TR – 2007-07

A MATLAB implementation of the seven-body mechanism for implicit integration of the constrained equations of motion

Naresh Khude, Dan Negrut

December 20, 2007

1. Introduction

Real-life applications do not have smooth motion due to the presence of phenomena such as impact/contact, friction, etc. Real-life models can rarely use integration formulas of orders beyond 2 and do so at large integration step-sizes. To simulate the motion of such models several low order numerical integrators have been developed. Authors have thoroughly discussed six of the low order numerical integrators in [1]. These numerical formulas are divided into two categories based on their formulation approach. The first three numerical methods discussed in [1], namely Newmark, HHT-I3 and NSTIFF are obtained from a direct index-3 discretization approach. These methods satisfy the constraints at the position level. The other three methods, i.e. HHT-ADD, HHT-SI2 and NSTIFF-SI2 are based on the stabilized overdetermined index-2 DAE approach. These methods satisfy the constraints at the position as well as the velocity level. In this report, these six numerical integrators are used to determine the time evolution of a seven body mechanism.

2. Seven Body Mechanism

The low order numerical formulas are implemented on simple mechanical models such as simple and double pendulums, slider crank, flexible slider crank, etc. in [1]. It is interesting to study how these formulas behave with more complex models. For this purpose, this work considers the seven body mechanism, popularly known as "Andrews' squeezer mechanism", as defined in [2] and shown in Figure 1.

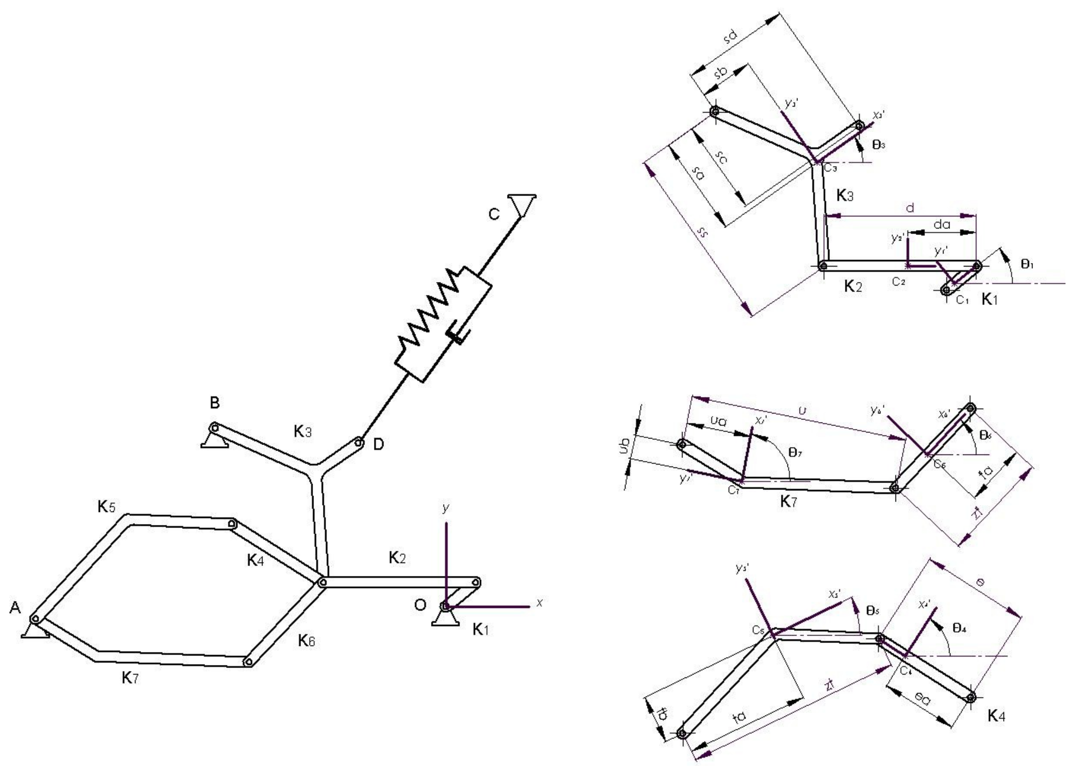

Figure 1: Seven body mechanism

The mechanism consists of seven rigid bodies connected by joints without friction in plane motion. The origin is at point O and it is supposed that the mechanism is driven by a motor, located at O, whose constant drive torque is given by mom=0.033. Throughout this document the units used are SI units. The coordinates of the other fixed points A, B and C are given in Table 1.

Table 1: Coordinates of fixed points

xa = -0.06934 ya = -0.00227	xb = -0.03635 yb = 0.0327	xc = 0.014 yc = 0.072

The spring connecting the point D and C has the spring coefficient C_0 and the unstretched length l_0. The geometrical parameters of the model are given in Table 2 and the mass properties of the seven bodies are provided in Table 3.

Table 2: Geometrical parameters

d = 0.028	ra = 0.00092	ta = 0.02308
da = 0.0115	ss = 0.035	tb = 0.00916
e = 0.02	sa = 0.01874	u = 0.04
ea = 0.01421	sb = 0.01043	ua = 0.01228
zf = 0.02	sc = 0.018	ub = 0.00449
fa = 0.01421	sd = 0.02	C_0 = 4530
rr = 0.007	zt = 0.04	l_0 = 0.07785

Table 3: Mass properties of the 7 bodies

Body No.	Mass	Inertia
1	0.04325	2.194e-6
2	0.00365	4.41e-7
3	0.02373	5.255e-6
4	0.00706	5.667e-7
5	0.07050	1.169e-5
6	0.00706	5.667e-7
7	0.05498	1.912e-5

3. MATLAB Code[1]

The six integrators are implemented in MATLAB to simulate the motion of the seven body mechanism. There are driver files for each integrator and MATLAB function files which are either integrator or model specific. The details of these files are given below.

a. Driver Files:

File name	Description
driverNEWMARK	Driver file for NEWMARK Method
driverHHTI3	Driver file for HHT-I3 Method
driverNSTIFF	Driver file for NSTIFF Method
driverHHTADD	Driver file for HHT-ADD Method
driverHHTSI2	Driver file for HHT-SI2 Method
driverNSTIFFSI2	Driver file for NSTIFF-SI2Method

b. MATLAB Function Files:

File name	Description
appForce	Computes the generalized force acting on the system
compute_AccLam	Given an initial position and velocity configuration, compute the associated acceleration and Lagrange multipliers
computeB	Computes the matrix B in the J = LDB product
computeJ	Computes the Jacobian associated with the formulation
computeL	Computes the left block of the Jacobian, the L matrix in
constr_derivQQ	Computes the second order derivative of the constraints wrt the level 0 (zero) generalized coordinates
constr_derivQT	Computes the second order derivative of the constraints wrt the level 0 (zero) generalized coordinates and time
constr_derivTT	Computes the second time derivative of the kinematic position constraint equations
constrF_derivQ	Computes the derivative of the constraint force wrt the level 0 (zero) generalized coordinates
Re_derivQ	Same as constrF_derivQ, used for some of the integrators
F_derivQ	Computes the derivative of the generalized force w.r.t. the generalized positions
F_derivV	Computes the derivative of the generalized force w.r.t the generalized velocities
invMass	Returns the inverse of the generalized mass matrix
Mass_derivQ	Computes the partial derivative of the inertia term (Mass(q)*acc)
matMass	Returns the mass matrix associated with the model
phi	Computes the kinematic position constraints
phi_derivQ	Computes the constraint Jacobian
phi_derivT	Computes the time derivative of the kinematic position constraints
read_ICs	Reads initial positions and velocities of the model
read_model	Loads in modelI and modelD the integer and double parameters associated with the model
read_simParams	Sets up simulation control parameters
read_simParams_new	Same as read_simParams, used for NEWMARK integrator

[1] Note that the MATLAB code associated with any of the six integration formulas as well as the seven body mechanism specific methods is freely available. Please contact negrut@wisc.edu in case you are interested in getting copies of the code.

rhsAccKinEq	Computes the RHS of the acceleration kinematic constraint equation
startupConfiguration	Computes the acceleration and Lagrange multipliers at the starting point
startupConfiguration_JN	Compute the starting point as far as aZero and psiZero are concerned, used for HHT-ADD integrator

4. **Numerical Experiments and Conclusions**

The numerical experiments are carried out to compare the integrators in terms of several metrics such as the convergence order, energy dissipation, velocity kinematic constraint drift, and efficiency [1]. Out of these metrics, the convergence order and the efficiency of these integrators based on seven body mechanism are presented here.

To determine the convergence order, a 2^{-6} seconds long simulation of the seven body mechanism is carried out using step sizes of 2^{-12}, 2^{-13}, …, 2^{-17} seconds. The angular position and the velocity of the link 5 are noted at the end of the simulation. The reference solution is obtained using very small step size $h = 2^{-20}$. The log-log plots of the errors in angular position and velocity of link 5 are shown in Figures 2 and 3.

It is observed that except NEWMARK all other integrators show convergence order two. NEWMARK shows convergence order one. From Figs. 2 and 3, it is observed that the convergence orders hold both for the generalized coordinates and their time derivative, that is, both for positions and velocities.

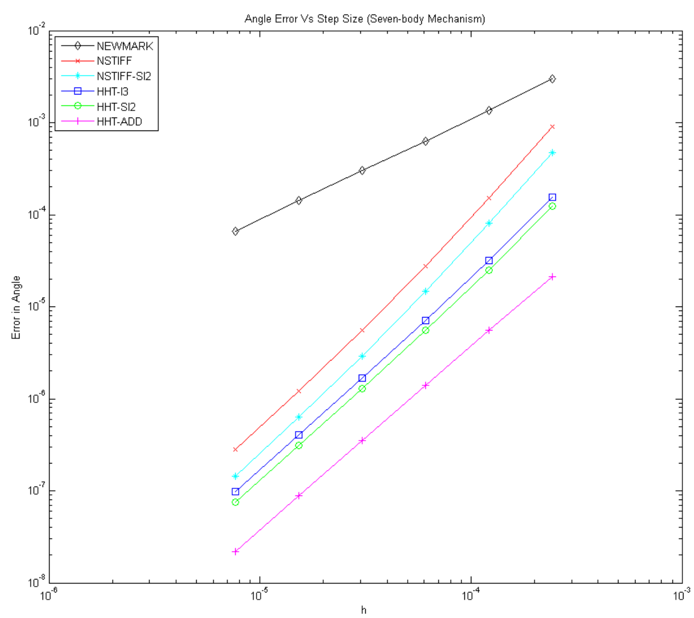

Figure 2: Angle error convergence order

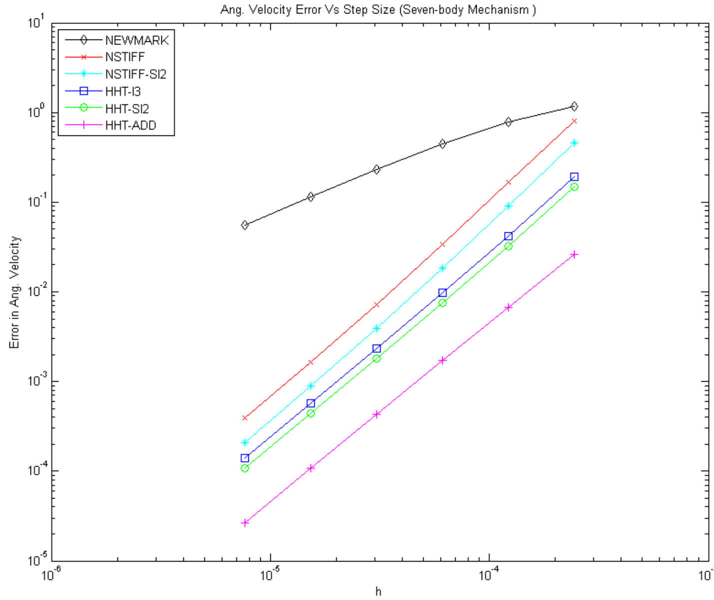

Figure 3: Angular velocity error convergence order

The integrators' performance is also compared in relation to their efficiency. For this purpose, the CPU time of each integrator for a 2^{-5} sec long simulation is measured. The step size, $h=2^{-15}$ is used to carry out these simulations. The CPU times observed are given in Table 4. It is more intuitive to compare the normalized CPU times as shown in Figure 4. The normalization is done with respect to the CPU time of HHT-I3 integrator.

Table 4: CPU time for 7 body mechanism

Integration Formula	CPU time (sec)
NEWMARK	3.284367
HHT-I3	3.639707
NSTIFF	3.387995
HHT-ADD	11.54743
HHT-SI2	6.127796
NSTIFF-SI2	7.109599

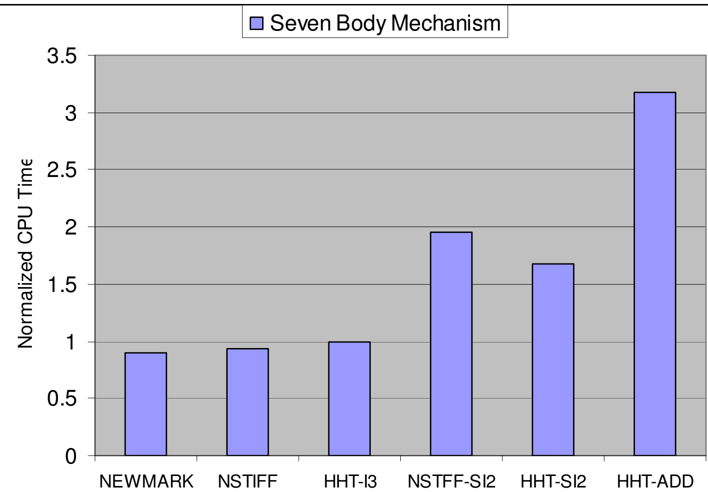

Figure 4: Normalized CPU Time

It is observed that stabilized index 2 integrators are relatively slow as compared to the index 3 integrators. HHT-ADD is the least efficient while NEWMARK being the most efficient for this particular model. Among the stabilized index 2 integrators HHT-SI2 seems to be the most efficient for the seven body mechanism.

References:

[1] Negrut, D., Jay, L., Khude, N., and Heyn, T., 2007. "A discussion of low-order integration formulas for rigid and flexible multibody dynamics". In Proceeding of Multibody Dynamics ECCOMAS Thematic Conference.

[2] Hairer, E., and Wanner, G., 1996. Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer