pmpd : Physical modelling for Pure Data

Cyrille Henry

cyrille.henry@la-kitchen.fr

Abstract

pmpd is a collection of objects for pure data (pd). These objects provide real-time simulations, especially physical behaviors. pmpd can be used to create natural dynamic systems, like a bouncing ball, string movement, chaos, fluid dynamics, sand, gravitation, and more. It can also be used to create displacements thus allowing a completely dynamic approach of pd computing. Pmpd can also be used for non-real-time audio synthesis.

With pmpd physical dynamics can be modeled without knowing the global equation of the movement. Only the cause of the movement and the involved structure are needed for the simulation. By combining pmpd's various objects one can simulate a very large variety of dynamics systems.

These objects are designed to be used with pd: a real-time graphical programming environment dedicated to audio signal processing. Pd's graphical programming environment is well adapted to the creation of particular physical modelling. GEM is a pd library dedicated to images processing. In the provided pmpd examples GEM is used for the movement visualisation.

1 Introduction

Physical modelling is widely used in audio and video synthesis. Physical modelling can be used not only to model real world dynamics, but also to produce dynamics that are not found in nature. pmpd is an approach to models made of particles, which are only one of many possible dynamics system. This approach is widely used for video animation and interactive simulation (Castagne and Cadoz 2002).

pd is a real-time programming environment dedicated to audio synthesis (Puckette 1996). Some pd object already provide physical modelling, only for audio synthesis (Rath, Rocchesso and Avanzini 2002). Since these objects do not aim to solve general physical modelling problems and often function at the audio rate, they are not well adapted to dynamic simulations.

pmpd provides a very flexible way to particle physical modelling simulation and other kind of comportment based modelling. Using pmpd within the pd programming environment allows real-time interactions with this simulation. pmpd allows real-time dynamics computing for audio and video applications thanks to pd and GEM. Furthermore, particle positions from real-time control-rate movement simulation can be recorded and then synthesized as audio to generate rich physically modeled sounds.

2 Fundamentals

pmpd is a collection of objects for pd. This library is designed to provide high level objects (each pmpd object can be made as a pd abstraction). These objects are low level comportment objects. Assembling these objects can generate complex behavior due to the interaction among the basic objects.

pmpd implements the basic objects that allow particle-base physical modelling (Cadoz, Luciani and Florens 1993). Very complex behaviors can be simulated without knowing the global equation of the movement. Only the cause of the movement and the structure are needed for the simulation. Pmpd can then easily be used for the simulation of a very large variety of comportments.

Complex simulations are basically made from two kind of elementary objects: “mass” and “link”.

2.1 Mass

“Mass” objects react like a point mass. It takes forces at its input, and outputs its position. It moves according to Newtonian dynamics:

$$\sum \vec{F} = m\vec{y}$$

“Mass” objects have inertia, but they have no volume (they cannot rotate). When told they make the summation of forces applied to their inlet to calculate their acceleration.

2.2 Link

“link” objects take two mass positions and output two opposite forces depending on the relative position and speed of the masses. Links are visco-elastic connections between two masses. The force generated by a visco-elastic link is :

$$\vec{F} = K\,\vec{X} + D\,\vec{V}$$

where K is the rigidity , D is the dampening, X is the elongation of the link, and V is the relative speed of two masses.

2.3 Time discretization

The new position of a mass: X(t) is defined by :

$$\vec{x}(t) = C \sum \vec{F} - \vec{X}(t-2) + 2\ \vec{X}(t-1)$$

where the constant C is defined by Weight of the mass and time discretization.

pmpd does not use specific units.

2.4 Forces and displacement

To allow real-time interaction, "mass" objects accept force and displacement messages from the user. Pd can then be used to provide automation or interaction between the user and the simulation. The extreme modularity of pd can offer a very large variety of interaction with the simulation.

Masses can also be translated without inertia by displacement message. Although this is "non-physical" behavior it can be useful for different reasons:

-If one knows the global equation of a movement, one can use displacement message to simulate this equation.

-this can be used to create any "unnatural" movements.

3 using pmpd with pd

3.1 Connections

"Mass" objects send position and receive force from "link" objects. "Link" objects receive the position of two masses and output forces for both of them.

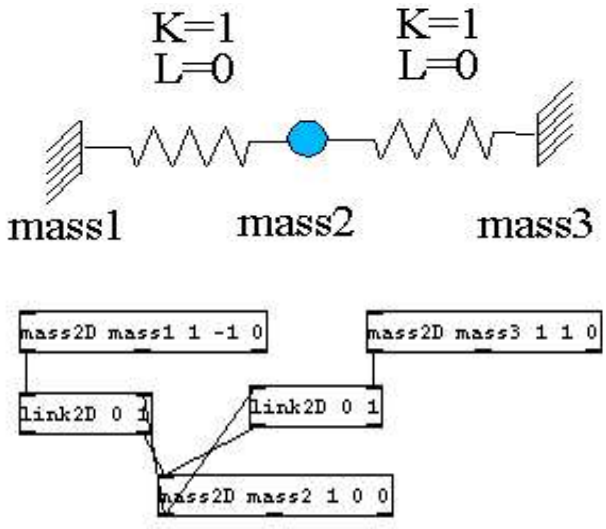

Figure 1: a small system, and it's equivalent using pmpd and pd

In figure 1, mass 1 and 3 do not receive forces (inlets are not connected), therefore they will never move; they act like a fixed point. Mass 2 is linked to 2 fixed points with springs with no dampening. Hence, movement will never end (it corresponds to a system without energy loss).

3.2 Name and interactions

To make patching simpler, mass and link objects have a name. It is used to receive information (pd messages). "Bang" messages can be passed to a set of masses and links without creating pd connection. It saves patching time and should be used to make patching easier.

In the same way, interactor objects behave like link objects but allow simpler patching. Essentially, a single object can create an interaction with an entire class of masses.

3.3 Test objects

These objects test the position—as well as distance, speed from a point, a line...—of a mass. Thereby, pd has access to a lot of data regarding the state of the system. This allows interaction with the rest of the patch. Another test object gives information about the link (deformation, speed of deformation, orientation...)

Figure 2. "tLink" example

The figure 2 is a 2D string. It shows how to use the "tLink2D" object to determine the size and orientation of a rendered link between two spherical masses.

3.4 Generality

It is possible to choose non-physical values for pmpd parameters. For example, you can set dampening to a negative value, which means the creation of energies. This is not physical and can lead to instability or saturation of the model, but can be useful for artistic reasons.

A lot of attention is required while changing parameters like rigidity. This can lead to energy creation or loss, depending on the deformation of the structure.

Using this kind of simulation, the most commonly encountered problem is instability. To reduce the risk of instability one's model should slowed down (increasing the metronome speed can be necessary to keep the desired speed of the simulation). Reducing force in the simulation is usually a good way to avoid instability problems. Initializing masses close to a stable state can also help.

Anyway, a standard personal computer is capable of computing the movement of about 100 particles at 500Hz which is much more than needed to get a smooth movement of a structure.

4 Examples and applications

All of the pmpd objects work with control data (as opposed to audio signals). This means that they do not generate audio directly. However, they can easily be used to *control* audio engines, i.e.: one cannot hear the sound of a vibrating string because it will not move fast enough; but one can use the movement of these particles along a string to perform additive synthesis.

All the figures of this article were made by using some of the pmpd files of examples (which can be downloaded at http://drpichon.free.fr/pmpd and tried). The provided examples are not necessarily aimed at making artful graphical effects, but rather at describing possible behaviors of a system. Testing the examples would give one a better overview of pmpd possibilities than static pictures.

4.2 Non-real-time sound synthesis

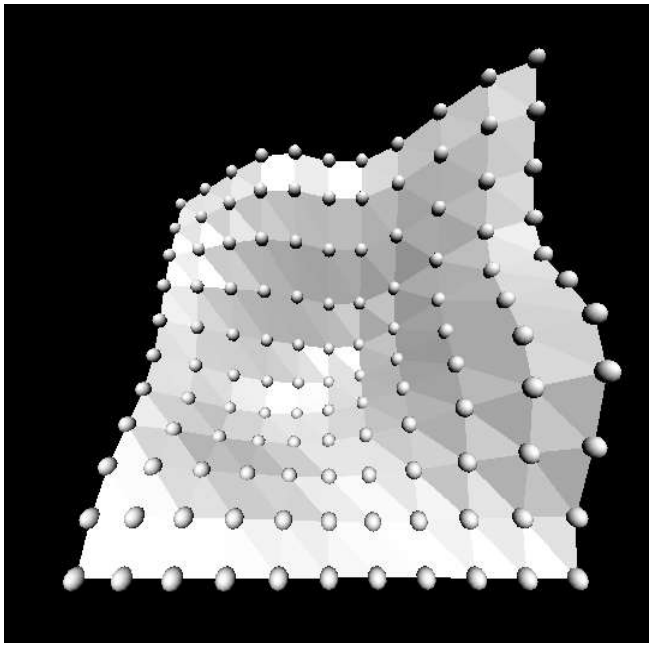

Figure 3 : A moving membrane

The movement of one of the masses in the figure 3 membrane is recorded on a wave table in the pd environment. Sound can be produced from this movement by using the recorded positions as audio wave-form.

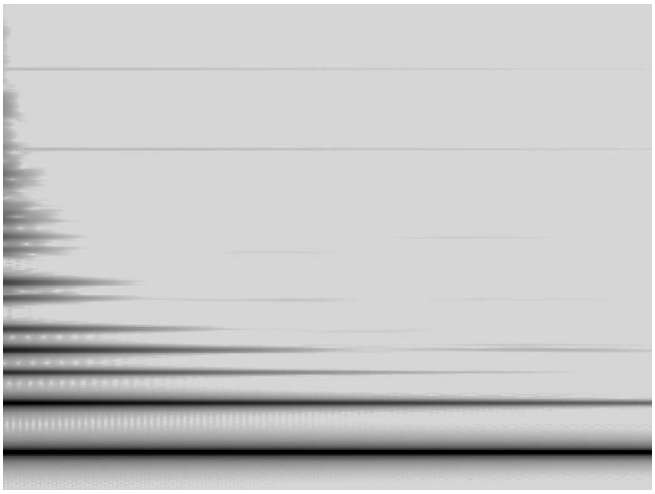

Figure 4. spectrum of a sound generated with pmpd

The figure 4 shows the spectrum of the first few seconds of the generated sound (the x-axis represents time, the y-axis represents frequency, darkness represents the amplitude of the specific frequency). This sound was produced using a membrane made with 121 masses moving in one dimension. The masses are distributed over the xy-plane. Initially, all the movement of the structure was generated by random z-positions of the masses. The movement of different points can be recorded and mixed to generate multi-channel sound. Physical properties of the structure can then be modified to change its sound (Avanzini and Rocchesso 2001, Sept.).

4.1 Complex movement

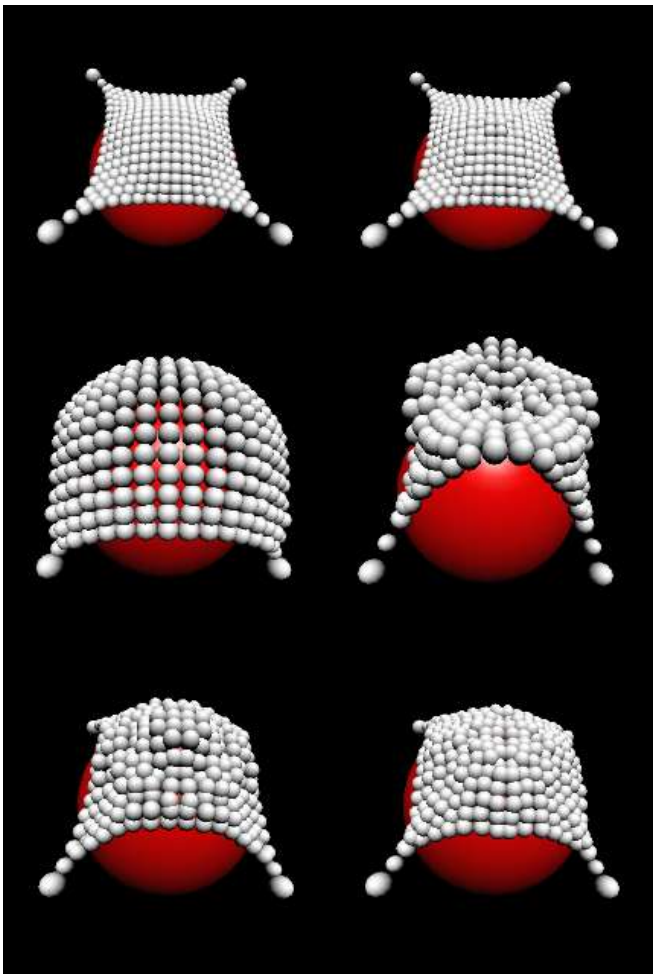

Figure 5. A "ball" interacting with a 3D elastic membrane

The figure 5 shows the deformation of an elastic membrane due to the interaction of a moving sphere (red).

4.3 real-time data flow generation

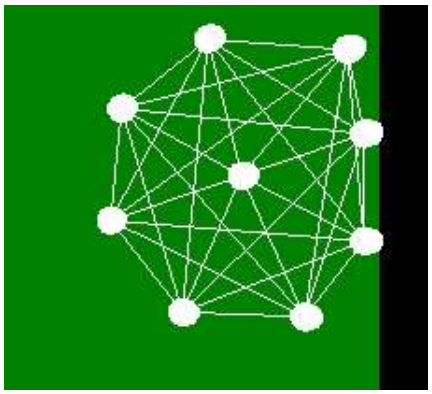

Figure 6. Bouncing "ball"

Figure 6 shows a structure bouncing off a wall. The internal forces of this structure were used to generate input for an additive synthesizer. This is an example of a "musical instrument" whose characteristics are defined by its topology, physical properties and the relationship between the structure and the synthesizer.

4.4 Non linear link

Many physical behaviors, like a hit or a continuous excitation of a musical instrument, cannot be modeled with linear equations (Avanzini and Rocchesso 2001). Pmpd uses pd programming environment to create specific interactions. An array can be used to draw the relationship between the deformation speed and the force generated by the link.

4.5 Perspectives

Pmpd can also be used for other purposes: generating chaotic movement, filtering control data (a single mass-link acts as a second order filter), rhythmic generation, composition etc. It is a collection of very simple objects whose mindful conglomeration can generate a very wide range of simulations used for many different applications.

References

Avanzini, F. and D. Rocchesso (2001a, Sept.). "Controlling Material Properties in Physical Models of Sounding Objects" *Proceedings of the International Computer Music Conference*

Avanzini, F., Rocchesso, D. (2001) "*modelling Collision Sounds: Non-Linear Contact Force" COST G-6 Conference on Digital Audio Effects* (DAFx01), Limerick, Ireland

Cadoz, C., A. Luciani and J.-L. Florens, (1993) "CORDIS-ANIMA : a modelling and Simulation System for Sound and Image Synthesis - The General Formalism", *Computer Music Journal* 17(1).

Castagne, N. Cadoz, C. (2002). "L'environnement GENESIS : créer avec les modèles physiques masse-interaction.", *Journées d'Informatique Musicale*, 9e édition, Marseille, 29 - 31 mai 2002

Djoarian, P. (1999). "Material design in physical modelling sound synthesis" Proc. of the 2nd COST G-6 Workshop on *Digital Audio Effects* DAFx99, NTNU, Trondheim, Dec. 9-11, 1999

Puckette, M. S. (1996). "Pure Data: another integrated computer music environment" *Proceedings of the International Computer Music Conference*, 37-41.

Rath, M., Rocchesso, D., Avanzini, F. (2002) "Physically based real-time modelling of contact sounds" *Proceedings of the International Computer Music Conference*

S.Rimell,S D,M.Howard, A,D.Hunt, P,R.Kirk and A,M.Tyrrell (2002) *"The Development of a Computer-based, Physically Modelled Musical Instrument With Haptic Feedback, for the Performance and Composition of Electroacoustic Music."* Proceedings of the 10th anniversary European Society for the Cognitive Sciences of Music Conference, Liege