developed from recordings the composer made of the sculptor at work. These were processed to create a continuum of relationships with the source of origin, ranging from quite close to extremely distant. These sonic materials were developed into a network of four sectional types, each of which involves a different response to the movement and occlusion data. The development of these sections, and the network among and through them, involved the creation of multiple GAIA configurations.

2. INSTALLATION SPECIFICS

Originally intended for an extended installation at the University of Virginia Art Museum, our demonstration at NIME will involve a scaled-down version to accommodate the reduced time-frame, and to enable participants to experience the full range of musical response types. We will set up a wireless camera with a wide-angle lens to track the movements of people in the exhibit area, send the data to GAIA, which will realize the musical results. We will also show the program configurations to interested parties so the ease of use of the GUI and scripting options, and the ready movement between the two, will be apparent.

2.1 Tree Music Description

The full version of *Tree Music* is comprised of four main sections, with complex networks of paths within and among them. The result is that multiple trajectories through the compositional structure will result in different musical experiences. Our demonstration will include three of these sections. Motion and occlusion data will be mapped to move through each of these sections. The path moves from most fixed to most open in structure.

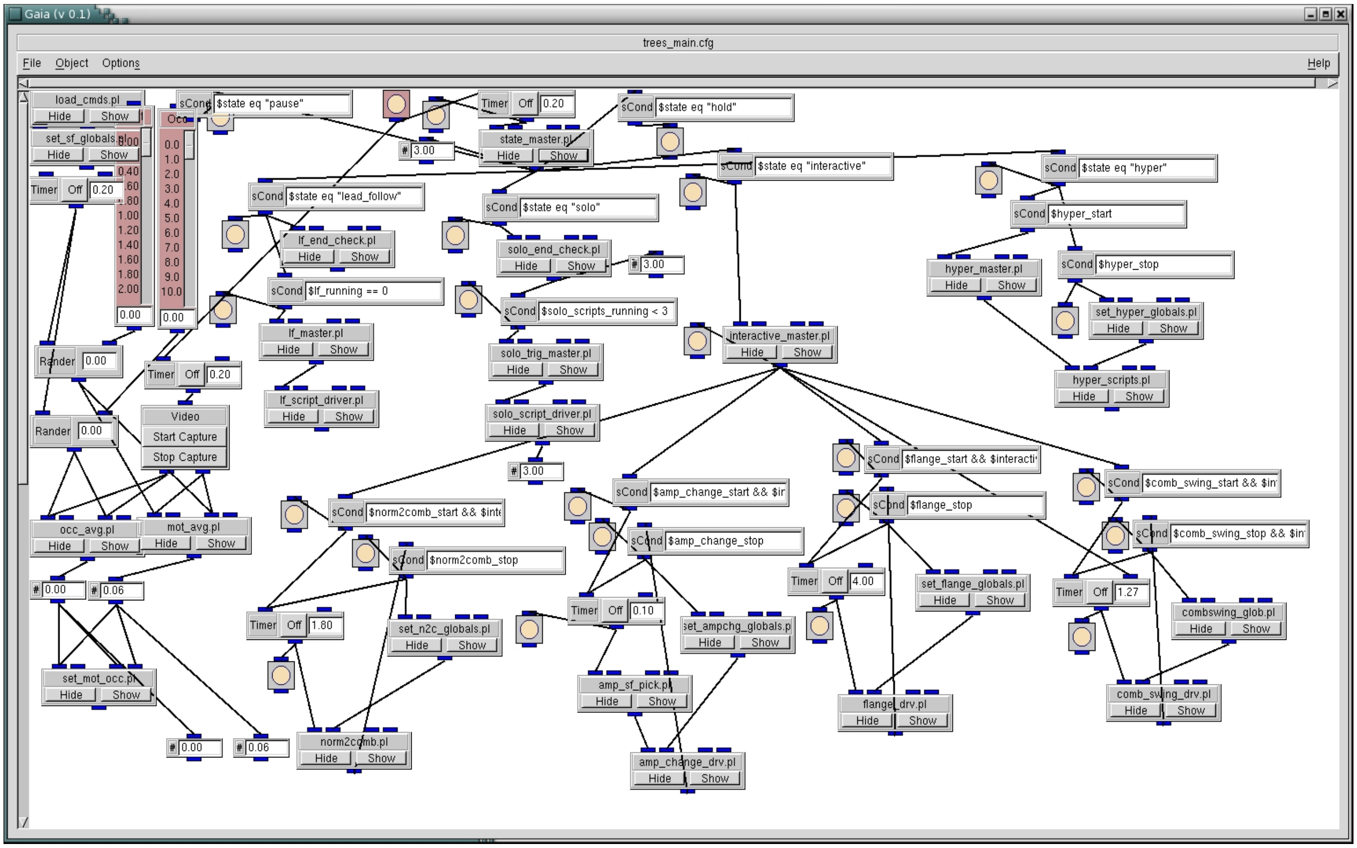
The most fixed section involves a controlled choice and playback of short set pieces. The next involves creation and playing back of ensembles of phrases based on certain conditions. The most interactive involves choice of an interactive script and the changing of its parameters in real time. Parameters affected by such change include, register, amplitude, pitch, timbre and density, depending on which interactive segment is accessed. The fourth section of the full version involves hypermeasures, with extended polyrhythms.

3. GAIA STATES

The various sections of Tree Music are mapped in GAIA to “states” which are monitored by a “master” control script which checks time, motion and occlusion variables to determine whether or not to remain in the current state or move on to another.

Motion is defined as pixel difference from the previously captured video frame. Occlusion is defined as pixel difference from the program start. The latter is used to determine when a subject enters the field of view. As a result, the program must be started when there are no movable subjects in the scene. Both variables were smoothed using simple ten point moving averages.

This control script is executed at a control rate by a GAIA timer object, which emits a pulse at a regular interval, in this case, 1/5th of a second. In order to create a smooth transition between these



states, there is a fifth state called “pause.” The specifics for each state are similar. For each, a GAIA object called a Perl signal condition object, checks the global state variable to see if it is currently set to the respective section. If it is, the top-level control signal is passed through the signal condition, thus initiating the state. Within each section are specific scripts which execute compositional structures. Depending on the type, they either have their own timing mechanism or use the global control rate pulse to trigger activity.

The first state chooses a soundfile from a specific list of files, all designed as mini set-pieces to greet those entering the space. These files were themselves created using RTcmix scripts, but since there were no changing variables, they were saved as soundfiles rather than running the same script with each new iteration.

The second state increases the level of interactivity between the audience and the composition. It plays ensembles of soundfiles. The specific type and number are governed by the global “motion” and “occlusion” variables being fed into the system by the camera. The system maintains a count of the number of files playing as well as a monitor of the global variables in order to decide which sounds to group next.

The third interactive state is actually a combination of “sub-states” with the control mirroring the main structure. There is an “interactive master” control object which coordinates playback of the individual sections. For most interactive states, a separate timer object is required. In the case of *Tree Music*, there are four interactive scripts. One is chosen at random when the “interactive state” signal is broadcast, but the next time through the system, a different script will be chosen, until all have been performed. One of these scripts triggers a soundfile and processes it through a comb filter, pulsing every 1.8 seconds. As motion increases, the register changes. A second script increases the level of interaction by changing the amplitude of audio playback every 0.10 second. A third script combines effects by changing both the register and strength of a comb filter depending on motion variable. The fourth interactive section changes both comb filter register and tempo in conjunction with motion.

When a given state is finished, a specific “state_end” semaphore is raised and control is passed back to the main controlling process, which decides where to go next. If there is continued motion in the camera's field of view, the master control script moves to the next state via change of the global “state” variable. If not, then the system goes into a “pause” state which will eventually move to an extended “hypermeasure” state if no motion is present for an extended period. The pause state is similar to the others, except it does nothing audible, simply counting down a timer. The hypermeasure state plays extremely extended polyrythms. It is a unique state in that it is interruptible. Within the state structure is a check for motion and occlusion variables. If either moves beyond a threshold value, the state ends and the control is passed back to the master process.

4. ACKNOWLEDGEMENTS

The Virginia Center for Computer Music at the University of Virginia; The University of Virginia Art Museum.

5. REFERENCES

1. Garton B. G., and D. Topper. 1997. RTcmix -- Using CMIX in Real Time, *Proceedings of the International Computer Music Conference, 1997.*
2. www.perl.org (n.d.) Retrieved 1/29/04