General description of Abunch (version 033)

Hans Roels

Introduction

Abunch is a collection of abstractions in Pure Data (Pd), a real-time graphical programming environment for audio. Pd is open source and therefor free to install and to use. It can be installed on Linux, Windows and Apple computers. Abunch is made in the Pd core version (called 'vanilla') meaning that no extra externals are required and that it can be easily used in any installed version of Pd (higher than 0.40).
Abunch provides a set of ready made objects to record, manipulate, analyse and control audio. It consists of synthesizers, samplers, sequencers, oscilloscopes, etc. Pd contains a wide set of native objects for programming in general and for audio in specific. Abunch uses these native Pd objects to build more than 50 higher level objects with a graphical user interface. For example Pd has objects like [tabread][tabwrite] and [array] to store and read values in an array. The [timeline] object is part of the Abunch library, is based on the native Pd objects [tabread][tabwrite] and [array] and it provides an easy way to read an array, to change the read tempo or the size, to write a text file to this array, etc. Thus Abunch offers extra features on top of the ones already existing in Pd.

Common architecture

The Abunch objects share certain structural features to make their use more intuitive and their combination more powerful. First a preset system was developed within all objects which enables the storing and loading of preset files in .txt format. The internal position of user interface objects as faders, knobs and number boxes and the name and location of opened (sound) files in arrays (of samplers or granular synthesis objects) can be saved with the object [presets]. Next to this global preset system some of the Abunch abstractions can also read or save local presets. In this way patterns can be stored within sequencer objects or synthesizers.
Another example of the unified architecture of Abunch is the help system. The normal procedure in Pd -clicking on a object opens a help file- was copied to Abunch and help files were made expaining every object in this library.
Almost all Abunch objects can be easily controlled by other objects via the inlets, the usage of the GUI elements can thus be automatized. The control values are normalized to a range of 0 to 127. This also facilitates the use of MIDI hardware to control Abunch objects.
Another common feature of Abunch is that time related objects can be controlled by an external clock (provided by the object [clock]). For example, several of the forementioned [timeline] objects can be synchronised by one common [clock] object. There is also a [multi-clock] object for more sophisticated polyrhythmic and polymetric music. It provides 5 clock signals in a adjustable relation to one main clock.
Abunch contains several objects to play and manipulate stored audio samples in an array. These can't only be used with prerecorded samples and audio files but also with fragments of the audio input that are recorded on the spot with the [record-sample] object. This object stores audio in the computers memory (not on the hard disk) and can be connected to other objects like [play-sample] for basic sample manipulation or [grain-sample] for granular synthesis. Moreover one of the external clocks can be used to control the start and stop time of this recording device so that a socalled 'loopstation' can be built with a combination of the [clock], [record-sample] and [play-sample] objects.

First steps in computer music

The first goal of Abunch was to provide an practical open source software tool that would enable beginners to learn more about the musical possibilities in real time of a computer. Users can listen to basic tools and techniques and start experimenting with computer sound by combining these objects and techniques with each other to create their own desired sound devices. What is a sampler, a synthesizer, an ADSR-envelope, a sequencer or a delay effect? How do these techniques sound? Abunch is a box full of sound objects and examples that should provide the

tools and knowledge to answer these questions. The objects for (audio) analysis and algorithmic composition and improvisation that are also included in this collection of Pd abstractions widen the target user group of Abunch even further to performers, composers and improvisers interested in computer music. Now and then these people might prefer easy-to-use objects as they don't have to waste time on building every object from scratch and can immediaely start combining these high-level objects.
Having a wide range of possibilities, being open source and multi-platform Abunch is very suitable for pedagogical goals. Both theoretical and practical courses for students, performers and composers were given using this set of abstractions. As Pd is a graphical programming language and Abunch is made in Pd, students or anyone interested in electronic music can also look to the internal structure (the source code) of every Abunch object and learn more in detail about these issues. This is a great pedagogical advantage as motivated students can autonomously decide to dig deep into the internals of electronic music at their own pace.
Abunch contains more than 35 example files that not only demonstrate the general application rules of Abunch objects but also the internal structure of general audio techniques (FM synthesizer, loopstation device,...) and general musical 'tricks' (how to produce a drone, what is an effective input sound for a phaser effect). Some of these demonstration files developed out of examples in the class room.

Advanced use

Although Abunch was intended for beginners, soon more advanced features were added. At first sight Abunch seems to be resticted by its common architecture aiming at user friendliness but this can be adjusted and bypassed fairly easy. The normal control range of 0 - 127 can be changed to any range and sent to any knob or fader. The ranges of these graphical user interface elements and their internal names (to which the control data are sent) can be printed in a screen for one or for all of the Abunch objects. The GUI of the Abunch abstractions is resticted to a set of adjustable parameters, but in the printed list more detailed parameters (that are hidden underneath this interface) can also be controlled.
The [text-score] is another example of the more advanced features. Values can be sent to any parameter with a receive name in another Abunch object within any range. The score is merely a text file that can be edited within a text editor or software with handy tables like Calc or Excel. All kinds of scores for electronic music can thus be created.

At last Abunch can also be a stepping stone to learning more about the powerful Pd programming language. Learning more about the basics of Pd enables users to modify or completely rebuild existing Abunch objects to their musical needs and desires. That's why the set of example files in Abunch also contains demonstrations about the workings of native Pd objects and about the combination of Abunch and Pd objects.