Status of MSE mechanical simulation library

G. Cella

31st July 2003

Contents

1 Library architecture

The MSE library is a C++ set of classes that can be used to simulate mechanical systems in both frequency and time domain. A simplified class diagram is depicted in Figure 1.

The fundamental class is the ***Object*** one, which is a schematization of a generic mechanical element. Basically we can think of it like an effective description of the static and dynamic interactions between a finite set of frames fixed on the modeled object. Each frame is completely specified by a set of six parameters, three for the position of its origin and three for its orientation in the space. For example, in the special case of the ***Wire*** class there are two frames, one fixed at the top and another at the bottom of the wire.

The ***Object*** class provides several methods that are used to describe its mechanical properties, in particular:

1. a method to evaluate its potential energy, for a given position and orientation of its frames. This method is used in the static regime to find the working point position of the system. Also first and second derivatives of the potential energy around the current position of the frames are provided, and can be used by special working point search algorithm (a prototype is the Conjugate Gradient algorithm which uses the potential energy and its first derivatives).

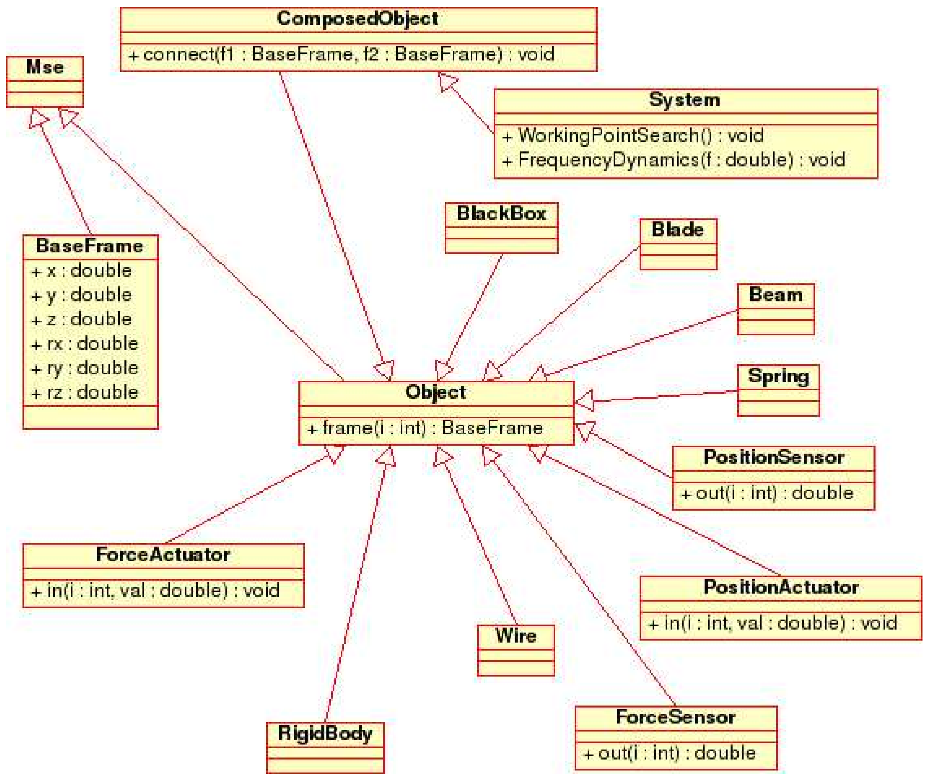

Figure 1: Simplified class diagram of the MSE library.

2. a method to evaluate the effective Lagrangian of the system in the frequency domain. This one is evaluated numerically at a particular frequency in the linearized regime around the current working point, and it is represented by a square matrix $\mathcal{L}$ with dimension equal to (6×number of frames), which is the number of degrees of freedom of the object. It is used to describe the response of the system in the frequency domain.

3. a method to evaluate the stiffness, damping and mass matrix K, Λ and M of the system. These can be seen as the first three pieces of the Taylor expansion of the effective Lagrangian in powers of ω^2[1]

$$\mathcal{L} = -\frac{1}{2}\left(K + i\omega\Lambda - \omega^2 M\right) + O(\omega^3)$$

   These quantities can be used to directly evaluate an approximate time evolution of the system in the time domain, neglecting internal modes and damping of non viscous nature, as esplained below.

To build a model for a particular mechanical system we connect together the fundamental building blocks available. The recommended way (though not the mandatory one) is to create a class derived from ***System***. ***System*** is mainly a container, but it is itself a mechanicall object which provides a unique peculiar frame which represent the inertial system where the dynamics is described. The connection between two mechanical objects is obtained using the ***System::connect*** method, which implement a rigid connection between two frames.

Fundamental building blocks currently available or foreseen in the library are the following:

Spring is the a simple spring with a given longitudinal stiffness and rest length. It has no mass and no internal structure. It has a frame at each edge, for a total of 12 degrees of freedom.

RigidBody is a simple rigid body with a given mass and inertia tensor. It has no internal structure. There is a single frame fixed on the center of mass, for a total of 6 degrees of freedom.

Wire is a wire with of a given material, with a given section and rest length. Longitudinal, torsional and transverse (when tensioned) stiffness are implemented, and a complete modelization of internal modes is available. Flexural stiffness is neglected. There is a frame at each edge, for a total of 12 degrees of freedom.

Blade is a complete modelization of a rectangular blade, including internal modes. There is a frame on each edge, for a total of 12 degrees of freedom. Tension effect are neglected.

[1]There is a slight abuse of language here, because dissipation cannot be introduced in a Lagrangian description, at least in the time domain. In the frequency domain there are no problems, and we can use the Lagrangian definitions like a bookkeping device.

Beam is a complete modelization of a blade, including the effect of tension (or compression) and internal modes. There is a frame on each edge, for a total of 12 degrees of freedom.

BlackBox is a container for a mechanical object with two frames, whose behaviour can be specified giving numerically the frequency domain response functions. It can be used to introduce in the simulation a subsystem with a description measured experimentally or known only numerically.

Additionally, some simple actuators and sensors are available:

PositionSensor is a simple idealization of a position sensor. There are two free frames which can be connected to other objects. The sensor generate an output of 6 numbers which describes their relative linear displacements (translation and rotations) during the frequency or time domain dynamics.

ForceSensor is a simple idealization of a position sensor. There are two frames which are rigidly connected together from the start, and can be connected to other objects. The sensor generate an output of 6 numbers which describes the small forces and torques variations between the two frames during the frequency or time domain dynamics.

PositionActuator is a simple idealization of a position actuator. There are two frames which are rigidly connected together from the start, and can be connected to other objects. The small variation (position and rotation) between these can be specified as an input of 6 number during the frequency or time domain dynamics.

ForceActuator is a simple idealization of a force actuator. There are two free frames which can be connected to other objects. The small force and torque between these frames can be specified as an input of 6 number during the frequency or time domain dynamics.

Other classes, which are not specified in Figure 1, can be classified in the following groups:

- A library of different methods for working point search.
- A database of materials. Each element is a simple class with methods to obtain the more commonly used parameters (Young modulus etc.)
- A database of shapes. Each element is a simple class with methods to obtain the inertia moments.
- A database of sections. Each element is a simple class with methods to obtain the commonly used parameters connected with the section, like the torsional stiffness constant.

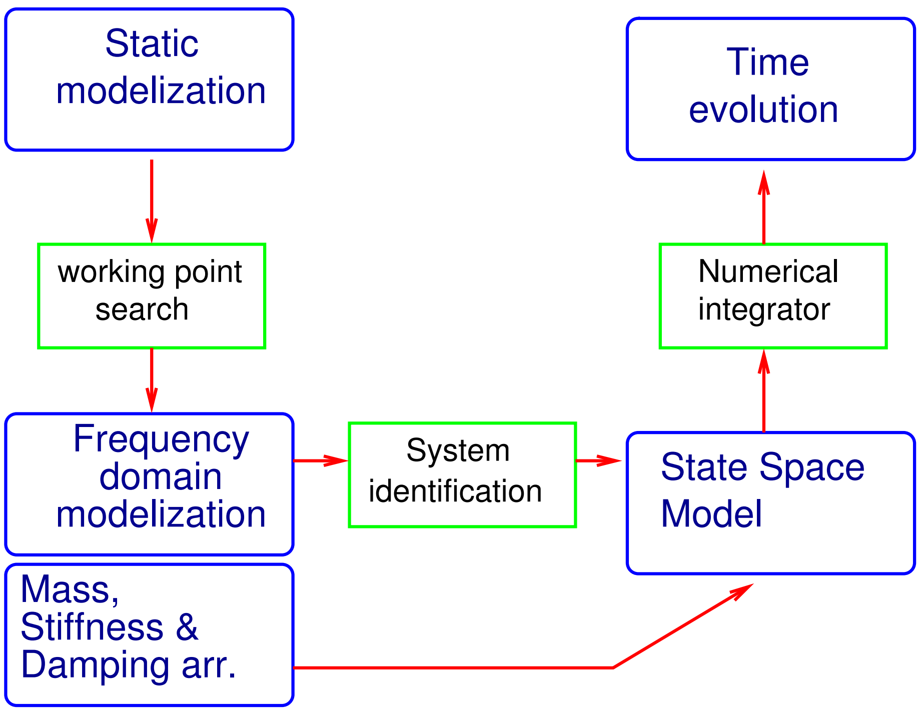

Figure 2: A logical diagram illustrating uses of MSE library.

A logical diagram of the dependency between different part of the library is depicted in Figure 2.

The simpler procedure is the evaluation of frequency domain quantities. The working point search is used after the construction of the system to correctly put it in the working point. After this there are essentially two kind of quantities that can be obtained:

1. transfer functions: some values (in modulus and phase) are set on the actuators, and the library evaluate the output of the sensors, at a particular frequency. Iterating this step the required set of transfer functions on the desired frequency range can be obtained.
2. thermal noise: setting the temperature, the thermal noise spectrum is evaluated.

In order to obtain the time domain evolution two different paths are foreseen.

In the first the relevant set of transfer functions in the frequency domain is used, using a system identification procedure, to generate a state space model for the system. In principle in many cases[2] no approximations are introduced before the system identification phase both for internal modes and dissipation, in the sense that the analytic modelization of the subsystem is used when available. So it is possible, for example, to specify a loss angle for a material which is an arbitrary function of the frequency. When a finite dimensional state space model is generated some approximations must be introduced, both for dissipation and internal modes. An important point is that these approximations can be optimized as a function of the bandwidth where the time domain model is requested to correctly reproduce the physics.

A second option is to use the approximate description in term of the K,Λ and M arrays. This description can promptly be converted in a state space model. However it is important to understand that both internal modes an peculiar features of a non viscous damping cannot be modelized in the $K\Lambda M$ formalism[3].

Once obtained the state space description it is quite easy to obtain the time evolution using one of the standard time integrators.

2 Status before my visit to Caltech

An initial version of the library, which contain a modelization in the $K\Lambda M$ formalism, was completely redesigned. The implementation of the new version is based on the approach discussed in the last section. Main features introduced in the new version, available and tested before my visit are:

[2] An analytic description of a subsystem is not avaliable in the general case. As an (important) example, the frequency domain Lagrangian of a triangular blade cannot be obtained in closed form. In this case an approximation is introduced modeling the blade as a set of connected rectangular blades of different dimensions.

[3] In fact, this is not entirely true. For internal modes, it is possible to obtain them simply discretizing a mechanical object in several subpieces. A similar approach could be used also to obtain a better approximation for dissipation, but it seems to me more involved.

- Full analytic description in the frequency domain of ***RigidBody***, ***Spring***, ***Wire*** objects.
- Redesigned ***PositionActuator*** and ***ForceSensor***. The new version avoid the use of constraint, which were implemented as Lagrangian multiplier. In this way the numerical behaviour and the computational cost is reduced, and a simple translation in the state space formalism is possible when the $K\Lambda M$ approach is used.
- New parametrization of rotations. Old parametrization was based on Euler's angles, the new one on the Lie algebra and quaternion representation. Numerical and analytical problems are reduced in this way.
- The approach for system composition was changed. The frames are positioned in absolute coordinates before connection, while their relative positions was requested before. This is much more intuitive and less error prone. Also, frames can be moved also before the working point search.
- Library is available with different numerical precisions, that is double, $2\times$ double and $4\times$ double precision.
- Methods for working point search and for the solution of the frequency dynamics in the frequency domain was reimplemented from scratch. The new code is parametric in the numerical precision and optimized for accuracy. Different algorithm are avaliable, from *LU* decomposition (relatively low computational cost) to *QR* decomposition with full pivoting (very high numerical stability).
- A mechanical system and its status (as for example its working point) can be saved on a file, and reloaded when needed.
- The graphical part of the library was separated from the main code. More precisely, a set of callbacks are provided to the user, that can in this way develop independently a particular application. As an example a very simple GUI application is avaliable, which can be used to verify in an interactive way both working point search and frequency domain dynamics.

The library was applied to the modelization of a set of mechanical objects, like the LF facility experiment working in Pisa. This is a work in progress, but initial results seems promising.

3 Work done during my visit to Caltech

During my visit in Caltech, together with V. Sannibale, we concentrated on some items that seems to give problems in past applications:

1. Internal modes of wires: the particular application was that of a LIGO-type suspended mirror, actuated with four coils.

2. Modelization of the blade. This is required for the construction of a quadruple pendulum-type suspension.

3. Modelization of LIGO stacks. The modelization of a stack gave a qualitative agreement, but wrong values for the frequencies of the vibrational modes.

As a result of our work

1. We verified, comparing the result of the simulation with analytical results, that internal modes of the wires are correctly reproduced with the new version of the library.

2. A complete description of the ***Blade*** object was implemented. Preliminary tests shows that also in this case low frequency dynamics and internal modes are correctly reproduced, as they agree with analytical calculations when these are available.

3. A complete description of the ***Beam*** object was implemented. It was not yet estensively tested.

A unresolved issue is connected with the LIGO stack. This is a separate problem, which require an analysis for the ad-hoc approach used for the modelization of the stack supports.

4 Future plans

In the near future the test of ***Beam*** object is foreseen. This will be applied to three types of problems:

- the VIRGO inverted pendulum
- the thermal noise of the LIGO suspended mirror. As the ***Wire*** object neglect flexural stiffness the correct damping factor cannot be introduced. This is possible modeling the suspension's wires as thin beams
- the LIGO stack, using ***Beam***s to modelize the supports

The ***Blade*** object will be used to build a set of quadruple pendulum-type suspensions. This will be the occasion to see if the modelization of a triangular blade as a set of connected rectangular blade will give accurate enough results. Also, the modelization of a complete VIRGO superattenuator is foreseen.

Two main point, which require a larger amount of time, remain to be solved:

- The methods for $K\Lambda M$ formalism, that were eliminated in the new version of the library, must be reintroduced.

- The system identification block depicted in Figure 2 must be implemented and tested.

The last point is the more demanding one. Standard tools for system identification exists (for example in MATLAB). We will start testing these on the results of MSE library.