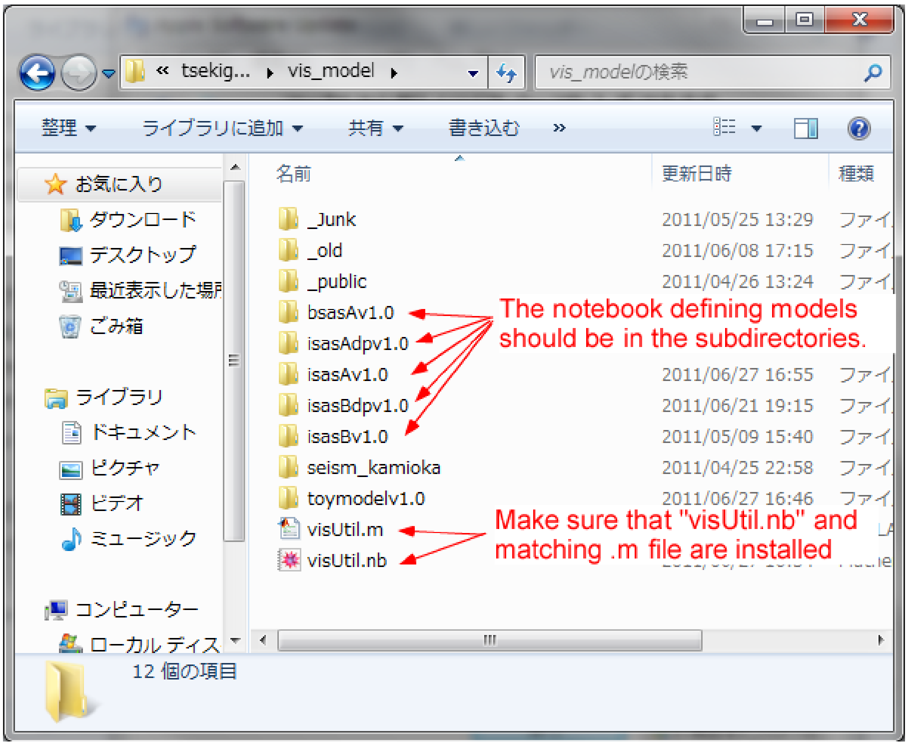
Usage Notes

04/26/11 T. Sekiguchi

Installation

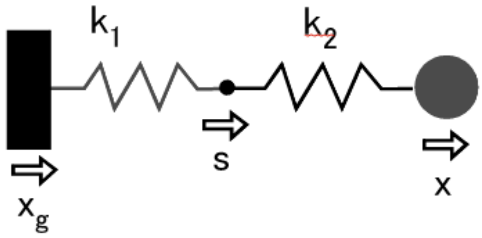
1. Install the file “visUtil.nb” and the matching .m file “visUtil.m” in a convenient directory. Make sure that notebooks defining models are in the subdirectories (e.g. isasAv1.0).
2. The notebook “visUtil.nb” is set as a Mathematica autosave package, so that whenever it is saved, the matching .m file is automatically updated. If you need to change the utility package, make the changes to the file “visUtil.nb”. If you add any new cells in this file, be sure to set the initialization cell attribute.



Usage

1. The notebook in the subdirectory (e.g. isasAv1.0 / isasA110426.nb) contains definitions of the model, setup and some codes to output 3D graphics, transfer function plots, and etc.
2. Make sure that in the notebook there is a code to import the utility package in the parent directory (`<<"../visUtil.m"`).
3. To define the model, you need to make a list of variables (`allvars`, `allparams`, `allfloats`), bodies (`bodylist`), wires (`wirelist`), springs (`springlist`) and dampers (`damperlist`).
4. Note. The `allvars` should be a list of the position and angle variables for all elements of the system that have mass/MOI associated with them. The `allparams` should be a list of positions and angles describing the state of the ground. The `allfloats` should be a list of positions and angles of connections where one elastic element is connected directly to another with no mass element between.

e.g.) Oscillator with two mass-less springs




```
allvars={x}; allparams={xg}; allfloats={s};
```

5. Read the usage notes of `bodylist`, `wirelist`, `springlist` and `damperlist` and define them. Their usage notes can be obtained by typing for example, `?bodylist` onto a blank cell.
6. Potential energy, kinetic energy and damping energy of the system are to be calculated from the lists defined above. Potential energy is divided into four parts: wire stretching potential, wire torsional potential, spring potential, and gravity potential (+ other potential energy if you want). These energy terms can be obtained by functions such as `makewirepot` and etc.
7. Use `FindMinimum` function to find an equilibrium point of the system. Keep in mind that `FindMinimum` is a function to search for a "local" minimum point in a given potential, so that you need to choose appropriate starting points (initial values).
8. Differentiate the potential energy of the system with respect to pairs of coordinates at equilibrium to create the stiffness matrix `matKxx`. In a similar way, differentiate the kinetic energy and damping energy with respect to coordinate velocities to create the mass matrix `matMxx` and damping matrix `matGxx`.

$$K_{ij} = \left.\frac{\partial E_P}{\partial x_i \partial x_j}\right|_{x=x_{(eq)}} ,\quad M_{ij} = \left.\frac{\partial E_K}{\partial \dot{x}_i \partial \dot{x}_j}\right|_{x=x_{(eq)}} ,\ G_{ij} = \left.\frac{\partial E_D}{\partial \dot{x}_i \partial \dot{x}_j}\right|_{x=x_{(eq)}}$$

9. The matrix `matKvv` is a differentiation of the potential energy with respect to pairs of variable coordinates. The matrix `matKpv` is a differentiation of the potential energy with respect to pairs of parameter coordinates and variable coordinates. In a similar way, `matKfv`, `matKfp`, `matKff`, `matMvv`, `matMpv` and `matGvv` are defined (`f` means float coordinates). From these matrices, effective stiffness matrices are calculated as:

$$K^{(\mathrm{eff})}{}_{\mathrm{vv}} = K_{\mathrm{vv}} - K_{\mathrm{fv}} K_{\mathrm{ff}}{}^{-1} K_{\mathrm{fv}}{}^{\mathrm{T}}$$
$$K^{(\mathrm{eff})}{}_{\mathrm{pv}} = K_{\mathrm{pv}} - K_{\mathrm{fv}} K_{\mathrm{ff}}{}^{-1} K_{\mathrm{fp}}{}^{\mathrm{T}}$$

For more details, see T020205-02-D ("Models of the Advanced LIGO Suspensions in Mathematica" by Mark Barton).

10. Diagonalize the stiffness and mass matrices (`Inverse[matMpv].matKvv`) to obtain the eigenfrequencies and eigenmodes of the system. Eigenvalues and eigenvectors of a square matrix can be obtained by a Mathematica function `Eigensystem`.

$$(M_{\mathrm{vv}}{}^{-1} K_{\mathrm{vv}})\vec{e}_i = (2\pi f_i)^2 \vec{e}_i$$

`eigenmodetable` is a function to generate a table of the eigenmode list and `eigenplot` is a function to generate 3D graphics of n-th eigenmode.

11. Use `tfplot`, `tfplotf` and `tfplota` to plot transfer functions from disp to disp, and from force to disp. To read their usage, type `?tfplot`, for example.
12. When simulating the system in Matlab environment, it is convenient to use state-space models. The notebooks contain a code to create state-space matrices from stiffness, mass and damping matrices. Structure damping is approximated by viscous damping in the state-space model since all the elements of state-space matrices must be real and not be frequency dependent. `matlabexport` and `matlabexportappend` are functions to export the matrices in Matlab format.