Dynamics and Real-Time Simulation (DARTS) Laboratory

Spatial Operator Algebra (SOA)

0. Introduction

Abhinandan Jain

June 19, 2024

https://dartslab.jpl.nasa.gov/

Jet Propulsion Laboratory
California Institute of Technology

Outline

- Goals
- Multibody context & needs
- SOA overview
- Discussions group topics

Spatial Operator Algebra (SOA)

- SOA is a mathematical framework for
  - studying and analyzing multibody dynamics
  - exploiting its structure for low-cost recursive (minimal coordinate) algorithms
  - SOA theory originated at JPL, and software used for several research and mission applications
- Lots of publications, but SOA is not taught in academia, and its methods are not well understood
- These set of slides will describe technical ideas towards developing a working knowledge of the SOA methodology
  - Initial focus on SOA foundations
  - Explore new topics and areas

See https://dartslab.jpl.nasa.gov/References/index.php for publications and references on the SOA methodology.

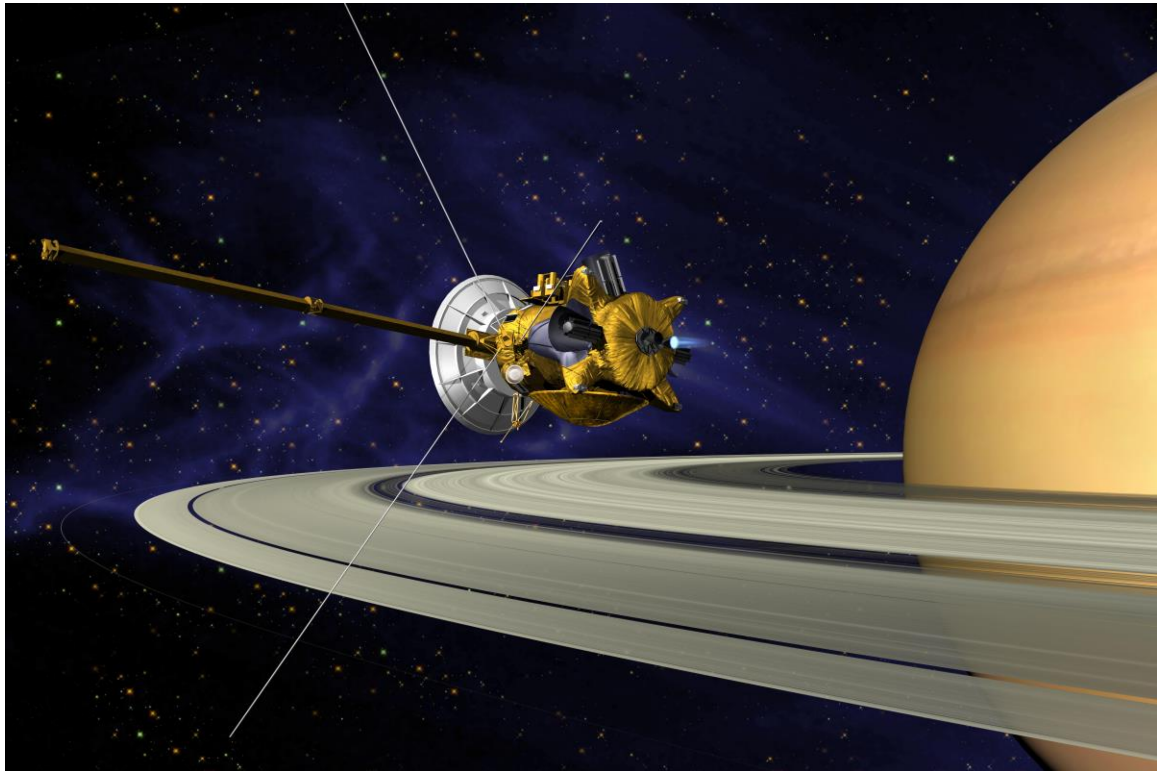
Multibody Background

Multibody System Examples

Multibody dynamics methods are used to model the dynamics of rigid/flexible bodies coupled together by motion constraints, subject to internal and external forces, and interacting with each other and the environment.

Important application areas for multibody system modeling include:

- Spacecraft
- Automotive vehicles
- Aircraft
- Rotorcraft
- Robotic
- Machines
- Biomechanics
- Molecular dynamics
- etc.



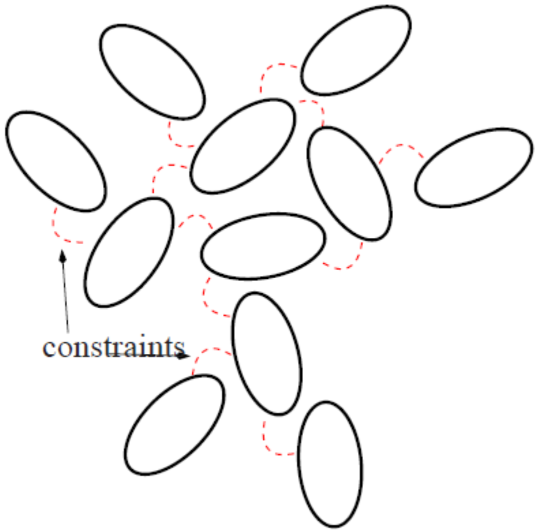
https://www.jpl.nasa.gov/images/pia03883-artistss-conception-of-cassini-saturn-orbit-insertion

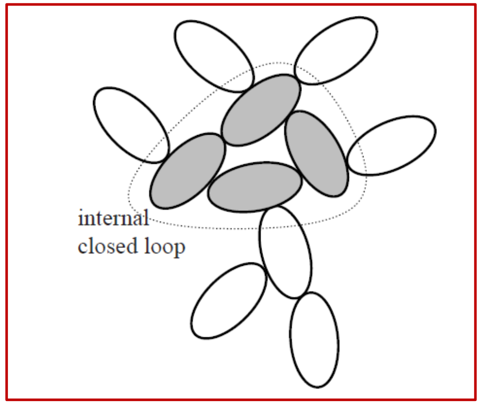
Example multibody domains and needs

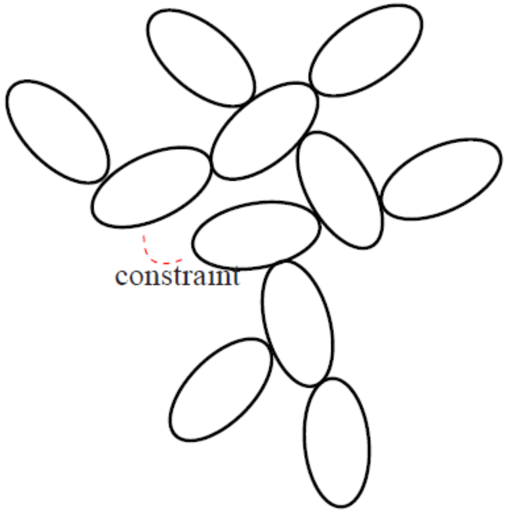
- **Simulation**
  - cross-cutting need across applications, high-fidelity modeling and fast algorithms, complex motions and changing configuration
  - design, optimization
- **Control**
  - Reduced order/derived models, state space representations
- **Robotics**
  - Embedded models for planning/monitoring/reasoning/control
  - Real-time configuration/constraint changes
  - Variety of model-based computations - Jacobians, forward/inverse kinematics, load-balancing, whole-body motion control, manipulation, legged locomotion
- **Molecular dynamics**
  - Very large number of dofs, internal coordinate dynamics, statistical simulations, multi-scale reduced order models

6

Multibody Dynamics Formulation Options







Absolute/Redundant coordinates

- All bodies are treated as independent
- Bilateral constraints used for all hinges, DAE form
- Large no. of dofs, but simple to set up
- Constant, block diagonal mass matrix

Minimal/Relative coordinates

- Minimal hinge coordinates, ODE form
- Smaller no. of dofs
- Dense, configuration dependent mass matrix

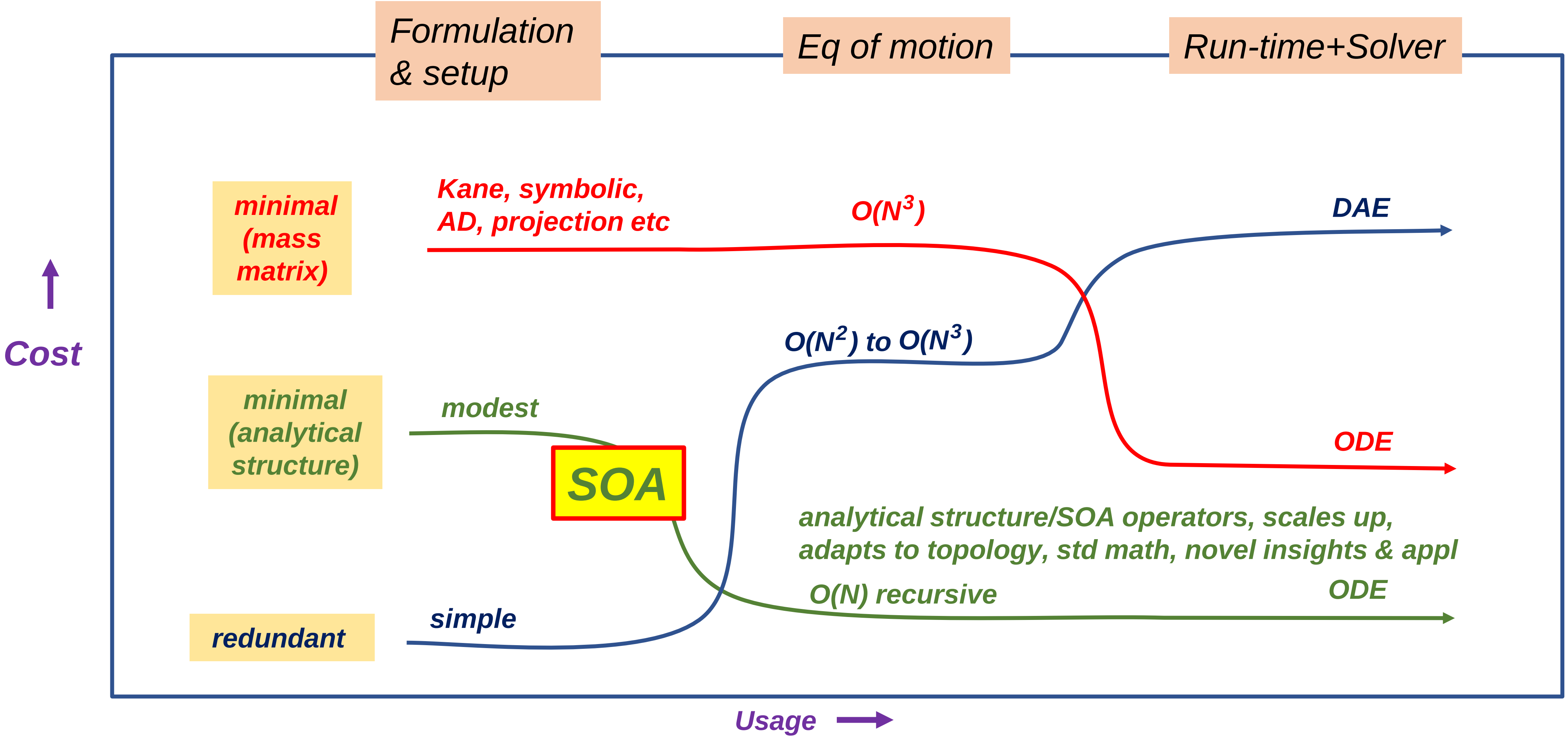
$$\mathcal{M}(\theta)\ddot{\theta} + \mathcal{C}(\theta,\dot{\theta}) + G_c{}^*(\theta,t)\lambda = \mathcal{T}$$

Algebraic constraints term for redundant coordinates only →

$$G_c(\theta,t)\dot{\theta} = \mathfrak{U}(t)$$

equations of motion

Formulations Comparison



Spatial Operator Algebra (SOA) enables systematic use of minimal coordinates for general dynamics while exploiting underlying structure for fast computations.

Spatial Operator Algebra (SOA) Overview

Spatial Operator Algebra for Minimal Coordinate Dynamics

- **Minimal coordinate**: while complex, have smaller models, ODE formulation & rich underlying structure
- **Operators: SOA's** expressive mathematical language for *'analytical multibody dynamics'* & exploiting this structure
  - **Breadth:** for **concisely** representing large family of key dynamics & kinematics quantities (Jacobian, mass matrix, OSC etc)
  - **Depth:** Operator expressions remain **invariant** to size, branching topology, rigid/flex bodies, constraint embedding!
- **Algebra:** Can combine and manipulate operator expressions to get **simpler expressions**
- **Computation:** Can directly operator expressions into fast "O(N)" structure-based recursive algorithms by simple **inspection**

SOA Analysis to Recursive Algorithms

Mass matrix

$$\mathcal{M} = H\phi M\phi^* \boldsymbol{H}^* \quad \rightarrow \quad O(\mathcal{N}) \text{ Newton-Euler Inverse Dynamics}$$

non-square factors

$O(\mathcal{N}^2)$ Composite Body Algorithm for $\mathcal{M}$

$O(\mathcal{N}^3)$ Forward Dynamics

$$\mathcal{M} = [I + H\phi K]D[I + H\phi K]^* \quad \rightarrow \quad O(\mathcal{N}^2) \text{ Forward Dynamics}$$

square factors

$$[I + H\phi K]^{-1} = [I - H\psi K]$$

analytical factor inverse

$$\mathcal{M}^{-1} = [I - H\psi K]^* D^{-1}[I - H\psi K] \quad \rightarrow \quad O(\mathcal{N}) \text{ Articulated Body Forward Dynamics}$$

Mass matrix inverse

$O(\mathcal{N}^2)$ Computation of $\mathcal{M}^{-1}$

SOA in relationship to Robotics/Aerospace communities

Aerospace community
Emphasis on **rigid/flex** systems, general purpose, based on **non-minimal** coords, rely on numerical solvers, geared towards GNC & sim. needs, Kane's method

Recursive methods for general rigid/flex topology systems

Spatial Operator Algebra
(minimal coordinates)

Enabled new applications

Novel applications
Operational space inertias, Constraint embedding, Fixman potential, diag. dynamics, underactuated, sensitivities etc.

Unified approach & expanded class of fast, recursive algorithms

Robotics community
Some novel **recursive** algorithms, **minimal** cords, exploiting structure, but limited to **rigid** body systems

Newton-Euler O(N) inverse dynamics

Composite body method for mas matrix

Articulated body O(N) method for forward dynamics

SOA history

- SOA was invented by Guillermo Rodriguez at JPL in the mid ‘80s for robotics research
- He recognized the rich mathematical parallels between the Kalman filtering in the time domain and multibody dynamics in the spatial domain
  - Kalman methods are lower cost and numerically more accurate compared to traditional methods (carries over to multibody dynamics as well!)
- Led to the SOA methodology that provides the mathematical language for making minimal coordinate dynamics tractable.
- Modernized and made standard the “spatial” vector/inertia conventions widely used now
- Guy Man convinced the Cassini project on the ability of SOA algorithms to speed up flex dynamics simulations – leading to initial DARTS multibody dynamics software
  - DARTS has been subsequently extended and applied to robotics, landers, ground vehicles, rotorcraft, molecular dynamics applications
- SOA has been significantly generalized and extended over the years. SOA book published in 2011.

SOA Novel Application Examples

SOA general purpose modeling approach

Analysis

Newton–Euler Factorization of $\mathcal{M}$

$$\mathcal{M} = H\phi M\phi^* H^*$$ Non-square factors

Innovations Factorization of $\mathcal{M}$

$$\mathcal{M} = [I + H\phi K]D[I + H\phi K]^*$$ LDU decomposition of $\mathcal{M}$

Analytic inverse of $[I + H\phi K]$

$$[I + H\phi K]^{-1} = [I - H\psi K]$$ Analytic inverse

Operator Factorization of $\mathcal{M}^{-1}$

$$\mathcal{M}^{-1} = [I - H\psi K]^* D^{-1}[I - H\psi K]$$ LDU decomposition of $\mathcal{M}^{-1}$

Algorithms

- Inverse dynamics
- Forward dynamics
- Hybrid dynamics
- Mass matrix comp.
- Mass matrix inverse
- Composite rigid body inertias
- Operational space inertia
- ……

Dynamics modeling

SOA computational mechanics

Serial-chain & tree systems

Rigid & flexible body systems

…..

Under-actuated systems

Flexible-joint & geared systems

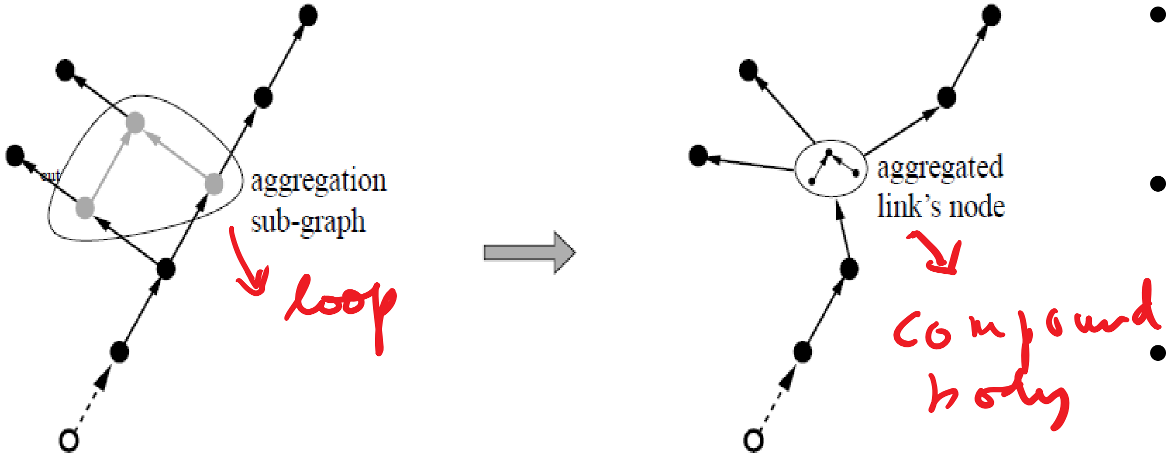
… or some combination thereof

SOA operator results apply to <u>broad class</u> and <u>size</u> of multibody systems – only the low level components change for different systems.

SOA Constraint Embedding for Closed-Loop Systems

With closed-loop, mass matrix is singular – paradise lost!

Constraint embedding transforms a closed-loop system graph into a tree graph



Original digraph

Transformed tree digraph with aggregated link

*The compound body is a **"variable geometry body"!***

- *Minimal coordinate ODE model for a closed-loop system*
- *Tree analytical structure is regained*
- *Well defined non-singular mass matrix*
- *Mass matrix inversion results hold again*
- *Recursive O(N) methods are available again as well*

Paradise regained!

Operational Space Inertia (OSI)

The **operational space inertia** is an important quantity that shows up in whole-body motion control for robotics and closed-chain dynamics for simulation

$$\underline{\Lambda} \stackrel{49}{=} \mathcal{J}\mathcal{M}^{-1}\mathcal{J}^* \stackrel{59}{=} \mathcal{B}^*\phi^*H^*(I - H\psi\mathcal{K})^*\mathcal{D}^{-1}(I - H\psi\mathcal{K})H\phi\mathcal{B}$$

$$= \mathcal{B}^*\Omega\mathcal{B} \text{ with } \Omega \stackrel{\triangle}{=} \psi^*H^*\mathcal{D}^{-1}H\psi$$

$$\Omega = \Upsilon + \tilde{\psi}^*\Upsilon + \Upsilon\tilde{\psi}$$

Using SOA techniques

Leads to the **fastest** available algorithm for computing the OSI!

Analytical Sensitivity Expressions

Spatial Operator expression for the mass matrix sensitivity.

$$\mathcal{M} = \mathrm{H}\phi\mathbf{M}\phi^*\mathrm{H}^*$$

$$\begin{aligned}
\dot{\mathcal{M}} &= \frac{d[H\phi M\phi^* \boldsymbol{H}^*]}{dt} \\
&= [\dot{H\phi}]M\phi^*\boldsymbol{H}^* + H\phi\dot{M}\phi^*\boldsymbol{H}^* + H\phi M[\phi^*\dot{\boldsymbol{H}}^*] \\
&= \boldsymbol{H}\phi[\tilde{\Omega}_\delta\phi - \tilde{\Omega}]M\phi^*\boldsymbol{H}^* + H\phi[\tilde{\Omega}M - M\tilde{\Omega}] \\
&\qquad + H\phi M[\tilde{\Omega} - \phi^*\tilde{\Omega}_\delta] \\
&= H\phi[\tilde{\Omega}_\delta\phi M - M\phi^*\tilde{\Omega}_\delta]\phi^*\boldsymbol{H}^*
\end{aligned}$$

$$\frac{\partial\mathcal{M}(\theta)}{\partial\theta(i)} = \boldsymbol{H}\phi\left[\mathcal{H}^i_\delta\phi\boldsymbol{M} - \boldsymbol{M}\phi^*\mathcal{H}^i_\delta\right]\phi^*\boldsymbol{H}^*$$

Useful for linearization, integrator design, optimization.

SOA Diagonalized Dynamics

$$\underbrace{\mathcal{M}(\theta)\ddot{\theta} + \mathcal{C}(\theta,\dot{\theta}) = T}_{\text{highly coupled}} \quad \rightarrow \quad \underbrace{\dot{\nu} + \mathcal{C}(\theta,\nu) = \epsilon}_{\text{largely decoupled}}$$

Global diagonalizing transformations require that the curvature tensor associated with the mass matrix vanish. This is rarely the case.

Use diagonalizing coordinates (θ, ν) $\quad \dot{\theta} \rightarrow \nu \stackrel{\triangle}{=} \mathcal{M}^{\frac{1}{2}}\dot{\theta}$

$$\nu = D^{\frac{1}{2}}[I + H\phi K]^*\dot{\theta} \quad \text{and} \quad \epsilon = [I - H\psi K]D^{-\frac{1}{2}}T$$

- These smooth diagonalizing transformations always exist.
- The ν's are non–integrable time derivatives of quasi–coordinates.
- Can derive closed–form operator expression and computational algorithm for $\mathcal{C}(\theta,\nu)$.
- Similarities to rigid body equations of motion
  - $\mathcal{C}(\theta,\nu)$ does no work, i.e., $\nu^*\mathcal{C}(\theta,\nu) = 0$.
  - If $\epsilon = 0$, then $\|\nu\|$ is constant.
- This formulation leads to some simple control laws.

Useful for control, numerical integration, statistical mechanics etc.

Fixman Potential

The Fixman potential is needed in molecular dynamics simulations for correcting statistical biases

$$U_f \triangleq \log\{\det\{\mathcal{M}\}\}$$

Computing and using it has been an intractable problem for decades

$$\frac{\partial \log\{\det\{\mathcal{M}\}\}}{\partial \theta_i} = 2\,\mathrm{Trace}\left\{\mathcal{P}(i)\Upsilon(i)\widetilde{H}^*(i)\right\}$$

Torque from the Fixman potential

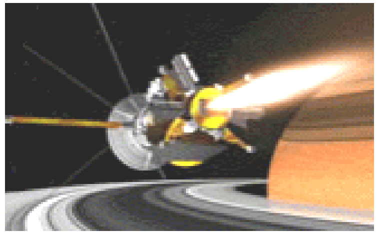
available from standard fwd dynamics

Explicit simple expression via SOA for previously <u>intractable</u> problem.

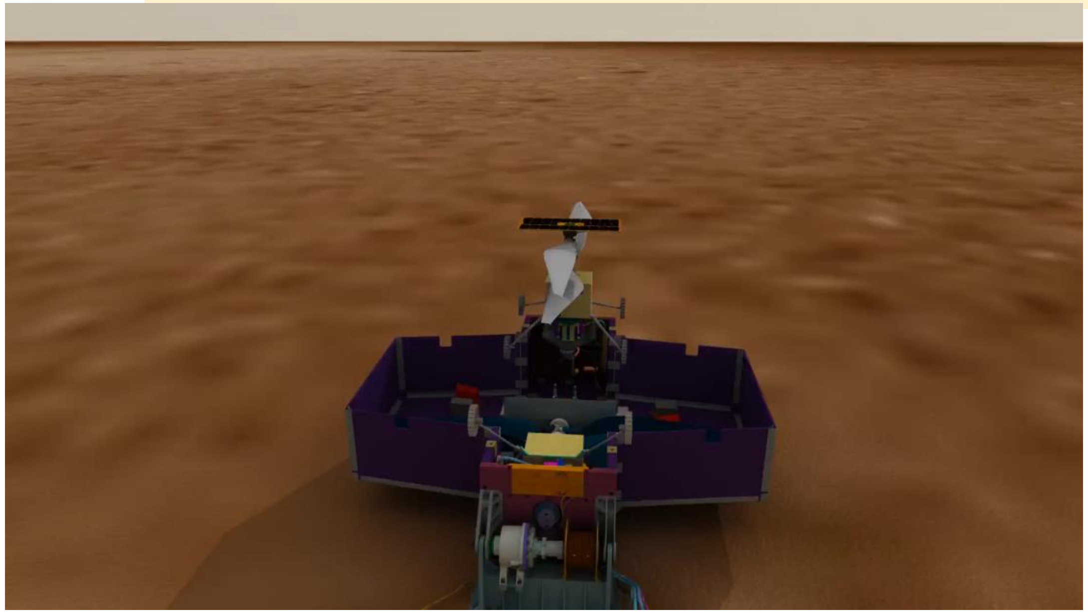
Applications using SOA DARTS software

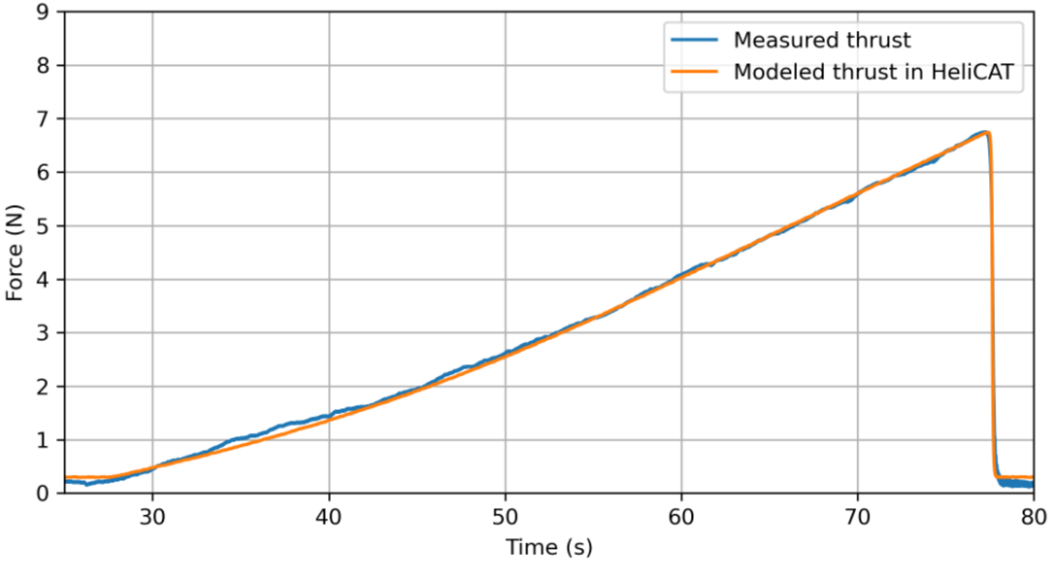
Aerospace/Robotics Context for Dynamics Modeling

*Develop engineering modeling & simulation capability that is **effective, scalable, reusable & sustainable** for complex life-cycle needs from aerospace to robotics*

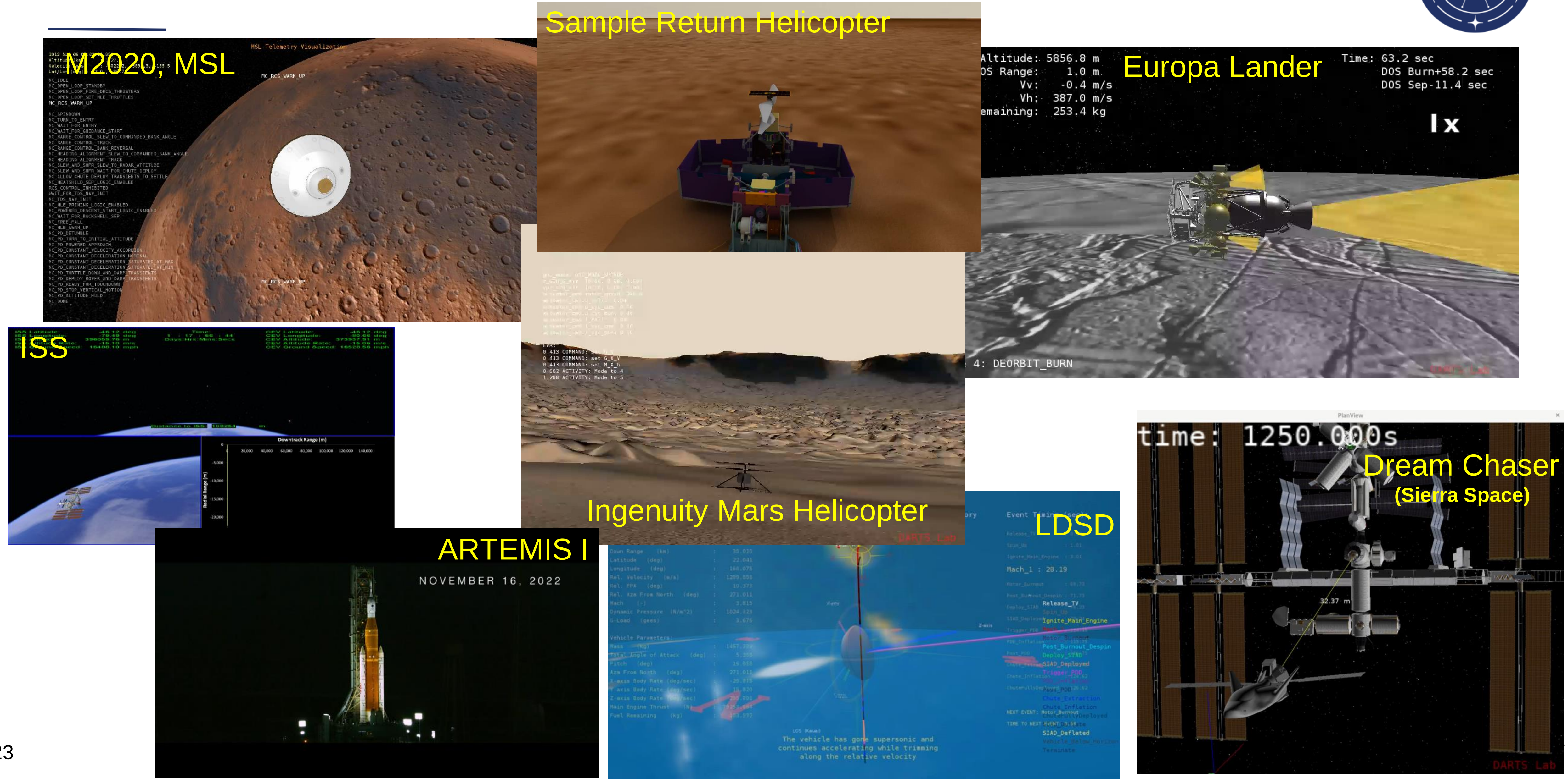


Need: General purpose, high fidelity, fast, versatile dynamics solution

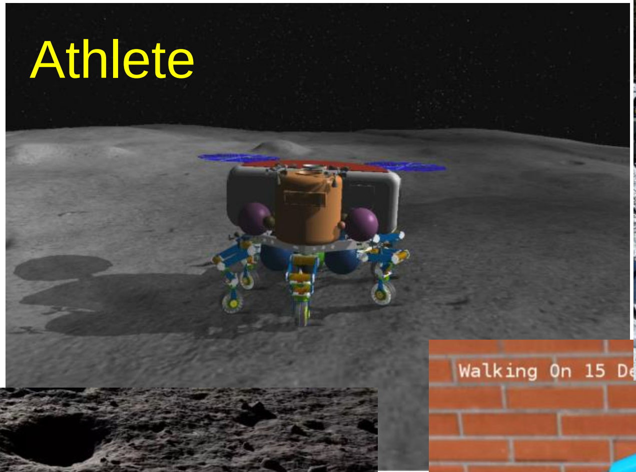


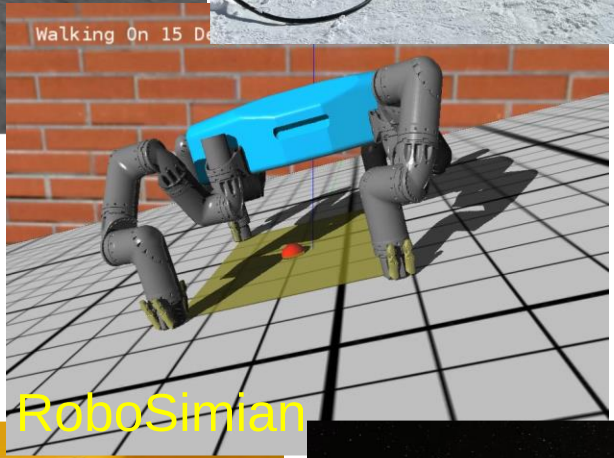


Flight Dynamics Simulations



Ground Vehicles & Robots





SOA Recap

SOA Analysis Impact

- **General:** Directly applies to large class of systems – rigid/flex, serial/arbitrary tree topology, small large
  - Also applies to closed-chain systems via constraint embedding (graph transformation)
- Provides a **unifying language** and **abstract framework** for tackling the complexities of dynamics
- **Can do old things better:** fast recursive algorithms (optimal, better numerical behavior)
- Can handle **changes to topology** easily: the structure remains the same, just continue with new topology!
- **Can do new things:** OSC techniques, mass matrix inverse, Fixman potential, diagonalized dynamics, inter-body forces

SOA Computational Impact

- Provides **fast O(N)** family of computational algorithms (optimal cost) – for simulation and embedded use
- **Structure based** – hence can accommodate changes to system topology
- **Expressive** – hence can accommodate large and diverse family of computations within single setting ideal for robotics and other applications
- Applies to broad class of **system topologies** – and hence can cover very broad needs
- Applies to rigid and **non-rigid bodies**, and hence allows for high fidelity physics
- Forms basis for **PyCraft workbench** for system level properties
- Operator structure carries over to **subgraphs** allowing the restriction of computational algorithms to smaller sub-systems as needed
- Math foundation allows **novel algorithms** as needed
- Provides a **model-based computational architecture** ideal for robotics – variable topology, constraints, task activities, control

Notional Plan for Discussion Group Sessions

Notional Plan for Sessions

- Focus initial sessions on **SOA foundations**
  - Use serial rigid body chain system to develop
    - SOA operators based equations of motion
    - Operator based analysis
    - Mass matrix factorization and inversion
    - Development of recursive O(N) algorithms
- Follow on - **Generalizations**
  - Step back and use graph theory ideas to generalize SOA methodology application
    - For branched/tree systems
    - Operational space inertia
    - For flexible body systems
    - For closed-chain systems (and constraint embedding)
- Optional follow on – **Selected topics**
  - Under actuated
  - Floating base systems
  - Sensitivities
  - Contact dynamics
  - Diagonalized dynamics
  - ….

SOA Foundations Track Topics
(serial-chain rigid body systems)

1. **Spatial (6D) notation** – spatial velocities, forces, inertias; spatial cross-product, rigid body transformations & properties; parallel axis theorem
2. **Single rigid body dynamics** – equations of motion about arbitrary frame using spatial notation
3. **Serial-chain kinematics** – minimal coordinate formulation, hinges, velocity recursions, Jacobians; first spatial operators; O(N) scatter and gather recursions
4. **Serial-chain dynamics** – equations of motion using spatial operators; Newton–Euler mass matrix factorization; O(N) inverse dynamics
5. **Mass matrix -** composite rigid body inertia; forward Lyapunov equation; mass matrix decomposition; mass matrix computation; alternative inverse dynamics
6. **Articulated body inertia -** Concept and definition; Riccati equation; alternative force decompositions
7. **Mass matrix factorization and inversion** – spatial operator identities; Innovations factorization of the mass matrix; Inversion of the mass matrix
8. **Recursive forward dynamics** – O(N) recursive forward dynamics algorithm; including gravity and external forces; inter-body forces identity