HSP v2: Haptic Signal Processing with Extensions for Physical Modeling

Edgar Berdahl
CCRMA
Stanford, CA, USA

Alexandros Kontogeorgakopoulos
Cardiff School of Art & Design
Cardiff, United Kingdom

Dan Overholt
Dept Architecture, Design, & Media Tech
Aalborg University, Denmark

ABSTRACT
The Haptic Signal Processing (HSP) platform aims to enable musicians to easily design and perform with digital haptic musical instruments [1]. In this paper, we present some new objects introduced in version v2 for modeling of musical dynamical systems such as resonators and vibrating strings. To our knowledge, this is the first time that these diverse physical modeling elements have all been made available for a modular, real-time haptics platform.

Author Keywords
Haptics, physical modeling, Cordis-Anima, digital waveguides, Max/MSP, and robotics.

ACM Classification Keywords
H5.2. User interfaces: Haptic I/O; H.6.5 Model Development: Modeling methodologies; I.2.9 Robotics: Operator interfaces.

INTRODUCTION
In Max/MSP 5, it has become possible to set the signal vector size to one in the *DSP Status* window. This feature allows audio signal objects to be interconnected with as little as one audio sample of delay around feedback loops. As a consequence, physical models can be easily and elegantly created directly in Max/MSP by connecting audio-rate filters with one another. We have used these models to control Falcon robotic arm haptic devices, which are currently available for as little as US$150 [1].

MASS, SPRING, AND DAMPER MODELING PARADIGM
It is possible to model any linear driving-point impedance or admittance using masses, springs, and dampers, so it could be argued that these are the most fundamental modeling elements [3]. Consequently, HSP v2 includes Max/MSP subpatches that model virtual masses, springs, dampers, and nonlinear elements that can vibrate at audio rates. For instance, any audio signals received in the inlet of the `mass~` object are interpreted as forces in Newtons, and the output from the `mass~` object is the resulting position [3,5]. In contrast, because both damper and spring objects can receive positions as inputs, they can be combined together into a single `link~` object. Consider the example shown in Figure 1 (upper left), in which a mass m of 0.05kg is connected by way of a spring with stiffness 4000 N/m in parallel with a damper with coefficient 0.0002 N/(m/s) to mechanical ground g.

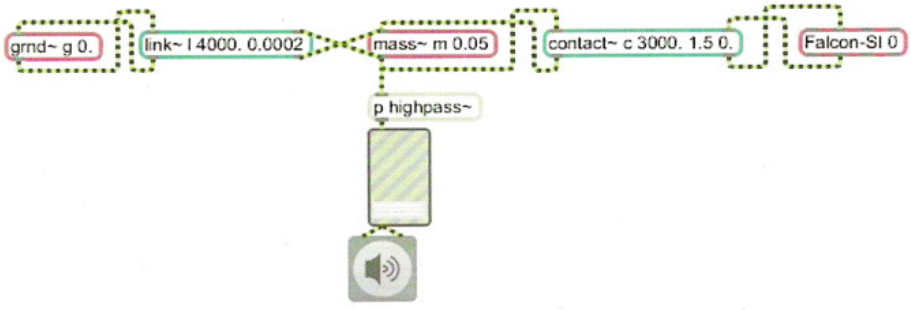

Figure 1. Example patch in HSP showing the signal flow

Model Implementation Details
The physical modeling objects in HSP v2 discretize the differential equations for the laws of motion of continuous-time masses and links according to the Cordis-Anima method. In particular, the mass dynamics are discretized using the second-order central finite-difference scheme, while the damper dynamics are discretized using a backward Euler difference [3,7].

The advantage of the schemes presented in this paper is that the virtual elements are modular [3,5]. Consequently, they can be interconnected in multitudinous configurations that all have physical meaning. In fact, the interconnections shown in the graphical user interface illustrate not only the audio signal flow, but also the physical relationships between the virtual objects. (While the pmpd library implements a subset of this kind of functionality, its sampling rate is much slower than the audio sampling rate, making it difficult to directly synthesize high quality sound using pmpd [5].)

Naming of Elements
Another advantage of modularity is that each element can be assigned a name as the first argument in the object box in Max/MSP [5]. Objects can be addressed by their names using the `send` object in Max/MSP. For instance, to change the parameters of the link l in Figure 2 to 8000 N/m and 0.0005 N/(m/s), the message "l 8000 0.0005" can be sent from any subpatch. Naming is especially useful when models become more complicated. For instance, naming makes it much easier to change the pitch of

a mass-spring model of a string incorporating 19 identical mass elements and 20 identical link elements (not shown).

Real-Time Force-Feedback Haptic Interaction

The Falcon force-feedback robotic arm can be accessed using the `Falcon-SI` object, which specifies the index number of the Falcon. We have tested controlling up to five Falcons connected simultaneously to a MacBook Pro with a 2.4GHz Intel Core 2 Duo processor. The primary function of `Falcon-SI` is to serve as a three-dimensional mass. It has force inputs in Newtons in the *x*, *y*, and *z* axes, and it outputs position in the *x*, *y*, and *z* axes in meters.

Because the Falcon object behaves essentially like a mass, it can be connected to virtual masses (or other `Falcon-SI` masses) by way of links. Figure 1 (right) shows how the *y*-axis of the Falcon is connected to the virtual mass by way of a `contact~` link named *c* with stiffness 3000 N/m and damping coefficient 1.5 N/(m/s). This type of contact link behaves like a spring and damper that only exert forces when the Falcon grip is pushed beyond or "inside" the virtual mass *m*. HSP v2 supports other kinds of contacts such as `contact-pluck~`, a plectrum-like link with hysteresis, and `contact-attract~`, which is similar to `contact-pluck~` but does not incorporate hysteresis.

Choosing Parameters

The physical interpretation of the elements' parameter values is useful in many contexts for designing models. For instance, a virtual mass can be chosen to have about the same mass as the Falcon grip. However, in some other contexts, choosing parameters can be less straightforward. For example, the resonance frequency of a simple mass-spring-damper oscillator depends on the mass, stiffness, and damping parameter. For this reason, we have created the `resonator~` object. Internally it contains a mass and a link, and it adjusts their parameters to approximately achieve a specified resonance frequency, decay time, and mass. In this sense, `resonator~` approximates the behavior of the CEL object in Cordis-Anima [4].

Digital Waveguide Modeling

An alternating mass-spring model (not shown) of a one-dimensional waveguide is not perfectly harmonic when the masses are identical and the springs are identical [6]. For this reason, it is more straightforward to construct one-dimensional waveguide models using digital waveguides [8]. Thus far, we have created the following objects in HSP v2 for digital waveguide modeling:

- `DWG-end~` implements a one-dimensional waveguide with a termination at one end, where the termination incorporates some damping at higher audio frequencies.

- `ex-junct~` implements an explicit digital waveguide junction [2]. When connected to two `DWG-end~`'s representing velocity waves, the junction provides a position output and a force input. Hence, `ex-junct~` provides for a single point of interaction and can easily be connected to `Falcon-SI` using objects such as `link~` or `contact`-type objects.

FINAL WORDS

The physical modeling extensions to HSP v2 make it easier to create new haptic musical instruments, which provide force feedback to the performer. This release[1] of HSP v2 includes example models of instruments that we are incorporating into live music performances. Further in the future, we plan to not only create many more models, but also to provide support for additional haptic devices, enabling a broader suite of force-feedback interactions. For example, an external could be written for Max/MSP and/or Pure Data to interface with any devices supported by the Chai3D environment.

ACKNOWLEDGMENTS

We would like to thank Claude Cadoz, Annie Luciani, Julius O. Smith III, and Jean-Loup Florens for pioneering many of the physical modeling formalisms upon which these physical modeling extensions are built.

REFERENCES

1. Berdahl, E., Niemeyer, G., and Smith, J. A Simple and Effective Open-Source Platform for Implementing Haptic Musical Instruments, In *Proc. NIME 2009*, ACM Press (2009), 262-263.
2. Berdahl, E., Niemeyer, G., and Smith, J. Using Haptic Devices to Interface Directly with Digital Waveguide-Based Musical Instruments, In *Proc. NIME 2009*, ACM Press (2009), 183-186.
3. Cadoz, C., Luciani, A., and Florens, J.-L. CORDIS-ANIMA: A Modeling and Simulation System for Sound and Image Synthesis: The General Formalism, *Computer Music Journal* (1993), 17(1), 19-29.
4. Castagne, C. and Cadoz, C. GENESIS: A Friendly Musician-Oriented Environment for Mass-Interaction Physical Modeling, Proc. International Computer Music Conference (2002), 330-337.
5. Henry, C. pmpd: Physical modelling for Pure Data, Proc. International Computer Music Conference (2004).
6. Incerti, E. Modeling Methods for Sound Synthesis, Proc. International Computer Music Conference (1997).
7. Kontogeorgakopoulos, A. and Cadoz, C. Cordis Anima Physical Modeling and Simulation System Analysis. In *Proc. SMC'07 the 4th Sound and Music Computing Conference*, Sound and Music Computing (2007), 275-282.
8. Smith III, J. Physical Audio Signal Processing, W3K Publishing (2010), http://ccrma.stanford.edu/~jos/pasp.

[1] http://ccrma.stanford.edu/~eberdahl/Projects/HSPv2.zip