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, specially 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. pmpd provides the basic objects for this kind of simulation. By combining pmpd's various objects one can simulate a very large variety of dynamic systems.

These object are designed to be used with pd: a real-time graphical programming environment dedicated for audio signal processing. Pd's graphical programming environment is well adapted of the creation of particular physical modeling. 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 to not only 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 modeling, namely 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.

On the other hand, some pd object provide simulation of complex behaviors, like various chaos objects. This kind of object can only model one kind of comportment and do not adapt themselves new and more sophistication dynamic behaviors and the interaction between numerous dynamic systems.

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 hight level objects (each pmpd object can be made as 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 evolved structure are needed for the simulation. pmpd can then easily be used for the simulation of a very large variety of comportment.

Complex simulations are basically made using 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:

$$\sum F = m\,\alpha$$

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

“mass” objects have outlets for their internal state; they report their position, the total force applied to them and their speed. These parameters can be used to generate control data for audio and video synthesis.

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 :

$$F = K\,X + D\,V$$

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

2.3 Time discretisation

Velocity at time = t: V(t) is define by:

$$\frac{X(t) - X(t-1)}{DT}$$

dT is the small time delta between two consecutive iterations. Similarly, acceleration at time t is define by:

$$\frac{V(t) - V(t-1)}{DT}$$

So :

$$X(t) = \sum F * C - X(t-2) + 2X(t-1)$$

where the constant C is given by:

$$C = \frac{dT^2}{M}$$

pmpd gives a default value of 1 to dT^2 .

This shows how to compute the current position of a mass while knowing the force applied to it and its last two positions.

2.4 Units

pmpd does not use specific units. Things are relative: a mass weighting 10 will react the same way to a force of 1 than a mass of 100 to a force of 10. Although chosing explicit units is not obligatory, if you choose to use units they must be consistent, i.e. you chose inches for the distance unit, then rigidity unit should be force unit divide by inches.

2.4 Forces and displacement

To allow real-time interaction, "mass" objects accept force and displacement messages from the user. pd can then be use 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.

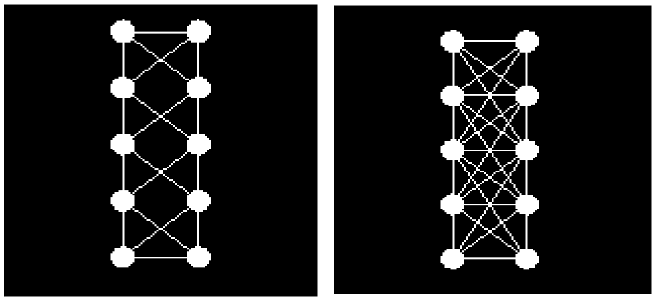
Masses can also be translated without inertia by to displacement message. Although this is "non-physical" behavior it can be useful for different reasons:

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

-You can use this to create "unnatural" movements of your desire.

2.5 Topology

A structure and its components can be modeled according to the topology, i.e. a defined set of interactions:



different dynamic systems can be modeled using different topologies.

Figure 1. Two different topology for the same object.

Figure 1 is an example of two different topologies. Masses are the spheres, link are the white lines. The same set of masses are linked in two different way. The physical behavior of this structure will differ according to its topology (Djoarian 1999).

3 using pmpd with pd

3.1 Connections

"mass" objects send position and receive force from "link" objects. "link" objects receive two mass positions and output forces for both masses.

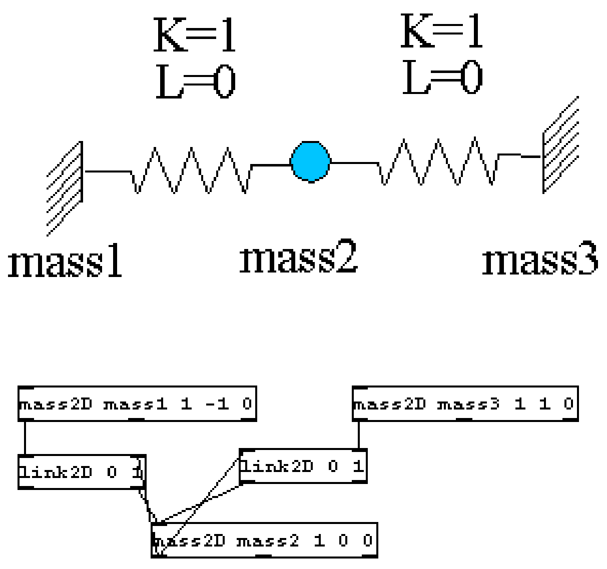


Figure 2: a small system, and it's equivalent using pmpd and pd

In figure 2, mass 1 and 3 do not receive forces (inlet 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 damping. Hence, movement will never end (it corresponds with a system without energy loss).

3.2 Metronome

A time reference is needed to compute the simulation. pmpd uses an external scheduler; the user has to send a scheduler event to pmpd object. This mechanism was chosen purposefully; the desired advantages were:

-You can easily change the speed of the simulation (adjust on the CPU speed).

-You can stop different parts of the simulation when you do not need them.

-You can synchronise to video rendering if desired, or other interactions from the user.

pmpd needs to compute all mass movements together and all link interactions together, an external metronome is well adapted to this task.

States of the model are computed in discreet time. i.e. displacement of a mass is not a function, but a finite serie of point: X(n). The metronome corresponds with the time discretization of the equations. The speed of the metronome should depend on the speed of the movement you want to simulate. There is no general law to know the metronome speed, however, the faster the metronome, the faster the simulation can be. The metronome should be faster than the highest frequency of the movement you want to simulate.

3.3 Name

For patching simplification mass and link objects have a name. It is the first argument for the object creation. It is used to receive information (pd messages). All masses with the same name defined a class of mass.

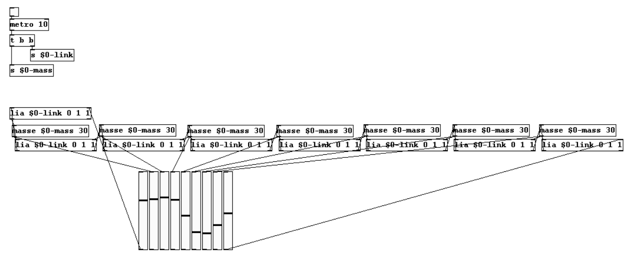


figure 3: sending bang without connection

Figure 3 shows how “bang” messages are pass to a set of masses and links without creating pd connection. It can save patching time and should be used for patch simplification.

3.4 Interactors objects

Interactor objects behave like link object but provide patching facility. Essentially, a single object can create an interaction with an entire class of masses. Interactor objects must be created with a name. This name is the name of the masses it is connected to.

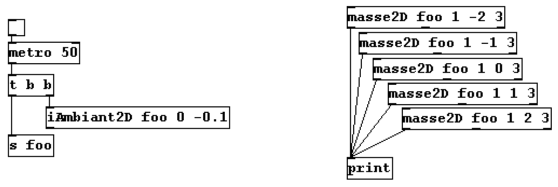


Figure 4: sending forces to a class of mass

In Figure 4, all masses are subject to an ambient constant force. This force can viewed as gravity force applied to every mass named “foo”. Masses with other names will not be subject to this interactor. Different interactor objects provide interactions with a point, a line and other simple primitives.

3.4 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 much informations 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...)

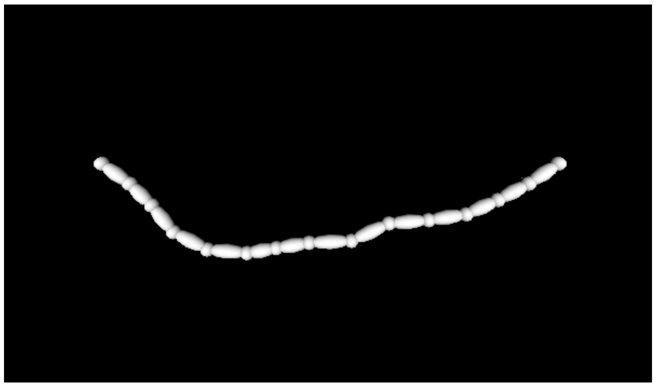


Figure 5. “tLink” example

The figure 5 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.

3.5 Generality

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

You should also take care while changing parameters like rigidity. This can lead to energy creation or lost, depending of the deformation of the structure.

The most commonly encountered problem while using physical modelling is the instability. To reduce the risk of instability you should slow down your model (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 attack instability problems. Initialising masses close to a stable state can also help.

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.: you can not hear the sound of a vibrating string because it will not move enough fast; but you can use the movement of this particles along a string to perform additive synthesis.

All figure of this articles were made using some of the pmpd examples files. You can download [http://drpichon.free.fr/pmpd] and try them. The aim of the provided examples are not necessarily to make artful graphical effects, but rather to describe possible behaviors of a system. Trying the examples would give you a better overview of pmpd possibilities than static pictures.

4.1 Complex movement

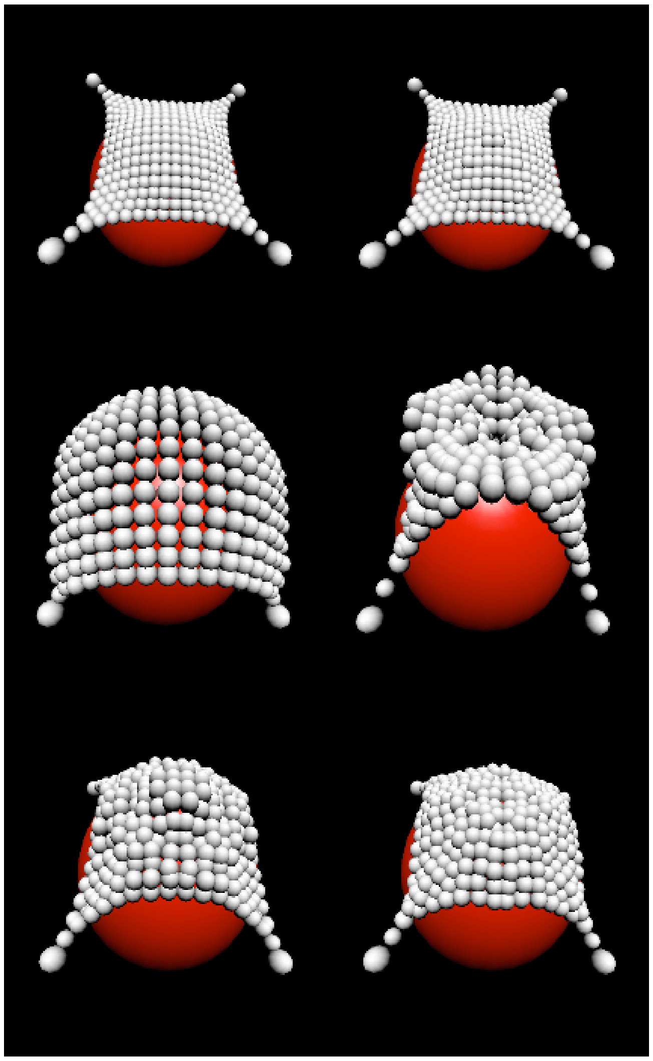


Figure 6. A “ball” interacting with a 3D elastic membrane

The figure 6 shows the deformation of an elastic membrane due to the interaction of a moving sphere (red sphere).

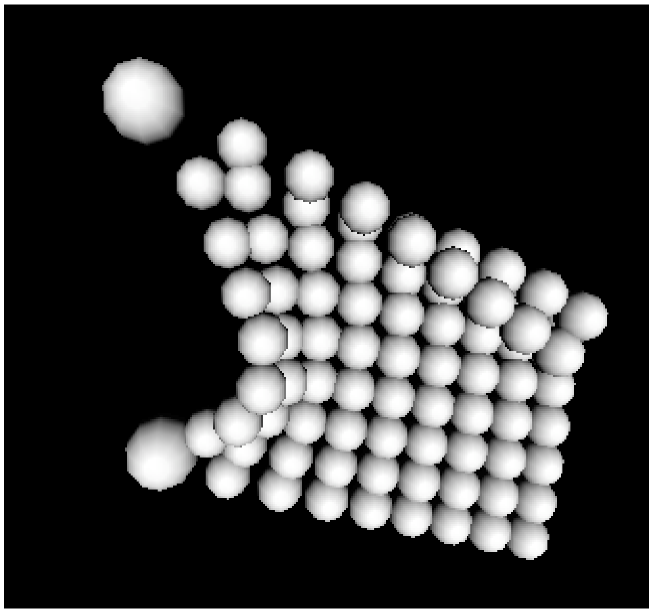


Figure 7 : a “flag” in the wind

The figure 7 is a snapshot of the movement of a flag made with 81 particles. Each particle is link to it's 8 neighbors. They move in 3D. A standard personal computer is able to compute the movement of this 81 particles at 500Hz which is much more than needed to get a smooth movement of this structure.

4.2 Non-real-time sound synthesis

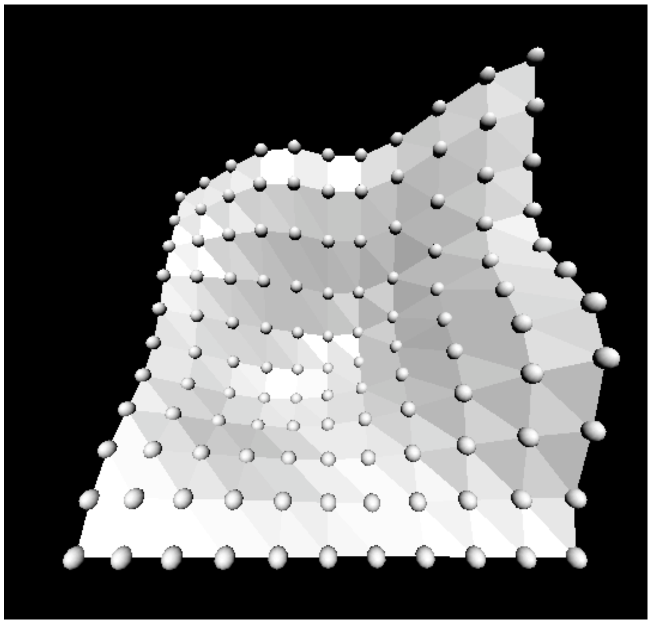


Figure 8 : a moving membrane

The movement of one of the masses in the figure 8 membrane is recorded in the pd environment. Sound can be produced using this movement by using the recorded positions as a wave table.

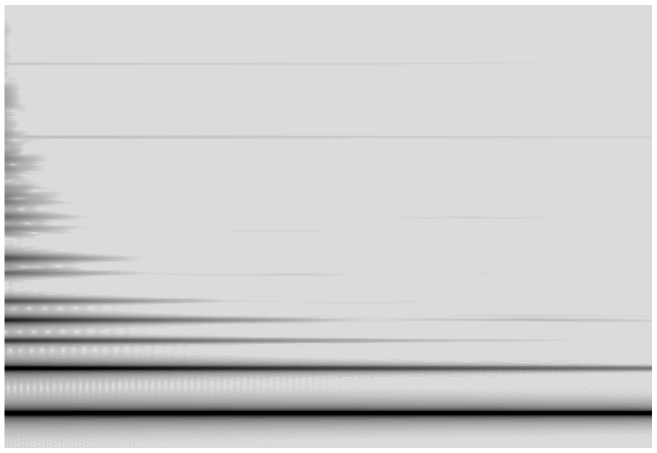


Figure 9. spectrum of a sound generated with pmpd

The figure 9 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 random z-positions of the masses generated all of the movement for the structure. The movement of different points can be recorded and mixed for generating multi-chanel sound. Physical properties of the structure can then be modified to change it's sound (Avanzini and Rocchesso 2001, Sept.).

4.3 real-time data flow generation

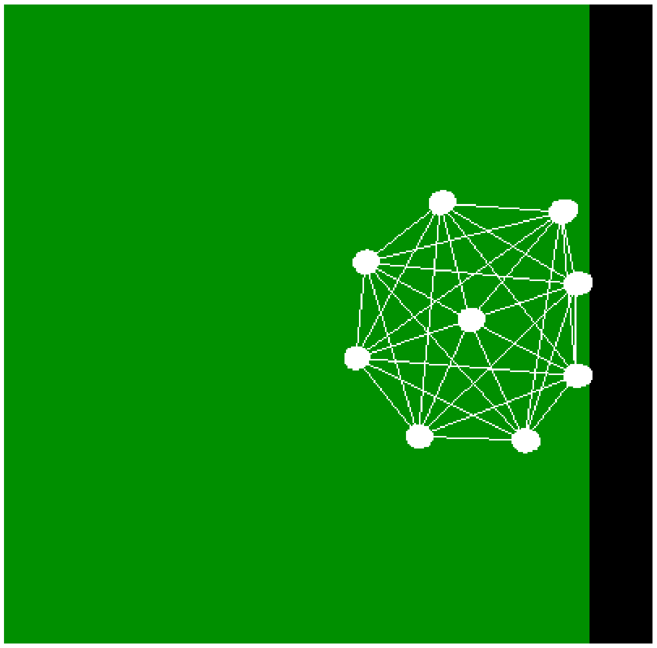
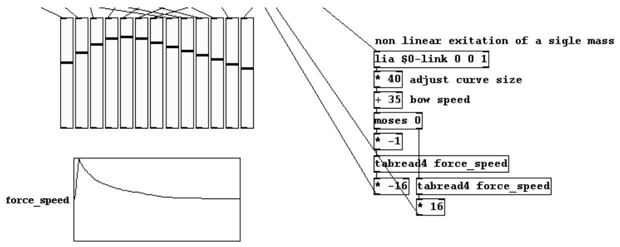


Figure 10. Bouncing “ball”

Figure 10 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 can not be modeled with linear equations (Avanzini and Rocchesso 2001).



```
non linear exitation of a sigle mass
lia $0-link 0 0 1
* 40  adjust curve size
+ 35  bow speed
moses 0
* -1
tabread4 force_speed
* -16  tabread4 force_speed
       * 16
```

Figure 11. using table to allow non linear interaction

Figure 11 shows an example of non linear link. A simple pd “table” object allows the possibility of non-linearizing the relationship between the deformation-speed and the force generated by the link. Such non-linearities can help model motions like that of a bow on a string, i.e. Helmholtz motion.

4.5 Perspectives

Pmpd can also be used for other usage: generation of chaotic movement, filtering control data (a single mass-link act as a 2nd 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*