IAC-19.C1.5.8x50478

Forward Dynamics Algorithm for Origami-Folded Deployable Spacecraft Structures

JoAnna Fulton[a,*] and Hanspeter Schaub[b]

[a] *Graduate Research Assistant, Department of Aerospace Engineering Sciences, University of Colorado, 431 UCB, Colorado Center for Astrodynamics Research, Boulder, CO 80309-0431.*, joanna.fulton@colorado.edu
[*] Corresponding Author
[b] *Glenn L. Murphy Endowed Chair, Department of Aerospace Engineering Sciences, University of Colorado, 431 UCB, Colorado Center for Astrodynamics Research, Boulder, CO 80309-0431.*, hanspeter.schaub@colorado.edu

Abstract

A fast dynamics model that captures the deployment dynamics of self-actuated, origami-inspired, folded planar spacecraft structures is desired for design and verification applications. In this paper, a general simulation framework for numerically generating the equations of motion of any structure that complies with a set of pattern assumptions is designed and presented. The framework is built through expansion of the articulated body forward dynamics algorithm and the tree-augmented approach for closed-chain forward dynamics. These are multi-body dynamics approaches developed in the literature for complex robotic manipulator systems. Unique adaptations are required to address the highly constrained nature of a folding structure, and summaries of the resulting algorithms are provided. This solution is desirable due to the computational efficiency of the base algorithms and the ability to analyze multiple systems without reformulation of the core dynamics algorithm.

1. Introduction

The development and analysis of origami-inspired spacecraft structures is an active area of interest. Folded spacecraft structure concepts have been developed to stow planar structures with large area relative to the spacecraft bus size, such as solar arrays,[1] star occulters,[2] and antenna.[3] A central challenge for this class of deployable structure is understanding the deployment dynamics and designing the deployment actuation of the folded structure and spacecraft system. Previous research in the literature indicates active interest in this area. Recent studies developing folding structure concepts have adapted pre-existing software tools for dynamics analysis such as MathWorks SimScape Multibody,[4] or JPL's DARTs[5] simulation toolkit. However these tools are not necessarily optimized for processing the high volume of closed-chain constraints presented by a folding system. This point is highlighted in a previous folded structure study where the fold pattern was designed specifically to avoid the presence of closed chains entirely, trading structural rigidity for computational simplicity.[5] This motivates the need for a fast dynamics simulation approach to be developed. In this research, a custom method for modeling the dynamics using advanced multi-body techniques is developed. This method provides a fast simulation where complex hinges can be implemented at the folds to actuate the deployment.

This paper presents a general approach to the equations of motion of an origami folded spacecraft structure. The dynamics model is derived using the articulated body forward dynamics algorithm and the tree-augmented approach for closed-chain forward dynamics. These are multi-body approaches developed in the literature for complex robotic manipulator systems.[6] The applicability of this approach for this problem is demonstrated, and the algorithms are expanded for the unique case of folded deployable spacecraft structures. The robotics approach is desirable due to the computational efficiency of the algorithms and the ability to implement multiple types of complex internal hinge behavior without reformulation of the algorithms as the configuration and number of bodies in the system are changed. Reference 7 investigates following the Lagrangian approach and provides an initial understanding of the problem, but is found to be insufficient for scaling to multiple closed chain systems. An additional investigation discussed in Reference 8 determines the suitability of the presented approach for the simplest version of the origami problem, reviews relevant mathematics in the context of the origami-folded spacecraft structures problem, and outlines the need for the new work discussed in this paper.

A few key assumptions are ingrained in the construction of this approach. First, only origami fold patterns with repeating structure, such as the Miura and Scheel patterns,[9] are considered. These patterns share the common property of having no more than four panels meeting

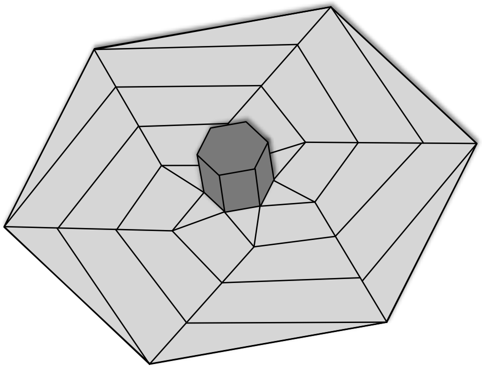


Fig. 1: Example structure concept: A spacecraft hub with a radially folding deployable structure.

at each vertex, as seen in Figures 3 and 4. It is assumed that any pattern modeled using this framework will only contain four panel vertices. Additionally, the fold lines of the pattern are treated as delineations between panels that are assumed to be rigid. Therefore, this approach is only appropriate for structures where the material of the fold hinge is sufficiently more flexible than the panels, and the panels are stiff enough that this assumption is valid. It is also assumed that only scleronomic holonomic constraints are enforced on the closed chain systems. Finally, it is assumed that the base-body of the structure is a free-flying spacecraft system, meaning the body is not rigidly attached to the ground and has six degrees of freedom. This assumption enables a computation shortcut in the constraint calculations and is consistent with the scope of the research.

2. Dynamics and Multi-body Systems Fundamentals

2.1 Spatial Vector Kinematics

The dynamics algorithms are structured using spatial vectors for computational and mathematical efficiency. Spatial vector algebra uses six dimensional representations of rigid body properties to capture both the rotational and linear components in a single expression. For example, a rigid body's orientation and position, referred to as the spatial coordinates $\boldsymbol{q}$ of frame $\mathcal{G}$ with respect to frame $\mathcal{F}$ is expressed as

$$\boldsymbol{q}(\mathcal{F},\mathcal{G}) = \begin{bmatrix} \boldsymbol{\sigma}(\mathcal{F},\mathcal{G}) \\ \boldsymbol{l}(\mathcal{F},\mathcal{G}) \end{bmatrix} \tag{1}$$

where $\boldsymbol{\sigma}$ is a three coordinate representation of orientation and $\boldsymbol{l}$ is the position vector in 3D Euclidean space. In this application, the spacecraft orientation is represented by the standard Modified Rodriguez Parameters (MRPs)[10] with shadow set switching to avoid geometric singularities. The spatial velocity is chosen as the angular rotation rates and the linear velocities of the body

$$\boldsymbol{\beta}(\mathcal{F},\mathcal{G}) = \begin{bmatrix} \boldsymbol{\omega}(\mathcal{F},\mathcal{G}) \\ \boldsymbol{v}(\mathcal{F},\mathcal{G}) \end{bmatrix} \tag{2}$$

Where the relative angular velocity is a non-integrable quasi-velocity, meaning it is not the time derivative of the spatial coordinates, and the notation $\boldsymbol{\omega}(\mathcal{F},\mathcal{G})$ denotes the angular velocity of frame $\mathcal{G}$ with respect to frame $\mathcal{F}$. The spatial orientations and spatial angular velocities are then related to each other using a linear transformation. For MRPs, this transformation is[10]

$$\dot{\boldsymbol{\sigma}} = \frac{1}{4}\left[(1-\boldsymbol{\sigma}^2)[I_{3\times3}] + 2[\tilde{\boldsymbol{\sigma}}] + 2\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathsf{T}}\right]\boldsymbol{\omega} = [B]^{\omega}\boldsymbol{\omega} \tag{3}$$

Then the full spatial transformation from spatial velocity to generalized coordinate derivatives is

$$\dot{\boldsymbol{q}} = [B]\boldsymbol{\beta} = \begin{bmatrix} [B]^{\omega} & 0_3 \\ 0_3 & I_3 \end{bmatrix}\boldsymbol{\beta} \tag{4}$$

2.2 Serial-chain ABFD Framework

A prominent dynamics algorithm developed for serial chains is presented in literature as the $\mathcal{O}(\mathcal{N})$ Articulated-Body Forward Dynamics (ABFD) algorithm developed independently by Featherstone[11] and Rodriguez,[12] and detailed in a unified manner by Jain.[13] Here $\mathcal{N}$ refers to the total number of velocity degrees of freedom in the system. The algorithm is developed to be appropriate for any multi-body robotic system that is treated as a network of serial-chain rigid bodies. The full derivation of the algorithm can be reviewed in the literature, but key formulations are repeated here to provide context to the derivations developed for spacecraft and deployable structure systems. In the articulated-body model, each of the rigid bodies down-chain of the current body being considered are treated as completely free with zero hinge force. Under this assumption, the articulated body inertia is calculated to represent those free bodies and a correction term is then developed to compensate for this assumption. This approach is in contrast to the composite body model, which treats the connected rigid bodies as fixed relative to each other, and uses a similar composite body inertia and correction term to derive the hinge force. However, the articulated body model is more appropriate for the forward dynamics problem. Additionally, the ABFD algorithm can be expanded to handle the multiple serial-chain branches of a tree-topology case.

The ABFD framework outlined by Jain[6] provides the basis of the version implemented here, with a few key adaptations that are described here as needed. The generalized spatial coordinates are chosen as hinge coordinates at the k^{th} hinge, or the k^{th} rigid body's outboard hinge frame, $\mathcal{O}_k$, orientation and position with respect to the

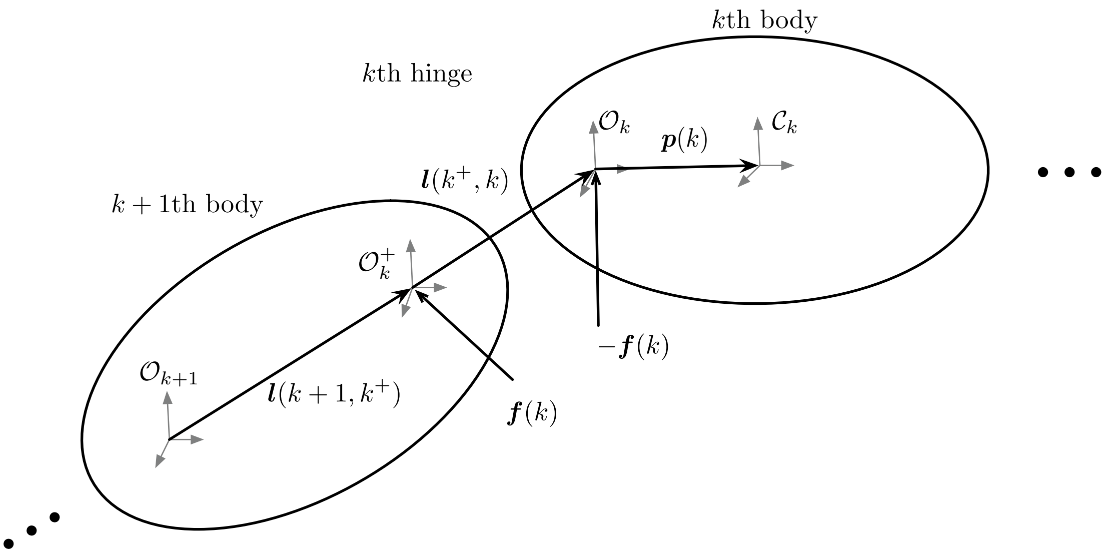


Fig. 2: Vector and frame notation between the $k+1^{\text{th}}$ and the k^{th} body.

$k+1$ rigid body's inboard hinge frame, $\mathcal{O}_k^+$, as illustrated in Figure 2,

$$\boldsymbol{q}(k) = \begin{bmatrix} \boldsymbol{\sigma}(\mathcal{O}_k^+, \mathcal{O}_k) \\ \boldsymbol{l}(\mathcal{O}_k^+, \mathcal{O}_k) \end{bmatrix} \tag{5}$$

and the generalized velocities are chosen as the hinge spatial velocities, taken as the time derivative with respect to the k frame

$$\boldsymbol{\beta}(k) = \begin{bmatrix} \boldsymbol{\omega}(\mathcal{O}_k^+, \mathcal{O}_k) \\ \boldsymbol{v}(\mathcal{O}_k^+, \mathcal{O}_k) \end{bmatrix} \tag{6}$$

For a given set of rigid bodies, these are collected in the full coordinate and velocity sets

$$q = \begin{bmatrix} \boldsymbol{q}(1) \\ ... \\ \boldsymbol{q}(k) \\ ... \\ \boldsymbol{q}(n) \end{bmatrix} \qquad \beta = \begin{bmatrix} \boldsymbol{\beta}(1) \\ ... \\ \boldsymbol{\beta}(k) \\ ... \\ \boldsymbol{\beta}(n) \end{bmatrix} \tag{7}$$

Where the tip of the chain is denoted as body 1 and the base body is denoted as body n. This leads to system equations of motion in the form

$$\mathcal{M}(q)\dot{\beta} + \mathcal{C}(q, \beta) = T \tag{8}$$

where $\mathcal{M}(q)$ is the full system mass matrix, $\mathcal{C}(q, \beta)$ contains the Coriolis contributions, and T is the vector of system generalized forces. The use of the quasi-velocities diverges from the assumptions implemented in Jain's text. In the forward dynamics problem, q, β and T are known quantities and the time derivative $\dot{\beta}$ is the desired quantity. Direct inversion of the mass matrix $\mathcal{M}$ is typically done for small order systems, but is a computationally expensive $\mathcal{O}(\mathcal{N}^3)$ matrix operation for an $\mathcal{N}$ degree of freedom problem. This becomes prohibitively slow for large DOF multi-body systems. The computational efficiency of the ABFD algorithm is achieved by applying the Innovations Operator Factorization of the mass matrix $\mathcal{M}$ and deriving an explicit and analytical expression of the inverse, $\mathcal{M}^{-1}$. The details of this factorization are left to the literature.[6] The dynamics are derived using body frame derivatives. The algorithm is set up in the following way. First, a recursive sweep that solves the velocities and Coriolis accelerations of the chain is run from the base body to the tip. Then, the articulated body inertias and corrections are solved for in a tip to base recursion. The final step is to do a base to tip recursion to solve for the body accelerations, yielding the system equations of motion.

3. Folded Structure Topology Processing

3.1 Graph Theory Applications

A system of hinge-connected rigid bodies can be described using graph theory by treating the rigid bodies as nodes and the hinges or fold lines as edges. This representation will aid in breaking down the complex system into a form that can be efficiently analyzed. The manner in which the system of nodes is connected determines the classification of the system. For a given graph, the node from which an edge leads from is designated the parent node and the node at the destination of that edge is referred to as the child node. A node with no parent is the root node. A parent node can have multiple child nodes, and if these nodes do not share edges within the graph, the graph is referred to as a tree topology. The basis of the dynamics algorithm discussed here is written to recursively solve for a serial chain of bodies, following the branch of a tree. At initial consideration, the closed-loop patterns of a folded spacecraft structure is a multiply connected graph where multiple child nodes span from a parent node and are interconnected, and there exist paths in the graph that lead back to a given node. The first step in modeling a folded spacecraft structure is to identify edges of the system to "cut" such that the bodies are segmented into a topology where there are no closed loops, also known as a tree topology. These cut edges must then be constrained to enforce the actual closed-chain topology.

3.2 Tree Topology of Planar Origami Patterns

The development and analysis of origami-inspired fold patterns appropriate for use in spacecraft structures is an active area of interest. A select number of patterns have received more study due to the clear applicability to spacecraft needs. The Miura pattern,[14] illustrated in Figure 3, is a highly efficient folding scheme with one theoretical degree of freedom that deploys linearly in dual directions and is thoroughly studied in the literature. Similarly, the Scheel pattern illustrated in Figure 4 is a radially wrapped pattern that is commonly studied for spacecraft structure applications.

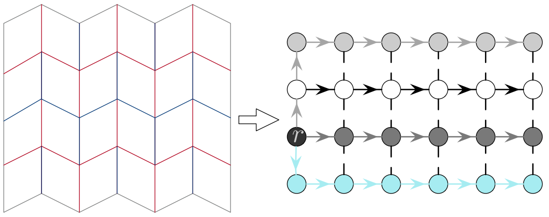


Fig. 3: Miura folding pattern and example system graph and cut edges where r denotes the root node.

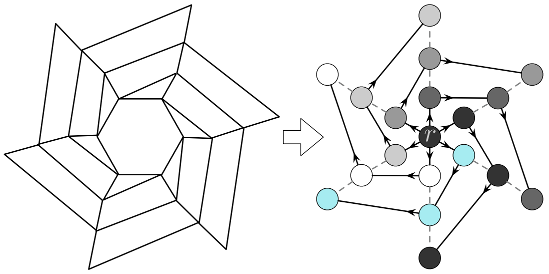


Fig. 4: Scheel folding pattern and example graph with cut edges where r denotes the root node.

Figures 3 and 4 also display example graph patterns for their corresponding origami pattern. The patterns are segmented such that a single root parent node spawns the serial chains of the origami pattern in a manner that itself displays a repeatable and expandable pattern. These serial chains are then constrained to each other at each adjacent node of their chains. For algorithm processing, it is assumed that the root node is always the free flying spacecraft body. The pattern is then defined through declaring each chain series and defining each set of constraint nodes. For this approach, these tree topologies are assumed to be cut and defined such that they form an organized grid , as clearly seen in the graph of Figure 3. A graph like the Scheel pattern in Figure 4 can be adapted to mimic a grid with minimal adaptation, as demonstrated in Figure 5. The chains of the structure are laided out like a grid, and the constraint nodes are defined as the dashed lines. This system will require an additional set of closure nodes defined between the chains on the edge of the grid (represented by a repeated set of the leftmost chain).

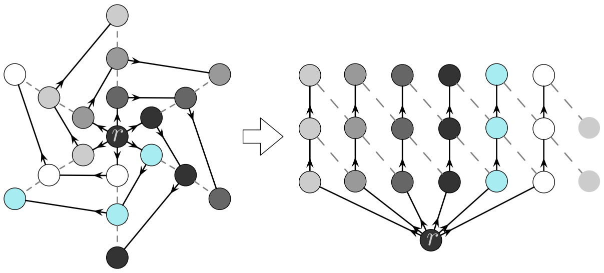


Fig. 5: Scheel folding pattern graph adapted to a grid format, where the closed grid is represented by the closure constraints on the repeated left edge chain's nodes.

3.3 Constraints for Grid Adapted Tree Topologies

A given panel can have more than one constraint node, as is present where there are three or more chains in a pattern. The cut kinematic chains are defined by recording the chain sequence in terms of the named bodies in the chain from tip to base in the chain matrix κ as

$$\kappa_a = \begin{bmatrix} a(1) & ... & a(n_a) \end{bmatrix} \tag{9}$$

For reference, n_p are the number of constraint node pairs or number of implemented constraints, n_c are the total number of constraint nodes, n_b are the number of rigid bodies in the system, and n_h are the number of chains in the cut tree topology. Then the constraints information is stored in the $n_p \times 2$ constraint node matrix, Γ, containing the constraint node pair designations.

$$\Gamma = \begin{bmatrix} a(1) & b(1) \\ ... & \\ a(n_{b_h1}) & b(n_{b_h2}) \end{bmatrix} \tag{10}$$

For a given set of bodies connected in a grid format that does not close onto itself, like the Miura pattern, the total number of constraints needed to adapt the set to a tree-topology system is summarized by Equation 11 and the total number of constraint nodes on the system can be predicted by Equation 12, assuming constraint nodes are unique to a constraint pair.

$$n_p = (n_h - 1)\left(\frac{n_b}{n_h} - 1\right) \tag{11}$$

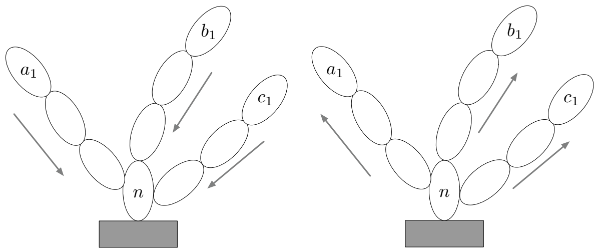
$$n_c = n_b - n_h \tag{12}$$

These are needed for constraint generating algorithms. Similar calculations can be derived for radially closed patterns by simply including the additional closure nodes.

3.4 Automated and Recursive Generation of Rigid Body Properties

For patterns that are built with repeating subgraphs, such as the Miura-Ori, populating the geometric definitions for the chain nodes and constraint nodes of each rigid body panel can be automated by using a uniform reference frame convention. For the numerical simulations presented here, the repeated panels of the Miura ori pattern are populated from two stock reference bodies. These provide the relative position and orientation of defined nodes of the bodies, the mass and inertia properties, and the relative position and orientation of nodes related to the inbound body on the chain. These last properties are provided through a base to tip recursive calculation.

4. Multi-body Algorithm Expansions for Folded Structure Tree Topologies



(a) Gather recursion (b) Scatter recursion

Fig. 6: illustrations of multiple serial chains algorithm processing schemes

The recursive forward dynamics algorithms from the literature are expanded to accommodate the generalized tree topology framework needed to handle folded structures. These expansions are suggested in the literature but are not explicitly presented. The expansions implemented by the author are now summarized as follows. Each tip-to-base recursion formula must be converted to a tips-to-base gather recursion. Each chain outbound from a branching node is computed recursively and summed to the branching node. In Figure 6 the branching node is denoted as n and the chains are labeled a, b, and c, with subscript 1 denoting the tip node of each respective chain. Conversely, each base-to-tip recursion formula must be converted to a base-to-tips scatter recursion. The base body node behavior is calculated first and propagated through each chain. These gather and scatter algorithms present an opportunity for parallel computation, where each chain recursion can be done simultaneously apart from the final gathering or initial scattering calculations.

The following algorithms are provided as supplemental material to the literature,[6] and therefore the mathematical derivations and significance of the variables and operators are not discussed. Algorithm 1, 2, and 3 are executed sequentially. Algorithm 1 summarizes the first recursion in the dynamics framework and calculates the kinematics and velocities of the bodies. Algorithm 2 demonstrates the gather operation for the spatial inertia and spatial body forces recursions on the system. Finally, Algorithm 3 summaries adaptation for the spatial accelerations of the cut chains. For gather recursions, the algorithm expansions for multiple chains assume that for a given branching node, the chains branching from that node are processed before the chain that the branch is a member of. Similarly, for scatter recursions, chains containing branching nodes are processed before the branch chains. This ordering is contained in the

```
Result: kinematics and velocities for each chain's
        bodies
$\boldsymbol{V}(n+1) = \boldsymbol{0}$
calculate all kinematics and velocities for the root
body, $n$
for $m = 1$ to $n_h$ do
    for $j = n_b(m) - 1$ to $1$ do
        set $k$ to be the $m(j)$th body
        set $l$ to be the next body, $m(j+1)$ in the $m$
        chain

            $\phi(k^+, k) = f(\boldsymbol{q}(k), \text{body geometry})$
            $\Delta\boldsymbol{v}(k) = H^{\intercal}(k)\boldsymbol{\beta}(k)$
            $\Delta\boldsymbol{v}^{\omega}(k) = [\Delta\boldsymbol{\omega}(k), 0, 0, 0]$
            $\boldsymbol{V}(k) = \phi^{\intercal}(k^+, k)\boldsymbol{V}(l) + \Delta\boldsymbol{v}(k)$
            $\boldsymbol{a}(k) = -\tilde{\Delta\boldsymbol{v}}^{\omega}(k)(\boldsymbol{V}(k) - \boldsymbol{\beta}(k))$
            $\boldsymbol{b}(k) = \boldsymbol{V}\bar{(}k)M(k)\boldsymbol{V}(k)$
            $\boldsymbol{T}(k) = f(\boldsymbol{q}(k), H(k))$

    end
end
```

Algorithm 1: articulated body spatial velocities algorithm for multiple chains

The variables referenced in this paper are consistent with that in the literature[6] and previous work.[8] The index n references the base body and $n+1$ references the inertial frame. $\boldsymbol{V}(k)$ is the spatial velocity of a given body k, $\phi(k^+, k)$ is the spatial transformation matrix, $\Delta\boldsymbol{v}(k)$ is the relative spatial velocity of a body with respect to the next body in the chain, $\boldsymbol{a}(k)$ is the Coriolis spatial acceleration, $\boldsymbol{b}(k)$ is the gyroscopic spatial acceleration, $\boldsymbol{T}(k)$ is the internal spatial force acting at the hinge, and $H(k)$ is the hinge map matrix that defines the configuration dependence of the hinge behavior and maps the hinge velocities to the generalized spatial velocities of the body.

Algorithm 2 contains the sequence for recursively calculating the articulated body spatial inertia $P(k)$ and the articulated body forces $\boldsymbol{\zeta}(k)$. This requires the definition and reference of several spatial operators, and these can be reviewed in the literature. The articulated body spatial inertia represents the inertia of all the bodies connected outbound of a given body, and similarly the articulated body force represents the cumulative body force of all the bodies outbound in the chain. The superscript $^+$ denotes the transition of a value to the inboard reference frame defined in Figure 2. The spatial inertia of just body k is denoted $M(k)$, $D(k)$ is the articulated body hinge inertia, $G(k)$ is the articulated body Kalman gain operator, $\bar{\tau}(k)$ is the complement of the articulated body projection operator, $\boldsymbol{\epsilon}(k)$ is the articulated body inertia innovations generalized force, and $\boldsymbol{\eta}(k)$ is the articulated body inertia innovations generalized acceleration. For the gather recursion, n_r is the number of root bodies in the cut tree topology, or bodies that have chains branching from them.

Algorithm 3 provides the accelerations of the generalized coordinates, $\dot{\boldsymbol{\beta}}(k)$, and the spatial accelerations, $\boldsymbol{\alpha}(k)$, from the spatial operators listed in the previous algorithms. At this point, the equations of motion for the cut tree-topology of the structure is obtained, and the constraints for enforcing the closed-chain topology must be implemented.

5. Closed-Chain Forward Dynamics

As discussed in Section 3., capturing the closed-chain behavior is achieved by cutting an edge of a closed-chain system and treating each leg of the cut as an open serial chain, emulating a tree topology. Then the cut edges are treated as motion constraints imposed on the free dynamics of the tree. There are several approaches to enforcing the closure constraints. The augmented approach compensates for the cut edge by including a correction acceleration, resulting in additional motion constraint equations and a non-minimal coordinate set. This approach requires the use of differential-algebraic equation integrators and faces issues with error drift that must be compensated for with error control techniques. The direct approach uses matrix solvers and absolute coordinates, resulting in a much larger system and greater computational complexity. This approach also shares similar issues as the augmented approach, and therefore is not considered, as the augmented approach is more desirable for this application. A new technique that provides a minimal coordinate set is the constraint embedding approach. In this approach, the non-tree graph is transformed into a tree topology by aggregating the closed-chain struc-

```
Result: serial chain articulated body spatial inertia
        and articulated body forces for each chain's
        bodies
$P^+(0) = \mathbf{0}$, $\zeta^+(0) = \mathbf{0}$, $T(0) = \mathbf{0}$, $\bar{\tau}(0) = \mathbf{0}$
for $m = 1$ to $n_h$ do
    for $j = 1$ to $n_b(m) - 1$ do
        set $k$ to be the $m(j)$th body
        set $i$ to be the previous body, $m(j-1)$ in the
        $m$ chain
            $P(k) = \phi(k,i)P^+(i)\phi^\intercal(k,i) + M(k)$
            $D(k) = H(k)P(k)H^\intercal(k)$
            $G(k) = P(k)H^\intercal(k)D^{-1}(k)$
            $\bar{\tau}(k) = \boldsymbol{I} - G(k)H(k)$
            $P^+(k) = \bar{\tau}(k)P(k)$
            $\boldsymbol{\zeta}(k) = \phi(k,i)\boldsymbol{\zeta}^+(i) + P(k)\boldsymbol{a}(k) + \boldsymbol{b}(k)$
            $\boldsymbol{\epsilon}(k) = \boldsymbol{T}(k) - H(k)\boldsymbol{\zeta}(k)$
            $\boldsymbol{\eta}(k) = D^{-1}(k)\boldsymbol{\epsilon}(k)$
            $\boldsymbol{\zeta}^+(k) = \boldsymbol{\zeta}(k) + G(k)\boldsymbol{\epsilon}(k)$
    end
end
for $n = 1$ to $n_r$ do
    initialize $P(n) = M(n)$
    for $m = 1$ to $n_h$ do
        set $j$ to be the node of chain $m$ connecting to
        that chain's root body
            $P(n) = P(n) + \phi(n,j)P^+(j)\phi^\intercal(n,j)$
    end
    calculate $D(n)$, $G(n)$, $K(n)$, and $\psi(n^+,n)$
    initialize $\boldsymbol{\zeta}(n) = P(n)\boldsymbol{a}(n) + \boldsymbol{b}(n)$
    for $m = 1$ to $n_h$ do
        set $j$ to be the node of chain $m$ connecting to
        the root body
            $\boldsymbol{\zeta}(n) = \boldsymbol{\zeta}(n) + \phi(n,j)\boldsymbol{\zeta}^+(j)$
    end
end
calculate $\boldsymbol{\epsilon}(n)$ and $\boldsymbol{\eta}(n)$
```

Algorithm 2: articulated body spatial inertia and articulated body forces algorithm for multiple chains

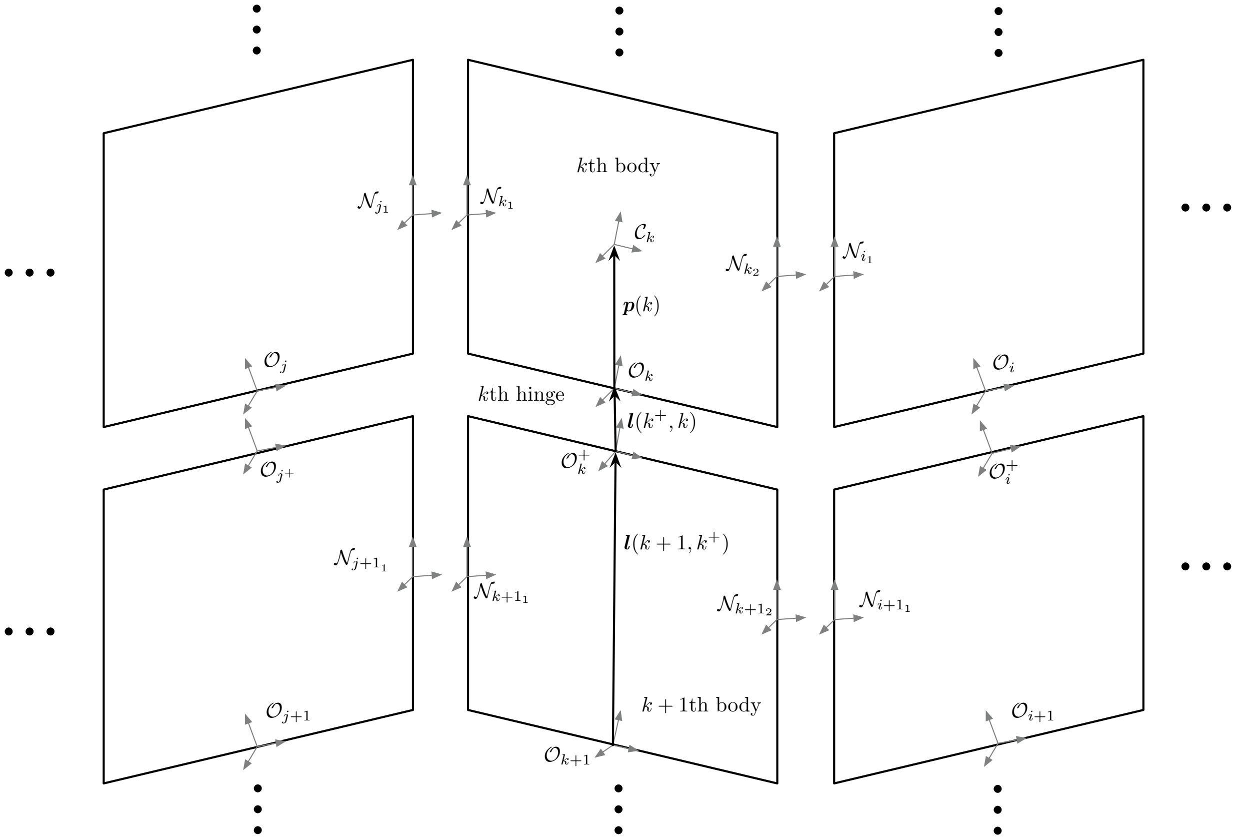


Fig. 7: Vector and frame notation between multiple serial chains subject to multiple closure constraints.

```
Result: serial chain relative coordinate accelerations
        and spatial accelerations for multiple chains
$\boldsymbol{\alpha}(n+1) = \mathbf{0}$
calculate accelerations for the root body, $n$
for $m = 1$ to $n_h$ do
    for $j = n_b(m) - 1$ to $1$ do
        set $k$ to be the $m(j)$th body
        set $l$ to be the next body, $m(j+1)$ in the $m$ chain
            $\boldsymbol{\alpha}^+(k) = \phi^\mathsf{T}(k^+, k)\boldsymbol{\alpha}(l)$
            $\dot{\boldsymbol{\beta}}(k) = \boldsymbol{\eta}(k) - G^\mathsf{T}(k)\boldsymbol{\alpha}^+(k)$
            $\boldsymbol{\alpha}(k) = \boldsymbol{\alpha}^+(k) + H^\mathsf{T}(k)\dot{\boldsymbol{\beta}}(k) + \boldsymbol{a}(k)$
    end
end
```

Algorithm 3: articulated body spatial accelerations and general coordinate accelerations algorithm

tures of the topology into a representative node. This is suitable for systems with a clear tree-like structure surrounding the closed-chain elements. The folded structures of interest contain multiply dependent systems of closed loops, as demonstrated in Figures 3 and 4, and therefore this approach is not well suited to the problems of interest and is not currently considered. Therefore, the tree-augmented approach is selected and developed for the general origami-folded spacecraft structure. Custom algorithms are developed specifically to handle the large number of closed-chain constraint calculations across multiple serial chains, as depicted in Figure 7, and are presented.

5.1 Tree-Augmented Approach to Closed Chain Structures

Implementing the correction terms to account for the motion constraints is captured in the system equations of motion by introducing the Lagrange Multipliers,[6] denoted as λ, to represent the constraint forces. Additionally, a new set of equations must be considered to include the constraint expression. The generalized acceleration is redefined as

$$\dot{\beta} = \dot{\beta}_f + \dot{\beta}_c \qquad (13)$$

where $\dot{\beta}_f$ are the free unconstrained accelerations and $\dot{\beta}_c$ are the correction accelerations. The correction acceleration is derived from the constraint expression, and is expressed in terms of global system spatial operators as

$$\dot{\beta}_c = [I - H\phi K]D^{-1}H\phi\mathcal{B}Q^{\intercal}\lambda \tag{14}$$

where H is the global hinge map matrix, ϕ is the $6n_b \times 6n_b$ global spatial transformation matrix, K is the $6n_b \times 6n_b$ spatial operator referred to in literature as the shifted Kalman gain operator, D is the $6n_b \times 6n_b$ articulated body hinge inertia, and $\mathcal{B}$ is th $6n_b \times 6n_c$ node pick-off operator that identifies relevant nodes on the bodies of the system. Q is the $n_{\text{CDOF}} \times 6n_b$ constraint matrix that defines the constrained degrees of freedom between nodes where n_{CDOF} are the total number of constrained node pair degrees of freedom. For a node that is rigidly constrained to another, the corresponding entry in Q is a 6×6 identity matrix. Finally, λ is the $n_{\text{CDOF}} \times 1$ Lagrange Multipliers. These are defined for loop constraints as

$$\lambda = -[Q\underline{\Lambda}Q^{\intercal}]^{-1}\ddot{\boldsymbol{d}} \tag{15}$$

where $\underline{\Lambda}$ is the operation space compliance matrix

$$\underline{\Lambda} = \mathcal{B}^{\intercal}\Omega\mathcal{B} \tag{16}$$

and Ω is the extended operational space compliance matrix. Additionally, $\ddot{\boldsymbol{d}}(\beta, t)$ is the derivative of a Pfaffian form constraint equation, $\dot{\boldsymbol{d}}(\beta, t)$. A holonomic coordinate based constraint equation, $\boldsymbol{d}(q, t)$, is not considered due to the complications of re-expressing the constraint in terms of non-integrable quasi-velocities. This use of rate-based constraints will introduce error control concerns. For the enforcement of closure constraints for cut edges that do not change over time, and therefore are scleronomic holonomic constraints, the Pfaffian form constraint equation is expressed as

$$\ddot{\boldsymbol{d}} = Q\boldsymbol{\alpha}_{n_c,f} \tag{17}$$

where $\boldsymbol{\alpha}_{n_c,f}$ is the free, unconstrained spatial acceleration of the constraint node bodies.

The diagonal terms of Ω are computed directly using a recursion for free-flying system from the literature[6] as

$$\Omega(k,k) = \Upsilon(k) = [P(k) + S(k)]^{-1} \tag{18}$$

for all bodies k, where $\Upsilon(k)$ is known as the operational space compliance kernel. The compliance properties of a free-flying system enables these terms to be expressed in terms of the articulated body inertia and what is referred to as the dual articulated body inertia, $S(k)$. The difference between these two inertias is simply whether the base or tip body is treated as hinged to free inertial space, and each is calculated using the recursive algorithm defined in

```
Result: serial chain dual spatial inertia for each
        chain's bodies
$P^+(0) = \mathbf{0}$, $\bar{\tau}(0) = \mathbf{0}$
for $m = 1$ to $n_h$ do
    for $j = n_b(m) - 1$ to $1$ do
        set $k$ to be the $m(j)$th body
        set $l$ to be the next body, $m(j+1)$ in the $m$
        chain

            $S^+(k) = \phi^{\intercal}(i,k)(S(i) + M(i))\phi(i,k)$
            $D_{dl}(k) = H(k)S^+(k)H^{\intercal}(k)$
            $G_{dl}(k) = -S^+(k)H^{\intercal}(k)D_{dl}^{-1}(k)$
            $\bar{\tau}_{dl}(k) = \boldsymbol{I} - G_{dl}(k)H(k)$
            $S(k) = \bar{\tau}_{dl}(k)S^+(k)$

    end
end
```

Algorithm 4: articulated body dual spatial inertia algorithm for multiple chains

Algorithm 4. Spatial operator expressions are identical to those defined for Algorithm 2, with subscript dl indicating the dual inertia distinction.

The diagonal terms of Ω are computed directly using Equation 18. Then, where for two bodies k and j

$$\Omega(k,j) = \Omega(j,k)^{\intercal} \tag{19}$$

the off diagonal terms are computed by propagating through the root node, r, as

$$\Omega(j,k) = \Omega(j,j)\psi(j,r)\psi(r,k) \tag{20}$$

where $\psi(r,k)$ represents the articulated body transformation matrix and is calculated from the articulated body projection operator and the spatial transformation matrix $\psi(r,k) = \phi(r,k)\bar{\tau}(k)$. This property enables all of the operational space matrix terms to be computed from the diagonal terms, and in turn from the recursive articulated body inertias.

To populate the operational space matrix, $\underline{\Lambda}$, only the diagonal terms and cross-diagonal terms of the extended operational space matrix that correspond with the constraint nodes are needed due to the structure of $\mathcal{B}$. The node pick off operator $\mathcal{B}$ is a $6n \times 6n_c$ sparse spatial operator matrix that contains the spatial rigid body transformation matrix from a given body frame to the constraint node frame at that body's row and that node's column, for example $\phi^{\intercal}(k, \mathcal{N}_{k_2})$. Then $\underline{\Lambda}$ is populated using the shortcut expression provided by[6]

$$\underline{\Lambda}(\mathcal{N}_{k_2}, \mathcal{N}_{j_1}) = \phi^{\intercal}(k, \mathcal{N}_{k_2})\Omega(k,j)\phi(j, \mathcal{N}_{j_1}) \tag{21}$$

when using the node frame definitions demonstrated in Figure 7, and k is shorthand for $\mathcal{O}_k$.

This summarizes the mathematical background needed to generate the node constraint expressions, and the steps are implemented in Algorithm 5. The Lagrange multipliers λ are now interpreted into a constraint force that is applied to the rigid body system. The constraint force is defined as

$$f_c = -Q^{\intercal}\lambda \tag{22}$$

and is converted into constraint correction body force at each hinge degree of freedom through Algorithm 6, an algorithm that is formulated for any general case of multiple serial chains subject to multiple constraints. The constraint correction body forces are then used to calculate the constraint correction accelerations $\dot{\beta}_c$ in Algorithm 7. Here, $\zeta(k)$ and $\eta(k)$ are not related to those used in previous algorithms but are representing similar roles.

```
Result: internal forces due to closure constraints
  1. Generate the extended operational space compliance
     matrix $\Omega$ using Equation 20 for all constraint node
     bodies in the system
  2. Project this into the operational space compliance
     matrix $\underline{\Lambda}$ for all constraint nodes using Equation 21
  3. Calculate the Lagrange multipliers $\lambda$ in Equation 15
  4. Express the constraint force $f_c$ using Equation 22
```

Algorithm 5: Converting constraint node information to constraint forces

5.2 *Origami-Folded Deployable Spacecraft Structure Algorithm*

The complete algorithm for solving the dynamics of a set of rigid bodies subject to any number of closure constraints is summarized in Figure 8. The connections between information obtained and required at multiple steps in the algorithm are depicted by arrows. This algorithm is written to only apply to multiple closed-chain constraints within a free-flying spacecraft system, but can be applied to any system that resembles the folding-structure topologies described in this paper. While the ABFD framework provides an $\mathcal{O}(\mathcal{N})$ solution to the free serial chain dynamics in Algorithm 3, the overall computational efficiency of the Algorithm summarized in 8 is less. The matrix inversion of Equation 15 represents an $\mathcal{O}({n_c}^3)$ operation that dramatically slows down the simulation as more constraint nodes of the origami pattern are introduced. Unfortunately the non-square structure of the Q matrices and the fully populated structure of the $\underline{\Lambda}$ matrix indicates that further decomposition of the matrices will not yield a convenient property as the Innovations Factorization Method lends to the articulated body model. Therefore, the overall computation efficiency of this algorithm is $\mathcal{O}({n_c}^3)$.

```
Result: constraint correction body forces for multiple
        serial chains
$\boldsymbol{\alpha}(n+1) = \mathbf{0}$
for $m = 1$ to $n_h$ do
    for $j = 1$ to $n_b(m) - 1$ do
        set $k$ to be the $m(j)$th body
        if $j = 1$ then
            $\boldsymbol{\zeta}(k) = -\phi(\mathcal{B}_k, \mathcal{N}_k^i) f_c(k)$
        else
            set $i$ to be the previous body, $m(j-1)$ in
            the $m$ chain
            $\boldsymbol{\zeta}(k) = \psi(k,i)\boldsymbol{\zeta}(i) - \phi(\mathcal{B}_k, \mathcal{N}_k^i) f_c(k)$
        end
        $\boldsymbol{\eta}(k) = -D^{-1}(k)H(k)\boldsymbol{\zeta}(k)$
    end
end
initialize $\boldsymbol{\zeta}(n) = \mathbf{0}$
for $m = 1$ to $n_h$ do
    set $j$ to be the node of chain $m$ connecting to the
    root body
    $\boldsymbol{\zeta}(n) = \boldsymbol{\zeta}(n) + \phi(n,j)\boldsymbol{\zeta}^{+}(j)$
end
```

Algorithm 6: Constraint correction body forces for multiple serial chains

```
Result: constraint correction accelerations for
        multiple serial chains
$\boldsymbol{\alpha}(n+1) = \mathbf{0}$
calculate accelerations for the root body, $n$
for $m = 1$ to $n_h$ do
    for $j = n_b(m) - 1$ to $1$ do
        set $k$ to be the $m(j)$th body
        set $l$ to be the next body, $m(j+1)$ in the $m$
        chain
        $\lambda(k) = \psi^{\intercal}(l,k)\lambda(l) + H^{\intercal}(k)\boldsymbol{\eta}(k)$
        $\dot{\boldsymbol{\beta}}_c(k) = \boldsymbol{\eta}(k) - K^{\intercal}(k)\lambda(l)$
    end
end
```

Algorithm 7: Constraint correction accelerations for multiple serial chains

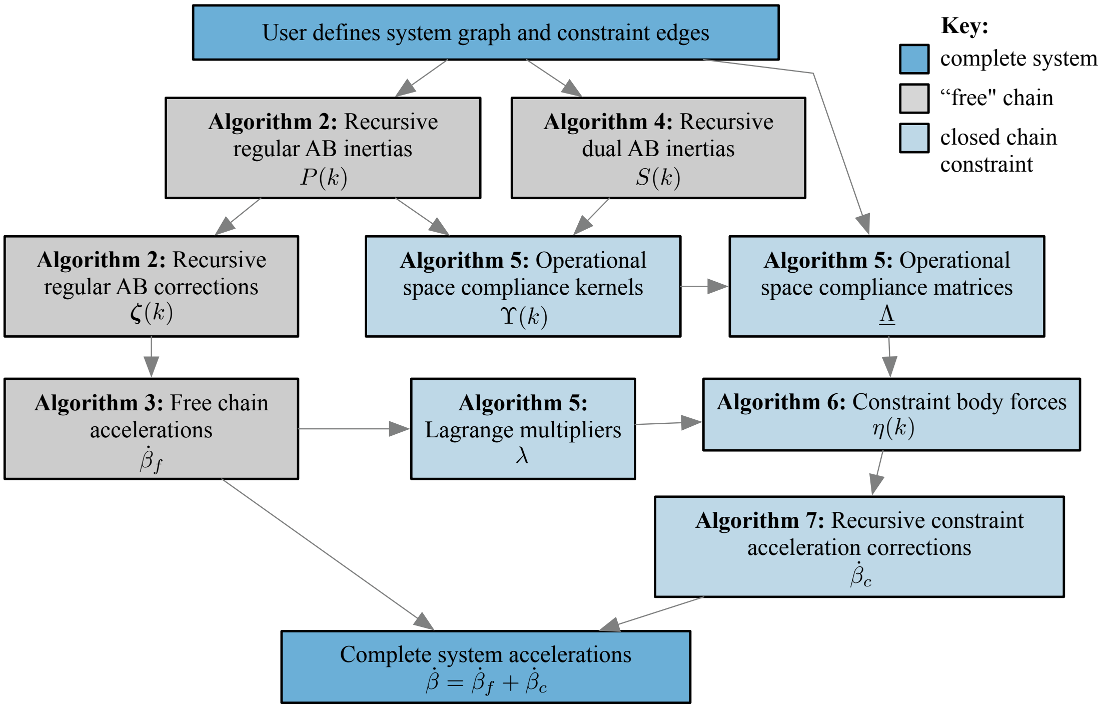

User defines system graph and constraint edges
Key:
complete system
"free" chain
closed chain constraint
Algorithm 2: Recursive regular AB inertias $P(k)$
Algorithm 4: Recursive dual AB inertias $S(k)$
Algorithm 2: Recursive regular AB corrections $\zeta(k)$
Algorithm 5: Operational space compliance kernels $\Upsilon(k)$
Algorithm 5: Operational space compliance matrices $\underline{\Lambda}$
Algorithm 3: Free chain accelerations $\dot{\beta}_f$
Algorithm 5: Lagrange multipliers λ
Algorithm 6: Constraint body forces $\eta(k)$
Algorithm 7: Recursive constraint acceleration corrections $\dot{\beta}_c$
Complete system accelerations $\dot{\beta} = \dot{\beta}_f + \dot{\beta}_c$

Fig. 8: Diagram of full closed-chain dynamics algorithm flow.

6. Conclusions and Future Work

A self-actuated folded deployable spacecraft structure presents a novel modeling challenge due to free-flying spacecraft dynamics coupled with a complexly constrained multibody system. An approach that blends several articulated body-derived robotics dynamics algorithms together is presented to address the multiple closed-chain folded structure problems. The topology is studied and interpreted for dynamics analysis using graph theory, and a simple map fold pattern of 9 bodies is analyzed for algorithm demonstration. Origami-inspired folding topologies with large number of bodies are shown to have algorithm gains for recursively calculated loop constraints. The articulated body forward dynamics algorithm is outlined as the basis for the approach, and derivations that generalize the ABFD algorithm to the spacecraft folded deployable structure scenario are provided. The tree augmented approach is developed for any grid formatted spacecraft structure. It is found that this approach provides significant value over the Lagrangian approach or Kane's equations. This is due to the computation gains of the recursive structure of the equations of motion and that the algorithm provides a framework for working with a high volume of rigid bodies and rigid body constraints. Future work will focus on applying the tool to complex folded structure design studies and on integrating complex hinge force models for deployment actuation.

7. Acknowledgments

The author acknowledges the NASA Space Technology Research Fellowship program and the Ann and H.J. Smead Department of Aerospace Engineering Sciences Smead Fellowship program for their generous support of this research.

REFERENCES

[1] Sungeun K. Jeon and Joseph N. Footdale. Scaling and design of a modular origami solar array. In *AIAA SciTech Spacecraft Structures Conference*, 2018.

[2] David Webb, Brian Hirsch, Vinh Bach, Jonathan Sauder, Case Bradford, and Mark Thomson. Starshade mechanical architecture and technology effort. In *3rd AIAA Spacecraft Structures Conference*, 2016.

[3] Daniel Kling, Sungeun Jeon, and Jeremy Banik. Novel folding methods for deterministic deployment of common space structures. In *3rd AIAA Spacecraft Structures Conference*, 2016.

[4] Nathan A. Pehrson, Samuel P. Smith, Daniel C. Ames, Spencer P. Magleby, and Manan Arya. Self-deployable, self-stiffening, and retractable origami-based arrays for spacecraft. In *AIAA SciTech Spacecraft Structures Conference*, San Diego, CA, 7-11 January 2019.

[5] Marco B. Quadrelli, Adrian Stoica, Michel Ingham, and Anubhav Thakur. Flexible electronics-based transformers for extreme environments. In *AIAA SPACE 2015 Conference and Exposition*, 2015.

[6] Abhinandan Jain. *Robot and Multibody Dynamics*. Springer Science+Business Media, LLC, 2011.

[7] JoAnna Fulton and Hanspeter Schaub. Dynamic modeling of folded deployable space structures with flexible hinges. In *2017 AAS/AIAA Astrodynamics Specialist Conference*, Stevenson, WA, 2017.

[8] JoAnna Fulton and Hanspeter Schaub. Closed-chain forward dynamics modeling of a four-panel folding spacecraft structure. In *International Astronautical Congress*, Bremen, Germany, Oct 1-5 2018.

[9] S.D. Guest and Sergio Pellegrino. Inextensional wrapping of flat membranes. In *Proceedings of the First International Seminar on Structural Morphology*, 1992.

[10] Hanspeter Schaub and John L. Junkins. *Analytical Mechanics of Space Systems*. American Institute of Aeronautics and Astronautics, Inc., Reston, Virginia, 20191-4344, 4th edition, 2018.

[11] Roy Featherstone. The calculation of robot dynamics using articulated-body inertias. *The International Journal of Robotics Research*, 2(1):13–30, Spring 1983.

[12] Guillermo Rodriguez. Recursive forward dynamics for multiple robot arms moving a common task object. *IEEE Transactions on Robotics and Automation*, 5(4):510–521, August 1989.

[13] Abhinandan Jain. Unified formulation of dynamics for serial rigid multibody sytems. *Journal of Guidance, Control, and Dynamics*, 14(3):531–542, May-June 1991.

[14] Koryo Miura. Folding a plane- scenes from nature technology and art, symmetry. In *Symmetry of structure, interdisciplinary symposium*, Budapest, Hungary, August 13-19 1989.