

LASER INTERFEROMETER GRAVITATIONAL  
WAVE OBSERVATORY  
- LIGO -

CALIFORNIA INSTITUTE OF TECHNOLOGY  
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Document Type T030158-00-R July 28, 2003

**Status of MSE  
the Mechanical Simulation Library for E2E**

Giancarlo Cella, Virginio Sannibale

*Distribution of this draft:*  
TBD

This is an internal working note  
of the LIGO Project.

California Institute of Technology	Massachusetts Institute of Technology
LIGO Project - MS 18-34	LIGO Project - MS 20B-145
Pasadena, CA 91125	Cambridge, MA 01239
Phone (626) 395-2129	Phone (617) 253-4824
Fax (626) 304-9834	Fax (617) 253-7014
E-mail: info@ligo.caltech.edu	E-mail: info@ligo.mit.edu

WWW:<http://www.ligo.caltech.edu/>

# Contents

1	Library Architecture	1
2	Status of the Library on June 2003	5
3	Recent Accomplishments	6
4	Future Plans	7

## 1 Library Architecture

The MSE library is a C++ set of classes designed to simulate mechanical systems in small the oscillation regime both in frequency and time domain. A simplified basic class diagram is depicted in Figure 1.

As show in the mentioned figure, the fundamental class is the *Object* class, which is a schematization of a generic mechanical element. Basically, we can think of it as an effective description of the static and dynamic interactions between a finite set of frames fixed on the mechanical element. 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 *Wire* class there are two frames, each one fixed at one wire's end.

The *Object* class must provide several methods to describe the mechanical element properties. In particular, it provides :

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 the minimum of the potential energy, the so called working point position of the system.  
Methods to compute first and second derivatives of the potential energy around the current position of the frames are provided also, and can be used by special working point search algorithms (a prototype method uses the Conjugate Gradient algorithm, which uses the potential energy and its first derivatives).
2. a method to evaluate the effective Lagrangian of the system in frequency domain. The Lagrangian is numerically evaluated at a particular frequency in the linearized regime around the current working point. It is represented by a frequency dependent square matrix  $\mathcal{L}$  with dimensions equal to  $N = 6 \times \text{number of frames}$ , which is the number of degrees of freedom of the mechanical system. It is indeed used to describe the response of the system in the frequency domain.
3. a method to evaluate the stiffness, damping and mass matrix  $K$ ,  $\Lambda$  and  $M$  of the system. These can be seen as the first three elements of the Taylor expansion of

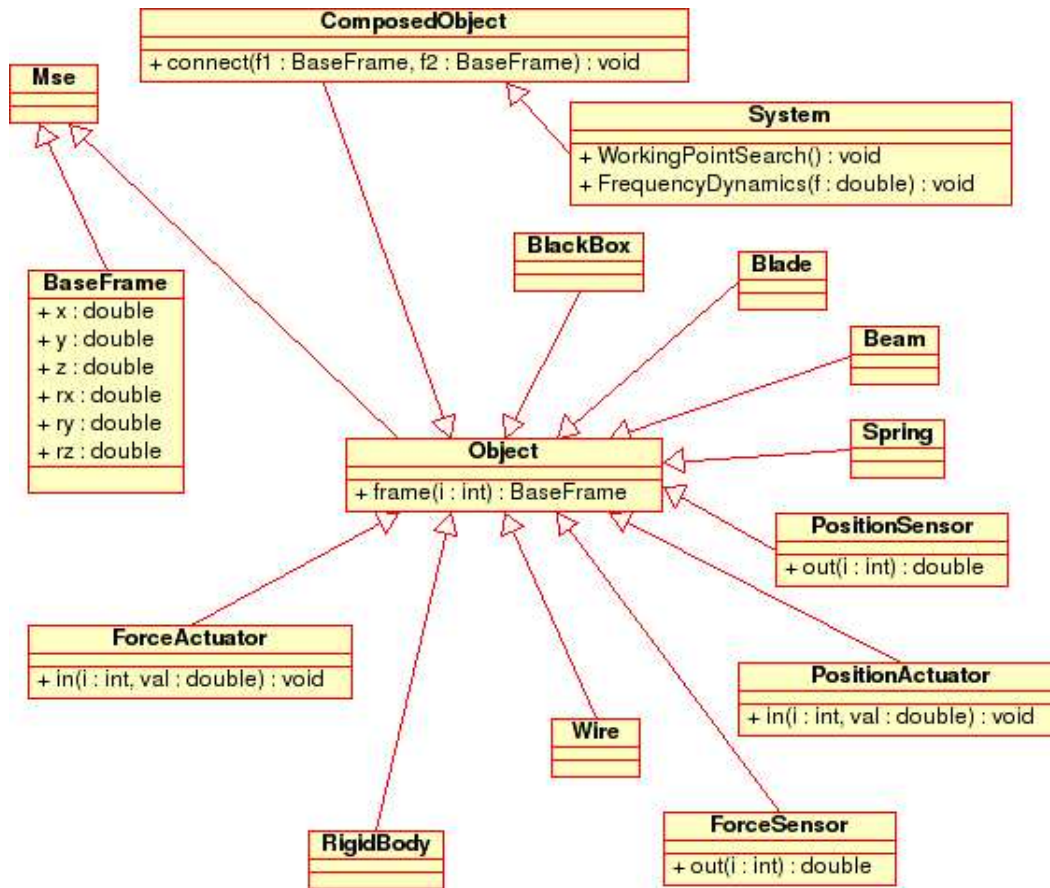


Figure 1: Simplified classes diagram of MSE library.

powers of  $\omega^2$  of the effective Lagrangian <sup>1</sup>

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

These quantity 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 explained below.

To create a model of a particular mechanical system, we connect together the available fundamental building blocks/*Objects*. The recommended way (though not mandatory) is to create a class derived from *System*. *System* is mainly a container, but it is itself a mechanical object which provides a single peculiar frame, which represent the inertial

<sup>1</sup>There is a slight of 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 bookkeeping device.

system where the dynamics is described. The connection between two mechanical objects is made using the *System::connect* method, which implements 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 on each end, 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 to 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 model 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 model 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.

**Beam** is a complete model 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 behavior 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 basic 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.

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<sup>2</sup> no approximations are introduced before the system identification phase both for internal modes and dissipation, in the sense that the analytic model 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

---

<sup>2</sup>An analytic description of a subsystem is not available 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.

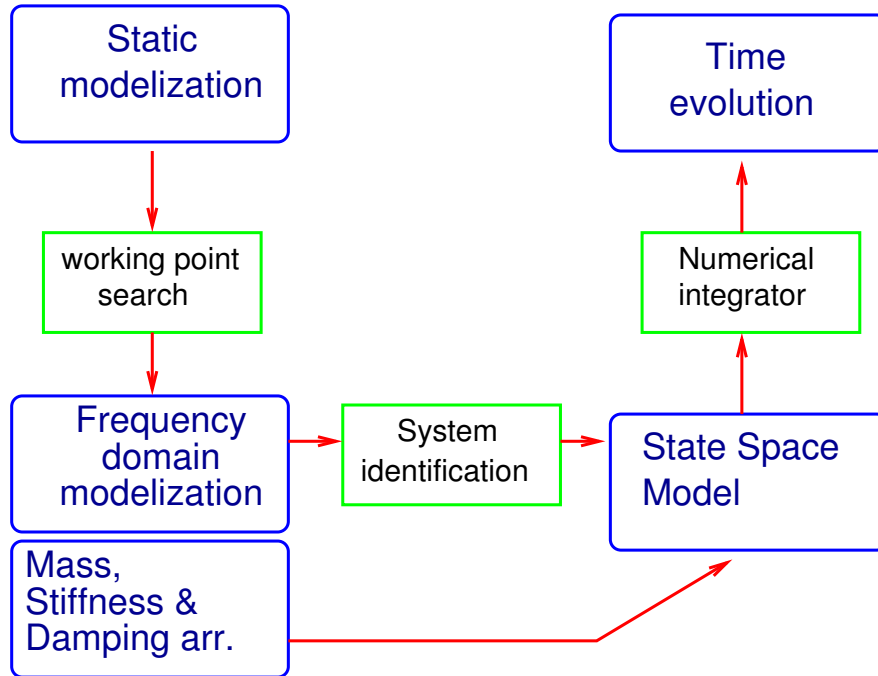


Figure 2: A logical diagram illustrating uses of MSE library.

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  $K, \Lambda$  and  $M$  arrays. This description can promptly be converted in a state space model. However it is important to understand that both internal modes and peculiar features of a non viscous damping cannot be model in the  $K\Lambda M$  formalism<sup>3</sup>.

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

## 2 Status of the Library on June 2003

The initial version of the library, which contains a model 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

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

tested:

- Full analytic description in the frequency domain of *RigidBody*, *Spring*, *Wire* objects.
- Redesigned *PositionActuator* and *ForceSensor*. The new version avoid the use of mechanical constraints, which were implemented as Lagrangian multiplier. In this way the numerical behavior and the computational cost is reduced, and a simple translation in the state space formalism is possible when the *KLM* approach is used.
- New parameterization of rotations. Old parameterization was based on Euler's angles, and the new parameterization 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 after 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 available, 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 call backs are provided to the user, that can in this way develop independently a particular application. As an example a very simple GUI application is available, which can be used to verify in an interactive way both working point search and frequency domain dynamics.

The library was applied to the model 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 Recent Accomplishments

We concentrated (during Giancarlo Cella's visit) on some items that gave the major problems in past applications. These are:

1. Internal modes of wires: the particular application was the LIGO suspension, actuated with four coils.
2. Model of the blade, required for the construction of the advanced LIGO suspensions.
3. Model of LIGO stacks. The model of BSC and HAM stacks gave a qualitative agreement, but wrong values for the frequencies of the vibrational modes.

These are the accomplishments that have been achieved

1. We verified, comparing the result of the simulation with analytical results, that internal modes of a single wire 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 extensively tested yet.

An unresolved issue is connected with the LIGO stacks. This is a separate problem, which requires an analysis ad-hoc for the model of the stacks system.

## 4 Future Plans

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

- VIRGO inverted pendulum
- Thermal noise of the LIGO suspensions and Advanced LIGO Suspensions. 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.
- LIGO stack, using *Beams* to model 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 model of a triangular blade as a set of connected rectangular blade will give accurate enough results. The model of a complete VIRGO super attenuator is foreseen also.

Two main points, 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 most demanding. Standard tools for systems identification are available (for example, in MATLAB). We will start using those tools using the actual MSE results to check the feasibility of this solution.