— NVM —
A MODULAR REAL-TIME PHYSICAL MODELLING SYNTHESIS SYSTEM FOR MSP

Richard Dudas
IRCAM, 1, Place Igor Stravinsky, 75004 Paris, France
dudas@ircam.fr

Abstract

NVM is a modular physical modeling synthesis system based on modal synthesis and implemented as a suite of external objects for MSP. There are two basic types of such objects: those which represent vibrating physical bodies, and those which establish physical connections between them. Control information is provided via MSP audio-rate signals.

Introduction

The driving force for the creation of NVM was the author's personal involvement with IRCAM's physical modeling synthesis program, Modalys. What makes Modalys so exciting from a musician's point of view is the freedom it offers the user to conceive and design practically any virtual instrument by combining the basic physically modeled objects.

The MSP extension to Max on the Macintosh offers real-time digital audio processing on a personal computer platform. For the musician, its modularity offers virtually unlimited possibilities for the construction of signal processing algorithms. Although MSP and Modalys share a similar modular design, MSP uses a graphic interface as opposed to a textual one.

When the first API for MSP became available in the Autumn of 1997, the author began working on NVM in order to discover just to what extent some of the basic synthesis ideas behind Modalys could be re-implemented for use in real-time.

It is worth noting that a preliminary real-time implementation of Modalys was created by Francisco Iovino in 1997 for IRCAM's FTS on SGI. This prototype was created to demonstrate the need for direct interaction with modal synthesis, rather than to provide a tool for musical production. The generality of the implementation limited its use for musical purposes since a heavily reduced sampling rate was needed to allow the large synthesizer to run. It nonetheless demonstrated that a real-time environment provides the composer the necessary feedback to create his virtual instrument quickly and discover intuitively the proper control parameters necessary for playing the instrument in a musical fashion.

Modal Synthesis

Modal synthesis is the musical application of *modal theory* which was developed by the aeronautics and engineering industries to simulate precisely the movement of vibrating structures. In modal theory, a vibrating structure is represented by a sum of simple vibrations or *modes*. Each mode has its own frequency, damping factor and set of points which define the *modeshape* — a description of the object's physical deformation for that mode.

Since all modal objects share this common data structure, they can be connected together in any manner to form larger structures, independent of the objects' real-world spatial properties. But, above all, the advantage that modal synthesis offers for the musician is the direct relationship between the modal frequencies and the spectral composition of the object — each of the object's modes corresponds to a partial of it's sound spectrum.

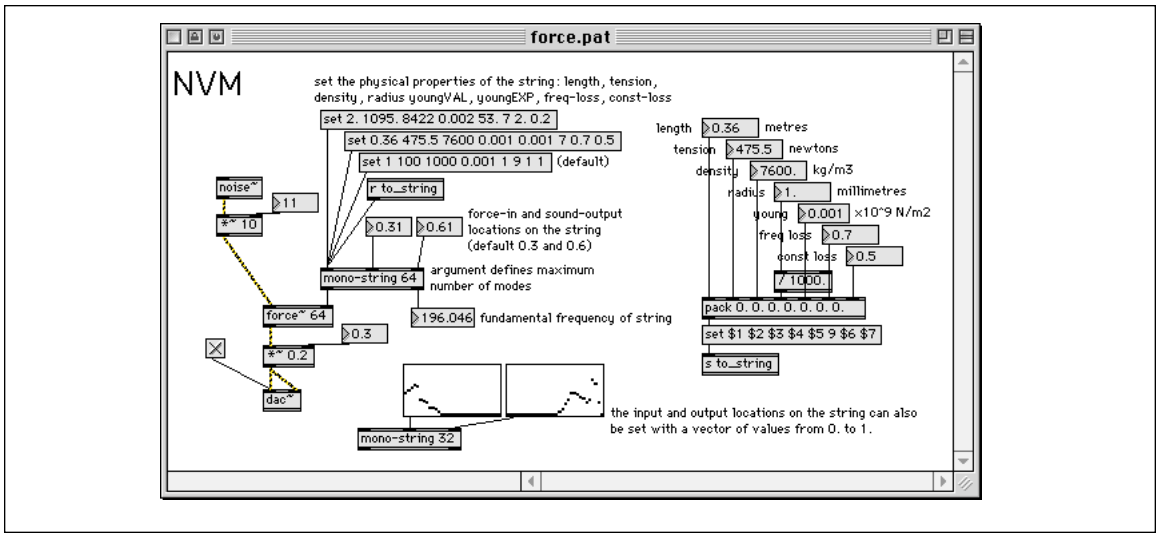

Figure 1. This patch shows an example of white-noise used to excite a string. The right-hand side of the patch shows the physical parameters of the string which can be modified in real-time.

The Building Blocks

There are two basic kinds of external objects in NVM: modal objects and interactions. Modal objects are data structures which include a mass, a string and a plate. They may be used either as resonators or exciters, and their physical parameters (length, density, tension, elasticity, etc…) may be modified in real-time.

Interactions are the MSP external objects which perform the DSP calculations using the data contained in the modal objects. The most basic interaction is expressed with the force~ object which forces a single modal object to resonate. Any incoming audio signal from an impulse, to white noise to a live microphone input can thus be used as a source of excitation. This was the first object created for NVM. A patch is shown in Figure 1.

Each NVM object is provided with an input and output location. In the example shown, the string's endpoints are represented by the numbers 0 and 1. The string's input (0.31, or about one-third along the length of the string) receives the incoming force. The output (at 0.61 — about two-thirds the length of the string) is the "listening point". This is the point on the string whose velocity will be measured to create the output sound. In addition to using a floating point number to specify a single input or output point along the length of the string, a list may be used in order to distribute the input or output among several points along the entire length of the string. In this case, the list represents a vector of scalar values for the points which define the modeshapes.

Slightly more complex interactions such as strike~ and pluck~ use two modal objects and calculate an interaction based on the objects' positions with respect to each other (this is analogous to the exciter/resonator model). In this case, an incoming force or position is sent to the input location of the first object (the exciter), the interaction takes place between the output location of the first object and the input location of the second. The sound output is obtained from the output location of the second object (the resonator). It is possible that this connection system will change as NVM develops, and more complex connections between multiple objects become necessary.

Of course, the program Modalys allows the construction of much more complex instruments due to the richer palette of objects and interactions available. However, despite its reduced nature, NVM does offer direct control over the objects' physical parameters themselves, something not possible with Modalys.

NVM and MSP

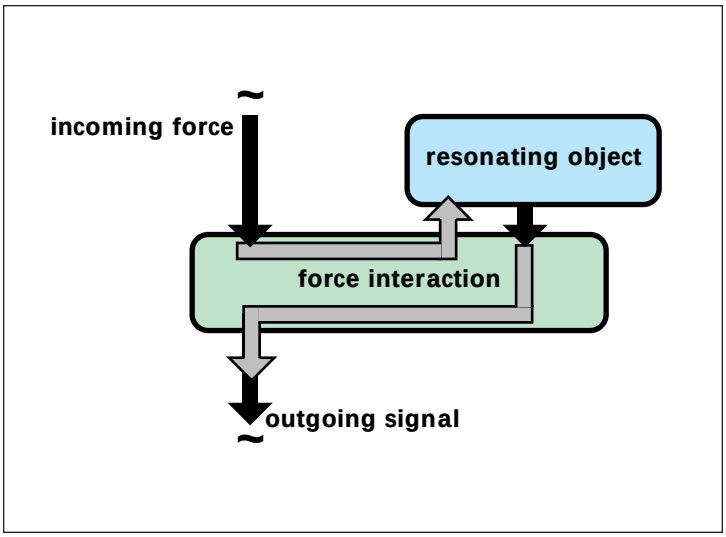

Figure 2. The most basic interaction in NVM — using an incoming force to resonate a modal object. (Black arrows represent actual connections between Max/MSP objects, while grey arrows represent virtual connections made within the interaction object.)

The main obstacle in designing a modular modal synthesis environment for MSP was the necessity for sample-by-sample DSP calculation as opposed to MSP's inherent block-calculation. The solution, at least for now, has been to keep the synthesis calculation within the interaction modules, but allow them to access and modify certain data structures within the object modules. Thus the NVM objects are connected to the interactions by means of a "symbolic" connection such as the connection between the standard MSP delay tap objects, tapin~ and tapout~. Figure 2 shows a diagram of the internal data flow of the force~ interaction shown in Figure 1. In this diagram we can see that, although the resonating object's output is connected to the interaction's input, it is the interaction module which accesses and modifies the resonating object's data, contrary to MSP's usual top-to-bottom data flow paradigm. Thus, a group of interconnected NVM modules act together as if they were a single MSP object. This allows different types of objects (resonators, interactions…) to be visually separated on the screen according to their function, while providing the sample-by-sample computation necessary for the physical modeling synthesis.

Although some basic real-world models (such as a string, an air column and a plate) are provided with NVM, there is also an abstract modal object which the user can fill with any modal frequency, damping and modeshape data desired. This idea is an extension of the Modalys single-point object. The modal data could be prepared with the aid of another program, or even calculated directly within the Max environment.

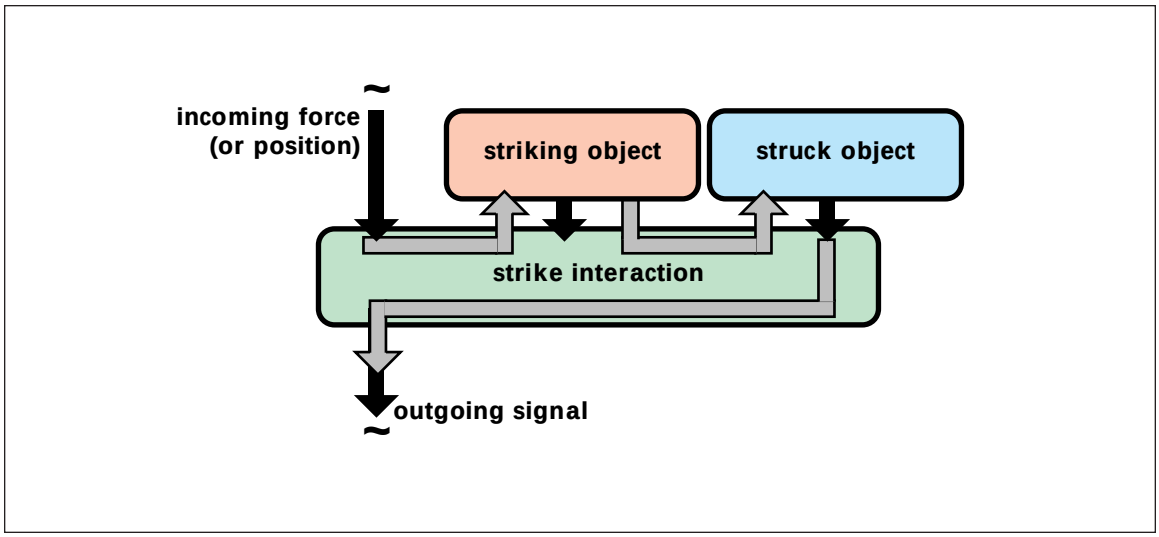

Figure 3. The "strike" interaction is an example of a two-object interaction in NVM. (Black arrows represent actual connections between Max/MSP objects, while grey arrows represent virtual connections made within the interaction object.)

Work on finalizing an interaction between two NVM objects is still underway, and should be completed in the summer of 1998. Figure 3 shows the internal workings of such an interaction. Because modal synthesis at its simplest requires a series of calculations for every mode of every object for each sample of sound generated, it was necessary to optimize the code in order to allow NVM to work as efficiently as possible in a real-time environment. An interaction between two objects increases the computation time necessary by more than twofold. So, for the moment, NVM is restricted to these simple interactions.

Conclusion

NVM is designed to bring real-time physical modeling synthesis to a personal computer platform (something which could not have easily happened without the MSP extension to the Max environment). Modal synthesis was chosen in order to allow maximum flexibility for the composer. Since all resonators share a common data structure, any resonator may be used with any type of excitation. This allows the composer to design such instruments as a plucked crotale, or hammered column of air.

Although still in its embryonic stage, NVM is already running in its most basic form and scheduled to be premiered in concert in the autumn of 1998. Without a doubt, NVM will undergo many changes as it continues to be developed.

Acknowledgments

NVM could have never existed without the help of Francisco Iovino, the current developer of IRCAM's Modalys, whose encouragement and never-ending patience explaining the inner-workings of modal synthesis were invaluable and served as a great inspiration for the creation of NVM. It is the author's hope that the knowledge acquired during the making of NVM will help to optimize Modalys, and eventually offer a model for an ergonomic and completely modular real-time implementation of that program.

The author would also like to acknowledge composer Per Martensson who will be first to use NVM in concert in the autumn of 1998.

Bibliography

Adrien, J. M., "The Missing Link: Modal Synthesis", in DePoli, G., Picalli, A., Roads, C., *Representations of Musical Signals*, MIT Press, Cambridge, Massachusetts, 1991.

Dobrian, C., *MSP Manual*, Cycling '74, Santa Cruz, California, 1997.

Iovino, F., Dudas, R., Caussé, R., Misdariis, N., Polfreman, R., "Modalys: a Synthesizer for the Composer-Luthier-Performer", to be published in a forthcoming issue of the *Computer Music Journal*, 1998/99.

Iovino, F., Schnell, N., "Preliminary Notes on the Modalys-FTS Implementation", Internal Report, IRCAM, 1996.

Morrison, J., Waxman, D., *Modalys Introduction, Modalys Tutorial, Modalys Reference*, IRCAM Documentation, Paris, France, 1991-1998.

Zicarelli, D., *Writing MSP Externals*, Cycling '74, Santa Cruz, California, 1997.