Improving Contact-Rich Robotic Simulation with Generalized Rigid-Body Dynamics Algorithms

Matthew Chignoli\textsuperscript{1}, Nicholas Adrian\textsuperscript{2}, Sangbae Kim\textsuperscript{1}, and Patrick M. Wensing\textsuperscript{2}

Abstract—We propose a novel approach for generalizing rigid-body dynamics algorithms to handle complex sub-mechanisms that are popular in robotic systems. The approach is based on the unification of local constraint embedding and spatial vector algebra, which allows for the development of compact and computationally efficient algorithms for simulating contact-rich robots. We derive the approach intuitively using the Gauss Principle of Least Constraint. Overall, our approach provides a valuable foundation for simulating and controlling complex robotic systems with traditionally difficult-to-simulate designs such as geared motors, differential drives, and four-bar mechanisms.

I. INTRODUCTION

In recent years, the field of humanoid robot design has been dominated by a trend of complex limb designs that aim to expand the robot’s range of motion, minimize the inertia of the limbs, and smoothly handle impacts with the environment. Designs with sub-mechanisms such as coupled belt drives \cite{1}, differential drives \cite{2}, \cite{3}, and four bar linkages \cite{4} have emerged as popular options to meet these design criteria. Unfortunately, the dynamics algorithms used to simulate these robotic systems have not kept pace with these recent hardware advancements. Fast, accurate algorithms are crucially important for both model-based controllers as well as “model-free” reinforcement learning, which in practice greatly depends on the quality of the underlying simulator. While the Articulated Body Algorithm (ABA) \cite{5} has traditionally served the role as that fast, accurate algorithm for simulating robotic systems, it is not suitable for systems with sub-mechanisms like those mentioned above.

In the late 2000s, the concept of Local Constraint Embedding (LCE) was introduced \cite{6}, which enabled the development of a forward dynamics algorithm that retained many of the favorable properties of the ABA while being general to systems with complex sub-mechanisms. However, this work has been largely neglected by the robotics community. Two contributing factors are likely (i) it did not use the ubiquitous Spatial Vector Algebra (SVA) conventions popularized by Featherstone \cite{7} and (ii) its matrix factorization-based derivation of the ABA-like algorithms is unintuitive to most roboticists. To that end, we propose an alternative development of constraint embedding-based dynamics algorithms that features:

- A unification of the concepts of LCE and SVA
- Compact, computationally efficient algorithms needed for simulating contact-rich robotic systems,
- An intuitive derivation based on the “Gauss Principle of Least Constraint” \cite{6}.

II. RELATED WORK

The ABA has formed the bedrock of modern dynamic simulators for robotic systems for many years \cite{9}–\cite{11}. The concept dates back to the 1970’s \cite{12}, but it was not popularized until a decade later when Featherstone developed the algorithm using SVA \cite{5}. In subsequent years, efforts were made to generalize the algorithm to systems with closed-loop constraints. Recognizing the importance of including the complete effects of gear ratios and the gyroscopic effects of the spinning motors, researchers working with high gear ratio manipulators developed minimally modified versions of the algorithm to handle this particular class of sub-mechanism \cite{13}, \cite{14}. This work has regained relevance in recent years as proprioceptive actuation has emerged as a popular paradigm for robots designed to make frequent contact with their environments \cite{15}. Featherstone later generalized the ABA for a larger class of sub-mechanisms by introducing the Divide-and-Conquer concept \cite{16}, \cite{17}. In the meantime, efficient non-recursive algorithms were developed that could handle arbitrary closure constraints. These algorithms use techniques such as sparse matrix factorization \cite{18} or Lagrange multiplier elimination \cite{19} to efficiently solve the large systems of linear equations arising from the equations of motion and closure constraints.

It wasn’t until the introduction of LCE that an algorithm was developed that both maintained the efficient recursive structure while being able to handle arbitrary closure constraints \cite{6}, \cite{20}. However, until recently, the concept has been largely ignored by roboticists. The recent efforts have focused on applications related to parallel kinematic actuators \cite{21}, with an emphasis on how hybrid numerical and analytical approaches to dealing with the constraints can lead to improved computational efficiency and accuracy \cite{22}, \cite{23}. We instead turn our attention to improving dynamics simulators for more general legged robots and manipulators. Specifically, we rederive an intuitive, generalized version of the ABA and we discuss the straightforward extension of our work to another state-of-the-art algorithm for computing the Operational-Space Inertia Matrix (OSIM) \cite{24}. Thus, we lay the foundation for a new dynamics engine that combines a generalized ABA for dynamics simulation of unconstrained mechanisms, with a generalized OSIM algorithm for contact simulation. Specifically, the OSIM algorithm will form the
backbone of an efficient time-stepping solver for the contact complementarity problem [25].

III. MODELING

A popular approach to modeling rigid-body systems is to describe the system in terms of its component parts via a system model [18]. A system model consists solely of bodies and joints, and can be expressed in the form of a Connectivity Graph (RB-CG) have the following properties [18]: nodes represent bodies, arcs represent joints, exactly one node represents a fixed base, and the graph is connected and undirected. Conventional recursive algorithms for rigid-body dynamics are only compatible with kinematic trees, i.e., systems whose RB-CGs have the property that there exists exactly one path between any two nodes in the graph. In other words, the recursive algorithms are only compatible for systems with no closure constraints.

LCE can be used to convert arbitrary RB-CGs to kinematic trees where each node of the tree is a collection of rigid bodies and each arc is a collection of joints [6], as is shown in Fig. [1]. Nodes containing more than one rigid body have a closure constraint associated with them that describes how the motions of each contained body are coupled. We assume that all constraints are expressed in explicit form [26]

\[
q = \gamma (y) , \quad \dot{q} = G\dot{y} , \quad \ddot{q} = \ddot{y} + g
\]

where \(q\) are the spanning tree coordinates and \(y\) are the independent coordinates.

In this paper, we refer to these collections of bodies as “body clusters” or simply “clusters”, and we refer to the collections of joints that connect pairs of body clusters as “cluster joints.” Furthermore, we refer to a connectivity graph comprised of clusters and cluster joints as a Cluster Connectivity Graph (C-CG). All C-CGs are kinematic trees. For the most part, C-CGs can be treated the same as RB-CGs. They both follow the widely popular “regular numbering” convention, wherein each node must have a higher number than its parents and arc \(i\) connects between node \(i\) and its parent [18]. However, in the case of C-CGs, both the individual rigid bodies as well as the clusters are separately numbered. For clarity, we will use indices \(i\) and \(j\) to exclusively refer to rigid bodies, and we will use indices \(k\) and \(l\) to exclusively refer to clusters. We use \(C_k\) to refer to the \(k\)th cluster in the C-CG

IV. SPATIAL VECTOR ALGEBRA FOR CLUSTERS

The difference between body clusters and conventional rigid bodies is the dimension of the spatial vectors that describe their motion and force interactions. We define \(M^0\) and \(F^0\) as vector subspaces for spatial motion vectors and spatial force vectors, respectively. Spatial motion and force vectors for any individual rigid body are elements of the subspaces \(M^0\) and \(F^0\), while the spatial motion and force vectors for a cluster are elements of the subspaces \(M^{6nb}\) and \(F^{6nb}\), where \(nb\) is the number of bodies contained by the cluster. Thus, we can consider the SVA quantities used to describe the motion and force interactions of rigid-body clusters as the concatenation of the SVA quantities of the cluster’s constituent bodies.

To that end, we define the following cluster concatenation operators: stack \((\{z_i\}_{i \in C_k})\) creates a single vector containing the vector quantities \(z\) corresponding to every rigid body in cluster \(k\), while block \((\{Z_i\}_{i \in C_k})\) creates a block diagonal matrix of the matrix quantities \(Z\) corresponding to every rigid body in cluster \(k\). Table I distinguishes the notation we use to represent SVA quantities for individual bodies versus clusters.

<table>
<thead>
<tr>
<th>SVA NOTATION FOR RIGID BODIES VS. CLUSTERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joint Coords.</td>
</tr>
<tr>
<td>(q, y)</td>
</tr>
<tr>
<td>Spatial Vectors</td>
</tr>
<tr>
<td>Spatial Matrices</td>
</tr>
</tbody>
</table>

An important nuance of SVA for clusters is the motion subspace matrix \(S\). This matrix corresponds to the constraint imposed by the cluster joint and indicates the allowable relative motion between two clusters. Thus, the spatial velocities between any cluster \(l\) and its parent cluster \(k\) satisfy

\[
v_l - v_k = S_l \dot{y}_l .
\]

The nuance comes from the fact that the motion of a body in cluster \(l\) might depend on the motion of another body in the same cluster, as is the case for bodies 2, 3, and 7 in Fig. [1]. As such, the motion subspace matrix for a cluster \(l\) is

\[
S_l = X_{I(l)} \text{block} \left( \{S_i\}_{i \in C_l} \right) G_l
\]

where \(X_{I(l)}\) is the intra-cluster transformation matrix that captures the relationship between bodies in the same cluster.

V. FORWARD DYNAMICS

The original ABA for computing forward dynamics consists of three passes: a forward pass for first order kinematics, a backward pass for updating the articulated-body inertias of each body in the tree, and a forward pass for propagating the accelerations outward from the base. In this section, we use the extension of SVA from Sec. IV and the Principle of Least Constraint (GPLC) to derive the Cluster-Based Articulated-Body Algorithm (C-ABA) for systems with closed-loop constraints. We focus on ABA since subsequent algorithms, such as the OSIM algorithm, rely on the ABA.
A. Forward Kinematics

Analogous to the original ABA, we show that the spatial velocities for every body in a cluster can be expressed in terms of the spatial velocities of bodies in the parent cluster and the position and velocity of the cluster joint. This relationship can be derived by applying the concatenation operator to the velocities of all of the bodies in an arbitrary cluster \( k \):

\[
\nu_k = \text{stack} \left( \left\{ i \mathbf{X}_{A(i)} \mathbf{v}_{A(i)} \right\}_{i \in C_k} \right) + \\
\text{stack} \left( \left\{ \sum_{j \in \xi(i)} i \mathbf{X}_j \mathbf{S}_j \mathbf{q}_j \right\}_{i \in C_k} \right)
\]

(4)

where \( A(i) \) gives the nearest supporting body of \( i \) in the parent cluster and \( \xi(i) \) gives all of the supporting bodies of body \( i \) that are in its same cluster. Successive iterations of (4) with \( k \) ranging from 1 to \( N_c \), where \( N_c \) is the number of clusters in the tree, constitute the first forward pass of the C-ABA.

B. Articulated Cluster Inertia

A body’s articulated-body inertia describes the inertia that rigid body appears to have when it is part of a rigid-body system [3]. The concept extends straightforwardly to clusters of bodies, wherein attaching a cluster \( l \) to cluster \( k \) via a cluster joint yields the articulated-cluster equation of motion

\[
f_k = \mathbf{l}_k^A a_k + \mathbf{p}_k^A,
\]

(5)

where \( \mathbf{l}_k^A \) and \( \mathbf{p}_k^A \) are the articulated-cluster inertia and bias force of the multi-cluster system, respectively. Articulated-cluster inertias have the same properties as articulated-body inertias: they are symmetric and positive-definite, they map from \( M^{6_{na}} \) to \( R^{6_{na}} \), they can be transformed between coordinate frames via \( i \mathbf{X}_k^T \mathbf{l}_k^A k \mathbf{X}_i \).

C. Calculating Articulated-Cluster Inertia via the Gauss Principle of Least Constraint

The backward pass of the C-ABA depends on a recursive relationship between the articulated-body quantities of child and parent bodies, and the second forward pass depends on a recursive relationship between the joint and spatial accelerations of the child and parent bodies. There are multiple ways to derive these recursive relationships, such as the assembly method [3] and the matrix-factorization method [27]. However, arguably the most physically intuitive derivation, especially for roboticists, is based on the GPLC.

Consider an arbitrary cluster of bodies \( C_k \) that is being attached to cluster \( C_l \) via a cluster joint. Applying the GPLC to the interaction between the bodies in these clusters involves evaluating their respective equations of motion in the following context: (i) neglecting the constraint forces at all of the joints, (ii) applying joint torques \( \tau_j \) at the actuated joints in the child cluster, and (iii) applying a test spatial force \( f_c \) to each of the bodies in the parent cluster.

The free-body diagram in Fig. 2 illustrates that the parent cluster follows the spatial equation of motion

\[
\mathbf{l}_k^A a_k = \text{stack} \left( \left\{ \mathbf{f}_i \right\}_{i \in C_k} \right) - \text{stack} \left( \sum_{j \in C_l(\mu(i))} \mathbf{\Psi}^q_j \tau_j \right) - \mathbf{p}_k^A
\]

(6)

where \( \mu(i) \) gives the children of body \( i \) and \( \mathbf{\Psi}^q_j \) is the active force subspace matrix of the \( j \)th joint. Similarly, the child cluster follows

\[
\mathbf{l}_l^A a_l = \text{stack} \left( \left\{ \mathbf{\Psi}^q_i \tau_i \right\}_{i \in C_l} \right) - \mathbf{p}_l^A
\]

(7)

The final step in preparing to apply the GPLC is to relate the accelerations of the child cluster to the acceleration of the parent cluster and the generalized coordinates of the child cluster’s joints. This relationship can be derived via a similar process as (4) and is given by

\[
a_l = i \mathbf{X}_k a_k + S_l \dot{q}_l + \dot{S}_l \ddot{q}_l.
\]

(8)

The GPLC states that the motions of the bodies in these two clusters satisfies

\[
\min_{a_k, a_l, \dot{q}_l} \frac{1}{2} (a_k^A - a_l^A)^T \mathbf{l}_k^A (a_k^A - a_l^A)
\]

\[
+ \frac{1}{2} (a_l^A - a_l)^T \mathbf{l}_l^A (a_l^A - a_l)
\]

s.t. \( a_l = i \mathbf{X}_k a_k + S_l \dot{q}_l + \dot{S}_l \ddot{q}_l \).

(9)

Using the substitutions from (6)-(8) and the first order optimality conditions for \( \ddot{q}_l \), we can derive the following recursive relationship with \( a_k \).

\[
\dot{\ddot{q}}_l = D_l^{-1} \left( u_l - U_l^T \left( i \mathbf{X}_k a_k + c_l \right) \right),
\]

(10)

where

\[
U_l = \mathbf{l}_l^A S_k, \quad D_l = S_l^T U_k, \quad u_k = \tau_k - S_k^T \mathbf{p}_k^A.
\]

(11)

This relationship is used in the second forward pass of the C-ABA to propagate the accelerations of the clusters from the base to the tips of the C-CG.

Lastly, we derive the recursive relationship for the backward pass by considering the spatial equation of motion for the two-cluster system created by joining clusters \( k \) and \( l \),

\[
f_k^A = \mathbf{l}_k^A a_k + \mathbf{p}_k^A + k \mathbf{X}_l^F \left( \mathbf{l}_l^A a_l + \mathbf{p}_l^A \right).
\]

(12)
Substituting $a_l$ and $q_l$ into (12) using (8) and (10) yields
\[ f_k^A = (l_k^A + kX_k (l_k^A - U_lD_l^{-1}U_l)^T X_k) a_k + p_k^A + kX_k \begin{bmatrix} p_k^A + l_k^A c_l + U_lD_l^{-1}u_l \end{bmatrix} . \] (13)

Noticing that (13) has the same form as (5), we have arrived at the recursive relationship for the articulated-cluster inertia and bias force, which are analogous to the original ABA relationships
\[ l_k^A = l_k + \sum_{(C_l \in \mathcal{C}_k)} l_l^A, \quad p_k^A = p_k + \sum_{(C_l \in \mathcal{C}_k)} p_l^A \] (14)
where
\[ l_l^A = l_l^A - U_lD_l^{-1}U_l^T, \]
\[ p_l^A = p_l^A + l_l^A c_l + U_lD_l^{-1} (\tau_l - S_l^T p_l^A) . \] (15)

These relationships constitute the backward pass of the ABA.

VI. EXTENSIONS

For the sake of brevity, we cannot include detailed derivations of how the proposed approach enables the Cluster-Based Extended-Force-Propagator Algorithm (C-EFPA) which is an extension of the reduced-order recursive algorithm for computing OSIM [24]. Instead, we outline the derivation qualitatively. The derivation is closely related to the derivation of the C-ABA with the distinction that the C-EFPA isolates the effects of end-effector forces that are needed in operational-space dynamics. The main feature of the EFPA is its use of extended force propagators to provide transformations of spatial forces from end-effector $k$ to rigid body $i$ as if the joints of the robot are free to move. This is contrary to a standard spatial force transforms, which assume all joints are locked. Deriving the EFPA relies on showing that analogous force propagators exist for clusters of bodies. Thus, the key step in the derivation of the EFPA is demonstrating that there exists a recursive relationship that can be used to efficiently compute the extended force propagator matrices between clusters of bodies, rather than between individual bodies.

The development of such an algorithm has significant ramifications for contact-rich dynamic simulation. Numerous contemporary dynamics engines for robotic systems utilize rigid contact models. These dynamics engines employ diverse solution methods to address the inherent complementarity problem for these models [28], [29], but they all share the common requirement of needing to compute the OSIM when solving for contact impulses. Therefore, achieving fast and accurate computations of the OSIM for robots with complex sub-mechanisms holds the potential to greatly enhance the overall efficiency of these contact dynamics solvers for a whole new class of robots.

VII. RESULTS

We present preliminary results demonstrating the computational efficiency of the C-ABA compared to two state-of-the-art algorithms for computing forward dynamics. The algorithms are applied to two robots. Robot A is a serial chain of links that are each actuated with a geared motor. Each link and its corresponding rotor form a cluster that is constrained by
\[ \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \gamma_A(y_1) = \begin{bmatrix} y_1 \\ \gamma_1 y_1 \\ \beta_{1,1} y_1 + \beta_{2,2} y_2 \\ y_2 \end{bmatrix} \] (17)
with $\gamma \in \mathbb{R}$ as a random gear ratio. Robot B is a serial chain of links, but for this robot every pair of links in the chain is actuated by a pair of geared motors. The distal link in the pair is coupled to the motions of both motors, such that each cluster is constrained by
\[ \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \gamma_B \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \beta_{1,1} y_1 + \beta_{2,2} y_2 \\ \gamma_1 y_1 \\ y_2 \end{bmatrix} \]

where
\[ \gamma_1 \gamma_2 \gamma_3 \gamma_4 \in \mathbb{R} \]

with $\gamma \in \mathbb{R}$ as random gear ratios. This coupled belt drive, while more complicated to simulate, enables lower inertia limits since it allows the rotors to be placed closer to the base [1]. The results in Fig. 3 prove that the C-ABA scales linearly with the number of degrees of freedom with the robot. The improvement relative to state-of-the-art is larger for Robot A because the constraints are “more local”, so the C-ABA more closely resembles the standard ABA.

VIII. CONCLUSIONS

In this paper, we presented a novel approach for generalizing rigid-body dynamics algorithms to handle closure constraints in complex robotic systems. We first introduced the C-CG and detailed how SVA can be extended to describe the motion and force interactions of rigid-body clusters. Next, the C-ABA was developed using the GPLC to derive recursive relationships for the articulated-cluster inertias of all of the clusters in the C-CG. We qualitatively described how the approach can extend to other algorithms relevant to contact-rich dynamic simulation. Finally, we presented preliminary results showing the efficiency of our proposed algorithm. Next steps include formal derivations of extended algorithms such as the C-EFPA and developing an open-source dynamics engine that implements these algorithms.

REFERENCES
