Multibody Simulation with Python (Talk at Europython 2015 in Bilbao)

Dr. O. Braun[1,2] Dipl.-Ing. J. Eckstein[1]

NuCOS (Numeric Cooperative Simulation)[1]

Center of Nonlinear Technology[2]

July 2015

Outline

Outline

Outline

New package name: mubosym

Aim

Using existing python packages to provide an advanced and (once) complete multibody simulation environment

Why this is important

- Independent from market leaders
- Scripting capability included
- Education

NuCOS
Numeric Cooperative Simulation

Multibody simulation

What is MBS?

Systematic approach to optain and solve Newton-Euler's equation of motion:

$$\vec{F} = m\vec{a} \tag{1}$$

$$\vec{M} = \dot{\vec{L}} + \vec{\omega} \times \vec{L} \tag{2}$$

Use Cases

- Mechanical Engineering
- Ground Vehicle Dynamics
- Robotics
- Biomechanics

NuCOS Numeric Cooperative Simulation

Used packages

Sympy

- Symbolic algebra package
- Includes already advanced Mechanics
- Replacement of Mathematica ®, Maple ®, MATLAB ®symbolic toolbox

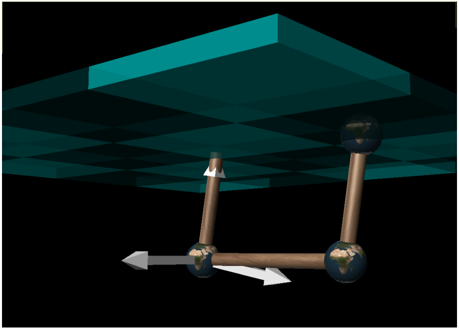
NuCOS
Numeric Cooperative Simulation

Used packages

NumPy/SciPy

- Well-known and tested numerical packages
- Linear algebra Solvers in Numpy
- ODE solvers in Scipy

Used packages



VPython

- 3d-Graphics Package
- Geometric primitives available (rod, spring, sphere, box)
- Fast enough for nice additional features like coordinate systems, forces ...

Outline

Building blocks of a mechanical system

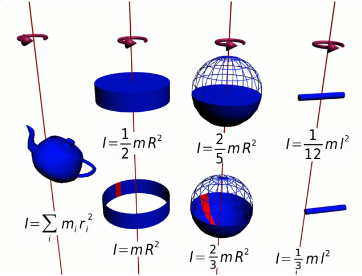
Bodies

Rigid body: 6 degrees of freedom (times 2 due to the time derivatives)

Mechanical properties

- Mass
- Moments of inertia

figure: https://en.wikipedia.org/wiki/Angular_momentum



NuCOS
Numeric Cooperative Simulation

Building blocks of a mechanical system

Joints

Joints reduce the degrees of freedom (every body has exactly one joint)

Types

- Cardanic
- Axis
- Revolute
- ...

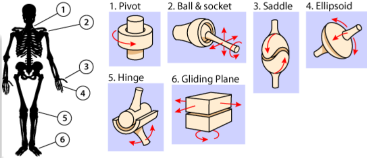


figure: https://askabiologist.asu.edu/.../joints540.gif

NuCOS
Numeric Cooperative Simulation

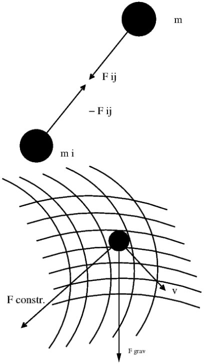
Building blocks of a mechanical system

Forces and torques

Forces/torques accelerate the masses

Types

- Pairwise forces
- External forces
- Constraint forces



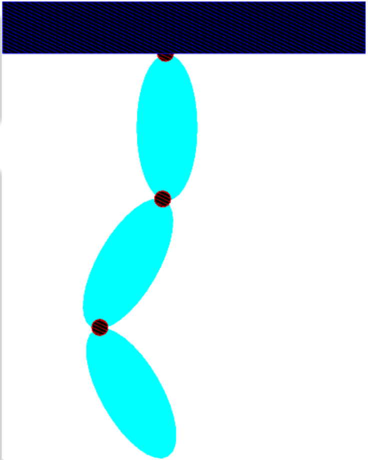
Building blocks of a mechanical system

Generalized coordinates

Generalized coordinates (if minimal) fullfill constraints automatically

Hints

- Generalized coordinates are not unique
- If the number exceeds the degrees of freedom one has to include constraint forces
- It is always possible to transform it into the 6 position/angle coordinates of a body



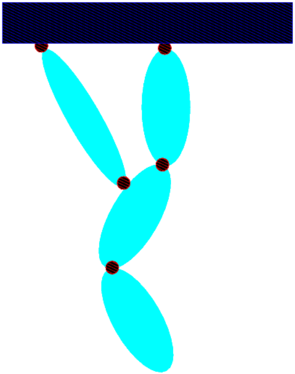
Building blocks of a mechanical system

Constraint loops

Contraint loops are always contraints which can not be formulated via joints

Hints

- There are several propositions to solve the resulting equations (DAE-Methods)
- We propose a solution according to Lagrange 1 with an additional drawback force (due to numerics)
- For linearization it can always be included into the ode-system with minimal coordinate number



Methods to generate the equations of motion

- Kane's Method (d'Alembert's Principle of virtual work)
- Lagrange's Method
- Hamilton's Equations

NuCOS
Numeric Cooperative Simulation

Outline

Layout

- Object of type MBSworld: general world setup
- Methods are provided to add bodies, markers, external forces, extra constraints, reflective walls
- An ODE solver is connected
- 3d graphical backend is connected
- Some physical quantities are provided: e.g. energy, forces, velocity

NuCOS
Numeric Cooperative Simulation

What's new and interesting

- Completeness of joints and tools (special joints can be produced by the user)
- Jacobian for linear stability analysis
- Linearization tool completed: dep. and indep. coordinates can be detected automatically (incl. the transformation into a complete set) and some pitfalls are removed
- A numerical highly stable method for constraint loops is provided
- External models and parameters can be included
- An object for characteristic curves (b-splines) for interaction forces is provided

NuCOS
Numeric Cooperative Simulation

Coding style

Setup the system

```
import mbs
myMBS = mbs.MBSworld()
myMBS.add_body_3d(bodyname, markername,mass,Inertia,joint-type ...)
myMBS.add_marker(markername, bodyname, X, Y, Z, theta, phi, xi)
myMBS.add_force(...)
myMBS.add_one_body_force_model(...)
myMBS.add_geometric_constaint(...)
```

Assembling and solving

```
myMBS.set_const_dict(...)
myMBS.kaneify()
myMBS.inte_grate_full(...)
```

Postprocessing

```
myMBS.calc_lin_analysis_n(...)
myMBS.prepare()
myMBS.animate(...)
```

Solving-problems-hints for developers:

Numerical calculation

Switch to numpy for numerics, do not try to use sympy to solve for Eigenvectors

Use lambdify for numbers (not subs)

Most usefull sympy-function to speed up: put in an algebraic expression, get out a python function

ODE solvers

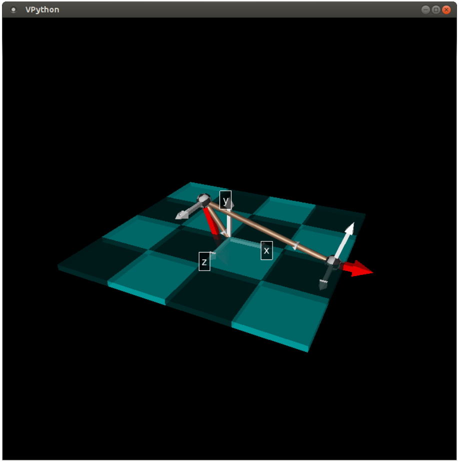
Never use your own, even if it looks like fun to programm one (we use LSODA-Solver), this is also a call for sundials

NuCOS
Numeric Cooperative Simulation

Outline

Example 1: Linear-rotation converter (crank-slider)

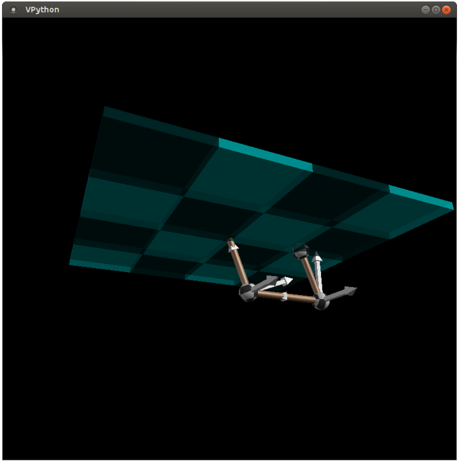
A constraint loop of the most easiest way



NuCOS
Numeric Cooperative Simulation

Example 2: Swing table

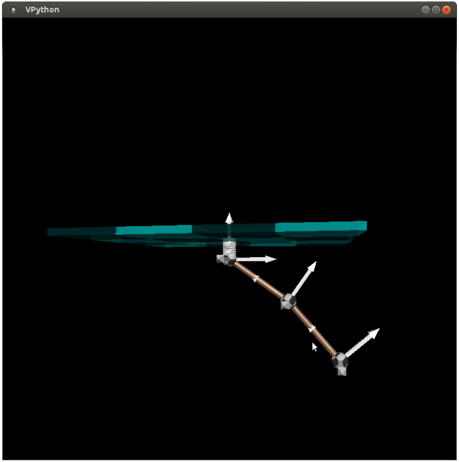
A second constraint loop example: linearization



NuCOS
Numeric Cooperative Simulation

Example 3: Reflective wall

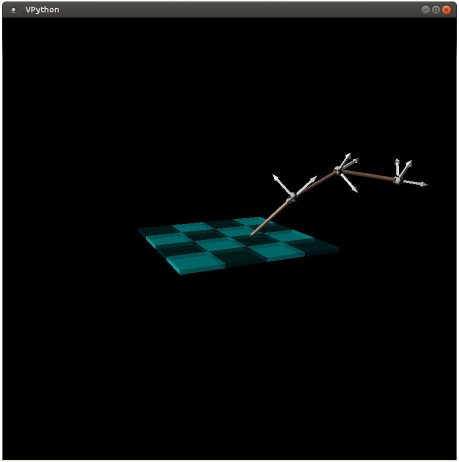
Nontrivial constraint handling



NuCOS
Numeric Cooperative Simulation

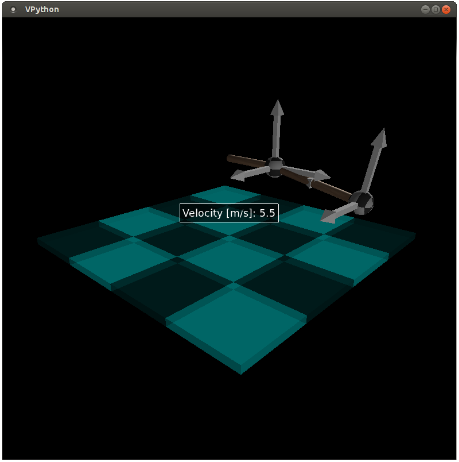
Example 4: Angle pendulum

Rotational degrees of freedom in a chain



Example 5: Rotating constraint

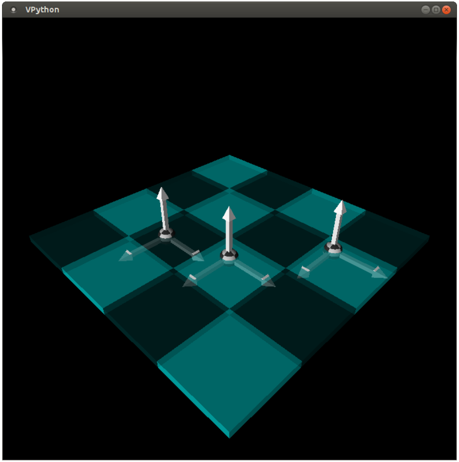
A rotating frame



NuCOS
Numeric Cooperative Simulation

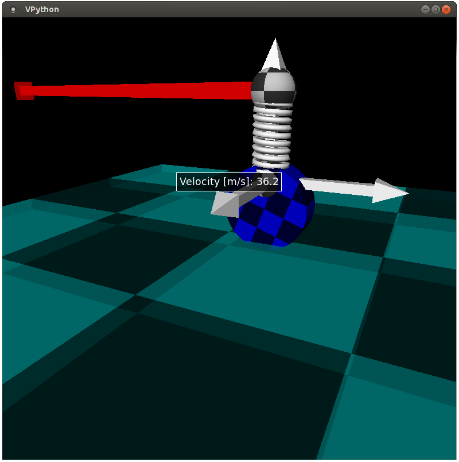
Example 6: Gravitation

An example for spline-use



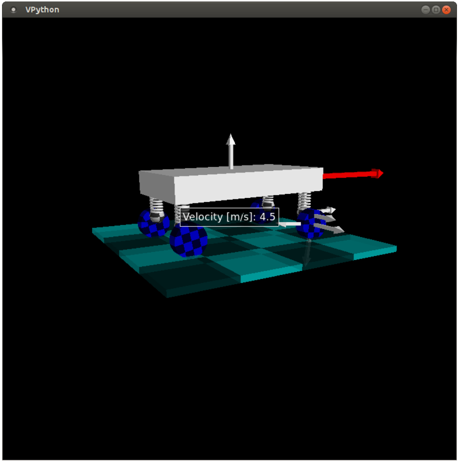
Example 7: Quarter car

A primitive vertical dynamic model



Example 8: Simple car model

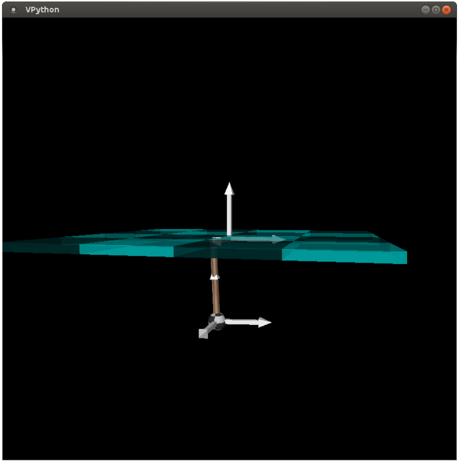
An educational car model with tire model



NuCOS
Numeric Cooperative Simulation

Example 9: Moving pendulum

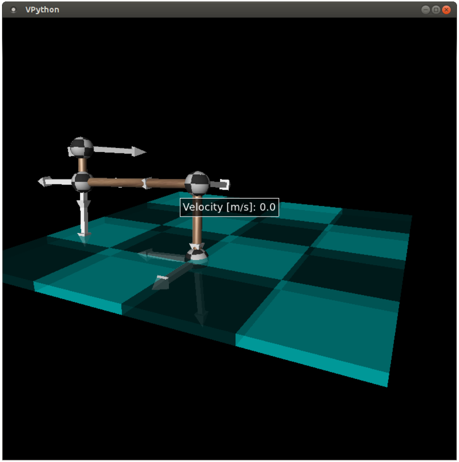
A moving frame due to time dependent parameter



NuCOS
Numeric Cooperative Simulation

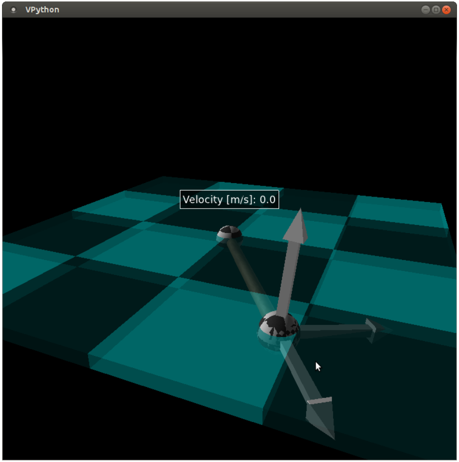
Example 10: Bending and rotation stiffness

To supply more joint functionality



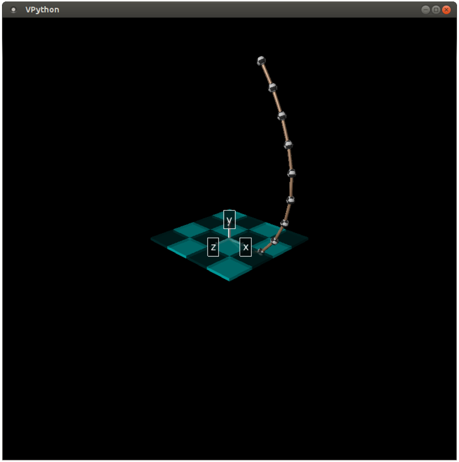
Example 11: Gyroscope

Validation of gyroscope movement (precession-nutation)



Example 12: Linear chain of masses

Check the calc-time dependency on number of bodies



NuCOS
Numeric Cooperative Simulation

Outline

Outline

Improvements

- ZODB - make assembled equations persistent
- Graphics - always some improvements to be done
- Model validation - automatic topology diagram
- Use of just-in-time compilation to speed up
- Postprocessing with pandas
- Implementing some model interface standards
- ...

NuCOS
Numeric Cooperative Simulation

Outline

Timeline

- Complete the basics until Sep. 2015
- Full vehicle simulation on Oct.-Dec. 2015
- ...

NuCOS
Numeric Cooperative Simulation

Thank you for your attention !!!

Outline

Outline

According to Lagrange 1

Equation of motion without constraint force

$$m\vec{a}_i = \sum_j \vec{F}_{ij} + \sum_k \vec{F}_{ext} \tag{3}$$

Equation of motion with geometric constraint f_{const}

$$\begin{aligned}
m\vec{a}_{i\|} &= \sum_j \vec{F}_{ij} + \sum_k \vec{F}_{ext} + \lambda\nabla(f_{const}) && (4)\\
m\vec{a}_{i\|} &= \sum_j \vec{F}_{ij} + \sum_k \vec{F}_{ext} - (\sum_k \vec{F}_{ext}\cdot\vec{n})\vec{n} - C_\infty m\ddot{\delta}\vec{n} + \vec{F}_{stable} && (5)\\
\vec{n} &= \nabla(f_{const}) && (6)\\
\vec{F}_{stable} &= (-C\delta - \gamma\dot{\delta})\vec{n} && (7)
\end{aligned}$$

The Jacobian

System of MBS-ODE

$$
\begin{aligned}
\mathbf{M}\dot{\vec{x}} &= \vec{F} && (8)\\
\dot{\vec{x}} &= \mathbf{M}^{-1}\vec{F} = \vec{f}(\vec{x}, t) && (9)
\end{aligned}
$$

Role of the Jacobian

$$
\begin{aligned}
\dot{x}_i &= f_i(\vec{x}_0, t_0) + \frac{\partial f_i}{\partial x_j}(x_j - x_{j0}) && (10)\\
\mathbf{J}_{ij} &= \frac{\partial f_i}{\partial x_j} && (11)
\end{aligned}
$$

NuCOS
Numeric Cooperative Simulation

Use of B-splines

B-spline (Basis: Φ_k) approximation of a set of points (x_i/y_i)

$$y_i = \sum_k c_k \Phi_k(x_i) \quad (12)$$

$$\mathbf{N}_{ki} = \Phi_k(x_i) \quad (13)$$

Gaussian optimization of coefficients (since the above equation (12) has no solution)

$$\mathbf{c} = (\mathbf{N}^T\mathbf{N})^{-1}\mathbf{N}^T\mathbf{y} \quad (14)$$

NuCOS
Numeric Cooperative Simulation