Recursive formulations for modeling multi-rigid-body system dynamics

Krzysztof Chadaj
Marcin Pękal

Scientific advisor:
Professor Janusz Frączek

Co-advisors:
Assistant Prof. Paweł Malczyk
Associate Prof. Marek Wojtyra

Division of Theory of Mechanics and Robots
The Institute of Aeronautics and Applied Mechanics
The Faculty of Power and Aeronautical Engineering
Warsaw University of Technology
1. Why this topic?
   - Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

   - Idea, algorithm overview, constrained systems

3. Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
   - Algorithm’s properties and examples, simple dot product example, DCA for MBS

4. Index 2 differential algebraic equations properties
   - Direct integration of Hamilton’s index 2 DAEs

5. Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
   - Idea, algorithm overview, results
1 Why this topic?
- Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

2 Recursive algorithm based on Newton–Euler equations (Bae & Haug (1987), Featherstone (1983))
- Idea, algorithm overview, constrained systems

3 Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
- Algorithm’s properties and examples, simple dot product example, DCA for MBS

4 Index 2 differential algebraic equations properties
- Direct integration of Hamilton’s index 2 DAEs

5 Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
- Idea, algorithm overview, results
Examples of MB applications

NADS vehicle simulator

Liebherr’s TI 272 truck

Examples of MB applications

Monster Inc. movie

Half-Life 2 game

Direct integration of index 3 DAEs

Standard approach (absolute coordinates)

\[
\begin{bmatrix}
M & \Phi_q^T \\
\Phi_q & 0
\end{bmatrix}
\begin{bmatrix}
\ddot{q} \\
\lambda
\end{bmatrix}
= \begin{bmatrix}
Q \\
c
\end{bmatrix}
\]

- Solution \((q\) and \(\dot{q}\)) quickly diverges
- Computational cost \(O(n^3)\)
Error sources

- Integrating routines error
- Initial conditions not exactly met
- Equations stiffness
- Constraint derivatives error
Constraint stabilization methods

Baumgarte’s stabilization

$$\Gamma^* = \Gamma - 2\alpha \dot{\Phi} - \beta^2 \Phi$$

- Smaller time steps
- Greater impact of high frequencies in solution

Augmented Lagrangian method

$$\lambda = \lambda^* + \alpha (\dddot{\Phi} + 2\Omega \mu \dot{\Phi} + \Omega^2 \Phi),$$

$$M\ddot{q} + \Phi_q^T \lambda = Q$$

- Iterative method
- Large penalty factors increase the stiffness of the equations

Independent coordinates selection

$$\dot{q} = R\dot{z}$$

- Integration of independent coordinates only
- Additional computational cost $O(n^3)$
Table of contents

1 Why this topic?
   - Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

2 Recursive algorithm based on Newton–Euler equations (Bae & Haug (1987), Featherstone (1983))
   - Idea, algorithm overview, constrained systems

3 Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
   - Algorithm’s properties and examples, simple dot product example, DCA for MBS

4 Index 2 differential algebraic equations properties
   - Direct integration of Hamilton’s index 2 DAEs

5 Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
   - Idea, algorithm overview, results
Idea
- Solve Newton–Euler equations recursively
- Use joint coordinates $q_{ij}$
- Obtain computational cost $O(n)$
Recursive algorithm based on Newton–Euler equations (Bae, Haug)

Algorithm steps

1. Forward kinematics loop (from base body)

\[ Y_j = B_{ij1} Y_i + B_{ij2} \dot{q}_{ij} \]

2. Backward dynamics loop (to base body)

\[ \delta Z_j^T \left( (M_j + K_j^\beta) \dot{Y}_j - (Q_j + L_j^\beta) \right) = 0 \]

3. Forward acceleration loop (from base body)

\[ \ddot{q}_{ij} = - (B_{ij2}^T (M_j + K_j) B_{ij2})^{-1} \cdot \left[ B_{ij2}^T (M_j + K_j) B_{ij1} \dot{Y}_i + B_{ij2}^T ((M_j + K_j) D_{ij} - (Q_j + L_j)) \right] \]
Recursive algorithm based on Newton–Euler equations (Bae, Haug)

### Lagrange multipliers method for constrained systems

- **Equations of motion**

\[
\sum_{i=1}^{n} (\delta Z_i^T \left( M_i \dot{Y}_i - Q_i \right) + \sum_{(j,k) \in Z_n} \delta Z_i^T \Phi_{Z_n}^{(j,k)T} \lambda_{jk} ) = 0
\]

- **Constraint equations**

\[
\Phi M_i \dot{Y}_i + \Phi L_i + \lambda - \text{RHS}_i - \gamma = 0
\]
Recursive algorithm based on Newton–Euler equations (Bae, Haug)

(Tricky) numerical example #1
Recursive algorithm based on Newton–Euler equations (Bae, Haug)

(Tricky) numerical example #1

MSC Adams solution of EOM

Recursive solution of Newton-Euler EOM

$X_{peak}$
Numerical example #2
1 Why this topic?
   - Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

2 Recursive algorithm based on Newton–Euler equations (Bae & Haug (1987), Featherstone (1983))
   - Idea, algorithm overview, constrained systems

3 Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
   - Algorithm’s properties and examples, simple dot product example, DCA for MBS

4 Index 2 differential algebraic equations properties
   - Direct integration of Hamilton’s index 2 DAEs

5 Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
   - Idea, algorithm overview, results
Divide and conquer algorithm

- Popular programming paradigm
- Divides a task into smaller sub-problems
- Use of multi-branched recursion
- Inherent parallelism

Examples

- Large number multiplication
- Sorting
- Fibonacci numbers generators
- Fast Fourier transform
D&C algorithm based on Newton–Euler equations (Featherstone)

Example: dot product of N-length vectors

\[ \mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^{N} u_i v_i \]

D&C \([O(\log n)]\)

1. Each thread performs 1 multiplication
2. \(N/2\) threads perform 1 addition
3. \(N/4\) threads perform 1 addition
4. \(N/8\) threads perform 1 addition

\(\ldots\) as long as \([N/k^2 > 1]\)
D&C algorithm based on Newton–Euler equations (Featherstone)

D&C for Newton–Euler equations

Having equations of motion of bodies A and B:

\[ \mathbf{a}_1^A = \Phi_1^A \mathbf{f}_1^A + \Phi_{12}^A \mathbf{f}_2^A + \mathbf{b}_1^A \]
\[ \mathbf{a}_2^A = \Phi_2^A \mathbf{f}_1^A + \Phi_{21}^A \mathbf{f}_2^A + \mathbf{b}_2^A \]
\[ \mathbf{a}_1^B = \Phi_1^B \mathbf{f}_1^B + \Phi_{12}^B \mathbf{f}_2^B + \mathbf{b}_1^B \]
\[ \mathbf{a}_2^B = \Phi_2^B \mathbf{f}_1^B + \Phi_{21}^B \mathbf{f}_2^B + \mathbf{b}_2^B \]

construct equations of motion of superbody C = A + B:

\[ \mathbf{a}_1^C = \Phi_1^C \mathbf{f}_1^A + \Phi_{12}^C \mathbf{f}_2^B + \mathbf{b}_1^C \]
\[ \phi_1^C = \phi_1^A - \phi_{12}^A \mathbf{W} \phi_{21}^A \]
\[ \phi_2^C = \phi_1^B - \phi_{12}^B \mathbf{W} \phi_{21}^B \]

\[ \mathbf{a}_2^C = \Phi_2^C \mathbf{f}_1^A + \Phi_{21}^C \mathbf{f}_2^B + \mathbf{b}_2^C \]
Numerical example
D&C algorithm based on Newton–Euler equations (Featherstone)

Numerical example

Energy error

$5 \times 10^{-7}$

$0 \quad 10 \quad 20 \quad 30 \quad 40 \quad 50 \quad 60 \quad 70 \quad 80 \quad 90 \quad 100$

$0 \quad 1 \quad 2 \quad 3 \quad 4$

norm($\Phi$)

$4 \times 10^{-12}$

$0 \quad 10 \quad 20 \quad 30 \quad 40 \quad 50 \quad 60 \quad 70 \quad 80 \quad 90 \quad 100$

$0 \quad 1 \quad 2 \quad 3$
D&C algorithm based on Newton–Euler equations (Featherstone)

Numerical example

![Graphs showing norm(dΦ/dt) and norm(d²Φ/dt²)]
Table of contents

1. Why this topic?
   - Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

   - Idea, algorithm overview, constrained systems

3. Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
   - Algorithm’s properties and examples, simple dot product example, DCA for MBS

4. Index 2 differential algebraic equations properties
   - Direct integration of Hamilton’s index 2 DAEs

5. Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
   - Idea, algorithm overview, results
Index 2 differential algebraic equations

Index 2 DAEs

- first-order equation
- constraint equations imposed on velocity level
- less intuitive
- more accurate
Direct integration of Hamilton’s index 2 DAEs

It is needed to:

- determine total energy of the system
- compute partial derivatives

Hamilton’s equations for unconstrained MBS

\[
\begin{align*}
\dot{q} &= + \frac{\delta H}{\delta p} \\
\dot{p} &= - \frac{\delta H}{\delta q} + Q
\end{align*}
\]

Hamilton’s equations for constrained MBS

\[
\begin{align*}
\dot{q} &= + \frac{\delta H}{\delta p} \\
\dot{p} &= - \frac{\delta H}{\delta q} + Q - \Phi_q^T \lambda
\end{align*}
\]
1. Why this topic?
   - Examples of MB applications, direct integration of index 3 DAEs, error sources, constraint stabilization methods

   - Idea, algorithm overview, constrained systems

3. Divide and conquer algorithm based on Newton–Euler equations (Featherstone (1999))
   - Algorithm’s properties and examples, simple dot product example, DCA for MBS

4. Index 2 differential algebraic equations properties
   - Direct integration of Hamilton’s index 2 DAEs

5. Recursive Hamiltonian approach to N–E eq. (Naudet (2005))
   - Idea, algorithm overview, results
Recursive Hamiltonian approach to N–E equations (Naudet)

Idea

- formulate the equations of motion in body-fixed coordinate frame
- use canonical momenta to write Newton–Euler EOM
  \[ \dot{P}^{(k)} + \tilde{\Omega}P = T \]
- solve equations recursively
Algorithm steps

1. Backward (to base body)
   - articulated and reduced mass matrix
   - reduced and remaining momentum vector

2. Forward (from base body)
   - coordinate and spatial velocity
   - transformation matrices

3. Backward (to base body)
   - accumulated force vector
   - time derivatives of canonical momenta
Numerical example

Recursive Hamiltonian approach to N–E equations (Naudet)
Results

- index 2 DAEs
- no need to compute energy of the system
- no need to compute partial derivatives $\frac{\delta H}{\delta q}$ and $\frac{\delta H}{\delta p}$
- three recursion loops
- computational cost $O(n)$
- easy extension for handling impacts
- augmented canonical momenta for constrained systems (ongoing):
  \[ p_i^+ = p_i + T^F E_c \sigma \]
Summary

- Recursive algorithms solving EOM are more efficient.
- It is possible to solve constraint systems recursively.
- The use of joint coordinates minimizes the number of system’s variables.
- No constrains for serial mechanisms if joint coordinates are used.
- It is possible to use recursion approach for the design of parallel algorithms.
- Index 2 DAEs are more accurate than index 3 DAEs (acceleration formulations).
- It is possible to solve index 2 DAEs recursively.
This work has been supported by the European Union in the framework of European Social Fund through the Didactic Development Program of the Faculty of Power and Aeronautical Engineering of the Warsaw University of Technology.
Any questions?