Jamoma Workshop

Alexander Refsum Jensenius,[a] Timothy Place,[b] Trond Lossius,[c]
Pascal Baltazar,[d] Dave Watson[e]
[a)] University of Oslo & Norwegian Academy of Music, a.r.jensenius@imv.uio.no
[b)] Electrotap, tim@electrotap.com
[c)] BEK - Bergen Center for Electronic Arts, lossius@bek.no
[d)] GMEA - Groupe de Musique Electroacoustique d'Albi-Tarn, pb@gmea.net
[e)] dave@elephantride.org

Jamoma

Jamoma[1] is an open-source project for developing a structured and modularized approach to programming in Max/MSP and Jitter. The main idea of Jamoma is the *module*, which is built up of a separate *algorithm* patcher and a patcher containing the graphical user interface. Figure 1 shows examples of the three main types of modules: *control*, *audio* and *video*.

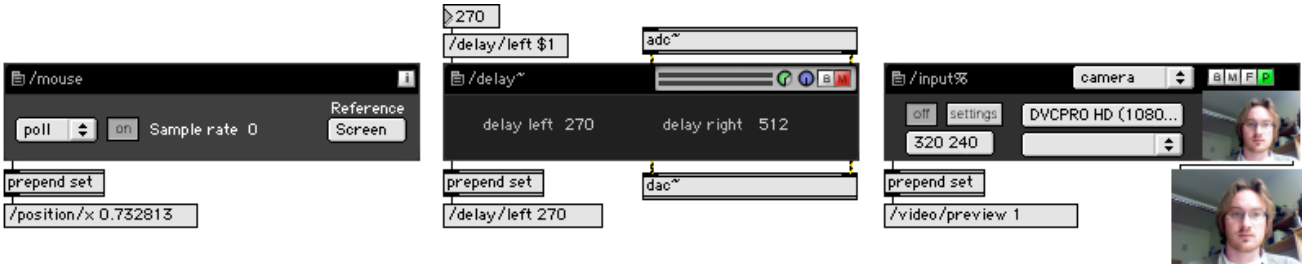

Figure 1. Examples of the three main types of modules (from left): control, audio and video.

Besides providing a large collection of ready-made modules, one of the core strengths of using Jamoma in Max development is that it simplifies the creation of large projects by enforcing a structured approach to the patching. Jamoma uses Open Sound Control (OSC) for internal and external messaging, thus making it easy to communicate to and from modules. Recent development adds support for *cues*, various types of *mappings* and an extensive *ramping library* including a modular *function* and *dataspace library*.

Description of the Workshop

The workshop is aimed at beginning and intermediate Jamoma users. Participants should be familiar with Max/MSP and Jitter to get the most out of the workshop. We advise participants to download the latest version of Jamoma before the workshop.

- Fundamental ideas of Jamoma:
  - Why proposing a structured approach to development and control of modules in Max?
  - The need for modularity and flexibility
  - A *model-view-controller* approach to modular development
- Working with Jamoma:
  - Creating a patch from modules
  - Communicating to and from modules
  - Ramping of parameters
  - How the concept of values and properties extends the possibilities for expression
  - Mapping between modules
  - Setting up cues
- Developing a module:
  - Model: The algorithm
  - View: Designing the graphical user interface
  - Controller: Setting up internal communication in a module
  - Handling presets
  - Documentation
- Examples of scientific and artistic projects using Jamoma

More information about the workshop at: http://www.jamoma.org/wiki/JamomaWorkshopInGenova2008

[1] http://www.jamoma.org