Java Astrodynamics Toolkit

Dave Gaylor, Emergent Space Technologies
Tobias Berthold, University of Texas
Noriko Takada, Perry Johnson Registrars

May 21, 2004

Background

- Open source software component library
- Started by University of Texas graduate students
- Provides a set of Java tools for:
  - Six degree of freedom (6DOF) spacecraft simulation
  - 2D and 3D visualization
- Potential Applications
  - Autonomous spacecraft navigation, guidance, and control simulations
    - GPS, GPS/INS
    - Autonomous rendezvous
  - Mission design and analysis
    - Trajectory optimization
  - Operational orbit determination and flight dynamics product generation

SourceForge

- Hosted on SourceForge
  - http://jat.sourceforge.net
  - Version Control via CVS
    - Anyone with Internet access can download
    - Only authorized developers can make changes
  - Bug Tracking and Feature Request Systems
  - Discussion Forums
  - Task Management
  - Release and Patch Management
  - Website hosting
  - Documentation hosting
- Licensed under GNU General Public License
  - Free to run the code, distribute source code, modify it, release modifications

Why Java?

- Portability
  - Write once, run anywhere
- Object Oriented
  - Manage complexity
  - Enhance reuse
  - Facilitates collaborative software development
- Development Tools
  - Free Integrated Development Environments (IDE)
    - No expensive compilers or Matlab licenses needed
  - Extensive Java Libraries
  - Javadoc – automated API documentation
- Easy to Create Robust, Bug-free Code
  - Strict type checking
  - Automatic memory allocation, no pointers!

Why Not Java?

- Not as fast as C/C++
  - Java gets a bad rap because of Java GUIs and applets on the Internet
  - Numerical calculations almost as fast as C
    - Java runs faster than Matlab
  - Java facilitates distributed processing
  - Java development time is about half of C++
- Have to learn Java
  - Uses C syntax
  - No pointers
  - Automatic memory allocation/deallocation

Why Open Source?

- Improved software development
  - Open source programs can evolve rapidly
  - Built-in peer review
  - Funding can be pooled from various sources
  - Fixes and enhancements from outside
- Enhanced collaboration across organizations
  - Minimize unnecessary restrictions on transfer
  - Open source programs create a community
- More efficient and effective dissemination
  - NASA charter says:
    - "Provide for the widest dissemination of information concerning its activities and the results thereof"

Why Open Source?

- Better software support
  - Open source allows competition for support
    - Lower cost, more responsive
  - User can play a more active role
- Long-term Accessibility
  - Supported as long as interest and skills exist
  - Immune to retirements, budget cuts, bankruptcy, business decisions
- Easier to evaluate and understand
  - Users need to understand the details
  - Ability to look “under the hood”

Current Features

- Orbit Propagation
  - Fixed and variable step-size Runge-Kutta integrators
  - JGM3 Gravity Model
  - Harris-Priester and Exponential Atmospheric Density Models
  - Third-body gravity perturbations
  - Solar radiation pressure model
  - Finite burn and impulsive maneuvers
  - JPL DE405 Planetary Ephemeris
  - Keplerian Two Body propagator, Restricted Three Body
  - CW equations
- Coordinate and Time System Transformations
  - J2000 ECI, ECEF, RIC
  - GPS, UTC, TT, MJD

Current Features

- Extended Kalman Filter
  - Ground-based range measurements
  - GPS and GPS/INS Navigation
    - Absolute and Relative Navigation
- GPS Models
  - GPS Constellation based on Rinex Nav file
  - Receiver clock errors
  - Ionospheric delay, multipath
  - SV clock and ephemeris errors
  - Earth and object blockage, Elevation masks
  - Pseudorange and carrier phase measurements
    - Integer ambiguity

Current Features

- Maneuver Planning
  - Lambert Solver
  - Glideslope Rendezvous Algorithm
- Spacecraft Attitude Dynamics Simulation
  - Constant Torque
  - Gravity Gradient
  - Spherical Damper
  - Reaction Wheel
  - Control Moment Gyro
  - Bang-bang
- Nonlinear equation solvers
  - Regula falsi, Newton, Secant, Ridder

Current Features

- Vector and Matrix math
  - Quaternions, Rotation matrices
  - Inverse, determinant, etc.
  - Random Vectors
  - Cholesky, Householder, Eigenvalue , LU, Singular Value Decompositions
- 3D Graphics using Java3D and OpenGL
  - Movie file creation
- 2D Plotting
- Ground track plots
- Audio File Player

Architecture

User Interface Layer
XML, Swing, SWT

Applications Layer
GPS/INS Navigation Simulations, JAT Propagator, Attitude Dynamics Applet

Models Layer
Force Models, GPS and GPS/INS Measurement and Process Models

Algorithm Interfaces Layer
Derivatives, Measurement Model, Process Model, Vector Function

Algorithms Layer
Integrators, Estimators, Solvers, Optimizers, Maneuver Planners

Foundation Layer
Vector/Matrix Math, General Math Utilities, Printing, Plotting, Visualization and Multimedia

Next Steps

- NASA – Open Source Policy
  - How can NASA civil servants and contractors contribute software to open source projects?
  - Legal issues
    - NASA indemnity
    - Intellectual property rights
    - Licensing and contracts
  - Export control
- JAT Propagator User Interface (XML)
- Orbital Events
  - Eclipse, Apses, Equator Crossing, SAA

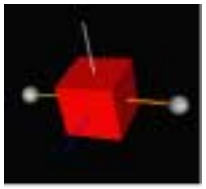
Demos, Q&A

- Attitude Applet
- 3D Visualization Movie

History of jat.attitude

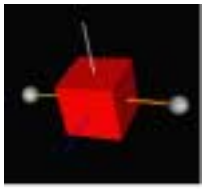
Noriko Takada

April, 2004



How I started…

- Need of Thesis Topics
- Interest in Attitude Dynamics
  - Continue with what was learned in classes
- Daniel Quock
  - Attitude Simulation Applet from ground-up
  - Left tasks…
    - On-line debut
    - More Scenarios

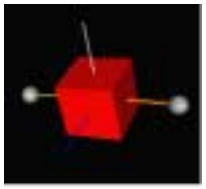


Daniel Quock- my predecessor

- Developed Runge Kutta 4th order numerical integration routine
- Developed means of 2D & 3D visualization
- Developed three scenarios
  - Constant, Body-Fixed Torques
  - Three Reaction Wheels, Actively Controlled Spacecraft
  - Gravity Gradient

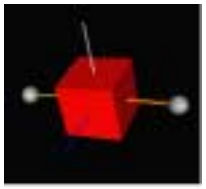
First Point of Contact

```
>From: "Dave Gaylor" <gaylor@csr.utexas.edu>
>To: <gaylor@csr.utexas.edu>
>Subject: Java Astrodynamics Toolkit
>Date: Fri, 10 Jan 2003 13:07:52 -0600
>X-Mailer: Microsoft Outlook IMO, Build 9.0.2416 (9.0.2911.0)
>Importance: Normal
>X-Perlmx-Spam: Gauge=XII, Probability=12%, Report="FROM_AND_TO_SAME"
>

The Java Astrodynamics Toolkit (JAT) Project is an open source
software library designed to support research in astrodynamics,
including space mission design, trajectory optimization, spacecraft
navigation, attitude determination and control. JAT is free software
distributed under the terms of the GNU General Public License.

The JAT Project is always looking for new users and developers. To
find out more about JAT, please visit our webpage at:
http:jat.sourceforge.net.

Please feel free to pass this email on to anyone who might be
interested.


>Dave Gaylor
>Research Assistant
>Center for Space Research
>The University of Texas at Austin
>http://gps.csr.utexas.edu/
```



What I have done…

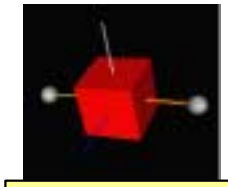
- Added attitude dynamics scenarios
- Developed Attitude Simulation Applet
- Made the applet accessible on-line
- Derived flexible-spacecraft EOMs
- Developed 3D visualization for flexible-body dynamics

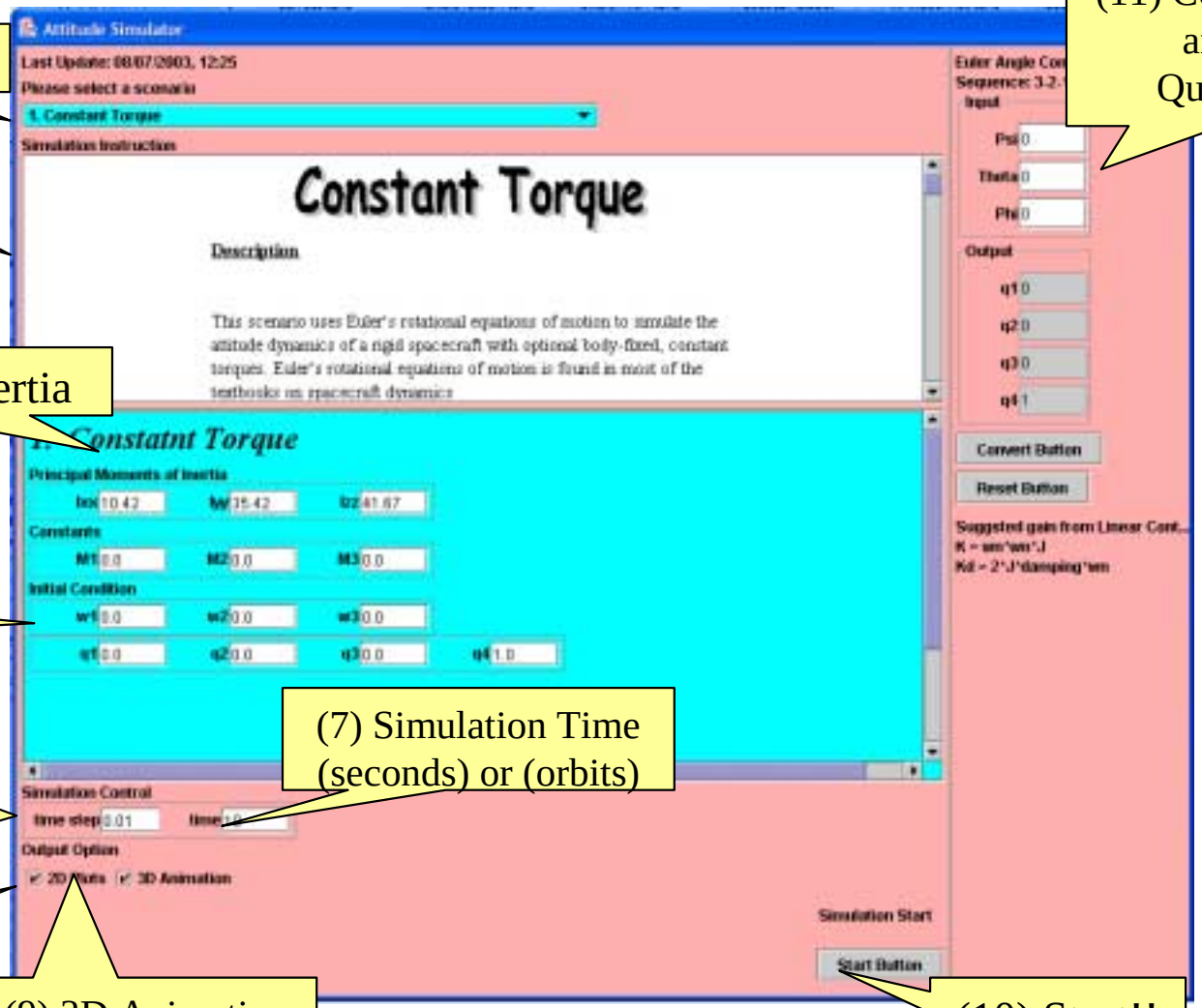


Added more scenarios

1. Constant Torque
2. Gravity Gradient, circular orbit
3. Gravity Gradient, eccentric orbit
4. Spherical Damper
5. Reaction Wheel maneuver
6. CMG maneuver
7. 1-axis Bang-Bang
8. Simple 2-D flexible SC
9. Simple 3-D flexible SC

Developed Simulation Applet





(1) Choose Scenario

(2) Help File

(3) Moments of inertia

(4) Constants

(5) Initial Condition

(6) Numerical Integration Control

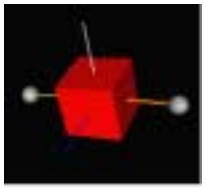
(7) Simulation Time (seconds) or (orbits)

(8) 2D Plots

(9) 3D Animation

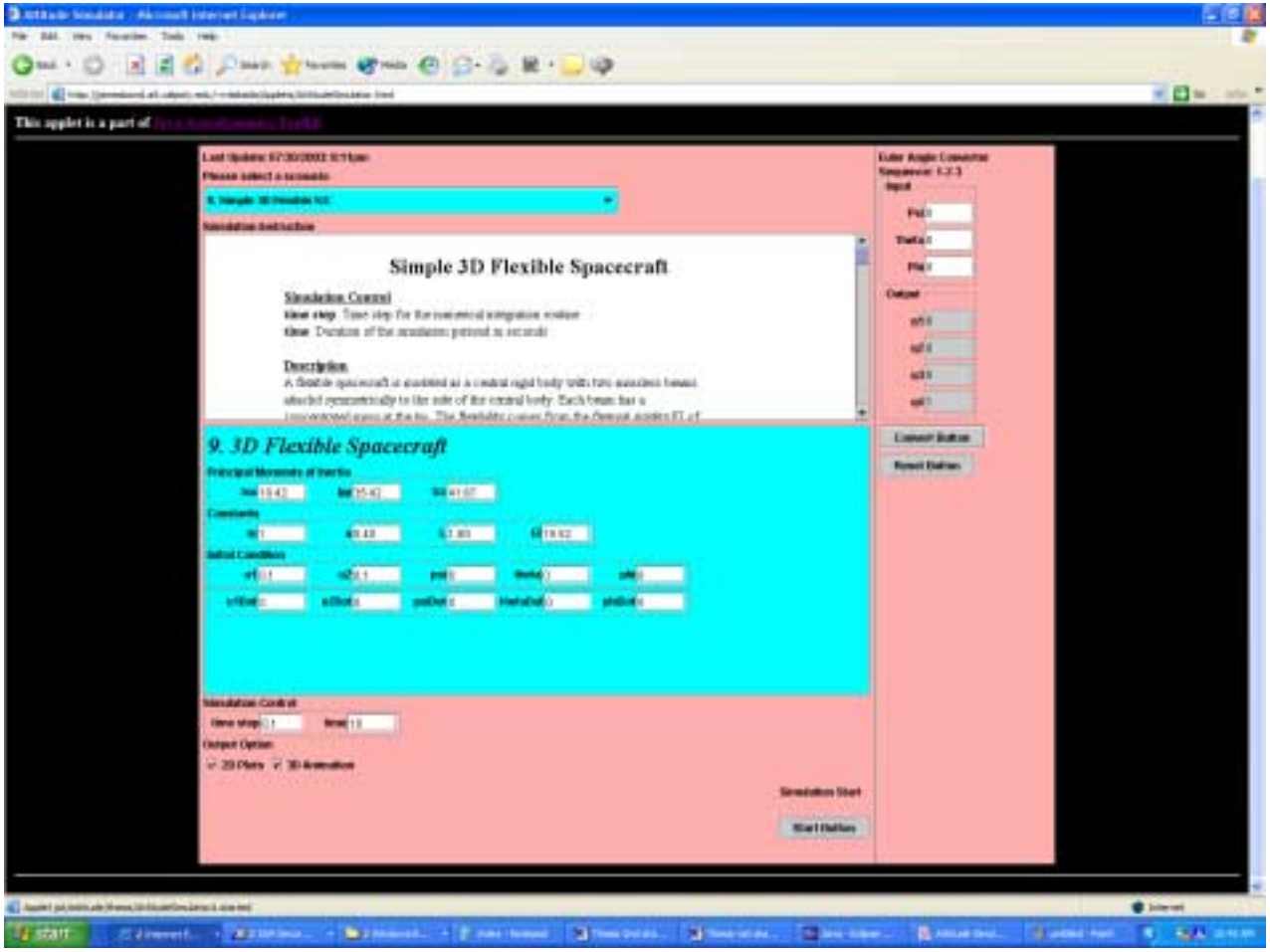
(10) Start!!

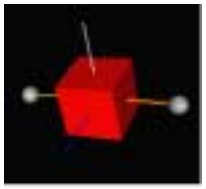
(11) Converts Euler angles to Quaternions



Made the applet accessible

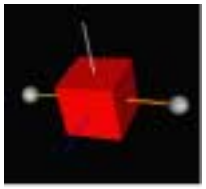
http://jamesbond.atl.calpoly.edu/~ntakada/Applets/AttitudeSimulator.html



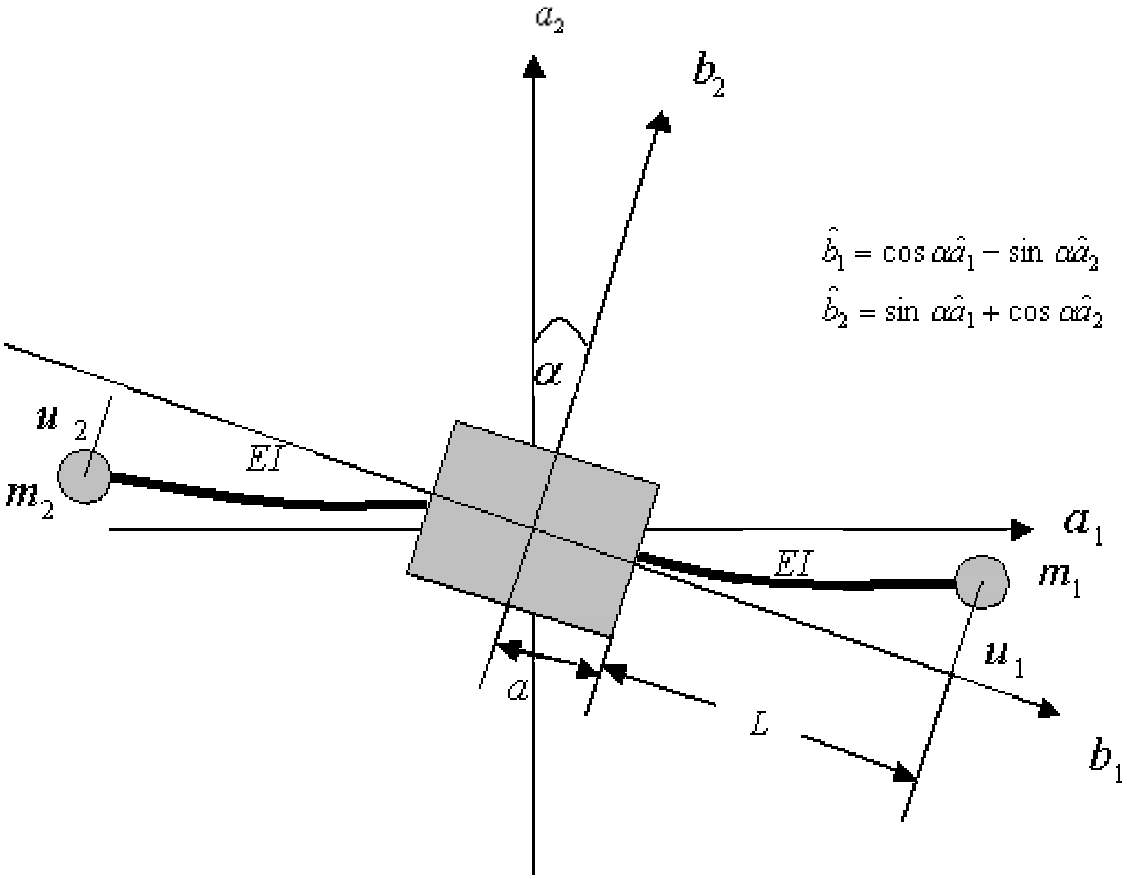


Derived Flexible SC EOMs

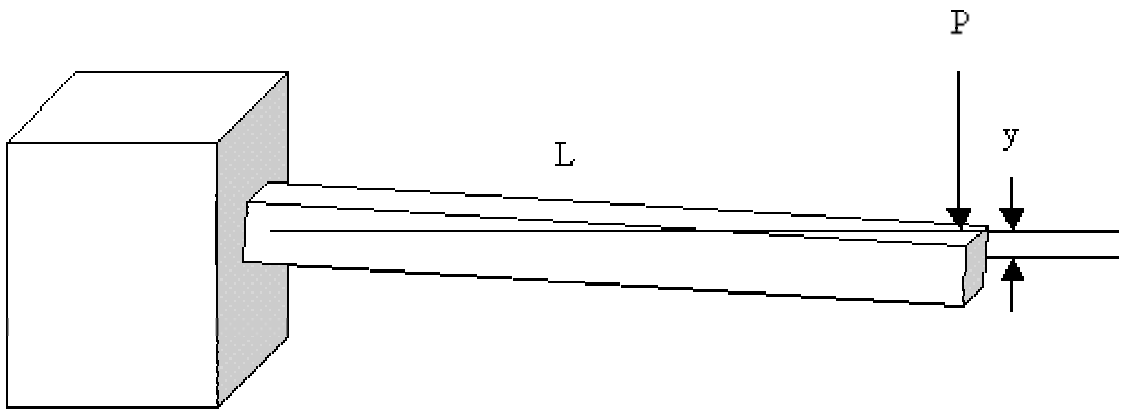
- Used Energy Method
- Used Maple to Solve for a Set of EOMs
- Incorporated Flexibility as beam appendages
- Validated EOMs by the Conservation of Angular Momentum



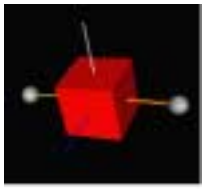
Flexible SC Model Used



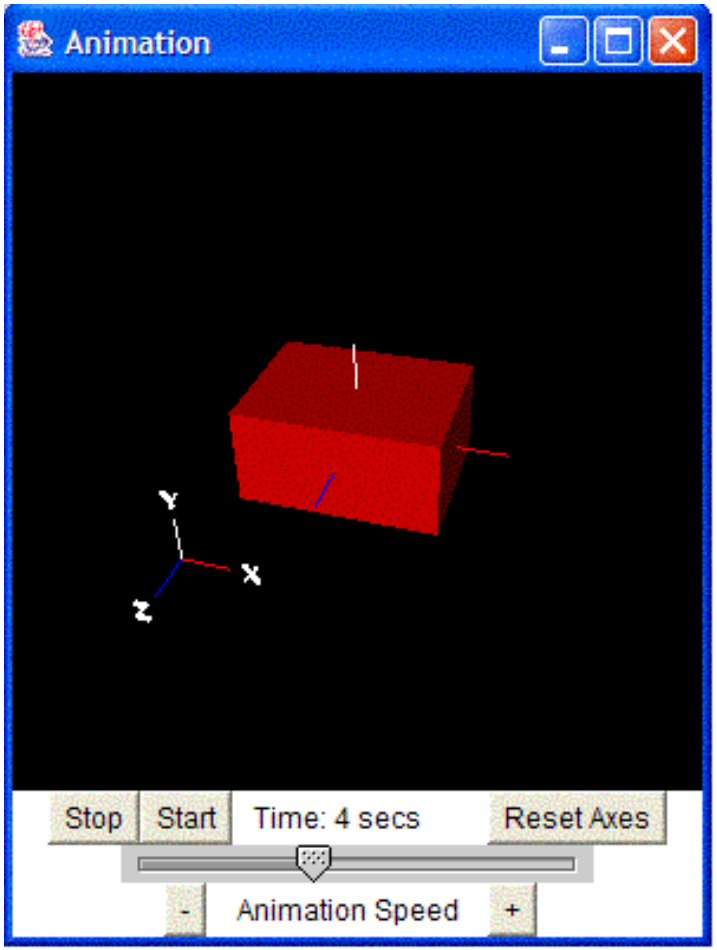
Incorporation of Flexibility

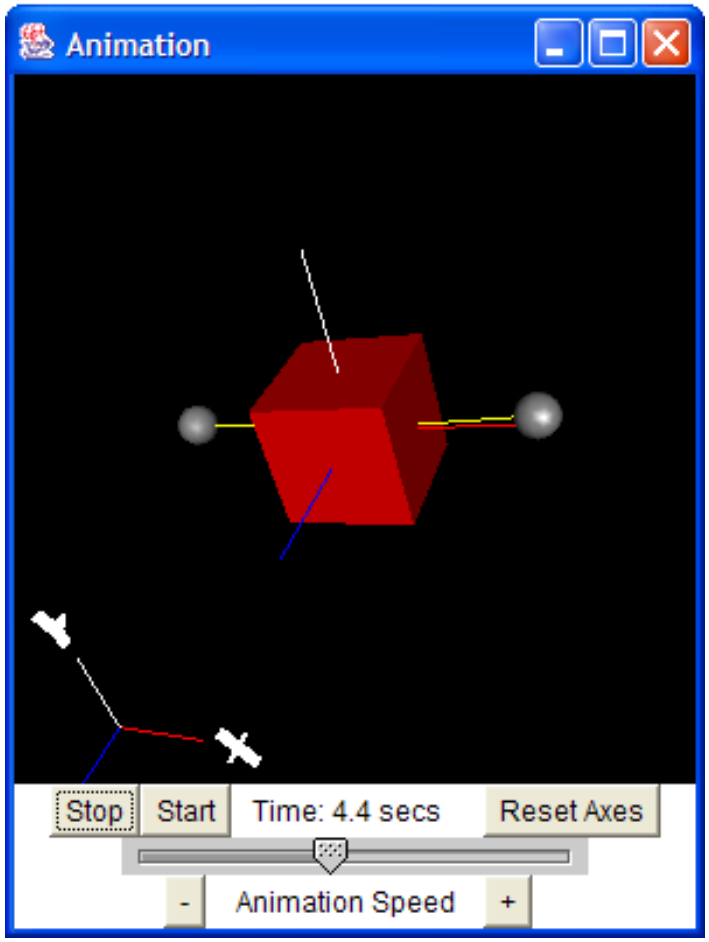


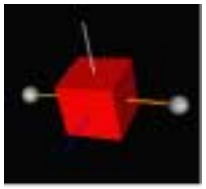
$$y = \frac{P}{6EI}(x^3 - 3Lx^2)$$



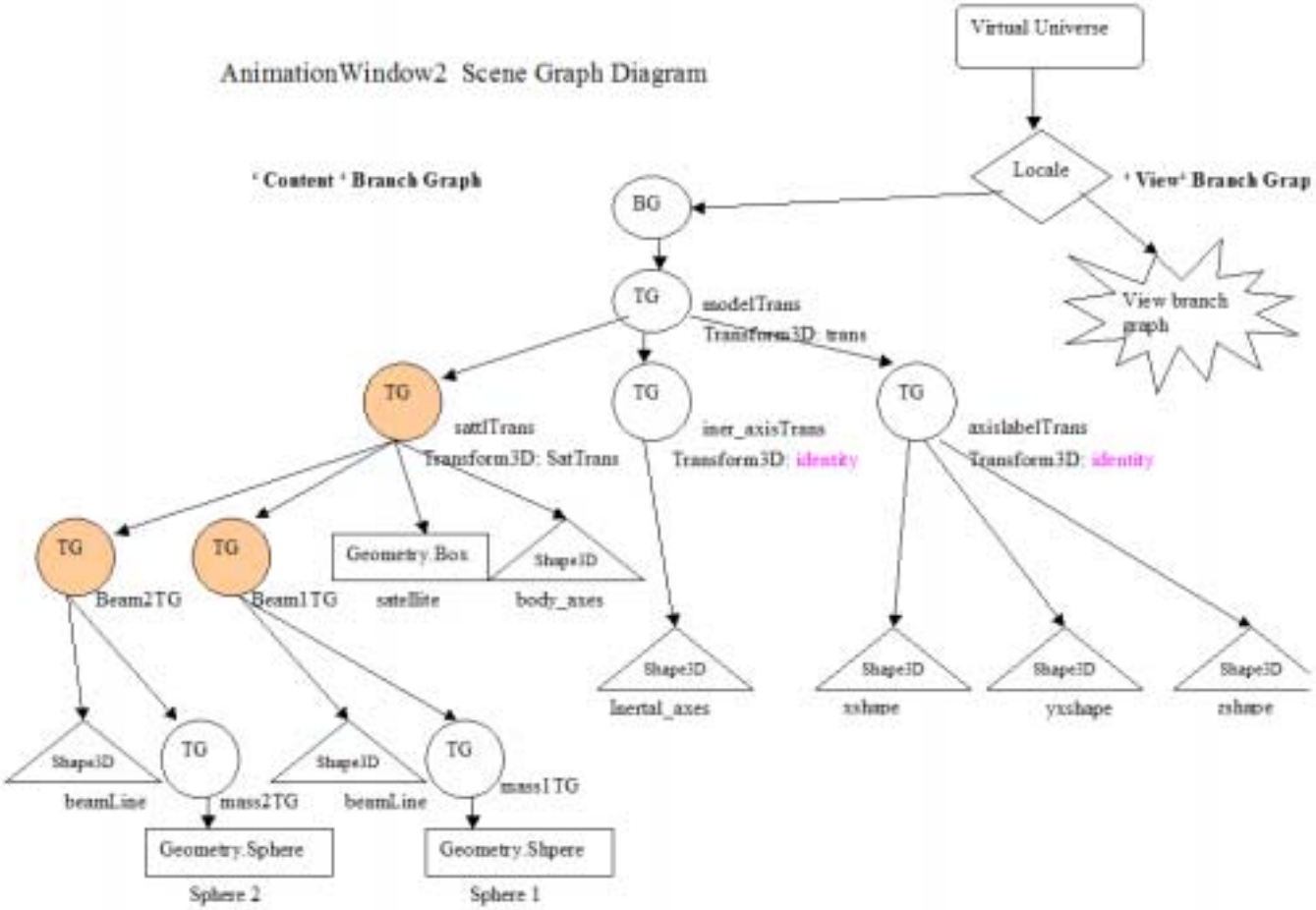
3D Visualization for Flexible SC

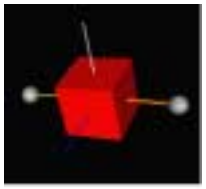




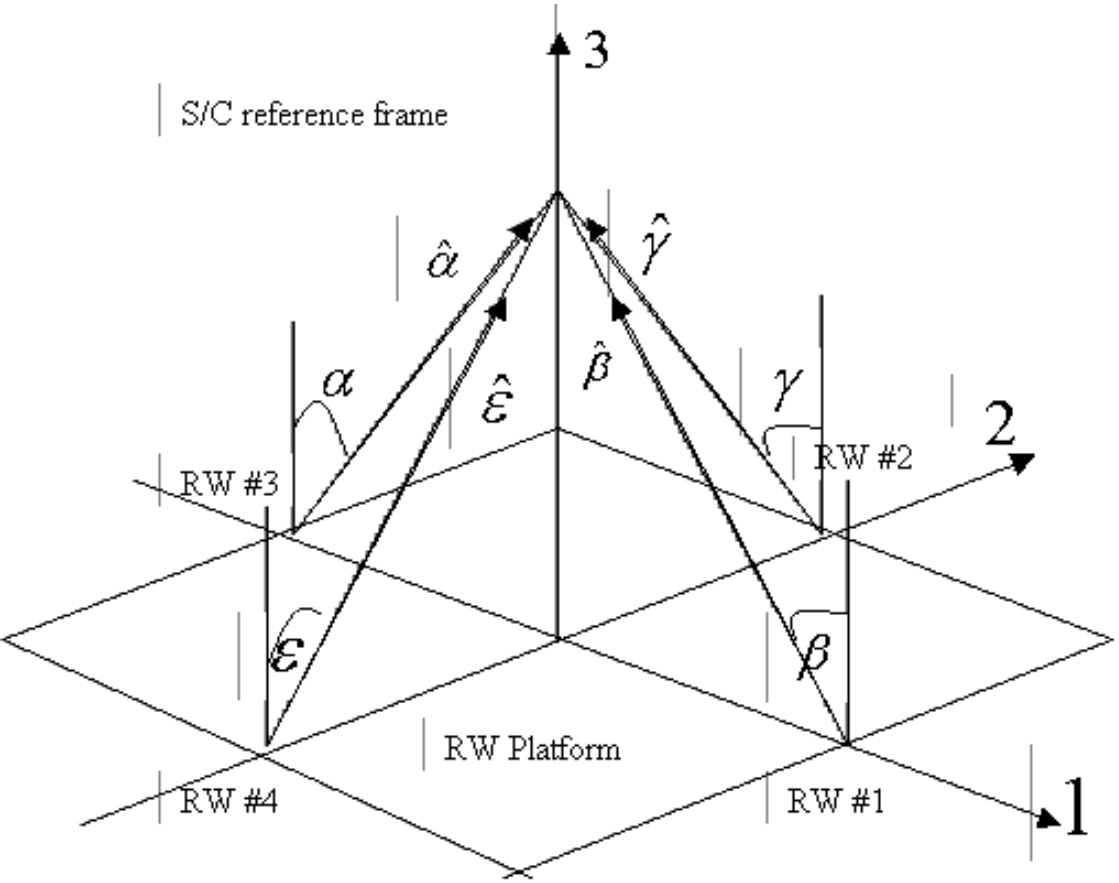


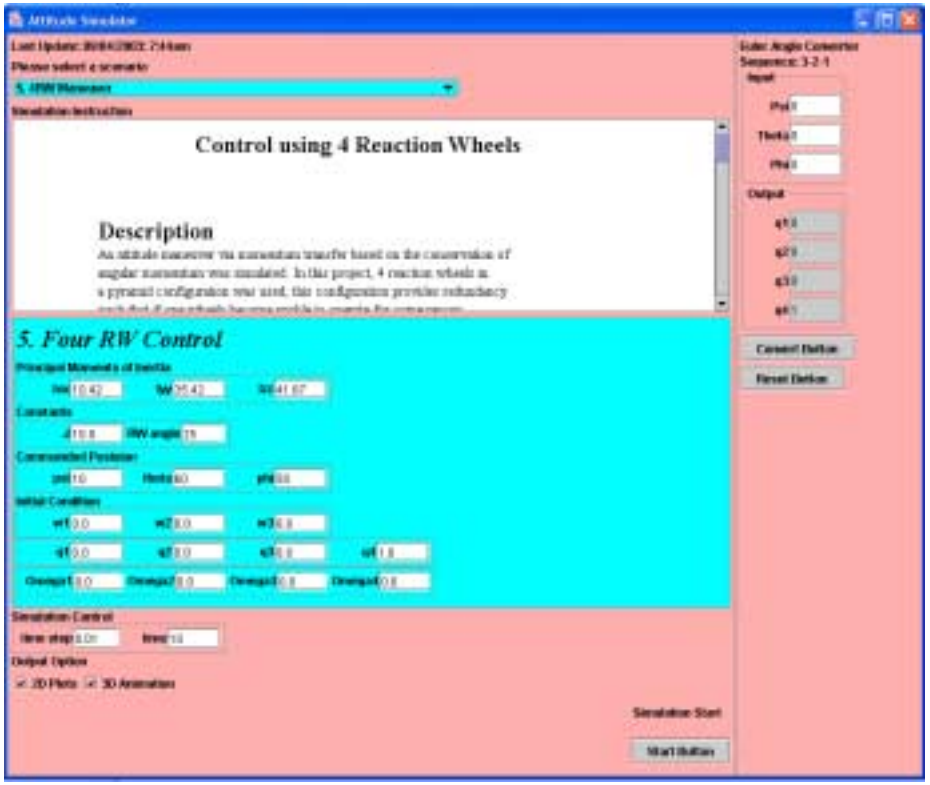
3D Flex SC Inner Workings

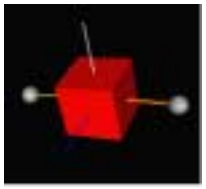




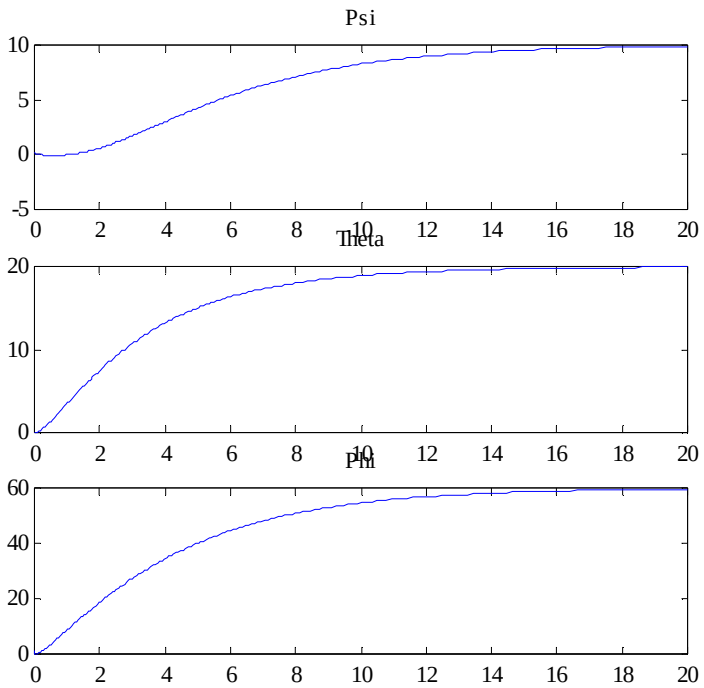
Demo 1: Four Reaction Wheel

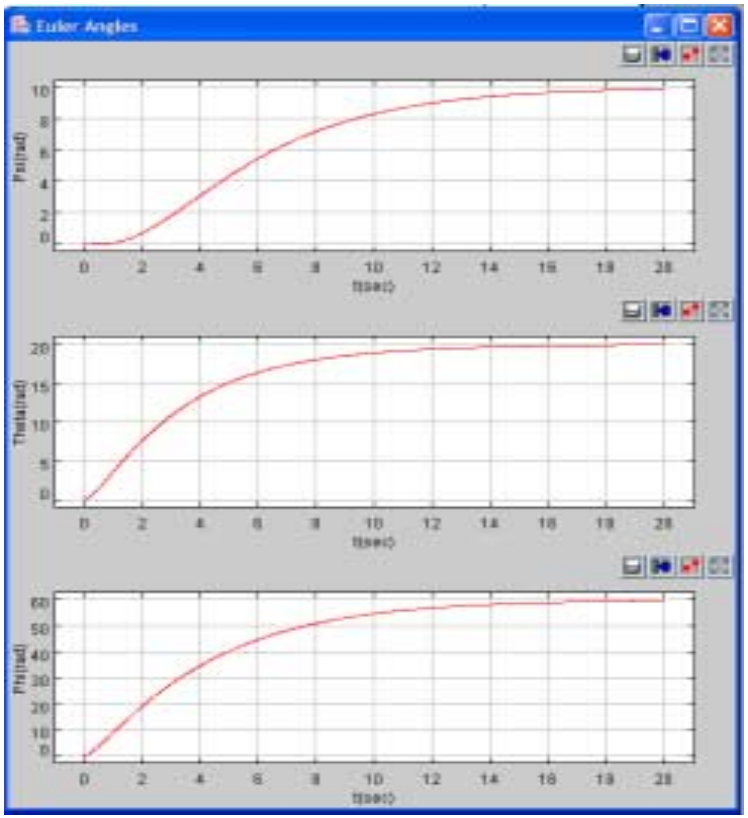


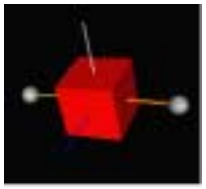




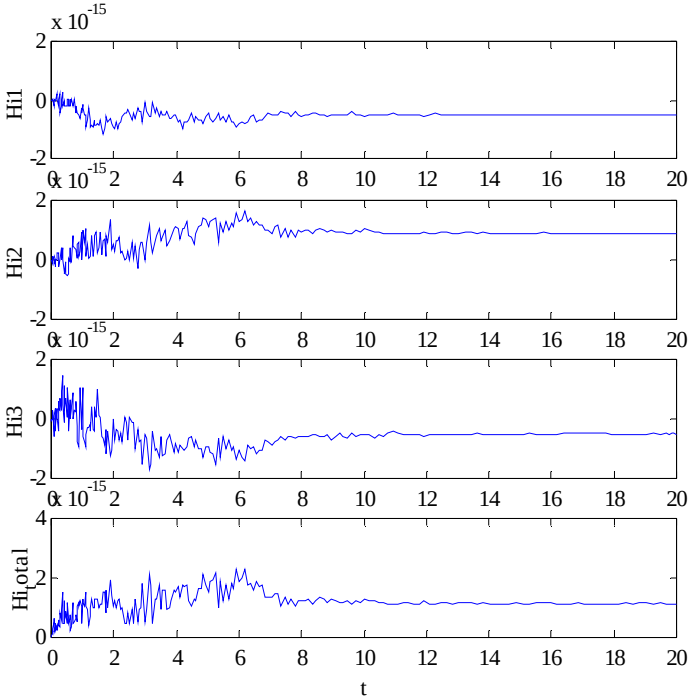
Four Reaction Wheels in action -1

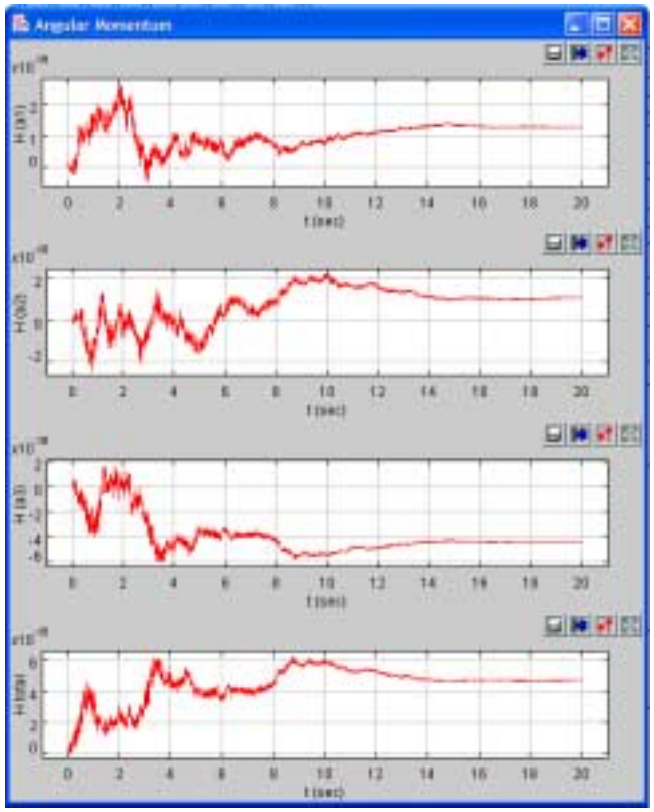


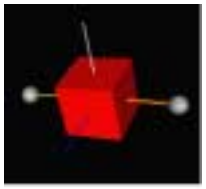




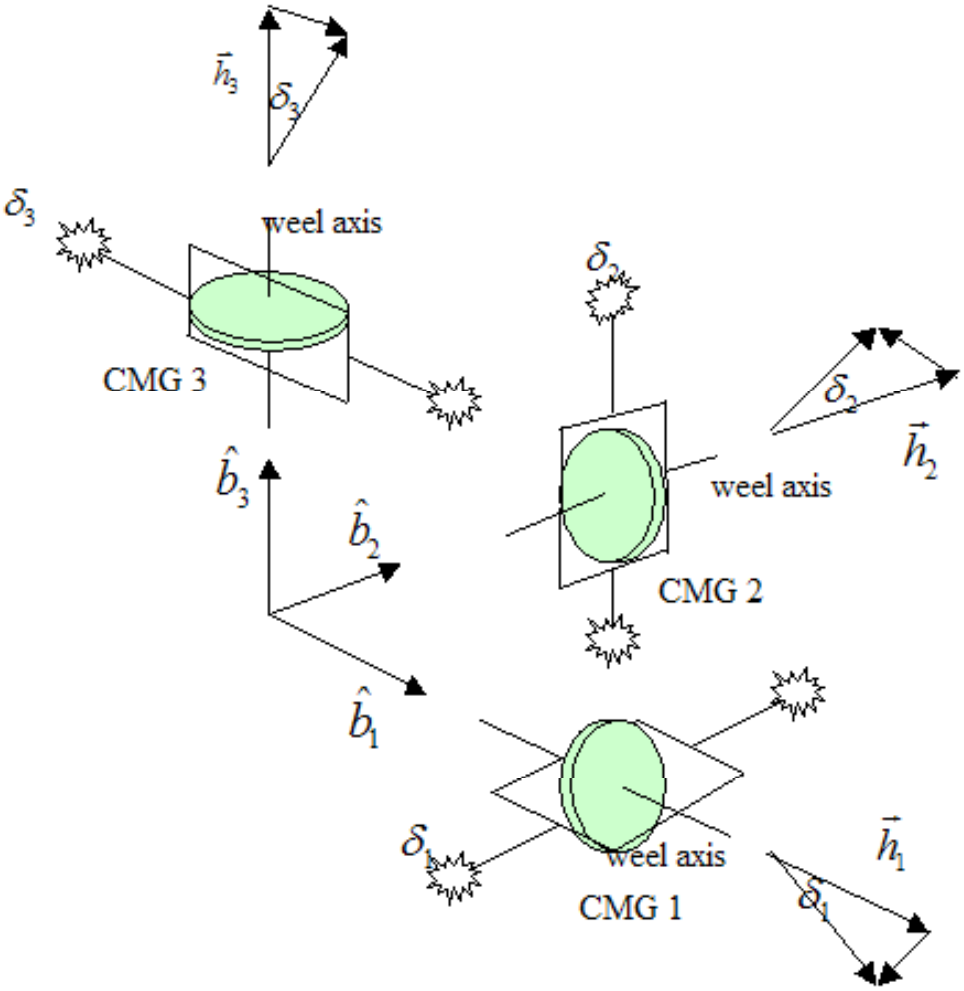
Four Reaction Wheels in action-2

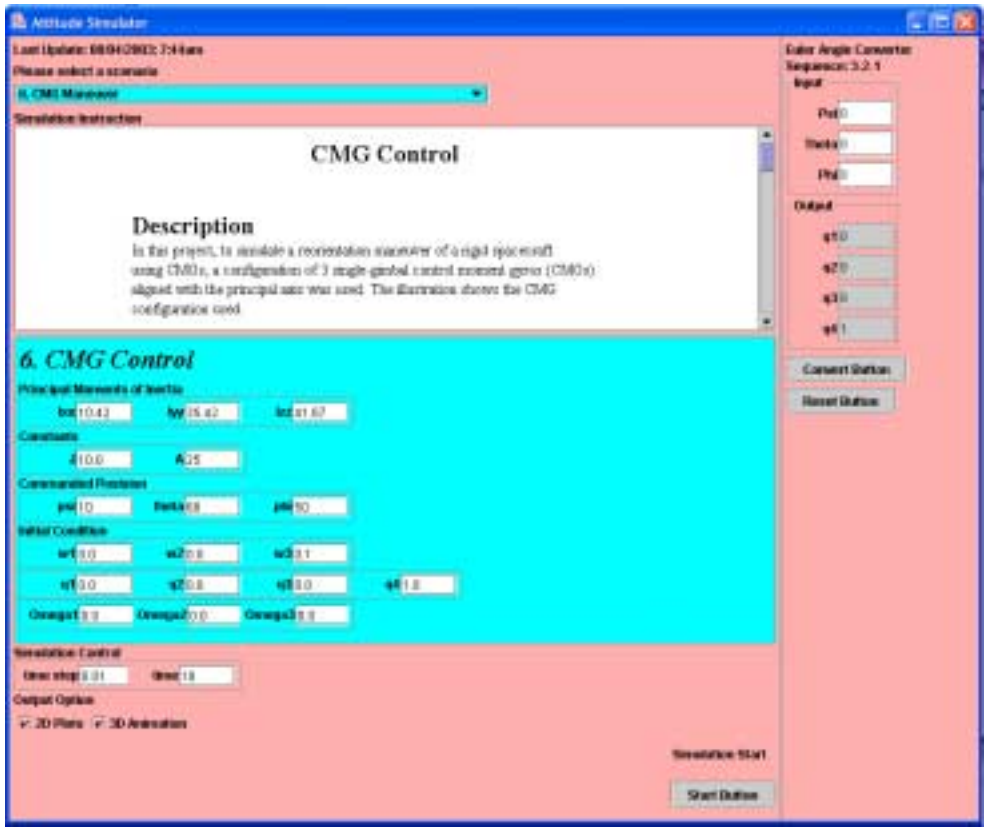


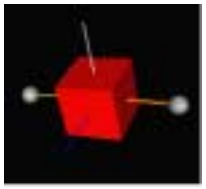




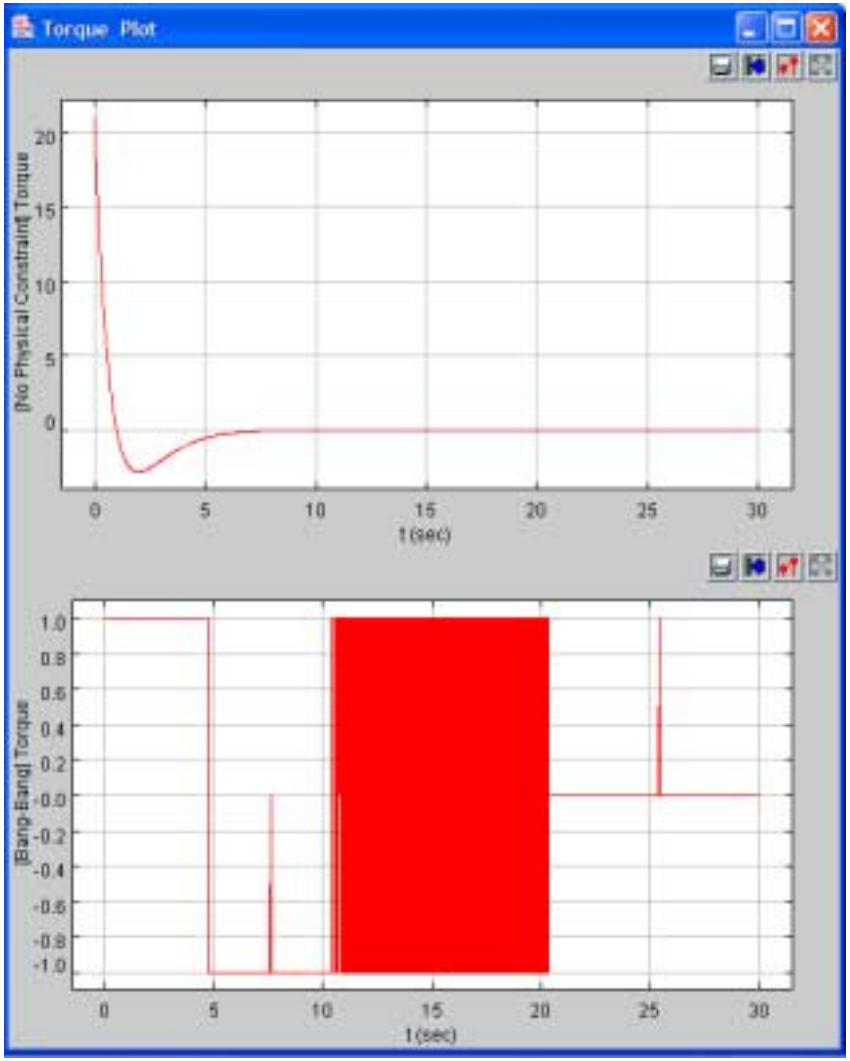
Demo 2: Three CMGs

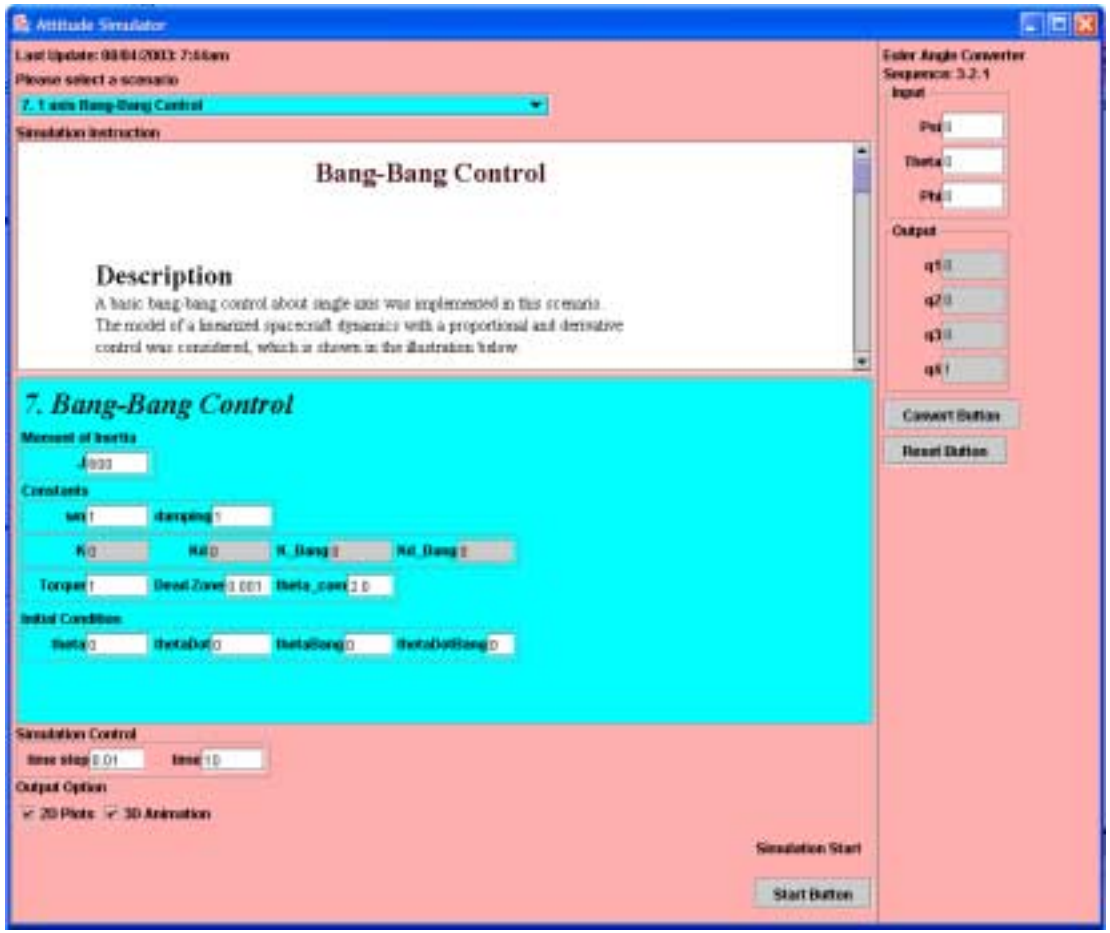


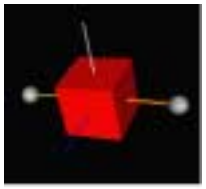




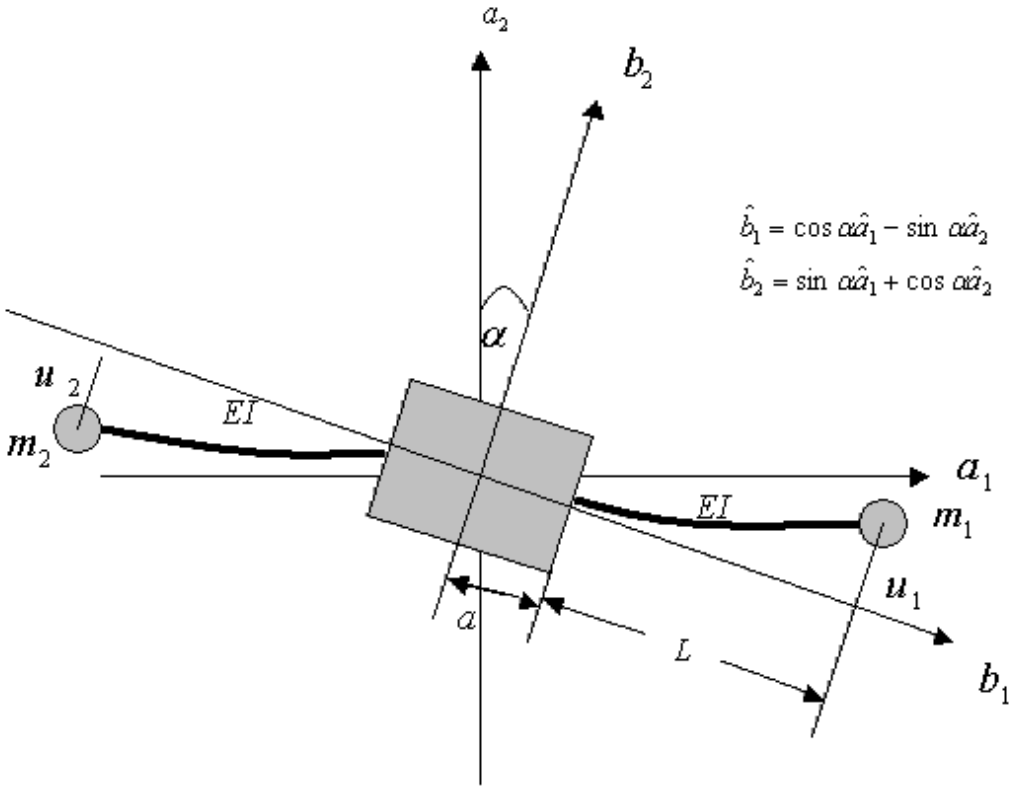
Demo 2: Bang Bang Control

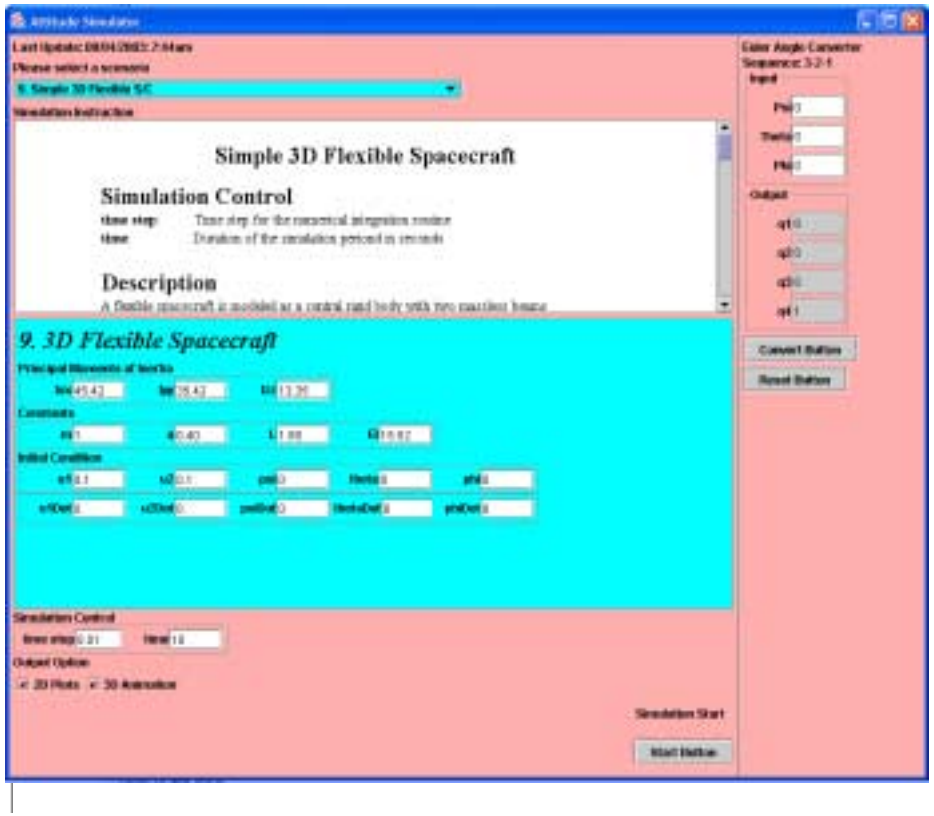


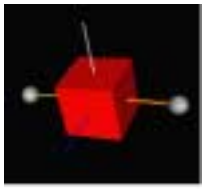




Demo 3- Flexible SC



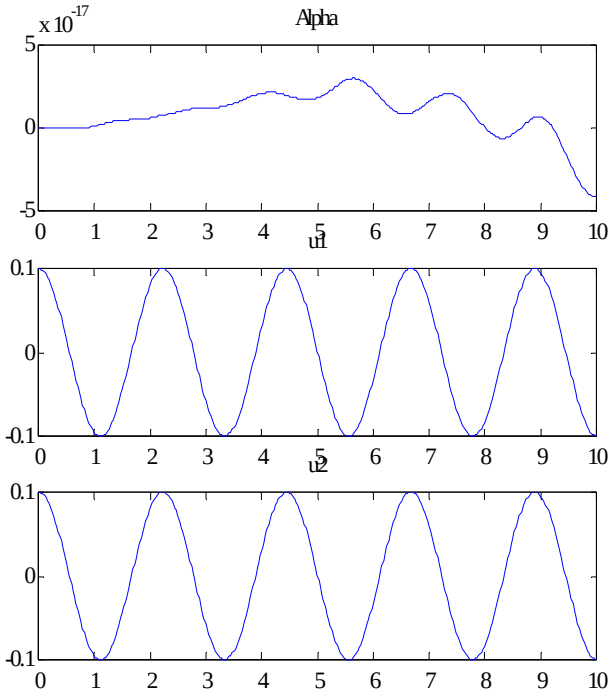


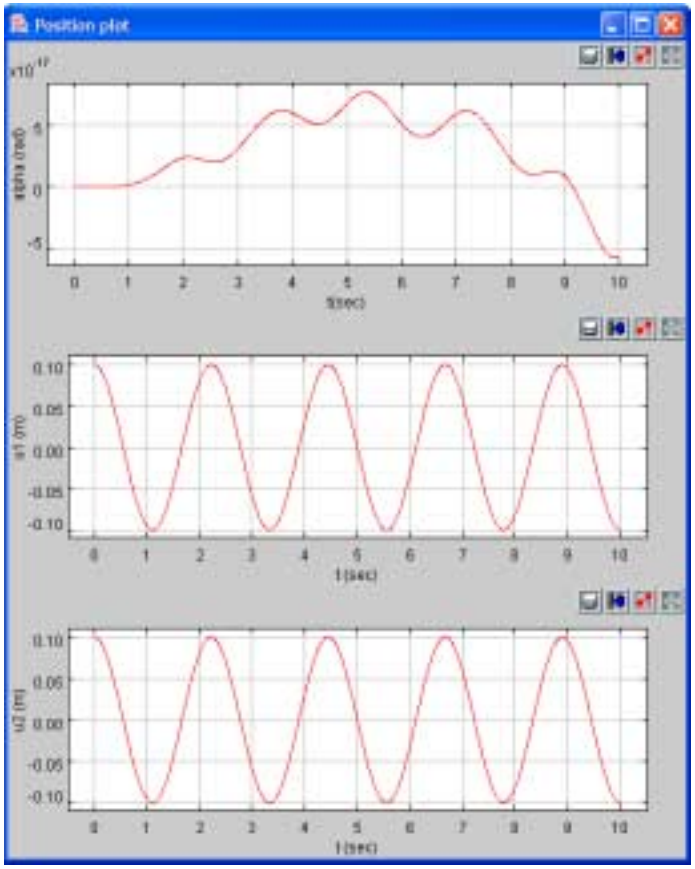


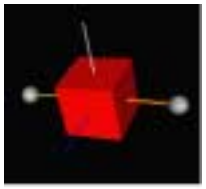
Flexible SC in action-1

Position Plot for a symmetric initial condition

$$\mathbf{x} = \{0 \quad 0.1 \quad 0.1 \quad 0 \quad 0 \quad 0\}$$



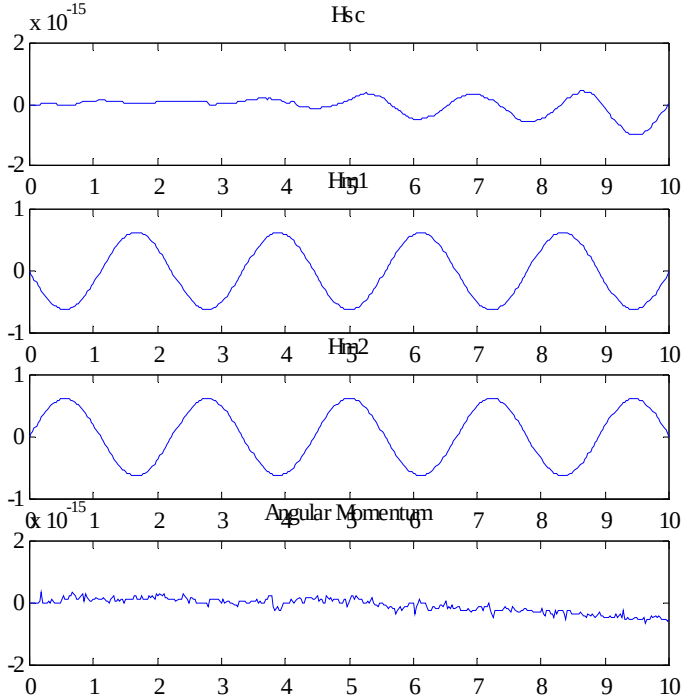


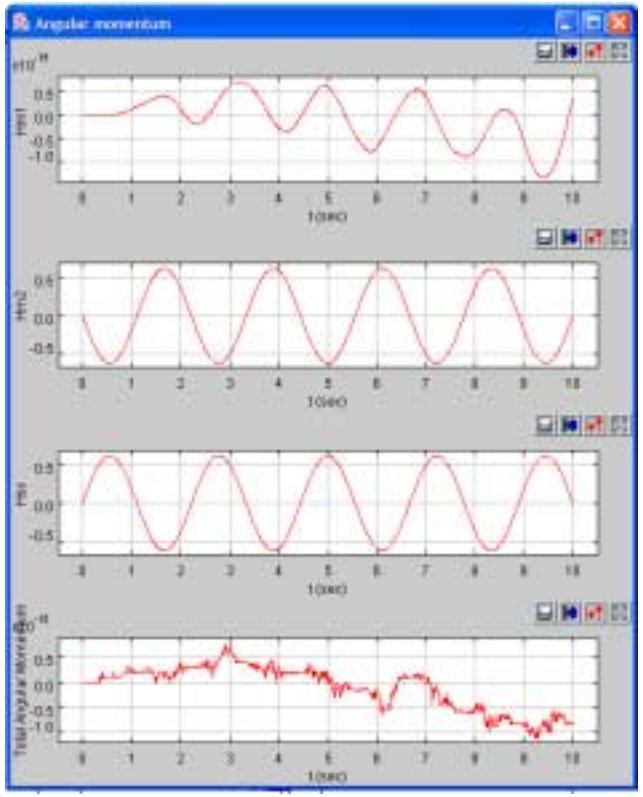


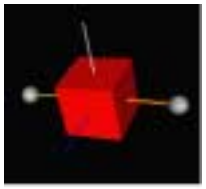
Flexible SC in action -2

Angular Momentum Plot for a symmetric initial condition

$$\mathbf{x} = \{0 \quad 0.1 \quad 0.1 \quad 0 \quad 0 \quad 0\}$$



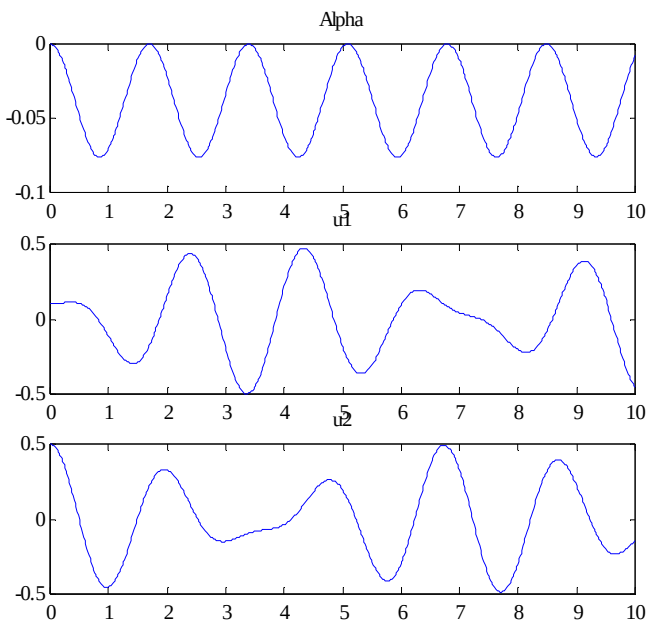


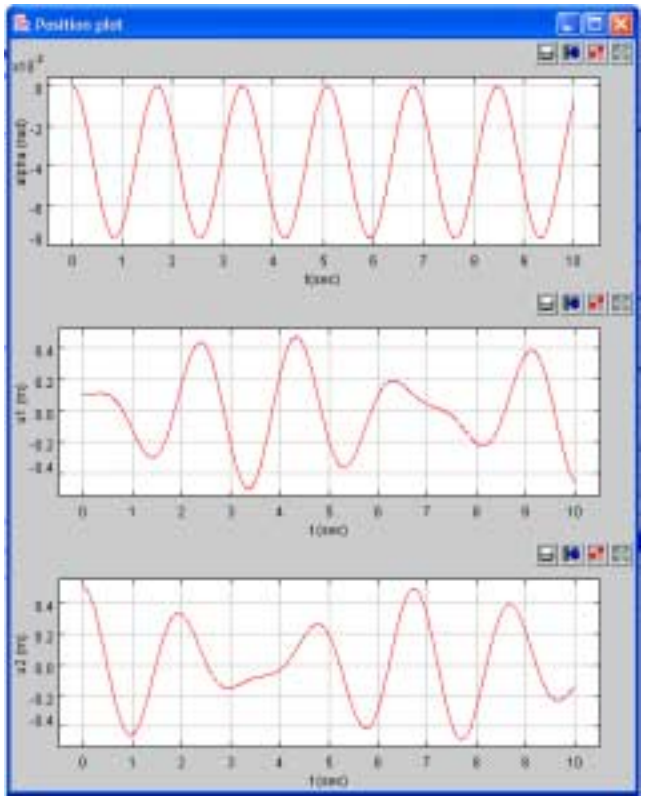


Flexible SC in action- 3

Position Plot for a non-symmetric initial condition

$$\mathbf{x} = \{0 \quad 0.1 \quad 0.5 \quad 0 \quad 0 \quad 0\}$$





JAT

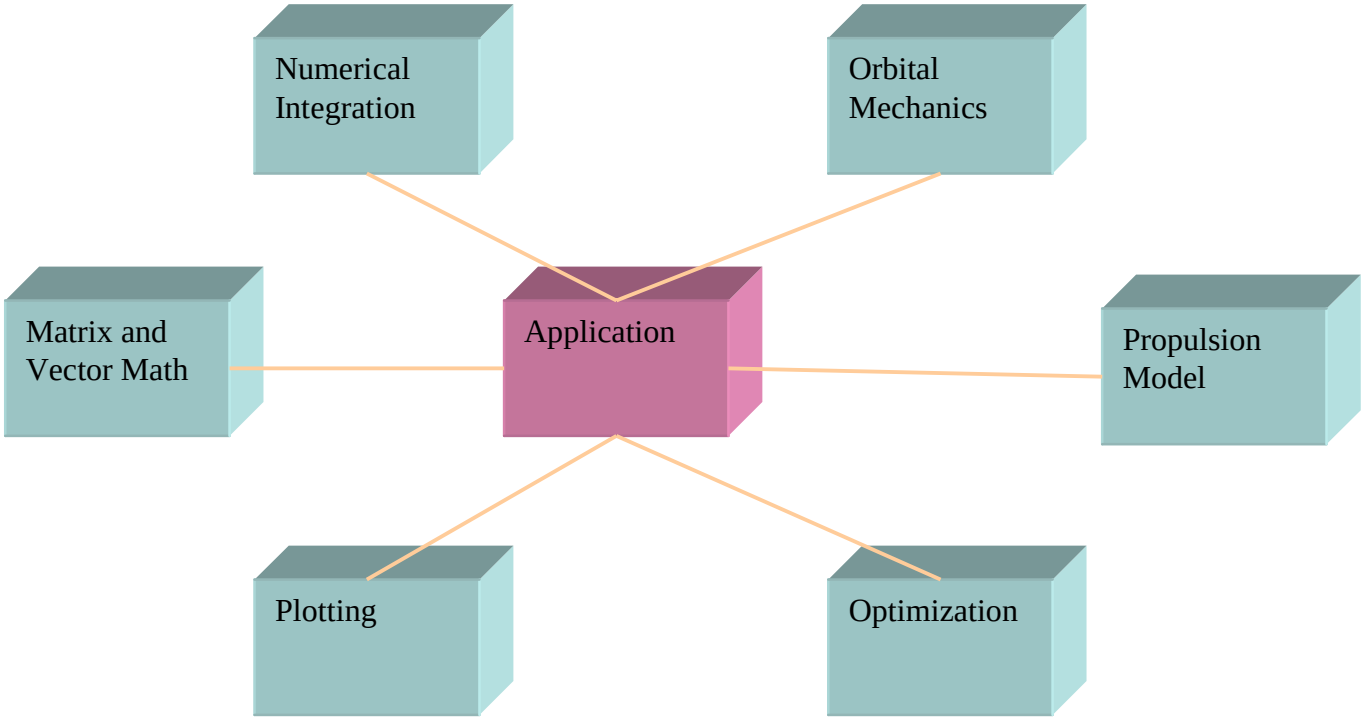
Tobias Berthold

5/21/04

JAT

- Building Aerospace software applications with a component based software library
- An example:
- Trajectory optimization

JAT: Components



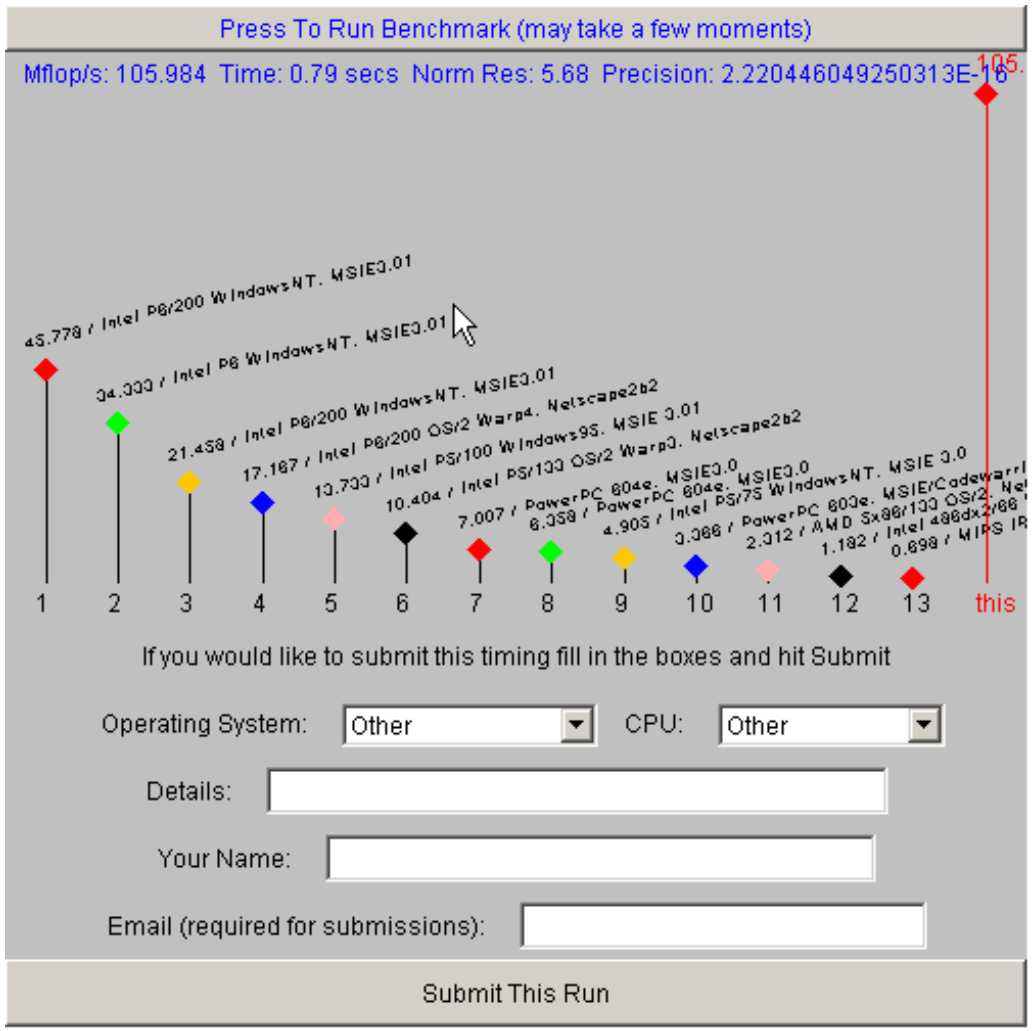
Can Java do the Job in Aerospace?

- Performance
- Numerical Accuracy

Performance

- http://www.netlib.org/benchmark/linpackjava/

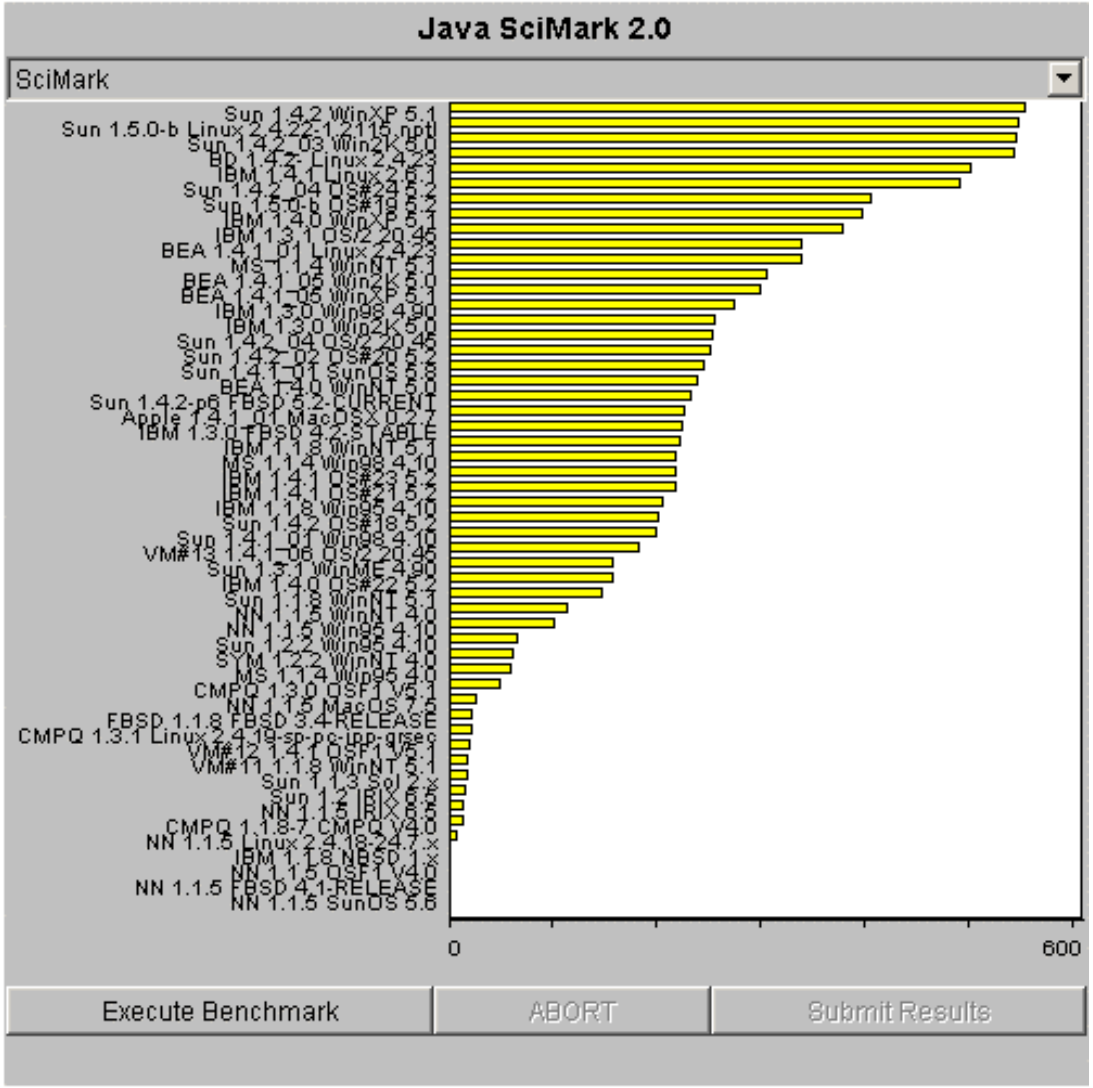
Performance: Netlib



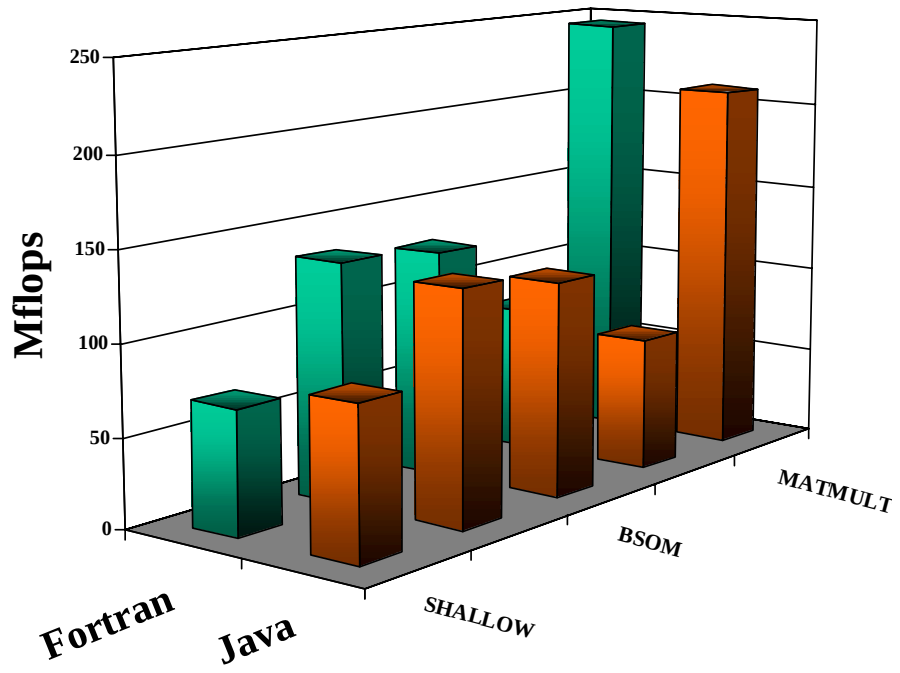
Performance

- Roldan Pozo from NIST
- SciMark
- http://math.nist.gov/scimark2/

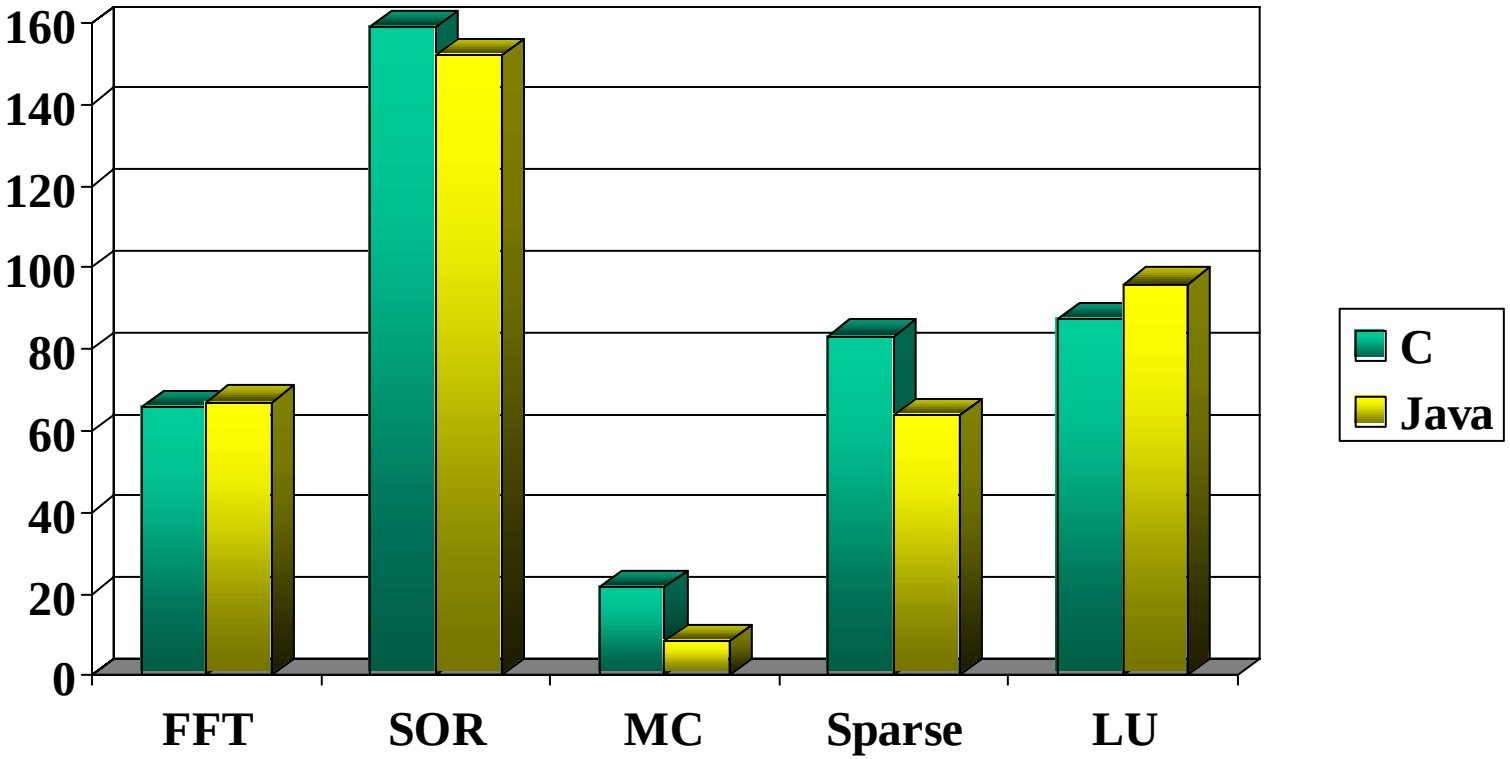
Performance: SciMark



Performance: FORTRAN



Performance: C



Numerical Accuracy

- IEEE-754 1985 floating-point standard
- LBFGS
- Trajectory optimization class, results compared

JAT 3D Visualization

- Takes advantage of Java3D
- Hardware 3D graphics acceleration
- OpenGL and DirectDraw
- Goal: intuitive, easy to use API
- Solar system bodies and spacecraft in JAT
- Loading of 3D models

JAT 3D Visualization

- Base class: Body3D
- Methods for all bodies:
- set_position(..)
- set_attitude(..)

JAT 3D Visualization

- Subclasses of Body3D:
- Planet3D, Star3D, Orbit3D, Ephemeris3D, Models from Lightwave, WaveFront, 3DStudio (space shuttle, etc.)

JAT 3D Visualization

Example:

```
eph = new eph(Path.JATPATH + "\\DE405\\");
earth = new Planet3D(this, Planet3D.EARTH);
jd=cm.juliandate(2004, 5, 21, 12, 0, 0);
earth.set_position(eph.get_planet_pos(eph.EARTH, jd));
```