Towards an Object-oriented Implementation of von Mises’
Motor Calculus Using Modelica

2nd International Workshop on Equation-based Object-oriented
Languages and Tools,
Paphos, Cyprus, July 8, 2008

Tobias Zaiczek, Olaf Enge-Rosenblatt

Fraunhofer-Institut Integrierte Schaltungen
Branch Lab Design Automation,
Dresden, Germany
Object-oriented Implementation of von Mises’ Motor Calculus

Contents

1. Introduction
2. Motor calculus
3. Aspects of implementation
4. Examples
5. Summary/Outlook
Object-oriented Implementation of von Mises’ Motor Calculus

Contents

1. Introduction
2. Motor calculus
3. Aspects of implementation
4. Examples
5. Summary/Outlook
Object-oriented Implementation of von Mises’ Motor Calculus

1. Introduction

Current situation
- description of the behaviour of multi-body systems is not an easy task
- Modelica Multibody Standard Library is a well-designed tool
- equations of motions are hard to read and understand

Idea
- usage of motor calculus proposed by Richard von Mises in 1924
- make equations easier to understand

What did we do?
- first phase: implementation of motor calculus by extending Modelica Multibody Standard Library
- approach corresponds with the object-oriented paradigm
- not equation-based to its full sense because of missing operator overloading possibilities
Contents

1. Introduction
2. Motor calculus
   2.1 Fundamentals
   2.2 Geometrical interpretation
   2.3 Application to rigid bodies
3. Aspects of implementation
4. Examples
5. Summary/Outlook
2. Motor calculus

2.1 Fundamentals

A motor

\[ \mathbf{h} = \begin{pmatrix} \mathbf{g} \\ \mathbf{h}_o \end{pmatrix} \]

can be represented by an ordered pair of vectors \( \mathbf{g} \) and \( \mathbf{h}_o \) defining a vector field in the three-dimensional space:

\[ \mathbf{h}(\mathbf{r}) = \mathbf{h}_o + \mathbf{g} \times \mathbf{r} \]

\( \mathbf{h}_o \): moment vector at the reference point \( O \)

\( \mathbf{g} \): resultant vector

\( \mathbf{r} \): position vector for (any) point \( P \)

\( \mathbf{h} \): moment vector for point \( P \)
Object-oriented Implementation of von Mises’ Motor Calculus

2. Motor calculus

Fundamental algebraic definitions

addition: 
\[ h_1 + h_2 = \begin{pmatrix} g_1 + g_2 \\ h_{o1} + h_{o2} \end{pmatrix} \]

multiplication with a scalar: 
\[ \alpha h = \begin{pmatrix} \alpha g \\ \alpha h_o \end{pmatrix} \quad \alpha \in \mathbb{R} \]

dot or inner product: 
\[ (h_1, h_2) = (g_1, h_{o2}) + (g_2, h_{o1}) \]

cross or outer product: 
\[ h_1 \times h_2 = \begin{pmatrix} g_1 \times g_2 \\ g_1 \times h_{o2} + h_{o1} \times g_2 \end{pmatrix} \]

multiplication with a dyad \( \mathcal{D} \): 
\[ \mathcal{D} \circ h_1 = \begin{pmatrix} D_{11} h_{o1} + D_{12} g_1 \\ D_{21} h_{o1} + D_{22} g_1 \end{pmatrix} \]
2.2 Geometrical interpretation of motors

- can be represented geometrically by an ordered pair of straight lines $(\mathcal{G}_1, \mathcal{G}_2)$
- all mathematical operations interpretable as geometrical constructions
- $\mathcal{N}$… motor axis = common normal of $\mathcal{G}_1$ and $\mathcal{G}_2$
- $h_n$… moment of the motor on the motor axis, connects $\mathcal{G}_1$ and $\mathcal{G}_2$ along $\mathcal{N}$
- $g$ represents the rotation of $\mathcal{G}_1$ when transferred into $\mathcal{G}_2$
- mapping $(\mathcal{G}_1, \mathcal{G}_2) \mapsto h$ is not a one-to-one mapping (motor $h$ is invariant w. r. t. translations and rotations of $\mathcal{G}_1$ and $\mathcal{G}_2$ across $\mathcal{N}$)
2.3 Application to rigid bodies

Force motor, velocity motor, momentum motor und inertia dyad

\[ \mathbf{f} = \begin{pmatrix} f \\ d_o \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} \omega \\ v_o \end{pmatrix}, \quad \mathbf{p} = \begin{pmatrix} p \\ I_o \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} mI & -mR_s \\ mR_s & \Theta_o \end{pmatrix} \]

Basic relations

- **momentum:** \[ \mathbf{p} = \mathbf{M} \circ \mathbf{v} \]
- **kinetic energy:** \[ T = \frac{1}{2} (\mathbf{v}, \mathbf{p}) = \frac{1}{2} (\mathbf{v}, \mathbf{M} \circ \mathbf{v}) \]
- **power:** \[ P = (\mathbf{f}, \mathbf{v}) \]

Equations of motion

- **inertial:** \[ \dot{\mathbf{p}} = \mathbf{f} \]
- **body-fixed:** \[ \ddot{\mathbf{p}} + \mathbf{v} \times \mathbf{p} = \mathbf{f} \]
Object-oriented Implementation of von Mises’ Motor Calculus

Goals of the Motor Calculus

• description of
  • rigid body movement
  • forces and torques acting on a rigid body
  • momentum and angular momentum
  each by a six-dimensional “vector”

• description independent of reference frame and chosen reference point
  (geometrical interpretation)

• very clear and simple structure of the fundamental mechanical laws

• formal equivalence to Newton’s Second Law
Object-oriented Implementation of von Mises’ Motor Calculus

Contents

1. Introduction
2. Motor calculus
3. Aspects of implementation
   3.1 Implementation of a motor library
   3.2 Modification of the MultiBody Library
4. Examples
5. Summary/Outlook
Object-oriented Implementation of von Mises’ Motor Calculus

3. Aspects of implementation

3.1 Implementation of a motor library

Objective

- taking advantage of the efficient description in terms of motor calculus in Modelica
- object-oriented implementation of all operations in the class Motor
- specialisation by means of inheritance and polymorphism

Issues in Modelica

- no overloading of operators or functions
- no attachment of functions to classes

=> Compromise
3.2 Modification of the MultiBody Library

Modification of the class Body

- object-oriented implementation of the equations of motion using the motor calculus

\[ f_a = \text{der}(p) + 'x'(v,p) - f_g; \]

Subclasses BodyBox, BodyShape, and BodyCylinder
Contents

1. Introduction
2. Motor calculus
3. Aspects of implementation
4. Examples
   4.1 Damped moveable double pendulum
   4.2 Damped fourfold pendulum on two movable sliders
5. Summary/Outlook
4. Examples

4.1 Damped moveable double pendulum

- Three rigid bodies moving in the Earth’s gravitational field
- trolley: mass $M_0$, viscose friction ($\rho_0$)
- 1st pendulum: mass $M_1$, moment of inertia $J_1$, viscose friction ($\rho_1$)
- 2nd pendulum: mass $M_2$, moment of inertia $J_2$, viscose friction ($\rho_2$)
Object-oriented Implementation of von Mises’ Motor Calculus

Animation of simulation results
Comparison of simulation results

Errors between both simulation results are sufficiently small and decay for increasing values of the time $t$. 
Object-oriented Implementation of von Mises’ Motor Calculus

4.2 Damped fourfold pendulum on two movable sliders

- Six rigid bodies moving in the Earth’s gravitational field
- Trolleys: masses $M_0, M_5$ viscose friction ($\rho_0$ and $\rho_5$)
- $i^{\text{th}}$ pendulum: mass $M_i$, moment of inertia $J_i$ viscose friction ($\rho_i$), $i = 1, \ldots, 4$

➢ closed planar kinematic loop
Object-oriented Implementation of von Mises’ Motor Calculus

Animation of simulation results
Comparison of simulation results

Errors between both simulation results are sufficiently small and decay for increasing values of the time $t$. 

---

Olaf Enge-Rosenblatt, Workshop EOOLT 2008, Paphos, Cyprus, July 8, 2008

© Fraunhofer-Gesellschaft 2008

Fraunhofer Institut Integrierte Schaltungen
Contents

1. Introduction
2. Motor calculus
3. Aspects of implementation
4. Examples
5. Summary/Outlook
Object-oriented Implementation of von Mises’ Motor Calculus

5. Summary/Outlook

presented:

• short introduction to von Mises’ motor calculus
• implementation of Modelica library for motor calculus
• first simple implementation of the motor calculus within the MultiBody Standard Library
• simulation results for different non-trivial mechanical problems

future tasks:

• more sophisticated MultiBody implementation
• numerical analysis in terms of effectiveness and accuracy
Thank You!